SIEMENS

SIMATIC

STEP 7 Basic V14 SP1

System Manual

Printout of the online help

System overview of STEP 7 and WinCC 1

What's new in TIA Portal 2

Readme 3

Installation 4

Migrating projects and programs 5

Introduction to the TIA Portal 6

Editing projects 7

Editing project data 8

Using libraries 9

Editing devices and networks 10

Programming the PLC 11

Visualizing processes 12

Using technology functions 13

Using online and diagnostics functions 14

Using Team Engineering 15

04/2017

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

⚠ DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

⚠ WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

⚠ CAUTION

indicates that minor personal injury can result if proper precautions are not taken.

NOTICE

indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

⚠ WARNING

Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Division Digital Factory
Postfach 48 48
90026 NÜRNBERG
GERMANY

℗ 03/2017 Subject to change

Copyright © Siemens AG 2017.
All rights reserved

Table of contents

1 System overview of STEP 7 and WinCC....37

1.1 Scaling of STEP 7 and WinCC in the TIA Portal....37

1.2 Options for STEP 7 and WinCC Engineering....39

1.3 Options for S7-1500 Runtime....40

1.4 Options for STEP 7 Engineering System....41

1.5 Options for WinCC Engineering and Runtime systems....42

2 What's new in TIA Portal....45

2.1 What's new in TIA Portal V14 SP1....45
2.1.1 Introduction....45
2.1.2 SIMATIC STEP 7....45
2.1.3 SIMATIC WinCC....48
2.1.4 SIMATIC STEP 7 Safety....49
2.1.5 SINAMICS Startdrive....51
2.1.6 SINUMERIK STEP 7 Toolbox....51
2.1.7 Runtime options....52

2.2 What's new in TIA Portal V14....54
2.2.1 Introduction....54
2.2.2 SIMATIC STEP 7....54
2.2.3 SIMATIC WinCC....57
2.2.4 SINAMICS Startdrive....58
2.2.5 SIMOTION SCOUT TIA....59
2.2.6 SIMOCODE ES V14....59
2.2.7 Engineering options....60
2.2.8 Runtime options....62

3 Readme....65

3.1 Notes on the TIA Portal....65
3.1.1 General notes....65
3.1.2 Notes on libraries....67
3.1.3 Notes on memory cards....68
3.1.4 Notes on the hardware configuration....69
3.1.5 Notes on instructions....69
3.1.6 Notes on using the TIA Portal in a virtual environment (private cloud)....70
3.1.7 Information on working with Multiuser Engineering....71

3.2 STEP 7 Basic....73
3.2.1 Security information....73
3.2.2 Notes on use....74
3.2.3 Editing devices and networks....75
3.2.3.1 General information on devices and networks....75
3.2.3.2 Use of modules on the S7-1200....76
3.2.3.3 CP 343-2 on SIMATIC S7 Embedded Controller EC31-RTX....77
3.2.3.4 F-CM AS-i Safety ST for ET 200SP....77

3.2.3.5 S7 routing via IE/PB Link....77
3.2.3.6 Notes on online and diagnostics....78
3.2.3.7 Network components....79
3.2.4 Programming a PLC....81
3.2.4.1 General notes on PLC programming....81
3.2.4.2 Instructions....83
3.2.4.3 Testing the user program....84
3.2.4.4 Compatibility of PLC programs from older versions....85
3.2.5 Inter Project Engineering (IPE)....94
3.2.5.1 Notes on IPE....94
3.2.6 Technological functions....94
3.2.6.1 Notes on technological functions (S7-1200)....94
3.3 WinCC Basic....97
3.3.1 Security information....97
3.3.2 News....98
3.3.3 Notes on use....99
3.3.4 Migration....100
3.3.5 Engineering System....102
3.3.5.1 Screens and Screen Objects....102
3.3.5.2 Tags and connections....106
3.3.5.3 Alarm system and alarm displays....107
3.3.5.4 System functions....108
3.3.5.5 Recipes....108
3.3.5.6 User administration....109
3.3.5.7 Communication....109
3.3.6 System-wide functions....111
3.3.7 Compiling and loading....112
3.3.8 Runtime....115
3.3.8.1 Notes on operation in Runtime....115
3.3.8.2 Notes on operation of panels in Runtime....117
3.3.9 HMI devices....118
3.3.9.1 Notes on HMI devices....118
4 Installation....123
4.1 Notes on the installation....123
4.2 System requirements for installation....125
4.2.1 Notes on licenses....125
4.2.2 Notes on the system requirements....126
4.2.3 System requirements STEP 7 Basic....127
4.2.3.1 Licensing of STEP 7....127
4.2.3.2 Handling licenses and license keys....129
4.2.3.3 Software and hardware requirements STEP 7....131
4.2.4 System requirement for WinCC Basic....135
4.2.4.1 Software and hardware requirements....135
4.2.4.2 Parallel installation....139
4.2.4.3 Licenses and Powerpacks....139
4.3 Installation log....145
4.4 Starting installation....146
4.5 Displaying the installed software....148

4.6 Modifying or updating installed products....149
4.7 Repairing installed products....151
4.8 Starting to uninstall....153
4.9 Installing updates and support packages....155
4.9.1 Checking availability of updates and support packages and installing them....155
4.9.2 Providing updates on a company-internal server....159
4.9.3 Configuring a company-internal server for updates....160
4.10 Installing and uninstalling the migration tool....164
4.10.1 System requirements....164
4.10.2 Installing the migration tool....164
4.10.3 Uninstalling the migration tool....165
4.11 Installing and uninstalling the multiuser server....166
4.11.1 Notes on the installation of the multiuser server....166
4.11.2 Installing the multiuser server....167
4.11.3 Uninstalling the multiuser server....168
5 Migrating projects and programs....171
5.1 Overview of Migration Options....171
5.2 Migrating projects to a TIA Portal project....173
5.2.1 Migration of projects with the TIA Portal....173
5.2.2 Check migration readiness of hardware components....174
5.2.3 Preparing projects with the migration tool....175
5.2.3.1 Migrating projects with the migration tool....175
5.2.3.2 Calling the migration tool....177
5.2.3.3 Creating a migration file....177
5.2.4 Migrating projects....178
5.2.5 Displaying the history of the migration....180
5.2.6 Display migration log....180
5.2.7 Migrating WinCC flexible projects....181
5.2.7.1 Basics (WinCC flexible)....181
5.2.7.2 Migrating engineering data (WinCC flexible)....186
5.2.7.3 Migrating runtime data (WinCC flexible)....197
5.2.7.4 Migrating integrated projects (WinCC flexible)....199
5.2.7.5 Reference (WinCC flexible)....203
5.2.8 Migrating integrated projects....211
5.2.8.1 Migrating an integrated project (S7-300, S7-400)....211
5.2.8.2 Post-editing integrated projects....213
5.2.8.3 Converting unspecified CPUs into specified CPUs....214
5.2.8.4 Creating an integrated HMI connection....216
5.2.8.5 Re-linking HMI tags....218
5.2.8.6 Delete non-integrated HMI connection....219
5.3 Migrating S7-1200 to firmware as of V4 (S7-1200)....221
5.3.1 Basic information on upgrading to V4 (S7-1200)....221
5.3.2 Migrating to V4 (S7-1200)....223
5.3.3 Special considerations after migrating to V4 (S7-1200)....224
6 Introduction to the TIA Portal....227
6.1 Getting started....227

6.1.1 Getting Started Documentation....227

6.2 User interface and operation....229
6.2.1 Starting, setting up, and exiting the TIA Portal....229
6.2.1.1 Starting and exiting the TIA Portal....229
6.2.1.2 Overview of the program settings....230
6.2.1.3 Overview of the settings for software updates....232
6.2.1.4 Overview of the script and text editor settings....232
6.2.1.5 Overview of the print settings....233
6.2.1.6 Overview of library settings....234
6.2.1.7 Overview of settings for searching in a project....234
6.2.1.8 Overview of the settings for cross-references....234
6.2.1.9 Overview of the settings for managing window layouts....234
6.2.1.10 Overview of settings for basic integrity check....235
6.2.1.11 Overview of the online and diagnostic settings....235
6.2.1.12 Changing the settings....236
6.2.1.13 Exporting and importing settings....236
6.2.2 Layout of the user interface....238
6.2.2.1 Views....238
6.2.2.2 Portal view....239
6.2.2.3 Project view....240
6.2.2.4 Library view....243
6.2.2.5 Project tree....243
6.2.2.6 Work area....247
6.2.2.7 Inspector window....255
6.2.2.8 Task cards....257
6.2.2.9 Reference projects....259
6.2.2.10 Details view....260
6.2.2.11 Overview window....262
6.2.2.12 User interface layout....266
6.2.3 Keyboard operation in the TIA Portal....270
6.2.3.1 Operation of the TIA Portal with the keyboard....270
6.2.3.2 Displaying an overview of all keyboard shortcuts....270
6.2.3.3 Basic functions of the TIA Portal....270
6.2.3.4 Using project-related functions....272
6.2.3.5 Arranging windows....272
6.2.3.6 Navigating through the program interface....273
6.2.3.7 Customizing editors....275
6.2.3.8 Editing objects....276
6.2.3.9 Text editing....277
6.2.3.10 Editing tables....278
6.2.3.11 Using online functions....279
6.2.3.12 Using the on-screen keyboard....280
6.2.4 Special features specific to the operating system....280
6.2.4.1 Influence of user rights....280
6.2.4.2 Expanding user rights....281

6.3 Help on the information system....283
6.3.1 General remarks on the information system....283
6.3.2 Opening the information system....288
6.3.3 Using the information system....289
6.3.4 Searching the information system....291
6.3.4.1 Basics of searching....291

6.3.4.2 Searching help topics....292
6.3.4.3 Examples of searching in the information system....293
6.3.5 Using favorites....295
6.3.6 Copying code examples into the program....296
6.3.7 Displaying more information in the Siemens Industry Online Support....297
6.3.8 Printing help topics....298
6.3.9 Configuring the display of tooltips and tooltip cascades....299
6.3.10 Assembling customized documentation....300
6.3.11 Safety Guidelines....301

6.4 Find and replace....303
6.4.1 Basics of searching....303
6.4.2 Search entire project....304
6.4.2.1 Basics for search in the project....304
6.4.2.2 Overview of the search editor....305
6.4.2.3 Search in entire project....307
6.4.2.4 Creating a new search index manually....308
6.4.2.5 Enabling and disabling "Search in project"....309
6.4.2.6 Examples for searching in the project....310
6.4.3 Searching and replacing in the editor....312
6.4.3.1 Basics for search in open editors....312
6.4.3.2 Overview of the "Find and replace" palette....314
6.4.3.3 Searching and replacing in the editor....315

6.5 Undoing and redoing actions....316
6.5.1 Basics of undoing and redoing actions....316
6.5.2 Undoing an action....317
6.5.3 Redoing an action....318

7 Editing projects....321

7.1 Creating and managing projects....321
7.1.1 The basics of projects....321
7.1.2 Creating a project....322
7.1.3 Opening projects....323
7.1.4 Saving projects....325
7.1.5 Closing projects....326
7.1.6 Removing projects from the project list....326
7.1.7 Deleting projects....327
7.1.8 Displaying properties of the project....328
7.1.9 Using logs....329
7.1.10 Security of industrial plants....330
7.1.11 Importing and exporting CAx data....331

7.2 Compatibility of projects....333
7.2.1 Compatibility between TIA Portal versions....333
7.2.2 Using projects created with optional software....334
7.2.3 Compatibility mode....334
7.2.4 Upgrading projects....335
7.2.4.1 General information on upgrading projects....335
7.2.4.2 Upgrading projects....336
7.2.4.3 Upgrading multiuser engineering projects....338
7.2.4.4 Upgrading know-how protected blocks....338
7.2.5 Compatibility with the plant configuration....339

7.3 Archiving and retrieving projects....341
7.3.1 Working with project archives....341
7.3.2 Creating a project archive....341
7.3.3 Retrieving compressed project....342

7.4 Using reference projects....344
7.4.1 Basics of reference projects....344
7.4.2 Opening and closing a reference project....344
7.4.3 Comparing reference projects....345

8 Editing project data....347

8.1 Compiling and loading project data....347
8.1.1 Compiling project data....347
8.1.1.1 General information on compiling project data....347
8.1.1.2 Compiling project data....348
8.1.2 Loading project data....348
8.1.2.1 General information on loading....348
8.1.2.2 Downloading project data to a device....350
8.1.2.3 Downloading project data to a memory card....352
8.1.2.4 Uploading project data from a device....354
8.1.2.5 Loading project data from a memory card....356

8.2 Comparing project data....358
8.2.1 Basics of project data comparison....358
8.2.2 Carrying out an online/offline comparison....360
8.2.3 Carrying out offline/offline comparisons....361
8.2.4 Using the compare editor....362
8.2.4.1 Overview of the compare editor....362
8.2.4.2 Show and hide table columns....369
8.2.4.3 Filtering the compare editor view....370
8.2.4.4 Running a detailed comparison....371
8.2.4.5 Updating the comparison results....372
8.2.4.6 Set comparison criteria....373
8.2.4.7 Synchronizing non-identical objects....374
8.2.4.8 Changing the view....376

8.3 Protecting project data....378
8.3.1 Protection concept for project data....378
8.3.2 Revoking access rights for devices....378

8.4 Printing project contents....380
8.4.1 Printing project documentation....380
8.4.1.1 Documentation settings....380
8.4.1.2 Printout of project contents....381
8.4.1.3 Changing the print settings....382
8.4.1.4 Specifying the print layout....383
8.4.1.5 Entering document information....384
8.4.1.6 Managing cover pages and frames....385
8.4.1.7 Designing cover pages and frames....388
8.4.1.8 Displaying print preview....394
8.4.1.9 Printing project data....397
8.4.2 Printing module labels....399
8.4.2.1 Print function for module labels....399
8.4.2.2 Printing labels....400

8.4.2.3 Exporting labeling data as XML....402
8.4.2.4 XML schema of the export file....403
8.4.2.5 Determining the print area offset....406
8.5 Working with multi-language projects....407
8.5.1 Project text basics....407
8.5.2 Categories of user texts....409
8.5.3 Select project languages....409
8.5.4 Setting the editing language....410
8.5.5 Translating all project texts in tabular form....410
8.5.6 Translating project texts....411
8.5.7 Translating project texts using reference texts....411
8.5.8 Exporting project texts....412
8.5.9 Importing project texts....414
8.5.10 Application examples for multilanguage projects....415
8.6 Working with text lists....417
8.6.1 Text lists....417
8.6.2 Creating user-defined text lists....418
8.6.3 Editing user-defined text lists....419
8.6.4 Editing system-defined text lists....419
8.7 Using memory cards....421
8.7.1 Basics about memory cards....421
8.7.2 Adding a user-defined card reader....421
8.7.3 Accessing memory cards....422
8.7.4 Displaying properties of memory cards....423
8.8 Using cross-references....425
8.8.1 Using cross-references....425
8.9 Providing user-defined documentation....426
8.9.1 Using user-defined documentation....426
8.9.2 Specifying settings in the TIA Portal....428
8.9.3 Specifying settings with an XML file....429
8.9.4 Creating a homepage....431
8.9.5 Conventions for the creation....432
8.9.6 Calling user-defined documentation....435
8.9.7 Displaying the call log....436
8.9.8 Creating user-defined documentation....436
9 Using libraries....439
9.1 Library basics....439
9.2 Using libraries in different TIA products....441
9.3 Using the "Libraries" task card....443
9.3.1 Overview of the "Libraries" task card....443
9.3.2 Using the element view....445
9.4 Using the library view....447
9.4.1 Overview of the library view....447
9.4.2 Opening and closing the library view....449
9.5 Using library management....451
9.5.1 Overview of the library management....451
9.5.2 Opening library management....454

9.5.3 Filtering the display of types....454
9.5.4 Displaying instances in the project....455
9.5.5 Displaying cross references of an instance....456
9.5.6 Displaying relations to other library objects....457

9.6 Using global libraries....458
9.6.1 Creating a global library....458
9.6.2 Compatibility of global libraries....459
9.6.3 Opening a global library....459
9.6.4 Upgrading global libraries....460
9.6.5 Defining library languages....461
9.6.6 Displaying properties of global libraries....462
9.6.7 Displaying logs of global libraries....463
9.6.8 Saving a global library....463
9.6.9 Closing a global library....464
9.6.10 Deleting a global library....465
9.6.11 Archiving and disabling global libraries....466
9.6.11.1 Working with archives of global libraries....466
9.6.11.2 Archiving global libraries....466
9.6.11.3 Retrieving global libraries....467
9.6.12 Using global corporate libraries....468
9.6.12.1 Basics of corporate libraries....468
9.6.12.2 Creating a configuration file for corporate libraries....470

9.7 Using master copies....472
9.7.1 Basics on master copies....472
9.7.2 Adding master copies....473
9.7.3 Filtering master copies....474
9.7.4 Using master copies....475

9.8 Using types and their versions....476
9.8.1 Basics on types....476
9.8.2 State of type versions....478
9.8.3 Displaying a released type version....479
9.8.4 Displaying properties of a type or version....480
9.8.5 Working with types in the project library....481
9.8.5.1 Adding types to the project library....481
9.8.5.2 Block requirements....482
9.8.5.3 Referencing of objects within a block....483
9.8.5.4 Duplicating types....484
9.8.5.5 Using types of the project library....485
9.8.5.6 Displaying types for an instance....487
9.8.5.7 Creating a test version of a type....487
9.8.5.8 Editing test version of a type....489
9.8.5.9 Creating an editing version of a type....489
9.8.5.10 Performing a consistency check for a version....490
9.8.5.11 Discarding versions....490
9.8.5.12 Discarding all versions within a folder....491
9.8.5.13 Release versions....491
9.8.5.14 Releasing all versions within a folder....493
9.8.5.15 Updating the project to the latest versions....494
9.8.5.16 Remove the link between instance and type....495
9.8.6 Working with types in global libraries....496
9.8.6.1 Adding types to a global library....496

9.8.6.2 Using types from a global library....497
9.8.6.3 Updating the project to the latest type versions....498
9.8.7 Assigning a version....500
9.9 Multiuser Engineering in libraries....502
9.9.1 Multiuser Engineering with library elements....502
9.9.2 Checking in types and master copies....504
9.10 Updating and upgrading types of a library....505
9.10.1 Updating library with the types of another library....505
9.10.2 Upgrading types in an existing project....506
9.10.3 Example of the updating and upgrading of types....509
9.11 Defining the project languages in the project library....514
9.12 Creating folders in a library....515
9.13 Editing library elements....516
9.14 Harmonizing names and path structure....518
9.15 Clean up library....519
9.16 Comparing library elements....520
9.17 Exporting library texts....521
9.18 Importing library texts....523
10 Editing devices and networks....525
10.1 Configuring devices and networks....525
10.1.1 Hardware and network editor....525
10.1.1.1 Overview of hardware and network editor....525
10.1.1.2 Overview of settings for hardware configuration....526
10.1.1.3 Network view....527
10.1.1.4 Device view....530
10.1.1.5 Device view: Objects in the device view....532
10.1.1.6 Device view: Area for unplugged modules....535
10.1.1.7 Topology view....536
10.1.1.8 Inspector window....539
10.1.1.9 Inspector window: Displaying UDTs....542
10.1.1.10 Hardware catalog....544
10.1.1.11 Browsing the hardware catalog....547
10.1.1.12 Enabling product support....548
10.1.1.13 Displaying product support for hardware components....549
10.1.1.14 Printing hardware and network configurations....550
10.1.1.15 Printing: Changing settings....552
10.1.1.16 Printing: Enabling page break preview....552
10.1.1.17 Keyboard operation: Navigation in the editor....552
10.1.1.18 Keyboard operation: Editing objects....554
10.1.2 Configuring devices....555
10.1.2.1 Introduction to configuring hardware....555
10.1.2.2 Adding a device to the hardware configuration....556
10.1.2.3 Adding an unspecified CPU....557
10.1.2.4 Selecting a CPU....559
10.1.2.5 Selecting fail-safe modules....560
10.1.2.6 Rack: Basics....561

10.1.2.7 Rack: General slot rules....563
10.1.2.8 Rack: Inserting a module....564
10.1.2.9 Select hardware component....566
10.1.2.10 Copying a hardware component....570
10.1.2.11 Moving a hardware component....571
10.1.2.12 Replacing a hardware component....572
10.1.2.13 Renaming hardware component....573
10.1.2.14 Deleting a hardware component....575
10.1.2.15 Editing properties and parameters....575
10.1.2.16 Input and output addresses in the address overview....577
10.1.2.17 Update module version....580
10.1.2.18 Distributed I/O in the project tree....581
10.1.2.19 Comparing devices....583
10.1.3 Configure networks....585
10.1.3.1 Networking devices....585
10.1.3.2 Communication via connections....607
10.1.3.3 Displaying and configuring topology....702
10.1.3.4 Industrial Ethernet Security....720
10.1.3.5 Secure Communication (S7-1500)....915
10.1.4 Creating configurations....958
10.1.4.1 Configuring automation systems (S7-1200)....958
10.1.4.2 Configuring PC systems....1096
10.1.4.3 Configuring AS-i devices....1146
10.1.4.4 S7-1200 CM/CP....1157
10.1.4.5 SCALANCE X, W and M....1228
10.1.4.6 Configuring PROFIBUS DP (S7-1200)....1735
10.1.4.7 Configuring PROFINET IO....1762
10.1.4.8 Bus coupling with PN/PN coupler....1812
10.1.4.9 Integrating external tools....1813
10.1.4.10 Loading a configuration....1815
10.1.5 Additional information on configurations....1826
10.1.5.1 Identification systems....1826
10.1.5.2 Distributed I/O....1907
10.1.5.3 IPv4 routing with CP 1543-1 / CM 1542-1....2020
10.1.5.4 IPv6 configuration....2020

10.2 Device and network diagnostics....2022
10.2.1 Hardware diagnostics....2022
10.2.1.1 Overview of hardware diagnostics....2022
10.2.1.2 Showing non-editable and current values of configurable module properties....2033
10.2.1.3 Showing the current values of dynamic modules properties....2040
10.2.1.4 Checking a module for defects....2043
10.2.1.5 Changing the properties of a module or the programming device/PC....2051
10.2.1.6 Diagnostics in STOP mode....2070
10.2.1.7 Online accesses in the Online and Diagnostics view....2072
10.2.1.8 Checking PROFIBUS DP subnets for faults....2075
10.2.2 Connection diagnostics....2078
10.2.2.1 Overview of connection diagnostics....2078
10.2.2.2 Displaying the connection status using icons....2079
10.2.2.3 Detailed connection diagnostics....2080

11 Programming the PLC....2085

11.1 Programming basics....2085

11.1.1 Operating system and user program....2085
11.1.1.1 Operating system....2085
11.1.1.2 User program....2085
11.1.2 Blocks in the user program....2086
11.1.2.1 Linear and structured programming....2086
11.1.2.2 Overview of the block types....2088
11.1.2.3 Organization blocks (OB)....2088
11.1.2.4 Functions (FCs)....2089
11.1.2.5 Function blocks (FB)....2090
11.1.2.6 Global data blocks (DB)....2091
11.1.2.7 Global ARRAY data blocks (DB)....2092
11.1.2.8 Example of the use of ARRAY data blocks....2092
11.1.2.9 Instance data blocks....2100
11.1.2.10 CPU data blocks....2101
11.1.2.11 Blocks with optimized access....2102
11.1.3 Block calls....2107
11.1.3.1 Principles of block calls....2107
11.1.3.2 Call hierarchy....2108
11.1.3.3 Instances....2110
11.1.3.4 Parameter transfer at block call....2122
11.1.4 Using and addressing operands....2134
11.1.4.1 Basic information about operands....2134
11.1.4.2 Keywords....2135
11.1.4.3 Using tags within the program....2140
11.1.4.4 Constants....2142
11.1.4.5 Addressing operands....2147
11.1.5 Handling program execution errors....2172
11.1.5.1 Causes of error....2172
11.1.5.2 Overview of mechanisms for error handling....2174
11.1.5.3 EN/ENO mechanism....2177
11.1.5.4 Evaluating errors with output parameter RET_VAL....2198
11.1.5.5 Use of the instructions GET_ERROR and GET_ERR_ID....2202
11.1.5.6 Example of how to handle program execution errors....2203

11.2 Programming recommendations (S7-1200, S7-1500)....2209
11.2.1 The new S7-1200/1500 CPU functions and programming recommendations at a glance (S7-1200, S7-1500)....2209
11.2.2 Symbolic addressing (S7-1200, S7-1500)....2213
11.2.3 Indirect addressing of ARRAY components (S7-1200, S7-1500)....2215
11.2.4 Using the DB_ANY data type (S7-1200, S7-1500)....2218
11.2.5 Using PLC data types (UDT) (S7-1200, S7-1500)....2229
11.2.6 Using MOVE instructions in STL (S7-1500)....2233
11.2.7 Using IEC timers and counters (S7-1200, S7-1500)....2235

11.3 Data types....2239
11.3.1 Overview of the valid data types....2239
11.3.2 Binary numbers....2243
11.3.2.1 BOOL (bit)....2243
11.3.2.2 Bit strings....2244
11.3.3 Integers....2248
11.3.3.1 SINT (8-bit integers) (S7-1200, S7-1500)....2248
11.3.3.2 USINT (8-bit integers) (S7-1200, S7-1500)....2249
11.3.3.3 INT (16-bit integers)....2250

11.3.3.4 UINT (16-bit integers) (S7-1200, S7-1500)....2251
11.3.3.5 DINT (32-bit integers)....2252
11.3.3.6 UDINT (32-bit integers) (S7-1200, S7-1500)....2253
11.3.3.7 LINT (64-bit integers) (S7-1500)....2254
11.3.3.8 ULINT (64-bit integers) (S7-1500)....2255
11.3.4 Floating-point numbers....2257
11.3.4.1 REAL....2257
11.3.4.2 LREAL (S7-1200, S7-1500)....2258
11.3.4.3 Invalid floating-point numbers....2259
11.3.5 Timers....2261
11.3.5.1 S5TIME (duration) (S7-300, S7-400)....2261
11.3.5.2 TIME (IEC time)....2262
11.3.5.3 LTIME (IEC time) (S7-1500)....2263
11.3.6 Date and time....2264
11.3.6.1 DATE....2264
11.3.6.2 TIME_OF_DAY (TOD)....2264
11.3.6.3 LTOD (LTIME_OF_DAY) (S7-1500)....2265
11.3.6.4 DATE_AND_TIME (date and time of day) 2265
11.3.6.5 LDT (DATE_AND_LTIME) (S7-1500)....2266
11.3.6.6 DTL (S7-1200, S7-1500)....2267
11.3.7 Character strings....2269
11.3.7.1 Character....2269
11.3.7.2 Character strings....2270
11.3.8 PLC data types (UDT)....2277
11.3.8.1 Basics of PLC data types (UDT)....2277
11.3.8.2 Transferring a tag of the PLC data type (UDT)....2281
11.3.8.3 Addressing PLC data types (UDT)....2281
11.3.9 STRUCT data structure (anonymous structures)....2282
11.3.9.1 Basic information on STRUCT....2282
11.3.9.2 Structure of a STRUCT tag....2284
11.3.9.3 Addressing STRUCT components....2285
11.3.9.4 Transferring tags of the STRUCT data type....2285
11.3.10 ARRAY....2286
11.3.10.1 Basic information on ARRAY....2286
11.3.10.2 Structure of an ARRAY tag....2290
11.3.10.3 Addressing ARRAY components....2292
11.3.10.4 Transferring tags of the ARRAY / ARRAY[*] data type....2294
11.3.10.5 Examples of how to use ARRAYs....2298
11.3.11 Pointer....2302
11.3.11.1 POINTER (S7-300, S7-400, S7-1500)....2302
11.3.11.2 ANY (S7-300, S7-400, S7-1500)....2305
11.3.11.3 VARIANT (S7-1200, S7-1500)....2308
11.3.12 Parameter types....2325
11.3.12.1 Parameter types....2325
11.3.13 System data types....2326
11.3.13.1 System data types....2326
11.3.14 Hardware data types....2328
11.3.14.1 Hardware data types....2328
11.3.15 Data type conversion for S7-1500: (S7-1500)....2330
11.3.15.1 Overview of data type conversion (S7-1500)....2330
11.3.15.2 Implicit conversions (S7-1500)....2333
11.3.15.3 Explicit conversions (S7-1500)....2382

11.3.16 Data type conversion for S7-1200 (S7-1200)....2465
11.3.16.1 Overview of data type conversion (S7-1200)....2465
11.3.16.2 Implicit conversions (S7-1200)....2467
11.3.16.3 Explicit conversions (S7-1200)....2487
11.3.17 Data type conversion for S7-300, S7-400 (S7-300, S7-400)....2531
11.3.17.1 Overview of data type conversions (S7-300, S7-400)....2531
11.3.17.2 Implicit conversions (S7-300, S7-400)....2533
11.3.17.3 Explicit conversions (S7-300, S7-400)....2545

11.4 Instructions (S7-1200, S7-1500)....2568
11.4.1 Example libraries (S7-1200, S7-1500)....2568
11.4.1.1 Sample Library for Instructions (S7-1200, S7-1500)....2568
11.4.1.2 Library of General Functions (S7-1200, S7-1500)....2570
11.4.2 Asynchronous instructions (S7-1200, S7-1500)....2571
11.4.2.1 Difference between synchronous and asynchronous instructions (S7-1200, S7-1500)....2571
11.4.3 Basic instructions (S7-1200, S7-1500)....2574
11.4.3.1 LAD (S7-1200, S7-1500)....2574
11.4.3.2 FBD (S7-1200, S7-1500)....2895
11.4.3.3 SCL (S7-1200, S7-1500)....3219
11.4.4 Extended instructions (S7-1200, S7-1500)....3455
11.4.4.1 Date and time-of-day (S7-1200, S7-1500)....3455
11.4.4.2 String + Char (S7-1200, S7-1500)....3490
11.4.4.3 Process image (S7-1500)....3562
11.4.4.4 Distributed I/O (S7-1200, S7-1500)....3571
11.4.4.5 PROFIenergy (S7-1200, S7-1500)....3728
11.4.4.6 Module parameter assignment (S7-1500)....3782
11.4.4.7 Interrupts (S7-1200, S7-1500)....3795
11.4.4.8 Alarms (S7-1200, S7-1500)....3882
11.4.4.9 Diagnostics (S7-1200, S7-1500)....3930
11.4.4.10 Pulse (S7-1200)....3991
11.4.4.11 Recipes and data logging (S7-1200, S7-1500)....3995
11.4.4.12 Data block functions (S7-1200, S7-1500)....4045
11.4.4.13 Addressing (S7-1200, S7-1500)....4073
11.4.5 Technology (S7-1200, S7-1500)....4094
11.4.5.1 S7-1200 Motion Control (S7-1200)....4094
11.4.5.2 High-speed counters (S7-1200)....4194
11.4.5.3 PID Control (S7-1200, S7-1500)....4199
11.4.6 Communication (S7-1200, S7-1500)....4383
11.4.6.1 S7 communication (S7-1200, S7-1500)....4383
11.4.6.2 Open User Communication (S7-1200, S7-1500)....4418
11.4.6.3 Web server (S7-1200, S7-1500)....4575
11.4.6.4 Communications processor (S7-1200, S7-1500)....4584
11.4.6.5 TeleService (S7-1200)....4676

11.5 Program editor....4684
11.5.1 Overview of the program editor....4684
11.5.2 Using the keyboard in the program editor....4688
11.5.3 Enlarging the programming window....4693
11.5.4 Setting the mnemonics....4695
11.5.5 Displaying symbolic and absolute addresses....4696
11.5.6 Find instructions....4697
11.5.7 Use instruction versions....4697
11.5.7.1 Basic information on instruction versions....4697

11.5.7.2 Upgrade project and instructions....4699
11.5.7.3 Update program in the current PLC....4699
11.5.8 Using instruction profiles....4701
11.5.8.1 Basics of instruction profiles....4701
11.5.8.2 Creating new instruction profiles....4701
11.5.8.3 Opening and editing instruction profiles....4703
11.5.8.4 Activating and deactivating instruction profiles....4704
11.5.8.5 Deleting instruction profiles....4705
11.5.9 Using autocompletion....4706
11.5.9.1 Basics of autocompletion....4706
11.5.9.2 Using autocompletion in graphic programming languages....4706
11.5.9.3 Using autocompletion in textual programming languages....4707
11.5.10 General settings for the PLC programming....4708
11.5.10.1 Overview of the general settings....4708
11.5.10.2 Changing the settings....4711
11.5.11 Eliminating syntax errors in the program....4711
11.5.11.1 Basic information on syntax errors....4711
11.5.11.2 Finding syntax errors in the program....4712
11.5.12 Changing the programming language....4713
11.5.12.1 Rules for changing the programming language....4713
11.5.12.2 Change the programming language....4713

11.6 Creating and managing blocks....4714
11.6.1 Creating blocks....4714
11.6.1.1 Block folder....4714
11.6.1.2 Creating organization blocks....4715
11.6.1.3 Creating functions and function blocks....4716
11.6.1.4 Creating data blocks....4716
11.6.1.5 Using blocks from libraries....4718
11.6.1.6 Copying and pasting blocks....4719
11.6.1.7 Entering a block title....4723
11.6.1.8 Entering a block comment....4724
11.6.2 Specifying block properties....4725
11.6.2.1 Basics of block properties....4725
11.6.2.2 Overview of block properties....4726
11.6.2.3 Block time stamps....4730
11.6.2.4 Displaying and editing block properties....4732
11.6.3 Managing blocks....4733
11.6.3.1 Opening blocks....4733
11.6.3.2 Saving blocks....4734
11.6.3.3 Closing blocks....4735
11.6.3.4 Renaming blocks....4735
11.6.3.5 Deleting blocks offline....4736
11.6.3.6 Deleting blocks online (S7-300, S7-400)....4737
11.6.3.7 Deleting CPU data blocks....4737
11.6.4 Using external source files....4738
11.6.4.1 Basics of using external source files....4738
11.6.4.2 Rules for programming external source files....4740
11.6.4.3 Saving blocks as external source files....4741
11.6.4.4 Inserting external source files....4743
11.6.4.5 Opening and editing external source files....4743
11.6.4.6 Generating blocks from external source files....4744

11.7 Protecting blocks....4746
11.7.1 Protecting blocks....4746
11.7.2 Connecting the password provider....4748
11.7.3 Show available password provider....4750
11.7.4 Recording activities of the password providers....4751
11.7.5 Define the type of password assignment....4753
11.7.6 Setting up and removing block copy protection (S7-1200, S7-1500)....4754
11.7.7 Setting up and removing block write protection....4755
11.7.8 Setting up block know-how protection....4758
11.7.9 Opening know-how protected blocks....4759
11.7.10 Printing know-how protected blocks....4760
11.7.11 Changing a password....4761
11.7.12 Removing block know-how protection....4762

11.8 Declaring the block interface....4764
11.8.1 Overview of the block interface....4764
11.8.2 Rules for declaring the block interface....4765
11.8.2.1 General rules for declaring the block interface....4765
11.8.2.2 Valid data types in the block interface (S7-300, S7-400)....4767
11.8.2.3 Valid data types in the block interface (S7-1200, S7-1500)....4768
11.8.2.4 Setting the retentivity of local tags....4769
11.8.3 Declaring the block interface....4771
11.8.3.1 Layout of the block interface....4771
11.8.3.2 Declaring local tags and constants in the block interface....4772
11.8.3.3 Declaring a local tag in the program editor....4773
11.8.3.4 Declaring tags of the ARRAY data type....4775
11.8.3.5 Declaring tags of STRUCT data type....4777
11.8.3.6 Declare variables of the STRING and WSTRING data types....4778
11.8.3.7 Declaring tags based on a PLC data type....4778
11.8.3.8 Declaring higher-level tags....4779
11.8.3.9 Declaring multi-instances....4780
11.8.3.10 Declaring parameter instances....4781
11.8.3.11 Hiding parameters during the block call....4782
11.8.3.12 Declaring predefined actual parameters....4783
11.8.3.13 Editing the properties of local tags and constants....4784
11.8.3.14 Edit table of the block interface....4788
11.8.4 Updating the block interface....4791
11.8.5 Extending the block interface (S7-1200, S7-1500)....4793

11.9 Programming data blocks....4795
11.9.1 Basic principles for programming of data blocks....4795
11.9.2 Structure of the declaration table for data blocks....4797
11.9.3 Creating data blocks....4798
11.9.4 Updating data blocks....4800
11.9.5 Extending data blocks (S7-1200, S7-1500)....4801
11.9.6 Creating a data structure for global data blocks....4802
11.9.6.1 Declaring tags of elementary data type....4802
11.9.6.2 Declaring tags of the ARRAY data type....4803
11.9.6.3 Declaring tags of STRUCT data type....4804
11.9.6.4 Declaring tags of the STRING data type....4806
11.9.6.5 Declaring tags based on a PLC data type....4806
11.9.7 Defining start values....4807
11.9.7.1 Basic information on start values....4807

11.9.7.2 Defining start values....4808
11.9.8 Setting retentivity....4809
11.9.8.1 Retentivity of tags in data blocks....4809
11.9.8.2 Setting retentivity in an instance data block....4809
11.9.8.3 Setting retentivity in a global data block....4810
11.9.9 Editing the properties of tags in data blocks....4811
11.9.9.1 Properties of the tags in data blocks....4811
11.9.9.2 Changing the properties of tags in instance data blocks....4814
11.9.9.3 Changing the properties of tags in global data blocks....4816
11.9.10 Editing the declaration table for data blocks....4817
11.9.10.1 Inserting table rows....4817
11.9.10.2 Inserting table rows....4817
11.9.10.3 Deleting tags....4818
11.9.10.4 Automatically filling in successive cells....4818
11.9.10.5 Show and hide table columns....4819
11.9.10.6 Editing tags with external editors....4819
11.9.11 Online and diagnostic functions for data blocks....4820
11.9.11.1 Editing data blocks in online mode....4820
11.9.11.2 Overview of the online and diagnostic functions in data blocks....4822
11.9.11.3 Monitoring tags....4823
11.9.11.4 Modifying tags....4824
11.9.11.5 Loading start values as actual values....4825
11.9.11.6 Creating a snapshot of the actual values....4826
11.9.11.7 Loading snapshots as actual values....4828
11.9.11.8 Copying the snapshot to the start values....4830
11.9.11.9 Using setpoints during commissioning....4831

11.10 Declaring PLC tags....4837
11.10.1 Overview of PLC tag tables....4837
11.10.2 Structure of the PLC tag tables....4838
11.10.3 Rules for PLC tags....4840
11.10.3.1 Valid names of PLC tags....4840
11.10.3.2 Permissible addresses and data types of PLC tags....4840
11.10.4 Creating and managing PLC tag tables....4843
11.10.4.1 Creating a PLC tag table....4843
11.10.4.2 Grouping PLC tag tables....4844
11.10.4.3 Opening the PLC tag table....4845
11.10.5 Declaring PLC tags....4846
11.10.5.1 Entering a PLC tag declaration....4846
11.10.5.2 Setting the retentivity of PLC tags....4848
11.10.6 Grouping PLC tags for inputs and outputs in structures....4849
11.10.6.1 Other useful information regarding structured PLC tags....4849
11.10.6.2 Creating structured PLC tags....4851
11.10.7 Declaring global constants....4852
11.10.7.1 Rules for global user constants....4852
11.10.7.2 Rules for global system constants....4853
11.10.7.3 Declaring global constants....4854
11.10.8 Editing properties....4855
11.10.8.1 Editing the properties of PLC tags....4855
11.10.8.2 Editing properties of global constants....4857
11.10.9 Monitoring of PLC tags....4858
11.10.9.1 Monitoring of PLC tags....4858
11.10.10 Editing PLC tag tables....4859

11.10.10.1 Inserting a table row in the PLC tag table....4859
11.10.10.2 Copying entries in the PLC tag table....4859
11.10.10.3 Deleting entries in the PLC tag table....4860
11.10.10.4 Sorting rows in the PLC tag tables....4860
11.10.10.5 Automatically filling in cells in the PLC tag table....4861
11.10.10.6 Show and hide table columns....4861
11.10.10.7 Exporting and importing PLC tags....4862

11.11 Declaring PLC data types (UDT) 4867
11.11.1 Structure of the declaration table for PLC data types....4867
11.11.2 Creating PLC data types....4868
11.11.3 Delete PLC data types....4869
11.11.4 Renumbering PLC data types....4869
11.11.5 Programming the structure of PLC data types....4870
11.11.5.1 Declaring tags of elementary data type....4870
11.11.5.2 Declaring tags of the ARRAY data type....4871
11.11.5.3 Declaring tags of STRUCT data type 4872
11.11.5.4 Declaring tags based on a different PLC data type....4873
11.11.6 Editing tag properties in PLC data types....4874
11.11.6.1 Properties of tags in PLC data types....4874
11.11.6.2 Changing the properties of tags in PLC data types....4875
11.11.7 Editing the declaration table for PLC data types....4875
11.11.7.1 Inserting table rows....4875
11.11.7.2 Inserting table rows....4876
11.11.7.3 Deleting tags....4876
11.11.7.4 Automatically filling in successive cells....4876
11.11.7.5 Show and hide table columns....4877

11.12 Creating LAD programs....4878
11.12.1 Basic information on LAD....4878
11.12.1.1 LAD programming language....4878
11.12.1.2 Overview of the LAD elements....4878
11.12.2 Settings for LAD....4881
11.12.2.1 Overview of the settings for LAD....4881
11.12.2.2 Changing the settings....4881
11.12.3 Working with networks....4882
11.12.3.1 Using networks....4882
11.12.3.2 Inserting networks....4883
11.12.3.3 Selecting networks....4884
11.12.3.4 Copying and pasting networks....4885
11.12.3.5 Deleting networks....4885
11.12.3.6 Expanding and collapsing networks....4886
11.12.3.7 Inserting network title....4887
11.12.3.8 Entering a network comment....4888
11.12.3.9 Navigating networks....4889
11.12.4 Inserting LAD elements....4890
11.12.4.1 Rules for the use of LAD elements....4890
11.12.4.2 Prohibited interconnections in LAD....4892
11.12.4.3 Inserting LAD elements using the "Instructions" task card....4893
11.12.4.4 Inserting LAD elements using an empty box....4893
11.12.4.5 Selecting the data type of a LAD element....4894
11.12.4.6 Using favorites in LAD....4897
11.12.4.7 Insert block calls in LAD....4899

11.12.4.8 Inserting complex LAD instructions....4903
11.12.4.9 Using free-form comments....4904
11.12.5 Editing LAD elements....4907
11.12.5.1 Selecting LAD elements....4907
11.12.5.2 Copying LAD elements....4908
11.12.5.3 Cutting LAD elements....4908
11.12.5.4 Pasting an LAD element from the clipboard....4909
11.12.5.5 Replacing LAD elements....4909
11.12.5.6 Inserting additional inputs and outputs in LAD elements (S7-1200, S7-1500)....4910
11.12.5.7 Removing inputs and outputs (S7-1200, S7-1500)....4911
11.12.5.8 Deleting LAD elements....4912
11.12.6 Inserting operands into LAD instructions....4913
11.12.6.1 Inserting operands....4913
11.12.6.2 Wiring hidden parameters....4914
11.12.6.3 Displaying or hiding tag information....4915
11.12.7 Branches in LAD....4916
11.12.7.1 Basic information on branches in LAD....4916
11.12.7.2 Rules for branches in LAD....4917
11.12.7.3 Inserting branches into the LAD network....4918
11.12.7.4 Closing branches in the LAD network....4918
11.12.7.5 Deleting branches in LAD networks....4919
11.12.8 Crossings in LAD....4919
11.12.8.1 Basic information on crossings in LAD....4919
11.12.8.2 Inserting crossings....4920
11.12.8.3 Rearranging crossings....4920
11.12.8.4 Deleting crossings....4921
11.12.9 Rungs in LAD....4922
11.12.9.1 Basic information on rungs in LAD....4922
11.12.9.2 Insert rung....4923
11.12.9.3 Deleting a rung....4923
11.12.10 LAD programming examples....4924
11.12.10.1 Example of controlling a conveyor belt....4924
11.12.10.2 Example of detecting the direction of a conveyor belt....4926
11.12.10.3 Example of detecting the fill level of a storage area....4928
11.12.10.4 Example of calculating an equation....4931
11.12.10.5 Example of controlling room temperature....4933

11.13 Creating FBD programs....4937
11.13.1 Basic information on FBD....4937
11.13.1.1 FBD programming language....4937
11.13.1.2 Overview of the FBD elements....4937
11.13.2 Settings for FBD....4939
11.13.2.1 Overview of the settings for FBD....4939
11.13.2.2 Changing the settings....4940
11.13.3 Working with networks....4940
11.13.3.1 Using networks....4940
11.13.3.2 Inserting networks....4941
11.13.3.3 Selecting networks....4942
11.13.3.4 Copying and pasting networks....4943
11.13.3.5 Deleting networks....4943
11.13.3.6 Expanding and collapsing networks....4944
11.13.3.7 Inserting network title....4944
11.13.3.8 Entering a network comment....4946

11.13.3.9 Navigating networks....4947
11.13.4 Inserting FBD elements....4948
11.13.4.1 Rules for the use of FBD elements....4948
11.13.4.2 Inserting FBD elements using the "Instructions" task card....4949
11.13.4.3 Inserting FBD elements using an empty box....4950
11.13.4.4 Selecting the data type of an FBD element....4951
11.13.4.5 Using favorites in FBD....4954
11.13.4.6 Inserting block calls in FBD....4956
11.13.4.7 Inserting complex FBD instructions....4960
11.13.4.8 Using free-form comments....4961
11.13.5 Editing FBD elements....4964
11.13.5.1 Selecting FBD elements....4964
11.13.5.2 Copying FBD elements....4964
11.13.5.3 Cutting FBD elements....4965
11.13.5.4 Pasting an FBD element from the clipboard....4966
11.13.5.5 Replacing FBD elements....4966
11.13.5.6 Adding additional inputs and outputs to FBD elements (S7-1200, S7-1500)....4967
11.13.5.7 Removing instruction inputs and outputs (S7-1200, S7-1500)....4968
11.13.5.8 Deleting FBD elements....4969
11.13.6 Inserting operands in FBD instructions....4970
11.13.6.1 Inserting operands....4970
11.13.6.2 Wiring hidden parameters....4971
11.13.6.3 Displaying or hiding tag information....4972
11.13.7 Branches in FBD....4973
11.13.7.1 Basic information on branches in FBD....4973
11.13.7.2 Rules for branches in FBD....4974
11.13.7.3 Inserting branches in FBD networks....4974
11.13.7.4 Deleting branches in FBD networks....4975
11.13.8 Logic paths in FBD....4976
11.13.8.1 Basic information on logic paths in FBD....4976
11.13.8.2 Inserting a logic path....4977
11.13.8.3 Deleting a logic operation path....4977
11.13.9 FBD programming examples....4978
11.13.9.1 Example of controlling a conveyor belt....4978
11.13.9.2 Example of detecting the direction of a conveyor belt....4980
11.13.9.3 Example of detecting the fill level of a storage area....4982
11.13.9.4 Example of calculating an equation....4986
11.13.9.5 Example of controlling room temperature....4988
11.14 Creating SCL programs....4991
11.14.1 Basics of SCL....4991
11.14.1.1 Programming language SCL....4991
11.14.1.2 Expressions....4992
11.14.1.3 Arithmetic expressions....4993
11.14.1.4 Relational expressions....4995
11.14.1.5 Logical expressions....5000
11.14.1.6 Operators and operator precedence....5001
11.14.1.7 Value assignments....5002
11.14.1.8 Calculating with floating-point numbers (REAL and LREAL) in SCL....5005
11.14.1.9 Calculating with constants in SCL....5010
11.14.2 Settings for SCL....5013
11.14.2.1 Overview of the settings for SCL....5013
11.14.2.2 Changing the settings....5014

11.14.3 The programming window of SCL....5015
11.14.3.1 Overview of the programming window....5015
11.14.3.2 Customizing the programming window....5016
11.14.3.3 Formatting SCL code....5017
11.14.3.4 Expanding and collapsing sections of code....5018
11.14.3.5 Navigate to definitions....5019
11.14.3.6 Using bookmarks....5020
11.14.4 Entering SCL instructions....5023
11.14.4.1 Rules for SCL instructions....5023
11.14.4.2 Entering SCL instructions manually....5024
11.14.4.3 Inserting SCL instructions using the "Instructions" task card....5024
11.14.4.4 Surround program code with structure elements....5026
11.14.4.5 Working with regions....5026
11.14.4.6 Defining the data type of an SCL instruction....5034
11.14.4.7 Displaying or hiding tag information....5036
11.14.4.8 Using Favorites in SCL....5037
11.14.4.9 Insert block calls in SCL....5039
11.14.4.10 Expanding and reducing the parameter list....5048
11.14.4.11 Insert pragmas (S7-1200, S7-1500)....5049
11.14.4.12 Inserting comments....5050
11.14.5 Editing SCL instructions....5051
11.14.5.1 Selecting instructions....5051
11.14.5.2 Copying, cutting and pasting instructions....5052
11.14.5.3 Deleting instructions....5052
11.14.6 SCL programming examples....5053
11.14.6.1 Example of controlling a conveyor belt....5053
11.14.6.2 Example of detecting the direction of a conveyor belt....5055
11.14.6.3 Example of detecting the fill level of a storage area....5057

11.15 Compiling and loading blocks....5061
11.15.1 Compiling blocks....5061
11.15.1.1 Basic information on compiling blocks....5061
11.15.1.2 Compiling blocks in the project tree....5063
11.15.1.3 Compiling blocks in the program editor....5064
11.15.1.4 Correcting compilation errors....5065
11.15.2 Downloading blocks for S7-1200/1500 (S7-1200, S7-1500)....5066
11.15.2.1 Introduction to downloading blocks (S7-1200, S7-1500)....5066
11.15.2.2 Downloading blocks in the "RUN" operating mode to the device (S7-1200, S7-1500)....5070
11.15.2.3 Downloading blocks from program editor to device (S7-1200, S7-1500)....5074
11.15.2.4 Downloading blocks from the project tree to the device (S7-1200, S7-1500)....5076
11.15.2.5 Uploading blocks from device (S7-1200, S7-1500)....5078
11.15.2.6 Downloading blocks to a memory card (S7-1200, S7-1500)....5079
11.15.2.7 Uploading blocks from a memory card (S7-1200, S7-1500)....5081
11.15.2.8 Switching off the sequencer prior to loading a GRAPH DB (S7-1500)....5081

11.16 Comparing PLC programs....5083
11.16.1 Basic information on comparing PLC programs....5083
11.16.1.1 Introduction to comparing PLC programs....5083
11.16.1.2 Comparing blocks and PLC data types....5085
11.16.1.3 Comparing PLC tags....5086
11.16.1.4 Comparison of PLC programs based on checksums....5086
11.16.2 Comparing blocks....5088
11.16.2.1 Comparing blocks in the compare editor....5088

11.16.2.2 Performing detailed block comparisons....5089
11.16.3 Comparing PLC tags....5115
11.16.4 Comparing PLC data types....5117

11.17 Displaying program information....5120
11.17.1 Overview of available program information....5120
11.17.2 Displaying an assignment list....5121
11.17.2.1 Introduction to the assignment list....5121
11.17.2.2 Layout of the assignment list....5122
11.17.2.3 Symbols in the assignment list....5123
11.17.2.4 Displaying an assignment list....5124
11.17.2.5 Setting the view options for the assignment list....5124
11.17.2.6 Filter options in the assignment list....5125
11.17.2.7 Defining filters for assignment list....5126
11.17.2.8 Filtering an assignment list....5127
11.17.2.9 Defining retentive memory areas for bit memories....5128
11.17.2.10 Enabling the display of retentive bit memories....5129
11.17.3 Displaying the call structure....5129
11.17.3.1 Introduction to the call structure....5129
11.17.3.2 Symbols in the call structure....5131
11.17.3.3 Layout of the call structure....5132
11.17.3.4 Displaying the call structure....5133
11.17.3.5 Setting the view options for the call structure....5133
11.17.3.6 Introducing the consistency check in the call structure....5134
11.17.3.7 Checking block consistency in the call structure....5135
11.17.4 Displaying the dependency structure....5135
11.17.4.1 Introduction to the dependency structure....5135
11.17.4.2 Layout of the dependency structure....5137
11.17.4.3 Symbols in the dependency structure....5137
11.17.4.4 Displaying the dependency structure....5138
11.17.4.5 Setting the view options for the dependency structure....5139
11.17.4.6 Introducing the consistency check in the dependency structure....5139
11.17.4.7 Checking block consistency in the dependency structure....5140
11.17.5 Displaying CPU resources....5141
11.17.5.1 Introducing resources....5141
11.17.5.2 Layout of the "Resources" tab....5143
11.17.5.3 Displaying resources....5144
11.17.5.4 Selecting the maximum load memory available....5145

11.18 Displaying cross-references....5146
11.18.1 General information about cross references....5146
11.18.2 Structure of the cross-reference list....5147
11.18.3 Settings in the cross-reference list....5149
11.18.4 Displaying the cross-reference list....5152
11.18.5 Sorting the cross-reference list....5153
11.18.6 Filter options for cross-references....5155
11.18.6.1 Introduction to filtering of cross-references....5155
11.18.6.2 Creating user-defined filters for cross-references....5157
11.18.6.3 Editing and deleting user-defined filters....5161
11.18.6.4 Exporting and importing user-defined filters....5163
11.18.6.5 Setting a filter as the default setting....5165
11.18.6.6 Duplicating a filter....5166
11.18.6.7 Filtering the cross-reference list....5166

11.18.7 Printing the cross-reference list....5167
11.18.8 Adding a new object to the cross-reference list....5168
11.18.9 Show overlapping accesses in the cross-reference list....5169
11.18.10 Restoring cross-references after project upgrade....5171
11.18.11 Displaying and editing cross-references in the Inspector window....5172

11.19 Testing the user program....5175
11.19.1 Basics of testing the user program....5175
11.19.2 Testing with program status....5176
11.19.2.1 Introduction to testing with program status....5176
11.19.2.2 Setting the call environment....5178
11.19.2.3 Switching test with program status on/off....5180
11.19.2.4 Monitoring loops....5182
11.19.2.5 Monitoring structures....5184
11.19.2.6 Editing blocks during the program test....5185
11.19.2.7 Modifying tags during monitoring....5186
11.19.2.8 Switching display formats in the program status....5187
11.19.2.9 Examples of program status display....5188
11.19.3 Testing with the watch table....5196
11.19.3.1 Introduction to testing with the watch table....5196
11.19.3.2 Layout of the watch table....5197
11.19.3.3 Basic mode and expanded mode in the watch table....5198
11.19.3.4 Icons in the watch table....5199
11.19.3.5 Creating and editing watch tables....5200
11.19.3.6 Entering tags in the watch table....5202
11.19.3.7 Monitoring tags in the watch table....5211
11.19.3.8 Modifying tags in the watch table....5216
11.19.4 Testing with the force table....5224
11.19.4.1 Introduction for testing with the force table....5224
11.19.4.2 Safety precautions when forcing tags....5225
11.19.4.3 Layout of the force table....5226
11.19.4.4 Basic mode and expanded mode in the force table....5227
11.19.4.5 Icons in the force table....5228
11.19.4.6 Open and edit force table....5229
11.19.4.7 Entering tags in the force table....5230
11.19.4.8 Monitoring tags in the force table....5238
11.19.4.9 Forcing tags in the force table....5241
11.19.4.10 Stop forcing tags....5249

12 Visualizing processes....5253

12.1 Creating screens....5253
12.1.1 Basics....5253
12.1.1.1 Elements and basic settings....5253
12.1.1.2 Working with screens....5259
12.1.1.3 Working with templates....5263
12.1.1.4 Working with styles and style sheets....5268
12.1.2 Working with objects and object groups....5283
12.1.2.1 Working with objects....5283
12.1.2.2 Working with object groups....5317
12.1.2.3 Configuring the keyboard access....5323
12.1.2.4 Examples....5326
12.1.3 Working with text lists and graphics lists....5329
12.1.3.1 Working with text lists....5329

12.1.3.2 Working with graphics lists....5337
12.1.4 Dynamizing screens....5347
12.1.4.1 Basics on dynamizing screens....5347
12.1.4.2 Dynamization with animations....5349
12.1.4.3 Dynamizing with system functions....5357
12.1.5 Working with function keys....5359
12.1.5.1 Working with function keys....5359
12.1.5.2 Assigning function keys globally....5361
12.1.5.3 Local assignment of function keys....5363
12.1.5.4 Assigning a function to a function key....5364
12.1.5.5 Assigning operator authorization for a function key....5366
12.1.5.6 Assigning a graphic to a function key....5368
12.1.5.7 Configuring LED tags....5370
12.1.5.8 Example: Using function keys for screen navigation....5372
12.1.6 Working with layers....5374
12.1.6.1 Basics on working with layers....5374
12.1.6.2 Moving objects on layers....5375
12.1.6.3 Setting the active layer....5376
12.1.6.4 Showing and hiding layers (Basic Panels, Panels, Comfort Panels, RT Advanced)....5377
12.1.6.5 Renaming layers....5378
12.1.7 Configuring screen navigation....5380
12.1.7.1 Basics on screen navigation (Basic Panels, Panels, Comfort Panels, RT Advanced)....5380
12.1.7.2 Assigning screen change to button (Panels, Comfort Panels, RT Advanced, RT Professional)....5380
12.1.7.3 Assigning screen change to function key (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5382
12.1.8 Display and operating elements....5382
12.1.8.1 Device dependency of the objects....5382
12.1.8.2 Objects....5383

12.2 Working with Tags (Basic Panels)....5421
12.2.1 Basics (Basic Panels)....5421
12.2.1.1 Basics of tags (Basic Panels)....5421
12.2.1.2 Overview of HMI tag tables (Basic Panels)....5422
12.2.1.3 External tags (Basic Panels)....5423
12.2.1.4 Addressing external tags (Basic Panels)....5425
12.2.1.5 Internal tags (Basic Panels)....5428
12.2.1.6 User-defined PLC data types (UDT) (Basic Panels)....5429
12.2.2 Working with tags (Basic Panels)....5430
12.2.2.1 Creating tags (Basic Panels)....5430
12.2.2.2 Editing tags (Basic Panels)....5435
12.2.2.3 Configuring tags (Basic Panels)....5443
12.2.3 Working with arrays (Basic Panels)....5463
12.2.3.1 Basics on arrays (Basic Panels)....5463
12.2.3.2 Creating array tags (Basic Panels)....5466
12.2.3.3 Examples of arrays (Basic Panels)....5467
12.2.4 Working with cycles (Basic Panels)....5468
12.2.4.1 Cycle basics (Basic Panels)....5468
12.2.5 Archive tags (Basic Panels)....5469
12.2.5.1 Basic principles for data logging....5469
12.2.5.2 Working with data logs (Basic Panels)....5470
12.2.6 Displaying tags (Basic Panels)....5480
12.2.6.1 Outputting tag values in screens (Basic Panels)....5480

12.2.6.2 Configuring trend displays for values from the PLC (Basic Panels)....5480

12.3 Working with alarms....5483
12.3.1 Basics....5483
12.3.1.1 Alarm system in WinCC (Basic Panels)....5483
12.3.1.2 Alarm types....5484
12.3.1.3 Alarm states (Basic Panels)....5486
12.3.1.4 Alarm classes....5487
12.3.1.5 Acknowledgment....5489
12.3.1.6 Alarm groups (Basic Panels, Panels, Comfort Panels, RT Advanced)....5491
12.3.1.7 Alarm number (Basic Panels, Panels, Comfort Panels, RT Advanced)....5492
12.3.2 Working with alarms....5492
12.3.2.1 Alarm components and properties (Basic Panels)....5492
12.3.2.2 Configuring alarms....5493
12.3.2.3 Configuring alarm output....5507
12.3.2.4 Acknowledging alarms....5512
12.3.3 Logging Alarms....5515
12.3.3.1 Basics on alarm logging....5515
12.3.3.2 Creating an alarm log....5517
12.3.3.3 Logging alarms....5519
12.3.3.4 Configuring an alarm view for logged alarms (Advanced)....5520
12.3.3.5 Configuring alarm buffer overflow (Advanced) (Basic Panels, Panels, Comfort Panels, RT Advanced)....5521
12.3.3.6 Log response to language switching in runtime....5521
12.3.3.7 Managing logging behavior when Runtime starts....5522
12.3.3.8 Controlling the Logging in relation to the Fill Level....5523
12.3.4 Operating alarms in Runtime....5525
12.3.4.1 Alarms in Runtime (Basic Panels)....5525
12.3.4.2 Simple alarm view, simple alarm window in runtime (Basic Panels)....5527
12.3.4.3 Alarm window, alarm view in Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced)....5528
12.3.4.4 Alarm indicator in Runtime (Basic Panels)....5530
12.3.4.5 Acknowledging alarms (Basic Panels)....5531
12.3.5 Reference....5532
12.3.5.1 System functions for alarms (Basic Panels)....5532
12.3.5.2 System events....5532

12.4 Working with logs....5561
12.4.1 Log basics....5561
12.4.2 Properties of logs....5561
12.4.3 Storage locations of logs (Basic Panels)....5563

12.5 Working with recipes....5565
12.5.1 Basics....5565
12.5.1.1 Definition and applications....5565
12.5.1.2 Examples for using recipes....5566
12.5.1.3 Structure of recipes....5566
12.5.1.4 Displaying recipes....5568
12.5.1.5 "Recipes" editor....5568
12.5.1.6 Flow of data for recipes....5571
12.5.1.7 Synchronization of recipe data records with the PLC....5572
12.5.2 Displaying and editing recipes in Runtime....5573
12.5.2.1 Simple recipe view....5573
12.5.2.2 Configuration options of the simple recipe view....5574

12.5.2.3 Advanced recipe view (as of V13) (Basic Panels, Panels, Comfort Panels, RT Advanced)..5578
12.5.2.4 Configuration options of the advanced recipe view (V13 or higher) (Basic Panels, Panels, Comfort Panels, RT Advanced)....5579
12.5.2.5 Behavior of the recipe view in Runtime....5581
12.5.2.6 Exporting and importing recipe data records (Panels, Comfort Panels, RT Advanced). ...5582
12.5.3 Configuring recipes....5583
12.5.3.1 Creating and Editing Recipes....5583
12.5.3.2 Configuring the display of recipes....5591
12.5.4 Using recipes in Runtime....5595
12.5.4.1 Using the simple recipe view....5595
12.5.4.2 Using the advanced recipe view....5602
12.5.5 Example....5609
12.5.5.1 Example of creating a recipe (Basic Panels, Panels, Comfort Panels, RT Advanced).....5609
12.6 Configuring user administration....5612
12.6.1 Basics....5612
12.6.1.1 Field of application of the user administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5612
12.6.1.2 Form of the user administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5612
12.6.1.3 Users (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional).....5613
12.6.1.4 Users work area (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)..5615
12.6.1.5 User groups (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional). ...5615
12.6.1.6 User groups work area (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5616
12.6.1.7 Settings for the user administration (Basic Panels)....5617
12.6.1.8 Target groups in the user administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5619
12.6.2 Setting up the user administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5620
12.6.2.1 Administering users for Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5620
12.6.2.2 Managing users in Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5627
12.6.2.3 Configuring access protection (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5636
12.6.3 Reference....5637
12.6.3.1 Objects with access protection (Basic Panels)....5637
12.6.3.2 Default user groups and authorizations (Basic Panels, Panels, Comfort Panels, RT Advanced)....5638
12.6.4 Examples....5638
12.6.4.1 Example: Configuring a button with logon dialog box (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5638
12.6.4.2 Example: Logging the logon and logoff events (Basic Panels, Panels, Comfort Panels, RT Advanced)....5639
12.6.4.3 Example: Structure of user management (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5640
12.6.4.4 Example: Creating and configuring authorizations (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5642
12.6.4.5 Example: Configuring a button with access protection (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5642
12.6.4.6 Example: Creating user groups and assigning authorizations: (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5644

12.6.4.7 Example: Creating users and assigning them to a user group (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5646
12.7 Working with system functions....5649
12.7.1 Basics....5649
12.7.1.1 System functions (Basic Panels)....5649
12.7.1.2 Use of system functions (Basic Panels)....5650
12.7.2 Working with function lists....5651
12.7.2.1 Basic of the functions list (Basic Panels)....5651
12.7.2.2 Properties of a function list (Basic Panels)....5652
12.7.2.3 Configuring a function list (Basic Panels)....5652
12.7.2.4 Editing a function list (Basic Panels)....5654
12.7.2.5 Executing a function list in Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5655
12.7.3 Example....5655
12.7.3.1 Changing the operating mode on the HMI device with the current display (Basic Panels, Panels, Comfort Panels, RT Advanced)....5655
12.7.4 Reference....5659
12.7.4.1 Function list....5659
12.7.4.2 Events....5763
12.8 Planning tasks....5780
12.8.1 Basics....5780
12.8.1.1 Field of application of the Scheduler (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5780
12.8.1.2 Working with tasks and triggers (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5781
12.8.1.3 Work area of the "Scheduler" editor (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5781
12.8.1.4 Function list (Basic Panels)....5783
12.8.1.5 Triggers (Basic Panels)....5783
12.8.1.6 Planning tasks with event triggers (Basic Panels, Panels, Comfort Panels, RT Advanced)....5784
12.8.2 Examples....5786
12.8.2.1 Example: Update user following change of user (Basic Panels, Panels, Comfort Panels, RT Advanced)....5786
12.9 Using diagnostics functions....5788
12.9.1 Configuring system diagnostics....5788
12.9.1.1 System diagnostics basics (Basic Panels)....5788
12.9.1.2 System diagnostics views (Basic Panels)....5788
12.9.1.3 Configuring the system diagnostic view (Basic Panels)....5791
12.10 Communicating with controllers....5792
12.10.1 WinCC Basic V14 SP1 - Communication....5792
12.11 Using global functions....5793
12.11.1 Working with libraries....5793
12.11.1.1 Basics on libraries....5793
12.11.1.2 Overview of the library view....5795
12.11.1.3 Master copies and types....5797
12.11.1.4 Libraries in WinCC....5797
12.11.1.5 Managing libraries....5800
12.11.1.6 Managing objects in a library....5810
12.11.1.7 Using types and their versions....5815
12.11.2 Importing and exporting project data....5819

12.11.2.1 Importing and exporting project data....5819
12.11.2.2 Importing and exporting recipes....5820
12.11.2.3 Importing and exporting alarms....5824
12.11.2.4 Importing and exporting tags....5831
12.11.2.5 Importing and exporting text lists....5837
12.11.2.6 Importing and exporting project texts....5841
12.11.3 Using cross-references....5844
12.11.3.1 General information about cross references (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5844
12.11.3.2 Special considerations when working with the cross-reference list....5844
12.11.3.3 Rewiring tags in the screens (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5845
12.11.3.4 Textual cross-references: Overview (Basic Panels)....5848
12.11.3.5 Invalid cross-references (Basic Panels)....5849
12.11.4 Managing languages....5851
12.11.4.1 Languages in WinCC....5851
12.11.4.2 Language settings in the operating system....5852
12.11.4.3 Operating system settings for Asian languages....5852
12.11.4.4 Setting project languages....5853
12.11.4.5 Creating one project in multiple languages....5856
12.11.4.6 Using language-specific graphics....5862
12.11.4.7 Languages in Runtime....5865
12.11.4.8 Example of multilingual configuration....5873
12.11.5 Replacing devices....5875
12.11.5.1 Basics (Basic Panels, Panels, Comfort Panels, RT Advanced)....5875
12.11.5.2 Device-specific functions....5876
12.11.5.3 Adjusting screens to the new device....5879
12.11.6 Copying between devices and editors....5884
12.11.6.1 Basics....5884
12.11.6.2 Copying and pasting....5886
12.11.6.3 Copying between different RT and ES versions (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5889
12.11.7 Using WinCC version compatibility....5891
12.11.7.1 Basics on version compatibility (Basic Panels, Panels, Comfort Panels, RT Advanced) ..5891
12.11.7.2 Editing projects of a previous WinCC version (Basic Panels, Panels, Comfort Panels, RT Advanced)....5893
12.11.7.3 Upgrading projects (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5894
12.11.7.4 Upgrading a global library (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5895
12.11.7.5 Changing between device versions (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5896
12.11.7.6 Changing the device version (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5897
12.11.8 Viewing memory card data....5898
12.11.8.1 Basics (Basic Panels, Panels, Comfort Panels)....5898
12.11.8.2 Working with backups....5898
12.11.9 HMI device wizard basics (Basic Panels, Panels, Comfort Panels)....5900
12.11.10 Managing colors centrally....5902
12.11.10.1 Basic principles for color management....5902
12.11.10.2 Finding and replacing colors....5903
12.11.11 Exchanging controller data from other projects....5905

12.12 Compiling and loading....5906
12.12.1 Establishing a connection to the HMI device....5906
12.12.2 Runtime settings....5907
12.12.2.1 Runtime settings....5907
12.12.2.2 Configuring display in Runtime (Comfort Panels)....5909
12.12.2.3 Configuring operation in Runtime....5910
12.12.2.4 Compatibility check with the controller (Panels, Multipanels, Comfort Panels, Mobile Panels, RT Advanced)....5911
12.12.2.5 Password protection in runtime....5912
12.12.3 Compiling and loading projects....5914
12.12.3.1 Overview of compiling and loading projects....5914
12.12.3.2 Compiling a project (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)....5915
12.12.3.3 Loading projects....5916
12.12.3.4 Compiling and loading with multiuser engineering (Basic Panels)....5922
12.12.3.5 Runtime start....5925
12.12.4 Simulating projects....5926
12.12.4.1 Simulation basics....5926
12.12.4.2 Simulating a project (Panels, Comfort Panels, RT Advanced)....5928
12.12.4.3 Working with the tag simulator (Basic Panels, Panels, Comfort Panels, RT Advanced)....5930
12.12.4.4 Simulation restrictions....5931
12.12.5 Reducing the project size (Basic Panels, Panels, Comfort Panels, RT Advanced)....5932
12.12.6 Basics of operating in Runtime....5933
12.12.6.1 Overview (Basic Panels)....5933
12.12.6.2 Operation with the touch screen....5934
12.12.6.3 Operation with keys....5938
12.12.6.4 Navigating in the display (BS)....5942
12.12.6.5 Triggering an action....5943
12.12.6.6 Entering a value....5943
12.12.6.7 Moving operator controls....5944
12.12.6.8 Displaying infotext....5945
12.12.6.9 Changing Runtime language....5946
12.12.7 Entering bar codes with optical handheld readers....5946
12.12.8 Security on the HMI device....5948
12.12.8.1 Signature basics (Basic Panels, Panels, Comfort Panels, RT Advanced)....5948
12.12.8.2 Switching off the signature check (Basic Panels, Panels, Comfort Panels, RT Advanced)....5949
12.12.9 Servicing the HMI device....5949
12.12.9.1 Overview of HMI device maintenance (Basic Panels)....5949
12.12.9.2 ProSave....5950
12.12.9.3 Backup of HMI data....5951
12.12.9.4 Backing up and restoring data of the HMI device....5952
12.12.9.5 Updating the operating system....5953
12.12.9.6 Updating the HMI device operating system....5954
12.12.10 Reference....5956
12.12.10.1 Error messages during loading of projects....5956
12.12.10.2 Adapting the project for another HMI device....5957
12.12.10.3 Establishing a connection to the HMI device....5958

12.13 Performance features....5960
12.13.1 General technical data....5960
12.13.1.1 Permitted characters....5960
12.13.1.2 Recommended printers....5961

12.13.1.3 Memory requirements of recipes for Basic Panels....5961
12.13.2 Performance features of the devices....5962
12.13.2.1 Basic Panel (Basic Panels)....5962
12.13.2.2 Basic Panel 2nd Generation (Basic Panels)....5966

12.14 Migrating to WinCC in the TIA Portal....5970
12.14.1 Overview of the migration to WinCC in the TIA Portal....5970
12.14.2 Libraries....5970
12.14.3 Screens and templates....5972

13 Using technology functions....5975

13.1 PID control....5975
13.1.1 Principles for control....5975
13.1.1.1 Controlled system and actuators....5975
13.1.1.2 Controlled systems....5976
13.1.1.3 Characteristic values of the control section....5978
13.1.1.4 Pulse controller....5980
13.1.1.5 Response to setpoint changes and disturbances....5984
13.1.1.6 Control Response at Different Feedback Structures....5985
13.1.1.7 Selection of the controller structure for specified controlled systems....5993
13.1.1.8 PID parameter settings....5994
13.1.2 Configuring a software controller....5994
13.1.2.1 Overview of software controller....5994
13.1.2.2 Steps for the configuration of a software controller....5996
13.1.2.3 Add technology objects....5997
13.1.2.4 Configure technology objects....5998
13.1.2.5 Call instruction in the user program....5999
13.1.2.6 Downloading technology objects to device....6000
13.1.2.7 Commissioning software controller....6001
13.1.2.8 Save optimized PID parameter in the project....6001
13.1.2.9 Comparing values....6002
13.1.2.10 Parameter view....6005
13.1.2.11 Display instance DB of a technology object....6023
13.1.3 Using PID_Compact (S7-1200, S7-1500)....6023
13.1.3.1 Technology object PID_Compact (S7-1200, S7-1500)....6023
13.1.3.2 PID_Compact V2 (S7-1200, S7-1500)....6024
13.1.3.3 PID_Compact V1 (S7-1200, S7-1500)....6044
13.1.4 Using PID_3Step (S7-1200, S7-1500)....6060
13.1.4.1 Technology object PID_3Step (S7-1200, S7-1500)....6060
13.1.4.2 PID_3Step V2 (S7-1200, S7-1500)....6061
13.1.4.3 PID_3Step V1 (S7-1200, S7-1500)....6079
13.1.5 Using PID_Temp (S7-1200, S7-1500)....6096
13.1.5.1 Technology object PID_Temp (S7-1200, S7-1500)....6096
13.1.5.2 Configuring PID_Temp (S7-1200, S7-1500)....6097
13.1.5.3 Commissioning PID_Temp (S7-1200, S7-1500)....6118
13.1.5.4 Cascade control with PID_Temp (S7-1200, S7-1500)....6126
13.1.5.5 Multi-zone controlling with PID_Temp (S7-1200, S7-1500)....6133
13.1.5.6 Override control with PID_Temp (S7-1200, S7-1500)....6135
13.1.5.7 Simulating PID_Temp with PLCSIM (S7-1200, S7-1500)....6139

13.2 Using S7-1200 Motion Control (S7-1200)....6140
13.2.1 Introduction (S7-1200)....6140
13.2.1.1 Motion functionality of the CPU S7-1200 (S7-1200)....6140

13.2.1.2 Hardware components for motion control (S7-1200)....6141
13.2.2 Basics for working with S7-1200 Motion Control (S7-1200)....6143
13.2.2.1 Stepper motor on the PTO (S7-1200)....6143
13.2.2.2 PROFIdrive drive / analog drive connection (S7-1200)....6150
13.2.2.3 Hardware and software limit switches (S7-1200)....6164
13.2.2.4 Jerk limit (S7-1200)....6165
13.2.2.5 Homing (S7-1200)....6166
13.2.3 Guidelines on use of motion control (S7-1200)....6168
13.2.4 Using versions (S7-1200)....6168
13.2.4.1 Overview of versions (S7-1200)....6168
13.2.4.2 Changing a technology version (S7-1200)....6172
13.2.4.3 Compatibility list of variables V1...3 <-> V4...5 (S7-1200)....6172
13.2.4.4 Compatibility list of variables V4...5 <-> V6 (S7-1200)....6175
13.2.4.5 Status of limit switch (S7-1200)....6176
13.2.5 Positioning axis technology object (S7-1200)....6177
13.2.5.1 Integration of the positioning axis technology object (S7-1200)....6177
13.2.5.2 Tools of the positioning axis technology object (S7-1200)....6180
13.2.5.3 Adding a positioning axis technology object (S7-1200)....6182
13.2.5.4 Configuring the positioning axis technology object (S7-1200)....6183
13.2.6 Technology object command table (S7-1200)....6241
13.2.6.1 Use of the command table technology object (S7-1200)....6241
13.2.6.2 Command table technology object tools (S7-1200)....6241
13.2.6.3 Adding the technological object command table (S7-1200)....6242
13.2.6.4 Configuring the command table technology object (S7-1200)....6243
13.2.7 Download to CPU (S7-1200)....6260
13.2.8 Commissioning (S7-1200)....6261
13.2.8.1 Axis control panel (S7-1200)....6261
13.2.8.2 Tuning (S7-1200)....6264
13.2.9 Programming (S7-1200)....6266
13.2.9.1 Overview of the Motion Control statements (S7-1200)....6266
13.2.9.2 Creating a user program (S7-1200)....6267
13.2.9.3 Programming notes (S7-1200)....6270
13.2.9.4 Behavior of the Motion Control commands after POWER OFF and restart (S7-1200)....6272
13.2.9.5 Monitoring active commands (S7-1200)....6272
13.2.9.6 Error displays of the Motion Control statements (S7-1200)....6283
13.2.9.7 Restart of technology objects (S7-1200)....6284
13.2.10 Axis - Diagnostics (S7-1200)....6285
13.2.10.1 Status and error bits (technology objects as of V4) (S7-1200)....6285
13.2.10.2 Motion status (S7-1200)....6288
13.2.10.3 Dynamics settings (S7-1200)....6288
13.2.10.4 PROFIdrive frame (S7-1200)....6289
13.2.11 Appendix (S7-1200)....6290
13.2.11.1 Using multiple axes with the same PTO (S7-1200)....6290
13.2.11.2 Using multiple drives with the same PTO (S7-1200)....6293
13.2.11.3 Tracking jobs from higher priority classes (execution levels) (S7-1200)....6294
13.2.11.4 Special cases when using software limit switches for drive connection via PTO (S7-1200)....6296
13.2.11.5 Reducing velocity for a short positioning duration (S7-1200)....6302
13.2.11.6 Dynamic adjustment of start/stop velocity (S7-1200)....6303
13.2.11.7 Move the axis without position control for servicing (S7-1200)....6303
13.2.11.8 List of ErrorIDs and ErrorInfos (technology objects as of V6) (S7-1200)....6304
13.2.11.9 Tags of the positioning axis technology object as of V6 (S7-1200)....6328
13.2.11.10 Versions V1...5 (S7-1200)....6364

14 Using online and diagnostics functions....6475

14.1 Displaying accessible devices....6475

14.2 Changing the device configuration online....6477

14.3 Connecting devices online....6478
14.3.1 General information about online mode....6478
14.3.2 View in online mode....6480
14.3.3 Default setting online connection data....6481
14.3.4 Establishing or changing an online connection....6482
14.3.5 Canceling an online connection....6484
14.3.6 Connecting online with several devices....6485
14.3.7 Disconnecting online connections of multiple devices....6486

14.4 Creating a backup of an S7 CPU....6487
14.4.1 Backup options for S7 CPUs....6487
14.4.2 Backing up S7-300 and S7-400 CPUs (S7-300, S7-400, PC)....6489
14.4.2.1 Creating a backup of a device (S7-300, S7-400, PC)....6489
14.4.2.2 Restoring the software and hardware configuration of a device (S7-300, S7-400, PC)....6490
14.4.2.3 Backing up a device configuration (S7-300, S7-400, PC)....6490
14.4.3 Backing up S7-1200 and S7-1500 CPUs (S7-1200, S7-1500, S7-1500, S7-1500)....6491
14.4.3.1 Creating a backup of a device (S7-1200, S7-1500, S7-1500, S7-1500)....6491
14.4.3.2 Backing up a device configuration (S7-1200, S7-1500, S7-1500, S7-1500)....6492
14.4.3.3 Restoring the configuration of a device (S7-1200, S7-1500, S7-1500, S7-1500)....6493

14.5 Configuring the PG/PC interface....6495
14.5.1 Online access....6495
14.5.2 Basics of assigning parameters for the PG/PC interface....6497
14.5.3 Showing or hiding interfaces....6497
14.5.4 Displaying and modifying interface properties....6498
14.5.5 Adding interfaces....6498
14.5.6 Setting parameters for the Ethernet interface....6499
14.5.6.1 Setting parameters for the Industrial Ethernet interface....6499
14.5.6.2 Displaying operating system parameters....6500
14.5.6.3 Connecting the PG/PC interface to a subnet....6500
14.5.6.4 Setting parameters for the Ethernet interface....6501
14.5.6.5 Assigning a temporary IP address....6501
14.5.6.6 Managing temporary IP addresses....6502
14.5.6.7 Resetting the TCP/IP configuration....6502
14.5.7 Setting parameters for the MPI and PROFIBUS interfaces....6503
14.5.7.1 Setting parameters for the MPI and PROFIBUS interfaces....6503
14.5.7.2 Setting MPI or PROFIBUS interface parameters automatically....6504
14.5.7.3 Setting parameters for the MPI interface....6505
14.5.7.4 Setting parameters for the PROFIBUS interface....6507
14.5.7.5 Overview of the bus parameters for PROFIBUS....6509
14.5.7.6 Resetting the MPI or PROFIBUS configuration....6510

14.6 Using the trace and logic analyzer function....6511
14.6.1 Preface....6511
14.6.1.1 Security information....6512
14.6.2 Description....6513
14.6.2.1 Supported hardware....6513
14.6.2.2 Recording of measured values with the trace function....6513
14.6.2.3 Trace configuration, recording, installed trace and measurement....6515

14.6.2.4 Data storage....6516
14.6.3 Software user interface....6517
14.6.3.1 Project navigator....6519
14.6.3.2 Working area....6524
14.6.3.3 Trace task card....6534
14.6.3.4 Inspector window....6536
14.6.4 Operation....6537
14.6.4.1 Quick start....6537
14.6.4.2 Using the trace function - overview....6541
14.6.4.3 Project tree....6542
14.6.4.4 Working area - general....6544
14.6.4.5 Working area - Configuration tab....6549
14.6.4.6 Working area - Diagram tab....6550
14.6.5 Devices....6557
14.6.5.1 S7-1200/1500 CPUs....6557

14.7 Establishing a remote connection with TeleService....6573
14.7.1 Basics of working with TeleService....6573
14.7.1.1 Introduction to TeleService....6573
14.7.1.2 TeleService functionality....6574
14.7.1.3 Telephone book at TeleService....6574
14.7.2 Working with the phone book....6575
14.7.2.1 Basics on working with the phone book....6575
14.7.2.2 Structure of the phone book....6576
14.7.2.3 Symbols in the phone book....6577
14.7.2.4 Manage phone book....6577
14.7.3 Remote connections as dial-up connections....6583
14.7.3.1 Basics for establishing a dial-up connection....6583
14.7.3.2 Telephone networks and modems....6584
14.7.3.3 Access protection for dial-up connections....6587
14.7.3.4 TS adapter MPI....6592
14.7.3.5 TS adapter IE....6599
14.7.3.6 Establishing a dial-up connection to a remote system....6605
14.7.4 Remote VPN connections....6606
14.7.4.1 Basics for establishing a VPN connection....6606
14.7.4.2 Basics of CA certificates....6607
14.7.4.3 Installing CA certificates for VPN connections....6609
14.7.4.4 Deleting CA certificates for VPN connections....6612
14.7.4.5 Establishing a VPN connection to a remote system....6613
14.7.4.6 TS Adapter IE Advanced....6615
14.7.5 CPU controlled TeleService remote connections....6619
14.7.5.1 Overview of CPU controlled remote connections....6619
14.7.5.2 Establishing a connection from and to remote systens (PG-AS-remote coupling)....6620
14.7.5.3 Data exchange between remote systems (AS-AS-remote coupling)....6621
14.7.5.4 Send SMS from a system....6623
14.7.5.5 Send an email from a system....6624
14.7.6 Notes on troubleshooting....6627
14.7.6.1 General information on troubleshooting for modem problems....6627
14.7.6.2 Recording a log file for the modem....6627
14.7.6.3 Dial-up connection to the TS Adapter is not established....6628
14.7.6.4 Dial-up connection from the TS Adapter is not established....6630
14.7.6.5 Modem connection is interrupted....6631
14.7.6.6 Checklist for troubleshooting the modem....6631

14.7.6.7 Modem alarms....6632
14.7.6.8 Possible error messages with VPN connections....6633

14.8 Simulating devices with S7-PLCSIM....6634
14.8.1 Simulation of devices....6634
14.8.2 Starting the simulation....6634

15 Using Team Engineering....6635

15.1 Shared commissioning of projects....6635
15.1.1 Basics for shared commissioning....6635
15.1.2 Requirements for shared commissioning....6638
15.1.3 Procedure for shared commissioning....6639
15.1.4 Rules for shared commissioning....6642

15.2 Exchanging data with Inter Project Engineering (IPE)....6648
15.2.1 Basics of Inter Project Engineering (IPE)....6648
15.2.2 Requirements for Inter Project Engineering (IPE)....6651
15.2.3 Overview for working with Inter Project Engineering (IPE)....6651
15.2.4 Creating "Device proxy data" in the source project....6654
15.2.5 Creating an IPE file by means of the "Device proxy data"....6655
15.2.6 Using controller data from other projects with IPE....6655
15.2.6.1 Using controller data from other projects in an HMI device....6655
15.2.6.2 Communication with device proxies....6667
15.2.6.3 Integrated configuring with WinCC and SIMATIC Manager....6674

15.3 Using Multiuser Engineering....6687
15.3.1 Introduction to Multiuser Engineering....6687
15.3.2 Overview of the mode of operation with Multiuser Engineering....6689
15.3.3 Basics of the multiuser server concept....6693
15.3.4 Requirements for working with Multiuser Engineering....6696
15.3.5 Supported objects for Multiuser Engineering....6699
15.3.6 Settings for working with Multiuser Engineering....6700
15.3.7 Requirements for multiuser projects....6703
15.3.8 Operator controls in Multiuser Engineering....6704
15.3.8.1 Multiuser icons in the user interface....6704
15.3.8.2 Multiuser icons in the user interface....6706
15.3.8.3 Multiuser banner in the user interface....6708
15.3.8.4 Functionality of the multiuser editor....6708
15.3.9 Installing and uninstalling the Multiuser server....6711
15.3.9.1 Notes on the installation of the multiuser server....6711
15.3.9.2 Installing the multiuser server....6712
15.3.9.3 Uninstalling the multiuser server....6713
15.3.10 Configuring and managing the multiuser server....6714
15.3.10.1 Options for configuring and administering the multiuser server....6714
15.3.10.2 Using the graphic tool....6715
15.3.10.3 Using the command line tools....6734
15.3.10.4 Confirm the certificate for a multiuser server....6746
15.3.10.5 Enter access data for the multiuser server....6746
15.3.10.6 Transferring server data to other multiuser servers....6747
15.3.10.7 Reading out error log for multiuser server....6748
15.3.10.8 Error log for multiuser server packed as Zip file....6750
15.3.10.9 Information on response of the multiuser server to errors....6750
15.3.10.10 Supported network configurations for the multiuser server....6751
15.3.11 Working with the multiuser server....6766

15.3.11.1 Introduction to the multiuser server....6766
15.3.11.2 Setting the multiuser network profiles....6767
15.3.11.3 Adding a new multiuser server connection....6768
15.3.11.4 Displaying the multiuser server connection....6769
15.3.11.5 Editing the multiuser server connection....6770
15.3.11.6 Restore multiuser server connection....6771
15.3.11.7 Deleting the multiuser server connection....6772
15.3.12 Working with the local multiuser server....6773
15.3.12.1 Introduction to the local multiuser server....6773
15.3.12.2 Configuring the local multiuser server....6774
15.3.12.3 Displaying the local multiuser server....6775
15.3.12.4 Starting the local multiuser server....6775
15.3.12.5 Exiting the local multiuser server....6777
15.3.13 Creating and managing multiuser server projects....6777
15.3.13.1 Introduction to working in multiuser server projects....6777
15.3.13.2 Adding a new multiuser server project....6778
15.3.14 Working in the local session....6779
15.3.14.1 Creating and managing a local session....6779
15.3.14.2 Editing objects in the local session....6788
15.3.15 Working in the server project view....6806
15.3.15.1 Introduction to working in the server project view....6806
15.3.15.2 Opening and closing the server project view....6808
15.3.15.3 Editing objects in the server project view....6809
15.3.15.4 Renaming objects in the server project view....6811

Index....6813

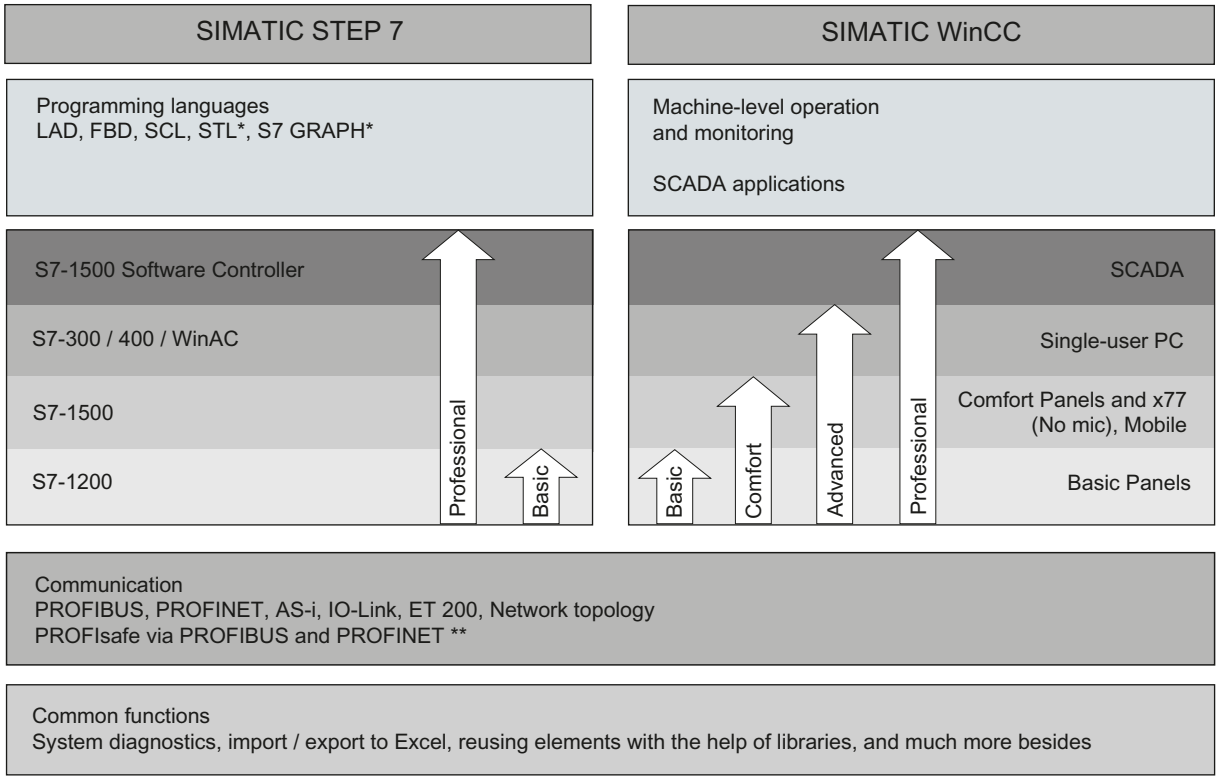
System overview of STEP 7 and WinCC

1

1.1 Scaling of STEP 7 and WinCC in the TIA Portal

Scope of performance of the products

The following graphic shows the scope of performance of the individual products of STEP 7 and WinCC:



* Only for STEP 7 Professional for S7-300 / 400 / WinAC and S7-1500
** With installed optional package "STEP 7 Safety Advanced"

STEP 7

STEP 7 (TIA Portal) is the engineering software for configuring the SIMATIC S7-1200, S7-1500, S7-300/400 and WinAC controller families. STEP 7 (TIA Portal) is available in two editions, depending on the configurable controller families:

- STEP 7 Basic for configuring the S7-1200
- STEP 7 Professional for configuring S7-1200, S7-1500, S7-300/400 and Software-Controller (WinAC)

WinCC

WinCC (TIA Portal) is an engineering software for configuring SIMATIC Panels, SIMATIC Industrial PCs, and Standard PCs with the WinCC Runtime Advanced or the SCADA System WinCC Runtime Professional visualization software.

WinCC (TIA Portal) is available in four editions, depending on the configurable operator control systems:

- WinCC Basic for configuring Basic Panels
  WinCC Basic is included with every STEP 7 Basic and STEP 7 Professional product.
- WinCC Comfort for configuring all panels (including Comfort Panels, Mobile Panels)
- WinCC Advanced for configuring all panels and PCs with the WinCC Runtime Advanced visualization software
  WinCC Runtime Advanced is a visualization software for PC-based single-station systems. WinCC Runtime Advanced can be purchased with licenses for 128, 512, 2k, 4k, 8k and 16k PowerTags (tags with a process interface).
- WinCC Professional for configuring panels and PCs with WinCC Runtime Advanced or SCADA System WinCC Runtime Professional. WinCC Professional is available in the following editions: WinCC Professional for 512 and 4096 PowerTags as well as "WinCC Professional max. PowerTags".
  WinCC Runtime Professional is a SCADA system for structuring a configuration ranging from single-station systems to multi-station systems including standard clients or web clients. WinCC Runtime Professional can be purchased with licenses for 128, 512, 2k, 4k, 8k, 64k, 100k, 150k and 256k PowerTags (tags with a process interface).

With WinCC (TIA Portal), it is also possible to configure a SINUMERIK PC with WinCC Runtime Advanced or WinCC Runtime Professional and HMI devices with SINUMERIK HMI Pro sl RT or SINUMERIK Operate WinCC RT Basic.

1.2 Options for STEP 7 and WinCC Engineering

The following cross-product options are available for STEP 7 and WinCC Engineering:

- TIA Portal Multiuser (multiple users work jointly on a TIA Portal project)
- TIA Portal Cloud Connector (access to local interface using RDP)
- TIA Portal Teamcenter Gateway (connection to Teamcenter)
- SIMATIC Visualization Architect (creating HMI contents based on STEP7 projects)
- SIMATIC Energy Suite (energy management)

1.3 Options for S7-1500 Runtime

The engineering of the following functions is integrated in the engineering packages of SIMATIC STEP 7 Professional and/or SIMATIC Energy Suite. The enabling takes place in CPU context:

- SIMATIC ProDiag (machinery and plant diagnostics for S7-1500 and SIMATIC HMI)
- SIMATIC OPC UA S7-1500 (connection of any third-party devices to the S7-1500)
- SIMATIC Energy Suite S7-1500 (energy management)

1.4 Options for STEP 7 Engineering System

The following options are available for STEP 7 Engineering:

- SIMATIC STEP 7 Safety Basic/Advanced (safety programs for F-CPUs)
- SIMATIC S7 PLCSIM Advanced (Virtual Controller S7-1500)
- SIMATIC Target 1500S™ for Simulink® (add-on for Simulink®)

1.5 Options for WinCC Engineering and Runtime systems

SIMATIC Panels as well as WinCC Runtime Advanced and WinCC Runtime Professional contain all essential functions for operator control and monitoring of machines or plants. Additional options allow you to extend the functionality in some cases to increase the range of available tasks.

Option for Basic Panel

The following add-on is available for Basic Panels:

- WinCC Sm@rtServer (remote operation)

Options for Comfort Panels, Mobile Panels

The following extension options are available for Comfort Panels and Mobile Panels:

- WinCC Audit (audit trail and electronic signature for regulated applications)
- SIMATIC ProDiag (machinery and plant diagnostics for S7-1500 and SIMATIC HMI)

Option for Multi Panels

The following add-on is available for Multi Panels:

- WinCC Audit (audit trail and electronic signature for regulated applications)

Note

In contrast to WinCC flexible 2008, functions from the WinCC flexible /Sm@rtService, WinCC flexible /Sm@rtAccess options as well as the WinCC flexible /OPC Server option are incorporated into the basic functionality.

Options for WinCC Runtime Advanced

The following possible extensions are available for WinCC Runtime Advanced:

- WinCC SmartServer (remote operation)
- WinCC Recipes (recipe system)
- WinCC Logging (logging of process values and alarms)
- WinCC Audit (audit trail for regulated applications)
- SIMATIC ProDiag (machinery and plant diagnostics for S7-1500 and SIMATIC HMI)

Note

In contrast to WinCC flexible 2008, functions from the WinCC flexible /Sm@rtService, WinCC flexible /Sm@rtAccess options as well as the WinCC flexible /OPC Server option are incorporated into the basic functionality.

Options for WinCC Runtime Professional

The following possible extensions are available for WinCC Runtime Professional:

- WinCC Client (standard client for structuring multi-station systems)
- WinCC Server (supplements WinCC Runtime to include server functionality)
- WinCC Recipes (recipe system, formerly WinCC / UserArchives)
- WinCC WebNavigator (Web-based operator control and monitoring)
- WinCC DataMonitor (display and evaluation of process states and historical data)
- WinCC ControlDevelopment (extension by means of customer-specific controls)
- WebUX (platform and browser-independent operation and monitoring via the Web)
- SIMATIC Information Server 2014 (Web-based and browser-independent analysis and reports of historic process data)
- SIMATIC Process Historian 2014 (plant-wide archive server for messages and process data)
- Industrial Data Bridge (configurable connections to databases and IT systems)
- Redundancy (increased availability due to redundant server)
- SIMATIC ProDiag (machinery and plant diagnostics for S7-1500 and SIMATIC HMI)

Note

In contrast to WinCC V7, functions from the WinCC /OPC-Server and WinCC / ConnectivityPack options are incorporated into the basic functionality. Likewise, the basic functionality includes the Runtime API from WinCC /ODK.

What's new in TIA Portal

2

2.1 What's new in TIA Portal V14 SP1

2.1.1 Introduction

Description

All important new features compared to the present version V14 are summarized here. You can find additional details on the various topics in the individual sections of the product documentation.

2.1.2 SIMATIC STEP 7

Description

All important new features of STEP 7 are shown grouped together based on the engineering workflow.

Hardware configuration

- New PS 60W 24/48/60VDC HF system power supply for the S7-1500 provides an energy buffer for data backup of up to 20 MB in the CPU.
- Tabular configuration of the I/O channels for modules of the ET 200MP DI HF and ET 200MP DO HF including copy-and-paste and auto-complete function
- Editing of devices names in the graphic views
- Enhanced zooming and optimized (multiple) selection in the graphic network and topology view
- Overview of I/O tags of an entire device (such as an ET 200SP station)
- Easy removal of unused GSD files in the project
- Wizard enhancement for UDP Multicast for Open User Communication (OUC) with S7-1500

Editors for programming languages

- In SCL program code structured tags and ARRAY elements can be replaced by other tags using drag-and-drop operation.
- In the SCL editor, it is possible to switch between insert and overwrite mode.

- In SCL, selected program code segments can be simply enclosed with specific structure elements, such as "IF..Then".
- The detailed comparison of individual blocks can be started directly from the project tree (offline/offline, but also online/offline).
- In LAD/FBD, the tag comment can be shown directly at the point of use.
- The F7 functions ("Open block/PLC data type…") can also be started with focus inside editors.
- Generating external source file from blocks: The export of objects as external source file takes into consideration the PLC data types (UDTs), multi-instances and all called blocks defined in the block interface.
- All the PLC instructions used can be updated to the latest version release in the "Instructions" task card.

New instructions

- The values can be moved between tags of the data type BYTE, WORD, DWORD, LWORD and tag ARRAYs of the data type BOOL using the new instructions "SCATTER" and "GATHER" for the S7-1200/1500. As a result, control and status words can, for example, be simply broken down into single bit signals and reassembled.
- Information about the inserted SIMATIC memory card - such as the memory size, the amount of storage space used or the number of already completed write/delete actions (as percentage) – can be read out using the "GetSMCInfo" instruction.
- The "Polyline" instruction maps the input value to the output value using a characteristic curve. The characteristic curve is defined as a polyline (with up to 50 interpolation points).

Language innovations

- PLC data types (UDT) and ARRAY comments can be modified instance-specific.
- The assignment of various PLC data types (UDTs) with the same structure but different symbolic names is supported.
- S7-GRAPH blocks as of version 4.0 enable the initial value acquisition of the disturbed operands. The actual values of the signal states of Boolean operands and the results of comparator operations in interlocks and transitions are recorded continuously from the time of activation of the initial value acquisition, even if no error occurs. The signal states are stored for each cycle in the GRAPH instance data block. A maximum of 32 signal states are recorded per interlock or per transition of a GRAPH step.

Libraries

- Project libraries and global libraries can now be separately translated into additional foreign languages.
- Simplified upgrading of types (blocks, PLC data types) in an existing project library.
- The user-defined help can be now also be started in the context of the project language. Until V14, this was only possible in the context of the TIA Portal language.

Messages/diagnostics

- The display of the alarm display in the Inspector window has been optimized and functionally enhanced with new functions such as "Receive alarms" and "Freeze alarms".

Migration

- The local constants for ARRAY limits that are used in the SCL sources are retained during the migration from STEP 7 V5.x to TIA Portal.

Trace

- Switching the display format (for example: A WORD can be shown in hexadecimal format, as signed or unsigned INT)
- The scroll bar of the X axis has a small preview window ("Time range display"), with which you can recognize which are has just been zoomed.

System functions

- CAx data based on XML, for exchange of ECAD system for example, can be exported and re-imported.
- Filters are supported for cross-references that are freely definable by the user, for example, in order to find multiple accesses to tags.
- The TIA Portal can be connected to an external password manager via a password provider. This offers the advantage that passwords do not have to be released. This increases the security of the program code.

SIMATIC Ident

The technology object (TO) for SIMATIC Ident connects the hardware configuration of communication modules and readers to the programming of an S7-1200 and S7-1500 controller. The TO hereby prevents incorrect settings and detects errors in the parameter assignment.

Hardware configuration:

- The plant builder assigns a "unique name" to each communication module / reader with which it can be identified by both commissioning engineers and programmers.
- The parameter assignment of each communication module, reader or reading point is set via the TO.
- The Ident devices can be configured with a single click for operation with the TO using the "Auto Configuration" button.

New instructions:

- All instructions for SIMATIC Ident have been revised for the collaboration with the TO. You can find the instructions in the TIA Portal under "Optional packages" > "SIMATIC Ident".
- The hardware addressing of a reader to the instructions has been changed so that the reader is selected through the "unique name" which was defined in the TO. There is no need, therefore, for the previous searching, noting and transferred of the HW identifier and logical address.
- Optional inputs at the instructions are hidden. The user thus recognizes immediately the important parameters that are required for a function.
- The programmer no longer has to assign parameters to the hardware. Only the RESET_READER instruction, which has no other input parameters, is needed for the PLC startup, regardless of the reader connected. The instruction acquires the required parameters from a data area which was configured with the TO.

Diagnostics:

- The last messages which were reported by the instructions to the application can be displayed with the TO.
- A time stamp and the exact error text help to achieve fast analysis of the error.

2.1.3 SIMATIC WinCC

Function enhancement - Openness

- Support of HMI faceplates
- Support of HMI slide-in screens
- Support of HMI pop-up screens

Enhancements for SIMATIC HMI panels

- SIMATIC HMI BASIC Panels:
  - Unicode CSV export of recipes for Basic Panels
  - Changing of IP addresses of controllers and additional SIMATIC components (such as CPs) via the HMI device.
  - Support for Chinese, Japanese, Korean, Russian and Arabic fonts in the browser.
- Updating of tested network cameras (AXIS M1013, AXIS M1011, D-Link DCS-942L, D-LINK 4701E, Siemens CCMS2025).
- Installation/deinstallation of Word, Excel, Win Mediaplayer, IE via ProSave.
- Unlimited number of usable scripts for Comfort.
- Configuration for Startup (additional value "60s" for RT start time and system function for Restart Panel).
- Additional system functions in the "Zones" editor for Mobile Panels.

Enhancement for SIMATIC Runtime Advanced

- Camera displays for showing live video streams from network cameras
- Enhancements of the PDF display (search and copy)
- License-free use of the Sm@rtServer on Runtime Advanced (similar to Comfort Panels)

Integration of SINUMERIK Operate (GUI and data) in Runtime Advanced

- Access to SINUMERIK Operate user interface via new NC operator display in screens of RT Advanced (also multitouch operation)
- Access to SINUMERIK NCU data via separate SINUMERIK channel

Enhancements for SIMATIC Runtime Professional

- Support for OPC UA DA client
- New VBS methods for screen windows. This provides you with the option of temporarily switching off the operability of the screen window.
- Customization of the recipe view. This provides you with the option of adapting the properties of the controls (buttons or table elements).

2.1.4 SIMATIC STEP 7 Safety

Description

SIMATIC STEP 7 Safety Basic/Advanced V14 SP1, the high-performance option package for programming fail-safe S7 controllers for the Totally Integrated Automation Portal V14 SP1.

Station upload

The consistent hardware and software upload enables a loading of project data (including safety-related project data) from a fail-safe S7-1500 CPU*) to a programming device / PC. The project data are loaded without an additional generation step and are consistently available in the TIA Portal for immediate continued work.

*) As of firmware version V2.1

Configuration control

With Safety V14 SP1, the configuration control (option handling) for SIMATIC fail-safe I/O devices is supported in the following configurations:

- Distributed in an F-CPU S7-300/400/1200/1500
- Centrally in an F-CPU S7-1500 (including fail-safe ET200SP CPUs)

Cross-project IO controller I-device communication

Safety V14 SP1 enables cross-project communication of fail-safe S7-1200/1500/1500 software controllers to S7-300/400/1200/1500 I-devices. The I-device project can be easily created with STEP 7 Safety V13 (or higher) or also S7 Distributed Safety V5.4, exported as GSD file, and imported into the IO controller project.

Global F-IO Status block

Fail-safe F I/O devices provide a wide range of status information about the status of the channel and module using the QBAD tags and the value status bits.

In the Safety Administration editor, you can create a default FB for each F-runtime group in order to record and further process any channel, communication or fail-safe I/O errors, even in automation systems with very large volumes of project data. The value status bits and the QBAD tags of the fail-safe I/O used in the corresponding F-runtime group are evaluated and the result of the evaluation is made available at the "QSTATUS" output.

You can use the default FB in the standard program.

Openness

You can use the Openness command set in STEP 7 Safety, as in the standard version. This requires that no password is set for the safety-relevant project data. The use of Openness in production operation is not permitted.

Supported functions include the following:

- Working with libraries (deleting, copying, instancing and upgrading elements etc.)
- Importing tags, tag tables, PLC data types (UDTs), etc.
- Compiling of SW, blocks, PLC data types (UDTs).
- Plugging/deleting F-CPUs and fail-safe I/O
- Copying/deleting F-CPUs and fail-safe I/O from master copies
- Configuring networks
- Compiling software (including the safety program)
- Reading/configuring F-parameters of the F-CPU
- Reading/configuring F-parameters of the fail-safe I/O
- Reading, declaring or deleting fail-safe tags in the PLC tag table
- Updating the project to the latest type versions of F-blocks

2.1.5 SINAMICS Startdrive

Startdrive for G120

- Optimized and enhanced commissioning wizard:
  - The two former wizards for the standard commissioning and the operation under a SIMATIC Motion Control axis are combined in a single wizard
  - Integration of the selection of message frame, brake resistor, motor-side filter, temperature sensor, and selection of the motor, also via the motor code number
- Online/offline parameter comparison
- Support for Windows 10

Startdrive option for S120

- Automatic hardware configuration
- Support for drive-based extended safety with graphic parameterization masks
- Support for torque motors (1FW3, 1FW6)
- Networking with ET200 SP CPU
- Support for Windows 10

2.1.6 SINUMERIK STEP 7 Toolbox

Product features

The SINUMERIK STEP 7 Toolbox V14 SP1 is an option package for SIMATIC STEP 7 Professional V14 SP1 (TIA Portal) with additional setup.

Scope of functions

The SINUMERIK STEP 7 Toolbox V14 SP1 enables the configuration of the SINUMERIK hardware in the TIA Portal using the following functionalities and tools:

- Addition of the following modules of SINUMERIK 840D sl to the hardware catalog (as of firmware V4.5 SP2):
  - NCU 710.3B
  - NCU 720.3B
  - NCU 730.3B
  - NX10.3
  - NX15.3
- Addition of the ADI4 module to the hardware catalog
- Basic SINUMERIK PLC program for the versions
  - V4.5.x.x
  - V4.7.x.x
  - V4.8.x.x
    The basic PLC program is installed in the form of the "SINUMERIK 840D sl PLC Basic Program" system library using the SINUMERIK STEP 7 Toolbox V14 SP1.
- Support for hardware migration
- Creating SINUMERIK PLC commissioning logs, PLC hardware upgrade logs and reload logs
- Exporting of PLC symbols for SINUMERIK Operate
- Importing of SINUMERIK user alarm texts
- Creating SINUMERIK PLC logs
- PROFINET IO IRT support for NCK
- Support for SINUMERIK Safety Integrated and Safety Integrated plus
- NC-VAR-Selector (external tool)

2.1.7 Runtime options

Description

The following new options are only offered as a runtime option with TIA Portal V14 SP1. The ProDiag product is are already integrated into the existing STEP 7 and WinCC products and only needs to be licensed when used on the real hardware. In comparison with the Engineering options, the runtime options are not version-dependent.

SIMATIC ProDiag

- S7-GRAPH blocks as of version 4.0 enable the initial value acquisition of the disturbed operands.
- The initial values can be displayed on the HMI device using the "PLC code view" object and the faulty operands can be detected immediately in the case of error (criteria analysis). The initial values remain stored until they are overwritten by a new error.
- For the initial value acquisition, all instructions are supported which are supported in an S7-GRAPH function block for transitions and interlocks. For bit logic operations, the status of the operand is stored. For comparator operations, the result of the comparison is stored.

SIMATIC Energy Suite S7-1500

SIMATIC Energy Suite as an integrated option for the TIA Portal for the first time efficiently combines energy management with automation, bringing energy transparency to production. In addition, the configuration work is significantly reduced due to the simplified configuration of components for energy recording.

Functions:

- Integrated and intuitive configuration of energy management
- Automatic generation of the PLC energy program for S7-1500 for basic energy data (energy and power)
- Archiving to WinCC Runtime Professional or to a PLC-internal SIMATIC memory card
- Seamless connection to SIMATIC Energy Manager PRO

New in V14 SP1:

- Enhancement of the automatic PLC program generation with additional energy data (currents, voltages, frequency)
- In addition to the manual energy data export, the automatic export is now also offered (for example, start of the export of the energy data of the last week, each Monday at 00:15 hrs)
- Program generator: Detailed representation of the generation progress and configuration error messages.

2.2 What's new in TIA Portal V14

2.2.1 Introduction

Description

All important new features compared to the present version V13 SP1 are summarized here. You can find additional details on the various topics in the individual sections of the product documentation.

2.2.2 SIMATIC STEP 7

Description

All important new features of STEP 7 are shown grouped together based on the engineering workflow.

Hardware configuration

- Support of the S7-1500 T-CPU family, with the Motion Control functions "constant velocity" and "cam disks".
- Support of the new CPU 1518(F)-4 PN/DP ODK (Open Development Kit), which enables C/C++ functions created via the SIMATIC ODK 1500S to be integrated into the user program
- Support of the CPU 1516pro (F)-2 PN
- Support of the new S7-1500 failsafe software controller
- IO devices can be grouped in the project tree to quickly find and technologically merge devices.
- The arrangement of the devices in the topological view can be synchronized with the network view.
- Own profiles can be defined in the hardware catalog for easy and quick access to frequently used modules.

Editors for programming languages

- Separate editors are available for the new technology objects Output Cam, Cam Track and Measuring Input. There is also a convenient cam disk editor for the S7-1500T controllers.
- SCL networks can be inserted into a LAD/FBD block.
- "Regions" can be used in SCL program code to improve structuring and navigation.
- You can hide the block parameters when calling the block in LAD or FBD.

- The comparison of PLC programs has been thoroughly revised and enables the quick and convenient comparison of source code with many filter options.
- Networks in the programming editors for LAD / FBD / STL are automatically displayed as they were when you last exited the block.

Language innovations

- ARRAYs of variable length can be passed with ARRAY[*].
- Multiple instances can be declared in ARRAYs in the block interface and called in loops. This helps to significantly reduce the program code and provides better readability and more efficient programming when processing the same program objects.
- The data type DB_ANY can also be used for technology objects. This allows technology objects to be handled more flexibly in the program. An "ARRAY of DB_ANY" may be a list of positioning axes, for example.
- Instances can be passed as a parameter in the "InOut" section of the block interface so that only one dedicated instance needs to be selected in the call.
- Message texts and comments for up to 3 languages can be loaded into the PLC S7-1500 and changed online.

Messages/diagnostics

- Using the new "Get_Alarm" instruction, messages can be directly read from the message server of the S7-1500 CPU and, sent and to a higher-level reporting system (e.g. process control system), for example.
- With the instruction "GET_DIAG" in Mode 1, the diagnostic data are output in the DIS structure. Bit 15 of the parameter IOState of this structure outputs the signal status "1" when a network / hardware fault has occurred.

Trace

- The Trace function enables different measurements to be displayed in a shared diagram ("overlaid measurements").
- In addition, with the Trace function, after completion, messages can be stored on the memory card independently of the power supply, and if required, can be automatically restarted.

Online functions

- Restructuring of the toolbar in the DB editor with the possibility to reinitialize complete data blocks with the defined start values.
- Structures in the block can be observed directly in the Inspector window including the call environment.

System functions

- The new global search enables cross-project searching for PLC and HMI objects. In addition, the search results can be restricted by filter. A "GoTo" function is also available.
- TIA Portal settings can be exported or imported and stored on a central server. If the server path is available, for example, the stored settings are used automatically as "default" every time you restart the TIA Portal.
- Innovative help/information system with tab technology, improved search function and the possibility to filter information by device families.
- The new cross-reference list provides a quick overview of the use of objects and devices within the project.
  - Support of overlapping accesses (PLC tags)
  - Support of know-how-protected blocks
  - Access to Individual elements of an ARRAY
  - Representation of objects which occur both in HMI and PLC devices
  - Free entry of objects to be searched for in the cross-reference list possible
  - Various filter options
  - Convenient and central provision of updates in company networks.
  - A company server can be configured with the TIA Automation Corporate Configuration Tool. The available updates / support packages are stored on the server and are thus available to users.

S7-PLCSIM

- Quick start of S7-PLCSIM in compact mode and without a simulation project.
- Devices can be exchanged in the simulation project so that existing TIM tables and sequence entries can continue to be used without any problems.
- Sequences can be automatically started on the basis of a trigger condition.
- Sequences can be temporarily deactivate.
- Decentralized I/O devices can be simulated in the device view.

Motion Control applications

- Pre-configured and integrated connection and configuration of the SINAMICS V90 PN drive message to a Motion Control technology object of SIMATIC S7-1500.
- Automatic data synchronization between the controller and the drive.

See also

Engineering options (Page 60)

Runtime options (Page 62)

2.2.3 SIMATIC WinCC

Configuring screens

- New view on distributed I/Os of the system diagnostic controls.
- Improvements of controls:
  - New scroll bar with configurable colors.
  - Dynamic number of commands in the drop-down menu for recipe control.
  - Expansion of the slider, bar, gauge controls with up to four limits.
- New "Siemens TIA Portal Icons" fonts.
- Graphics library includes new graphics and completely revised structure.

Working with tags

- Symbolic address multiplexing of PlcUDT instances within a ARRAY variable of a data block.
- Hierarchical IntelliSense for connecting PLC data to HMI tags.

SIMATIC Panels and SIMATIC WinCC Runtime Advanced

- Transfer of configuration via USB stick / SD card for Basic Panels 2ndGeneration, Comfort Panels, Outdoor Panels, Mobile Panels and WinCC Runtime Advanced.
- Sm@rt Server Option for Basic Panel 2nd Generation.
- The safety of the following features has been enhanced:
  - The following components are signed:
    Comfort and Mobile Panel: Runtime, ProSave options and Runtime add-ons.
    Basic Panels 2nd Generation: Image of the panel
  - Security of passwords used in Runtime
  - Web browser connection to a Sm@rtServer via HTTPS protocol possible and secure connection from a Web SmartClient to the Sm@rtServer possible.
- The maximum number of power tags in WinCC Runtime Advanced was increased to 16K.

WinCC Runtime Professional

- WebUX option
  Platform and browser-independent operation and monitoring via the Web.
- Process diagnostics option
  When a process fault occurs, process diagnostics provide information about the cause and location of the error and supports troubleshooting.
- Extension: Quantity structure for 1200/1500 PLCs
  There can now be 64 connections to S7-1200/1500.

- Extension: Multitouch for certain controls
  Gestures can now be used to move the display area of controls (trend-based controls, grid-based controls, axis controls).
- Harmonization: Licenses for WEBNAVIGATOR, DATAMONITOR and WebUX
  The license type of Web options can be counted consistently. These SCADA options as of V7.4 or V14 in the future will have uniform MLFBs and therefore there will no longer be separate MLFBs for Runtime Professional.

See also

Engineering options (Page 60)

Runtime options (Page 62)

2.2.4 SINAMICS Startdrive

Simplified commissioning and simplified (remote) servicing

Extended routing functionality for communication between Startdrive and drive:

- Routing across various PROFINET systems
- Routing from PROFIBUS to PROFINET

Simple modification of a project to the machine

Upgrade G120 firmware version in the Startdrive project (beginning with firmware version 4.4)

Reuse of the parameter assignment across various performance ranges.

Also for G120C

Faster access to Smartdrive download package in SIOS

- No waiting time for export control
- Starting with the Startdrive V13 SP1, export restriction no longer applies (export identifier AL=N, ECCN=N)

More efficient configuration

Optimized compilation time of the S7-CPU for networked drives configured with Startdrive

Modification of the supplied S7 data blocks for drive control

Drive library now without know-how protection

Simplified commissioning

Assignment of PROFINET device names is now easy to do in the project navigation and in the HWCN network view via the shortcut menu of the drive.

Simplified update process

Automatic installation of Startdrive HSPs and updates via the TIA Portal Updater

Support for new hardware

New 1FP1 synchronous Reluktanz motors are supported.

No manual update required

HSPs for Sinamics Firmware V4.7.3 and V4.7.6 are integrated.

2.2.5 SIMOTION SCOUT TIA

Description

The SIMOTION P platform (PC-based) is now available in the TIA Portal. (Planned for approx. 3 months after release of STEP 7)

- You can now upload the hardware configuration for SIMOTION devices.
- Additional PROFINET functions for SIMOTION devices:
  - Shared I-device
  - Shared device
  - Seamless Media Redundancy (MRPD)
  - "Check of VendorID/DeviceID" and "Overwriting of the NameOfStation" (automatic configuration of the topology)
  - Address tailoring (modular machines)

2.2.6 SIMOCODE ES V14

Routing via Ethernet - PROFINET

SIMOCODE ES V14 supports routing to SIMOCODE pro devices with PROFINET. Devices can now also be reached from an engineering station with SIMOCODE ES V14 Premium via Ethernet, if they are not directly connected to the PROFINET network to which the SIMOCODE pro devices are connected. This improves the capabilities of the software in systems in which SIMOCODE pro devices are used with PROFINET.

Migration of SIMOCODE ES 2007 group files

Configurations created with SIMOCODE ES 2007 in which the parameter files for SIMOCODE pro devices have been assembled into "group files", can now be migrated directly to one and the same TIA Portal V14 project without additional configuration steps. This function improves the handling of the software when migrating from SIMOCODE ES 2007 to SIMOCODE ES V14 TIA Portal.

Improved editor functions for charts with SIMOCODE pro interconnections

With SIMOCODE ES V14 Standard and Premium, CFC-based charts can be copied in whole or part, saved as library elements and reused. This feature simplifies the handling of recurrent configurations for typical configurations.

Clear presentation of all the important parameter settings in the parameter editor

All important parameter settings for fast commissioning of a motor feeder can be displayed and edited in a new overview in SIMOCODE ES V14, which significantly helps to improve clarity.

Uniform presentation of recorded trends for measured values

Measured values and data recorded online and offline with SIMOCODE ES "Live Trend" and "Analog Value Recording" functions are displayed using the TIA Portal "Trace" function. The operation of this function has been significantly improved thanks to the TIA Portal-wide standardization.

2.2.7 Engineering options

TIA Portal Multiuser

TIA Portal Multiuser Engineering enables several users to simultaneously work together on a project. This reduces configuration times considerably and enables quicker commissioning of projects.

The basic principle:

- An independent server application carries out project management. The server application can be installed independent of a TIA Portal.
  or
  project management is located on a server. The server can be installed independent of a TIA Portal.
- Different users work in local sessions, based on the projects managed by the server.
- Users work independently from each other in local sessions; changes made by other editors are displayed and can be applied.
- The changes from the local sessions are transferred to the server project on check-in.

TIA Portal Teamcenter Gateway

The TIA Portal Teamcenter Gateway allows you to store and manage TIA Portal projects and global libraries in the Teamcenter. The operation is performed in the TIA Portal.

TIA Portal Cloud Connector

The TIA Portal Cloud Connector lets you access local PG/PC interfaces and the SIMATIC hardware connected to them in the TIA Portal engineering even though the engineering itself is operated in a private cloud using Remote Desktop.

SIMATIC Energy Suite

SIMATIC Energy Suite as an integrated option for the TIA Portal for the first time efficiently combines energy management with automation, bringing energy transparency to production. In addition, the configuration work is significantly reduced due to the simplified configuration of components for energy recording.

- Integrated and intuitive configuration of energy management
- Automatic creation of PLC energy program for S7-1500
- Archiving to WinCC Runtime Professional or to a PLC-internal SIMATIC memory card
- Seamless connection to SIMATIC Energy Manager PRO

SIMATIC S7-PLCSIM Advanced

S7-PLCSIM Advanced is suitable for the simulation of an S7-1500 PLC. Compared to standard S7-PLCSIM, S7-PLCSIM Advanced provides the following additional features:

- API for interfacing with co-simulations
- Multiple, distributed instances possible
- Adjustable, virtual time
- Web server and OPC UA access

SIMATIC ODK 1500S

The SIMATIC ODK 1500S enables the development of function libraries with the high-level languages C/C++ for ODK-capable SIMATIC S7-1500 PLCs. These function libraries can be directly loaded and run directly from the user program. ODK-capable S7-1500 PLCs:

- S7-1500 Software Controller (CPU 1507S (F))
- ET 200SP Open Controller (CPU 1515SP PC (F))
- S7-1500 Advanced Controller (CPU 1518(F)-4 PN/DP ODK)

SIMATIC Target 1500S™ for Simulink®

SIMATIC Target 1500S™ for Simulink® is an add-on for Simulink from MathWorks. It enables the integration of Simulink models directly in the program cycle of an ODK-capable S7-1500 PLC.

- Enables model-based development with MATLAB® and Simulink for SIMATIC controllers
- Seamless integration in Simulink as a so-called target file
- No C/C++ or ODK know-how required
- Support of the Simulink external mode for direct monitoring and control of model parameters in the controller from Simulink
- The generated program code can be executed on the CPUs 1507S(F) V2.0, CPU 1515SP PC (F) V2.0 and CPU 1518(F)-4 PN/DP ODK.

SIMATIC Visualization Architect

The SIMATIC Visualization Architect (SiVArc) enables easy, fast and flexible automated production of HMI content based on the STEP 7 user program.

- Supported HMI devices:
  Comfort Panel, Mobile Panel 2nd Generation, WinCC RT Professional, WinCC RT Advanced
- Supported HMI objects:
  Screens, text lists, power tags, captions, buttons, I/O fields, symbolic I/O fields, graphic I/O fields, switches, round buttons
- Configuration via control editor
- Templates are managed through libraries

2.2.8 Runtime options

Description

The following new options are only offered as a runtime option with TIA Portal V14. The products ProDiag and OPC UA S7-1500 are already integrated into the existing products STEP 7 and WinCC and only need to be licensed when used on the real hardware. In comparison with the Engineering options, the runtime options are not version-dependent.

SIMATIC ProDiag

SIMATIC ProDiag enables precise and fast machine and plant diagnostics for SIMATIC S7-1500 and SIMATIC HMI:

- Central cycle-precise time stamp of error messages
- Automatic generation of monitoring logic and message calls
- Automatic update of the SIMATIC HMI when changes are made to message configuration for 3 languages

- HMI systems do not have to leave runtime mode in the event of changes
- Is directly available in the language editors LAD, FBD, SCL and STL.
- Monitoring can be parameterized later on F blocks and know-how protected blocks
- Central project definition of message structure
- Alarm texts are automatically derived from existing information in the project
- Simple visualization of the HMI through prepared controls

SIMATIC OPC UA S7-1500

The OPC UA S7-1500 option allows easily connection of any third-party devices to the S7-1500 via the OPC UA server integrated in the S7-1500 CPU:

- OPC UA server directly in S7-1500
- OPC UA data access, read/write, subscription to value changes
- Security
- XML export for offline configuration of OPC UA clients

SIMATIC Energy Suite S7-1500

10 energy objects are supplied with the SIMATIC Energy Suite Engineering. Additional energy objects can be extended by the Runtime packages.

WinCC / WebUX

WinCC / WebUX allows us to monitor plant processes over the Internet or Intranet using mobile devices and if required, to also control them. The software offers a visualization solution which is suitable for HTML5 and SVG-capable browsers, independent of the operating system – and which does not require engineering work.

Readme

3

3.1 Notes on the TIA Portal

3.1.1 General notes

The information in this readme file supersedes statements made in other documents.

Read the following notes carefully because they include important information for installation and use. Read these notes prior to installation.

Using new functions of service pack 1 in compatibility mode

If you are using a project in compatibility mode and are using a new function of the TIA Portal V14 SP1, the project is automatically upgraded to project version V14 SP1 in most cases.

To use the following functions, however, upgrade the project manually beforehand to project version V14 SP1:

- Assigning various PLC data types (UDTs) with the same structure but different symbolic names
- Selection of the new instructions SCATTER, GATHER, GetSMCinfo and polyline

Safety note for block protection

If a block is protected by a password that was entered manually, manual input is always required to edit the block or its protection settings. This is true even if a password provider contains a password with the same content at a later time.

Upgrading from TIA Portal V13 SP1 projects with master copies from TIA Portal V10.5

Before upgrading a project, check if the master copies created with TIA Portal V10.5 are contained in its project library. This is necessary, for example, when you receive a message during the upgrade about an S7-1200 CPU with firmware version V1.0 used in the project, even though you have not configured such a CPU. The CPU is a master copy in the project library in this case.

Before upgrading a project with master copies from TIA Portal V10.5, therefore, do the following:

1. Use each master copy created with the TIA Portal V10.5 once in the project.
2. Delete the affected master copies from the project library.
3. Create the master copy again with the objects from the project.
4. Save the project in TIA Portal V13 SP1.

You can then upgrade the project to the latest version.

Display of Asian characters in the TIA Portal

Due to a change in behavior in Microsoft Windows, it may occur that texts are not displayed correctly in the TIA Portal when a Chinese TIA Portal is installed on another Asian operating system (e.g. Korean). To view the texts in TIA Portal correctly, open the Windows Control Panel and select "English" under "Language for non-Unicode programs". Note, however, that this may cause display problems in other programs.

Installing new .Net versions or .Net service packs

- Close the TIA Portal before installing a new .Net version or a new .Net service pack on your programming device/PC.
- Restart the TIA Portal only after successful installation of the new .Net version or the new .Net service pack.

Notes on handling

- If a project in the list of projects last used is located on a network drive that is not connected, you may experience delays when opening the "Project" menu.
- When you insert a CPU, you may need to wait for some time if the project editor is open at the same time. This generally takes longer when you insert the first CPU in a newly created project. To be able to continue working at a good pace, you should close the project text editor before inserting a CPU.
- The alarm "Application is not responding" may appear in Windows 7 with functions that take a long time to run (loading the CPU for example). If this occurs, wait until the function has correctly finished.
- If you have installed a Microsoft mouse with IntelliPoint, you may find that it superimposes components over the buttons of the title bar. If this is the case, uninstall the IntelliPoint software from Microsoft.
- Enabling the "Virtual Desktop" options with NVIDIA graphics cards can cause problems. In this case, disable the "nView virtual desktop manager" of your NVIDIA graphics driver.

Working with automatically synchronized network drives

A TIA Portal project consists of multiple files that are saved together in one directory. If you store a project that is automatically synchronized on a network drive or in a cloud directory (e.g. Dropbox, Syncplicity or GoogleDrive), this can lead to data loss if the synchronization only takes place partially or asynchronously. For this reason, we do not recommend editing TIA Portal projects directly on synchronized network drives or in cloud directories. Always close the TIA Portal project before synchronization and make sure that all directories and files from the project directory are synchronized together and completely. You may wish to disable automatic synchronization while working with the TIA Portal. The synchronization itself must be implemented in such a way that the current (local) project data replaces the project data on the network drive or in the cloud directory.

Entry of decimal places

With certain Windows language settings, it may occur that the entry of values with a comma as decimal place is not recognized (entering "1,23" leads to an error). Instead, use the international format ("1.23").

Information on the TIA Portal in online support

Overview of the most important technical information and solutions for the TIA Portal in the Siemens Industry online support.

Internet link: TIA Portal in Siemens Industry online support (https://support.industry.siemens.com/cs/ww/en/view/65601780)

All information on service and support in the Siemens Industry online support:

Internet link: Service and support in Siemens Industry online support (https://support.industry.siemens.com/cs/de/en/)

Here, you can also subscribe to the newsletter that provides you with latest information relating to your products.

Starting the TIA Portal

When you start the TIA Portal, Windows attempts to update the Certificate Revocation List (CRL) of "windowsupdate.com".

If no Internet access is available and there are multiple DNS servers, a timeout may occur and delay the start of the TIA Portal.

3.1.2 Notes on libraries

Contents

Information that could not be included in the online help and important information about product characteristics.

Comparing library elements

If master copies and types have the same name, the corresponding objects in the project are overwritten when the action "Copy" is used. Please note that this takes place without a prompt. The same behavior occurs when the name of the master copy is different to the name of the type, but an object within the master copy has the same name as the type.

Library setting: "Disable editing of the type version number"

You can use the library setting "Disable editing of the type version number" to prevent that the type version number can be assigned by the user for the actions "Release version" and "Assign version". By activating the library setting, the type version number can not be changed by the user. The build number is still incremented automatically when a version is released or assigned. When the high limit of the build number in a block (999) has been reached, the version number is incremented in the next block.

3.1.3 Notes on memory cards

Contents

Information that could not be included in the online help and important information about product characteristics.

Notes on SIMATIC memory cards

The SIMATIC memory cards have been formatted and set up by Siemens for use with S7-1200 and S7-1500 modules. This format must not be overwritten; otherwise, the card will no longer be accepted by the modules. Formatting with Windows tools is therefore not permitted.

Behavior in case of open force job

Note that an active force job is retained even after you have loaded a new project to the SIMATIC memory card. This means you should first delete the active force job before you remove a SIMATIC memory card from the CPU and before you overwrite the card in the PC with a new project. If you use a SIMATIC memory card with unknown content, you should format the SIMATIC memory card before the next download.

Access protection for memory cards in USB card readers

By improving the security mechanisms for online access and engineering of S7-1500 CPUs, the data storage on memory cards has been changed. For this reason, this version of STEP 7 cannot evaluate the passwords of the configured protection level when reading project data from memory cards that is accessed via a USB card reader. The changed response affects the memory cards for CPUs of the S7-1200/1500 series. Therefore, use physical safeguards to protect critical project data on memory cards for these devices.

Note

This restriction is not related to online access to devices or the know-how protection of program blocks.

3.1.4 Notes on the hardware configuration

Content

Information that could not be included in the online help and important information about product characteristics.

Subnet addressing for CP 1613 and CP 1623

CP 1613 and CP 1623 are communication modules with microprocessor. To ensure secure management of communication links, these are processed on the module. The protocol stack in your PC is used for diagnostic purposes (SNMP, DCP). To allow both protocol stacks (i.e. CP 1613/23 Firmware and CP 1613/23 NDIS access) access to the same partners, is recommended to place both stacks of a module in the same subnet.

Editing a device IP address

Do not use the address range from 192.168.x.241 to 192.168.x.250 when editing a device IP address. If necessary, this address range is automatically assigned by the system to a programming device. Depending on the subnet mask, this applies also for all network classes.

Drivers for the CP 5512 communications processor

The drivers for the CP 5512 are no longer included in this software version. If you want to use the CP 5512, the following needs to be installed prior to the installation of this software:

http://support.automation.siemens.com/WW/view/78453460 (http://support.automation.siemens.com/WW/view/en/78453460)

Please note that the CP 5512 can be used with a maximum of Windows 7 32-bit. As of Windows 8 or with 64-bit operating systems, the CP 5512 is not compatible.

Note

Since 2009, the functional successor the USB 2.0 module CP 5711 has been available.

3.1.5 Notes on instructions

Contents

Information that could not be included in the online help and important information about product characteristics.

Instructions not valid for all CPU firmware versions

The firmware version of your CPU determines the following:

- Whether a specific instruction is available for this CPU:
  - If you select CPU 1211C DC/DC/DC with firmware version V1.0, for example, the extended instruction "T_COMBINE" will not be available to you. It is grayed out in the "Instructions" task card.
  - However, if you select CPU 1211C DC/DC/DC with firmware version V3.0, the extended instruction "T_COMBINE" will be available to you in versions V1.1 and V1.2.
- Which versions of the instruction are available. You can select the different versions in the "Version" column of the "Instructions" task card.

3.1.6 Notes on using the TIA Portal in a virtual environment (private cloud)

Contents

Information that could not be included in the online help and important information about product characteristics.

Instructions for using the TIA Portal in a virtual environment (private cloud)

You can find instructions on how to use the TIA Portal in a virtual environment (private cloud) on the installation disk in the directory "Documents\Readme\<language directory>". You can open the PDF document "TIAPortalCloudConnectorHowTo<language ID>.pdf" here.

The TIA Portal Cloud Connector can be used with the Windows 7 (64-bit) and Windows 8.1 (64-bit) operating systems. Using the TIA Portal Cloud Connector with Windows 10 is in development.

Environment variables "TiaUserSettingsPath" and "TiaDefaultProjectPath"

The environment variables "TiaUserSettingsPath" and "TiaDefaultProjectPath" can be used to set the location of the user and project settings different from the system default. This can be used, for example, to save the user and project settings at a central location.

You can find additional information on the use of environment variables in the Instructions on Using the TIA Portal in a Virtual Environment (Private Cloud).

Note on using self-signed certificates when using HTTPS as the communication protocol

Microsoft regularly checks whether the certificates in the Windows Certificate Store are from a trusted source. Certificates from untrusted sources are deleted. Therefore, ensure that your self-signed certificates are signed with a secure certificate.

3.1.7 Information on working with Multiuser Engineering

Contents

Information that could not be included in the online help and important information about product features of Multiuser Engineering and about the multiuser server.

Installation of the multiuser server gives you access to the "Multiuser Engineering" function that is available with TIA Portal V14.

Microsoft system account for multiuser server

When the multiuser server is installed, a Microsoft system account is created for the multiuser server. Microsoft changes the password cyclically for this account.

If the password for this account has expired, the password needs to be changed and then the multiuser server needs to be restarted.

Error message during start of the multiuser server

If an error message is displayed during or after the installation of the multiuser server indicating that the service cannot be started for the multiuser server V14, the following procedure is recommended:

1. Open the Windows event view and check whether an error entry with the error number "1069" is present.
2. If this is the case, information on how to proceed is provided below.
   Internet link: FAQ with ID 109739926 (https://support.industry.siemens.com/cs/ww/en/view/109739926)
3. Alternatively, you can contact Customer Support directly.
   Internet link to Customer Support: TIA Portal in Siemens Industry online support (https://support.industry.siemens.com/cs/ww/en/view/65601780)

Working with active monitoring jobs and forcing jobs in the local session

Active monitoring jobs and forcing jobs must be closed before a check-in or update in the local session. It is not possible to check in or update in the local session while monitoring or forcing jobs are active.

Working in the local session with the "Keep local session" option

If the "Keep local session" option is selected after the check-in, all checked-in objects in the local session appear as outdated, even if their content is identical.

Working with the NTFS file system

Make sure that the Multiuser-server repository is shared only on NTFS partitions.

See also

TIA Portal in Siemens Industry online support (https://support.industry.siemens.com/cs/ww/en/view/65601780)

FAQ with ID 109739926 (https://support.industry.siemens.com/cs/ww/en/view/109739926)

3.2 STEP 7 Basic

3.2.1 Security information

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.

Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.

Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit

http://www.siemens.com/industrialsecurity (http://www.industry.siemens.com/topics/global/en/industrial-security/Pages/Default.aspx)

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under

http://www.siemens.com/industrialsecurity (http://www.industry.siemens.com/topics/global/en/industrial-security/Pages/Default.aspx)

Network settings

The following tables show the network settings of each product you need to analyze the network security and to configure external firewalls:

STEP 7 Basic					
Name	**Port number**	**Transport protocol**	**Direction**	**Function**	**Description**
ALM	4410*	TCP	Inbound/outbound	License service	This service provides the complete functionality for software licenses and is used by both the Automation License Manager as well as all license-related software products.
RFC 1006	102	TCP	Outbound	S7 communication	Communication to the S7 controller via Ethernet/PROFINET for programming and diagnostic purposes.

STEP 7 Basic					
DCP	---	Ethernet	Outbound	PROFINET	The DCP protocol (Discovery and Basic Configuration Protocol) is used by PROFINET and provides the basic functionality for locating and configuring PROFINET devices.
SNMP	161	UDP	Outbound	PROFINET	The SNMP client functionality is used by STEP 7 to read status information from PROFINET devices.
* Default port that can be changed by user configuration					

WinCC ES Basic (without simulation)					
Name	**Port number**	**Transport protocol**	**Direction**	**Function**	**Description**
ALM	4410*	TCP	Inbound/outbound	License service	This service provides the complete functionality for software licenses and is used by both the Automation License Manager as well as all license-related software products.
HMI Load	1033	TCP	Outbound	HMI Load (RT Basic)	This service is used to transmit images and configuration data to Basic Panels.
* Default port that can be changed by user configuration					

Simulation RT Basic					
Name	**Port number**	**Transport protocol**	**Direction**	**Function**	**Description**
HMI Load	1033	TCP	Inbound	HMI Load (RT Basic)	This service is used to transmit images and configuration data to Basic Panels.
Ethernet/IP	44818	TCP	Outbound	Ethernet/IP channel	The Ethernet/IP protocol is used for connections to Allen Bradley PLCs.
	2222	UDP	Inbound	Ethernet/IP channel	The Ethernet/IP protocol is used for connections to Allen Bradley PLCs.
Modbus TCP	502	TCP	Outbound	Modbus TCP channel	The Modbus TCP protocol is used for connections to Schneider PLCs.
RFC 1006	102	TCP	Outbound	S7 channel	Communication to the S7 controller via Ethernet/PROFINET
Mitsubishi MC	5002	TCP	Outbound	Mitsubishi MC channel	The Mitsubishi protocol is used for connections to Mitsubishi PLCs.

3.2.2 Notes on use

Contents

Information that could not be included in the online help and important information about product characteristics.

Online operation

Simultaneous online operation of STEP 7 V5.5 or earlier versions and STEP 7 Basic V14 is not approved.

Simultaneous online connections on an S7-1200 CPU

It is not possible to establish an online connection from multiple instances of the TIA Portal simultaneously to the same S7-1200 CPU.

Removing/inserting the memory card

After removing or inserting a memory card, always perform a memory reset on the CPU in order to restore the CPU to a functional condition.

Removing and inserting Ethernet modules

If Ethernet modules are removed and re-inserted during operation, you must boot the PC; otherwise, the "Accessible devices" functionality in STEP 7 or NCM PC will not display all devices. While the PC boots, Ethernet modules must be activated.

Loading project data with TIA Portal V12 and V13 (S7-1200)

If you load the project data of an S7-1200 CPU with the TIA Portal V13, you can no longer use TIA Portal V12 to access this data. To do this, first restore the factory settings of the CPU. Read the additional information on this in the online help under "How to reset a CPU to factory settings".

Compatibility

The device configuration and program of an S7-1200 CPU must always be configured with the same STEP 7 version. Usually, the TIA Portal makes sure that no version conflicts occur by outputting appropriate notifications during loading to the device.

This automatic verification is not possible with S7-1200 CPUs with firmware version V1.x. In this case, users themselves must ensure that no version conflicts occur.

3.2.3 Editing devices and networks

3.2.3.1 General information on devices and networks

Contents

Information that could not be included in the online help and important information about product characteristics.

S7 PCT IO-Link

The S7 Port Configuration Tool is available for free download at the following link.

http://support.automation.siemens.com/WW/view/37936752 (https://support.industry.siemens.com/cs/ww/en/view/32469496)

3.2.3.2 Use of modules on the S7-1200

Content

Information that could not be included in the online help and important information about product characteristics.

Use of modules on the S7-1200

The modules listed below are not supported on the S7-1200.

Family	Module	Order number
S7-300 FMs	SM 338	6ES7 338-4BC01-0AB0
	FM 350-1	6ES7 350-1AH03-0AE0
	FM 350-2	6ES7 350-2AH00-0AE0, 6ES7 350-2AH01-0AE0
	FM 351	6ES7 351-1AH01-0AE0, 6ES7 351-1AH02-0AE0
	FM 352	6ES7 352-1AH02-0AE0
	FM 355 S	6ES7 355-1VH10-0AE0
	FM 355 C	6ES7 355-0VH10-0AE0
	FM 355-2 C	6ES7 355-2CH00-0AE0
	FM 355-2 S	6ES7 355-2SH00-0AE0
S7-300 PtP-CP	CP 340	6ES7 340-1AH02-0AE0, 6ES7 340-1BH02-0AE0, 6ES7 340-1CH02-0AE0
	CP 341	6ES7 341-1AH01-0AE0, 6ES7 341-1AH02-0AE0, 6ES7 341-1BH01-0AE0, 6ES7 341-1BH02-0AE0, 6ES7 341-1CH01-0AE0, 6ES7 341-1CH02-0AE0
Network component	Diagnostics repeater	6ES7 972-0AB01-0XA0
ET 200S	1 Count 24 V	6ES7 138-4DA04-0AB0
	1 Count 5 V	6ES7 138-4DE02-0AB0
	1 Step 5 V	6ES7 138-4DC00-0AB0, 6ES7 138-4DC01-0AB0
	2 pulses	6ES7 138-4DD00-0AB0, 6ES7 138-4DD01-0AB0
	1 SI	6ES7 138-4DF01-0AB0
	1 SI Modbus	6ES7 138-4DF11-0AB0
	1 SSI	6ES7 138-4DB02-0AB0, 6ES7 138-4DB03-0AB0
	1 Pos Universal	6ES7 138-4DL00-0AB0
	SIWAREX	7MH4910-0AA01, 7MH4912-0AA01, 7MH4920-0AA01
ET 200M	SIWAREX	7MH4 900-2AA01, 7MH4 900-3AA01, 7MH4 950-1AA01, 7MH4 950-2AA01

Loading S7-1200 module comments to the PG/PC

In central configurations with S7-1200, comments of modules, submodules and signal boards are not loaded. With CPs/CMs, only the comments of the IE interface or DP interface are loaded. In distributed configurations with ET 200SP or ET 200MP, only the comment of the channels is loaded from the I/O modules.

3.2.3.3 CP 343-2 on SIMATIC S7 Embedded Controller EC31-RTX

Contents

Information that could not be included in the online help and important information about product characteristics.

CP 343-2 on SIMATIC S7 Embedded Controller EC31-RTX

The module AS-Interface CP 343-2 (article no.: 6GK7 343-2AH01) can be inserted in an expansion rack of the SIMATIC S7 Embedded Controller EC31-RTX (article no.: 6ES7 677-1DDxx-0BB0), but the CP 343-2 cannot be operated with the EC31-RTX.

3.2.3.4 F-CM AS-i Safety ST for ET 200SP

Content

Information that could not be included in the online help and important information about product characteristics.

F-CM AS-i Safety ST (3RK7 136-6SC00-0BC1) from HSP0070

When using an F-CM AS-i Safety ST (3RK7 136-6SC00-0BC1) using HSP0070 in TIA Portal V13.0, after updating to Service Pack 1, the HSP0070 (F-CM AS-i Safety ST) must be updated to version V2.0. This is necessary to allow use of the module in TIA Portal V13 Service Pack 1.

3.2.3.5 S7 routing via IE/PB Link

Content

Information that could not be included in the online help and important information about product characteristics.

S7 routing via IE/PB Link

In the following situations, no routing via the IE/PB Link PN IO is possible:

- S7 routing between two CPUs of the S7-1500
- S7 routing of PG connections to CPUs of the S7-1200/1500
- S7 routing of HMI connections to CPUs of the S7-1200/1500

This behavior relates to the IE/PB Link with firmware version V2.1 (6GK1 411-5AB00).

3.2.3.6 Notes on online and diagnostics

Contents

Information that could not be included in the online help and important information about product characteristics.

Hardware detection followed by online connection

When the "Online > Hardware detection" command is performed for an unspecified CPU, the online configuration is not loaded from the CPU. If you do not load the configuration resulting from the hardware detection to the CPU, the device and network views will always show a difference between the offline and online configurations. It will appear that there are different configurations in the online and diagnostic views, although the MLFBs are identical in the actual CPU and the offline CPU.

Transferring numerous and large trace measurements from the device (memory card)

Transferring trace measurements from the device to the project increases the memory requirement.

Avoid copying a large number of measurements with large amounts of data at the same time lead to high memory consumption and extended periods needed for copying.

Deleting measurements in the device (memory card) (S7-1200)

Do not delete any measurements in the device as long as a trace in the device still generates new measurements on the memory card.

3.2.3.7 Network components

Network components (STEP 7 Basic)

Security modules

Migration of projects with Ethernet CPs and activated security functions

In STEP 7 V5.5 projects that contain Industrial Ethernet CPs with activated security functions, the security settings are disabled during migration to STEP 7 Professional.

If necessary, follow these steps after the migration:

1. Activate the security functionality.
2. Configure the required security settings.

Migration of IP access protection lists when activating the security functions

An active IP access protection list is converted to firewall rules when security functions are activated. These rules are visible in advanced firewall mode and can be adjusted there. The advanced firewall mode is activated automatically.

Security online diagnostics of S7 CPs

Security online diagnostics of a CP with security capability is only possible if the online connection is established directly via the CP. If the online connection was established by STEP 7 via the CPU to the station, with the "Connect online" button on the security diagnostics page "Security" > "Status" of the security CP you can establish a direct connection to the CP to run security online diagnostics. As an alternative, you can also terminate the online connection to the CPU and enter the IP address of the CP in the "Station address" input box under the entry "Online access" in online diagnostics.

IKE mode

When negotiating the key in phase 1 the IKE mode "Main" should be preferred. This mode normally provides a reliable procedure compared with the "Aggressive" mode. One reason for using the "Aggressive" setting is when you have VPN groups with different pre-shared keys.

The IKE "Aggressive" mode should not be used in conjunction with certificates. In the IKE "Aggressive" mode, only use pre-shared keys.

A security module must not be used in VPN groups that use different IKE modes.

Downloading configuration data to an S7-300/400 via a VPN tunnel

When you download configuration data via the gigabit interface of a CP x43-1 Advanced to an S7-300/S7-400 station, the path via which the download takes place is stored in the project. If the project is then downloaded via a VPN tunnel established between a SCALANCE S module and the CP x43-1 Advanced, the download fails due to the changed path.

To download via the VPN tunnel, follow the steps below:

1. Using the "Go online" button, connect the engineering station to the gigabit interface of the CP x43-1 Advanced.
2. Disconnect the online connection to the CP x43-1 Advanced.
3. Download the project to the station via the gigabit interface of the CP x43-1 Advanced.

VPN tunnel establishment with 1200/1500 CPs capable of VPN

The establishment of VPN tunnel connections by 1200/1500 CPs capable of VPN with the authentication method "pre-shared key" is only possible if the VPN connection partner is also a 1200/1500 CP capable of VPN. The establishment of VPN tunnel connections by a 1200/1500 CP capable of VPN to all other VPN connection partners is only possible with the "certificate" authentication method.

Security configurations from old projects

Before a security configuration from an older STEP 7 version in STEP 7 V14 can be used, a user with administrator rights needs to log in to this security configuration in STEP 7 V13 SP1 and save the project. After opening the project in STEP 7 V14, a user must log on again to the security configuration with administrator rights. Thereafter, it is possible to use this security configuration in STEP 7 V14.

Copying security modules between STEP 7 instances

Copying security modules between different instances of STEP 7 is not supported.

Device certificates for CP 1543-1 V2.0 and CP 1543SP-1

For CP 1543-1 V2.0 and CP 1543SP-1, only those device certificates that are assigned to the module via the local certificate manager as device certificates are loaded to the module. This assignment takes place in the local security settings of the module in the "Certificate manager" entry via the "Device certificates" table editor. The certificates of the global certificate manager are available for the certificate assignment.

No fallback VPN profiles for CP 1200/1500

Fallback VPN profiles are not supported by the CP 1200/1500. Configured fallback VPN profiles are therefore not loaded on the CP 1200/1500.

Media converters

When you migrate a project, you need to reconfigure the media converters.

3.2.4 Programming a PLC

3.2.4.1 General notes on PLC programming

Contents

Information that could not be included in the online help and important information about product characteristics.

Using the "TDiag_StatusExt" system data type

Alternatively to the "TDiag_Status" system data type, the extended system data type "TDiag_StatusExt" can be used to obtain more detailed information about the cause of a connection break. In particular, the reading out of the structure element "LastConnErrTimeStamp" causes the CPU to switch to "Defect" system state and must not be used currently.

Upgrading know-how-protected blocks

Know-how protected blocks that were set up with a version <V14 can be loaded without entering the password in the controller and can be run after upgrading from V13 SP1 to V14. To edit a know-how protected block in V14, remove the know-how protection and reset it when done. The block can then be renumbered and renamed in the know-how-protected state (S7-1200 / S7-1500).

- For know-how protected blocks that were set up with V13 SP1:
  - To edit a know-how protected block in V14, remove the know-how protection and reset it when done. The block can then be renumbered and renamed in know-how-protected state.
- For know-how protected blocks that were set up PRIOR TO V13 SP1:
  - To edit, remove the know-how protection in V13 SP1 and reset it when done.

Information about network security

For communications access between the TIA Portal and CPU or between HMI (except for HMI access using "GET/PUT communication") and CPU, there are integrated security functions. These provide greater protection from manipulation and higher access protection. To protect against unauthorized network access to a CPU with standardized communications access such as "GET/PUT", "TSEND/TRCV", "Modbus", "FETCH/WRITE", you should also take suitable additional measures (e.g. cell protection concept).

Online/offline differences in the project tree (S7-1200 FW V2.0 and V2.1)

If you use the instruction "WRIT_DBL: Write to data block in the load memory" to change a data block, the resulting difference between online and offline block is initially not displayed correctly by the symbols in the project tree. The difference is only shown when you terminate an online connection and then go back online.

Loading inconsistent programs to a device

In TIA Portal, it is not possible to load inconsistent programs to a device without a consistency check. During the loading process, all blocks of the program are implicitly checked and are compiled again in the event of inconsistencies. If, however, there are programs on your CPU which were loaded with earlier versions of STEP 7, these programs could demonstrate inconsistencies.

In this case, note the following:

If you load an inconsistent program from a device, you will not be able to load the program unchanged to the device afterwards, because a consistency check always takes place during the loading process and existing inconsistencies are corrected.

Changed access rights for displaying the comparison status

Valid for:

- **CPU 1500 < V2.0 (as well as ET200 SP and Software Controller)**
- **CPU 1200 V4.0 and V4.1**

As of TIA Portal V14, the required access rights for displaying the comparison status of blocks have changed for the specified CPUs from the S7-1200/1500 series.

Previously, you only needed the "HMI access" access level for the listed CPUs to display the comparison status. As of V14, you need the "read access" access level to display the comparison status.

This change means that you are prompted to enter the password for "read access" when you go online.

If you only have the password for "HMI access", you can still establish an online connection.

In this case, click "Cancel" in the password prompt for read access and enter the password for "HMI access" in the subsequent dialog. The online connection is then established with the available "HMI access rights", but no comparison status of the blocks is displayed.

Only questions marks are displayed in place of the icons for the comparison status.

Valid for:

- **CPU 1200 <=V3.x.**

As of TIA Portal V14, there is now a password prompt for the above specified CPUs when you go online for displaying the comparison status of blocks, provided that these CPUs are configured with write protection and/or read protection.

If you do not know the password for access to read-protected blocks on a protected CPU, you can still establish an online connection.

In this case, leave the displayed password prompt dialog with "Cancel".

The online connection is still established, but no comparison status of the blocks is displayed.

Only questions marks are displayed in place of the icons for the comparison status.

Data exchange between standard and F-program when using Team Engineering

To ensure data consistency between standard and safety program, you should:

• Not exchange data via bit memory, but rather using global data blocks.

• Limit access between safety programs and standard user programs to two standard data blocks.

You can find information on this in the section 5.13 of the Programming Guidelines under: https://support.industry.siemens.com (https://support.industry.siemens.com/cs/document/81318674/programming-guideline-and-programming-styleguide-for-s7-1200-and-s7-1500?dti=0&lc=en-WW)

Avoid using PLC data types generated by the system in libraries

Some instructions generate their own PLC data types during instancing which are saved in the "PLC data types" project folder. However, you should not use these system-generated PLC data types in any library, because they may be recreated by the system at any time and may result in an unfavorable system behavior.

See also

https://support.industry.siemens.com (https://support.industry.siemens.com/cs/document/81318674/programming-guideline-and-programming-styleguide-for-s7-1200-and-s7-1500?dti=0&lc=en-WW)

3.2.4.2 Instructions

Contents

Information that could not be included in the online help and important information about product characteristics.

Optimizing the run time of instructions such as "Serialize", "Deserialize", "CMP", which can process structures

A series of instructions (e.g. "Serialize: Serialization", "Deserialize: Deserialization", "CMP" (comparator) and "MOVE: Copy value") can process very large, complex structured tags. In doing so, the CPU analyzes the form of the tag structure and executes the corresponding instruction for each substructure contained in the total structure or for all contained elementary components.

With a very complex structure, this structure analysis may lead to an unexpected increase in the run time of the corresponding instruction. In addition to the complexity of structured tags specified in the operation, the total number of anonymous structures declared in the program also has an effect on the run time. A very large number of different anonymous structure definitions can also increase the run time.

Solution:

- Avoid anonymous structures These are structures that are defined directly on the tag with the keyword "Struct" and not through a user-defined data type (UDT).
- Avoid multiple declaration of data structures that are structured very similarly. Try to assemble these into a structure declaration.
- Avoid the declaration of numerous individual tags in structures and data blocks, if they have the same data type and represent an array of the function. In this case, you should use the ARRAY data type if possible.

Instruction "Get_IM_Data: Reading identification and maintenance data"

Currently, the "Get_IM_Data" instruction outputs the error code 16#0000 when a string that is too short is used at the DATA parameter.

The 16#8752 error code would be correct. You can find the corresponding description of the 16#8752 error code in the description of the instruction in the information system.

3.2.4.3 Testing the user program

Testing with the watch table

Contents

Information that could not be included in the online help and important information about product characteristics.

Multiple access to the same CPU

Access to a CPU from a PG/PC is permitted only when a TIA Portal is open. Multiple access to the same CPU is not permitted and can lead to errors.

Loading data blocks during an active control job

Note

Loading changed data blocks during an active control job can result in unforeseen operating states. The control job continues to control the specified address, although the address assignment may have changed in the data block. Complete active control jobs before loading data blocks.

Testing programs converted from older STEP 7 versions

To monitor and test a program converted from an older STEP 7 version, you first need to compile and download the program with the current STEP 7 version.

Changed display formats in the watch table

The number of display formats has been reduced so that display formats that are used but no longer supported are reset to the defaults during a project upgrade.

Example: Pointers can now only be shown in "Pointer" or "Hexadecimal" format.

Testing with the force table

Contents

Information that could not be included in the online help and important information about product characteristics.

Example

If I/O access to the address "IB0:P" takes place in the user program, it is not permitted to force the following I/O address areas: I0.0:P, IB0:P, IW0:P and ID0:P.

Changed display formats in the force table

The number of display formats has been reduced so that display formats that are used but no longer supported are reset to the defaults during a project upgrade.

Example: Pointers can now only be shown in "Pointer" or "Hexadecimal" format.

3.2.4.4 Compatibility of PLC programs from older versions

Compatibility of PLC programs from versions prior to V14.1

Content

Information that could not be included in the online help and important information about product characteristics.

Compatibility

In V14 SP1 you can generally continue to use all programs that were created with older versions of the TIA Portal. However, because improvements were made to the compiler and errors corrected there in V14 SP1, it can occur in rare cases that the program reacts differently after the upgrade or that you have to adjust the program code manually. These cases are described in detail below.

In addition, you have the option of editing the project in compatibility mode. You can find additional information on the compatibility under "Compatibility of projects".

Program loops in SCL

In S7-1200/S7-1500 it is not possible to change the index of a program loop from within the loop with SCL.

The following program example would therefore be invalid:

```
FOR #i := 1 TO 10 DO
#i := #i + 1;
END_FOR;
```

As of TIA Portal V14 SP1, a more exact syntax check is conducted in this regard during compilation and a syntax error is output.

CASE statements in SCL

Labels without associated "GOTO" instruction are no longer permitted in a CASE statement. A syntax error is signaled.

ARRAYs of multi-instances

In some cases ARRAYs of multi-instances could not be correctly be addressed in V14 when individual tags within a multi-instance had the retentivity setting "Set in IDB". This behavior has been corrected in V14 SP1.

If you have used such tags in your program, you may have to recompile the affected block after upgrading.

Parameter type "Block_DB" for entry of the instance DB

In LAD and FBD, you cannot enter the instance DB of an instruction using an input of the data type "BLOCK_DB". This behavior has been made uniform for all CPU families and all block types.

As of TIA Portal V14 SP1, a syntax error is output.

If you have transferred instance DBs in your program with the help of the "Block_DB" data type, you change your program. Instead use a parameter instance to transfer the instance during runtime.

Compatibility of PLC programs from versions prior to V14

Content

Information that could not be included in the online help and important information about product characteristics.

Compatibility

In principle, you can continue to use in V14 all PLC programs that were created with older versions of the TIA Portal. However, because improvements were made to the compiler and errors corrected there in V14, it can occur in rare cases that the program reacts differently after the upgrade or that you have to adjust the program code manually. These cases are described in detail below.

Representation of the BCD format

The representation of the BCD format has changed in regard to the sign from TIA Portal V13 SP1 to TIA Portal V14.

The values in BCD format are show without signs in version V13 SP1.

As of version V14, the values in BCD format are show with sign. This can lead to an altered representation of the values in BCD format.

Integers (decimal system)	Hexadecimal numbers	Representation in V13 SP1	Representation in V14
0	16#0000	BCD#0	BCD#0
-26215	16#9999	BCD#9999	BCD#-999
1365	16#0555	BCD#555	BCD#555
21845	16#5555	BCD#5555	BCD#555
4096	16#1000	BCD#1000	BCD#0
-28672	16#9000	BCD#9000	BCD#0

Instructions "SET_BF: Set bit field" and "RESET_BF: Reset bit field"

The response of the instruction in regard to the structures has changed from TIA Portal V13 SP1 to TIA Portal V14.

In version V13 SP1, the number of bits that set or reset are always indicated at the input. This also applies, for example, if you have specified a structure of data type of the STRUCT or ARRAY PLC data type. If you want, for example, you want set or reset 10 bits and the structure contains only 5 bits, then the following 5 bits within the address sequence is also set or reset.

With structures of the type PLC data type, in version V14 STRUCT or ARRAY the number of bits contained in the structure represents the maximum number of bits that can be reset: If you specify the value "20", for example, and the structure only contains 10 bits, only these 10 bits are set. If you specify the value "5", for example, and the structure contains 10 bits, then exactly 5 bits are set.

Unused bits of PLC data types (UDT) with firmware >= V1.8.1

The unused bits of PLC data types in standard memory areas are occupied or overwritten, for example, for a PLC data type that contains 4 bits.

With firmware versions < V1.8.1, you could not use the unused bits of a PLC data type elsewhere.

With firmware version >= V1.8.1, all bits are occupied or overwritten even if only 4 bits are used.

Note

Address assignments

Ensure you do not assign the same absolute address twice to different symbolic address assignments.

Explicit data type conversion in SCL (S7-1200) with firmware >= V4.2

With firmware versions < V4.2, the string was transferred aligned to the right and filled with leading spaces during explicit data type conversion of SINT/INT/DINT/REAL_TO_STRING/ WSTRING in SCL.

Example: REAL_TO_WSTRING(12) = ' 1.200000E+1'

As of TIA Portal V13, the string is displayed with a leading sign during explicit data type conversion of SINT/INT/DINT/REAL_TO_STRING/WSTRING in SCL and transferred aligned to the left.

Example: REAL_TO_WSTRING(12) = '+1.200000E+1'

SCL: EN/ENO mechanism with block parameters of data type (W)STRING (S7-1200/1500)

As of TIA Portal V14, it is checked whether block parameters of data type (W)STRING are truncated when parameters are transferred during runtime. This can occur if formal and actual parameters have different declared lengths. If the declared length of the target parameter during runtime is insufficient to accept the (W)STRING, the (W)STRING is truncated and the enable output ENO is set to "FALSE".

If you evaluate the enable output ENO in your program, the semantics of your program may change.

Forwarding STRING parameters between optimized blocks and standard blocks (S7-1200/1500)

Prior to TIA Portal V14, no length information about STRINGs was transferred when parameters were transferred between optimized and non-optimized blocks. This meant that access errors not detected by the ENO mechanism could occur during the further processing of the STRING. The enable output ENO remained at TRUE despite an access error.

As of TIA Portal V14, the STRING length information is transferred and access errors are monitored by the ENO mechanism. If you evaluate the enable output ENO in your program, the semantics of your program may change.

Length declaration for constants of the "STRING" data type

Length declarations are no longer permitted for local and global STRING constants. If your program includes length declarations, for example, "MyStringConst [7]", these are automatically removed during the upgrade. This change has no semantic effect on your program, however.

Changing the base data type for HW_ANY from WORD to UINT

The hardware data type "HW_ANY" is used to identify any hardware component, such as a module. In V14, the base data type of "HW_ANY" was changed from WORD to UINT. After the conversion to V14, compiling errors can therefore occur if you have assigned an explicit WORD constant (such as W#16#1) or a tag of data type WORD at a parameter of data type "HW_ANY".

In this case, change the notation of the constant (e.g. to 16#1) or change the data type of the tag to UINT.

Length of block comments

As of TIA Portal V14, the block comments must not exceed a length of 32767 Unicode characters.

Syntax check of the instructions "SR:Set/reset flip-flop" and "RS:Reset/set flip-flop"

Inputs, outputs, bit memories as well as static or local data can be used as operands at the "S" input. Use of constants is not permitted. So far, the use of the constants "0" and "1" was not rejected by the syntax check. The syntax check was improved in TIA Portal V14 so that the use of "0" or "1" is now rejected at the "S" input. You may now receive an error message during the compiling of a program that has been compiled without errors up to this point. Change your program in this case and use a permitted operand.

Any pointer: Access to optimized data

For a CPU of the S7-1500 series, the ANY pointer can also only point to memory areas with "Standard" access mode. Access to optimized data is not permitted with an ANY pointer.

A more detailed syntax check of ANY pointers is performed as of TIA Portal V14. Access to optimized data is now rejected. You may now receive an error message during the compiling of a program that has been compiled without errors up to this point.

"(L)REAL" data type

With 64-bit Windows versions, the LREAL data type is sometimes evaluated differently than with the previous 32-bit versions of Windows. A difference may occur when rounding and (L) real values of constants in the least significant bit.

The potential difference is in the last digit of the mantissa and in most cases therefore only has a slight effect on the accuracy.

If you evaluate the last digit of the mantissa, the semantics of your STEP 7 program may change after the conversion.

"ST" parameter for IEC timers

The "ST" parameter of an IEC timer is used internally and it is prohibited to write it. This why this parameter is no longer visible as of TIA Portal V14. If you have write access in your program to the "ST" parameter, an error message is generated during compiling after the upgrade.

If you exchange data with Inter Project Engineering and you have access to the "ST" parameter with HMI, you must first export the project and then re-import it into the proxy CPU. Only then does the "ST" parameter disappear in the HMI configuration.

Implicit data type conversion for block parameters of the "DB_ANY" data type

When accessing a block parameter of the "DB_ANY" data type with the syntax DB_ANY. %DB(B|W|D), no implicit data type conversion was performed previously. The bit pattern of the source value was simply converted into the target data type. If the target data type was smaller than the source data type, an overflow occurred and the written value may have been wrong or inaccurate. The enable output ENO is not set to "FALSE" when accuracy is lost or runtime errors occur.

As of V14, conversion is performed according to the rules of implicit data type conversion when parameters are passed with the syntax DB_ANY.%DB(B|W|D). Note that after the upgrade a different value may be written and the semantics of your program may change. In addition, the enable output ENO is set to "FALSE" when a runtime error occurs.

Interface of organization blocks with standard access

The interface of organization blocks with standard access must have a minimum size of 20 bytes. In older versions of the TIA Portal, only the interface of OB1 was checked with regard to the minimum size during the compilation run. As of V14, interfaces of all organization blocks are checked. You may now receive an error message during the compiling of a program that has been compiled without errors up to this point.

Compatibility of PLC programs from versions prior to V13.1

Content

Information that could not be included in the online help and important information about product characteristics.

Compatibility

In principle, you can continue to use in V13 SP1 all programs that were created with older versions of the TIA Portal. However, because improvements were made to the compiler and errors corrected there in V13 SP1, it can occur in rare cases that the program reacts differently after the upgrade or that you have to adjust the program code manually. These cases are described in detail below.

Unused bits of PLC data types (UDT) with firmware >= V1.8.1

The unused bits of PLC data types in standard memory areas are occupied or overwritten, for example, for a PLC data type that contains 4 bits.

With firmware versions < V1.8.1, you could not use the unused bits of a PLC data type elsewhere.

With firmware version >= V1.8.1, all bits are occupied or overwritten even if only 4 bits are used.

Note

Address assignments

Ensure you do not assign the same absolute address twice to different symbolic address assignments.

Explicit data type conversion in SCL (S7-1200) with firmware >= V4.2

With firmware versions < V4.2, the string was transferred aligned to the right and filled with leading spaces during explicit data type conversion of SINT/INT/DINT/REAL_TO_STRING/ WSTRING in SCL.

Example: REAL_TO_WSTRING(12) = ' 1.200000E+1'

As of TIA Portal V13, the string is displayed with a leading sign during explicit data type conversion of SINT/INT/DINT/REAL_TO_STRING/WSTRING in SCL and transferred aligned to the left.

Example: REAL_TO_WSTRING(12) = '+1.200000E+1'

Reading an invalid peripheral input

In the TIA Portal as of version 12, the error code 16#2942 for the reading of an invalid peripheral input is only output if you have programmed the faulty access in such a way that this becomes effective, for example %MW10 := “InvalidWordAccess":P, and does not lie within an irrelevant sequence. For example, this would be #tmp := “InvalidWordAccess":P, if the tag #tmp is not used in the rest of the program block.

Instruction "S_CONV: Convert character string"

The EN/ENO mechanism behaves differently in TIA Portal V13 SP1 than in TIA Portal V13.

In version V13, the ENO enable output returns the signal state "0" in case of error, even if you have deactivated the ENO enable output. If you have switched an additional instruction to the EN enable output, this is then not executed.

In version V13 SP1, the ENO enable output returns the signal state "1" in case of error, if you have deactivated the ENO enable output. If you have switched an additional instruction to the EN enable output, this is then executed as expected.

Instruction "MUX: Multiplex" (SCL)

Up to and including TIA Portal V13, the value of the tag at the input parameter was output unchanged as a function value, even if the K parameter had a negative integer. This behavior has changed in TIA Portal V13 SP1.

In TIA Portal V13 SP1, if you use a tag with a valid data type at the input parameters of the MUX instruction and the K parameter is a negative integer, the value of the tag is changed. You can find the valid data types in the description of the MUX instruction in the information system.

Instruction "DEMUX: Demultiplex" (SCL)

Up to and including TIA Portal V13, no value was output at the OUTELSE output parameter if the value of the K parameter was < 0. In contrast, the value of the IN input parameter was output at the OUTELSE output parameter if the value of the K parameter was > available outputs. This behavior has changed in TIA Portal V13 SP1.

In the TIA Portal V13 SP1, if you specify at the K parameter a value that is outside the available outputs (K < 0 or K > available outputs), then the value of the IN input parameter is output at the OUTELSE output parameter.

Instructions "READ_DBL: Read from data block in the load memory" and "WRIT_DBL: Write to data block in the load memory"

TIA Portal V13 SP1 reports a compilation error if you use "READ_DBL" V1.0 or "WRIT_DBL" V1.0 and access a data block with the attribute "Only store in load memory" using the parameter DSTBLK.

Instead, use version 1.2 of both instructions.

"READ_DBL instructions: Read from data block in the load memory" and "WRIT_DBL: Write to data block in the load memory" (SCL)

As of TIA Portal V13 SP1, the "READ_DBL" and "WRIT_DBL" instruction in SCL are subject to the same strict syntax rules as all other programming languages:

In TIA Portal V13, it was possible to specify "standard" data blocks at the SRCBLK and DSTBLK parameters that contained tags with different data types.

As of TIA Portal V13 SP1, the individual tags contained must have the same data type for both "standard" as well as for "optimized" data blocks; otherwise, compiling errors are reported.

Function values (Return)

As of TIA Portal V13 SP1, more stringent syntax rules apply to function call:

It is checked that the function value (Return) is written in any case, even if multiple possible program paths can be run through in the function. Therefore, there is no longer any risk that the function values will accidentally not be written during runtime.

However, you may possibly receive syntax errors in functions during compiling, which it was still possible to compile without errors in V13. In such cases change the program code so that the function valve is written in all possible program paths.

Example:

SCL

```
IF #MyIn1 = #MyIn2 THEN
   #Block_3 := #MyIn1 + 1;
END_IF;
```

In this example, the function value of "Block_3" is not written if the condition of the IF instruction is not fulfilled. The function value then contains an undefined value.

SCL

```
#Block_3 := #MyIn1;
IF #MyIn1 = #MyIn2 THEN
   #Block_3 := #MyIn1 + 1;
END_IF;
```

In this example the function valve of "Block_3" is definitely written in the program, since "MyIn1" is set as default before the IF loop is completed.

Comparing the hardware data types HW_IO and HW_DEVICE

As of TIA Portal V13 SP1, there is a stricter syntax rule that is valid for the comparison of the data types HW_IO and HW_DEVICE:

Up until and including TIA Portal V13 it was possible to compare the data types HW_IO and HW_DEVICE directly with each other.

If you want to compare these data types in TIA Portal V13 SP1 you have to first create a tag of the data type HW_ANY in the section "Temp" for the block interface and then copy the LADDR (from data type HW_DEVICE) to the tag. It is then possible to compare HW_ANY and HW_IO.

EN/ENO mechanism for STRING conversion

Conversion	Description
Strg_TO_Chars: Convert character string to Array of CHAR	The ENO enable output returns the signal state "0" even if you have deactivated the ENO enable output: • For an invalid character at the CHARS parameter • For an invalid ARRAY index at the PCHARS parameter • If the sum of the PCHARS and STRG parameters exceeds the length of the target ARRAY.
Chars_TO_Strg: Convert Array of CHAR to character string	The ENO enable output returns the signal state "0" even if you have deactivated the ENO enable output: • If the sum of the PCHARS and CNT parameters exceeds the length of the source ARRAY.

3.2.5 Inter Project Engineering (IPE)

3.2.5.1 Notes on IPE

Contents

Information that could not be included in the online help and important information about product characteristics.

Using system diagnostics in the device proxy

To use the "system diagnostics" function in an IPE device proxy, for example, a system diagnostics view, insert the PLC alarms as content of a device proxy.

3.2.6 Technological functions

3.2.6.1 Notes on technological functions (S7-1200)

Parameters of the technology objects in TIA Portal Openness

You can find a list of the available technology object parameters in the product information "Parameters of the technology objects in TIA Portal Openness" on the internet (https://support.industry.siemens.com/cs/ww/en/view/109744932).

S7-1200 Motion Control - variable <axis name>.StatusBits.NonPositionControlled

Contrary to the information in the online help, the content of the variable <axis name>.StatusBits.NonPositionControlled has the following significance:

Position control enabled

FALSE = The axis is in position-controlled mode.

TRUE = The axis is in non-position-controlled mode.

S7-1200 Motion Control - Data connection for PROFIdrive drives and PROFIdrive encoders through a data block

Compile the data block for the data connection before you use it in the configuration of the axes. The required offset addresses are calculated this way.

S7-1200 Motion Control - "ReferenceMarkPosition" parameter at MC_Home

The "ReferenceMarkPosition" parameter shows the position at which the technology object was homed (homing mark), valid for "Done" = TRUE.

However, in TIA Portal V14 (S7-1200 Motion Control V6.0), the "ReferenceMarkPosition" parameter shows the position at which the technology object was homed (homing mark), plus the configured home position offset (<TO>.Sensor[n].ActiveHoming.HomePositionOffset).

If you configure a home position offset ≠ 0.0, you cannot use the "ReferenceMarkPosition" parameter of the "MC_Home" Motion Control instruction.

S7-1200 Motion Control – ErrorID 16#8400

ErrorInfo 16#0030 has been supplemented for ErrorID 16#8400:

ErrorID	ErrorInfo	Description	Remedy
16#8400		**Invalid value at parameter "Position" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
	16#0030	Value has an incorrect number format or is outside the valid number range	

PtP module with CM 1243-5 PROFIBUS Master

You have to make the following settings in the instance DB of the instructions when using the PtP modules CM PtP RS232 BA, CM PtP RS422/485 BA, CM PtP RS232 HF, CM PtP RS422/485 HF and CM PtP with a CM 1243-5 PROFIBUS Master with firmware ≤ V1.3.4:

- **Send_P2P**
  max_record_len = 240
- **Modbus_Master**
  Send_P2P.max_record_len = 240
- **Modbus_Slave**
  Send_P2P.max_record_len = 240

3.3 WinCC Basic

3.3.1 Security information

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.

Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.

Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit

http://www.siemens.com/industrialsecurity (http://www.industry.siemens.com/topics/global/en/industrial-security/Pages/Default.aspx)

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under

http://www.siemens.com/industrialsecurity (http://www.industry.siemens.com/topics/global/en/industrial-security/Pages/Default.aspx)

Passwords

Various passwords are set by default in WinCC. For security reasons, you should change these passwords.

- For the user "Administrator", the default password is "administrator".

Communication via Ethernet

In Ethernet-based communication, end users themselves are responsible for the security of their data network. The proper functioning of the device cannot be guaranteed in all circumstances; targeted attacks, for example, can lead to overload of the device.

Network settings

The following tables show the network settings of each product which you need in order to analyze the network security and for the configuration of external firewalls:

WinCC Basic (without simulation)					
Name	**Port number**	**Transport protocol**	**Direction**	**Function**	**Description**
ALM	4410*	TCP	Inbound, Outbound	License service	This service provides the complete functionality for software licenses and is used by both the Automation License Manager as well as all license-related software products.
HMI Load	1033	TCP	Outbound	HMI Load (RT Basic)	This service is used to transmit images and configuration data to Basic Panels.
* Default port that can be changed by user configuration					

WinCC Simulation for Basic Panels					
Name	**Port number**	**Transport protocol**	**Direction**	**Function**	**Description**
HMI Load	1033	TCP	Inbound	HMI Load (RT Basic)	This service is used to transmit images and configuration data to Basic Panels.
EtherNet/IP	44818	TCP	Outbound	Ethernet/IP channel	The Ethernet/IP protocol is used for connections to Allen Bradley PLCs.
	2222	UDP	Inbound	Ethernet/IP channel	The Ethernet/IP protocol is used for connections to Allen Bradley PLCs.
Modbus TCP	502	TCP	Outbound	Modbus TCP channel	The Modbus TCP protocol is used for connections to Schneider PLCs.
RFC 1006	102	TCP	Outbound	S7 channel	Communication with the S7 controller via Ethernet/PROFINET
Mitsubishi MC	5002	TCP	Outbound	Mitsubishi MC channel	The Mitsubishi protocol is used for connections to Mitsubishi PLCs.

3.3.2 News

News about the TIA Portal

You can learn about the latest news about various topic areas of the TIA Portal in Siemens Industry Online Support.

All news concerning WinCC can be found here: News (http://support.automation.siemens.com/WW/view/en/88360672)

3.3.3 Notes on use

Contents

Information that could not be included in the online help and important information about product features.

Virus scanners during installation

Virus scanners should be disabled during the installation of WinCC.

Change in response after upgrading

The new WinCC version contains new features and improvements as compared to the previous version. The new version may therefore demonstrate slight differences in the names or the response of properties and functions after the upgrade. Unlike in previous versions, the standard properties of newly created objects can also be changed.

These differences in response may lead to errors when compiling the project.

Working with styles

Please note that the system styles are reset to the default style "WinCC Dark" after upgrading the project to V14.

To restore the system style that you had set before upgrading the project, follow these steps:

1. Open the Runtime settings of the HMI device.
2. Select the desired system style under "Style of the HMI device".

Compatibility with WinCC V13

In WinCC V14, it is not possible to save the project in the WinCC V13 format.

Specifying the time of modification in the overview window

The times of modification displayed in the overview window only refer to changes to the object itself. Changes to subordinate objects, e.g. screen objects in a screen, do not cause the time of the last change to the screen to change in the overview window.

Installing East Asian project languages on a PC without Asian operating system

If you select an East Asian project language on a PC that does not have an Asian operating system installed, the default font is marked as invalid in the "Runtime settings > Language & font" editor.

To resolve this problem, open the "Regional and Language Options > Languages" dialog in the Control Panel and enable the "Install files for East Asian languages" option.

PostScript fonts

Only TrueType fonts can be used in the TIA Portal. PostScript fonts are not supported.

3.3.4 Migration

Contents

Information that could not be included in the online help and important information about product features.

Project languages in WinCC

WinCC V14 does not support all project languages that were available in WinCC flexible, such as Arabic. If you receive an empty project as the result of your migration, you may want to check the set editing language. Do not set the project languages that are not supported as editing language in the source project. Proceed as follows:

1. Open the project with WinCC flexible.
2. Change the editing language to English, for example.
3. Save the project.
4. Restart the migration.

Migrating an integrated project with ProTool objects

The "PROTOOL option package(s) missing in STEP 7" error message output during migration of a WinCC flexible project that is integrated in STEP 7 indicates that WinCC flexible 2008 SP3 is installed on your system. Moreover, the project still contains objects that were configured using ProTool. Do not open the project with WinCC flexible 2008 SP3! To migrate the project, proceed as follows:

1. Open the project in the SIMATIC Manager.
2. Execute the "Save as" command in the "File" menu.
3. Enable the option "With reorganization" in the "Save project as" dialog.
4. Click OK.
5. Copy the project back to the original computer.
6. Restart the migration.

Migrating a WinCC V7 project: Border line of rectangles

In a WinCC V7 project, you have configured a rectangle with the settings "Line weight = 1" and "Draw insider border = yes".

Migrate the WinCC V7 project to WinCC V14. To have the rectangle displayed correctly, follow these steps.

1. Open the Inspector window of the rectangle.
2. Open the property list.
3. Disable "Widen border line inwards".

Progress bar

As long as the progress bar still shows a value of 100%, the software is still busy running remaining tasks such as the closing of references. The software will not respond to user input while this status is given.

Supported project languages in WinCC V7

Due to different Windows DotNet versions, there may be different names in WinCC V7 and the TIA Portal within the same language. This may result in different sorting orders in the same project.

The following table shows some examples with English spelling:

WinCC V7.x	TIA Portal
Bengali (Bangladesh)	Bangla (Bangladesh)
Bengali (India)	Bangla (India)
Chinese (Simplified, PRC)	Chinese (People's Republic of China)
Chinese (Simplified, Singapore)	Chinese (Singapore)
Chinese (Traditional, Hong Kong S.A.R)	Chinese (Hong Kong S.A.R)
Chinese (Traditional, Macao S.A.R)	Chinese (Macao S.A.R)
Chinese (Traditional, Taiwan)	Chinese (Taiwan)
French (Monaco)	French (Principality of Monaco)
Oriya (India)	Odia (India)
Sami (Inari, Finland)	Sami, Inari (Finland)
Sami (Skolt, Finland)	Sami, Skolt (Finland)
Sami, Northern (Norway)$Sami	Sami, Northern (Norway)
Spanish (Bolivarian Republic of Venezuela)	Spanish (Venezuela)
Spanish (Spain, Traditional Sort)	Spanish (Spain)
Spanish (Spain, International Sort)	Spanish (Spain)
Yakut (Russia)	Sakha (Russia)

3.3.5 Engineering System

3.3.5.1 Screens and Screen Objects

Contents

Information that could not be included in the online help and important information about product features.

Screen objects after HMI device replacement

If you upgrade a device to the new HMI device versions, you should check the screens contained in the project. Because of the new appearance and improved operability, texts of symbolic I/O fields may not be completely legible and may be concealed by operator controls.

Grouping of screen objects

When you group screen objects in WinCC, performance problems can arise in WinCC in the case of large nesting depths.

ActiveX and .NET controls

ActiveX and .NET controls are always positioned in the foreground in runtime.

The configuration of ActiveX and .NET controls on levels is not supported.

Use of bitmaps as icons

In Windows 8 and Windows 8.1, the use of bitmaps with a size of 48x48 pixels and a color depth of 32 bits as icons is not supported.

Border line of rectangles

In a WinCC V7 project, you have configured a rectangle with the settings "Line weight = 1" and "Draw insider border = yes".

You then migrate the WinCC V7 project to WinCC V12. To have the rectangle displayed correctly, follow these steps.

1. Open the Inspector window of the rectangle.
2. Open the property list.
3. Disable "Widen border line inwards".

Border line of "Graphic I/O field" object

In WinCC V14, the dynamization of the border line of the "Graphic I/O field" object in "Two states" mode has no effect during runtime.

Dynamization of instances of a faceplate type in a group

You are using the instance of a faceplate type in an object group. The properties of the instance are also displayed as properties of the group. Any dynamization with tags, scripts or animations of the group is not displayed in Runtime.

Preview in screen window

You use your own designs with shadows for screen objects. The screen objects can be displayed in a screen window.
The shadows of the screen objects are not displayed in the preview of the screen window. The response occurs only in the engineering system. It is displayed correctly in Runtime.

Many visually different screen objects

The use of screen objects with many visually different properties (e.g. many different styles) can reduce performance in runtime and can increase the amount of available memory space used. Avoid using, for example, very many different corner radii: 0 pixel, 1 pixel, 2 pixels, 3 pixels, etc.

Dynamization of grouped objects (RT Professional)

For groupings with multiple nestings (group in group, faceplate in group, group in faceplate, etc.), only the events of the outermost group and the innermost objects can be used for dynamization with system functions. System functions that are configured at events of the lower-level group or lower-level faceplate are not executed.

The configuration of simultaneous motion animation of an object group within this group can be executed in runtime in a way other that what is expected. Check the sequence of the execution in Runtime and adapt the configuration if necessary.

Number of characters in text fields, lists and alarm texts

The number of characters that can be used in the text of a screen object is not constant. Control instructions and formatting are taken into consideration when entering text data and the maximum number of characters is reduced accordingly.

Transparency in WinCC as of V13

Transparent graphics can be displayed without any problems in Runtime. This is true for all Comfort Panels and WinCC Runtime Advanced as of version 13.0.
To use the transparency in a graphic view or in a graphic I/O field, the "Fill pattern" property must be set to "Transparent" and the property "Use transparent color" must be disabled.

If the property "Use transparent color" is enabled in a device as of version 13.0, the transparency of the graphic is lost and the transparent pixels are displayed in black. We recommend that you check the "Use transparent color" property at the points of use of transparent graphics after upgrading existing projects to device version 13.0 or higher. If the graphics are not displayed correctly, disable this property.

Displaying SIMOTION PLC websites in the HTML browser

If the SIMOTION PLC websites are not displayed correctly in the HTML browser, insert ""/basic"" after the website URL to display the websites in basic mode.

Display differences between the configuration and the display on the HMI device

The display of the text configured in a screen object may be different on the HMI device due to the display configuration. If you are using the options for automatic size adjustment in the configuration, check the display on your HMI device in every language.

If texts that were configured with the "Fit object to contents" option cannot be displayed in full, they are reduced slightly on the HMI device. If this reduction leads to a distorted display of texts, disable the "Fit object to contents" option and expand the text with additional blanks. Optionally, you can increase the width of the object or use a shorter formulation for the text.

Exporting and importing function keys

Function keys are synchronized during the import. If a function key is created in the global screen and the key is empty in the screen, the corresponding function key will use the global definition in all screens.

If you want to disable the global use of function keys after the import, define empty keys in the screens and import the screen types in the following order: Global screen, templates, screens.

If you want to ensure when exporting the screens that the global definition of a function key is not used by the template or by the global screen, create an empty function key in the screen. Select the required function key in the screen, then enable the "Use global assignment" property and disable it again.

Corner radius of the "Symbolic I/O field" object

The "corner radius" property is not available for a symbolic I/O field on Comfort Panels or RT Advanced.

The corner radius that you specify for the "Symbolic I/O field" object in the style editor has no effect in the screen editor for these target systems.

PLC code view

The following data types are not supported by the PLC Code Viewer:

- Date and time: DATE, TOD, LTOD, DT, LDT
- Characters: CHAR, WCHAR, STRING

Complete DTL structures are not supported. Only single elements of DTL structures are supported.

Jump from a monitoring alarm to the PLC code view

For the jump from a monitoring alarm to the PLC code view, the instance name must conform to the following naming convention <FB name>_DB when using supported local operands in a function block.

Note: The jump to a function or an organization block is only possible if only global operands are used.

The following restrictions apply to viewing upstream networks:

- The upstream network must not contain any tags from the Temp or InOut section of an FB.
- The following data types may not be used for tags:
  - STRING
  - WSTRING
  - CHAR
  - WCHAR
  - S5TIME
- No array element may be used with a tag to be used as an index.
- Only bit slicing is supported for slice access.

Special features: Operands and UDTs

Operands that are declared in the "#Temp" or "#InOut" area are generally not supported by the PLC code view. This applies both to elementary data types and to data types that are contained in UDTs.
Data types of a UDT can be declared in the "#In" and "#Out" area and displayed in the PLC code view. The same limitations as for elementary data types apply to the data types of the UDT.

Updating an FUP network in the PLC code view

When an FUP network contains a counter or time instruction that is not interconnected to output Q, the status of the network is not updated in the PLC code view in runtime. To refresh the status of the network in the in PLC code view, interconnect the output to an fill tags yet unused.

Faceplates for Runtime Advanced

When configuring styles at the faceplate type, the "Use style/design" property has to be available in the interface before the "Style item appearance" property is used in the interface of the faceplate. Otherwise, the style at the instance of the faceplate is lost when the configuration is updated.

GRAPH overview for Runtime Professional

When upgrading to V14, the signature of the C function at the "Click" event is changed. The propertyName parameter from the previous versions has been removed. Functions that access this parameter are shown as faulty after the upgrade.

Updating faceplates with the UDTs of the type "IEC timer"

The "ST" element was automatically removed from the user-defined PLC data types (UDT) that use the "IEC timer" data type. Adapt the faceplates containing these UDTs to a new version of the UDT without an ST element.

A detailed description of this can be found in the FAQ with the entry ID "109740393" in the SIMATIC Customer Online Support.

Unsupported PLC data types (UDTs)

The unsupported, user-defined PLC data types (UDTs) cannot be selected as a faceplate property.

Higher-level controls with visibility animation

The higher-level configuration of table-based controls with visibility animation can lead to problems for touch operation with devices using the operating system Windows CE/Embedded Compact.

See also

http://support.automation.siemens.com (http://support.automation.siemens.com/WW/view/en/62101921)

3.3.5.2 Tags and connections

Contents

Information that could not be included in the online help and important information about product features.

Multiplexing tags on a Basic Panel

If you multiplex a tag with an external tag on a Basic Panel, the address is read from the PLC in runtime during the first read cycle. The value of the address read is not available until the second read cycle.

Tag names in faceplates

Use of the "." or "@" character in names of tags in faceplate types is not permitted. Do not use these special characters in the tag names in faceplates.

Tags with symbolic addressing and "Char Array" data type (RT Professional)

Tags with symbolic addressing and the "WString" data type are not released for communication of RT Professional and SIMATIC S7-1200 V3.

Length specification for tags of the "String" type

The length specification for the internal HMI tags of the "String" type have no effect in the "Tag" editor. If you want to limit the tag length of an input field or output field, for example, limit the length using VB scripting or C scripting.

Tag of the “String” type for connections with SIMOTION controllers.

The length of the String tags must not exceed 210 characters for SIMOTION controllers.

It is possible to create a String tab with more than 210 character in the engineering system. However, the tag in question is not written in Runtime. A message stating that the value limit is outside the tag appears.

In this case, adjust the length of the String tag.

Tags under "Tag box ID"

The tag that you configure under "Tag box ID" with "Runtime Settings > General" may only be used in read-only mode.

No values may be written in this tag, however, they may be read in scripts, for example, or output to screens.

Array elements in WinCC

If you have connected an HMI tag with an array from a STEP 7 data block which does not start with a low limit of 0, the array elements are mapped in WinCC to the low limit of 0.

To ensure that you do not have to rethink between the STEP 7 indices and the WinCC indices when accessing the individual array elements, the low limits of arrays should also start at 0 in STEP 7.

Duration of the initialization of historical data

Initialization of the archives on some storage media can take up to 5 minutes. The successful completion of initialization is confirmed by a system message once it has been completed. If there a lack of any storage medium for archiving when Runtime starts, the appearance of the system message can also take up to 5 minutes.

Use of 64-bit data types

The use of 64-bit data types may result in a slight loss of accuracy as these data types are mapped in the HMI channel to the data type Double. It is therefore possible that integer data types are displayed with decimal places.

3.3.5.3 Alarm system and alarm displays

Contents

Information that could not be included in the online help and important information about product properties.

Tags in alarm texts of Runtime Advanced

Tags of data type WChar or WString cannot be displayed in the alarm view in Runtime Advanced.

Boolean tags in alarm logs

Bool type tags are recognized as 0 and -1 in the alarm log. If you use a text list that is controlled by a tag of the type Bool in an alarm log, then add the entry for the value -1 to the corresponding text list.

Duration of the initialization of historical data

Initialization of the archives on some storage media can take up to 5 minutes. The successful completion of initialization is confirmed by a system message once it has been completed. If there a lack of any storage medium for archiving when Runtime starts, the appearance of the system message can also take up to 5 minutes.

PLC alarms

A PLC alarm is only visible in the "Alarm" editor of an HMI device when the FB calls for this alarm have been invoked in an OB.

3.3.5.4 System functions

Content

Information that could not be included in the online help and important information about product properties.

The "SimulateTag" system function on Basic Panels

If you use the system function "SimulateTag" with a short cycle time on a Basic Panel, the HMI device may be overloaded.

3.3.5.5 Recipes

Contents

Information that could not be included in the online help and important information about product features.

Arrays in recipe elements

If you have configured both an array as well as the elements of this array for recipe elements of a recipe, the loading of data records aborts with the following error message: "290055: Import of data records aborted with error"

Use either the array or just the array elements for recipe elements of a single recipe.

Saving recipes on Comfort Panels

Recipes should be preferably saved on the SD card rather than in the built-in flash memory.

3.3.5.6 User administration

Contents

Information that could no longer be included in the online help and important information about product features.

3.3.5.7 Communication

Contents

Information that could not be included in the online help and important information about product features.

Connection interruptions with Mitsubishi PLCs

After multiple connection interruptions, a situation may arise where all the connection resources of the Mitsubishi PLC are in use and the connection can no longer be established. We recommend that you check these connection resources in the PLC program of the controller and enable them again.

Limited number of possible HMI connections

An error message is displayed during compilation of a device indicating that the configuration of the HMI connection in the "Devices & Networks" editor is invalid. The reason may be that the maximum number of possible connections of the HMI device or PLC has been exceeded.

Check the maximum number of available connections. Consult the device manuals of the devices you are using.

Use of PROFINET IO with panel HMI devices

When using PROFINET IO to connect the direct keys and LEDs of HMI devices to the PLC, you can define an offset for the address area of the inputs and outputs during configuration in HW Config.

The following restriction applies when a PROFINET IO-capable S7-400 CPU is used with one of the HMI devices listed below:

The offset for the start of the address area of the inputs must not be greater than the offset for the start of the address area of the outputs.

The restriction applies to the following HMI devices:

- OP 177B
- OP 277
- Mobile Panel 177

To configure the address parameters, open the PLC that contains the CPU of the 400 series in HW Config. Select the HMI device that is connected via PROFINET IO from the station window of HW Config. A table with the properties of the HMI device is displayed at the bottom of the station window in the detail view. Select the line containing the addresses of the HMI device in the table and open the object properties using the shortcut menu.

Enable the "Addresses" tab in the "Object properties" dialog. Configure the offset for the inputs under "Inputs > Start". Configure the offset for the outputs under "Outputs > Start".

Changing IP settings and device name of a PLC in the Control Panel of the HMI device

The Control Panel is open in the "Service and Commissoning > IP-Adaptation" menu on the HMI device. If you want to change the IP settings or the device name of a PLC, note the following:

In the Engineering System, you need to have activated the following options in the Inspector window of the PLC under "Properties > General > PROFINET interface > Ethernet addresses" beforehand:
- "Set IP address using a different method" and
- "Set PROFINET device name using a different method".

HMI connections in WinCC V14

HMI connections to SIMATIC S7-1200 controllers with firmware versions earlier than V2.0 are not possible in WinCC V14.

Connections via PROFIBUS DP

When a connection between a PLC and an HMI device via PROFIBUS DP is interrupted and then re-established, sporadically all other PROFIBUS DP connections in the communication network are interrupted and re-established.

De-energize the disconnected station before reconnecting it.

"Set the IP suite (address) of the PLC in the Control Panel" with SIMATIC S7-1200 V1

The function "Set the IP suite (address) of the PLC in the Control Panel" has not been approved for the following PLCs:

- SIMATIC S7-1200 V1

Switching a connection

A connection may be interrupted when it is switched from an HMI device to a SIMATIC S7-300/400, to a SIMATIC S7-1500 or to a SIMATIC S7-1200 PLC.

Note the following settings in the SIMATIC S7-1500 or SIMATIC S7-1200 controllers:

- Absolute addressing of tags
- The "Disable PUT-GET communication" option must be selected
- The "Complete protection" protection level may not be set

Raw data communication in redundant projects

Simatic.NET, Named Connections and various communication blocks, such as BSEND/BRCV, for example, can only be used to a limited extent in a redundantly configured PC station because the connection parameters for the redundant partner server cannot be configured.

Getting diagnostics data by means of HMI devices

In contrast to the description of diagnostics icons for modules and devices in the information system, the following icon has the meaning specified here for HMI devices:

Icon	Meaning
	No diagnostics data is available.

Enabling/disabling a connection in Runtime (Basic 2nd Generation)

Contrary to the description in the online help, the alarms of the respective controller are displayed after a connection is disabled. The system diagnostics for this controller is also available.

3.3.6 System-wide functions

Contents

Information that could no longer be included in the online help and important information about product features.

Using system diagnostics in the device proxy

To use the system diagnostics function in an IPE device proxy, for example, a system diagnostics view, insert the PLC alarms as content of a device proxy.

Initialize device proxy with data from a V13 project

You cannot initialize a device proxy with data from a V13 project in a project with the version V13 SP1.
Upgrade the V13 source project to version V13 SP1 to initialize the device proxy in the target project with the data from the source project.

Import and export of library texts

The following objects are not supported when importing and exporting library texts:

- Import: Recipes, Recipe elements, Recipe data records, HMI alarms, Text list entries, Screens, Reports.
- Export: Text list entries, Screens, Reports.
- Import and export of library type "Screen".

3.3.7 Compiling and loading

Contents

Information that could no longer be included in the online help and important information about product features.

Compiling and loading

If internal errors or warnings occur during compiling, compile the complete project using the command "Compile > Software (rebuild all)" in the shortcut menu of the HMI device.

Before you start productive operation with your project, compile the entire project using the "Compile > Software (rebuild all)" command from the shortcut menu of the HMI device.

If you are using HMI tags that are connected to control tags in your project, compile all modified blocks with the command "Compile > Software" in the shortcut menu before you compile the HMI device.

Incorrect installation of ProSave

If you receive an error message during installation of ProSave when loading data to a target device or maintenance of an HMI device, then you cannot remedy this error using the repair function of setup. Remove ProSave via the Control Panel. Then start setup and install the "ProSave" component again.

Checking the address parameters

During compilation of an HMI device in the project tree with the command "Compile > Software" in the shortcut menu, the address parameters of the HMI device, such as the IP address, will not be checked. If you want to ensure that the address parameters are checked as well, you will have to compile the HMI device using the "Compile" button in the "Devices & Networks" editor of the toolbar.

Error message when downloading data to the PLC

A panel and a PLC are connected and communicating with other.

If a tag is accessed while downloading data from the panel to the PLC, an error message is displayed on the panel.

Delayed reaction in the "Extended download to device" dialog

If the settings in the "Extended download to device" dialog for "Type of the PG/PC interface" and "PG/PC interface" do not match the settings on the HMI device, this can result in the application not responding for up to a minute.

Extended download with an S7-1200 and a Comfort Panel

An S7-1200 PLC and a Comfort Panel are located in the same physical network as the PG/PC. You open the "Extended download to device" dialog for the Comfort Panel.

If you enable the option "Show all accessible devices", it may occur that the application stops responding.

OP77A, OP73, TP177A: Loading projects

When loading a project to an HMI device, it can happen that Runtime is not automatically ended, even though "Remote Transfers" is activated in the Panel.

If this happens, stop Runtime and manually set the transfer mode on the HMI device.

Project transfer via USB

If you have connected more than one HMI device via USB to your configuration PC, project transfer is only possible to the last connected device.

Comfort panels as of device version 13.0: Backing up data while loading projects

If the transfer is interrupted for Comfort Panels with a device version 13.0 or higher, WinCC automatically ensures that no data is lost and that existing data is only deleted on the HMI device after complete transmission.

Loading projects to a file

The "Load project to file" functionality is only available in Runtime Advanced devices with version V13 or older.

Displaying characters in transfer alarms

If characters are not displayed correctly in the transfer alarms during transfers to HMI devices with the device version V12 or older, please check the region and language settings in Windows. Set the corresponding language under "Language for non-Unicode programs".

Missing fonts (panels)

The following fonts no longer belong to the standard installation pack in Windows 10 and are not pre-installed:

- MS PGothic (is used as standard for displaying Japanese characters)
- Gulim (is used as standard for displaying Korean characters)
- MingLiU for Chinese (is used as standard for displaying Chinese characters for Taiwan, Hong Kong and Macao)

If a font is not installed, an error message is displayed in the engineering system. The error message appears in every TIA Portal session if you navigate to “Runtime settings > Language and font” or open screen objects which use the missing font. The missing font is replaced with a suitable font by the TIA Portal.

Missing fonts cause problems during compilation and in Runtime.

For example, if you open an error-free V13 SP1 project that was created on a configuration PC with Windows 7 on a configuration PC of V14 SP1 with the Windows 10 operating system, error messages can be output during compilation because of the fonts that are not installed.

You can find more detailed information on fonts in Windows 10 at https://support.microsoft.com/en-us/kb/3083806 (https://support.microsoft.com/en-us/kb/3083806)

Solution

1. Compile the project before you transfer it to your HMI device. How to determine whether the fonts you require are missing.
2. Install these fonts and/or the corresponding Windows language pack.

Alternatively, you can also compile the projects with substitute fonts.

The missing font is saved in the configuration. As soon as this font is available again, it is displayed and used in the configuration.

Reducing the project size

When compiling your HMI device, a message appears informing you that the size of your project is approaching the system limits for the corresponding HMI device. In this case, perform a complete compilation of the software to reduce the project size. To do this, use the command "Compile > Software (rebuild all)".

See also

http://support.microsoft.com (http://support.microsoft.com/default.aspx?scid=kb;EN-US;224305)

http://msdn.microsoft.com/en-us/library/ms968493.aspx (http://support.microsoft.com/default.aspx?scid=kb;EN-US;224305)

3.3.8 Runtime

3.3.8.1 Notes on operation in Runtime

Contents

Information that could not be included in the online help and important information about product features.

Focus in runtime

If you have configured a low-contrast combination of focus color and border color in a V12 project, the focus may no longer be identifiable after changing the HMI device version in Runtime. Change one of the two colors.

Language behavior - Layout of on-screen keyboard

The layout of the on-screen keyboard does not change when you switch to a runtime language that is not installed for the keyboard layout.

In this case, the language setting for the keyboard remains set at the most recent valid language or the language setting for the default keyboard layout of Windows is used.

Tag values exceed the maximum length

You enter a character string in a string tag via an I/O field. If the character string exceeds the configured number of tags, the character string will be shortened to the configured length.

Empty alarm texts

Runtime is running with a project. The project is saved on a network drive.

In the event of interruptions to the network drive connection, Runtime may attempt to load alarm texts from the network drive.

In the event of disconnection, the alarm window or the alarm view remains empty.

To avoid this, copy the project to a local drive before the starting the project in Runtime.

Large logs delay the ending of Runtime (Basic Panels 2nd Generation)

When very large logs are used, ending Runtime can take a long time. Use segmented logs as an alternative to very large circular logs.

Complete download in Service mode

If you need to perform a "complete download" to the OS in Service mode from the engineering station, Runtime automatically stops and then starts again.

The project is then no longer in Service mode.

In this state, the power supply is interrupted and WinCC Runtime no longer starts automatically on the OS.

Remedy:

1. Switch the project manually to Service mode after you have performed the "complete download".
2. Close the project manually.
3. Enable Service mode.
4. Start Runtime again using the surrogate icon in the taskbar.

Slow reaction of SmartServer

The following programs may start and respond very slowly under Windows 7 and Windows 2008 servers:

- HMI TouchInputPC
- SmartServer: <Ctrl+Alt+Del> shortcut in the logon dialog

The delay is caused by the callback for the Internet certificate validation.

Remedy:

You can find the following files
on the product DVD under:
Support\Windows7\CRL_Check or CD_RT\ Support\Windows7\CRL_Check\:

- DisableCRLCheck_LocalSystem.cmd
- DisableCRLCheck_CurrentUser.cmd

1. Run the "DisableCRLCheck_LocalSystem.cmd" file with administrator rights. Select the command "Run as administrator" from the shortcut menu of the file.
2. Reboot the PC.

If the problem persists, follow these steps:

1. Double-click the file and run the "DisableCRLCheck_CurrentUser.cmd" file with user rights.
2. Reboot the PC.

Note

The callback for the certificate validation is disabled for all users or PCs. To restore the original state, perform the following files:

- RestoreDefaults_LocalSystem.cmd
- RestoreDefaults_CurrentUser.cmd

You can find the files in the following directory of the product DVD:

- Support\Windows7\CRL_Check or CD_RT\Support\Windows7\CRL_Check\

Ending screensaver on the Sm@rtServer

When the screensaver is active on the Sm@rtServer on the server HMI device, you require write access to the Sm@rtClient side in order to end the screen server on the server HMI device.

Avoiding corrupt files during power failure

If a power failure occurs in Windows systems while the WinCC system is active, files may be corrupt or destroyed. Operation with the NTFS file system provides better security.

Secure, continuous operation is only ensured by using an uninterruptible power supply (UPS).

3.3.8.2 Notes on operation of panels in Runtime

Contents

Information that could not be included in the online help and important information about product features.

License transfer via S7USB

You always need to run WinCC to transfer a license to a panel via S7USB.

Using the mouse wheel in Runtime

The use of the mouse wheel in Runtime is not supported on all panels.

Basic Panels: Backup on the memory card of the PLC

Create the backup file "A.psb" on the memory card of the PLC. An error, for example a connection break, occurs when creating the backup.
This will create a corrupt file on the memory card of the PLC. Such a file has "~$" as prefix. Delete the file with the prefix "~$" if you want to save a backup again under the same name "A.psb".

Basic Panels: Panel Data Storage and S7-1500F

The "Panel Data Storage" PDS function cannot be used on Basic Panels in conjunction with S7-1500F when the password for the protection level "Full access incl. fail-safe" is used.

"Panel Data Storage" function on Basic Panels

The "Panel Data Storage" (PDS) function provided by Basic Panels is only supported by SIMATIC S7-1200 as of firmware V4.0 and SIMATIC S7-1500. For the PDS function, the panel must be connected directly with the CPU and must not be connected via the CP.

Fonts with file extension ".ttx"

The fonts with the file extension ".ttx" are available on all panels and Runtime Advanced and can be correctly displayed on all devices, for example, the "WinCC_flexible_smart" font.

Mass data transfer with USB

If you have installed Windows 10 on the configuration PC, the "USB" transfer channel is not available.

This restriction applies to the following devices and tools:

- Basic Panels: KP300 Basic, KP400 Basic, TP1500 Basic
- Panels: OP and TP of 70 series, 177 series and 270 series
- Multi Panels: Multi Panels of the170 series, 270 series and 370 series
- Mobile Panels: 170 series and 270 series
- WinCC Engineering System
- "ProSave" service tool

Note

The "USB" transfer channel remains available if Windows 7 or Windows 8 is installed on the configuration PC.

Solution

Use other transfer channels to transfer the project data, such as the "Industrial Ethernet" or "PROFIBUS DP" channel.

3.3.9 HMI devices

3.3.9.1 Notes on HMI devices

Contents

Information that could not be included in the online help and important information about product features.

If the PC goes into standby or hibernate mode while the transfer is in progress, the panel status after interruption of the transfer is not defined.

Multi-key operation

Unintentional actions can be triggered by multi-key operation:

- When you are using a key device, you cannot press more than two function keys at the same time.
- When you are using a touch device, a standard PC or a panel PC, you can only press one function key or button at the same time.

HMI devices with high communication load

S7 Diagnostics should be enabled if a Panel is assigned many connections to PLCs or other HMI devices. Otherwise, you will risk overload on the Panel.

TS Adapter with Ethernet interface

If an HMI device is connected via Ethernet and a TS adapter, it can not be reset to factory settings.

Simulation with real PLC connection

The access point used by the simulation is independent from the settings of the engineering system and can only be altered in the Control Panel with the "Setting PG/PC Interface" tool. If the PLC connection is terminated right after the start of the simulation with alarm 140001, you should check the access point used by the simulation with "Setting PG/PC Interface".

1. Double-click "Setting PG/PC Interface" in the Control Panel. A dialog opens.
2. Select "S7ONLINE" in the "Access point of application" field as standard for HMI.
3. Select the interface in the "Interface Parameter Assignment Used" area.
4. Exit the dialog "Setting PG/PC Interface" with OK

Simulation of the Basic Panels

Use an output field in an alarm text to output an external tag. The content of the output field will always be displayed with "0" during simulation.

Upgrading Basic Panels to device version V12

Before you upgrade Basic Panels from device version V11 to device version V12, transfer the image of the V11 SP2 Update 5 or higher to the devices.
In the "SIMATIC ProSave [OS Update]" dialog, select the setting "Reset to factory settings".

In this way, you always start a functional update of the image.

Affected devices:

- KP300 Basic mono PN
- KP400 Basic color PN
- KTP400 Basic color PN

Basic Panels 2nd Generation

Basic Panels 2nd Generation support the Sm@rtServer option.

If you are not using a USB hub, select the USB port USB_X60.1 as storage path.

HMI devices with operating system Windows CE 5.0 or higher

Owing to a modified client-server communication security setting, the time difference between the HMI device (client) and PC (server) must not exceed 1 day. If you back up recipe data from the HMI device on a network drive, for example, make sure that the time is set correctly on the PC (server) and the HMI device (client).

Floating point numbers on MP 277, MP 377, TP 177B 4" and CP4

Only floating point numbers in the range from 10^{-293} ... 10^{+307} are displayed correctly on the HMI devices MP 277, MP 377, TP 177B 4" and CP4. If the tag value is outside this range, it is displayed as 0.

Restrictions for the HMI device, MP 377 15" Touch daylight readable

The following functions are not supported in WinCC V12 for the MP 377 15" Touch daylight readable HMI device:

- Option: Sm@rtServer
- System function: SetAndGetBrightness
- Direct keys

Connection switch in the Control Panel with Basic Panels

If you use the "Override protected connection information" function, the following restriction applies:

You cannot perform a connection switch in the Control Panel of a Basic Panel from a PLC without a protection level to a PLC with a "Complete protection" level.

Mobile Panel

Communication between fail-safe Mobile Panel and a S7-1200F is not possible with V14.0 and will not be enabled until a later update.

Mobile Panels 277F IWLAN

Mobile Panels 277F IWLAN are approved for use with CPUs of the type S7-1500F with WinCC V13 SP1 Update 4.

If this connection is not established automatically after the panel is switched on or after a fail-safe connection to a CPU S7-1500F is interrupted, a STOP/RUN transition needs to be performed on the CPU.

Affected MLFBs:

- 6AV6645-0EB01-0AX1
- 6AV6645-0EC01-0AX1
- 6AV6645-0EF01-0AX1
- 6AV6645-0GB01-0AX1
- 6AV6645-0GC01-0AX1
- 6AV6645-0GF01-0AX1

Using Mobile Panel 277F IWLAN in the industrial environment

To avoid communication problems between the HMI device and access point, observe the following:

IWLAN planning: Wireless connection between the HMI device and access point

- Wireless field planning needs to be carried out. Radar influences need to be taken into consideration for channels requiring Dynamic Frequency Selection (DFS).
- Avoid radio frequency interference from other devices, such as hand-held devices with Bluetooth or WLAN functionality.
- For adjacent access points, use data channels that are separated by at least one wireless channel (e.g. 36 and 44 rather than 36 and 40).
- Do not place several access points in the immediately vicinity of one another. We recommend a minimum distance of 5 m.
- For the Mobile Panel 277F IWLAN, use the "Software retry number" 15 instead of the default value 3.

Using the "Rapid roaming" function with iPCF-MC

- Always use iPCF or iPCF-MC (rapid roaming) for PROFIsafe communication.
- Use the same SSID for all access points.
- The "Background scan" should be switched off on the HMI device ("Background scan interval" for the HMI device = 1 (default value)).
- Use the same "Management channel" and different, non-overlapping "Data channels" for all access points.
- Configure the "PN-IO cycle time" higher than the "iPCF scan cycle time".
- Observe the configured timeout when assigning PROFIsafe parameters.

"Zone ID/Connection point ID" tag of a Mobile Panel 277 IWLAN V2

The tag used for the "Zone ID/Connection point ID" must be of data type INT for Mobile Panel 277 IWLAN V2 devices. Adapt this data type if necessary when migrating a project.

Mobile Panels 277F IWLAN (RFID tag): F_DB_STATES

The F_DB_STATES block is no longer supplied with WinCC V14. The block F_DB_STATES serves only for data exchange. You can configure a block to replicate the function of the F_DB_STATES, however. Note the description of the F_DB_STATES in the operating instructions of your HMI device or in the information system.

IPC427D

If an IPC427D is configured in a V12 or V13 project in which IE1 of the PC station is assigned and the device version of the PC station changes from V1 to V2 after upgrading to V14, the interface assignment of CP_IE_1 must initially be set to "None, or different Windows settings" and then changed to "SIMATIC PC Station" before compiling the project . Then, the project can be compiled without errors.

Sm@rtServer with KTP Mobile Panels

The evaluation of the safety operation via PROFIsafe is only guaranteed if the Sm@rtServer is disabled on the panel.

The use of Sm@rtServer on devices without a safety function is not recommended.

Printing alarms (Mobile Panels 2nd Generation)

It is not possible to print alarms as PDF or HTML when the internal memory "/flash" is selected as "Storage Location". To print alarms as PDF or HTML, select an SD memory card or a USB stick as storage location.

System functions in the zone editor on KTP Mobile Panels

In contrast to what is described in the help, the logon and logoff of the current user cannot be initiated by system functions. Use the ShowLogonDialog system function to log users onto the HMI device via the logon dialog.

For KTP700F Mobile and KTP900F Mobile, system functions that are configured to the "Connected" event are duplicated.

DHCP on KTP 400 PN, KTP 700 PN and KTP 900 PN

Operation with DHCP is not recommended for the KTP 400 PN, KTP 700 PN and KTP 900 PN devices.

In some circumstances, it is possible that the device will no longer boot. If that is the case, detach the network cable and reboot the device. After it has successfully started up, please assign a fixed IP address.

Installation 4

4.1 Notes on the installation

Contents

Information that could not be included in the online help and important information about product characteristics.

Installation in a virtual environment (private cloud)

You can find instructions on how to install the TIA Portal in a virtual environment (private cloud) on the installation data medium in the directory "Documents\Readme\<language directory>". You can open the PDF document "TIAPortalCloudConnectorHowTo<language ID>.pdf" here.

Automated Installation

A description of automated installation is available on the product DVD in the directory "Documents\Readme\<Language directory>".

Use of the same versions of TIA Portal products during installation

When installing different TIA Portal products, make sure that you use the same versions of service packs and updates for the installation. For example, if you have installed SP1 for STEP 7 V13, you must also install SP1 for WinCC V13. The service packs and updates must be installed for all products at the same time. Do not start the TIA Portal until all products have been upgraded.

You can download the service packs from the Internet under Siemens Industry online support (https://support.industry.siemens.com/cs/de/en/).

Target directory of the installation

Do not use any UNICODE characters (for example, Chinese characters) in the installation path.

Use of antivirus programs

During the installation, read and write access to already installed files is necessary. Some antivirus programs block this access. We therefore recommend that you disable antivirus programs during installation of the TIA Portal and enable them again afterwards.

Compatibility with V12 SP1 and V13

Empty projects from previous versions are installed in the installation directory under .. \<INSTALLDIR>\Sample Projects to allow the current TIA Portal to be opened in compatibility mode. These projects must be copied to a local directory with full access before they can be used. For more information on this, refer to FAQ ID 82169157.

FAQs on the TIA Portal

FAQs on the TIA Portal are available under FAQs (http://support.automation.siemens.com/WW/view/en/28919804/133000).

4.2 System requirements for installation

4.2.1 Notes on licenses

Availability of licenses

The licenses for the products of the TIA Portal are usually supplied on an installation data medium and transferred automatically by the Automation Licence Manager during the installation process of the TIA Portal.

Before you uninstall the TIA Portal, you must transfer and back up the licenses still required. Use the Automation License Manager for this purpose.

Provision of the Automation License Manager

The Automation License Manager is supplied on the installation data medium and is transferred automatically during the installation process.

If you uninstall the TIA Portal, the Automation License Manager remains installed on your system.

Working with the Automation License Manager

The Automation License Manager is a product of Siemens AG, which is used for handling license keys (technical representatives of licenses).

Software products that require license keys for operation, such as the TIA Portal, register the need for license keys automatically with the Automation License Manager . If the Automation License Manager finds a valid license key for this software, the software can be used according to the license usage terms associated with this license key.

Note

For additional information on how to manage your licenses with the Automation License Manager , refer to the documentation supplied with the Automation License Manager .

See also

Notes on the system requirements (Page 126)

Starting installation (Page 146)

Displaying the installed software (Page 148)

Modifying or updating installed products (Page 149)

Repairing installed products (Page 151)

Starting to uninstall (Page 153)

Installation log (Page 145)

4.2.2 Notes on the system requirements

System requirements for individual products

The system requirements may differ depending on the products you want to install. You should therefore check the individual system requirements of your products.

If you want to install several products, make sure that your system meets the demands of the product with the highest requirements.

Displaying PDF files

To be able to read the supplied PDF files, you require a PDF reader that is compatible with PDF 1.7 e.g. Adobe (R) Reader version 9.

Displaying the Welcome Tour

You need the Adobe (R) Flash Player Version 9 or higher to start the Welcome Tour for the TIA Portal.

See also

Notes on licenses (Page 125)

Starting installation (Page 146)

Displaying the installed software (Page 148)

Modifying or updating installed products (Page 149)

Repairing installed products (Page 151)

Starting to uninstall (Page 153)

4.2.3 System requirements STEP 7 Basic

4.2.3.1 Licensing of STEP 7

Introduction

You require a License Key to license the following STEP 7 editions:

- STEP 7 Basic
- STEP 7 Professional

You can install the corresponding License Key when you are installing STEP 7 or transfer it using the Automation License Manager after the installation has been completed.

Licenses for STEP 7

The following licenses with the corresponding license keys are available:

- STEP 7 Basic
- STEP 7 Professional
- STEP 7 Professional Combo

Validity of license keys for older versions of STEP 7

With a valid License Key for V14.x of STEP 7 Professional and STEP 7 Professional Combo, you can also operate older versions of STEP 7 without restrictions. The following table provides more detailed information about this:

Edition	License	Valid for
STEP 7 Basic V14.x	STEP 7 Basic	• STEP 7 Basic V14.x • STEP 7 Basic V13.x • STEP 7 Basic V12.x • STEP 7 Basic V11.x • STEP 7 Basic V10.5
STEP 7 Professional V14.x	STEP 7 Professional	• STEP 7 Professional V14.x • STEP 7 Professional V13.x • STEP 7 Professional V12.x • STEP 7 Professional V11.x • STEP 7 Basic V14.x • STEP 7 Basic V13.x • STEP 7 Basic V12.x • STEP 7 Basic V11.x • STEP 7 Basic V10.5
STEP 7 Professional V14.x	STEP 7 Professional Combo	• STEP 7 Professional V14.x • STEP 7 Professional V13.x • STEP 7 Professional V12.x • STEP 7 Professional V11.x • STEP 7 Basic V14.x • STEP 7 Basic V13.x • STEP 7 Basic V12.x • STEP 7 Basic V11.x • STEP 7 Basic V10.5 • STEP 7 V5.5 • STEP 7 V5.4 • STEP 7 Professional 2010 • STEP 7 Professional 2006

Starting without a valid license key

If you start an edition of STEP 7 without a valid License Key , the system alerts you that you are working in non-licensed mode. You have the one-time option of activating a Trial License. However, this license is valid for a limited period only and expires after 21 days.

When the Trial License expires, the following scenarios can occur:

- STEP 7 has never been licensed on the PC in question:
  - Operations requiring a license can no longer be performed in STEP 7.
- STEP 7 was already licensed on the PC in question:
  - A window requiring acknowledgment presents an alert for non-licensed mode every 10 minutes and for every action requiring a license.

License requirements for simulation

You do not require additional licenses when you use the menu command "Online > Simulation" to start the simulation in STEP 7.

If the following conditions are met, the appropriate licenses for the edition of STEP 7 you have installed are also required for the simulation.

- The engineering station is connected to a PLC.
- The connection to the PLC is configured and active.

Licenses for the TIA Portal Cloud Connector

For working with the TIA Portal Cloud Connector, you need a valid License Key for each device that you specify as "User device" in the TIA Portal Cloud Connector. The License Key is also required if the TIA Portal is installed on this device. A License Key is not required for devices that you are using as "remote device".

See also

Handling licenses and license keys (Page 129)

4.2.3.2 Handling licenses and license keys

Introduction

In each case, you require a valid License Key to use STEP 7 Basic and STEP 7 Professional.

Installing license keys

When you install STEP 7 Basic, the License Key is installed automatically during setup. When you install STEP 7 Professional you will be prompted at the end of the setup to transfer the license from the supplied storage medium to your PC.

If you want to install additional License Keys, you have to use the Automation License Manager to do this.

When you install a license, the associated license key is removed from the license key storage location.

NOTICE
Destruction of license keys by copying It is not possible to copy a License Key. The copy protection prevents the license keys from being copied. If you attempt to copy a License Key this will be destroyed.

Uninstalling license keys

License keys are always uninstalled using the Automation License Manager. You uninstall a License Key in the following cases:

- When backing up data.
- If you no longer require the license.

You can then use a valid license on another PC or HMI device.

Data backup

When backing up data on the HMI device or creating a backup during device replacement, remove the License Keys on the HMI device. To do this, open the Automation License Manager and back up the uninstalled license key to another storage location.

NOTICE
Destruction of license keys on PCs Start by removing all license keys in the following situations: • Before you format the hard disk. • Before you compress the hard disk. • Before you restore the hard disk. • Before you start an optimization program that moves fixed blocks. • Before you install a new operating system. Read the description of Automation License Manager ("Start > Siemens Automation > Documentation"). Observe all warnings and notices.

On PC-based HMI devices and on non-PC-based HMI devices where Automation License Manager is used, the license key storage location may contain multiple license keys. This capability means you can store multiple licenses of the same type at one location. Save all license keys of an HMI device to the same storage location.

NOTICE
Always keep the original storage location of the license keys.

Invalid license after time zone change

The installed license no longer functions in the following case.

- If you change the time zone on a PC as follows:
  From a time based on a complete hour to a time not based on a complete hour.
  Example: You change the time zone from GMT +3:00 to GMT +3:30.

To avoid this inconvenience, uninstall the license key using the Automation License Manager under the time zone setting that was set when the license key was installed.

This behavior does not apply to the Trial License.

Defective license

A license is defective in the following cases:

- If the license key is no longer accessible at the storage location.
- If the license key disappears during its transfer to the destination drive.

You can use the Automation License Manager to repair the defective license. To do this, use the "Restore" function or the "Restore wizard" of the Automation License Manager. To perform this restore operation you must contact Customer Support.

You can find more detailed information on the Internet: http://support.automation.siemens.com (http://support.automation.siemens.com/WW/llisapi.dll?aktprim=99&lang=en&referer=%2fWW%2f&func=cslib.csinfo2&siteid=csius&extranet=standard&viewreg=WW&groupid=4000002)

See also

Licensing of STEP 7 (Page 127)

4.2.3.3 Software and hardware requirements STEP 7

System requirements for installation

The following table shows the minimum software and hardware requirements that have to be met for installation of the "SIMATIC STEP 7 Basic" software package:

Hardware/software	Requirement
Processor	Intel® Core™ i3-6100U, 2.30 GHz
RAM	4 GB
Hard disk	S-ATA with at least 8 GB available memory space **
Network	From 100 Mbit

Hardware/software	Requirement
Screen resolution	1024 x 768
Operating systems *	**Windows 7 (64-bit)** • Windows 7 Home Premium SP1 • Windows 7 Professional SP1 • Windows 7 Enterprise SP1 • Windows 7 Ultimate SP1 **Windows 8.1 (64-bit)** • Windows 8.1 • Windows 8.1 Professional • Windows 8.1 Enterprise **Windows 10 (64-bit)** • Windows 10 Home Version 1607 • Windows 10 Professional Version 1607 • Windows 10 Enterprise Version 1607 • Windows 10 Enterprise 2016 LTSB • Windows 10 Enterprise 2015 LTSB **Windows Server (64-bit)** • Windows Server 2012 R2 StdE (full installation) • Windows Server 2016 Standard (full installation)

* Including all security updates up to Microsoft Security Bulletin Summary for February (MS17-Feb). For more detailed information on operating systems, refer to the help on Microsoft Windows or the Microsoft Web site.

** For operation of SIMATIC STEP 7 Basic we recommend free storage space of at least 20 GB

Recommended PC hardware

The following table shows the recommended software and hardware for the operation of STEP 7 Basic.

Hardware/software	Requirement
Computer	SIMATIC FIELD PG M5 Advanced or higher (or comparable PC)
Processor	Intel® Core™ i5-6440EQ (up to 3.4 GHz)
RAM	16 GB or more (32 GB for large projects)
Hard disk	SSD with at least 50 GB available memory space
Network	1 Gbit (for multiuser)
Monitor	15.6" Full HD Display (1920 x 1080)

Supported virtualization platforms

You can install the "SIMATIC STEP 7 Basic" software package on a virtual machine. For this purpose, use one of the following virtualization platforms in the specified version or a newer version:

- VMware vSphere Hypervisor (ESXi) 6.0
- VMware Workstation 12.5
- VMware Player 12.5
- Microsoft Hyper-V Server 2016

The following operating systems are recommended as host operating systems for these virtualization platforms:

Operating system	VMware vSphere Hypervisor (ESXi)	VMware Workstation	VMware Player	Microsoft Hyper-V
Windows Server 2016 Standard (full installation, 64-bit)	-	-	-	x
Windows 7 Professional SP1	x	x	x	-
Windows 7 Enterprise SP1	x	x	x	-
Windows 7 Ultimate SP1	x	x	x	-
Windows 10 Professional Version 1607	x	x	x	-
Windows 10 Enterprise Version 1607	x	x	x	-
Windows 10 Enterprise 2016 LTSB	x	x	x	-

x: Can be used as host operating system

-: Cannot be used as the host operating system

The following guest operating systems are recommended for installing "SIMATIC STEP 7 Basic" within the selected virtualization platform:

Operating system	VMware vSphere Hypervisor (ESXi)	VMware Workstation	VMware Player	Microsoft Hyper-V
Windows Server 2008 R2 StdE SP1 (full installation, 64-bit)	-	-	-	x
Windows Server 2012 R2 StdE (full installation, 64-bit)	-	-	-	x
Windows Server 2016 Standard (full installation, 64-bit)	-	-	-	x
Windows 7 Professional SP1	x	x	x	x
Windows 7 Enterprise SP1	x	x	x	x
Windows 7 Ultimate SP1	x	x	x	x
Windows 10 Professional Version 1607	x	x	x	x

Operating system	VMware vSphere Hypervisor (ESXi)	VMware Workstation	VMware Player	Microsoft Hyper-V
Windows 10 Enterprise Version 1607	x	x	x	x
Windows 10 Enterprise 2016 LTSB	x	x	x	x

x: Can be used as guest operating system

-: Cannot be used as the guest operating system

Note

- The same hardware requirements apply to the host operating system as for the respective TIA products.
- The plant operator must ensure that sufficient system resources are available for the host operating systems.
- The hardware certified by the manufacturers is recommended for the use of HyperV server and ESXi.
- When you use Microsoft Hyper-V, accessible stations cannot be displayed.

Supported security programs

The following security programs are compatible with "SIMATIC STEP 7 Basic":

- Antivirus programs:
  - Symantec Endpoint Protection 12.1
  - Trend Micro Office Scan Corporate Edition 11.0
  - McAfee VirusScan Enterprise 8.8
  - Kaspersky Anti-Virus 2016
  - Windows Defender (as of Windows 8.1)
  - Qihoo "360 Total Security Essential" 9.7

  The last update of the virus pattern was on 08 MAR 2017.
- Encryption software:
  - Microsoft Bitlocker
- Host-based Intrusion Detection System:
  - McAfee Application Control 6.2.0

4.2.4 System requirement for WinCC Basic

4.2.4.1 Software and hardware requirements

Introduction

Specific requirements for the operating system and software configuration must be met for the installation.

Note

WinCC is generally authorized for use in a domain or workgroup.

However, be aware that domain group policies and restrictions of the domain may hinder the installation. If this happens, remove the computer from the domain prior to installing Microsoft Message Queuing, Microsoft SQL Server and WinCC. Log onto the computer in question locally with administrative rights. Then perform the installation. After successful installation, you can enter the WinCC computer back into the domain. If the domain group policies and restrictions of the domain do not impede the installation, the computer need not be removed from the domain during the installation.

Be aware that domain group policies and restrictions of the domain may also hinder operation. If you cannot avoid these restrictions, run the WinCC computer in a workgroup.

Consult with the domain administrator if needed.

Installation requirements

The following table shows the minimum software and hardware requirements that have to be met for the installation of the "SIMATIC WinCC Basic" software package:

Hardware/software	Requirements
Processor type	Intel® Core™ i3-6100U 2.30 GHz
RAM	4 GB
Free hard disk space	S-ATA with at least 8 GB available memory **

Hardware/software	Requirements
Operating systems *	**Windows 7 (64-bit)** • Windows 7 Home Premium SP1 • Windows 7 Professional SP1 • Windows 7 Enterprise SP1 • Windows 7 Ultimate SP1 **Windows 8.1 (64-bit)** • Windows 8.1 • Windows 8.1 Professional • Windows 8.1 Enterprise **Windows 10 (64-bit)** • Windows 10 Home Version 1607 • Windows 10 Professional Version 1607 • Windows 10 Enterprise Version 1607 • Windows 10 Enterprise 2016 LTSB • Windows 10 Enterprise 2015 LTSB **Windows Server (64-bit)** • Windows Server 2012 R2 StdE (full installation) • Windows Server 2016 Standard (full installation)
Screen resolution	1024 x 768
Network	From 100 Mb
Optical drive	DVD-ROM
Software	Microsoft .Net Framework 4.6.2

* Including all security updates up to Microsoft Security Bulletin Summary for February (MS17-Feb). For more detailed information on operating systems, refer to the help on Microsoft Windows or the Microsoft Web site.

** For operation of SIMATIC WinCC Basic, we recommend free storage space of at least 20 GB.

Simultaneously opening multiple instances of WinCC on a configuration PC can also increase the hardware capacity required.

Note

"Aero Glass Style" of Microsoft Windows 7

A powerful graphics card is required for "Aero Glass Style". It requires DirectX9 capabilities and 128 MB of dedicated graphics memory.

The performance of the architecture of the graphics system can significantly influence the performance of WinCC.

Recommended hardware

The following table shows the recommended hardware for the operation of SIMATIC WinCC.

Hardware	Requirements
Computer	As of SIMATIC FIELD PG M5 Advanced (or comparable PC)
Processor	As of Intel® Core™ i5-6440EQ (up to 3.4 GHz)
RAM	16 GB or more (32 GB for larger projects)
Hard disk	SSD with at least 50 GB available memory space
Monitor	15.6" Full HD Display (1920 x 1080)
Network	From 1 GB
Optical drive	DL MULTISTANDARD DVD RW

Supported virtualization platforms

You can install the "SIMATIC WinCC Basic" software package on a virtual machine. For this purpose, use one of the following virtualization platforms:

- VMware vSphere Hypervisor (ESXi) 6.0
- VMware Workstation 12.5
- VMware Player 12.5
- Microsoft Windows Hyper-V Server 2016

All released operating systems can be used as host operating systems on a virtualization platform.

Note

- The same hardware requirements apply to the guest operating system as to the respective TIA products.
- The plant operator must provide sufficient system resources for the guest operating systems.
- The hardware certified by the manufacturers is recommended for the use of HyperV server and ESXi.

Supported security programs

The following security programs are compatible with "SIMATIC WinCC Basic":

- Virus scanners:
  - Symantec Endpoint Protection 12.1
  - Trend Micro Office Scan Corporate Edition 11.0
  - McAfee VirusScan Enterprise 8.8
  - Kaspersky Anti-Virus 2016
  - Windows Defender (as of Windows 8.1)
  - Qihoo "360 Total Security Essential" 9.7

  The last update of the virus patterns will be performed 08/03/2017.
- Encryption software:
  - Microsoft Bitlocker
- Host-based Intrusion Detection System:
  - McAfee Application Control 6.2.0

Installing Microsoft .Net Framework

You will be prompted to install the required .Net Framework version if this cannot be performed by the WinCC installation. A restart may be necessary after installation of .Net Framework.

Security settings during installation

When you install WinCC V14 SP1, security settings are changed in your operating system.

The security settings in question are listed during the installation.

You must confirm the changes to the security settings.

If you make changes to your operating system after the installation, the changes to the security settings made by the installation of the TIA Portal could be changed.

You can restore the changes to the security settings made by the installation of the TIA Portal: "Start > All Programs > Siemens Automation > Security Controller > Restore settings".

SQL instance of WinCC V14 SP1

A new WinCC SQL 2014 instance is installed with the installation of WinCC V14 SP1. If you already had a WinCC version installed, you must uninstall the WinCC instance of older SQL servers before the installation.

See also

Licensing of WinCC Engineering System (Page 139)

http://support.microsoft.com/kb/917607 (http://support.microsoft.com/kb/917607)

4.2.4.2 Parallel installation

Parallel installations in TIA Portal V14

You will be prevented from starting the TIA Portal if you perform a non-permitted parallel installation of STEP 7 and WinCC. The following parallel installations are permitted in the TIA Portal:

- STEP 7 V14 and WinCC V14

A dialog opens during installation to inform you of any inconsistencies in your parallel installation. The following parallel installations are permitted:

- WinCC V14 and RT Advanced V14
- WinCC V14 and RT Professional V14

The engineering system and Runtime must always be of the same version after an installation.

Parallel installation of WinCC V14 and other SIMATIC HMI products

Parallel installation of WinCC V14 and versions of WinCC flexible earlier than WinCC flexible 2008 SP1 is not allowed.

Parallel installation of WinCC V14 with versions of WinCC earlier than WinCC V7.0 SP2 is not allowed. Parallel installation of WinCC V14 with WinCC V7.0 SP2 or WinCC V7.0 SP3 is not allowed for:

- WinCC V 14 Professional
- WinCC V 14 Runtime Professional

Parallel use

If the term "Combo" appears in the name or license key of the software after installation, the use of the following products/versions is permitted in accordance with paragraph 4.3 of the General Terms and Conditions (see also setup text):

- With the license "WinCC V14 Comfort Combo": WinCC flexible 2008 Standard
- With the license "WinCC V14 Advanced Combo": WinCC flexible 2008 Advanced

4.2.4.3 Licenses and Powerpacks

Licensing of WinCC Engineering System

You require a license key for the following:

- WinCC Engineering System, for example, WinCC Professional
- Add-ons for WinCC Engineering System

You can install the license key during the installation of WinCC. You transfer licenses for the WinCC add-ons after installation with the Automation License Manager.

Starting without a valid license key

If you start WinCC without a valid license, the system alerts you that you are working in non-licensed mode. You have the one-time option of activating a trial license. The trial license for Engineering editions WinCC Basic, Comfort/Advanced, and Professional expires after 21 days.

When the trial license has expired, the following scenarios may occur:

- WinCC was never licensed on the PC in question.
  - Operations requiring a license can no longer be performed in WinCC.
- WinCC was already licensed on the PC in question.
  - Non-licensed mode is indicated every 10 minutes and at every action requiring a license by a window, which you must acknowledge.

License requirements for simulation

When you want to start simulation in WinCC using the "Online > Simulation> With tag simulator" menu command, you do not need licenses for WinCC Runtime or for add-ons that require a license.

If the following conditions are met, you will need the appropriate licenses for WinCC Runtime and add-ons that require a license even for simulation:

- The engineering station is connected to a PLC.
- The connection to the PLC is configured and active.

You start the simulation with the "Online > Simulation > Start" menu command.

Validity of license keys for older versions of WinCC

With a valid license key for WinCC Basic V14.x, you can also operate older versions of WinCC without restrictions:

Edition	License	Validity
WinCC Basic V14.x	WinCC Basic	• WinCC Basic V11.0 SP2 • WinCC Basic V12.0 • WinCC Basic V13.0 • WinCC Basic V14.0

See also

Software and hardware requirements (Page 135)

Licensing of HMI devices (Page 140)

Working with license keys (Page 141)

Licensing of HMI devices

Non-PC-based HMI devices are always equipped with "Runtime licenses". A license key is not required for runtime operation.

A license may be required for an add-on for non-PC-based HMI devices. The license key of the respective license always activates one add-on for use.

License key

In order to license non-PC-based HMI devices with license keys, the "SIMATIC HMI License Manager Panel Plugin" add-on is needed.

WinCC Setup installs this add-on by default. You open the License Manager Panel Plugin in the Automation License Manager with the "Edit > Connect Target System > Connect HMI Device" menu command.

If WinCC is not installed, an installation of ProSave 7.2 or higher is required.

Note

Further information on license handling can be found in the Automation License Manager help.

Note

Verify that the current version of the operating system (or higher) is installed on the HMI device before you start licensing. If necessary, update the operating system using ProSave.

Non-licensed mode

Runtime add-ons can be used for a short time without restrictions without a valid license. Non-licensed mode is indicated every 10 minutes by a window, which you must acknowledge.

See also

Licensing of WinCC Engineering System (Page 139)

Working with license keys

Introduction

You transfer a license key to the HMI device in the following cases:

- To use the WinCC engineering system
- To operate WinCC Runtime
- To use add-ons for WinCC Runtime on PC-based HMI devices
- To use add-ons on non-PC-based HMI devices

You transfer a license key from the HMI device in the following cases:

- When backing up data
- If you no longer require the license

You can then use this license on another PC or HMI device.

When you transfer a license to an HMI device, the associated license key is removed from the license key storage location.

Note

A license key cannot be copied. The copy protection employed prevents the license keys from being copied.

Data backup

You transfer the license keys from the HMI device for data backup on the HMI device or as a backup for device replacement.

You use the Automation License Manager to back up license keys from an HMI device to the storage area of the license keys.

NOTICE

Destruction of license keys on non-PC-based HMI devices

License keys transferred as a result of backup/restore operations are destroyed in the case of the following HMI devices.

- 270 series
- 370 series

Carry out the following before beginning restoring:

- Use the Automation License Manager and ProSave to check whether license keys are on the HMI device.
- Transfer the license keys present on the HMI device to a storage location.
  After restoring has been carried out, transfer the license keys back to the HMI device.

NOTICE

Destruction of license keys on PCs

Transfer all license keys to a storage location in the following cases:

- Before you format the hard disk
- Before you compress the hard disk
- Before you restore the hard disk
- Starting an optimization program that moves fixed blocks
- Installing a new operating system

Read the description of Automation License Manager ("Start > Simatic Automation > Documentation"). Observe all warnings and notices.

The license key storage location on PC-based HMI devices and on non-PC-based HMI devices where Automation License Manager is used may contain multiple license keys. This capability

means you can store multiple licenses of the same type at one location. For the backup, use a single storage location for all license keys present on an HMI device.

Note

Always keep the original storage location of the license keys.

Invalid license after time zone change

The transferred license no longer functions in the following case.

- If you change the time zone on a WinCC PC as follows:
  - From a time based on a complete hour to a time not based on a complete hour. Example: You change the time zone from GMT +3:00 to GMT +3:30.

To work around this, transfer the license key from the HMI device with a time zone setting that was set when the license key was transferred to the HMI device.

Example:

You transferred the license key to the HMI device with a time zone setting based on a full hour. Then, also transfer the license key from the HMI device with a time zone setting based on a full hour.

This behavior does not apply to the trial license.

Defective license

A license is defective in the following cases:

- If the license key is no longer accessible at the storage area.
- If the license key disappears during its transfer to the destination drive.

Note

Resetting of the system date to an earlier time causes all licenses to be become defective.

You can use the Automation License Manager to repair the defective license. Use the "Restore" function or the "Restore Wizard" of the Automation License Manager for this purpose. You must contact Customer Support in order to restore the license. For additional information see: http://support.automation.siemens.com (http://support.automation.siemens.com)

Note

The Runtime software can also be operated without errors if the license is missing or defective. The system alerts you at brief intervals that you are working in non-licensed mode.

Note

If you start the WinCC Engineering System without a valid license key, the system alerts you that you are working in non-licensed mode. You have the one-time option of activating a trial license. The trial license expires after 21 days.

When the trial license has expired, the following scenarios may occur:

- WinCC was never licensed on the PC in question.
  WinCC can no longer be started.
- WinCC was already licensed on the PC in question.
  WinCC can be started. Non-licensed mode is indicated every 10 minutes by a window, which you must acknowledge.

See also

Licensing of WinCC Engineering System (Page 139)

4.3 Installation log

Function of the installation log

The progress during the following installation processes is logged in a file:

- Installing products
- Modifying or updating already installed products
- Repairing an existing installation
- Uninstalling products

If errors occur during the installation process or warnings are issued, these can be evaluated with the help of the log file. You can do this yourself or contact product support.

Installation logs storage location

The log file is the most recent file with the file extension ".log" and the name of which that starts with "SIA".

The location of the log file is stored in the environment variable "%autinstlog%". You can enter this environment variable in the address bar of Windows Explorer to open the folder with the log files. Alternatively, you can navigate to the corresponding directory by entering "CD %autinstlog%" in the command line.

The storage location is dependent on the operating system, e.g. "C:\ProgramData\Siemens\Automation\Logfiles\Setup" in English-language Windows.

Setup_Report (CAB file)

To make it easier to provide Product Support with all necessary files, an archive file that contains the installation log and all other required files is saved in CAB format. This log can be found at "%autinstlog%\Reports\Setup_report.cab". Send this CAB file to Product Support if you need assistance with installation. With this information, Product Support can determine whether the installation was executed properly. CAB files that were generated during earlier installation processes are saved with a date ID in the "Reports" directory.

See also

Notes on licenses (Page 125)

Starting installation (Page 146)

Checking availability of updates and support packages and installing them (Page 155)

Displaying the installed software (Page 148)

Modifying or updating installed products (Page 149)

Repairing installed products (Page 151)

Starting to uninstall (Page 153)

4.4 Starting installation

Introduction

Software packages are installed automatically by the setup program. The setup program starts once the installation medium has been inserted in the drive.

Requirement

- Hardware and software of the programming device or PC meet the system requirements.
- You have administrator privileges on your computer.
- All running programs are closed.

Procedure

To install the software packages, follow these steps:

1. Insert the installation medium in the relevant drive.
   The setup program starts automatically unless you have disabled Autostart on the programming device or PC.
2. If the setup program does not start up automatically, start it manually by double-clicking the "Start.exe" file.
   The dialog for selecting the setup language opens.
3. Choose the language in which you want the setup program dialogs to be displayed.
4. To read the information on the product and installation, click the "Read Notes" or "Installation Notes" button.
   The help file containing the notes opens.
5. Once you have read the notes, close the help file and click the "Next" button.
   The dialog for selecting the product languages opens.
6. Select the languages for the product user interface, and click the "Next" button.

   Note

   "English" is always installed as the basic product language.

   The dialog for selecting the product configuration opens.
7. Select the products you want to install:
   - If you wish to install the program in a minimal configuration, click on the "Minimal" button.
   - If you wish to install the program in a typical configuration, click on the "Typical" button.
   - If you wish to personally select the products to be installed, click on the "User-defined" button. Then select the check boxes for the products you wish to install.
8. If you want to create a shortcut on the desktop, select the "Create desktop shortcut" check box.
9. Click the "Browse" button if you want to change the target directory for the installation. Note that the length of the installation path must not exceed 89 characters.

10. Click the "Next" button.
    The dialog for the license terms opens.
11. To continue the installation, read and accept all license agreements and click "Next".
    If changes to the security and permission settings are required in order to install the TIA Portal, the security settings dialog opens.
12. To continue the installation, accept the changes to the security and permissions settings, and click the "Next" button.
    The next dialog displays an overview of the installation settings.
13. Check the selected installation settings. If you want to make any changes, click the "Back" button until you reach the point in the dialog where you want to make changes. Once you have completed the desired changes, return to the overview by clicking on "Next".
14. Click the "Install" button.
    Installation is started.

    Note

    If no license key is found during installation, you have the chance to transfer it to your PC. If you skip the license transfer, you can register it later with the Automation License Manager.

    Following installation, you will receive a message indicating whether the installation was successful.

15. It may be necessary to restart the computer. If this is the case, select the "Yes, restart my computer now." option button. Then click "Restart".
16. If the computer does not reboot, click "Exit".

Result

The TIA Portal along with the products and licenses you have ordered and the Automation License Manager have been installed on your computer.

See also

Installation log (Page 145)

Notes on the system requirements (Page 126)

Notes on licenses (Page 125)

Displaying the installed software (Page 148)

Modifying or updating installed products (Page 149)

Repairing installed products (Page 151)

Starting to uninstall (Page 153)

4.5 Displaying the installed software

You can find out which software is installed at any time. In addition, you can display more information on the installed software.

Procedure

To display an overview of the software installed, follow these steps:

1. Click "Installed software" in the "Help" menu.
   The "Installed software" dialog opens. You will see the installed software products in the dialog. Expand the entries to see which version is installed in each case.
2. If you would like to display additional information on the installed automation software, click the link on the "Detailed information about installed software" dialog.
   The "Detailed information" dialog opens.
3. Chose the topic you want more information about in the area navigation.

See also

Notes on the system requirements (Page 126)

Notes on licenses (Page 125)

Starting installation (Page 146)

Modifying or updating installed products (Page 149)

Repairing installed products (Page 151)

Starting to uninstall (Page 153)

Installation log (Page 145)

4.6 Modifying or updating installed products

You have the option to modify installed products using the setup program or to update to a new version.

Blocks with know-how protection from earlier versions of the TIA Portal are not automatically upgraded with the project. Remove the know-how protection of the blocks before you update the TIA Portal. Then set up the know-how protection with the current version of TIA Portal. For more detailed information, refer to the information system.

Requirement

- Hardware and software of the programming device or PC meet the system requirements.
- You have administrator privileges on your computer.
- All running programs are closed.

Procedure

To modify or update installed products, follow these steps:

1. Insert the installation medium in the relevant drive.
   The setup program starts automatically unless you have disabled Autostart on the programming device or PC.
2. If the setup program does not start up automatically, start it manually by double-clicking the "Start.exe" file.
   The dialog for selecting the setup language opens.
3. Choose the language in which you want the setup program dialogs to be displayed.
4. To read the information on the product and installation, click the "Read Notes" or "Installation Notes" button.
   The help file containing the notes opens.
5. Once you have read the notes, close the help file and click the "Next" button.
   The dialog for selecting the installation variant opens.
6. Select the "Modify/Upgrade" option button and click the "Next" button.
   The dialog for selecting the product languages opens.
7. Select the check boxes of the product languages that you want to install. You can remove previously installed product languages by clearing the corresponding check boxes.

   Note

   Note that the product language "English" cannot be removed.

8. Click the "Next" button.
   The dialog for selecting the product configuration opens.
9. Select the check boxes of the components that you want to install. You can remove previously installed components by clearing the corresponding check boxes.

10. Click the "Next" button.

    Note

    Note that you cannot change the target directory because the existing installation is being modified.

    If changes to the security and permission settings are required in order to install the TIA Portal, the security settings dialog opens.

11. To continue the installation, accept the changes to the security and permissions settings, and click the "Next" button.
    The next dialog displays an overview of the installation settings.

12. Click the "Modify" button.
    This starts the installation of the additional components.

    Note

    Following installation, you will receive a message indicating whether the existing installation was successfully changed.

13. It may be necessary to restart the computer. If this is the case, select the "Yes, restart my computer now." option button. Then click "Restart".

14. If the computer does not reboot, click "Exit".

Result

The existing installation has been modified on your computer.

See also

Notes on the system requirements (Page 126)

Notes on licenses (Page 125)

Starting installation (Page 146)

Displaying the installed software (Page 148)

Repairing installed products (Page 151)

Starting to uninstall (Page 153)

Installation log (Page 145)

4.7 Repairing installed products

You have the option to repair installed products by completely reinstalling them using the setup program.

Requirement

- Hardware and software of the programming device or PC meet the system requirements.
- You have administrator privileges on your computer.
- All running programs are closed.

Procedure

To repair installed products, follow these steps:

1. Insert the installation medium in the relevant drive.
   The setup program starts automatically unless you have disabled Autostart on the programming device or PC.
2. If the setup program does not start up automatically, start it manually by double-clicking the "Start.exe" file.
   The dialog for selecting the setup language opens.
3. Choose the language in which you want the setup program dialogs to be displayed.
4. To read the information on the product and installation, click the "Read Notes" or "Installation Notes" button.
   The help file containing the notes opens.
5. Once you have read the notes, close the help file and click the "Next" button.
   The dialog for selecting the installation variant opens.
6. Select the "Repair" option button, and click the "Next" button.
   The next dialog displays an overview of the installation settings.
7. Click the "Repair" button.
   This starts the repair of the existing installation.

   Note

   Following installation, you will receive a message indicating whether the existing installation was successfully repaired.

8. It may be necessary to restart the computer. If this is the case, select the "Yes, restart my computer now." option button. Then click "Restart".
9. If the computer does not reboot, click "Exit".

Result

The installed products have been reinstalled.

See also

Notes on the system requirements (Page 126)
Notes on licenses (Page 125)
Starting installation (Page 146)
Displaying the installed software (Page 148)
Modifying or updating installed products (Page 149)
Starting to uninstall (Page 153)
Installation log (Page 145)

4.8 Starting to uninstall

Introduction

Software packages are removed automatically by the setup program. Once started, the setup program guides you step-by-step through the entire removal procedure.

You have two options for removing:

- Removing selected components via the Control Panel
- Removing a product using the installation medium

Note

The Automation License Manager will not be removed automatically when you remove the software packages, because it is used for the administration of several license keys for products supplied by Siemens AG.

Removing selected components via the Control Panel

To remove selected software packages, follow these steps:

1. Open the Control Panel with "Start > Control Panel".
2. Click "Programs and Features".
   The "Uninstall or change a program" dialog opens.
3. Select the software package to be removed and click the "Uninstall" button.
   The dialog for selecting the setup language opens.
4. Select the language in which you want the dialogs of the setup program to be displayed and click "Next".
   The dialog for selecting the products you want to remove opens.
5. Select the check boxes for the products that you want to remove and click "Next".
   The next dialog displays an overview of the installation settings.
6. Check the list with the products to be removed. If you want to make any changes, click the "Back" button.
7. Click the "Uninstall" button.
   Removal begins.
8. It may be necessary to restart the computer. If this is the case, select the "Yes, restart my computer now." option button. Then click "Restart".
9. If the computer does not reboot, click "Exit".

Removing a product using the installation medium

To remove all software packages, follow these steps:

1. Insert the installation medium in the relevant drive.
   The setup program starts automatically unless you have disabled Autostart on the programming device or PC.
2. If the setup program does not start up automatically, start it manually by double-clicking the "Start.exe" file.
   The dialog for selecting the setup language opens.
3. Select the language in which you want the setup program dialogs to be displayed.
4. To read the information on the product and installation, click "Read product information" or "Read installation notes".
   The help file containing the notes opens.
5. Once you have read the notes, close the help file and click the "Next" button.
   The dialog for selecting the installation variant opens.
6. Select the "Uninstall" option button and click the "Next" button.
   The next dialog displays an overview of the installation settings.
7. Click the "Uninstall" button.
   Removal begins.
8. It may be necessary to restart the computer. If this is the case, select the "Yes, restart my computer now." option button. Then click "Restart".
9. If the computer does not reboot, click "Exit".

See also

Installation log (Page 145)

Notes on the system requirements (Page 126)

Notes on licenses (Page 125)

Starting installation (Page 146)

Displaying the installed software (Page 148)

Modifying or updating installed products (Page 149)

Repairing installed products (Page 151)

4.9 Installing updates and support packages

4.9.1 Checking availability of updates and support packages and installing them

By default, the TIA Portal checks automatically if new software updates or support packages are available, for example, Hardware Support Packages (HSPs). The automatic search for updates takes place after each computer restart and then cyclically every 24 hours. You can also deactivate the automatic search at any time or reactivate it. You can also search for updates manually.

If updates are found, you can download and install them.

Note

Updates and support packages from TIA Portal V13 or higher are supported.

Deactivate or activate automatic search for software updates

To deactivate or reactivate the automatic search for software updates, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General > Software Updates" group in the area navigation.
3. Deselect the "Check for updates daily" check box if you want to deactivate the automatic search for software updates.
4. Select the "Check for updates daily" check box if you want to reactivate the automatic search for software updates.

Manually searching for software updates

If you want to search for software updates manually, follow these steps:

1. Click "Installed software" in the "Help" menu.
   The "Installed software" dialog opens.
2. Click "Check for updates".
   The "TIA Updater" opens and the available updates are displayed.

Or:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General > Software Updates" group in the area navigation.
3. Click "Check for updates now".
   The "TIA Updater" opens and the available updates are displayed.

Or:

1. Open the TIA Updater with "Start > All Programs > Siemens Automation > Totally Integrated Automation UPDATER".
2. Click "Check for updates".
   The available updates are displayed.

Setting the server

You must set the corresponding server depending on whether you want to download updates and/or support packages from the TIA Automation Update Server or from a company-internal server. To do this, follow these steps:

1. Open the TIA Updater.
2. Click "Options" and select the "TIA Automation Software Update Server" option in the dialog that appears in the "Server used to check for updates" area.
   The software searches for available updates on the server of the manufacturer.

Or:

1. Open the TIA Updater.
2. Click "Options" and select the "User-defined server" option in the dialog that appears in the "Server used to check for updates" area.
3. Enter the server URL which you received from your administrator.
   The software searches on your company's server for available updates.

You can set a different server on which the updates are to be searched for at any time. However, changing this setting is blocked during a download process. After switching the server, all downloaded updates and support packages are displayed, even if they are not available on the currently set server.

Downloading and installing software updates separately

To download and install available software updates and support packages, follow these steps:

1. If the TIA Updater is not displayed, open it in one of these ways:
   - If the automatic search has found updates, you receive a message in the information area of the Windows taskbar and the icon for the TIA Update is displayed in the information area. Click either the message or the icon to open the TIA Updater.
   - Click "Installed software" in the "Help" menu and then "Check for updates".
   - In the Windows Start menu, click "Start > All Programs > Siemens Automation > Automation Software Updater".
2. Click "Download" in the line of the update or support package that you want to install. The update or support package is downloaded. The associated "Install" button becomes active as soon as the download process has been completed.

   Note

   Please note the following:

   1. You can initiate multiple download processes simultaneously.
   2. You can log off or also shut down the computer while the download process is running. In such cases, the download process continues in the background as soon as you log on again.
   3. In some cases, a link to an external web page is displayed instead of the "Download" button. In this case, download the software update from this site and install it manually.

3. Close the TIA Portal if it is still open.
4. In the TIA Updater, click the "Install" button of the software package you want to install. The installation dialog appears.

   Note

   Please note the following:

   1. It is not possible to install multiple updates simultaneously.
   2. Do not log off during an installation and do not shut down the computer. This avoids inconsistent versions of the software on your computer.

5. Click "Next".
   The selected product is installed.

Downloading and installing several support packages in one step

To download and install available support packages in one step, proceed as follows:

1. Activate the required lines. You have several options for this:
   - To download and install all displayed support packages, activate the check mark in the title of the list.
   - To download and install several support packages, select the first required support package, hold down the <Ctrl> key and then select all further required support packages. Then set the check mark in one of the selected lines.
   - To download and install several support packages in a certain order, select the first required support package, hold down the <Shift> key and then select the last required support package.
2. Click the "Download and Install" button.
3. After downloading click "Next" in the displayed dialog to start the installation. A dialog box is then displayed which informs you about the progress of the installation.

Note

Only support packages that are not subject to export restrictions can be downloaded and installed.

Alternative procedures for the installation of support packages

Another procedure is available for the installation of a support package. To do this, follow these steps:

1. Click "Support packages" in the "Options" menu of the TIA Portal.
   The "Detailed information" dialog opens. A table lists all support packages from the directory that you selected as the storage location for support packages in the settings.
2. If you want to install a support package that is not in the list, you have the following options:
   - If the support package is already on your computer, you can add it to the list by selecting "Add from file system".
   - If you add a support package from the "Service & Support" page on the Internet, first you download it by selecting "Download from the Internet". Then you can add it from the file system.
3. Select the support package that you want to install.
4. Click "Install".
5. Close and then restart the TIA Portal.

See also

Installation log (Page 145)

4.9.2 Providing updates on a company-internal server

Introduction

Using the TIA Updater Corporate Configuration Tool, you can configure a company-internal server on which the available updates/support packages are stored and can be made available to the users.

Requirement

You must have administrator rights.

Adding updates from the TIA Automation Software Update Server

To add updates from the TIA Automation Software Update Server, follow these steps:

1. Open the TIA Updater Corporate Configuration Tool.
   The available updates are displayed.
2. Click the "Add update" button and select the "Add TIA Automation Software Update Server" check box in the dialog that appears.
3. In the dialog "Add update from TIA Automation Software Update Server", select the required updates (software and support packages) and click "Add".
   If updates are already located on the company-internal server, these are grayed out. You cannot load them again. During the download process, the status and the remaining time is displayed in the dialog.

Canceling the download process

To cancel the process, follow these steps:

1. Click the "Cancel download" button.
2. Click "Yes" to confirm the dialog that appears.
   The download of the update is cancelled and it is deleted from the list.

Deleting updates from the company-internal server

To delete updates, follow these steps:

1. Select the required updates and click the "Remove" button.
2. Click "Yes" to confirm the dialog that appears.
   The selected updates have been deleted from the list.

Server options

In the dialog "TIA Automation Software Update Server", click "Settings" to determine the following:

1. Under "Server path", specify the folder in which the downloads are to be saved. You can select either a local drive or a network drive.

   Note

   When you work directly on the server, the target directory is a equivalent to the provision directory on the server. All changes are applied directly. This procedure is not recommended, as conflicts can occur with active downloads.

   We recommend that you work on another computer; the target directory can be any directory. The content of this directory must then still be copied to the provision directory. Here, also, make sure that no conflicts occur with active downloads.

2. Select the check box "Always run server in background (icon is shown in the taskbar)", if the server is to always run in the background.
3. Confirm your entries with "OK".

4.9.3 Configuring a company-internal server for updates

Introduction

To provide available updates and support packages to users from a central location, you must configure a company-internal server. Use the Microsoft Server Manager to do this. The procedure for creating and configuring the server with the Microsoft Server 2008 R2 operating system is described below.

Note

Please note that the structure of the start menu and the storage location of the programs may vary depending on the different operating systems.

For more detailed information on operating the Server Manager, refer to the Microsoft help.

Requirement

You must have administrator rights.

Install Web server role (IIS)

To install the required Web server role, follow these steps:

1. Open the start menu by clicking the command "Start" > "All programs" > "Administration" and select the "Server Manager".
2. Click "Add roles" under "Role overview".

3. Use the "Add roles" wizard to add the Web server roles.
   Before the wizard starts, it is verified that a complex password has been assigned to the administrator, that the network settings have been configured accordingly and that the latest Windows security updates have been installed.
4. Note the pre-selected role services that are installed by default, and then select the additional role service ASP.NET.
5. Select the required features and click "Next".
6. To complete the installation, click "Close".
   The Web server role has been added.

Create website

1. Click "Roles > Web Server (IIS) > Internet Information Services (IIS) Manager".
2. In the "Bindings" area, right-click "Sites" and then click "Add website".
3. In the "Add website" dialog box, enter a display name for the website in the "Website name" field, for example "TIAPortalUpdates".
4. In the "Physical path" field, enter the physical path of the folder in which the website is located or click the button to browse (...), in order to search for the folder in the file system.
5. Click "OK".

Check website

1. In the area "Connections > Sites", select the website you created. Activate the "Browse directory" entry in the "Features" view.
2. To check whether the website is displayed with content, select the menu command "Manage website > Browse". The updates available in the virtual directory are now displayed in the dialog that appears.

Add MIME type

1. Double-click "MIME type" in the "Features" view.
2. Click "Add" in the "Actions" area.
3. Enter .* in the "File name extension" text field in the "Add MIME type" dialog.
4. Enter a MIME type in the "MIME type" text field. For example, enter "all/files".
5. Click "OK".

Add BITS-IIS server extension

1. Click "Server Manager" in the tree view.
2. In the "Feature overview" section, click "Add features".
3. In the wizard for adding features, select the options "Background Intelligent Transfer Service (BITS)" and "IIS server extension".
4. Follow the instructions of the wizard, which may also include installing the required software.

Activate BITS-IIS server extension

1. Switch to the virtual directory in the IIS Manager.
2. Double-click "BITS-Uploads" in the Features view of the virtual directory.
3. Select the "Allow clients to upload files" check box and click "Apply".

Create a self-signed server certificate

1. In the "Connections" area, navigate to the level which you want to manage.
2. Double-click "Server Certificates" in the "Features" view.
3. Click "Create Self-Signed Certificate" in the "Actions" area.
4. In the "Create Self-Signed certificate" dialog, enter a display name for the certificate in the "Specify a friendly name for the certificate" field and click "OK".

Create SSL binding

1. Expand the "Sites" entry in the "Connections" area, and then click the site to which you want to add a binding.
2. Click "Bindings" in the "Actions" area.
3. In the "Site Bindings" dialog, click "Add".
4. In the "Add Site Binding" dialog, select the certificate you created under "Type" "https" and under "SSL certificate" and then click "OK".

Verify SSL binding

1. In the "Actions" area, click the previously created binding under "Browse website".
   An error page is opened in Internet Explorer, because the self-signed certificate was created by your computer.
2. Click "Continue to this website (not recommended)".
   This message is no longer displayed when you add the certificate to the "Trusted Root Certification Authorities" certificate memory.

Configure SSL settings

1. Double-click "SSL settings" in the "Features" view.
2. Use one of the following procedures in the "Client certificates" area of the "SSL Settings" dialog:
   - Select "Ignore" if no client certificates are to be accepted even if a client has a certificate.
   - Select "Accept" if client certificates are to be accepted.
   - Select "Require" if client certificates are to be required. The "Require SSL" option must be activated to allow the "Require client certificates" option to be used.
3. Click "Apply" in the "Actions" area.

Distributing updates to different areas

It may be the case that different departments need different updates or support packages. In this situation, we recommend operating multiple servers which offer different updates and support packages. Proceed as follows:

1. Set up different company-internal servers as described above.

or

1. Install the Web server role.
2. Set up multiple websites.
   Make sure that you assign unique names for these websites and the physical paths so that there will be no confusion.
3. Set the features described above for the Web server or the websites.

You can now store the required updates and support packages for the different departments in the defined directories.

4.10 Installing and uninstalling the migration tool

4.10.1 System requirements

System requirements for the migration tool

The following system requirements apply to the use of the migration tool:

- All products used to create the source project must be installed. The following products are supported:
  - STEP 7 V5.4 SP5
  - WinCC V7.3 with the latest updates
  - WinCC flexible 2008 SP3
  - Integrated projects from STEP 7 V5.4 and the WinCC products listed above
  - STEP 7 Distributed Safety V5.4
  - SINUMERIK STARTER and Startdrives
  - As of SIMOTION SCOUT V4.4
    You need the SCOUT Migration Tool PlugIn in the corresponding SCOUT TIA version to migrate SIMOTION SCOUT projects.
- All optional packages needed to process the STEP 7 project are installed. For example, all HSPs for the devices used in the source project are required.

4.10.2 Installing the migration tool

Distribution of the migration tool

You can find the migration tool in the "Support" directory on the installation DVD of the TIA Portal. Alternatively, it is available for download from the Siemens Industry Online Support. For some products, (such as SIMATIC Failsafe or SIMOTION) additional plug-ins are required for the migration tool. These plug-ins can also be downloaded from the Siemens Industry Online Support or installed from the installation DVD of the specific products.

Normally, the migration tool is installed without the TIA Portal. Because the TIA Portal has its own integrated migration function, a separate installation of the migration tool is not necessary.

Procedure

To install the migration tool, proceed as follows:

1. Download the installation file from the Siemens Industry Online Support, or use the installation file from the "Support" directory of the installation DVD of the TIA Portal to perform the installation.
2. Run the installation file.
   The setup program for the migration tool will open.
3. First, select the language in which the setup should be displayed and click the "Next" button.
   The page for selecting the software language is displayed.
4. Since the migration tool is provided exclusively in English, you cannot choose any other language for the installation. Therefore, click "Next" to proceed to the next step.
   The page for selecting the product is displayed.
5. The migration tool consists solely of a software component. Therefore, the migration tool is already selected.
   To create a Desktop icon for starting the migration tool, select the check box "Create Desktop icon". Then click the "Next" button.
   The page for confirming the licensing terms is shown.
6. Click on an entry in the list of license terms to read the selected license term. If you agree with all license terms, select the check box "I accept the terms of the displayed license agreement". Then click the "Next" button.
   An overview of the installation is displayed.
7. Click the "Install" button.
   The installation is performed with the displayed settings.

4.10.3 Uninstalling the migration tool

The migration tool can be removed using the Control Panel.

Procedure

To remove the migration tool, follow these steps:

1. Open the Control Panel.
2. Double click on "Add or Remove Programs" in the Control Panel.
   The "Add or Remove Programs" dialog opens.
3. Select the entry for the migration tool in the "Add or Remove Programs" dialog, and click the "Remove" button.
   A security prompt appears.
4. Click the "Uninstall" button to confirm this prompt.
   The migration tool will be removed.

4.11 Installing and uninstalling the multiuser server

4.11.1 Notes on the installation of the multiuser server

Introduction

Installation of the multiuser server gives you access to the Multiuser Engineering functionality that is available with TIA Portal V14.

You can install the multiuser server as follows:

- Together with the installation of the TIA Portal.
- As standalone installation by means of a separate installation process without TIA Portal.

Note

Administrator rights are required for the server installation.

Installation of the multiuser server with the TIA Portal

The multiuser server is installed together with the following TIA Portal products:

- SIMATIC STEP 7 Basic V14.0
- SIMATIC STEP 7 Professional V14.0
- SIMATIC WinCC Basic V14.0
- SIMATIC WinCC Comfort/Advanced V14.0
- SIMATIC WinCC Professional V14.0

The multiuser server is usually activated in the product selection by default and is then installed with the TIA Portal. Check the settings during installation of the TIA Portal. You can find the specific default settings for the multiuser server during setup when selecting the product in the directory "Tools". By selecting or clearing the options, you can specify if the multiuser server is to be installed along with the TIA Portal or not.

Installation of the multiuser server as standalone installation

The installation of the multiuser server can also be a standalone installation.

A description of automated installation is also available on the product DVD in the directory "Documents\Readme\<language directory>".

Identical TIA Portal versions during installation

When installing different TIA Portal products on various clients, make sure that all participants use the same versions of software products for the installation.

For example, if you have installed STEP 7 Professional, WinCC Advanced and Safety as well as the multiuser server with version V14.0, all other Team Engineering participants must also have installed this software in identical versions.

The service packs and updates must be installed for all products at the same time.

Installation path

Do not use any UNICODE characters (for example, Chinese characters) in the installation path.

Antivirus programs

During the installation, read and write access to installed files is necessary. Some antivirus programs block this access. We therefore recommend that you disable antivirus programs during the installation of the multiuser server and enable them again afterwards.

4.11.2 Installing the multiuser server

Introduction

The multiuser server V14 can be installed as part of the TIA Portal V14 installation or as standalone product.

A description of automated installation is also available on the product DVD in the directory "Documents\Readme\<language directory>".

Installation rules

Multiple versions of the multiuser server can be installed on a computer, but only one version of the multiuser server can be run with the corresponding version of the TIA Portal.

Example: The multiuser server V14.0 can only be run with TIA Portal V14.0.

Requirements for installation

The same requirements apply to the installation of the multiuser server V14.0 as for the installation of the TIA Portal V14.0.

For more details, see: AUTOHOTSPOT

Installing the multiuser server with the TIA Portal

Follow these steps for installation:

1. Note the requirements for the installation of the selected software package.
2. Start the setup program of the required product by inserting the installation medium into the respective drive.

3. Select the preferred settings and click the "Install" button.
4. Make sure that the multiuser server is selected in the product selection and observe the notes in the setup dialogs.

Result

The multiuser server is installed on your computer together with the TIA Portal.

Installing the multiuser server as standalone product

Follow these steps for installation:

1. Note the requirements for the installation of the multiuser server.
2. Start setup for the standalone installation of the multiuser server with a double-click on the self-extracting .exe file "TIA_Portal_Multiuser_Server_V14.exe". You can find these files in the directory "Support" on the corresponding product DVD.
3. Select the preferred settings and click the "Install" button.
4. Follow the instructions in the setup dialogs.

Result

The multiuser server is installed on your computer.

4.11.3 Uninstalling the multiuser server

Introduction

You can use the Control Panel to remove the multiuser server.

Requirements for removal

There is a multiuser server installed on your computer.

Uninstalling the multiuser server

Proceed as follows for removal:

1. Open the Control Panel.
2. Click on "Uninstall a program" in the Control Panel.
   The "Uninstall or change a program" dialog opens.
3. Right-click the entry for the "Multiuser server" in the "Uninstall or change a program" dialog, and click "Uninstall".
   A security prompt appears.
4. Click "Yes" to confirm this prompt.

Result

The multiuser server is removed from your computer.

Migrating projects and programs

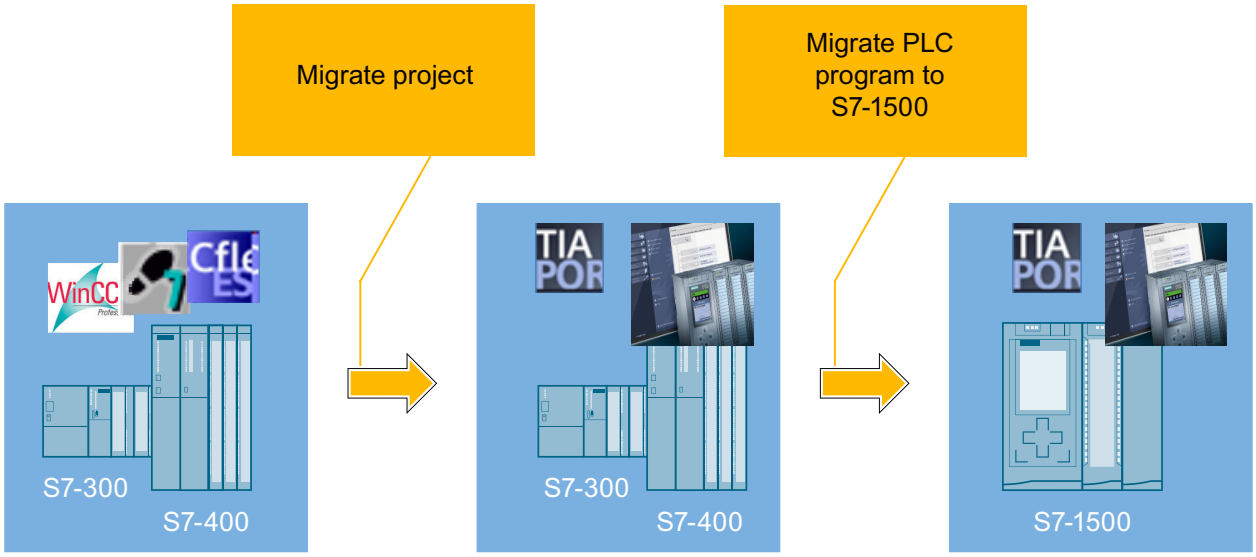
5

5.1 Overview of Migration Options

Migration paths

If you want to continue to use existing projects or programs with the latest version of the TIA Portal and the S7-1500, you have several options for migrating projects.

The following graphic provides an overview of the migration options:



Migrating projects to the TIA Portal

You use the "Migrate project" function to migrate projects that were created outside of the TIA Portal, for example, with STEP 7, WinCC, WinCC flexible, or SINUMERIK.

The result of the project migration is a TIA Portal project that you can use with your existing S7-300/400 series hardware and existing HMI devices.

Migrating PLC programs from S7-300/400 to S7-1500

To migrate a PLC program within the TIA Portal from an S7-300/400 series device to an S7-1500, you use the PLC migration.

The result of the PLC migration is an executable PLC program that is automatically adapted to the new system architecture of the S7-1500 as far as possible.

Optimizing PLC program for S7-1500

You also have the option to optimize your program for the S7-1500 by employing newly introduced programming methods. The optimization enables you to effectively use the higher performance, innovated memory technology and new features of the S7-1500 system.

Upgrading projects

You can also continue to use projects from previous versions of the TIA Portal. However, these projects do not need to be migrated. You can upgrade projects from previous versions to the current version of the product. You can find additional information on compatibility and upgrading projects in the section "AUTOHOTSPOT".

Note

Additional support for migration

You can find the latest information about PLC migration in the Siemens Industry online support:

Migration of complete systems:

http://support.automation.siemens.com/WW/view/en/83558085 (http://support.automation.siemens.com/WW/view/en/83558085)

Migration of controllers:

http://support.automation.siemens.com/WW/view/en/83557459 (http://support.automation.siemens.com/WW/view/en/83557459)

Migration of visualization:

http://support.automation.siemens.com/WW/view/en/76878921 (http://support.automation.siemens.com/WW/view/en/76878921)

Migration of communication:

http://support.automation.siemens.com/WW/view/en/83558087 (http://support.automation.siemens.com/WW/view/en/83558087)

If you need further support, please contact SIMATIC Customer Support.

5.2 Migrating projects to a TIA Portal project

5.2.1 Migration of projects with the TIA Portal

Migration of existing projects

You can migrate projects from earlier automation solutions to the TIA Portal. Each time you migrate, a new project is created for the migrated data with which you can then work. Any TIA Portal projects already open are closed first.

The migration is then displayed in the table of the project history. From there, you have access to the migration log that is created automatically for the migration.

After the migration of hardware configurations and program blocks from earlier automation solutions, first check the functionality of the migrated project before you use it in productive operation.

Supported products for migration

Refer to the chapter "System overview STEP 7 and WinCC" to find out which products are available for the TIA Portal. All products listed there are generally supported by the TIA Portal during migration.

Any additional requirements that must be met depend on the initial products that were used and the currently installed products. For more information on the migration options for your products, you can, for example, refer to the Siemens Industry Online Support and the documentation of your software products.

See also: Scaling of STEP 7 and WinCC in the TIA Portal (Page 37)

Procedure during migration

The migration process is divided into the following basic steps:

1. Preparing the initial project
   If the software for editing the initial project is not installed or not fully installed on the programming device/PC with the TIA Portal, or if the initial project is an integrated project, you must first convert the initial project to a migration file. To do this, install the migration tool on a programming device/PC on which the required software for editing the initial project is installed. Then, use the migration tool to convert the initial project, and copy the file to the programming device/PC on which the TIA Portal is installed. You can omit this step if the initial project and its associated software are on the same programming device/PC as the TIA Portal, and if the initial project is not an integrated project.
2. Performing migration
   Perform the migration within the TIA Portal. For the migration, specify as source either the migration file which you created with the migration tool or the initial project when all required software has been installed.

3. Checking the migration log
   A migration log is created for each migration. It contains information about modified project parts. You can call the log under "Common data > Logs" in the project tree or in the project history. After completion of the migration, the migration log will be displayed in the TIA Portal. Check the log following completion of the migration.
   If the migration failed, an XML file is created as a log under "\Logs" in the project directory. You can use any XML editor to open this log and view the reasons why the migration failed.
4. Correcting the migrated project
   Because the configurations of the initial project may not always be completely compatible with the TIA Portal, all configurations may not be reproduced identically in the migrated project. You should therefore work through the points in the migration log systematically. If you have not included the hardware configuration in the migration, you also have to convert the unspecified devices to the appropriate hardware.

Including the hardware configuration in the migration

By default, only the software parts of the project are included in the migration. An unspecified device is generated in the migrated project for the devices contained in the initial project. The hardware and network configurations and the connection are not migrated. You therefore convert the unspecified devices into suitable devices after the migration and recreate any network configurations and connections manually.

If you are certain that the hardware used in the initial project has a corresponding equivalent in the TIA Portal, you can include the hardware configuration in the migration. In this case, both the hardware configuration and the software are migrated. You can check with a tool (Page 174) to see which hardware components are supported.

See also

Display migration log (Page 180)

Scaling of STEP 7 and WinCC in the TIA Portal (Page 37)

5.2.2 Check migration readiness of hardware components

Introduction

Siemens offers a tool that can be used to check whether the hardware configuration used in an initial project is ready for migration to the TIA Portal.

Components integrated via GSD or GSDML files cannot be checked. For such modules, check manually in the TIA Portal whether the modules are available in the hardware catalog. If the modules are not available there, install the required GSD or GSDML files in the TIA Portal. You can obtain the required files from the manufacturer of the components.

In the result you can also see which software products and licenses have to be available on the programming device/PC with the installation of the TIA Portal to perform a migration. You can also see as of which firmware version the individual modules of the initial project are supported in TIA Portal. The result of the check can be output in a Microsoft Excel file or in PDF format.

Download

The tool for checking the migration readiness is available for download in the FAQs of the Siemens Industry Online Support under entry number 60162195 (http://support.automation.siemens.com/WW/view/en/60162195).

Source files for the check

To check readiness, you require one of the following source files which contains the article number of the hardware used in the initial project:

- .cfg file
  You can export the .cfg file from HW Config (STEP 7) with the menu command "Export as .cfg file" in the "Station" menu. The .cfg file contains all MLFBs of the devices used in the currently open station.
- Microsoft Excel file (in .xls file format)
  Regardless of the initial project used, you can create a Microsoft Excel list containing all MLFBs of the devices you want to migrate.
- File in .csv format
  The article numbers that are to be checked can also be saved to a .csv file instead of a Microsoft Excel list. To do this, use a standard text editor and enter the MLFBs separated by comma and without space after the comma. Save the text file with the extension ".csv".

See also

Tool for checking migration readiness (http://support.automation.siemens.com/WW/view/en/60162195)

5.2.3 Preparing projects with the migration tool

5.2.3.1 Migrating projects with the migration tool

Preparation for migration

In many cases, a project that you wish to migrate will not be located on the same programming device/PC on which the latest version of the TIA Portal is installed. Therefore, the initial project must first be converted to a compatible format for the migration. The same applies to integrated projects.

After creation of the migration file, you copy the migration file to the programming device/PC on which the current version of the TIA Portal is installed. In the TIA Portal, you enter the migration file as the source for the migration and can create a project in the current file format of the TIA Portal.

Procedure for migration with the migration tool

The following steps are necessary to prepare a migration with the migration tool:

1. Install the migration tool on the programming device/PC where the source project is located. To do this, download the installation file from the Siemens Industry Online Support or install the migration tool from the setup DVD of the TIA Portal.
2. Start the migration tool, and use it to convert the source project to the migration file format with the file extension ".am14".
   For this step, make sure that all software needed to process the source project is installed on the programming device/PC. This also includes all necessary service packs, hardware support packages and all expansion software that is needed to process the initial project. If individual products are not installed it may not be possible to perform the migration or the migration may be incomplete.
3. Copy the migration file to the target system on which a current version of the TIA Portal is installed.
   Note that the target system must have been installed with all software needed to configure the complete set of devices contained in the migration.
4. Perform the migration within the TIA Portal and specify the migration file with the extension ".am14" as the source.
5. Once migration is complete, check the migration log and systematically work through the information provided there for the newly created project. Read the information in the Inspector window with special care after the first compilation of the configuration.

Including the hardware configuration in the migration

By default, only the software parts of the project are included in the migration. An unspecified device is generated in the migrated project for the devices contained in the initial project. The hardware and network configurations and the connection are not migrated. You therefore convert the unspecified devices into suitable devices after the migration and recreate any network configurations and connections manually.

If you are certain that the hardware used in the initial project has a corresponding equivalent in the TIA Portal, you can include the hardware configuration in the migration. In this case, both the hardware configuration and the software are migrated. You can check with a tool to see which modules are supported.

See also

Migration of projects with the TIA Portal (Page 173)

Migrating projects (Page 178)

Calling the migration tool (Page 177)

Creating a migration file (Page 177)

5.2.3.2 Calling the migration tool

Starting the migration tool

During the installation, a "Migration to TIA Portal V14" shortcut is created by default in the Start menu under "Siemens Automation > Migration Tool". Click this shortcut.

Alternatively, you can call the migration tool directly in Windows Explorer. During the installation, the migration tool is saved by default in one of the following folders:

- On a 64-bit operating system:
  C:\Program Files (x86)\Siemens\Automation\MIGTOOL_V14\Bin
- On a 32-bit operating system:
  C:\Program Files\Siemens\Automation\MIGTOOL_V14\Bin

To start the migration tool, click the "Siemens.Automation.MigrationApplication.exe" file in one of the directories.

See also

Creating a migration file (Page 177)

5.2.3.3 Creating a migration file

The section below describes how you can use the migration tool to convert the initial project to a migration file that can be read by the TIA Portal. Following conversion, this file is transferred to the target system and migrated there.

You can specify whether the migration file should contain the entire project, including the complete hardware configuration and the associated software, or whether you want to migrate the software only.

Requirement

- The suitable, original software with a valid license is installed for all configurations used in the initial project.
- The initial project is not provided with access protection.
- The initial project must be consistent, otherwise problem-free migration cannot be assured.

Procedure

To create the migration file, proceed as follows:

1. Select the path of the source file for the migration in the "Storage Location (Path)" field.
2. Specify the project parts that are to be migrated:
   - Select the "Include HW and Network data during the migration" check box to migrate not only the software but also the complete hardware parts and the network configuration of the project.
   - Select the "Copy SCADA runtime data" check box if you also want to migrate the runtime data, such as alarm archives, tag archives and user archives, in addition to the data of the engineering system.
3. Select the path and the file name for the migration file in the "Intermediate file" field.
4. Click the "Migrate" button.

Result:

A migration file is created. You now copy this file to the target system and migrate this file in the TIA Portal.

See also

Migrating projects (Page 178)

Calling the migration tool (Page 177)

Migrating projects with the migration tool (Page 175)

5.2.4 Migrating projects

Requirement

- A converted file in the format ".am14" is already available or the original software with a valid license is installed for all configurations used in the initial project.
- The initial project is not provided with access protection.
- The initial project must be consistent, otherwise problem-free migration cannot be assured.

Read the additional information on the requirements in the help for the respective products installed.

Note

System hibernation during the migration

While a migration is running, the system should not be changed to the standby or hibernate mode. Otherwise the migration will be aborted.

Procedure

To migrate a project, follow these steps:

1. Select the "Migrate project" command in the "Project" menu.
   The "Migrate project" dialog opens.
2. Specify the path and the file name for the project to be migrated in the "Source path" field.
   Select a project either in the ".am14" migration format or in the format of the initial project.
3. To include the hardware configuration in the migration, select the "Include hardware configuration" check box.
   If you have selected a migration file that was created with the migration tool, the check box cannot be selected. In this case, you must specify if you wish to include the hardware configuration in the migration before the conversion with the migration tool .
4. Select the "Copy WinCC Runtime Professional data" check box, if you also want to migrate the runtime data, such as alarm archives, tag archives and user archives, in addition to the data of the engineering system.
   If you have selected a migration file that was created with the migration tool, the check box cannot be selected. In this case, you must specify if you wish to include the SCADA runtime data in the migration before the conversion with the migration tool .
5. Select a name for the new project in the "Project name" box.
6. Select a path in the "Target path" box where the new project is to be created.
7. Enter your name or the name of another person responsible for the project in the "Author" field.
8. Enter a comment in the "Comment" box, if you require one.
9. Click "Migrate".

Result

The initial project is converted and a message appears after conversion is complete. The newly created project is then opened in the project view, and the migration log is opened in the TIA Portal.

Even if the migration failed, a project directory is created and a migration log in the form of an XML file is generated in this directory. The completion message that appears after the migration contains a link to this XML file. Click the link to open the XML file. Alternatively, you can find the XML file in the project directory under "\Logs".

See also

Post-editing integrated projects (Page 213)

Display migration log (Page 180)

Migrating projects with the migration tool (Page 175)

Creating a migration file (Page 177)

5.2.5 Displaying the history of the migration

If a project was created by migration, the migration will be listed in the table of the project history. You can open the migration log in the table. The time of the migration is also shown.

Procedure

To display the migration in an overview table, follow these steps:

1. Select the open project in the project tree.
2. Select "Properties" in the shortcut menu of the project.
   The dialog with the properties of the project opens.
3. Select the "Project history" group in the area navigation.
   The overview table is displayed.

5.2.6 Display migration log

A log is created for each successful migration. The log contains the following information:

- Migrated objects
- Modifications to objects made during migration
- Errors that occurred during migration
- In certain cases a link to more help with specific events.
  In this case, click the question mark to obtain more help.

Procedure

To display the log file of the migration, follow these steps:

1. Open the "Common data > Logs" folder in the project tree.
2. Double-click the desired log in the list.
   The contents of the log are displayed in the work area.

See also

Migration of projects with the TIA Portal (Page 173)

5.2.7 Migrating WinCC flexible projects

5.2.7.1 Basics (WinCC flexible)

Migration (WinCC flexible)

Introduction

You can continue to use projects in WinCC from WinCC flexible. The following version of WinCC flexible is supported:

- WinCC flexible 2008 SP2

The following sections describe the operating devices that are supported and the required conditions for a successful migration.

Projects from ProTool and from earlier versions of WinCC flexible cannot be migrated directly to WinCC. If you wish to continue to use such projects in WinCC, you must first migrate them to a supported version of WinCC flexible.

See also

Object support during migration (WinCC flexible) (Page 187)

Projects from Migrating WinCC flexible projects (WinCC flexible) (Page 183)

Compiling and loading a migrated project (WinCC flexible) (Page 185)

Migrating runtime data (WinCC flexible) (Page 197)

Migrating integrated projects (WinCC flexible) (Page 199)

Supported HMI devices (WinCC flexible) (Page 186)

Migration of data types (WinCC flexible) (Page 203)

Basics on migration (WinCC flexible)

Introduction

During migration, the project data from a WinCC flexible project are converted to the new format of WinCC. The data will not be evaluated to see if they are consistent in the project you want to migrate. If errors or warnings are output in a source project during compilation, these will not be resolved as part of the migration. This means you should be able to compile the project without errors prior to migration. Note the scope of a project during migration. The features of WinCC apply for migration. For more information on this topic, refer to the "Visualize Processes > Performance Features > Engineering System" chapter in the online help.

Unique object names

The objects are clearly identified by the folders in which they are contained in WinCC flexible. Screen elements in groups are clearly identified by the group name.

In WinCC, an object name must be unique within an HMI device. The name of screen elements must be unique within a screen.

The uniqueness of the name is verified during migration. If a name is not unique according to the new rule, the object in question will be renamed. A renamed object will receive the suffix "#Mign", where "n" stands for a sequential number.

Example:

In WinCC flexible, tags located in different folders may have the same name. In WinCC, the tag name must be unique on the configured HMI device. This means tags with the same name from different folders will be renamed during migration.

Tags are renamed as follows:

Before migration	After migration
Folder_1/Tag_1	Folder_1/Tag_1
Folder_1/Tag_2	Folder_1/Tag_2
Folder_2/Tag_1	Folder_2/Tag_1#Mig1
Folder_2/Tag_2	Folder_2/Tag_2#Mig1
Folder_3/Tag_1	Folder_3/Tag_1#Mig2
Folder_3/Tag_2	Folder_3/Tag_2#Mig2

Affected objects

The following objects are renamed if necessary:

- Screens
- Screen objects
- Recipes
- Tags

Cancelling migration

The migration is cancelled in the following cases:

- If the project to be migrated is opened in the engineering system or in Runtime.
- If not enough memory space is available on the hard disk to create a a copy for migration of the project.
- If the migration cannot address the project database due to problems with the installed SQL-Server.
- If the migration cannot address the project database due to missing user authorization.

- If you select the "*.hmi" file for the migration in an integrated project. You must select the "*.s7" file for the migration in an integrated project.
- If the project was created with a version not supported by the migration.

Saving the project in the migration format

You do not have to execute the migration of a WinCC flexible project completely on the PC on which the project is available. You can prepare the migration by saving the project in the migration format. The migration tool is available for saving a WinCC flexible project in the migration format. The migration tool exports the engineering data from the WinCC flexible project and saves the data in the migration format "*.AM11".

For the actual migration, copy the data in the migration format to a PC on which the TIA Portal is installed.

See the section "AUTOHOTSPOT" for more information on the migration tool.

Projects from Migrating WinCC flexible projects (WinCC flexible)

Introduction

When you migrate a project, data from a WinCC flexible project is loaded into a new project for WinCC. A new project is therefore created automatically for project migration. You cannot migrate to an existing project.

The migration can be started in both the Portal view and the Project view.

You should only migrate a project to a re-started TIA Portal.

Information on the migration of an integrated project can be found in the section Migrating integrated projects (WinCC flexible) (Page 199).

If you only want to save the project in migration format, you can use the migration tool. See AUTOHOTSPOT for additional information.

Note

Further support for migration

You find current information on migration in the Siemens Industry Online Support:

Migration of visualization

If you need further assistance, please contact the SIMATIC Customer Support.

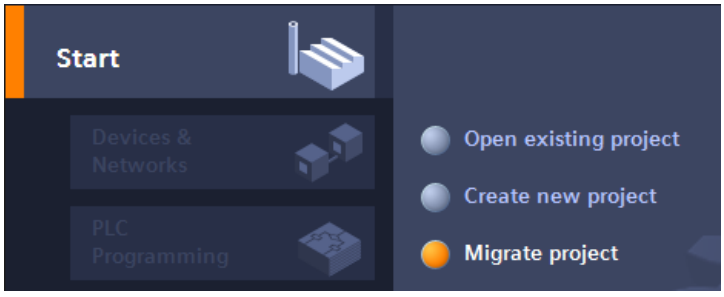
Requirement

- A project from WinCC flexible is available.
- The project is not open in WinCC flexible.

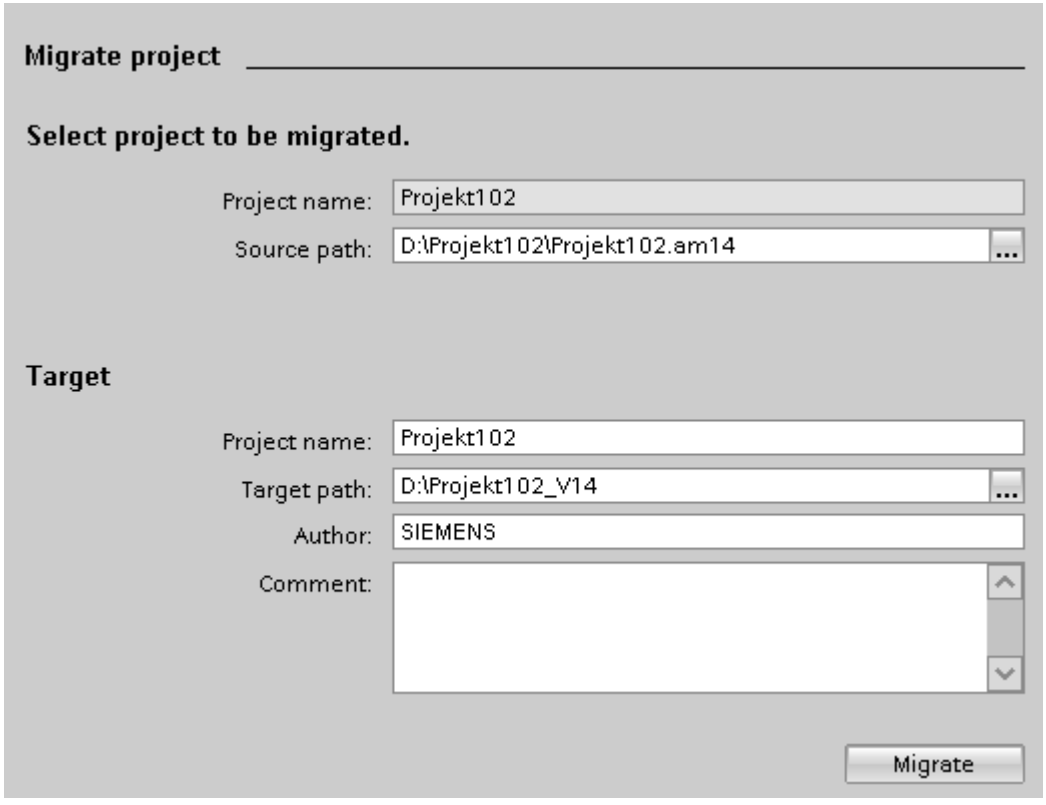
Procedure

Migrate a project in the Portal view as follows:

1. Select the action "Start > Migrate Project".



2. In the "Source path" box, navigate to the project you want to migrate.



3. Select the WinCC flexible project file "*.hmi" or "*.am14".

4. Change the information for the project to be created, if necessary. For example, change the project name or project path. The data to be migrated is created in the new project.
5. Click "Migrate".
   A new project is created and migration of the data is started:
   - The Project view opens.
   - The progress of the migration is shown in a migration window.
   - Warnings and errors about the migration process are displayed in the Inspector window under "Info > General".
   - All information about the migration is saved in a log file.
   - The project is saved and a message displayed upon completion of the migration. The message contains a link that you can use to open the log file.

When migration is complete, you will find a newly created device for each migrated HMI device in the project tree. These devices contain the migrated data, such as screens, alarms and tags.

Opening the migration log at a later point in time

The migration log is saved together with the migrated project. You can view the log at a later point in time. Open the log file as follows:

1. Open the "Common data > Logs" folder in the project tree. It contains the logs of all previously performed migrations.
2. Double-click the required migration log.
   The log is opened.

See also

Migrating integrated projects (WinCC flexible) (Page 199)

Compiling and loading a migrated project (WinCC flexible)

Compiling a migrated project

Once you have successfully migrated a WinCC flexible project, you need to recompile it before loading it to the HMI device. The project will only compile successfully if it was capable of error-free compiling prior to migration.

If errors occur during compilation of the migrated project, they have to be eliminated.

Once compiling is successfully completed, load the project to the HMI device.

Settings for download to the HMI device

The settings for loading the HMI device are not included in the migration. Once you have migrated the project, you must configure the settings for loading.

Select the HMI device in the project tree and select "Loading in device > Software (complete loading)" from the shortcut menu. The "Extended download to device" dialog opens. Configure

the required settings for the interface. Click the "Load" button. The project is recompiled and the dialog "Load preview" is opened.

Expand the "Overwrite" entry and verify the settings for the following options:

- Would you like to overwrite the existing user administration data from this device
- Would you like to overwrite the existing recipe data on HMI system

Configure the options as you want to use them in the project in the future. Subsequently, load the project to the HMI device.

5.2.7.2 Migrating engineering data (WinCC flexible)

Supported HMI devices (WinCC flexible)

Introduction

Note that WinCC only supports the following HMI device types when migrating projects from WinCC flexible:

- KTP400 Basic mono PN
- KTP400 Basic mono PN Portrait
- KTP600 Basic DP
- KTP600 Basic DP Portrait
- KTP600 Basic PN
- KTP600 Basic PN Portrait
- KTP600 Basic mono PN
- KTP600 Basic mono PN Portrait
- KTP1000 Basic DP
- KTP1000 Basic PN
- TP1500 Basic PN

WinCC only supports the functions provided by these HMI device types.

If your WinCC flexible project contains an HMI device that is not supported by WinCC, then the migration process will abort. To migrate the project, you must change the HMI device in WinCC flexible to a HMI device type supported by WinCC.

There may be some functions in a WinCC flexible project that are not supported by a Basic Panel, for example, because the device type has been switched. These unsupported functions are not migrated.

Adaptations before migration

If the HMI device has changed in the project being migrated, the project needs to be recompiled before migration. The compilation process will adjust the size of the screens and screen elements.

See also

Object support during migration (WinCC flexible) (Page 187)

Migration (WinCC flexible) (Page 181)

Migration of alarm classes and alarm groups (WinCC flexible) (Page 191)

Migration of language-dependent contents (WinCC flexible) (Page 193)

Migrating libraries (WinCC flexible) (Page 196)

Migration tags (WinCC flexible) (Page 190)

Changes of values of object properties by the migration (WinCC flexible) (Page 189)

Object support during migration (WinCC flexible)

Introduction

When migrating projects from WinCC flexible, all configuration data involving an HMI device supported by WinCC will be migrated. Basically, all object types and functions that are available and can be mapped to the new project environment will be fully migrated.

Some global object types are not migrated, for example, dictionaries and global libraries.

Supported object types

The following object types are supported for migration:

- Animations
- Scheduler
- User administration
- Area pointer
- Screens
- Screen template
- Data types
- Function lists
- Graphics lists
- Display and operating elements
  Migration supports all display and operating elements available on the supported HMI devices.
- Alarms
- Alarm classes
- Alarm groups
- Project library
- Project languages

- Recipes
- Runtime languages
- Runtime scripting
- System functions
- Texts
- Text lists
- Tags
- Connections

Unsupported object types

The following object types are not supported by migration:

- Global libraries
- Dictionaries
- Project versions
- Change log

Migration of the screen template

WinCC offers an extended concept for working with screen templates. WinCC offers a global screen and several templates for each device. During migration of a template from WinCC flexible, the objects contained there and the properties configured in the template are migrated to different templates of WinCC.

The following objects are migrated to the "global screen" of WinCC:

- Alarm window
- Alarm indicator
- Function keys of HMI devices with function keys

All other objects and properties are migrated to a template of WinCC.

The connection of the objects and properties to the respective template is automatically adapted.

Migration of system functions

The names of some system functions have changed in WinCC.

System functions which have changed their names are renamed.

This concerns the following system functions:

Function name in WinCC flexible	Function name in WinCC
IncreaseValue	IncreaseTag
DecreaseValue	DecreaseTag
SetValue	SetTag

See also

Supported HMI devices (WinCC flexible) (Page 186)

Changes of values of object properties by the migration (WinCC flexible) (Page 189)

Changes of values of object properties by the migration (WinCC flexible)

Introduction

The standardization of object properties from WinCC V7 and WinCC flexible requires changes to the object properties during the migration process. The migration calculates the changes in such a way that the representation of the objects after migration is the same as prior to migration. Changes made during migration result in different units of measurements and values in the configuration for some object properties.

Migrating the font settings of an object

In WinCC V7 and WinCC flexible, the unit of measurement "point" is used to denote the size of the fonts used for an object. In WinCC, the unit of measurement "pixel" is used to denote the size of the fonts used for an object. During migration, the font size is converted accordingly to ensure that the representation of the font is the same size at zoom level 100%. Due to the different units of measurement, the numerical values for the font sizes change after migration.

Example:

Font style before migration	Font style after migration
Arial 10 points	Arial 13 pixels
Arial 16 points	Arial 21 pixels
Tahoma 10 points	Tahoma 13 pixels
Tahoma 16 points	Tahoma 21 pixels

Migration of object margins

In WinCC flexible, some objects permit the entry of values <0 and >127 for setup of the object margins for the configuration of the representation. In WinCC, the value range for object margins is limited to values between 0 and 127. Migration changes values <0 to "0" and values >127 to "127".

See also

Supported HMI devices (WinCC flexible) (Page 186)

Object support during migration (WinCC flexible) (Page 187)

Migration tags (WinCC flexible)

Introduction

You need to make some special considerations when migrating tags. The following aspects should be distinguished:

- Migrating data types of tags
- Migrating internal tags
- Migrating external tags
- Tag names

Migrating data types

WinCC features some other data types and uses different data type names than WinCC flexible. When migrating a relevant tag, the data type from WinCC flexible is mapped to the corresponding data type in WinCC. You can find additional details on this in the section "Migration of data types (WinCC flexible) (Page 203)".

Migrating tags

Tags are always fully migrated. Only the data type names and tag names may change due to migration.

Migrating names of tags

In WinCC flexible, tags located in different folders can have the same name. In WinCC, the tag name must be unique on the configured HMI device. This means tags with the same name from different folders will be renamed during migration. You can find additional details on this in the section "Basics on migration (WinCC flexible) (Page 181)".

See also

Basics on migration (WinCC flexible) (Page 181)

Migration of data types (WinCC flexible) (Page 203)

Supported HMI devices (WinCC flexible) (Page 186)

Migration of alarm classes and alarm groups (WinCC flexible)

Changing the names of alarm classes

In contrast to WinCC flexible, the names of the predefined alarm classes are not dependent on the user interface language currently in use. During migration, the names of the alarm classes are assigned as follows:

WinCC flexible	WinCC
Error	Alarms
System	System
Warnings	Events

The names of the alarm classes can be changed as necessary after migration.

Migrating alarm groups

Migration will migrate only those alarm groups actually in use.

Alarm groups with an ID from 1-31 will be migrated 1:1.

A corresponding alarm group is created in WinCC for each alarm class in the system. These alarm groups created by the system are assigned IDs beginning with the number 32 and consecutively incremented. The 4 pre-defined alarm classes in every WinCC project are automatically given IDs 32-35 by their alarm groups. Additionally created alarm group and an additional ID is assigned to each user-defined alarm class. Therefore, the IDs for alarms groups with IDs > 31 may be changed after migration. This step also changes the assignment of the alarm group names to the IDs.

Example:

In the example, you can see the assignment of the IDs in WinCC for the migration.

Alarm groups	ID in WinCC flexible	ID in WinCC	
Alarm group 1-16	1-16	1-16	Default for alarm groups from system alarms
Alarm group 17-31	17-31	17-31	Custom alarm groups
		32-35	Default in WinCC for alarm groups of pre-defined alarm classes.
Alarm group 32	32	36	Changed assignment of ID to alarm group in WinCC
Alarm group 33	33	37	Changed assignment of ID to alarm group in WinCC

Also note:

When migrating alarm groups that supposedly have the same group name, the migration adapts the name. This occurs, for example, when a group name contains a space at the end of the name. The migration deletes all existing spaces at the end of names. If two groups obtained the same group names due to this deletion, the migration adds the suffix "# Mign" to the group name of the following alarm groups, where "n" stands for a sequential number.

Example:

The following alarm groups exist in WinCC flexible:

"AlarmGroup_18"

"AlarmGroup_18 " - group name contains one space

"AlarmGroup_18 " - group name contains two spaces

"AlarmGroup_18" is the alarm group with the highest number.

Result after migration:

"AlarmGroup_18"

"AlarmGroup_18#Mig1"

"AlarmGroup_18#Mig1.1"

Changing the names of alarm classes

In contrast to WinCC flexible, the names of the predefined alarm classes are not dependent on the user interface language currently in use. During migration, the names of the alarm classes are assigned as follows:

WinCC flexible	WinCC
Error	Errors
System	System
Warnings	Warnings

The names of the alarm classes can be changed as necessary after migration.

Display of ALARM_S messages and SIMATIC SFM messages

In WinCC flexible you can activate the display classes for ALARM_S messages in integrated projects. In WinCC flexible, you activate the display of SIMATIC SFM messages via a separate setting. The separate setting for activating the display of SIMATIC SFM messages is not required in WinCC. You control the display of SIMATIC SFM messages, and also the display of ALARM_S messages in WinCC only by activating the corresponding display class.

The changed concept may cause the display of messages to change following migration.

If all the display classes for ALARM_S messages are activated and the display of SIMATIC SFM messages is deactivated in the WinCC flexible project , ALARM_S messages and SIMATIC SFM messages are displayed following migration.

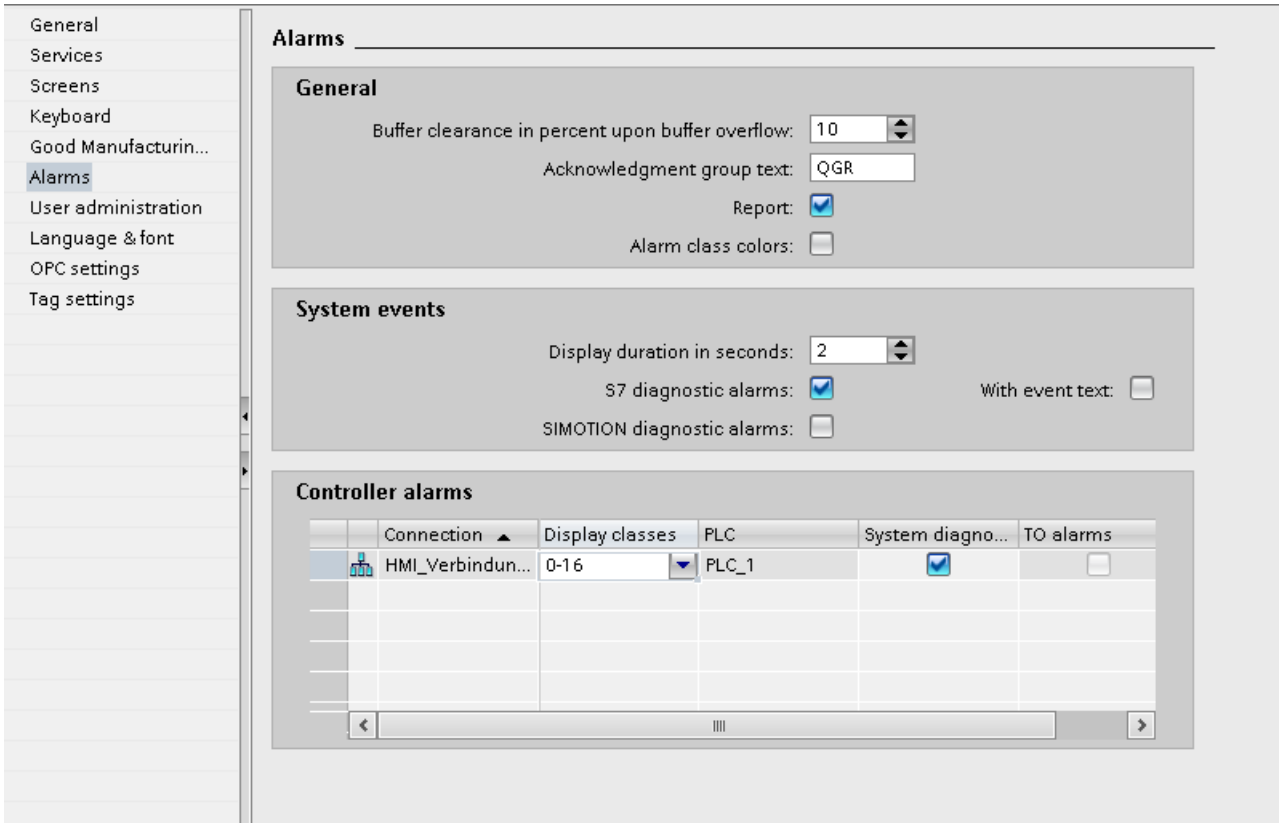
To ensure that only ALARM_S messages are displayed following migration, you have to assign the SIMATIC SFM messages to an unused display class after migration to STEP 7. You then have to deactivate this display class in WinCC.

If all the display classes for ALARM_S messages are deactivated and the display of SIMATIC SFM messages is activated in the WinCC flexible project , ALARM_S messages and SIMATIC SFM messages are not displayed following migration.

To ensure that only SIMATIC SFM messages are displayed following migration, you have to assign the SIMATIC SFM messages to an unused display class after migration to STEP 7. You then have to activate this display class in WinCC.

The display class is dependent on the settings in STEP 7. The default setting for SIMATIC SFM messages in Step 7 is the display class "0". To activate the display in WinCC, the display class "0" must be activated.

You activate the display class in WinCC in the Runtime settings of the respective HMI device in the "Messages" category.



See also

Supported HMI devices (WinCC flexible) (Page 186)

Migration of language-dependent contents (WinCC flexible)

Introduction

WinCC offers the same options for configuring projects in different languages as those available in WinCC flexible. All languages supported by WinCC are included in the migration of a project.

Migrating language-dependent content

The following language-dependent content is migrated:

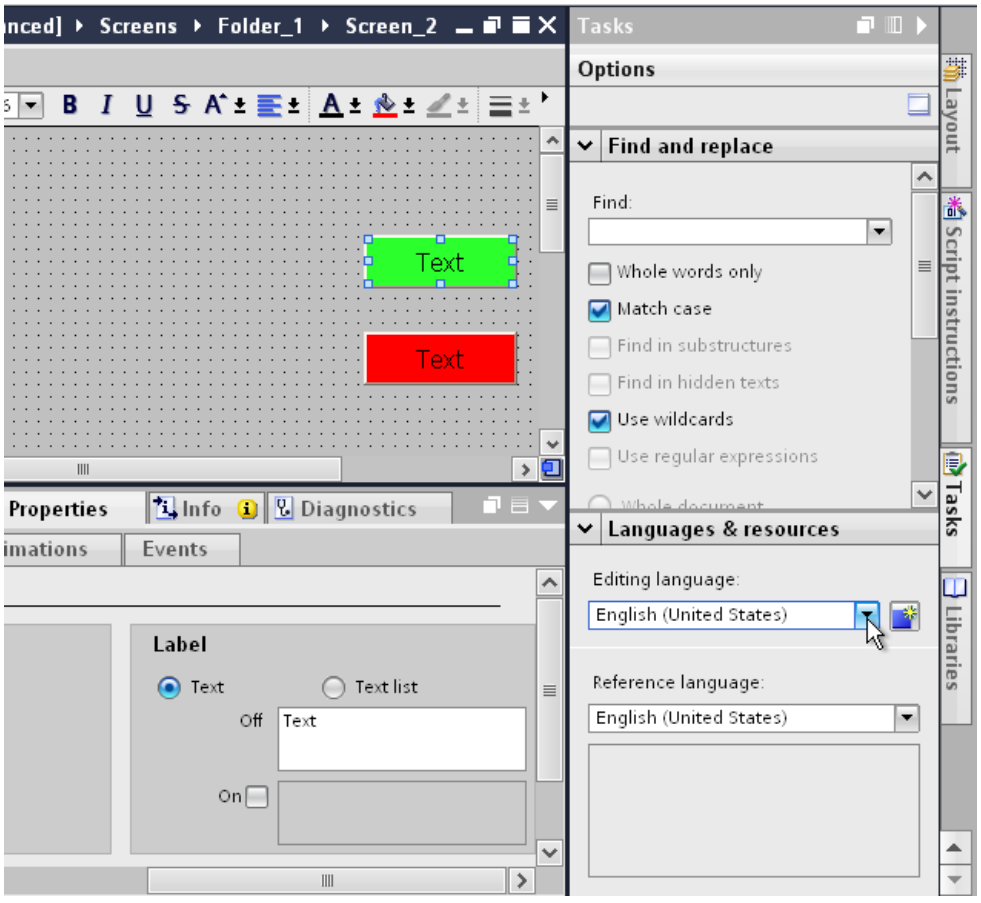
- Project languages
- Project texts
- Fonts for display in runtime
- Language-dependent graphics

You need to consider the following when migrating language-dependent content:

- The operating system on the PC performing the migration must support the project languages used in the project.
- The fonts used for runtime display must be installed on the PC performing the migration.
- Dictionaries are not supported by the migration.

Editiing language of integrated projects following migration

During migration of an integrated project, the project components to be migrated from STEP 7 and WinCC flexible also bring their respective settings for the editing language. In WinCC there is only one editing language for all project components. Migration activates for the mgrated project the editing language which was set in STEP 7 prior to migration. If this setting is not the same as the setting from WinCC flexible, the configured texts are no longer visible in WinCC. No text is displayed at the usage locations, or only the entry "Text" can be seen. To make the texts visible, you must change the editing language. Click the "Tasks" taskcard at the right-hand edge of the TIA portal and select the correct editing language in the "Language & Resources" area.



Unsupported languages

The migration of language-dependent content depends on whether or not WinCC supports the respective language.

If a project only contains project languages not supported by WinCC, the project will not be migrated.

If a project contains supported and unsupported project languages, only the supported languages will be migrated. The editing language and reference language are set to a supported language.

The following languages are not supported by WinCC:

- Arabic
- Hebrew
- Dhivehi
- Gujarati
- Kannada
- Tamil
- Telugu
- Urdu
- Punjabi
- Persian
- Syrian

See also

Supported HMI devices (WinCC flexible) (Page 186)

Migrating libraries (WinCC flexible)

Introduction

You need to consider two different cases when migrating from libraries:

1. Migrating a project library
2. Migrating a global library

Migrating a project library

A project library is stored together with the project data in the project file. For this reason, a project library is migrated with the same restrictions as the project data.

Migrating a global library

Global libraries are not supported by the migration. The library objects used in the project will be migrated, however. The library objects are copied when used in the project and then no longer have a connection to the library.

To migrate a global library, you must copy or move the objects contained in the library to the project library. The objects are then included in the migration. In WinCC, you move the migrated objects to a new global library that is created. You can copy or move both individual objects or entire library categories.

See also

Supported HMI devices (WinCC flexible) (Page 186)

5.2.7.3 Migrating runtime data (WinCC flexible)

Migrating runtime data (WinCC flexible)

Introduction

When migrating a project, only the configuration data will be migrated. The runtime data are not affected. You need to update the runtime data following migration.

The runtime data consists of the following:

- Runtime project
  The runtime project contains the compiled project data.
- Recipe data and user administration
  The recipe data and user administration are data that can be changed in runtime.

Migrating runtime data

You update the runtime project by compiling the project in WinCC again and loading it to the HMI device.

If the recipe data and user administration were changed in runtime, you need to back up this information from the HMI device before you load the migrated project. You can then load the migrated project to the HMI device. Finally, you load the saved recipe data and user administration back to the HMI device.

See also

Migration (WinCC flexible) (Page 181)

Backing up recipe data and user administration (WinCC flexible) (Page 197)

Restoring recipe data and user administration (WinCC flexible) (Page 198)

Backing up recipe data and user administration (WinCC flexible)

Introduction

To continue using the recipe data and user administration in a migrated project, you first need to back up this data from the HMI device. Then load the data into the migrated WinCC project. Use ProSave to back up the data.

Requirement

- The WinCC flexible project is running on the HMI device in Runtime.
- The HMI device is connected to a PC on which ProSave is installed.

Procedure

Proceed as follows to back up the recipe data and user administration:

1. Start ProSave.
2. Select the device type and the connection parameters in the "General" tab.
3. Open the "Backup" tab.
4. Select the "Recipes from the device memory" entry in the "Data type" box.
   Do not select "Complete backup" because otherwise you will not be able to select separately when restoring the recipe data.
5. Navigate to the desired location in the "Save as" box and click "Start Backup".
   The recipe data are saved.
6. Select "User administration" in the "Data type" box and click "Start Backup".
   The user administration is saved.

For additional information refer to the online help for ProSave.

Alternative procedure

ProSave is automatically installed with WinCC flexible. The entire functional range of ProSave is available on the configuration PC within WinCC flexible via the menu command "Project > Transfer".

Alternatively, you can back up the recipe data and user administration via the ProSave integrated in WinCC flexible. Start WinCC flexible and select the menu command "Project > Transfer > Backup". Back up the recipe data and user administration as described in steps 4-6.

See also

Migrating runtime data (WinCC flexible) (Page 197)

Restoring recipe data and user administration (WinCC flexible) (Page 198)

Restoring recipe data and user administration (WinCC flexible)

Introduction

To continue using saved recipe data and user administration after the migration, you first need to compile the migrated project and load it to the HMI device. You can then transfer the saved data to the HMI device. Use ProSave to restore the data.

Requirement

- The migrated project has been transferred to the HMI device and is running in runtime.
- The HMI device is connected to a PC on which ProSave is installed.

Procedure

Proceed as follows to load the saved recipe data and user administration to the HMI device:

1. Start ProSave.
2. Select the device type and the connection parameters in the "General" tab.
3. Open the "Restore" tab.
4. Navigate to the location of the saved recipe data in the "Opening..." box and select the file.
5. Click "Start Restore".
   The recipe data will be transferred to the HMI device..
6. Repeat steps 4-5 to restore the user administration.
   The user administration will be transferred to the HMI device.

For additional information refer to the online help for ProSave.

Alternative procedure

ProSave is automatically installed with WinCC. The entire functional range of ProSave is available on the configuration PC within WinCC flexible via the menu command "Project > Transfer".

You can also restore the recipe data and user administration via the ProSave integrated in WinCC. Start WinCC and select the menu command "Online > Device maintenance > Restore". Restore the recipe data and user administration as described in steps 4-6.

See also

Migrating runtime data (WinCC flexible) (Page 197)

Backing up recipe data and user administration (WinCC flexible) (Page 197)

5.2.7.4 Migrating integrated projects (WinCC flexible)

Migrating integrated projects (WinCC flexible)

Introduction

The controllers and HMI devices contained in a project integrated in STEP 7 are linked together by the configuration. The configuration data of WinCC flexible and STEP 7 are also connected. When an integrated project is migrated, the complete project will be migrated with components from WinCC flexible and STEP 7. The connections remain intact.

Note

It is advisable to compile and save an integrated project in WinCC flexible before you migrate it. You can be sure that the data in WinCC flexible and STEP 7 is synchronized if compilation was completed without errors.

Migrating an integrated project

When migrating an integrated project, the same requirements apply for the WinCC flexible component as those for the migration of a non-integrated WinCC flexible project. The objects and properties contained in the WinCC flexible component must be supported by WinCC, for example, the HMI device or the communication driver. The "Online" property must be activated on the configured connection. A connection with deactivated "Online" property is not migrated.

In addition to the requirements for the WinCC flexible component, there are also requirements for the STEP 7 component of the integrated project. The objects and properties contained in the STEP 7 V5.4 SP5 or V5.5 component must be supported in STEP 7. For detailed information, refer to the documentation for STEP 7.

To fully migrate an integrated project and then edit it, the following components must be installed on the PC performing the migration:

- STEP 7 V5.4 SP5 or STEP 7 V5.5
- WinCC flexible 2008 SP2 or WinCC flexible 2008 SP3
- STEP 7

If you only want to save the project in migration format, you can use the migration tool. See Basics on migration (WinCC flexible) (Page 181) for additional information.

An integrated project is always fully migrated. If you only want to migrate the WinCC flexible project it contains, you need to separate it from the STEP 7 project before the migration. To separate the project from the integrated form, open the project in STEP 7 V5.4 SP5 or V5.5. Open the WinCC flexible project in the SIMATIC Manager. The project is opened with WinCC flexible. In WinCC flexible, select the menu command "Project > Copy project from STEP 7". WinCC flexible saves a non-integrated copy of the project.

See also

Basics on migration (WinCC flexible) (Page 181)

Migrating an integrated project

Introduction

When migrating an integrated project, the components from both the WinCC flexible project and the STEP 7 project will be migrated. This means you need to select the project file with the file extension "*.s7p" for migration. During migration, the data is copied from the existing project and migrated to a new project. You cannot migrate to an existing project.

The migration can be started in both the Portal view and the Project view.

You should only migrate a project to a re-started TIA Portal.

If you only want to save the project in migration format, you can use the migration tool. For more information, refer to basics of migration (WinCC flexible).

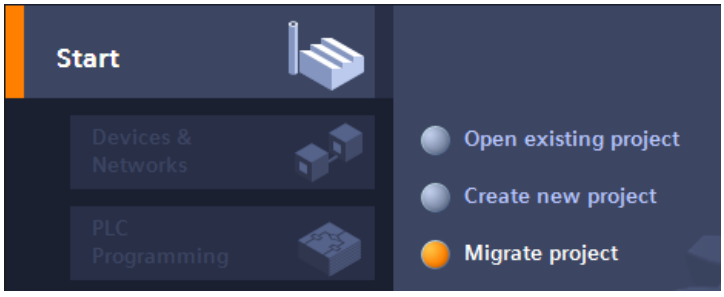
Requirement

- STEP 7 V5.4 SP5 or STEP 7 V5.5 and all option packages used are installed.
- STEP 7 and all option packages used are installed.
- The TIA Portal is restarted.
- No project is open in WinCC.
- An integrated project is available.
- The integrated project is not open.

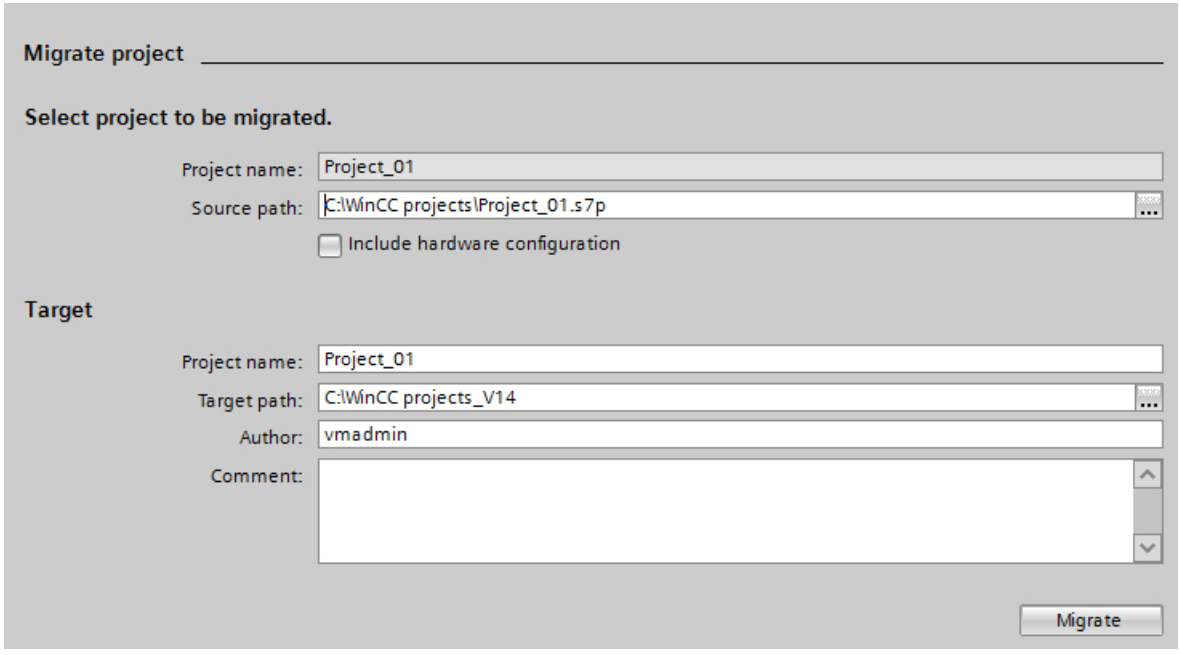
Procedure

To migrate an integrated project in the portal view, follow these steps:

1. Select the action "Start > Migrate Project".



2. In the "Source path" box, navigate to the project you want to migrate.



3. Select the "*.s7p" project file.
4. Change the information for the project to be created, if necessary. For example, change the project name or project path. The data to be migrated is created in the new project.
5. To migrate the project with the hardware configuration, enable "Include hardware configuration".

6. Click "Migrate".
   A new project is created and migration of the data is started:
   - The Project view opens.
   - The progress of the migration is shown in a migration window.
   - Warnings and errors about the migration process are displayed in the Inspector window under "Info > General".
   - All information about the migration is saved in a log file.
   - A message is displayed upon completion of the migration. The message contains a link that you can use to open the log file.
7. Once migration is completed, save the project.

Once the migration is complete, you will find a newly created device for each migrated HMI device and controller in the project tree. These devices include the migrated data.

Note

Integrated projects with alarm views

The alarm classes in the alarm view can be disabled during migration of the project. Once migration of the project is complete, check the settings in the alarm view. Enable the required alarm classes in the Inspector window of the alarm view if needed under "Properties > General".

Opening the migration log at a later point in time

The migration log is saved together with the migrated project. You can view the log at a later point in time. Open the log file as follows:

1. Open "Common data > Logs" in the project navigation.
2. Double-click the log file. The migration log opens.

See also

Basics on migration (WinCC flexible) (Page 181)

5.2.7.5 Reference (WinCC flexible)

Migration of data types (WinCC flexible)

Introduction

To harmonize the data types used by controllers and HMI systems, some types of internal HMI tags are renamed. The naming takes place in accordance with IEC conventions. Because only the names change, there are no changes to the internal tags for the configuration.

The following table describes the mapping of data types from WinCC flexible to the data types in WinCC.

Migration of data types

The internal data types are mapped as follows during migration:

Internal data types WinCC flexible	Internal data types WinCC
Bool	Bool
Char	SInt
Byte	USInt
Int	Int
UInt	UInt
Long	DInt
ULong	UDInt
Float	Real
Double	LReal
String	WString
DateTime	DateTime

Migrating external data types

See the following pages for how to map the available communication drivers.

See also

Migration (WinCC flexible) (Page 181)
Migrating data types of Allen-Bradley DF1 (WinCC flexible) (Page 204)
Migrating data types of Allen-Bradley Ethernet IP (WinCC flexible) (Page 205)
Migrating data types of Mitsubishi FX (WinCC flexible) (Page 206)
Migrating data types of Modicon Modbus (WinCC flexible) (Page 206)
Migrating data types of Modicon Modbus TCP/IP (WinCC flexible) (Page 207)
Migrating data types of Omron Hostlink/Multilink (WinCC flexible) (Page 207)
Migrating data types of SIMATIC S7 200 (WinCC flexible) (Page 208)
Migrating data types of SIMATIC S7 300/400 (WinCC flexible) (Page 208)

Migrating data types of Allen-Bradley DF1 (WinCC flexible)

Migrating data types Allen-Bradley DF1

The data types of the Allen-Bradley DF1 communication driver are mapped as follows in the migration to WinCC:

Data type in WinCC flexible	Data type in WinCC
ASCII	ASCII
BCD4	UInt

Data type in WinCC flexible	Data type in WinCC
BCD8	UDInt
Bit	Bool
Int	Int
Long	DInt
Real	Real
UInt	UInt
ULong	UDInt

See also

Migration of data types (WinCC flexible) (Page 203)

Migrating data types of Allen-Bradley Ethernet IP (WinCC flexible)

Migrating data types Allen-Bradley Ethernet IP

The data types of the Allen-Bradley Ethernet IP communication driver are mapped as follows in the migration to WinCC:

Data type in WinCC flexible	Data type in WinCC
Bool	Bool
DInt	DInt
Int	Int
Real	Real
SInt	SInt
String	String
UDInt	UDInt
UInt	UInt
USInt	USInt

See also

Migration of data types (WinCC flexible) (Page 203)

Migrating data types of Mitsubishi FX (WinCC flexible)

Migrating data types Mitsubishi FX

The data types of the Mitsubishi FX communication driver are mapped as follows in the migration to WinCC:

Data type in WinCC flexible	Data type in WinCC
12 Bit Block	12-Bit Block
16 Bit Block	16-Bit Block
20 Bit Block	20-Bit Block
24 Bit Block	24-Bit Block
28 Bit Block	28-Bit Block
32 Bit Block	32-Bit Block
4 Bit Block	4-Bit Block
8 Bit Block	8-Bit Block
Bit	Bool
Double	DWord
IEEE-Float	Real
String	String
Word	Word

See also

Migration of data types (WinCC flexible) (Page 203)

Migrating data types of Modicon Modbus (WinCC flexible)

Migrating data types Modicon Modbus

The Modicon Modbus communication driver is not supported by WinCC, it is replaced by the Modicon Modbus RTU driver. The data types of the Modicon Modbus communication driver are mapped as follows in the migration to WinCC:

Data type in WinCC flexible	Data type in WinCC
+/-Double	+/- Double
+/-Int	+/- Int
16 Bit Group	16 Bit Group
ASCII	ASCII
Bit	Bit
Double	Double
Float	Float
Int	Int

See also

Migration of data types (WinCC flexible) (Page 203)

Migrating data types of Modicon Modbus TCP/IP (WinCC flexible)

Migrating data types Modicon Modbus TCP/IP

The data types of the Modicon Modbus TCP/IP communication driver are mapped as follows in the migration to WinCC:

Data type in WinCC flexible	Data type in WinCC
+/-Double	+/- Double
+/-Int	+/- Int
16 Bit Group	16 Bit Group
ASCII	ASCII
Bit	Bit
Double	Double
Float	Float
Int	Int

See also

Migration of data types (WinCC flexible) (Page 203)

Migrating data types of Omron Hostlink/Multilink (WinCC flexible)

Migrating data types Omron Hostlink/Multilink

The Omron Hostlink/Multilink communication driver is not supported by WinCC, it is replaced by the Omron Host Link driver. The data types of the Omron Hostlink/Multilink communication driver are mapped as follows in the migration to WinCC:

Data type in WinCC flexible	Data type in WinCC
+/-DEC	Int
+/-LDEC	DInt
ASCII	String
BIN	Bool
BYTE	Byte
DEC	UInt
IEEE	Real
LDEC	UDInt

See also

Migration of data types (WinCC flexible) (Page 203)

Migrating data types of SIMATIC S7 200 (WinCC flexible)

Migrating data types SIMATIC S7 200

The data types of the SIMATIC S7 200 communication driver are mapped as follows in the migration to WinCC:

Data type in WinCC flexible	Data type in WinCC
Bool	Bool
Byte	Byte
Char	Char
DInt	DInt
DWord	DWord
Int	Int
Real	Real
StringChar	StringChar
Timer	Timer
Word	Word

See also

Migration of data types (WinCC flexible) (Page 203)

Migrating data types of SIMATIC S7 300/400 (WinCC flexible)

Migrating data types SIMATIC S7 300/400

The data types of the SIMATIC S7 300/400 communication driver are mapped as follows in the migration to WinCC:

Data type in WinCC flexible	Data type in WinCC
Bool	Bool
Byte	Byte
Char	see below
Counter	see below
Date	Date
Date and Time	Date_And_Time
DInt	DInt
DWord	DWord
Int	Int
Real	Real

Data type in WinCC flexible	Data type in WinCC
String	String
StringChar	see below
Time	Time
Time of Day	Time_Of_Day
Timer	see below
Word	Word

Special considerations for some data types

There are special considerations to be made when migrating external tags that contain data types of a SIMATIC S7-300/400 PLC.

Mapping of the S7 data type "Char"

The S7 data type "Char" is a data type for mapping characters according to the specification. However, since this data type is often used for reading and writing numerical values, it is mapped in WinCC to the S7 data type "Byte". If this should be the case during migration, an alarm will appear in the output window.

If the S7 data type "Char" is used for numerical values and negative numbers were configured at the point of use, the result is an error in mapping to the S7 data type "Byte". The S7 data type "Byte" cannot map any negative numbers. You have to adapt the configuration accordingly to correct the error. Use a signed data type, such as the data type "Int", for processing positive and negative numerical values.

If the S7 data type "Char" is used for mapping characters, you must change the configuration after migration. To represent characters, use the data type "String".

When an integrated project is migrated, the data type "Char" in WinCC is also migrated to the data type "Byte". With a connected PLC tag, the data type "Char" remains "Char". As a result of changing the data type of the HMI tag, symbolic addressing of the tags in question is not migrated. After migration. the tags are interconnected by absolute addresses and continue to work. If you want to restore symbolic addressing, you have to change the configuration accordingly after the migration.

Mapping an array of the S7 data type "Char"

An array of the S7 data type "Char" is mapped to an array of the data type "Byte" during migration.

If an array of the S7 data type "Char" is used for numerical values and negative numbers were configured at the point of use, the result is an error in mapping to an array of the S7 data type "Byte". The S7 data type "Byte" cannot map any negative numbers. You have to adapt the configuration accordingly to correct the error. Use a signed data type, such as the data type "Int", for processing positive and negative numerical values.

Mapping of the S7 data type "Counter"

An external tag with the S7 data type "Counter" with counter address is mapped to the S7 data type "Counter". The address will be retained.

If an external tag with the S7 data type "Counter" addresses a data block or a bit memory address, it is is mapped to the S7 data type "Word". The address will be retained. The migration sets the coding to "SimaticBCDCounter".

The S7 data type "Counter" has a value range of 0-999. When supplied by the S7 data type "Word" the value range may be exceeded on the PLC side. Ensure that you are observing the value range.

Example:

WinCC flexible

Tag	S7 data type	Address	Comment
Counter_Actual_Value	Counter	C10	BCD coded counter value
Counter_Setpoint_Value	Counter	DB10.DBW200	BCD coded counter value
Counter_Setpoint_Value#2	Counter	MW20	BCD coded counter value

WinCC

Tag	S7 data type	Address	Coding	Comment
Counter_Actual_Value	Counter	%C10	<Standard>	BCD coded counter value
Counter_Setpoint_Value	Word	%DB10.%DBW200	SimaticBCDCounter	BCD coded counter value
Counter_Setpoint_Value#2	Word	%MW20	SimaticBCDCounter	BCD coded counter value

Mapping of the data type "StringChar"

In WinCC there is no corresponding data type to which the "StringChar" data type can be mapped. Mapping in WinCC depends on the property "Length" of the S7 data type.

A tag of the "StringChar" data type with the "Length" property > 1 is migrated to an array of the S7 data type "Char". The length of the array corresponds to the length of the originally configured data type "StringChar".

If the property "Length" = 1, the data type in WinCC is migrated to an array of the S7 data type "Char" with length = 1. The expression for an array with an element is "Array[0 ..0] of Char".

Mapping of the S7 data type "Timer"

An external tag with the S7 data type "Timer" with timer address is mapped to the S7 data type "Timer". The address will be retained.

If an external tag with the S7 data type "Timer" addresses a data block or a bit memory address, it is is mapped to the S7 data type "S5 Time". The address will be retained.

Example:

WinCC flexible

Tag	S7 data type	Address	Comment
Timer_Actual_Value	Timer	T10	BCD coded timer value
Timer_Setpoint_Value	Timer	DB10.DBW200	BCD coded timer value
Timer_Setpoint_Value#2	Timer	MW20	BCD coded timer value

WinCC

Tag	S7 data type	Address	Comment
Timer_Actual_Value	Timer	%T10	BCD coded timer value
Timer_Setpoint_Value	S5Time	%DB10.%DBW200	BCD coded timer value
Timer_Setpoint_Value#2	S5Time	%MW20	BCD coded timer value

See also

Migration of data types (WinCC flexible) (Page 203)

5.2.8 Migrating integrated projects

5.2.8.1 Migrating an integrated project (S7-300, S7-400)

Introduction

In integrated projects, you use SIMATIC controllers and WinCC components together in a project. When an integrated project is migrated, the complete project will be migrated with components from WinCC and STEP 7. Configured connections between control and visualization remain intact.

Migrating an integrated project

When migrating an integrated project, the same requirements apply for the STEP 7 component as those for migration of a non-integrated STEP 7 project. The objects and properties contained in the WinCC component must also be supported in WinCC (TIA Portal).

For an operating station (OS) to be migrated, it has to be located below a PC station and the WinCC application in the project tree of SIMATIC Manager. The following figure shows the assignment of the operating station within the initial project:



Additional migration requirements for integrated projects can be found in the documentation for WinCC.

Also note that the initial project must be compiled before the migration.

In order to fully migrate an integrated project, the following components must be installed on the programming device/PC for the migration:

- STEP 7 V5.4 SP5
- WinCC V7.3 with the latest update or WinCC Flexible 2008 SP2 or SP3

To be able to fully post-edit an integrated project, the latest version of the following components must be installed on the PC for post-editing:

- STEP 7 Professional
- WinCC Basic, WinCC Comfort/Advanced or WinCC Professional, depending on the components used

Using the migration tool

It is necessary to use the migration tool under the following circumstances:

- The initial project is not located on the same programming device/PC as the installation of the TIA Portal.
- SCADA devices are included in the initial project. These can only be migrated with the migration tool.
- WinCC Professional V14 and STEP 7 with WinCC V7.3 cannot be installed on the same programming device/PC. Therefore, integrated projects with WinCC V7.3 parts must be prepared for migration using the migration tool.

Migration of the STEP 7 part of an integrated project

An integrated project is always fully migrated. Individual components cannot be migrated on their own. You can only migrate the included STEP 7 project alone, if you have previously deleted all HMI stations in the SIMATIC stations in the SIMATIC Manager and then recompiled the project in NetPro.

Alternatively, you can open the project in an installation of STEP 7 V5.4 SP5 without an installation of WinCC. Then, save the project again and select the "Reorganize" function during saving. The WinCC parts are then automatically removed when the copy is saved.

You can then migrate the STEP 7 project without the WinCC project.

Migration of an integrated project with the hardware configuration

In integrated projects, HMI devices are migrated even if the hardware configuration is not included in the migration. The STEP 7 part of the hardware configuration, including network configurations, and connections and interrupts, is migrated only if you include the hardware configuration in the migration. Otherwise, unspecified modules will be created for the STEP 7 devices and you will need to convert them into suitable modules after the migration.

HMI modules that are plugged into a PC station are converted to a separate Station during the migration. If you perform the migration without including the hardware configuration, the

migrated project then contains a non-specified SIMATIC PC Station and a SIMATIC PC Station with the HMI devices. References to HMI devices are not imported during migration. When the hardware configuration is included, the migrated project contains two separate stations: the HMI Station and the PC Station.

Storage location of an integrated WinCC project

If you migrate an integrated project, the HMI part of it must be on the same PG/PC as the STEP 7 part of the project. If the HMI part is on a different PG, then only the STEP 7 part will be migrated.

Unsupported objects

The following components are not supported for migration:

- STEP 7 multiproject
  A STEP 7 multiproject cannot be migrated. Migration will be canceled.
- Central Archive Server - CAS
  If a CAS is part of an integrated project, then the migration will be carried out but the CAS data will not be migrated.

See also

Post-editing integrated projects (Page 213)

5.2.8.2 Post-editing integrated projects

If you have migrated an integrated project without hardware configuration, unspecified CPUs are used instead of the CPUs of the original project. Since no connection can exist between an unspecified CPU and an HMI device, connections from the source project are also imported only unspecified.

Procedure

To continue to use an integrated project after the migration, follow these steps:

1. Convert the unspecified devices into suitable devices again.
2. Restore the integrated HMI connection between the HMI device and the PLC.
3. Connect all HMI tags to the newly created integrated connection.
4. Restore the connection between HMI tags and PLC tags.
5. Delete the non-integrated HMI connection.

In the following chapters a sample project is used to describe the individual steps in more detail.

See also

Converting unspecified CPUs into specified CPUs (Page 214)

Creating an integrated HMI connection (Page 216)

Re-linking HMI tags (Page 218)

Delete non-integrated HMI connection (Page 219)

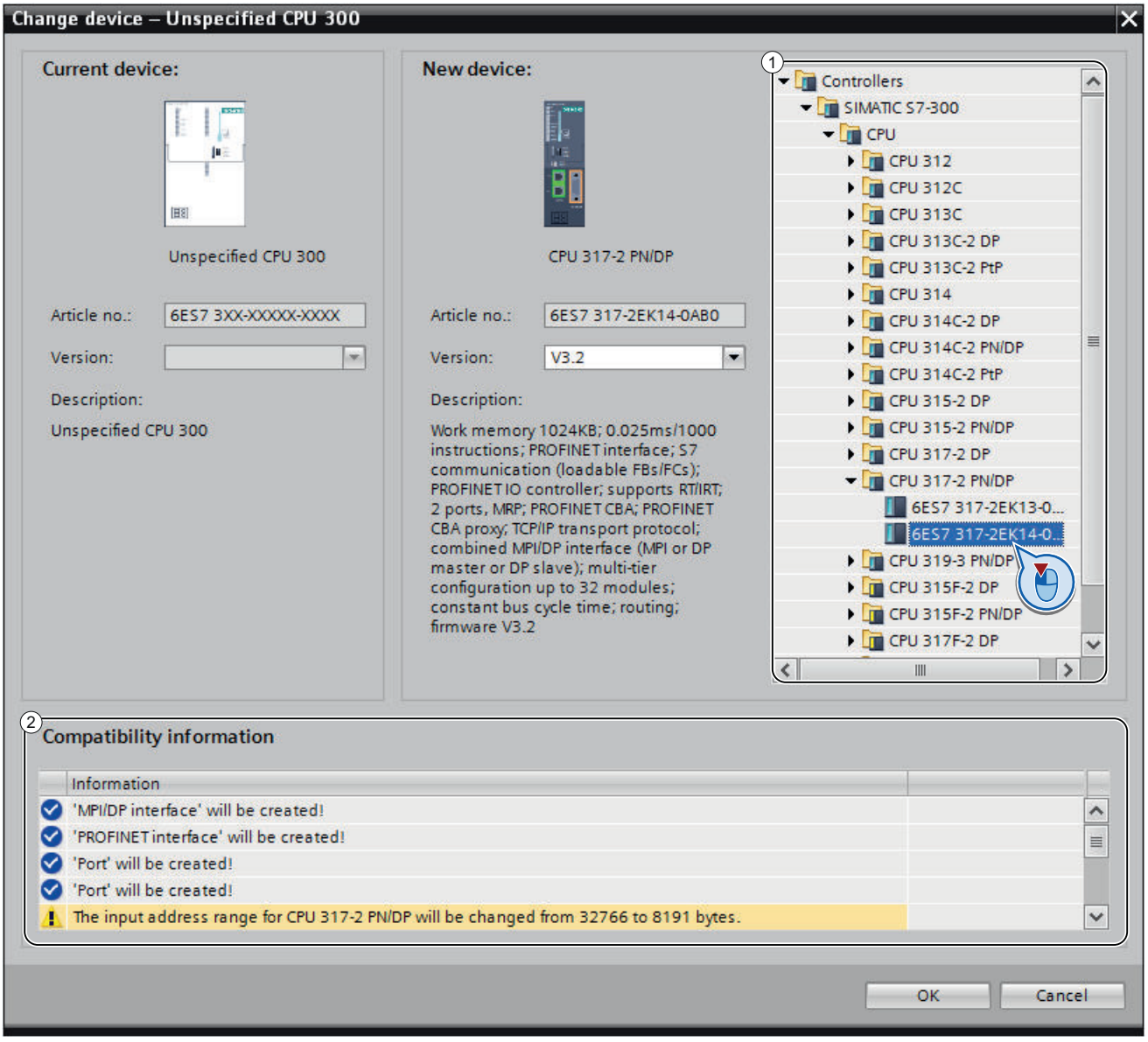
5.2.8.3 Converting unspecified CPUs into specified CPUs

The first step after the migration without hardware configuration is the conversion of the unspecified CPUs into specified CPUs. Unspecified CPUs are placeholders for certain CPUs from the hardware catalog that are not currently known. You can define general parameters and home the CPUs already in the user program. However, the project is not fully functional until the unspecified CPU has been specified.

Specifying a CPU using module replacement

To use module replacement to specify an unspecified CPU, follow these steps:

1. Select the unspecified CPU in the network or device view.
2. Select the "Replace device" command in the shortcut menu.
   The "Replace device" dialog opens.



3. Under "New device" in the tree structure, select the module with which you want to replace the unspecified CPU. (Area 1)
   "Compatibility information" provides you with information on the extent to which the selected CPU is compatible with the configuration in source project. (Area 2)
4. Click "OK".
5. Perform the above-described steps for all unspecified CPUs.

See also

Creating an integrated HMI connection (Page 216)

5.2.8.4 Creating an integrated HMI connection

After you have specified the unspecified CPU, establish the connection to the HMI-device.

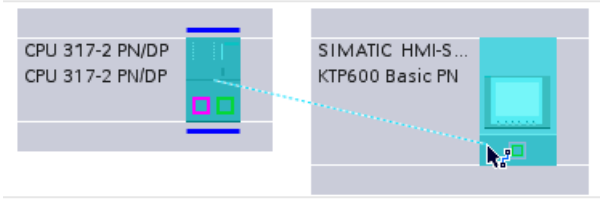
Procedure

To create a connection graphically, follow these steps:

1. On the toolbar, click the "Connections" icon. This activates connection mode.

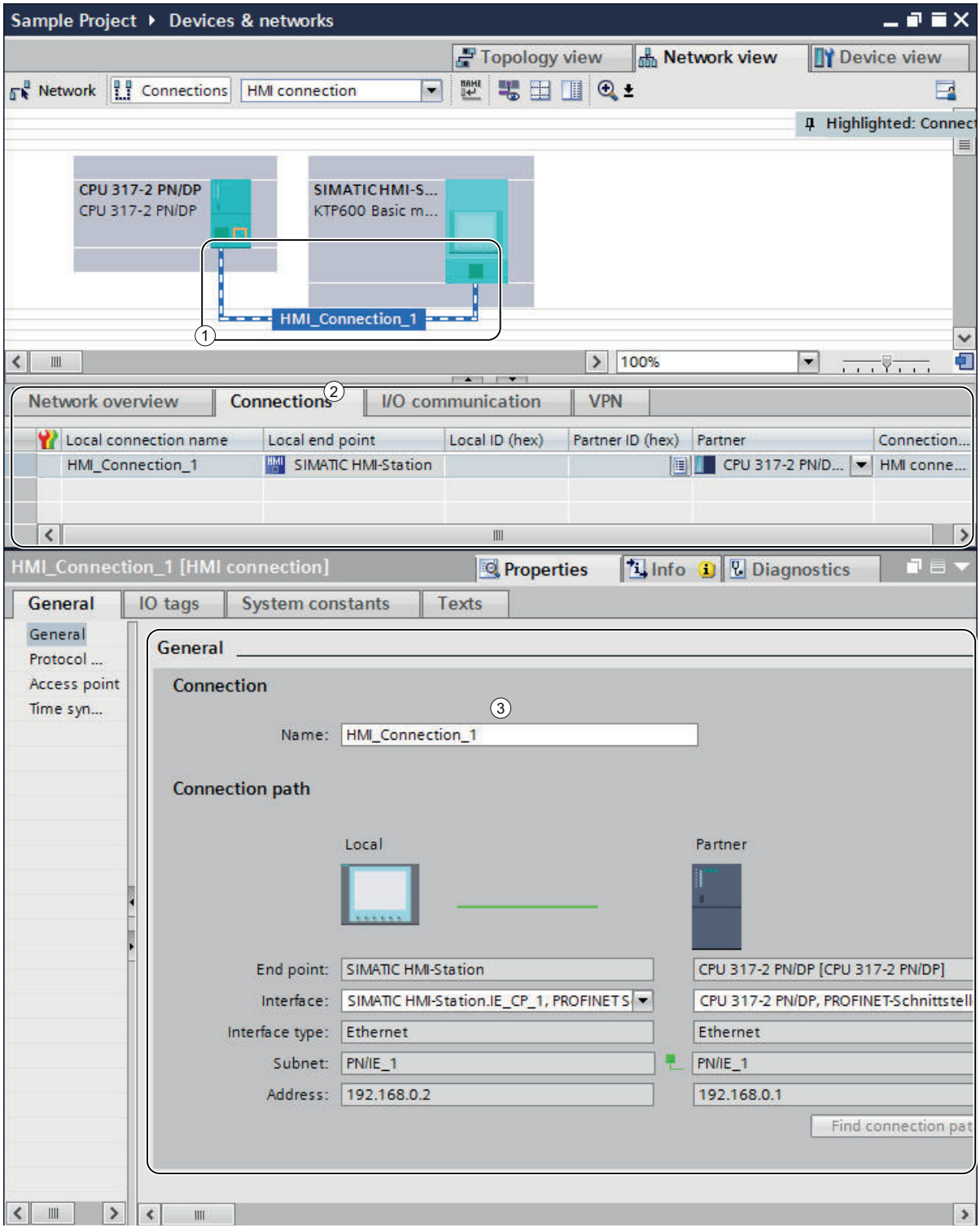


2. Select the connection type "HMI connection" in the adjacent drop-down list.
   The network view highlights in color all CPUs and HMI devices that can be used for an HMI connection.
3. You can now have the connection path automatically determined, or explicitly select a connection path via specific interfaces:
   - Allow connection path to be automatically determined
     Select the source CPU for a connection. Drag the mouse to the target components. Confirm the connection endpoint with another mouse click.
     Alternatively: While holding down the shift button, select the target components and with the right mouse button select the "Add new connection" command.
   - Selecting an explicit connection path from interface to interface
     Click on the subnet interface in the device for which you want to create a connection. Hold down the mouse button, drag the cursor to the relevant interface in the target device and then release the mouse button.



Result

The following figure shows the state after the integrated connection has been created:



① An integrated HMI connection is created and highlighted in the network view.

② The connection is shown in the connection table of the components.
③ The connection can be edited in the connection properties.

See also

Re-linking HMI tags (Page 218)

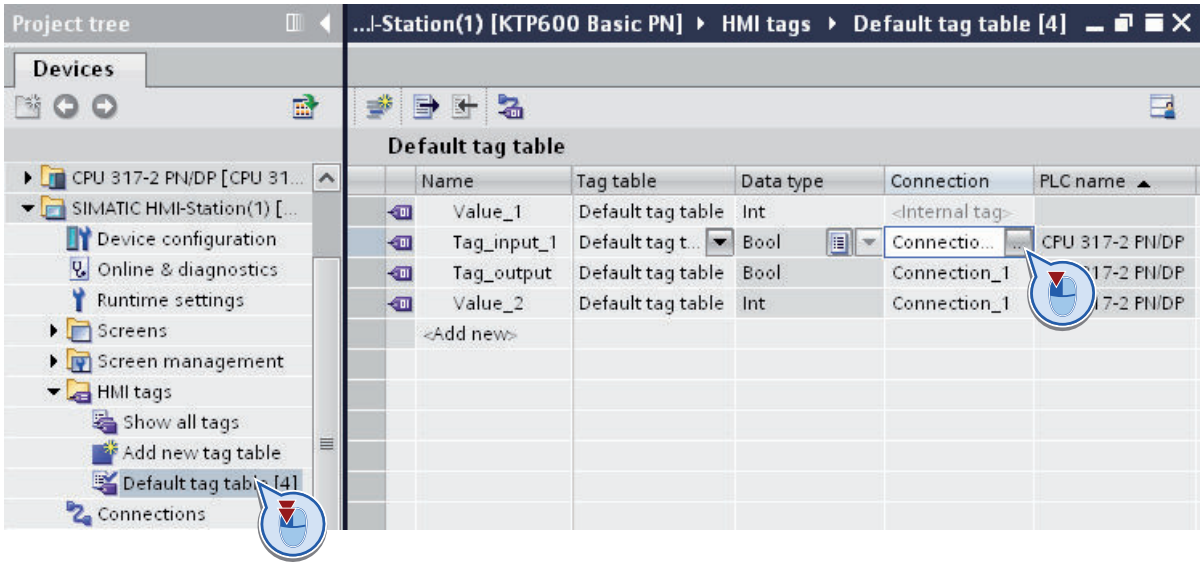
5.2.8.5 Re-linking HMI tags

When you have created a new HMI connection between the CPU and HMI device, you have to assign the existing HMI tags to the new connection. Perform the following steps for each line in the relevant tag table.

Procedure

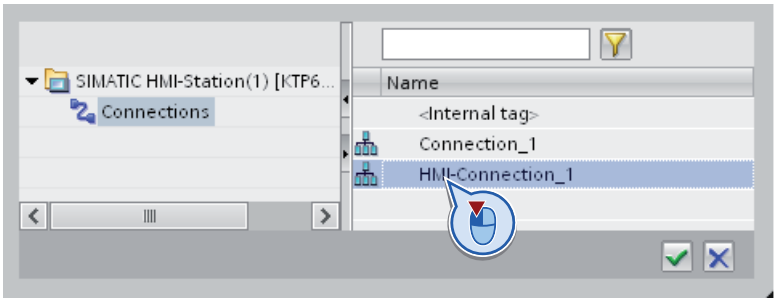
To re-link HMI tags, follow these steps:

1. In the project tree, navigate to the HMI tags and double-click the relevant tag table to show this in the work area.
   The tag table opens.

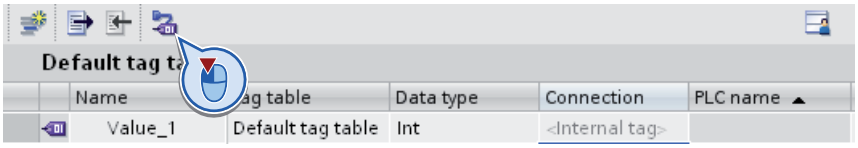


2. Click the " ... " button in the "Connection" column.
   A dialog box for selecting the connection opens.

3. Select the newly established HMI connection.



4. Click the "✓" button to apply the selected connection.
5. On the toolbar, click the "Re-connect PLC tag" button.



See also

Delete non-integrated HMI connection (Page 219)

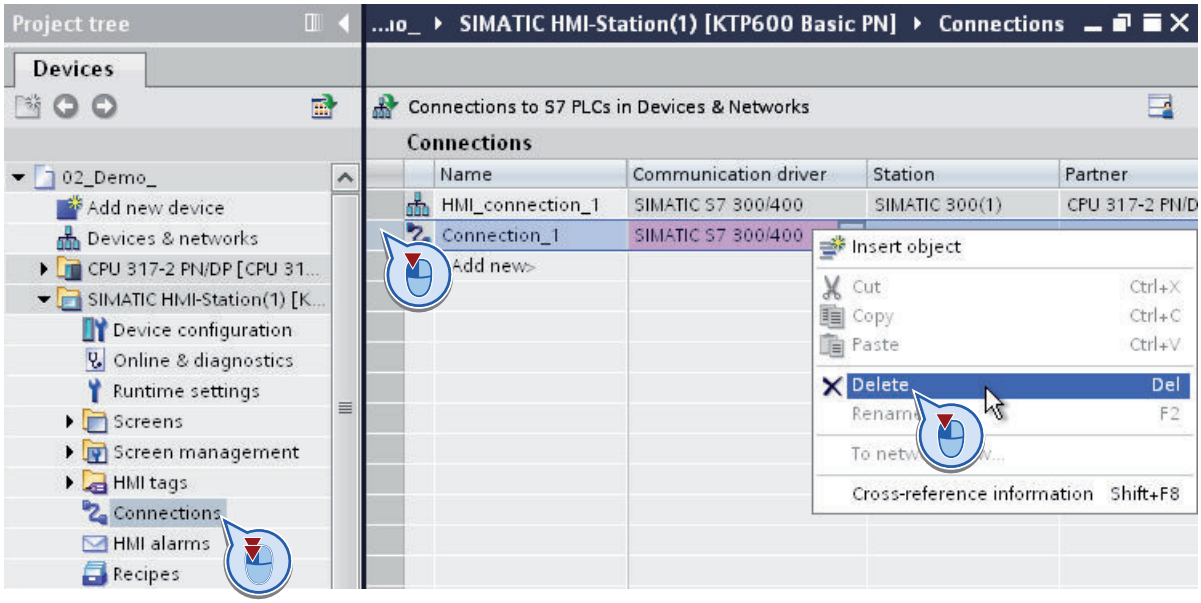
5.2.8.6 Delete non-integrated HMI connection

Finally, you can remove non-integrated HMI connections that still remain from the source project.

Procedure

To delete the non-integrated HMI connections, follow these steps:

1. In the project tree, open the HMI device and double-click the "Connections" entry. The connection table opens.



2. Select the row with the old connection in the table.
3. Select the "Delete" command in the shortcut menu of the connection line.
4. Perform the above-described steps for all non-integrated HMI connections of the source project.

5.3 Migrating S7-1200 to firmware as of V4 (S7-1200)

5.3.1 Basic information on upgrading to V4 (S7-1200)

Introduction

If you have used a CPU with firmware version V3 in your project and want to upgrade to a CPU with firmware V4.0 or later, you can easily replace the device.

The TIA Portal offers the "Change device" function for this purpose. The project remains unchanged when the device is replaced. You can continue using the programs that you created with firmware version V3.

Rules

The following basic rules apply when replacing a device:

- Replacing a device is only possible if the project was created based on a CPU with firmware version V3.0. If your project was created based on firmware version V1.0 or V2.0, create a new CPU with firmware version V3.0 offline in the project, and copy your program to this CPU.
- It is not possible to replace a V4 CPU with a V3 CPU. If you want to continue using the existing V3 CPU, create a copy of this CPU before replacing the device.
- The program cannot be transferred to the new CPU via a memory card. Instead, use the "Change device" function, which is described in the following sections.

HMI panels

Configured HMI panels are treated differently during device replacement depending on the firmware version of the panel and the communication mode.

The following table shows the HMI connections that are supported with the migration:

Firmware version of the panel	PUT/GET communication	Migration to V4
V11 or later	No	S7-1200 does not support this configuration. Upgrade the firmware of the HMI panel to V12.0. Then compile and load the configuration.
V11 or later	Yes	S7-1200 supports this configuration. The connection is established automatically while you compile and load the project after replacing the device.
V12 or later	No	S7-1200 supports this configuration. The connection is established automatically while you compile and load the project after replacing the device.

During compilation of the program, you receive specific information on migrating the HMI panel.

Note

HMI TP 177B 4"

The HMI TP 177B 4" with firmware version V11.0.2 cannot be operated with S7-1200 V4. Replace the panel with a new device, if necessary.

S7-1200 expansion modules

If you are already using the following centrally plugged S7-1200 modules in your system, you must perform a firmware update for these modules in order to guarantee operation with S7-1200 V4.

- ASi - Master - CM 1243
- DP - Master - CM 1243-5
- WAN CP - CP1243-1

Newly shipped S7-1200 modules have the latest firmware installed ex works.

Protected blocks

Blocks equipped with know-how protection or copy protection cannot be converted to V4. If the project contains protected blocks, you must remove the protection prior to migration.

If these are supplied blocks and you do not know the password, ask your supplier either for the password or for a V4-compatible block.

WARNING

Preventing personal injury and material damage

Changes are made to the program during device replacement in some cases. Therefore, thoroughly check the program in a test environment after replacing the device and before putting it in operation.

Note

Additional support

You can find the latest FAQs about migrating to S7-1200 V4 in Siemens Industry Online Support:

http://support.automation.siemens.com/WW/view/en/82140966 (http://support.automation.siemens.com/WW/view/en/82140966)

If you need further assistance with migrating to S7-1200 V4, please contact SIMATIC Customer Support.

See also

Migrating to V4 (Page 223)

Special considerations after migrating to V4 (Page 224)

5.3.2 Migrating to V4 (S7-1200)

Requirement

- A CPU with firmware version V3 is available in the project.
- The project contains no protected blocks.

Procedure

Proceed as follows to replace a CPU:

1. Select the V3 CPU you want to replace.
2. Select the "Change device" command in the shortcut menu.
   The "Change device" dialog box appears.
3. Under "New device" in the tree structure, select the V4 CPU with which you want to replace your current V3 CPU.
4. Click "OK".
   The existing CPU is replaced by the new one.
5. Select the new CPU and select the "Compile > Hardware and software (only changes)" command in the shortcut menu.
   The device configuration and the user program are compiled again.
6. Optional: If necessary, apply know-how protection or copy protection to individual blocks in the program.
7. Select the new CPU and select the "Download to device > Hardware and software (only changes)" command in the shortcut menu.
   The device configuration and the user program are loaded into the new CPU.
   This completes the device replacement.

⚠ WARNING

Preventing personal injury and material damage

Changes are made to the program during device replacement in some cases. Therefore, thoroughly check the program in a test environment after replacing the device and before putting it in operation.

Note

Additional support

You can find the latest FAQs about migrating to S7-1200 V4 in Siemens Industry Online Support:

http://support.automation.siemens.com/WW/view/en/82140966 (http://support.automation.siemens.com/WW/view/en/82140966)

If you need further assistance with migrating to S7-1200 V4, please contact SIMATIC Customer Support.

See also

Basic information on upgrading to V4 (Page 221)

Special considerations after migrating to V4 (Page 224)

5.3.3 Special considerations after migrating to V4 (S7-1200)

Functional changes in V4

S7-1200 V4 offers significantly enhanced functionality. The most important functional changes that you need to consider after migrating from V3 are described briefly below.

You can find more information on S7-1200 in the "SIMATIC S7-1200 Automation System" system manual:

https://support.industry.siemens.com/cs/ww/en/view/65601780 (http://support.automation.siemens.com/WW/view/en/65601780)

Organization blocks

With S7-1200 V4, you can specifically set the interruptibility of each organization block used. When a device is replaced, all the organization blocks are configured as non-interruptible, to ensure that the executability of your V3 program remains unchanged. The OB priorities from the V3 program also remain unchanged. After migration, you can change the settings for priority and interruptibility as needed.

The behavior of diagnostic interrupts in V4 has changed as follows:

In V3, the start information always contained information on the triggering module, including the channel number. In V4, this information is only generated for a pending diagnostics event. If no diagnostic event is pending, for example, because the fault has already been corrected, only the triggering module is indicated.

Access levels

S7-1200 V4 offers an extended access level concept. The following table shows how the protection levels of the V3 firmware are indicated in V4:

V3 protection level	V4 access level	Meaning
No protection	Full access (no protection)	Unrestricted access without password protection.
Read-only	Read access	HMI access and unimpeded communication between CPUs without password protection. A password is required for changes (write access) in the CPU and for changing the operating mode of the CPU (RUN/STOP).
Write/read protection	HMI access	HMI access and unimpeded communication between CPUs without password protection. A password is required to read the data in the CPU, for changes (writing) in the CPU, and for changing the operating mode of the CPU (RUN/STOP).
-	No access (complete protection)	No access without password input. A password is required for HMI access, for reading the data in the CPU, for changing (writing) data in the CPU, and for changing the operating mode of the CPU (RUN/STOP).

Instruction libraries

After migration to S7-1200 V4, instructions from the libraries of the firmware version V3 are still available. This ensures that you can continue to use your program unchanged. In addition, S7-1200 V4 offers many new instructions which are also compatible with the instructions of the S7-1500.

You can find more information on the instruction libraries in the "SIMATIC S7-1200 Automation System" system manual:

https://support.industry.siemens.com/cs/ww/en/view/65601780 (http://support.automation.siemens.com/WW/view/en/65601780)

Motion Control

When a device is replaced, the Motion Control objects from the libraries of the firmware versions V1 and V2 are replaced with the corresponding objects from the V3 libraries. The objects from V3 libraries are compatible so that you can continue to use the programs unchanged.

The libraries of the S7-1200 V4 offer many new Motion Control functions, which are compatible with the functions of the S7-1500. If you want to use V4 libraries, select them on the "Instructions" task card after replacing the device.

You can find more information on the new Motion Control functions in the "SIMATIC S7-1200 Programmable controller" system manual.

https://support.industry.siemens.com/cs/ww/en/view/65601780 (http://support.automation.siemens.com/WW/view/en/65601780)

Web server

The following settings for operation via a web server are transferred from the V3 CPU to the V4 CPU during device replacement:

- Activate web server on this module
- Permit access only with HTTPS

If you want to operate the V4 CPU via a web server, you need to set up user accounts with assigned user rights in the user management. Only the standard web pages are available to standard users without any additional rights.

Note

Additional support

You can find the latest FAQs about migrating to S7-1200 V4 in Siemens Industry Online Support:

http://support.automation.siemens.com/WW/view/en/82140966 (http://support.automation.siemens.com/WW/view/en/82140966)

If you need further assistance with migrating to S7-1200 V4, please contact SIMATIC Customer Support.

Communication via PUT/GET

Communication via PUT/GET is enabled after the device replacement. Note that the new integrated connection types offer a security standard higher than PUT/GET communication. If you do not use PUT/GET communication, you should disable it.

See also

Basic information on upgrading to V4 (Page 221)

Migrating to V4 (Page 223)

Introduction to the TIA Portal 6

6.1 Getting started

6.1.1 Getting Started Documentation

Getting Started with the TIA Portal

The Getting Started documentation is available to help you begin using the TIA portal.

The Getting Started documentation contains instructions which show you, step-by-step, how to create a project in the TIA Portal and give you the chance to get a quick overview of all the possibilities the TIA Portal offers you.

Contents

The Getting Started documentation describes the creation of a continuous project for STEP 7 and WinCC that is expanded with each section. You start with simple basic functions, and use more complex ones as you continue with the creation of the project.

In addition to the step-by-step instructions, the Getting Started documents also give you background information that explains the functions used and illustrate how they relate to each other.

Target audience

The Getting Started documents are intended for beginners, but are also useful for users migrating from previous versions of SIMATIC STEP 7 and WinCC.

Getting Started documentation

You can download the Getting Started documents via the following links:

- Step-by-step instructions for simple operation of the TIA Portal (STEP 7 Basic, S7-1200)
  https://support.industry.siemens.com/cs/ww/en/view/40263542 (http://support.automation.siemens.com/WW/view/en/40263542/0/en)
- Configuration of an automation solution with STEP 7 Professional/WinCC Advanced (S7-300)
  https://support.industry.siemens.com/cs/ww/en/view/54430386 (https://support.industry.siemens.com/cs/ww/en/view/54430386)
- Multimedia Getting Started for S7-1500:
  http://www.siemens.com/s7-1500-getting-started (http://www.automation.siemens.com/salesmaterial-as/interactive-manuals/getting-started_simatic-s7-1500/_content/EN/content_en.html)

See also

Service and Support (https://support.industry.siemens.com/cs/de/en/)

6.2 User interface and operation

6.2.1 Starting, setting up, and exiting the TIA Portal

6.2.1.1 Starting and exiting the TIA Portal

Starting the TIA Portal

To start the TIA Portal, follow these steps:

1. In Windows, select "Start > All Programs > Siemens Automation > TIA Portal V14".
   The TIA Portal opens with the last settings used.

Exiting the TIA Portal

To exit the TIA Portal, follow these steps:

1. In the "Project" menu, select the "Exit" command.
   If the project contains any changes that have not been saved, you will be asked if you wish to save them.
   - Select "Yes" to save the changes in the current project and close the TIA Portal.
   - Select "No" to close the TIA Portal without saving the most recent changes in the project.
   - Select "Cancel" to cancel the closing procedure. The TIA Portal will remain open if you select this option.

6.2.1.2 Overview of the program settings

Overview

The following table shows the application settings that you can make:

Group	Setting	Description
General settings	User name	The user name of the user. The user name is stored in the project properties when a new project is created.
	User interface language	Language for the program interface
	Mnemonic	Specifies the mnemonics for programming: "German" uses the German mnemonics, for example, "E1.0". "International" uses international mnemonics, for example, "I1.0". For information on the differences in the mnemonics of the individual commands, refer to the description of the relevant programming language.
	Show list of recently used projects	Number of entries in the list of recently used projects in the "Project" menu
	Load most recent project during startup	The last opened project is opened automatically after the TIA Portal starts.
	Displaying truncated text in full	Texts which are truncated due to their length are displayed in a tooltip.
	Show tooltips (context-sensitive help is available)	Tooltips are displayed and you get context-sensitive help. If this function is disabled, you can open the tooltip with <F1>. The setting also affects rollouts with context-related syntax errors and input aids in STL.
	Open cascade automatically in tooltips	After a brief time, the tooltips automatically expand to display a cascade containing additional help. If this option is cleared, the tooltips must be manually expanded.
	Show hidden banner	Banners with additional information in the editor are hidden from display by default.
Reset to default	All application settings	All changes that you made in the TIA Portal after installation are undone.
	Layout of the editors	Resets the complete layout of the application to the factory state.
	Show all alarm windows	All alarm windows whose appearance was manually suppressed are displayed again.
Start view	Most recent view	Starts the program in the last view that was used. This can be either the portal view or the project view.
	Portal view	Starts the TIA Portal in the portal view every time, irrespective of the last view you worked in.
	Project view	Starts the TIA Portal in the project view every time, irrespective of the last view you worked in.
Import/export settings	Export settings to file	Opens a dialog in which you can export the selected program settings to a file.
	Import settings from file	Enables the import of previously exported program settings from a file.
Central settings	Storage location for central settings	Here you define the storage location for the file in which the settings are stored centrally.

Group	Setting	Description
View for objects in overview	Details	When several views are available, the detail view opens by default, for example, in the overview window.
	List	When several views are available, the list view opens by default, for example, in the overview window.
	Thumbnails	When several views are available, the symbol view opens by default, for example, in the overview window.
Storage settings	Recently used storage location	When a project is saved the first time, the most recently used file path is set by default.
	Default storage location	Enables the specification of file paths for: • Projects • Libraries • Configuration file for corporate libraries
	Storage location for new and retrieved projects	Enables the specification of the storage path for projects. New projects are created automatically in this path.
	Storage location for libraries	Enables the specification of the storage path for global libraries. When you create a new library, it is automatically saved to this path.
	Storage location for the configuration file for corporate libraries	Shows the path for the configuration file for the company libraries. The path cannot be changed.
Data exchange	Storage location for data import	Imported files are searched for at this storage path by default.
	Storage location for data export	This is the default storage path for the data export.
	Storage location for support packages	When support packages are downloaded, they are stored in the specified storage path and can be installed from there.
	Storage location for log files	Log files are stored at the location specified here.
	Storage location for archiving of projects	This storage path for the filing project archives is set by default.
User documentation	Show user-defined documentation in the selected editing language	The user documentation is opened in the project editing language (Shift + F1). If the user documentation does not exist in the selected editing language, the English version is opened by default.
	Display call log for user-defined documentation	Shows call logs for user documentation. Alarms for calling user-created help files are output in the Inspector window. This makes it easier to identify and correct difficulties in calling user-created help. This function is particularly useful when creating your own help files.
	Search for user-defined documentation in a central directory	Enables the saving of user-defined documentation in a directory outside the current project directory in order, for example, to make documentation available across projects.
	Central directory for user-defined documentation	Enables the specification of the storage path on which user help files for user-defined documentation can be stored.

See also

Starting and exiting the TIA Portal (Page 229)

Resetting the user interface layout (Page 269)

Changing the settings (Page 236)

Configuring the display of tooltips and tooltip cascades (Page 299)

6.2.1.3 Overview of the settings for software updates

Overview

The following table shows the available settings for software updates:

Group	Setting	Description
TIA Portal update	Check for updates daily	Checks if updates are available after every restart of the computer and then every 24 hours. If updates are found, you receive a message in the information area of the Windows taskbar.
	Check for updates now	Starts the search for updates and displays the result in the TIA Updater.

6.2.1.4 Overview of the script and text editor settings

Overview

The following table shows the available settings for script and text editors:

Group	Setting	Description
Font	Font type and size	Sets the font type and size for the text in text editors.
Font colors	Color settings	You can choose the colors for individual text elements from the respective drop-down lists in the text editors. Optional settings are available for the following text elements: • Text • Keywords • Comments • Operators • Scripts • Standard functions • Instructions/system functions • Constant strings • Numeric constants • Constant tags • Tags • Object models • Formal parameters
	Reset to default	Resets all font colors in editors to their factory settings.
Tabs	Tab width	Sets the width of tabs.
	Use tabs	Enables the use of tabs.
	Use spaces	Specifies use of space characters instead of tabs.

Group	Setting	Description
Indent	None	Text entries are not automatically indented.
	Block	The lines or the selected paragraph are automatically indented.
	Smart	The lines or the selected paragraph are automatically indented. All unnecessary spaces are also removed.
View	Show white spaces	Shows control characters within a text.
STL (Statement List)	Font type and size	Sets the font type and size for STL program code.
SCL (Structured Control Language)	Show line numbers	Shows the row numbers in SCL programs.

See also

Changing the settings (Page 236)

6.2.1.5 Overview of the print settings

Overview

The following table shows the available settings for printing:

Group	Setting	Description
General	Always print table data as pairs of values	Tables are printed as a list and not in tabular form. The corresponding values are listed for each column. Enable this option, for example, if you want to print a table that is too large for the print area.
	Timeout for print job (minutes)	If elements cannot be completely printed, a message is displayed after the configured timeout in minutes.
Hardware configuration	Active graphic view	The graphics of network and device view are included in the printout.
	Active table	A table associated with an editor is included when printing with the editor.
PLC Programming	Zoom factor	Specifies the size in which blocks are to be printed out.
	With interface	The interfaces of blocks are included in the printout.
	With comments	Comments on blocks are included in the printout.
	With line numbers	The line numbers of the program code are printed for text-based programming languages.
Motion & Technology	Dialog display/graphic	The content of the editor is printed out as a graphic if the editor supports this.
	Table	The parameters of the technology objects are printed out in the form of a table.
HMI screens	Show tab order	In the printout you can specify the order in which the runtime objects can be selected with the TAB key.

See also

Changing the settings (Page 236)

6.2.1.6 Overview of library settings

Overview

The following table shows the library settings that you can make:

Group	Setting	Description
Type connection	Disable the shortcut menu command "Terminate connection to type"	When the setting is enabled, the instance of a type in the project tree can no longer be separated from the type in a library with the help of the shortcut menu. The shortcut menu command "Terminate connection to type" is disabled.

6.2.1.7 Overview of settings for searching in a project

Overview

The following table shows the available settings for searching in a project:

Group	Setting	Description
Search in project	Activate "Search in project"	Enables or disables the search in project.
	Recreate search index	Regenerates the search index for the search in project.

6.2.1.8 Overview of the settings for cross-references

Overview

The following table shows the available settings for cross-references:

Group	Setting	Description
Recreate the cross-reference information	Recreate the cross-reference information	Creates new cross-reference information about the selected project.

6.2.1.9 Overview of the settings for managing window layouts

Overview

The following table shows the available settings for managing window layouts:

Group	Setting	Description
Manage all window layouts	Active	Enables the selection of a window layout when additional window layouts are created in addition to the default window layout.

6.2.1.10 Overview of settings for basic integrity check

Overview

The following table shows the available settings for the basic integrity check:

Group	Setting	Description
Basic integrity check	Enable basic integrity check when opening a project	By default activates the option to run the basic integrity check in the "Open project" dialog. In addition, the basic integrity check is performed automatically when opening projects from the list of recently opened projects, the reference projects as well as in multi-user engineering. Depending on the scope and application, the check may be quite time-consuming especially when using multi-user engineering.
	Enable basic integrity check when opening a library	Activates the automatic basic integrity check when opening libraries. Depending on the scope and application, the check may be quite time-consuming.

See also

Changing the settings (Page 236)

6.2.1.11 Overview of the online and diagnostic settings

Overview

The following table shows the settings that you can make for the online and diagnostic functions:

Group	Setting	Description
Preset connection path for online access	Type of the PG/PC interface	Specifies the type of the PG/PC interface that is used as the presetting in the dialogs for online access, e.g., in the "Go online" dialog.
	PG/PC interface	Specifies a particular PG/PC interface that is used as the presetting in the dialogs for online access, e.g., in the "Go online" dialog.
	Use preset connection path for online connection	Activates or deactivates the presettings for the PG/PC interface. If the check box is selected, the connection path specified in the settings is used as the presetting in the dialogs for online access.
Alarm display	Multiline	Displays the alarms in the Inspector window using multiple lines.
	Autoscroll	When a new alarm occurs, the display scrolls automatically to this alarm.
	Archive size	Maximum number of alarms that can be displayed in the Inspector window. If the set number of alarms is exceeded, older alarms are automatically deleted.

6.2.1.12 Changing the settings

Procedure

To change the settings, proceed as follows:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General" group in the area navigation to change the settings described in the previous sections. Or click on one of the other entries in area navigation to make settings for your installed products.
3. Change the settings.

Result

The change will be adopted immediately, there is no need to save it explicitly.

See also

Overview of the program settings (Page 230)

Overview of the script and text editor settings (Page 232)

Overview of the print settings (Page 233)

6.2.1.13 Exporting and importing settings

Exporting and importing settings

You can export settings of the TIA Portal to a file. When you import the settings, they are applied together. This function makes it possible to send settings to other users. Likewise, system settings can be specified centrally, to provide a unified environment for multiple engineering PCs. The TIA Portal searches for a initialization file at startup. When a initialization file is found, the settings saved therein are automatically applied after rebooting. If a more recent version of this file is available, the local settings are updated.

You can export all settings, for example:

- Language of the user interface
- Path of the company libraries
- Path of the user-defined documentation
- Window layouts
- Profiles of the hardware catalog

The initialization files are version-specific and can be imported automatically within a product version, but not in other versions of the product.

Exporting settings

You can export settings of the TIA Portal to a file, for example, to back them up or to share them with other users.

Procedure

1. Select the "Settings" command in the "Options" menu.
   The settings of the TIA Portal are displayed.
2. Select the "General" entry in the area navigation.
3. In the "Import/export settings" area, click on the "Export settings ..." button.
   The "Export settings" dialog opens.
4. Select the settings for the export.
   - Click "Select all" to select all the settings for the export.
   - Click "Deselect all" to then select individual settings for the export. It all the nested settings are selected as well.
5. If you want to accept the settings selected from previous export operations, for example for a different use of the initialization files, select the check box "Use selection from a previous export file" and enter the file in the box.
6. Enter a file name for the initialization file in the "File name" box.
7. Click "Export".
   Settings are exported to a file with the extension ".tps14".

Note

Empty entry fields in the export file

Entry fields, such as "Central settings", are available in the settings. If you do not fill out one of these entry fields and nevertheless select the initialization for the export, an empty initialization will be exported. If the subsequently import the initialization file, the corresponding local settings are overwritten by the empty contents of important initialization file.

If you do not want these fields to be deleted by an import, disable the non-filled entry fields during export of the initialization file.

Importing settings automatically

When the TIA Portal starts up, you can automatically import the settings from a initialization file, for example, to incorporate a centrally defined environment.

Procedure

1. Select the "Settings" command in the "Options" menu.
   The settings of the TIA Portal are displayed.
2. Select the "General" entry in the area navigation.
3. In the "Central settings" area, click on the "Browse" button.

4. Select the setting file with the extension ".tps14".
5. Click "Open".
   The settings are applied after a restart of the TIA Portal.

Note

Priority of central settings

Central settings have priority over manually imported settings and override them.

Importing settings manually

You can import settings of the TIA Portal manually from a initialization file, for example, to take over settings from other users.

Procedure

1. Select the "Settings" command in the "Options" menu.
   The settings of the TIA Portal are displayed.
2. Select the "General" entry in the area navigation.
3. In the "Import/export settings" area, click on the "Import settings ..." button.
4. Select the setting file with the extension ".tps14".
5. Click "Open".
   The settings are applied after a restart of the TIA Portal.

6.2.2 Layout of the user interface

6.2.2.1 Views

Views

Three different views are available for your automation project:

- The portal view is a task-oriented view of the project tasks.
- The project view is a view of the components of the project, as well as the relevant work areas and editors.
- The library view (Page 243) shows the elements of the project library and the open global libraries.

You can change over between the two views using a link.

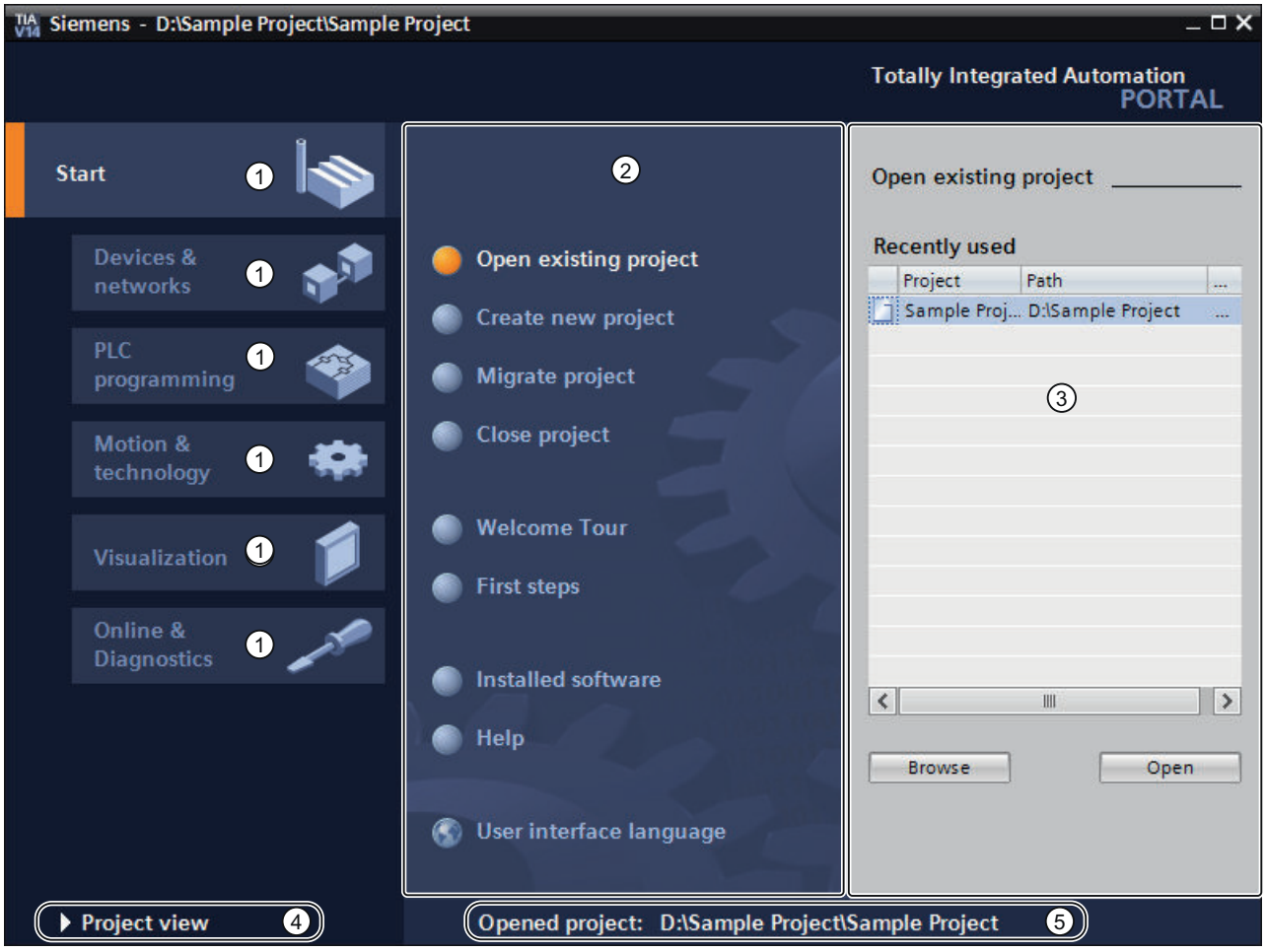
6.2.2.2 Portal view

Purpose of the portal view

The portal view provides you with a task-oriented view of the tools. Here, you can quickly decide what you want to do and call up the tool for the task in hand. If necessary, the view changes automatically to the project view (Page 240) for the selected task.

Layout of the portal view

The following figure shows an example of the components in the portal view:



① Portals for different tasks
② Actions for the selected portal
③ Selection panel for the selected action
④ Switch to project view
⑤ Display of the project that is currently open

Portals

The portals provide the basic functions for the individual task areas. The portals that are provided in the portal view depends on the products that have been installed.

Actions for the selected portal

Here, you will find the actions available to you in the portal you have selected. You can call up the help function in every portal on a context-sensitive basis.

Selection panel for the selected action

The selection panel is available in all portals. The content of the panel adapts to your current selection.

Switch to project view

You can use the "Project view" link to change to the project view.

Display of the project that is currently open

Here, you can obtain information about which project is currently open.

See also

Function and structure of the project tree (Page 243)

Basics of the work area (Page 247)

Inspector window (Page 255)

Basics on task cards (Page 257)

Details view (Page 260)

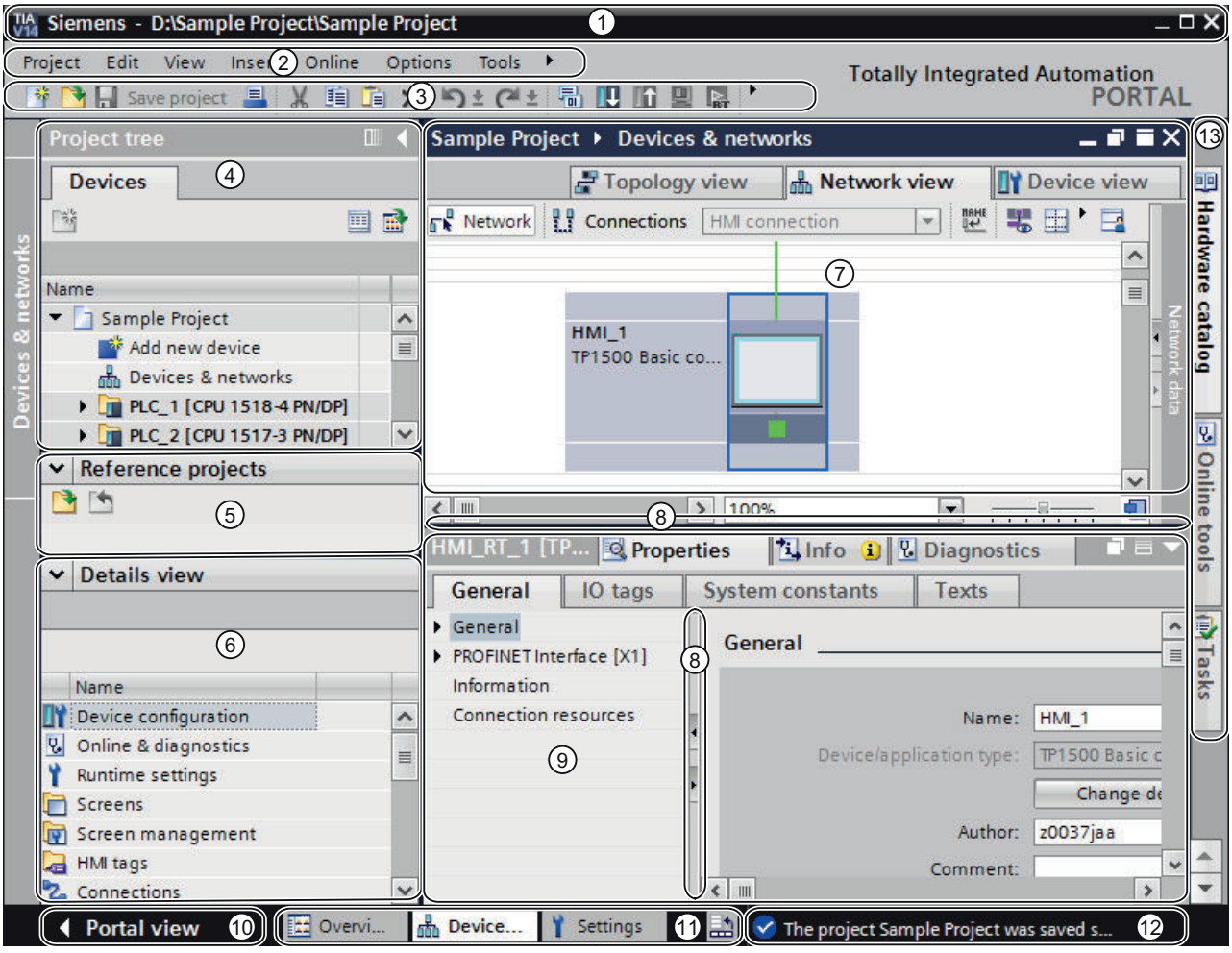
6.2.2.3 Project view

Purpose of the project view

The project view is a structured view of all components of the project.

Layout of the project view

The following figure shows an example of the components of the project view:



① Title bar
② Menu bar
③ Toolbar
④ Project tree (Page 243)
⑤ Reference projects (Page 259)
⑥ Details view (Page 260)
⑦ Work area (Page 257)
⑧ Dividers
⑨ Inspector window (Page 255)
⑩ Changing to the portal view (Page 239)
⑪ Editor bar
⑫ Status bar with progress display
⑬ Task cards (Page 257)

Title bar

The name of the project is displayed in the title bar.

Menu bar

The menu bar contains all the commands that you require for your work.

Toolbar

The toolbar provides you with buttons for commands you will use frequently. This gives you faster access to these commands.

Dividers

Dividers separate individual components of the program interface. The arrows on the dividers allow you to display and hide the adjacent sections of the user interface.

Changing to the portal view

You can use the "Portal view" link to change to the portal view.

Editor bar

The Editor bar displays the open editors. This allows you to quickly switch between open elements. If you have opened numerous editors, you can display the editors of the same type as a group.

Status bar with progress display

In the status bar, you will find the progress display for processes that are currently running in the background. This also includes a progress bar that shows the progress graphically. Hover the mouse pointer over the progress bar to display a tooltip providing additional information on the active background process. You can cancel the background processes by clicking the button next to the progress bar.

If no background processes are currently running, the status bar displays the last generated alarm.

See also

Basics of the work area (Page 247)

6.2.2.4 Library view

Function of the library view

The library view provides an overview of the elements in the project library and the open global libraries. You can switch to the library view using the "Libraries" task card.

See also: Overview of the library view (Page 447)

6.2.2.5 Project tree

Function and structure of the project tree

Function of the project tree

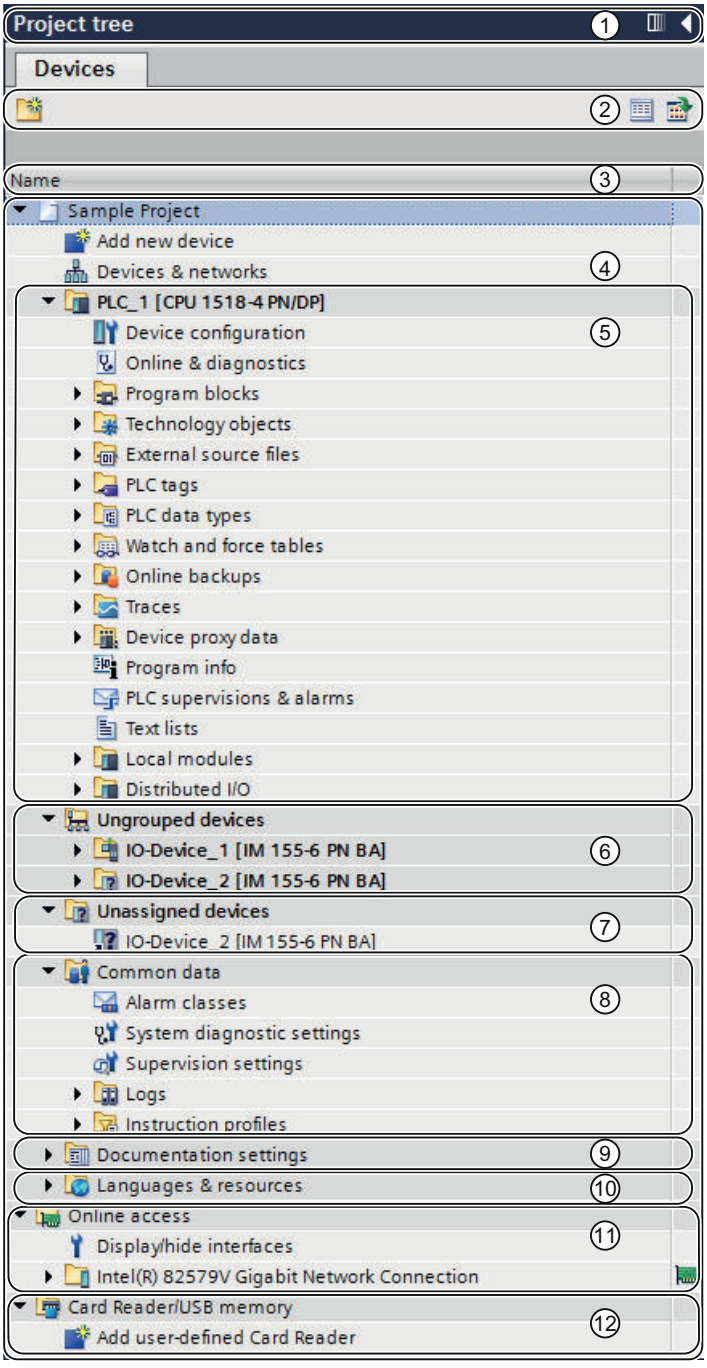
Using the project tree features gives you access to all components and project data. You can perform the following tasks in the project tree:

- Add new components
- Edit existing components
- Scan and modify the properties of existing components

You can select the objects of the project tree either with the mouse or via the keyboard by typing the first letter of the desired object. If more than one object begins with the same letter, the next lower object is selected. The project tree must be the focused user interface element in order for you to select an object with its initial letter.

Layout of project tree

The following figure shows an example of the project tree areas:



① Title bar
② Toolbar
③ Table header
④ Project

⑤ Devices
⑥ Ungrouped devices
⑦ Unassigned devices
⑧ Common data
⑨ Documentation settings
⑩ Languages & resources
⑪ Online access
⑫ Card Reader / USB memory

Title bar

The title bar of the project tree has a button for automatically and manually collapsing the project tree. Once it is collapsed manually, the button is "Reduced" to the left-hand margin. It changes from an arrow pointing left to one that is pointing right, and can now be used to reopen the project tree. You can use the "Collapse automatically" button collapse to project tree automatically when you do not need it.

See also: Maximizing and minimizing the work area (Page 249)

Toolbar

You can do the following tasks in the toolbar of the project tree:

- Create a new user folder; for example, in order to group blocks in the "Program blocks" folder or group devices.
- Navigate forward to the source of a link and back to the link itself.
  There are two buttons for links in the project tree. You can use these to navigate from the link to the source and back.
- Show an overview of the selected object in the work area.
  When the overview is displayed, the lower-level objects and actions of the elements in the project tree are hidden.

Table header

The "Name" column is shown by default. You can also show the columns "Type name" and "Version". If you are showing the additional columns, you see the name of the respective type as well as the version used for instances of types from the library.

Project

You will find all the objects and actions related to the project in the "Project" folder, e.g.:

- Devices
- Languages & resources
- Online access

Device

There is a separate folder for each device in the project, which has an internal project name. Objects and actions belonging to the device are arranged inside this folder.

Ungrouped devices

All distributed I/O devices in the project are collected together in the "Ungrouped devices" folder.

Unassigned devices

Distributed I/O devices that are not assigned to a distributed I/O system appears as links in the "Unassigned devices" folder.

Common data

This folder contains data that you can use across more than one device, such as common alarm classes, logs and scripts.

Documentation settings

In this folder, you can specify the layout for the printed project documentation.

Languages & resources

You can determine the project languages and texts in this folder.

Online access

This folder contains all the interfaces of the programming device / PC, even if they are not used for communication with a module.

Card Reader / USB memory

This folder is used to manage all card readers an other USB storage media connected to the programming device / PC.

See also

Portal view (Page 239)

Project view (Page 240)

Basics of the work area (Page 247)

Inspector window (Page 255)

Basics on task cards (Page 257)

Details view (Page 260)

Showing and hiding columns (Page 247)

Showing and hiding columns

You can show additional columns in the project tree, if necessary. These additional columns show the name of the type associated with an instance as well as its version number.

Procedure

To show or hide additional table columns for the associated type and its version number, follow these steps:

1. Click the "Display/hide column heading" in the toolbar of the project tree.
   The column view of the project tree is enabled and a table header is displayed.
2. Right-click on the table header of the project tree.
3. Select the "Show/Hide" command in the shortcut menu, and select the columns you want to display.
   The selected columns are displayed or hidden.

See also

Function and structure of the project tree (Page 243)

6.2.2.6 Work area

Basics of the work area

Function of the work area

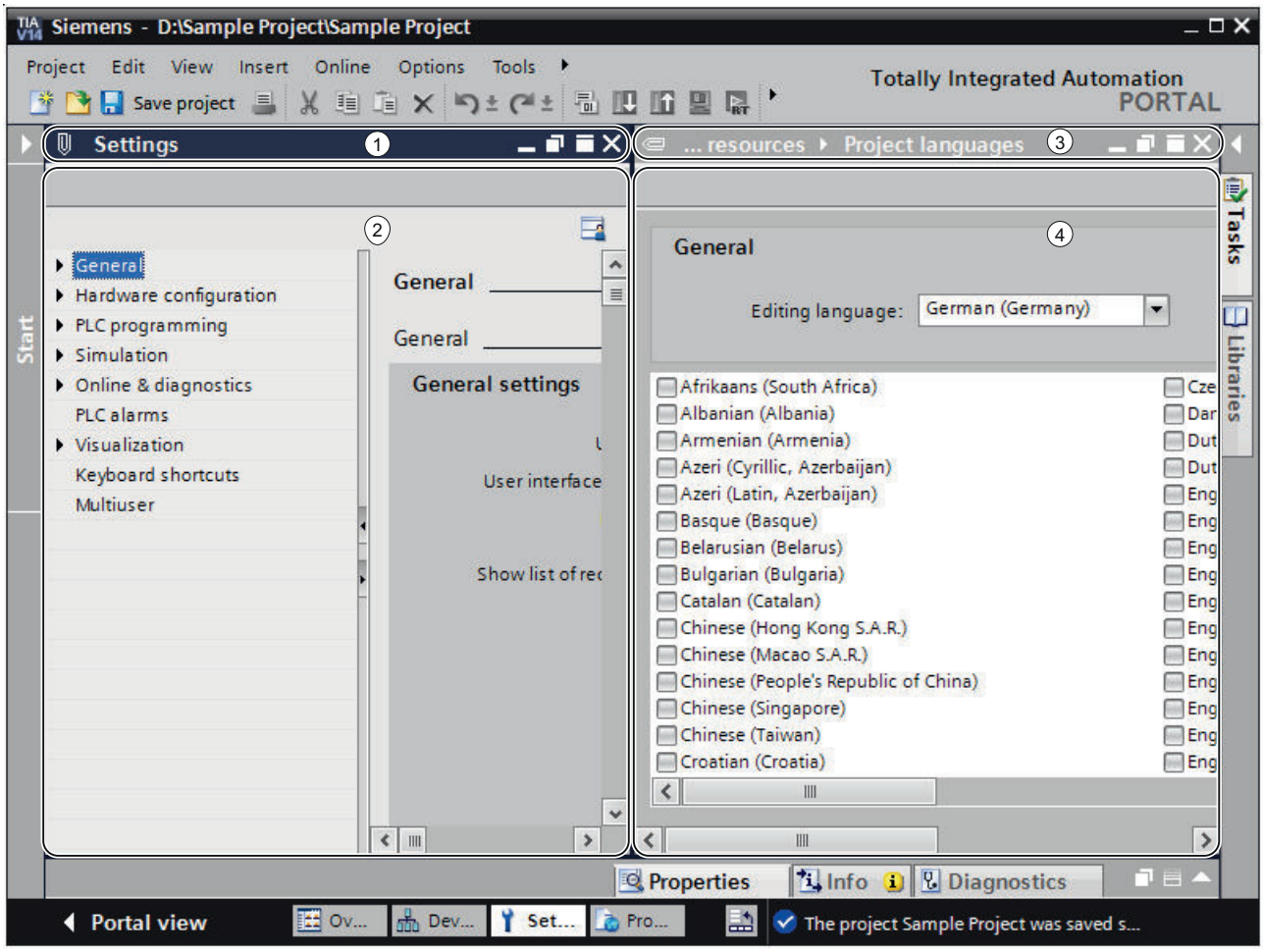
The objects that you can open for editing purposes are displayed in the work area. These objects include, for example:

- Editors and views
- Tables

You can open several objects. However, normally it is only possible to see one of these at a time in the work area. All other objects are displayed as tabs in the Editor bar. If, you would like to view two objects at the same time when performing certain tasks, you can tile the work area either horizontally or vertically or undock elements of the work area. If no objects are open, the work area will be empty.

Layout of the work area

The following figure shows an example of a vertically split work area:



① Title bar of left-hand editor
② Work area of left-hand editor
③ Title bar of right-hand editor
④ Work area of right-hand editor

See also

Maximizing and minimizing the work area (Page 249)

Splitting the work area (Page 250)

Floating the work area elements (Page 251)

Using grouped elements of the work area (Page 252)

Minimizing and maximizing elements of the work area (Page 254)

Switching between the elements in the work area (Page 254)

Saving a layout of editors and tables (Page 268)

Save user interface layout (Page 266)

Maximizing and minimizing the work area

You have the option to adapt the work area to make it as large as possible. You can use the following function for this:

- Maximizing the work area
  You can close the task cards, project tree and inspector window with a single click. This increases the size of the work area. You can minimize the work area again at any time in order to return to the previous view.
- Collapsing task cards, project tree, and Inspector window automatically
  You can use the "Collapse automatically" option for the task cards, project tree, and Inspector window. This function causes these items to collapse automatically when you don't need them.

Maximizing and minimizing the work area

To maximize the work area, follow these steps:

1. Open an element such as an editor or a table.
   The element appears in the work area.
2. Click the "Maximize" button in the title bar of the element.
   The task cards, project tree and inspector window collapse, and the work area is shown with its maximum dimensions.

To minimize the work area again, follow these steps:

1. Click the "Embed" button in the title bar of the displayed element.
   This restores the view that existed before the work area was maximized. That is, if the task cards, project tree, or Inspector window were expanded before, they will be expanded again.

Collapsing task cards, project tree, and Inspector window automatically

To collapse the task cards automatically, follow these steps:

1. Click "Collapse automatically" in the title bar of the task cards.
   The task cards collapse when you click anywhere outside the task cards.
2. To use the task cards, click the collapsed task cards.
3. The task cards expand and are available for use. The "Collapse automatically" option remains enabled.

To collapse the project tree automatically, follow these steps:

1. Click "Collapse automatically" in the title bar of the project tree.
   The project tree collapses when you click anywhere outside the project tree.
2. To use the project tree, click the collapsed project tree.
   The project tree expands and is available for use. The "Collapse automatically" option remains enabled.

To collapse the Inspector window automatically, follow these steps:

1. Click "Collapse automatically" in the title bar of the Inspector window.
   The Inspector window collapses when you click anywhere outside the Inspector window.
2. To use the Inspector window, click the collapsed Inspector window.
   The Inspector window expands and is available for use. The "Collapse automatically" option remains enabled.

To disable the automatic collapse option, follow these steps:

1. Click "Expand permanently" again in the relevant window.
   The Collapse automatically" option is disabled, and the window remains expanded.

See also

Basics of the work area (Page 247)

Splitting the work area (Page 250)

Floating the work area elements (Page 251)

Using grouped elements of the work area (Page 252)

Minimizing and maximizing elements of the work area (Page 254)

Switching between the elements in the work area (Page 254)

Saving a layout of editors and tables (Page 268)

Splitting the work area

You can split the work area vertically or horizontally.

Procedure

To split the work area vertically or horizontally, follow these steps:

1. In the "Window" menu, select the "Split editor space vertically" or "Split editor space horizontally" command.
   The element you have clicked and the next element in the Editor bar will be displayed either next to one another or one above the other.

Note

If no elements are open in the work area, the "Split editor space vertically" and "Split editor space horizontally" functions will not be available.

See also

Basics of the work area (Page 247)

Maximizing and minimizing the work area (Page 249)

Floating the work area elements (Page 251)

Using grouped elements of the work area (Page 252)

Minimizing and maximizing elements of the work area (Page 254)

Switching between the elements in the work area (Page 254)

Saving a layout of editors and tables (Page 268)

Floating the work area elements

You can float work area elements in their own separate window:

- Editors
- Tables
- Setting windows
- Task cards
- Inspector window

You can embed floating elements again in the work area at any time.

Floating the work area elements

To float work area elements, follow these steps:

1. Click the "Float" button in the title bar of the element.
   The element will be released from the work area and displayed in its own window. You can now place the window wherever you wish. If you have minimized the window, you can restore it via the editor bar.

Embedding elements in the work area

To embed elements in the work area again, follow these steps:

1. Click the "Embed" button in the title bar of the element.
   The element will appear in the work area again.

See also

Basics of the work area (Page 247)

Maximizing and minimizing the work area (Page 249)

Splitting the work area (Page 250)

Using grouped elements of the work area (Page 252)

Minimizing and maximizing elements of the work area (Page 254)

Switching between the elements in the work area (Page 254)

Saving a layout of editors and tables (Page 268)

Using grouped elements of the work area

If you have opened several similar elements as editors or spreadsheets, you can have them displayed in the editor bar as a group. You can use these groups as follows:

- Displaying individual elements of a group
- Displaying all elements of a group in separate windows
- Embedding all displayed elements of a group in the work area
- Minimizing all displayed elements
- Closing all elements of a group

You can undo the grouping of similar elements at any time.

Grouping or ungrouping similar elements

To group similar elements in the editor bar or ungroup them again, follow these steps:

1. Click on the "Group editors/Ungroup" button on the right in the editor bar.
   Each time you click the button, similar elements are grouped or existing groups are dissolved. The icon of the button changes slightly here to enable you to quickly identify the mode in which you are working. The configured mode is retained even after the elements are closed.

Displaying individual elements of a group

To display individual elements of a group, follow these steps:

1. In the editor bar, click the group containing the element you want to display.
   All list of all available elements of the group is displayed.
2. Click the element that you want to display.

Displaying all elements of a group in separate windows

To display all elements of a group in separate windows, follow these steps:

1. In the editor bar, right-click the group whose elements you want to display.
2. Select "Restore group" in the shortcut menu.
   All elements of the group are displayed in separate, overlapping windows. Move the windows in order to see the individual element, or choose an element via the group in the editor bar.

Embedding all displayed elements of a group in the work area

To embed all elements of a group displayed in separate windows in the work area again, follow these steps:

1. In the editor bar, right-click the group whose elements you want to embed.
2. Select "Embed group" in the shortcut menu.
   All elements of the group are embedded in the work area again.

Minimizing all displayed elements

To minimize all elements of a group, follow these steps:

1. In the editor bar, right-click the group whose elements you want to minimize.
2. Select "Minimize group" in the shortcut menu.
   All elements of the group are minimized. However, the minimized elements remain open and can be quickly maximized again via the group in the editor bar.

Closing all elements of a group

To close all elements of a group, follow these steps:

1. In the editor bar, right-click the group whose elements you want to close.
2. Select "Close group" in the shortcut menu.
   All elements of the group are closed. The group is removed.

See also

Basics of the work area (Page 247)

Maximizing and minimizing the work area (Page 249)

Splitting the work area (Page 250)

Floating the work area elements (Page 251)

Minimizing and maximizing elements of the work area (Page 254)

Switching between the elements in the work area (Page 254)

Saving a layout of editors and tables (Page 268)

Minimizing and maximizing elements of the work area

You can minimize the elements that are open in the work area, such as editors or tables, as needed. However, an element remains open even if it has been minimized, and can quickly be maximized again using the editor bar.

Minimizing elements in the work area

To minimize elements in the work area, follow these steps:

1. Click the "Minimize" button in the title bar of the element.
   The element is minimized, but can still be accessed via the editor bar.

To minimize all elements at the same time, follow these steps:

1. In the "Window" menu, select the "Minimize all" command.

Maximizing elements in the work area

To maximize elements in the work area again, follow these steps:

1. Click the required element in the editor bar.
   The element is maximized and appears in the work area.

See also

Basics of the work area (Page 247)

Maximizing and minimizing the work area (Page 249)

Splitting the work area (Page 250)

Floating the work area elements (Page 251)

Using grouped elements of the work area (Page 252)

Switching between the elements in the work area (Page 254)

Saving a layout of editors and tables (Page 268)

Switching between the elements in the work area

You can switch between the elements in the work area at any time.

Switching between the elements in the work area

To switch to the previous or next editor, follow these steps:

1. In the "Window" menu, select the "Next editor" or "Previous editor" command.
   The next or previous editor will be displayed.

See also

Basics of the work area (Page 247)

Maximizing and minimizing the work area (Page 249)

Splitting the work area (Page 250)
Floating the work area elements (Page 251)
Using grouped elements of the work area (Page 252)
Minimizing and maximizing elements of the work area (Page 254)
Saving a layout of editors and tables (Page 268)

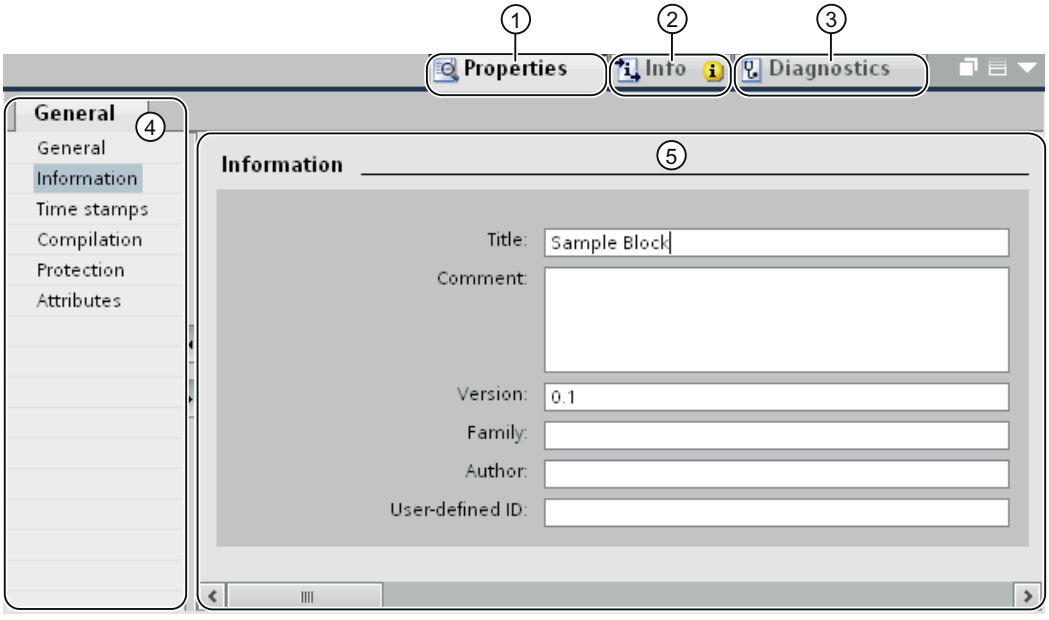
6.2.2.7 Inspector window

Function of the Inspector window

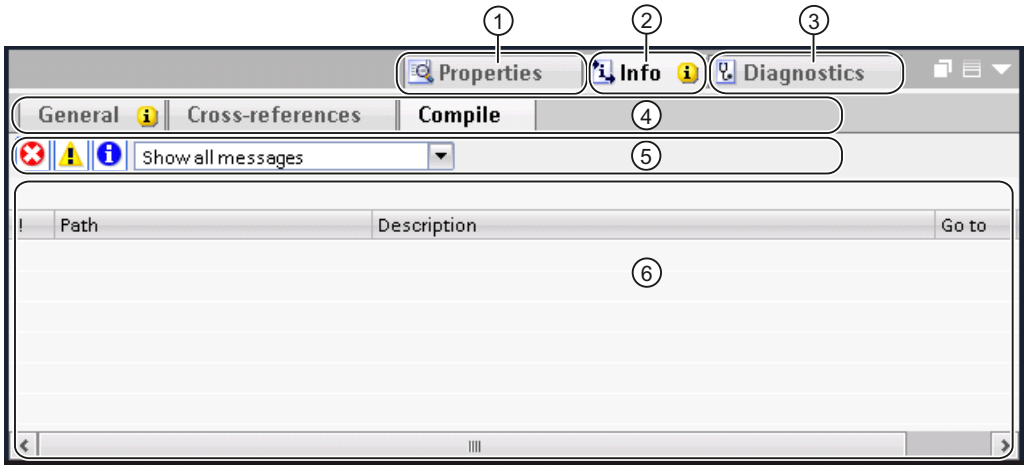
Additional information on an object selected or on actions executed are displayed in the inspector window.

Layout of the Inspector window

The following figures show the components of the Inspector window:



① "Properties" tab
② "Info" tab
③ "Diagnostics" tab
④ Area navigation within the "Properties" tab
⑤ Content of the "Properties" tab



① "Properties" tab
② "Info" tab
③ "Diagnostics" tab
④ Navigation through additional tabs within a tab (only available in the "Info" and "Diagnostics" tabs)
⑤ Toolbar (only available in the secondary "General" and "Compile" tabs of the "Info" tab)
⑥ Content of the "Compile" tab in the "Info" tab

"Properties" tab

This tab displays the properties of the object selected. You can change editable properties here.

"Info" tab

This tab displays additional information on the object selected, as well as alarms on the actions executed (such as compiling).

"Diagnostics" tab

This tab provides information on system diagnostics events, configured alarm events, and connection diagnostics.

Navigation within the tabs

You can use area navigation and the lower-level tabs to display the information you require within the tabs.

Toolbar

You can use the toolbar in the "General" and "Compile" tabs within the "Info" tab to specify which types of alarms are to be displayed. You can enable or disable the display for the following alarm types:

- Errors
- Warnings
- Information

See also

Function and structure of the project tree (Page 243)

Basics of the work area (Page 247)

Portal view (Page 239)

Project view (Page 240)

Basics on task cards (Page 257)

Details view (Page 260)

6.2.2.8 Task cards

Basics on task cards

Function of task cards

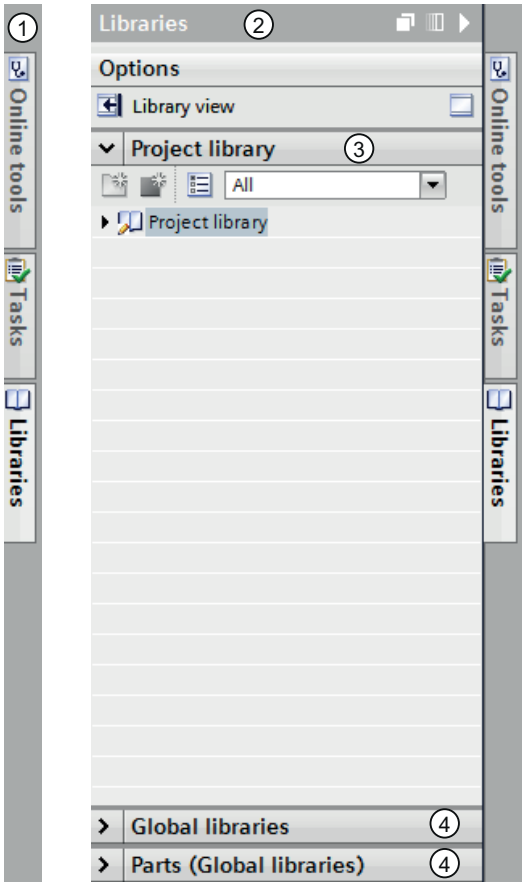
Depending on the edited or selected object, task cards that allow you perform additional actions are available. These actions include:

- Selecting objects from a library or from the hardware catalog
- Searching for and replacing objects in the project
- Dragging predefined objects to the work area

The task cards available can be found in a bar on the right-hand side of the screen. You can collapse and reopen them at any time. Which task cards are available depends on the products installed. More complex task cards are divided into panes that you can also collapse and reopen.

Layout of task cards

The following figure shows an example of the bar with the task cards:



① Task cards closed
② Task card open
③ Opened palette of a task card
④ Closed palette of a task card

See also

Changing the pane mode (Page 259)

Function and structure of the project tree (Page 243)

Basics of the work area (Page 247)

Inspector window (Page 255)

Portal view (Page 239)

Project view (Page 240)

Details view (Page 260)

Changing the pane mode

You can choose between two pane modes:

- Single pane mode:
  Only one pane is open at any given time. If you open another pane, the previously opened pane is closed automatically.
- Multi-pane mode:
  You can open several panes at the same time.

Procedure

To change the pane mode, follow these steps:

1. Click the "Change pane mode" button above the panes inside a task card.

See also

Basics on task cards (Page 257)

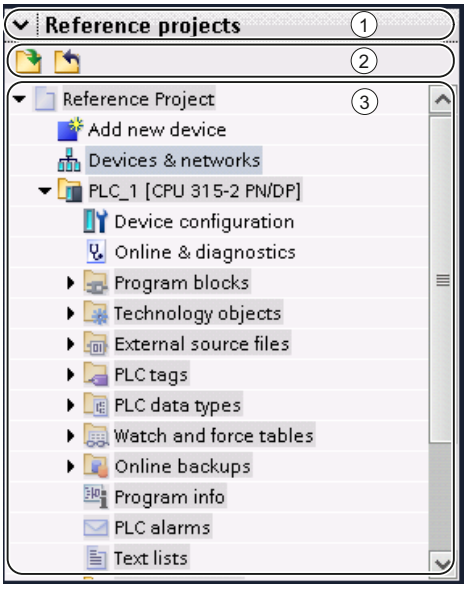
6.2.2.9 Reference projects

Function of reference projects

In the "Reference projects" palette, you can open other projects in addition to the current project. These reference projects are write-protected and cannot be edited. However, you can drag the objects of a reference project into your current project and further edit them there. You can also compare the objects of a reference project to the objects of your current project.

Layout of the "Reference projects" palette

The following figure shows the layout of the "Reference projects" palette:



① Title bar
② Toolbar
③ Opened reference projects

Title bar

The arrow for closing the palette is located in the title bar of the "Reference projects" palette. Once it is closed, the direction in which the arrow is pointing changes from downwards to right. It can now be used to reopen the palette.

Toolbar

The toolbar contains buttons for opening and closing reference projects.

Opened reference projects

Opened reference projects are displayed as read-only with their objects and their hierarchical structure.

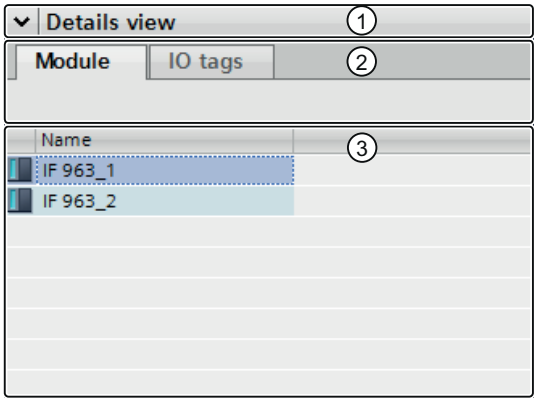
6.2.2.10 Details view

Purpose of the details view

The detail view shows certain content of the selected object is in the overview window or in the project tree. This might include text lists or tags.

Layout of the details view

The following figure shows an example of the details view:



① Title bar
② Tab
③ Content of the selected object

Title bar

The arrow for closing the details view is located in the title bar of the details view. After it has closed, the direction in which the arrow is pointing changes from left to right. It can now be used to reopen the details view.

Tab

The tab displayed in the detail view depends on the selected object.

- The "Modules" tab is displayed when you select a device, a device link or a folder that contains one of these objects.
  The "Modules" tab displays the installed modules or sub-modules of a device in the detail view, when you select the device itself or a device link in the project tree.
- The "IO tags" displays existing IO tags for selected devices and device links to below the "Local modules" folder or a device folder. The name of the simple tag or data element of a PLC data type is displayed in addition to the data type and the address.

  Note

  If a simple tag is defined for a channel address but it is also within the address range of the data element of a PLC data type, only the name of the simple tag is displayed.

Objects

The displayed content varies depending on the selected object. You can move the content of objects from the details view to the required location using drag-and-drop.

See also

Function and structure of the project tree (Page 243)

Basics of the work area (Page 247)

Inspector window (Page 255)

Basics on task cards (Page 257)

Portal view (Page 239)

Project view (Page 240)

6.2.2.11 Overview window

Overview window

Functions of the Overview window

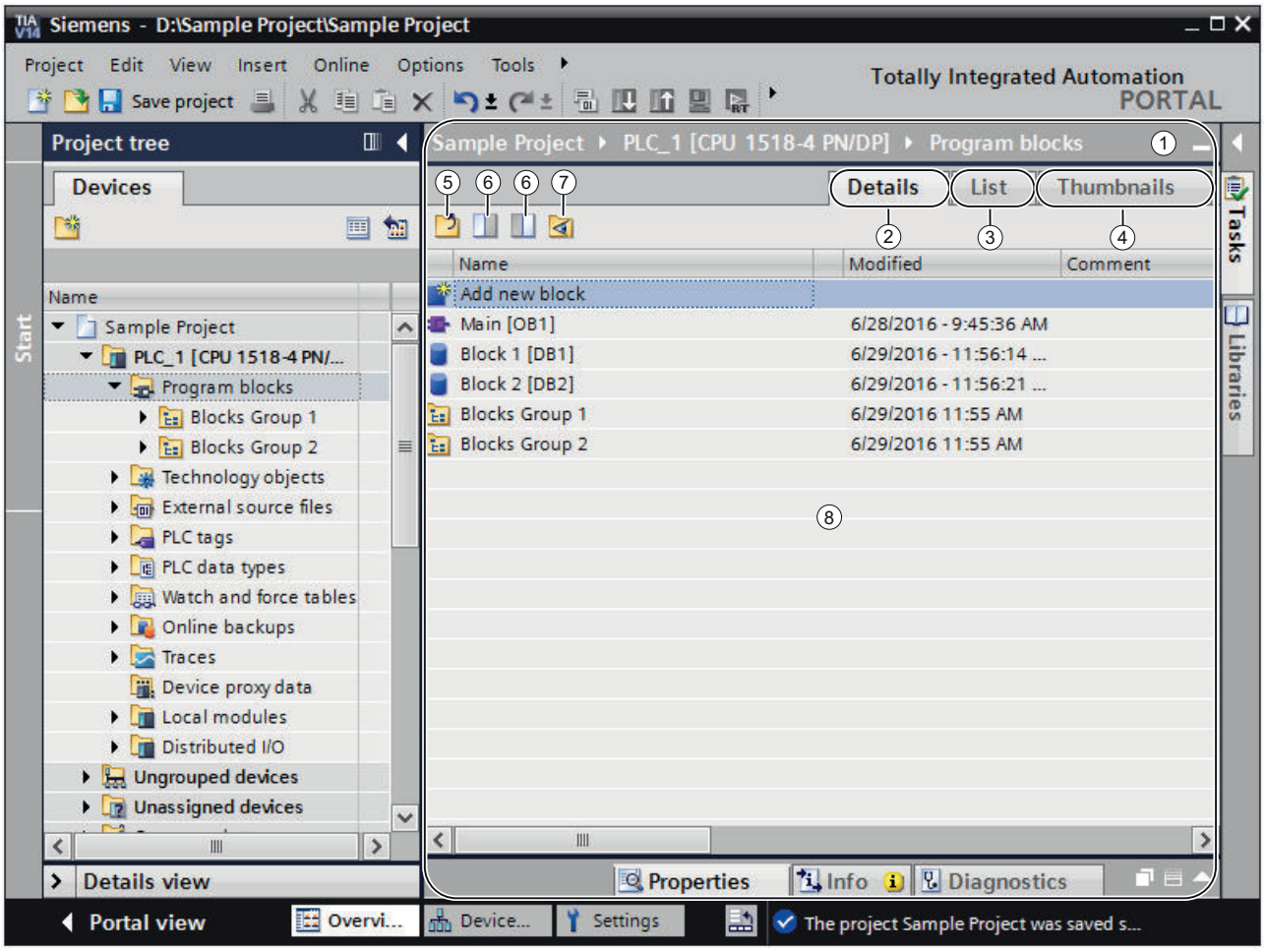
The Overview window supplements the project tree. The Overview window shows the contents of the folder currently selected in the project tree.

In addition, you can perform the following actions in the Overview window:

- Open objects
- Display and edit the properties of objects in the Inspector window
- Rename objects
- Call object-specific actions from the shortcut menu
- Compare objects side by side
- Perform various object operations, such as inserting objects from the library via drag-and-drop and moving, copying, pasting, and deleting objects

Layout of the Overview window

The following figure shows the components of the Overview window:



① Overview window

② Switch to the Details view

③ Switch to the List view

④ Switch to the Icon view

⑤ Move to higher level

⑥ Split the overview window in two. Either the right or left half of the overview window is synchronized. Clicking again cancels the split.

⑦ All elements within a selected folder are displayed even if these are located in lower-level groups. This option is only available in details view.

⑧ Contents of the object selected in the project tree.

Display forms of the Overview window

The content of the Overview window can be displayed as follows:

- Details view
  The objects are displayed in a list with additional information, such as the date of the last change.
- List view
  The objects are displayed in a simple list.
- Icon view
  The objects are displayed as icons according to category.

See also

Comparing objects in the overview window (Page 264)

Sorting the details view of the overview window (Page 265)

Overview of the library view (Page 447)

Comparing objects in the overview window

You can display the contents of two folders or objects side by side in the Overview window. The Overview window is split in half and you can display different contents on the left and right sides.

In addition, you can use a drag-and-drop operation to move objects between the split windows. Thus, for example, you can move contents from one window to the other.

Procedure

To split the Overview window in half or cancel the split, follow these steps:

1. In the toolbar, click on the "Synchronize left side" or "Synchronize right side" icon to split the overview window. Either the left or the right side of the overview window synchronized with the contents of the selected object in the project tree.
2. To cancel the split, click again on the previously selected icon.

See also

Overview window (Page 262)

Sorting the details view of the overview window

You have several options for adapting the display in the details view of the overview window:

- Adding additional columns
  Some columns are hidden by default to increase clarity. You can display hidden columns if needed. The columns available depend on the selected object.
- Display of folder contents in a flat hierarchy
  Folder contents can be displayed in a flat hierarchy. All the content is displayed at once even if it is located in different groups.
- Sorting the table columns
  You can sort individual columns of the table in ascending or descending order.

Showing or hiding columns

To show or hide additional table columns, follow these steps:

1. Right-click the title bar of the table.
2. Select the "Show/Hide" command in the shortcut menu, and select the columns you want to display.

Displaying folder contents in flat hierarchy

To display the content of a folder in a flat hierarchy, follow these steps:

1. Select the required folder in the project tree or in the library navigation of the library view.
2. Click the "Show subordinate elements" icon in the toolbar.
   All elements are displayed at once in the table even if they are located in subfolders.

Sorting a table in ascending or descending order

To sort the table by a column in ascending or descending order, follow the steps below:

1. Click the table header of a column if you want to sort the column in ascending order.
2. Click again on the same column of the table header to sort the column in descending order.
3. Click a third time on the table header of the same column to cancel the sorting.

See also

Overview window (Page 262)

6.2.2.12 User interface layout

Save user interface layout

Options for backing up the user interface layout

When you make a change to the user interface, this is retained even after a restart of the TIA Portal. A change to the user interface layout includes, for example, moving a window or adjusting the size of an editor.

In addition to the automatic saving of the user interface layout, you have the option of manually backing up specific layouts:

- Save the window layout
  You can save the layouts of the windows and editors of the TIA Portal manually and restore these at a later time. It is possible to call five window layouts using a key combination. Use this function, for example, if you are working with a notebook to which you connect an external monitor when necessary. You can create a window layout for mobile use on the notebook display and another layout for when you work at the office with an external monitor.
- Save the layout within editors
  With some editors, you can adjust the display. You can, for example, adjust the width of tables or show or hide individual table columns.

See also

Save window layout (Page 266)

Load window layout (Page 267)

Managing window layouts (Page 268)

Saving a layout of editors and tables (Page 268)

Resetting the user interface layout (Page 269)

Basics of the work area (Page 247)

Save window layout

You can save the current window layout in order to call it again in the same form at a later time.

Procedure

To save a window layout, follow these steps:

1. Arrange all windows in the way in which you want to save them.
2. In the "Window" menu, select the "Save window layout as" command.
   The "Save current window layout as" dialog opens.
3. Enter a name for the window layout in the "Name" field.

4. Enter a description of the window layout in the "Description" field in order to be able to identify the window layout more easily later.
5. Click "Save".

Result

The new window layout is saved in the last position after the existing saved window layouts. The first five window layouts can be called using a key combination.

See also

Save user interface layout (Page 266)

Load window layout

If you have already saved a window layout, you can load this, allowing you to quickly adjust your work environment to the respective conditions. You can load the first five window layouts using quick access via the "Window" menu or via a key combination.

If you load a window layout and then make changes to the arrangement of the window, you can restore the originally saved window layout.

Using quick access to load window layouts 1 to 5

To load one of the first five saved window layouts, follow these steps:

1. In the "Window" menu, select a window layout or select the key combination <Alt+Shift+[1 ... 5]>.

Loading additional window layouts

To load a window layout that is not among the first five window layouts, follow these steps:

1. In the "Window" menu, select the "Additional window layouts" command.
   The "Manage all windows layouts" dialog opens.
2. Select the desired window layout.
3. Click "OK".

Restore window layout

To go back to a saved window layout, follow these steps:

1. In the "Window" menu, select the "Restore window layout" command or select the key combination <Alt+Shift+0>.

See also

Save user interface layout (Page 266)

Managing window layouts

You can carry out the following actions with existing window layouts:

- Changing the order of window layouts
  The order of the window layouts is important, as the first five window layouts can be called directly via the "Window" menu and via a key combination.
- Select a window layout
  If a window layout is not one of the first five window layouts, you can call it using the "Manage all window layouts" dialog box.
- Deleting window layouts

Procedure

To manage the existing window layouts, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The settings of the TIA Portal are displayed.
2. In the area navigation, navigate to the entry "General > Manage all window layouts".
3. Select a window layout.
   The selected window layout is immediately activated.
4. Click the "Up" or "Down" symbol to move the window layout up or down.
5. Click the "Delete" symbol to delete the selected window layout.

See also

Save user interface layout (Page 266)

Saving a layout of editors and tables

You have the option of adapting editors and tables to meet your requirements. For example, you can hide columns in tables that you don't need. You can then save your customized view.

Procedure

To save the layout of editors and tables in the work area, follow these steps:

1. Adapt the editor or table according to your requirements.
2. Click the "Save window settings" button in the editor or table.

Result

The layout is saved. When you reopen the editor or table, this layout will be used.

See also

Basics of the work area (Page 247)

Maximizing and minimizing the work area (Page 249)

Splitting the work area (Page 250)

Floating the work area elements (Page 251)

Using grouped elements of the work area (Page 252)

Minimizing and maximizing elements of the work area (Page 254)

Switching between the elements in the work area (Page 254)

Save user interface layout (Page 266)

Resetting the user interface layout

Every change you make to the layout of the user interface is saved. The changes are thus available even after a restart of the TIA Portal. For example, if you change the height and width of a text editor or the division of a table, your changes are retained so that you don't have to re-customize elements every time.

In some cases, however, it may be helpful to restore the original layout settings; for example, if another user prefers a different arrangement of the user interface.

Procedure

To reset the user interface settings to the default, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General" group in the area navigation.
3. Click the "Reset to default" button under "Reset to default > Editor layout".

Result

The default settings for the user interface are restored.

See also

Overview of the program settings (Page 230)

Save user interface layout (Page 266)

6.2.3 Keyboard operation in the TIA Portal

6.2.3.1 Operation of the TIA Portal with the keyboard

You can navigate through the TIA Portal using the keyboard, for example if you do not have a mouse available at the given moment. Many functions are also accessible via keyboard shortcuts. You can find an overview of all keyboard shortcuts in the settings for the TIA Portal.

In the following sections, you will learn how to navigate in the TIA Portal using the keyboard, edit objects, and customize the TIA Portal to your needs.

See also

Displaying an overview of all keyboard shortcuts (Page 270)

6.2.3.2 Displaying an overview of all keyboard shortcuts

You can display an overview of all keyboard shortcuts.

Procedure

To display an overview of all available keyboard shortcuts, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The settings of the TIA Portal are displayed.
2. Open the "Keyboard shortcuts" entry in the area navigation.
   You can see an overview of all keyboard shortcuts, which are valid for the currently installed products.

6.2.3.3 Basic functions of the TIA Portal

Below, you will learn how you can use the basic functions of the TIA Portal using only your keyboard.

Using the basic functions of the TIA Portal with the keyboard

The following table shows how you can access basic functions of the TIA Portal with keyboard shortcuts:

Function	Keyboard shortcut	Menu command
Change between the project view and the portal view	<Alt+F7>	
Change between the project view and the library view	<Alt+F5>	
Open the online and diagnostics view	<Ctrl+D>	
Start Runtime	<Ctrl+Shift+Z>	
Display cross-references	<Shift+Alt+F11>	
Cross-reference information	<Shift+F11>	

Function	Keyboard shortcut	Menu command
Open the Help system If you need help on the TIA Portal, press <F1>.	<F1>	Help > Show help
Cancel the current action	<Esc>	
Find	<Ctrl+F>	
Replace an object You can replace found objects when searching in the editor.	<Ctrl+H>	
Find next If you have started a search in the editor, you can jump to the next hit with <F3>.	<F3>	
Print object	<Ctrl+P>	Project > Print
Run default action of the object	<Enter>	
Action	<F5>	
Scroll horizontally to the right	<Ctrl+Arrow right>	
Scroll horizontally to the left	<Ctrl+Arrow left>	
Open block/PLC data type	<F7>	

Operating menus

The following table shows how you can navigate through menus using the keyboard:

Function	Keyboard shortcut
Start keyboard shortcuts in the menu You can access the menu using the <Alt> key and then continue to navigate with the arrow keys to scroll through the menu. Confirm your selection of the menu command with <Enter>.	<Alt>
Go directly to a specific menu You can go directly to an individual menu command by holding down the <Alt> key. There is an underlined letter for each menu command. Press the <Alt> key along with the underlined letter.	<Alt+underlined letter in respective menu>
Open shortcut menu of an object With the shortcut menu key (on Microsoft Windows compatible keyboards), you can open the shortcut menu of the selected object. Alternatively, you can use <Shift+F10> if you are not using a Microsoft Windows compatible keyboard. You can use the arrow keys to scroll through the shortcut menu and select a menu command with <Enter>.	<Shortcut menu key> Alternative: <Shift+F10>

Operating expandable elements

The following table shows how you can operate expandable elements using the keyboard:

Function	Keyboard shortcut
Expand a folder in a tree With <Arrow right>, for example, you expand a folder in the project tree.	<Arrow right>
Close a folder in a tree With <Arrow left>, for example, you collapse a folder in the project tree.	<Arrow left>
Open a drop-down list You can open drop-down lists with <F4> and then navigate with the arrow keys to scroll through the drop-down list. Finally, press the <Enter> key to confirm your selection.	<F4>

Function	Keyboard shortcut
Open autocompletion	<Ctrl+Space> <Ctrl+I>
Show object selection	<Ctrl+J>

6.2.3.4 Using project-related functions

Editing a project

Function	Keyboard shortcuts	Menu command
Open a project	<Ctrl+O>	Project > Open
Close a project	<Ctrl+W>	Project > Close
Save a project	<Ctrl+S>	Project > Save
Save a project under a different name	<Ctrl+Shift+S>	Project > Save as
Delete a project	<Ctrl+E>	Project > Delete project
Print project	<Ctrl+P>	Project > Print
Undo last action	<Ctrl+Z>	Edit > Undo
Redo last action	<Ctrl+Y>	Edit > Redo

Calling up the help function

Function	Keyboard shortcuts	Menu command
Calling up the help function	<F1> or <Shift+F1>	Help > Show help

6.2.3.5 Arranging windows

Below, you will learn how to open and close individual windows of the TIA Portal and work with saved window layouts using the keyboard.

Opening and closing windows

The following table shows how you can open and close windows with keyboard shortcuts:

Function	Keyboard shortcut	Menu command
Open/close project tree	<Ctrl+1>	View > Project tree
Opening/closing the detailed view	<Ctrl+4>	View > Details view
Opening/closing the overview	<Ctrl+2>	View > Overview
Opening/closing a task card	<Ctrl+3>	View > Task card
Open libraries	<Ctrl+Shift+L>	
Open hardware catalog If you are in the device or network view, the hardware catalog opens.	<Ctrl+Shift+C>	
Open/close inspector window	<Ctrl+5>	View > Inspector window

Function	Keyboard shortcut	Menu command
Open the "Properties" tab in the inspector window	<Ctrl+6>	
Open the "Info" tab in the inspector window	<Ctrl+7>	
Open the "Diagnostics" tab in the Inspector window	<Ctrl+8>	
Display or hide reference projects	<Ctrl+9>	
Display the on-screen keyboard You can display a keyboard on the screen, for example, to operate with a touch screen.	<Ctrl+Shift+K>	
Close all editors	<Ctrl+Shift+F4>	Window > Close all

Using saved window layouts

You can save individual window arrangements and restore them at a later point in time. The following table shows how you to access saved window layouts with keyboard shortcuts:

Function	Keyboard shortcut	Menu command
Restore active window layout If you use a saved window layout and have made changes to the program interface in the meantime, you can restore the original state of the active window layout with <Alt+Shift+0>.	<Shift+Alt+0>	Window > Restore window layout
Load window layout You can use <Alt+Shift+[number of the window layout]> to activate the first of the five saved window layouts.	<Shift+Alt+[Number of the window layout]>	Window > Window layout 1 to 5

6.2.3.6 Navigating through the program interface

The TIA Portal is divided into various user interface areas, for example, individual windows, toolbars, and editors. If you want to work with the keyboard within an interface area, you first have to place the focus on it. Below, you will learn how to place the focus on individual interface areas using the keyboard. You will also learn how to move within an interface area in the TIA Portal using the keyboard.

Switching between interface areas and editors

The following table shows how to move between individual interface areas of the TIA Portal:

Function	Keyboard shortcut
Move clockwise between the interface areas You can use the <F6> key to move clockwise between the individual interface areas of the TIA Portal. The interface area currently in focus is highlighted with a blue title bar. If you are in the project tree, for example, and press the <F6> key, you jump to the currently open editor. If you press <F6> again, the task cards are in focus. If you press <Shift+F6>, on the other hand, you move counterclockwise between the work areas.	<F6>
Move counterclockwise between the interface areas With <Shift+F6> you move counter-clockwise between the interface areas of the TIA Portal.	<Shift+F6>

Function	Keyboard shortcut
Go to the next open editor With <Ctrl+Alt+Arrow right> you go to the next open editor. You can see the open editors in the editor toolbar.	<Ctrl+Alt+Arrow right> Alternative: <Ctrl+F6>
Go to the previous open editor With <Ctrl+Alt+Arrow left>, you go to the last opened editor.	<Ctrl+Alt+Arrow left> Alternative: <Ctrl+Shift+F6>
Go to the next higher section of the interface area With <Shift+Esc>, you move to the next higher section of the program interface. If, for example, you have selected a device in the project tree and you press <Shift+Esc>, the entire project navigation is put into focus.	<Shift+Esc> Alternative: <Alt+Arrow up>
Go to the next lower section of the interface area With <Enter>, you place the focus on the next lower section of the program interface. For example, if you have just opened the properties of a device in the Inspector window in order to assign parameters to the device, press <Enter> to go one level deeper in the program interface. You can then navigate to the desired parameter using the Tab key.	<Enter> Alternative: <Alt+Arrow down>

Navigation within interface areas and editors

The following table shows how you can navigate within a user interface area using the keyboard:

Function	Keyboard shortcut
Jump to the next element within an interface area You can use the Tab key to jump from one element to the next within a work area. If, for example, you have opened the properties of a device and want to jump from one field to the next, press the Tab key. Any changes you have made to the current text box are applied in this case.	<Tab>
Jump to the previous element within an interface area With <Shift+Tab>, you can jump to the previous element in a work area, for example, a previous text box. Any changes you have made to the current text box are applied in this case.	<Shift+Tab>
Jump to the higher-level element within an interface area With <Shift+Home> you can jump to the higher-level element of a work area, e.g., to the higher-level folder in the project tree.	<Shift+Home>
Move to the next tab within an interface area If an interface area is divided into separate tabs, you can move between the tabs with the shortcut keys <Ctrl+Tab>. If, for example, you are in the "Properties" tab of the Inspector window and you want to jump to the "Info" tab, press the shortcut keys <Ctrl+Tab>.	<Ctrl+Tab>
Go to the previous tab With <Ctrl+Shift+Tab>, you move to the most recently open tab within an interface area.	<Ctrl+Shift+Tab>

Function	Keyboard shortcut
Jumping to the toolbar of an editor You can use the <Alt+F10> key to jump to the toolbar of an editor. For example, if you have opened the print preview and want to switch to the next page of the printout in the toolbar, press <Alt+F10>. Then, use the arrow keys to navigate to the appropriate icon in the toolbar and confirm the selection with <Enter>.	<Alt+F10>
Use arrows on the divider to display or hide user interface components The table in the work area can be minimized and maximized. First, navigate to the work area and use the Tab key to place the focus on one of the little arrows on the separator line above the table. The arrows have the focus as soon as they are highlighted in blue. Then, press the space bar to minimize or maximize the table.	<Space>

6.2.3.7 Customizing editors

Below, you will learn how to arrange editors using the keyboard. You will also learn how to select the display size and the area within a graphical editor.

Arranging and customizing editors

The following table shows how to arrange open editors above and below each other or side-by-side, and how to close an open editor:

Function	Keyboard shortcut	Menu command
Close active editor	<Ctrl+F4>	
Split editor space vertically If, for example, you have opened the overview window and the network view and want to display them side-by-side, press the <F12> key.	<F12>	Window > Split editor space vertically
Split editor space horizontally You can display two open editors in the work area above and below each other.	<Ctrl+F12>	Window > Split editor space horizontally
Remove window split If you have displayed two editors in the work area horizontally or vertically in split mode, you can remove the split with <Alt+Shift +F12>.	<Alt+Shift+F12>	Window > Unsplit editor space

Customizing the display in an editor

The following table shows how you how to zoom in and out in graphical editors and how to move the area selection in an editor:

Function	Keyboard shortcut
Zoom in step-by-step in an editor With <Ctrl> and the <Plus> key on the numeric keypad of the keyboard, you can zoom in on the display of the editor.	<Ctrl+Plus> Alternative: <Ctrl+Scroll up>
Zoom out step-by-step in an editor Use the <Ctrl> and the <Minus> key on the numeric keypad of the keyboard to zoom out of the display of the editor.	<Ctrl+Minus> Alternative: <Ctrl+Scroll down>

Function	Keyboard shortcut
Set view in the editor to 100% Press <CTRL+0> to enlarge or reduce the current view in a graphical editor to 100%.	<Ctrl+0>
Move the area selection of the editor If you hold down the spacebar, you can move the displayed section of an editor using the mouse.	<Space>

6.2.3.8 Editing objects

Selecting objects

The following table shows how to select individual objects, for example, devices in the project tree:

Function	Keyboard shortcut	Menu command
Select an object located at the left, right, above or below	<Arrow keys>	
Jump to the first object within the current interface area The first object in the interface area currently in focus is selected. In the project tree, this would be the project node at the top, for example.	<Home>	
Jump to the last object within the current interface area The last object in the interface area currently in focus is selected, for example, the last item in the project tree.	<End>	
Select all objects in an area All objects in the work area currently in focus are selected.	<Ctrl+A>	Edit > Select all
Select multiple objects If you want to select several objects that are not located directly next to each other, you first have to move the focus (gray outline of an object) to the next desired object using <Ctrl+Arrow keys>. The current selection is maintained. Then, press the space bar to select the new focused object as well. Repeat this process until all desired objects are selected.	<Ctrl+Arrow keys> + <Space>	

Editing objects

The following table provides an overview of all the keyboard shortcuts required for editing objects:

Function	Keyboard shortcut	Menu command
Insert new object A new object is inserted depending on your current context. If, for example, you are in the device view, the "Add Device" dialog opens for creating a new device.	<Ctrl+N>	
Open object	<Enter>	
Rename an object	<F2>	Edit > Rename
Copy an object	<Ctrl+C> Alternative: <Ctrl+Ins>	Edit > Copy

Function	Keyboard shortcut	Menu command
Cut an object	<Ctrl+X> Alternative: <Shift+Delete>	Edit > Cut
Paste an object	<Ctrl+V> Alternative: <Shift+Insert>	Edit > Paste
Delete an object	<Del>	Edit > Delete
Compile an object	<Ctrl+B>	Edit > Compile
Open properties of an object Many objects in the TIA Portal have editable properties. Press the shortcut keys <Alt+Enter> to display the properties of an object.	<Alt+Enter>	-

6.2.3.9 Text editing

Below, you will learn how to operate text editing functions using only the keyboard.

Editing text

The following table shows the basic editing functions for text:

Function	Keyboard shortcuts
Switch to insert or overwrite mode	<Insert>
Exit edit mode	<Esc>
Delete	<Del>
Delete characters	<Backspace>
Confirm entry in a text box and leave the text box	<Return>
Line break in a multiline text box In a multiline text box, hold down the <Shift> button to create a line break.	<Shift+Return>
Reset input in a text box If you are in an text box and press <Esc>, you exit the box and the changes are discarded.	<Esc>

Navigating within a text area

The following table shows how to navigate in a text area with the keyboard:

Function	Keyboard shortcuts
Jump to start of line	<Home>
Jump to end of line	<End>
Jump to start of text	<Ctrl+Home>
Jump to end of text	<Ctrl+End>
Jump to the previous page	<PgUp>
Jump to the next page	<PgDn>
Confirm entry in a text box and leave the text box	<Return>

Function	Keyboard shortcuts
Line break in a multiline text box	<Shift+Return>
Reset input in a text box If you are in an text box and press <Esc>, you exit the box and the changes are discarded.	<Esc>

Selecting text

The following table shows how to select text with the keyboard:

Function	Keyboard shortcuts
Expand selection to the word at the left or right The text or current text selection is marked up to the end of word. If you are at the beginning or end of a word, the previous or next word is selected.	<Ctrl+Shift+Arrow left or Arrow right>
Expand selection to beginning of line	<Shift+Home>
Expand selection to end of line	<Shift+End>
Expand selection to beginning of text The text is selected up to the beginning or the end.	<Ctrl+Shift+Home>
Expand selection to end of text The text is selected up to the beginning or the end.	<Ctrl+Shift+End>

6.2.3.10 Editing tables

Below, you will learn how to navigate with the keyboard in tables, edit individual fields, and select parts of tables.

General keyboard operation in tables

The following table shows how you can edit tables using only the keyboard:

Function	Keyboard shortcut
Place a cell in edit mode	<F2> or <Return>
Confirm entry and exit edit mode	<Enter>
Cancel editing and discard changes	<Esc>
Open drop-down list in a cell Open the drop-down list with <F4>. Use the arrow keys to select the desired entry and then confirm the selection with <Return>.	<F4>
Close drop-down list in a cell and discard changes	<Esc>

Navigate in tables

The following table shows how you can navigate within a table using the keyboard:

Function	Keyboard shortcut
Go to line	<Ctrl+G>
Go to the next cell	<Arrow keys>

Function	Keyboard shortcut
Go to the next editable cell on the right	<Tab>
Go to the next editable cell on the left	<Shift+Tab>
Move a screen upwards	<PgUp>
Move a screen downwards	<PgDn>
Go to the first cell in the row	<Home>
Go to the last cell in the row	<End>
Go to the first cell in the table	<Ctrl+Home>
Go to the last cell in the table	<Ctrl+End>
Go to the top cell in the column	<Ctrl+Arrow up>
Go to the bottom cell in the column	<Ctrl+Arrow down>
Insert table row below	<Alt+Ins>
Insert line break in an entry field/insert table row above	<Ctrl+Return>

Selecting areas in tables

The following table shows how you can select areas within a table using the keyboard:

Function	Keyboard shortcut
Select column	<Ctrl+Space>
Select line	<Shift+Space>
Select all cells	<Ctrl+A>
Expand selection by one cell	<Shift+arrow keys>
Extend selection up one page	<Shift+PgUp>
Extend selection down one page	<Shift+PgDn>
Expand selection up to the first row	<Ctrl+Shift+Arrow up>
Expand selection down to the last row	<Ctrl+Shift+Arrow down>
Expand selection to the first cell in the row	<Ctrl+Shift+Arrow left>
Expand selection to the last cell in the row	<Ctrl+Shift+Arrow right>

6.2.3.11 Using online functions

Controlling online functions with the keyboard

The following table provides an overview of the shortcut keys that you can use for the online functions of the TIA Portal:

Function	Keyboard shortcut	Menu command
Establish an online connection	<Ctrl+K>	Online > Go online
Go offline	<Ctrl+M>	Online > Go offline
Download project data to the device	<Ctrl+L>	Online > Download to device

Function	Keyboard shortcut	Menu command
Show accessible devices This opens a dialog showing all devices that are connected to the PG/PC interface of the PG/PC.	<Ctrl+U>	Online > Show accessible devices
Start CPU The CPU is set to "RUN" mode. The CPU must be online for this.	<Ctrl+Shift+E>	Online > Start CPU
Stop CPU The CPU is set to "STOP" mode. The CPU must be online for this.	<Ctrl+Shift+Q>	Online > Stop CPU
Start simulation The hardware and software of the project can be tested in a simulated online environment, without the modules actually being online.	<Ctrl+Shift+X>	Online > Simulation > Start

6.2.3.12 Using the on-screen keyboard

Introduction

When working with the TIA Portal, you also have the Microsoft on-screen keyboard available.

Displaying the on-screen keyboard

To display the on-screen keyboard, follow these steps:

1. In the "View" menu, select the "Screen keyboard" command.

Exiting the on-screen keyboard

To exit the on-screen keyboard, follow these steps:

1. In the "File" menu of the on-screen keyboard, select the "Exit" command.

6.2.4 Special features specific to the operating system

6.2.4.1 Influence of user rights

Restrictions when user rights are limited

The software provides several functions that require direct access to the hardware of the programming device / PC and therefore also to the installed operating system. To make full use of the range of functions, the software must cooperate closely with the operating system. To ensure problem-free interaction, you should therefore be logged on to the operating system with adequate user rights.

In particular, you may not be able to use functions requiring an online connection or those that change the settings of interface cards if you work with limited user rights.

Recognizing restricted functions

You can recognize functions requiring special rights as follows:

- A shield icon is displayed beside the function.
  
  The function can be used but is regulated by the user account control.
- A box is grayed out and cannot be accessed.
  You require administrator privileges to access the box. In some operating system environments, you can obtain administrator privileges by entering an administrator password.

Note

A box being grayed out does not necessarily mean a lack of rights. You should also check the additional information in the tooltip cascades to find out the conditions for editing the box.

6.2.4.2 Expanding user rights

Counteracting restrictions due to user rights

Certain functions may not be available if you are not logged on to the operating system with adequate rights. You can counteract these restrictions in the following ways:

- Enabling of extended rights using Windows user account control
- Logging on to the operating system with administrator privileges
- Using temporary administrator rights

Enabling extended rights using Windows user account control

To be able to use a function indicated by the shield icon of the Windows user account control, follow these steps:

1. Click on the box or button with the shield icon.
   The security prompt of the Windows user account control opens.
2. Follow the instructions of the Windows user account control and, when prompted enter an administrator password, if possible.

The function can now be used once without restrictions.

Logging on to the operating system with administrator privileges

To be able to use a function that is disabled due to lack of user rights, follow these steps:

1. Close the software.
2. Log off from the operating system.
3. Log on to the operating system with administrator privileges.
4. Restart the software.

Using temporary administrator rights

To obtain administrator privileges temporarily, follow these steps:

1. Click the "Change settings" button. You will find this button in dialogs that allow the temporary assignment of administrator privileges.
   An operating system dialog box for entering an administrator password opens.
2. Enter an administrator password.

The settings can be temporarily changed. When you call the dialog again, the procedure must be repeated.

Note

This function is not supported by all operating systems. If no "Change settings" button is present or the button is grayed out, you will need to log on to the operating system with administrator privileges instead.

6.3 Help on the information system

6.3.1 General remarks on the information system

Quick answers to your questions

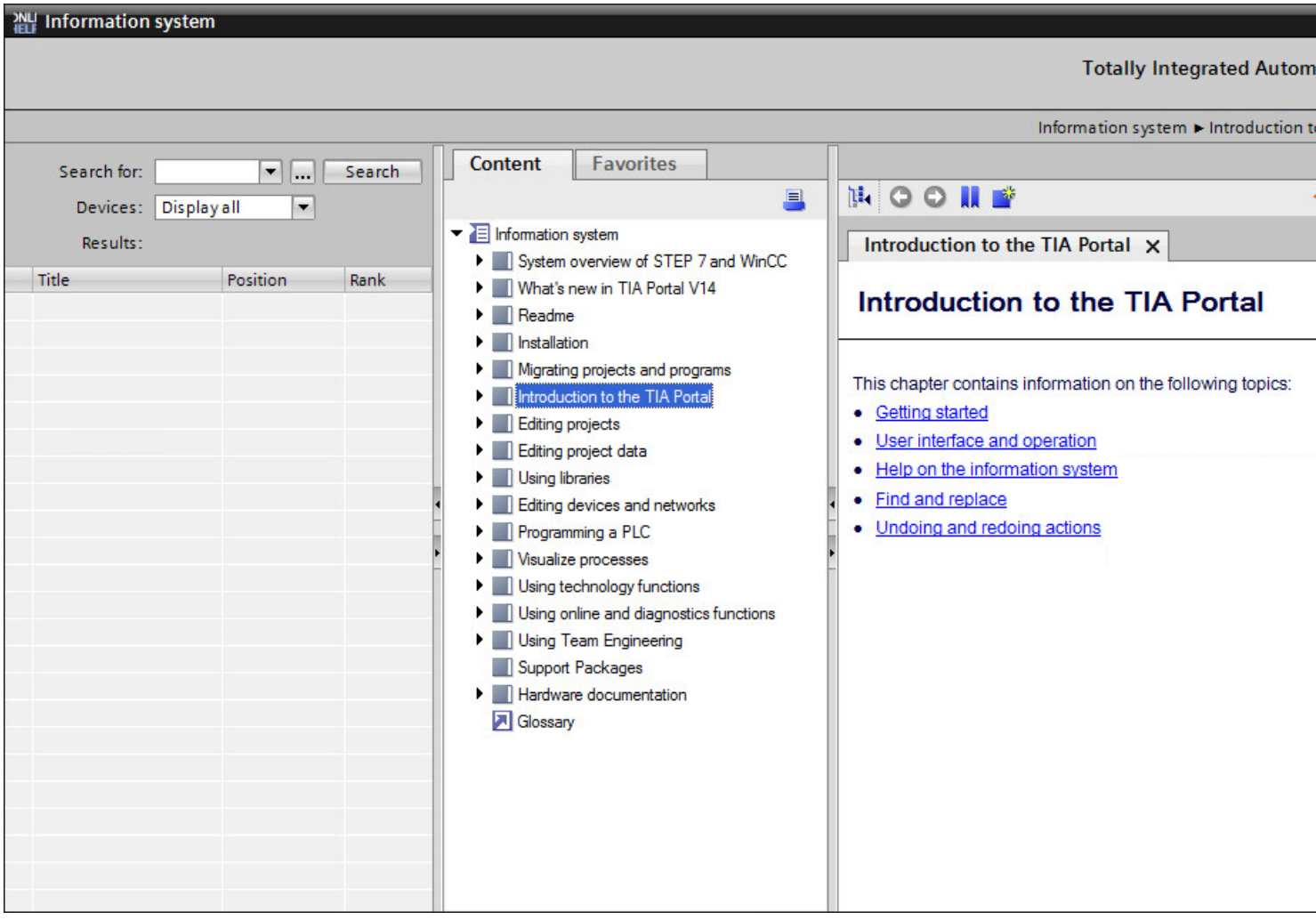
The information system of the TIA Portal helps you solve your problems and offers the required help topics at each step of the configuration.

While working with the program, you receive the following support:

- Information system with all the background information, step-by-step instructions and examples that you need when working with the TIA Portal.
- Tooltips for information on elements of the user interface, for example text boxes, buttons and icons. Some of the tooltips are supplemented by cascades containing more precise information.
- Help on the current context, for example on menu commands when you press the <F1> key.
- Help on messages
- Roll-outs for correct inputs in dialog boxes
- Banner on the software interface

Information system

The information system opens in a separate window. The following figure shows the information system for the TIA Portal:



The information system is divided into the following areas:

- Search area
  In the search area, you can perform a full text search across all help topics.
- Navigation area
  You can find the table of contents and favorites in the navigation area.
- Content area
  The help pages appear in the content area. You can open multiple tabs to view various help pages at the same time.

The arrows on the window dividers allow you to display and hide the individual sections. You can open both the search area and the navigation area to increase the contents area as needed.

Buttons in the contents area

The following buttons are available in the contents area of the information system:

Button	Function
	Synchronize Shows the position of the open help topic in the table of contents.
	Forward / backward Navigates through the history of the help topics previously opened in this tab.
	Favorites Adds the current help topic to the favorites.
	New tab Opens a new, empty tab.
	Home page Displays the start page of the information system.
	Printing Prints the current help topic.
	Help Displays help on the information system.

Identification of the topics according to the information type

The information system provides you with a wide range of information. The help topics are identified by different symbols depending on the type of information they contain. Symbols tell you at a glance what kind of information you have just opened.

Symbol	Information type	Explanation
	Operating instructions	Describes the steps to follow in order to carry out a particular task.
	Example	Contains a practical application example that explains how to solve an automation task.
	Factual information	Provides background information on the functions of the TIA Portal and provides basic knowledge.
	Reference	Provides detailed reference information about instructions and objects for reference.

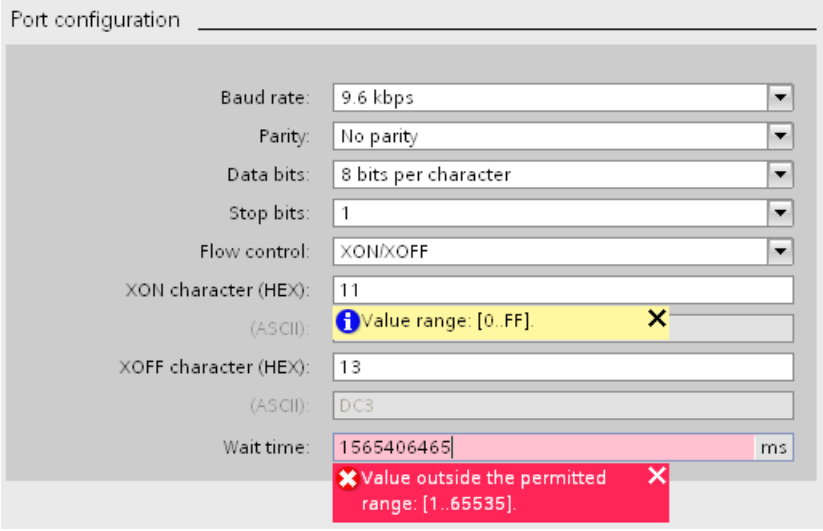
Identification of the topics according to the target system

Depending on the scope of the installation, the help system may contain sections or pages that apply to specific devices only. To be able to recognize such sections or pages at first glance, you will find a note in brackets in the table of contents. The search results in the full text search are marked in the same way if they only apply to a specific device. You will find an icon in the title bar of such pages. Move the cursor over the icon to obtain information about the valid target system.

Roll-out

Certain text boxes offer information that rolls out and helps you to enter valid parameters and values. The roll-out informs you about permitted value ranges and data types of the text boxes.

The following figure shows a roll-out (yellow) and a roll-out error message (red), which indicates an invalid value:



The roll-out is closed as soon as you exit the window or click the x in the upper right corner.

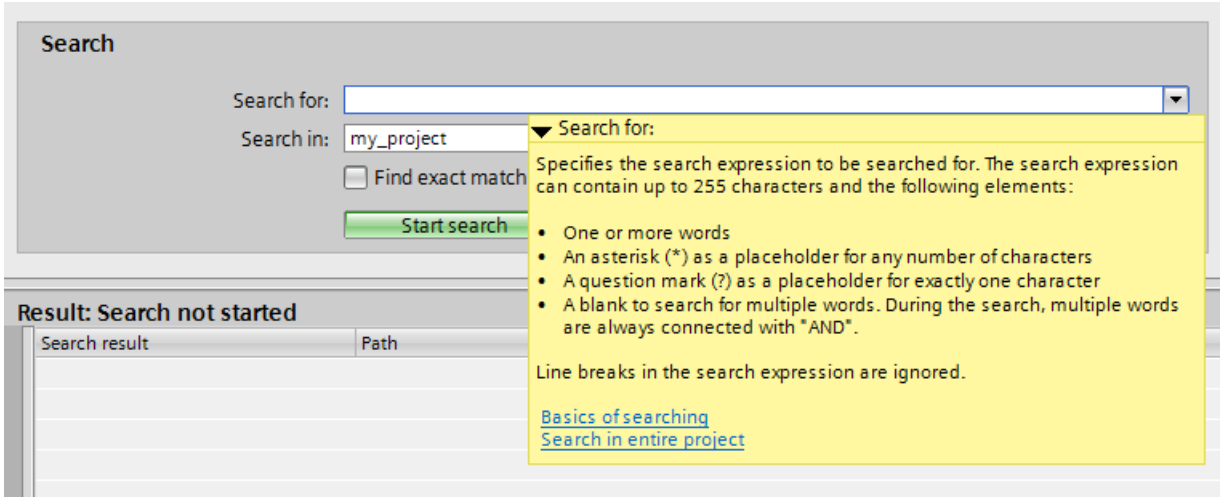
Tooltip

User interface elements offer brief information in the form of a tooltip.

Tooltips, which have an arrow icon on the left, contain additional information in tooltip cascades. If you position the mouse pointer briefly over the tooltip or click the arrow icon, this information is displayed. The automatic display of tooltip cascades can be disabled.

If additional information is available, a link to the corresponding help topic appears in the cascade. If you click on the link, the corresponding topic opens in the information system.

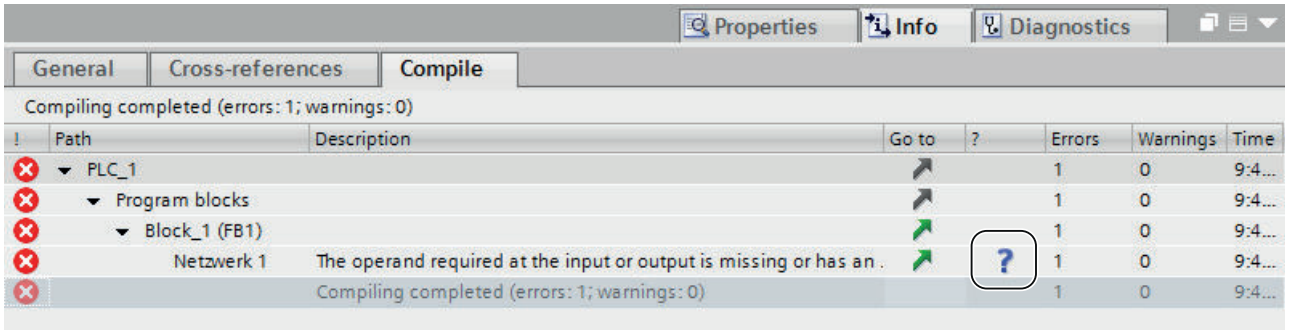
The following figure shows a tooltip with opened cascade:



Help on messages

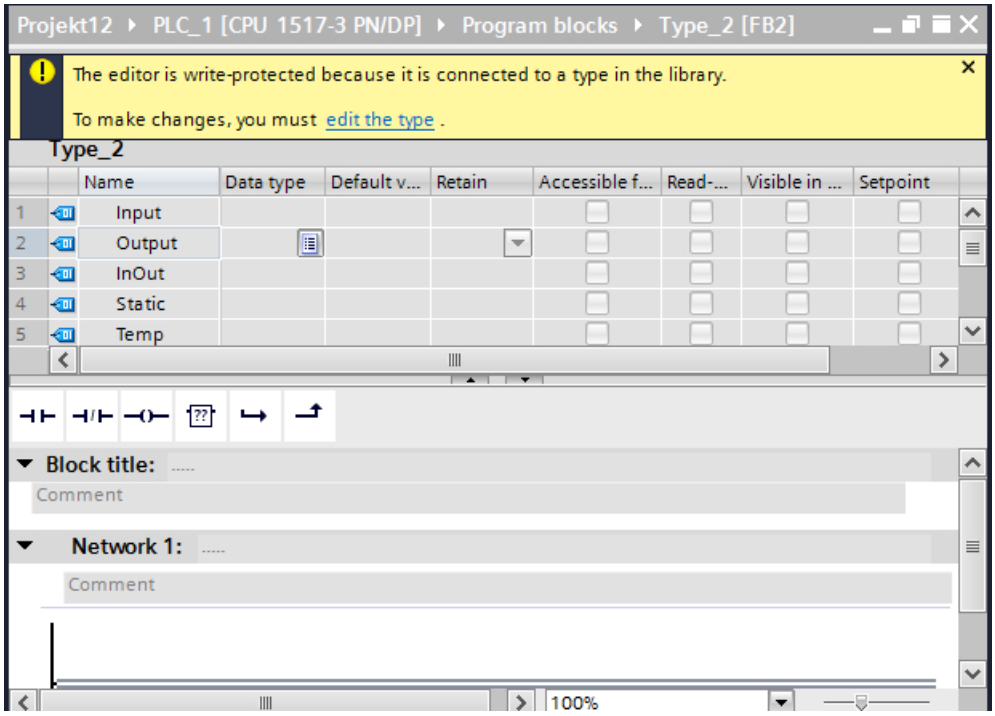
Numerous actions are accompanied by messages in the Inspector window of the TIA Portal. The messages give information about whether or not an action was successful. In addition, you see which changes have been made in the project. Further help is available for some messages. If further help is available for an message, you access the help by clicking the question mark symbol.

The following figure shows the "Info" tab in the Inspector window with several messages and a question mark symbol:



Banner on the software interface

In some editors, you can obtain additional support with banners at the top of the editor. The following figure shows a banner for an open library type:



To hide the banner, click on the vertical blue bar at the left edge of the banner or on the x in the upper right corner. To expand a collapsed banner, click on the information symbol at the top left of the opened editor. In the settings of the TIA Portal, you can specify that banner is to be hidden by default.

If multiple banners are available in an editor, the most relevant banner is displayed first. To move to the next banner, scroll to the next banner at the bottom right corner.

See also

Configuring the display of tooltips and tooltip cascades (Page 299)

Using user-defined documentation (Page 426)

6.3.2 Opening the information system

Opening the information system with the menu

To access the information system on the home page, follow these steps:

1. Select "Display help" command from the "Help" menu.

The start page of the information system opens.

Opening the information system with <F1>

To access the information system and help for the current context, proceed as follows:

1. Select an object for which you want to view the help, for example, a menu command or a program element.
2. Press <F1>.

The information system opens. If information on the current context is available, the appropriate topic appears. If no information on the current context is available, the home page of the information system is displayed.

Opening the information system via tooltip cascades

To open the information system from a tooltip cascade, proceed as follows:

1. Move the mouse pointer over an object with a tooltip cascade.
   The tooltip cascade opens. If additional information is available, the tooltip cascade contains links to the corresponding help topics.
2. Click on a link.

The information system opens and the additional help topic appears.

Opening help for error messages

Numerous actions are accompanied by messages in the Inspector window of the TIA Portal. If help on such a message is available, a blue question mark appears behind the message.

To view the help for a message, follow these steps:

1. Display the Inspector window.
2. Click on the blue question mark behind a message.

The information system opens and the help error message appears. It contains an exact description of cause of the error and how to remedy it.

See also

Calling user-defined documentation (Page 435)

Using the information system (Page 289)

6.3.3 Using the information system

Opening a help topic in the current tab

To open a help topic in the current tab, follow these steps:

1. Click on the help topic in the table of contents.

Or:

1. Double-click on the help topic in the list of search results.

Opening a help topic in a new tab

To open a help topic in a new tab, follow these steps:

1. Select the help topic in the table of contents or in the list of search results.
2. Select the "Open in new tab" command from the shortcut menu.

Or:

1. Click on the "New tab" button in the contents area of the information system.
   A new empty tab opens.
2. Click on the help topic in the table of contents or double-click on the help topic in the list of search results.

Closing a tab

To close a tab, follow these steps:

1. Click on the cross at the right in the tab.
   The current tab is closed.

Or:

1. Click on the tab.
2. Select the "Close other tabs" command from the shortcut menu.
   All tags except the current one will be closed.

Showing the history

The history of the previously opened help topics is stored individually for each tab. To remove a help topic from the history, follow these steps:

1. In the contents area of the information system, click the "Forward" or "Backward" buttons to scroll chronologically through the history.

Displaying the position of a help topic in the table of contents

To display the position of the open help topic in table of contents, follow these steps:

1. Position the mouse pointer on the help topic.
2. Click "Synchronize" in the contents area of the information system.

Displaying help for the information system

To display the help on the information system, follow these steps:

1. Click on the "Help" button in the contents area of the information system.

6.3.4 Searching the information system

6.3.4.1 Basics of searching

Introduction

You have the option to search the information system for terms. You are provided a search area for this.

Search options

You can search for the following text:

- Search for a word:
  `<Search term>`
  This finds all help pages that contain the search term exactly or in a slightly changed form.
- Search for several words:
  `<Search_term_1 Search_term_2 ... Search_term_n>`
  This finds all help pages that contain all search terms exactly or in a slightly changed form. Neither the order nor the distance between the search terms plays a role here.
- Search for phrases
  `<"Search_term_1 Search_term_2 ... Search_term_n">`
  This finds all help pages that contain all search terms in the exact order specified. To find a phrase, place your search expression in quotation marks.

You can further specify your search with additional criteria:

- Boolean search with "AND":
  `<Search_term AND Search_term>`
  `<Search_term + Search_term>`
  You can link your search terms using the Boolean "AND" operator. This finds all help pages that contain all search terms exactly or in a slightly changed form.
  Alternatively, you can enter multiple search terms even without the operators "AND" or "+". This too finds all help pages that contain all search terms.
- Boolean search with "OR":
  `<Search_term OR Search_term>`
  `<Search_term - Search_term>`
  You can link your search terms using the Boolean "OR" operator. This finds all help pages that contain one of the search terms exactly or in a slightly changed form.
- Negate search with "NOT"
  `<Search_term_1 NOT Search_term_2>`
  `<Search_term_1 ! Searchterm_2>`
  You can negate a search term with "NOT". You must enter at least one additional search term for this purpose, however. This finds all help pages in which "Search_term_1" occurs. Pages in which "Search_term_2" occurs are not included in the hits, since this search term has been negated.

- Fuzzy search with "~"
  `<Search~>`
  You can place a tilde ("~") after the search term. This finds all help pages in which the search term appears with approximately this spelling.
- Distance search with "~<proximity>"
  `<"Search_term_1 Search_term_2" ~proximity>`
  The proximity search allows you to find help pages where two search terms are no further apart than the defined distance. Enter the proximity as a positive integer. Enter the two search terms in quotes. There should be no gap between the tilde ("~") and the space.

Search filter

You can limit the search to a specific device or a device series by specifying a device as a search filter:

- If you select a device as a filter criterion, the keyword is only searched for in the pages or sections that apply to the specified device.
- If select the "Show all" option from the list instead of specifying a device, the search term is sought throughout the entire information system.

Extended search

You can also use the extended search to search the information system. It offers the same search options as the default search. However, you also obtain direct information about the possible selections for search options.

See also

Searching help topics (Page 292)

Examples of searching in the information system (Page 293)

6.3.4.2 Searching help topics

To quickly find the required information, you can perform a full text search across all help topics.

Procedure

To search the information system for a specific terms, follow these steps:

1. Open the search area.
2. In the "Find what" box, enter the search term or select a recently searched phrase from the drop-down list.
3. Optional: To obtain information on the search options, click "...".
   The "Extended search" dialog opens. It shows an overview of all options.
4. Optional: In the "Devices" selection box, select a device as a search filter.

5. Click "Find".
   The search is performed and the results are listed:
   - The title of the help topic found is displayed in the "Title" column.
   - The "Position" column shows the section containing the help topic.
   - The relevance of the topic to the search term appears in the "Rank" column.

   You can change the sorting by clicking on the header of a column.
6. Double-click on a search result or select the "Open in new tab" command in the shortcut menu.

Result

The help topic opens in the contents area.

To display the location of the open help topic in the table of contents, click on the "Synchronize" button.

See also

Basics of searching (Page 291)

Examples of searching in the information system (Page 293)

6.3.4.3 Examples of searching in the information system

Examples of the search for one word

Searching for a word finds all help pages that contain the search term exactly or in a slightly changed form. Terms that contain special characters must be enclosed in quotes:

- Search for `<PLC data type>`
  This finds all pages containing the words "PLC data type", "PLC data types" or similar related terms.

Examples if searching for several words

Searching for multiple words finds all help pages that contain all search terms exactly or in a slightly changed form. Neither the order nor the distance between the search terms plays a role.

- Search for `<variant example>`
  This finds all pages containing the terms "variant", "VARIANT", "variant instruction" and "example", "examples" or similar related terms.
- Search for `<LED tag>`
  This finds all pages containing the terms "LED tag", "LED-tag", "LED status of the tag" or similar related terms.

Examples of searching with phrases

Searching with phrases finds all help pages that contain all search terms in the exact order specified. To find a phrase, place your search expression in quotation marks.

- Search for `<"examples for using recipes">`
  This finds all pages containing the terms "Examples for using recipes" or similar terms in the specified order.

Examples of the Boolean search with "AND"

A Boolean search with "AND" finds all help pages that contain all search terms exactly or in a slightly changed form. Neither the order nor the distance between the search terms plays a role.

- Search for `<user groups AND authorization>`
  This finds all pages containing the terms "user groups", "user group" and "authorization", "authorizations" or similar related terms.
- Search for `<examples AND "Use in the recipes">`
  In this example, the Boolean search using "AND" is combined with the phrase search: This finds all pages containing exactly the terms "use in the recipes" in the specified order and "examples" at any position.

Examples of the Boolean search with "OR"

A Boolean search with "OR" finds all help pages that contain one or more of the specified search terms exactly or in a slightly changed form.

- Search for `<block OR function>`
  This finds all pages containing the terms "block", "blocks" and/or "function", "functions" or similar related terms.
- Search for `<block OR function OR function block>`
  This finds all pages containing the terms "block", "blocks" and/or "function", "functions" and/or "function block", "function blocks" or similar related terms.

Examples of the Boolean search with "NOT"

A Boolean search with "NOT" finds all help pages that contain the specified search terms but not the negated search terms.

- Search for `<alarm CPU NOT GRAPH>`
  This finds all pages containing the terms "alarms", "alarm" and "CPU" but not the word "GRAPH".
- Search for `<alarm NOT GRAPH CPU>`
  This finds all pages containing the terms "alarms", "alarm" and "CPU" but not the word "GRAPH".
- Search for `<alarm NOT CPU NOT GRAPH>`
  All pages containing the term "message" are found but not the word "GRAPH" and not the word "CPU".

Examples of the fuzzy search with "~"

A fuzzy search finds all help pages in which the search term appears with approximately this spelling.

- Search for `<trends~>`
  This finds all pages containing the terms "trend", "trend bit", "trend value", "trend display" "trend recording" or similar related terms.

Examples of a proximity search with "~ <distance>"

A proximity search finds all help pages where two search terms are no further apart from each other than the defined distance.

- Search for `<trend measurement~10>`
  This finds all pages in which the terms "trend" and "measurement" are no more than 10 words apart.

Examples of searching with filters

When searching with a filter, keywords are searched for only in the pages or sections that are valid for the specified device.

- Search for `<example>` with search filter "S7-1500"
  This finds all pages containing the terms "example", "examples" or similar related terms and that apply to S7-1500.
- Search for `<SCL>` with search filter "S7-400"
  This finds all pages containing the term "SCL" and that apply to S7-400.

6.3.5 Using favorites

Using favorites

You can save individual help pages or sections as favorites. This saves you searching for the help topic a second time.

Favorites are retained even if you close the TIA Portal. They are still available when the information system opens the next time.

Saving favorites

To save a page or section as a favorite, follow these steps:

1. Open the page or the section that you want to save as a favorite.
2. Click on the "Add to favorites" button.
   Or:

1. Select a page or section in the table of contents.
2. Select the "Add to favorites" command in the shortcut menu.

The help topic or chapter is saved as a favorite and is available the next time you open the help system.

Calling favorites

To open a page from your favorites, follow these steps:

1. Open the "Favorites" tab.
2. Double-click on the help topic that you want to open.
   Or:

1. Open the "Favorites" tab.
2. Select the shortcut menu command "Open in new tab".

The help topic opens in the contents area of the current tab or in a new tab.

Go to position in the table of contents

To navigate from your favorites to the position of the subject in the table of contents, follow these steps:

1. Select the topic in the favorites where you want to go in the table of contents.
2. Select the shortcut menu command "Go".
   The menu command is enabled only when you select a topic that is included in the table of contents. Context-sensitive help topics, for example, help on syntax errors, are not included in the table of contents. The "Go to" is not executable for this.

The table of contents opens and expands at the desired position.

Deleting favorites

To delete an entry from the favorites, proceed as follows:

1. Open the "Favorites" tab.
2. Select the topic you want to remove from the list.
3. Select the "Remove" command from the shortcut menu.

6.3.6 Copying code examples into the program

Many help topics include code examples that illustrate instructions and functions for programming in the TIA Portal. You can copy code examples in the programming languages SCL and STL and use them directly in your program.

Example

You recognize an example that can be copied by the "Copy" symbol in the header of the program example:

SCL

```
"Tag_Result1" := ABS("Tag_Value");
"Tag_Result2" := ABS("Tag_Value1"*"Tag_Value2");
```

Procedure

To copy a code example, follow these steps:

1. Open the Help topic with the sample program you require.
2. Click on the "Copy" button in the header of the program example.
   The program code is copied to the clipboard.
3. Open the programming editor and insert the copied code. To do this, select the "Paste" command in the shortcut menu.
   The copied program code is inserted from the clipboard into the program code being edited.

6.3.7 Displaying more information in the Siemens Industry Online Support

The information system provides an introduction to the "TIA Portal Information Center" in the Siemens Industry Online Support. Here you obtain additional up-to-date information about the TIA Portal, for example, application examples, training opportunities, FAQs, hardware documentation and much more.

Note

Internet connection required

To access the information in Siemens Industry Online Support, you need an Internet connection.

Procedure

To open more information in Siemens Industry Online Support, follow these steps:

1. Open the start page of the information system. You have the following options here:
   - Select the top entry in the table of contents. The entry has the title "Information system".
   - Select the "Home page" button in the contents area.

   The start page of the information system opens. The "TIA Portal Information Center" displayed at the bottom of the home page.
2. Click on the screen of the "TIA Portal Information Center".
   The "TIA Portal Information Center" opens.

6.3.8 Printing help topics

You have the following options for printing the contents of the information system:

- You can print the selected sections or pages from the index together. You can select any content of the information system. It need not involve contents of contiguous sections. The selected content will be printed in a print-optimized layout with pagination.
- You can print a single open page.

Print sections or pages from the table of contents

To print the contents from the table of contents, follow these steps:

1. Open the table of contents of the information system.
2. Click on the "Print" button in the "Contents" tab.
   The "Print" dialog opens.
3. In the "Print" dialog, select the sections and pages that you want to print.
4. Click the "Print" button to print the selected information.
   The standard print dialog box opens.
5. Select the printer on which you want to print the information.
6. Click "Properties" if you want to make additional printer settings.
7. Confirm your entries with "OK".
   The information is printed on the selected printer.

Printing single opened page

To print a single page, follow these steps:

1. Open the page to be printed in the contents area of the information system.
2. Select the "Print" button in the contents area.
3. The standard print dialog box opens.
4. Select the printer on which you want to print the information.
5. Click "Properties" if you want to make additional printer settings.
6. Confirm your entries with "OK".
   The page is printed on the selected printer.

6.3.9 Configuring the display of tooltips and tooltip cascades

Configuration options for tooltips and tooltip cascade

You can customize the display of tooltips and tooltip cascades to suit your needs. You can make the following settings:

- Display or hide truncated texts
  Sometimes texts may be too long for a text field. The texts are then fully displayed in a tooltip when you hover your mouse over the text field. You can enable or disable this function.
- Enable or disable tooltips
  Tooltips provide more detailed information about an element of the user interface. You can also have tooltips displayed in a cascade. If you disable the tooltips, the cascade with context-sensitive help is also no longer displayed. However, you have the option of manually displaying the tooltip for the currently active interface element by pressing <F1>. The setting also affects rollouts with context-related syntax errors and input aids in STL.
- Enable or disable automatic opening of tooltip cascades
  By keeping the mouse pointer over a tooltip for a brief time, any available cascades are displayed automatically. You can enable or disable the automatic display of cascades. When automatic display is disabled, you must open the cascade manually if necessary. To do this, click on the arrow icon in the tooltip.

Procedure

To configure the display of tooltips and tooltip cascades, follow these steps:

1. Select the "Settings" command in the "Options" menu.
2. Select the "General" group in the area navigation.
3. Select or clear the individual check boxes in the "Tooltips" area to suit your needs. The "Open cascade automatically in tooltips" check box is only available if you have enabled the display of tooltips.

See also

General remarks on the information system (Page 283)

6.3.10 Assembling customized documentation

Customized documentation

In the Siemens Industry Online Support, in the "mySupport Documentation" category, you can assemble customized documentation that is tailored to your needs. All configurable manuals and operating instructions of the Siemens Industry Online Support are available to you for this purpose. You can select the parts that are of interest to you and combine them into in a library to form personal documentation. You can organize the documentation using folders in your personal library. The folders will later become the individual chapters of your custom documentation.

You can open "mySupport Documentation" here:

https://support.industry.siemens.com/My/ww/en/documentation (https://support.industry.siemens.com/My/ww/en/documentation)

Requirement

- The manuals or operating instructions used must be configurable. You can recognize configurable manuals by the suffix "configurable" in their name.
- To use all the functions, you have to register in the Siemens Industry Online Support and log on.

Documentation in different languages

You can change the language of the assembled documentation to German, French, Spanish, Italian and Chinese. This gives you the possibility, for example, to gather relevant information for a specific project and make it available to colleagues who speak other languages.

Export function in the documentation

You can perform an export at any part of your library in various formats (PDF, XML, RTF).

Help on creating documentation

You can find more help on creating and using custom documentation in the Siemens Industry Online Support:

https://support.industry.siemens.com/My/ww/en/documentation (https://support.industry.siemens.com/cs/helpcenter/en/index.htm)

6.3.11 Safety Guidelines

Safety guidelines

This Help manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

⚠ DANGER

indicates that death or severe personal injury will result if proper precautions are not taken.

⚠ WARNING

indicates that death or severe personal injury may result if proper precautions are not taken.

⚠ CAUTION

with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

NOTICE

without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

Note

indicates that an unintended result or situation can occur if the corresponding information is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by qualified personnel. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

Prescribed Usage

Note the following:

WARNING

This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens. Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

Trademarks

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

6.4 Find and replace

6.4.1 Basics of searching

Introduction

You can use the following search options within TIA Portal:

- Search entire project
- Searching and replacing in the editor
- Browsing the hardware catalog

In addition, individual products in TIA Portal offer more search options.

Search entire project

You can search the entire project for a specific text. The search editor helps you in the search, for example, by further limiting the search. All objects containing the search text are displayed clearly in a table. You can open each object in the search editor to view the relevant location.

Searching and replacing in the editor

You can search for texts in the editors. The search function finds all texts containing the search key in the currently opened editor. The results are selected in sequence in the opened editor.

You also have the following options:

- Narrowing down the search with additional options
- Replacing found texts

Which additional options are available and which types of text can be searched for depends on the installed products and the open editor.

Browsing the hardware catalog

You can search for a specific hardware component in the hardware catalog.

See also: Browsing the hardware catalog

See also

Searching and replacing in the editor (Page 315)

Search in entire project (Page 307)

6.4.2 Search entire project

6.4.2.1 Basics for search in the project

Introduction

You can search the open project for a specific text (search term). The search term can be one word, several words or parts of words. Only spaces are considered as word limits. You can use search terms that are up to 255 characters long. You can create any complex search terms using wildcards. You can also limit the search to specific areas, objects or object properties. Uppercase and lowercase letters are not taken into account in the search.

For the search, an index of the texts of the editors which can be searched in the project is built up automatically in the background. The search index is updated with each change. You cannot start the search before indexing is complete. You can generate the search index again at any time.

Note that the search index is saved with the project. This may increase the file size of the project. If you want to pass on the project, you can reduce the file size by archiving or minimizing the project. The search index is removed in this case. You can also completely disable the search in the project if necessary.

In the case of changes to STEP 7 objects, e.g. name changes of tags, PLC data types or blocks, you must update the point of use of the these object before any search in the project in order that the search can take these changes into account. To do this you can, for example, recompile your program.

The search only includes the texts in the set editing language. The search index is generated again when you change the editing language. This may take some time, depending on the number of texts in the project.

The search editor helps you with your search. In it, you can specify search options and start the search. After the search is completed, the search editor lists the objects which contain the search text. You can open the individual objects directly from the search editor.

Search options

The following options are available for the search:

- Limit search:
  - Search in: You can select the sections of the project that are to be searched.
  - Find exact match: Finds exactly the text you have entered. The search term is searched as individual word or as phrase. You cannot use wildcards in the search term when you use this option, for example, when you enter "*" or "?" the search is exactly for these characters.
  - Limit search: You can limit the search to specific object properties or object types.
- Use "*" as wildcard for any number of characters and "?" as wildcard for exactly one character within a word:
  When you use the "*" or "?" wildcards in a search term, you will receive whole words that match this search term as hits. When you enter "Motor*start", for example, you will receive hits such as "Motor starter" or "Motor1_Start" but not "Starting motors" or "Start motor1". An implicit wildcard search is always conducted. If you enter "Motor" as search term, for example, the search also finds the texts "Motor_ON", "Motor starter" and "123_Motor". This corresponds to the explicit wildcard search with the search term "*Motor*". You can deactivate the implicit wildcard search by enabling the option "Find exact match".

Replace texts in the project

You have the option to replace individual texts in the project. To do this, you can open the location immediately after a search by double-clicking on the hit list and editing the text. Once it is changed, the text is automatically indexed again. If you then want to open the next match, you receive a message that offers you the chance to refresh the hit list. In this way, you can replace a text in the entire project until the hit list no longer shows results.

See also

Basics of searching (Page 303)

Overview of the search editor (Page 305)

Search in entire project (Page 307)

Creating a new search index manually (Page 308)

Enabling and disabling "Search in project" (Page 309)

Examples for searching in the project (Page 310)

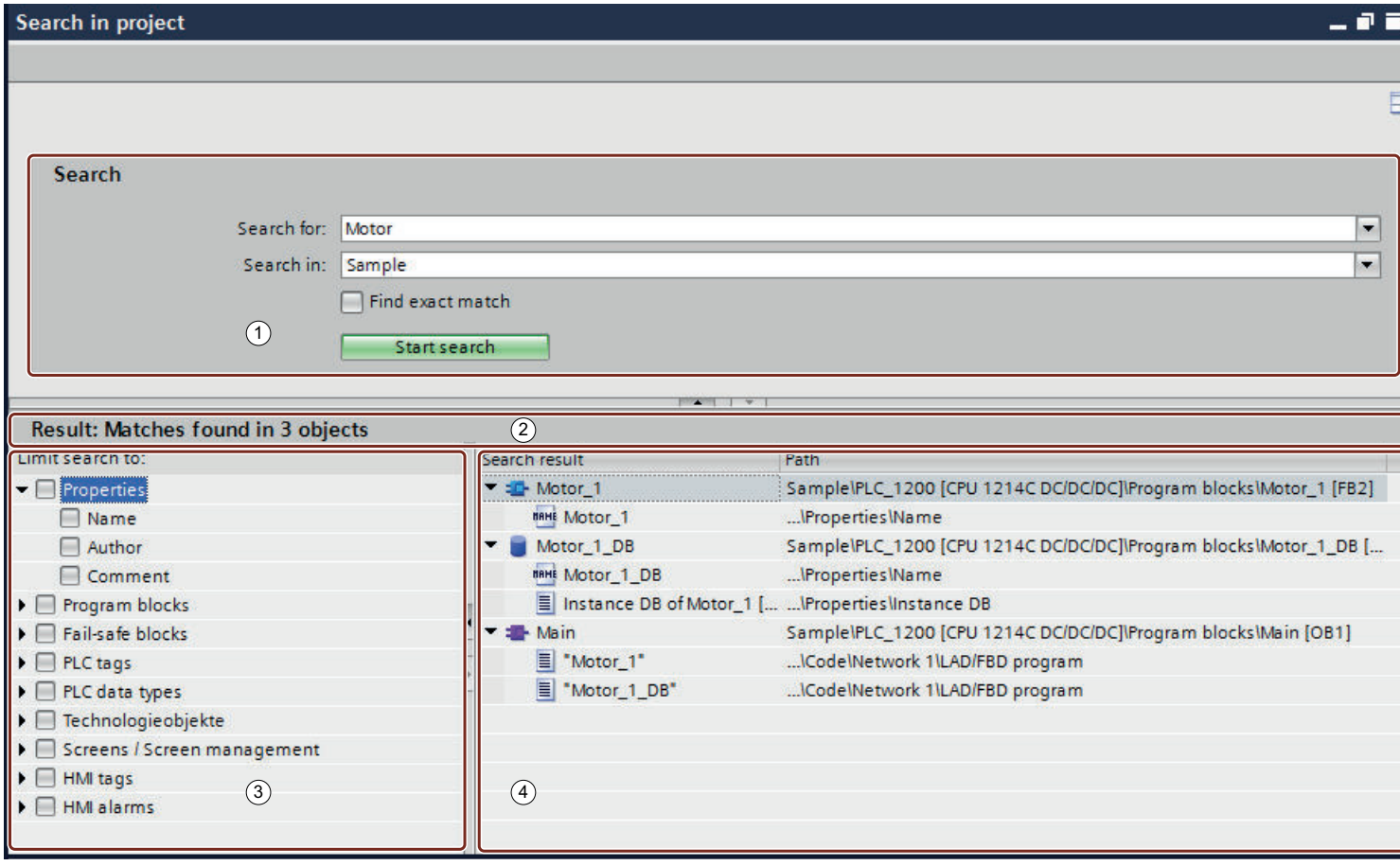
6.4.2.2 Overview of the search editor

Function of the search editor

The search editor searches the entire project for specific texts and represents the search results clearly in a table.

Layout of the search editor

The figure below shows the components of the search editor:



① Search mask

② Status bar

③ Search filter

④ Results list

Search mask

You enter the search term in the search mask and limit the search area. You also have the option to select "Find exact match" which means the search is limited exactly to the entered search term in the project and similar text passages are ignored. You start the search in the search mask.

Status bar

The status bar includes information about the search result.

Search filter

You can use the search filter to further limit the search results. If you have not activated any filters, the search includes all objects and object properties. If you activate a filter, only the search results which match the filter setting are displayed.

Results list

The results list shows all objects which include the search term. Double-click on a search result to open the corresponding editor and highlight the reference.

See also

Basics of searching (Page 303)

Basics for search in the project (Page 304)

Search in entire project (Page 307)

Creating a new search index manually (Page 308)

Enabling and disabling "Search in project" (Page 309)

Examples for searching in the project (Page 310)

6.4.2.3 Search in entire project

Requirement

A project is open.

Procedure

To search the current project for a specific text, follow these steps:

1. Enter the search term in the text box for the search in the TIA Portal toolbar.
2. Click "Search in project".
   The search editor opens and a search for the search term is conducted across the entire project. The objects containing the search text are listed in the overview.

Note

You cannot enter any search options with this search. The entire project is searched and all objects that can be searched are included in the search. If you want to further limit the search or use search options, you can start the search once again. To do so, follow the alternative instructions under "Or" as of step 2.

Or:

1. Open the search editor in one of two ways:
   - Select the "Search in project" command in the "Edit" menu.
   - Right-click an object in the project tree and select the "Search in project" command from the shortcut menu. The object is set as default search area in the "Search in" box.
   - Make sure that the focus is outside of an editor that supports an editor-specific search and press the shortcut <Ctrl+F>.
2. Enter the search term in the "Search" box, or select a term from the list of terms for which you have already searched.
3. To search exactly for the search term, select the check box "Find exact match". You cannot use wildcards in the search term when you use this option, for example, when you enter "*" or "?" the search is exactly for these characters.
4. To limit the search area, select the sections you want to include in the search under "Search in".
5. To limit the search to specific objects or object properties, select the objects or object properties you want to include in the search under "Limit search to".
6. Click "Start Search".
   The objects containing the search text are listed in the overview.
7. Double-click an object to open it.

See also

Basics of searching (Page 303)

Basics for search in the project (Page 304)

Overview of the search editor (Page 305)

Creating a new search index manually (Page 308)

Enabling and disabling "Search in project" (Page 309)

Examples for searching in the project (Page 310)

Searching and replacing in the editor (Page 315)

6.4.2.4 Creating a new search index manually

For the search, an index of the texts of the editors which can be searched in the project is built up automatically in the background. The search index is updated with each change. You cannot start the search before indexing is complete. You can also generate the search index manually at any time.

Procedure

To automatically generate the search index, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the group "General > Search in project".
3. Click "Recreate search index".

Result

The indexes of the project are generated again.

See also

Basics of searching (Page 303)
Basics for search in the project (Page 304)
Overview of the search editor (Page 305)
Search in entire project (Page 307)
Enabling and disabling "Search in project" (Page 309)
Examples for searching in the project (Page 310)

6.4.2.5 Enabling and disabling "Search in project"

You can disable "Search in project" centrally, if you do not need it. If necessary, you can enable again it at any time. By default, "Search in project" is enabled.

Procedure

To disable or enable "Search in project", follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the group "General > Search in project".
3. Select or clear the "Search in project" check box.

See also

Basics of searching (Page 303)
Basics for search in the project (Page 304)
Overview of the search editor (Page 305)
Search in entire project (Page 307)
Creating a new search index manually (Page 308)
Examples for searching in the project (Page 310)

6.4.2.6 Examples for searching in the project

Examples for use of property and object filters

The search editor offers filters in the "Limit search to" area which you can use to limit the search to specific properties or objects. Keep in mind the following mechanism:

- If you do not activate any filters, the search term is searched in all objects and properties that can be searched within the set search area.
- If you activate filters, only the selected properties or objects are searched.

Examples for use of property and object filters:

- Search for an object with the name "Motor1":
  Enter the search term and activate the filter "Name". All objects which include "Motor1" in their name are found.
- Search for an object which includes the text "Motor1" either in its name or in a comment:
  Enter the search term and activate the filters "Name" and "Comment". All objects which include "Motor1" in their name or in the comment are found.
- Search for a block with the name "Motor1":
  Enter the search term and activate the filters "Name" and "Block". All blocks which include "Motor1" in their name are found.
- Search for a block which includes the text "Motor1" either in its name or in a comment:
  Enter the search term and activate the filters "Name", "Comment" and "Block". All blocks which include "Motor1" in their name or in the comment are found.
- Search for a block or a technology object with the name "Motor1":
  Enter the search term and activate the filters "Name", "Block" and "Technology object". All blocks or technology objects which include "Motor1" in their name are found.

When entering the search term, note the following examples for searching for a term or several terms.

Examples for the search for one term

When your search entry consists of only one term and the option "Find exact match" is disabled, the search is always an implicit wildcard search. The system adds the wildcard "*" in front of and behind the search term. You can insert wildcards in the search term yourself to conduct an explicit wildcard search. If you want to find exactly the term you have entered, select the option "Find exact match". Uppercase and lowercase letters are not taken into account in the search.

Examples for the search for one term:

- Search for "motor", "Find exact match" option is disabled
  An implicit wildcard search is conducted. The search term is interpreted as "*motor*" and all terms which include "motor" are found, for example, "motor", "motor_1", "Motorstart" or "Start_Motor".
- Search for "m*tor" (explicit wild card search), "Find exact match" option is disabled
  The search term is interpreted as "*m*tor*" in this wildcard search. This means terms, such as "motor", "motor_1", "Start_Motor", "GLOB.Mode.motor[1]" or "Mo_tor" are also found.

- Search for "motor", "Find exact match" option is enabled
  No wildcards are used for the search, which means only the terms "motor" and "Motor" are found.
- Search for "m*tor" (explicit wild card search), "Find exact match" option is enabled
  The character "*" is no longer a wildcard and is interpreted as a regular character. The search only finds the character strings "m*tor" and "M*tor" but not, for example, the terms "motor", "", "Motormotor_1".

Application examples:

- If you want to determine from where the "GLOB_DB.MODE.FG[6].IN.ERROR_DROP_START" element of a data block is accessed, you can use the cross-reference list. If access takes place by indirect addressing (e.g. with "#GLOB.MODE.FG[#FgNr].IN.ERROR_DROP_START := TRUE;"), however, the cross-reference list does not return any results. Use a wildcard search for "MODE.FG[*].IN.ERROR_DROP_START" across the entire project in this case.
- You want to find all instances where "ERROR_DROP_START" is used. Perform a search for "ERROR_DROP_START" across the entire project. The search also finds all "ERROR_DROP_START" elements within the "FG[]" array.

Examples for the search for multiple terms

When you enter a search term which consists of multiple terms and the "Find exact match" option is disabled, you will find those objects which include all these terms. The texts can appear in any order. If you want the terms to appear exactly as you have entered them in an object, select the "Find exact match" option. Uppercase and lowercase letters are not taken into account in the search.

Examples for the search for multiple terms:

- Search for "motor start", "Find exact match" option is disabled
  All objects which include "motor" and "start" are found. It does not matter how far apart they are from each other or in which order they appear. Possible hits therefore include: "Motor start", "start my motor", "my motor is in start mode"

  Note

  The texts of a column are treated as a text object in data blocks and in block interfaces. Therefore, the search also returns data blocks and blocks in which the terms occur distributed over different lines.

- Search for "motor start*", "Find exact match" option is disabled
  As with the search for a word, an implicit wildcard search is always conducted for all words with a multiple word search. This means it is not necessary to explicitly enter a wildcard character. "motor start*" is interpreted as "*motor* *start*" and the same hits are found as with "motor start".

- Search for "motor start", "Find exact match" option is enabled
  When the option "Find exact match" is enabled, only objects which include both terms one after the other as a phrase are found. The search finds the search term, for example, in objects which include the texts "push motor start button" or "motor start on". No hits are returned for "start motor", "motorstart", "start my motor", "my motor is in start mode".
- Search for "motor start*", "Find exact match" option is enabled
  The character "*" is no longer a wildcard and is interpreted as a regular character. The search only finds objects in which the character string "motor start*" appears in uppercase or lowercase letters. "motor starting", "start my motor", "motor start", "Motor Start", "motor start_mode" do not result in any hits.

Application examples:

- You want to find out where you have used the function "Setbit" to set the "MotorOn" tag. To do so, conduct a search for "setbit motoron".
- You want to find a specific network comment but you only know that it includes the words "CPU" and "Cycle". To do so, start a search for "CPU cycle".

See also

Basics of searching (Page 303)

Basics for search in the project (Page 304)

Overview of the search editor (Page 305)

Search in entire project (Page 307)

Creating a new search index manually (Page 308)

Enabling and disabling "Search in project" (Page 309)

6.4.3 Searching and replacing in the editor

6.4.3.1 Basics for search in open editors

Introduction

You can start a search which is limited to the editor currently open in the workspace. You can customize the search as follows to suit requirements:

- You can optimize the search with additional options.
- You can specify the search area.
- You can specify the search direction.

Additional options for searching

You can refine the search with the help of the following additional options:

- Whole words only
  Only whole words are taken into account for the search. Compound words that contain the search key as a part are ignored.
- Match case
  Upper- and lowercase letters are taken into account in the search.
- Find in substructures
  The search also includes texts contained in another object.
- Find in hidden texts
  Texts that are assigned to another text but that are currently hidden are also included in the search.
- Use wildcards
  Enter an asterisk as the wildcard for any number of characters. Example: You want to search for all words starting with "Device". Type in "Device*" in the search key box. Enter a question mark as the wildcard, however, if you only want to exclude a single character.
- Use regular expressions (for searching in scripts only)
  A regular expression is a character string used to describe sets of values and for filtering. This allows you to create complex search patterns.

The additional options available depend on the installed products and the editor opened.

Search area and search direction

You have the following options for define the search area:

- Whole document
  The entire is searched regardless of the current selection and position.
- From current position
  The search starts at the current position. Depending on the search direction, all texts above or to the left or below or to the right of the current position are not included in the search.
- Selection
  The search term is only searched for in the current selection.

You can define the search direction independent of the search area:

- Down
  Search takes place in editor from top to bottom or from left to right.
- Up
  Search takes place in editor from bottom to top or from right to left.

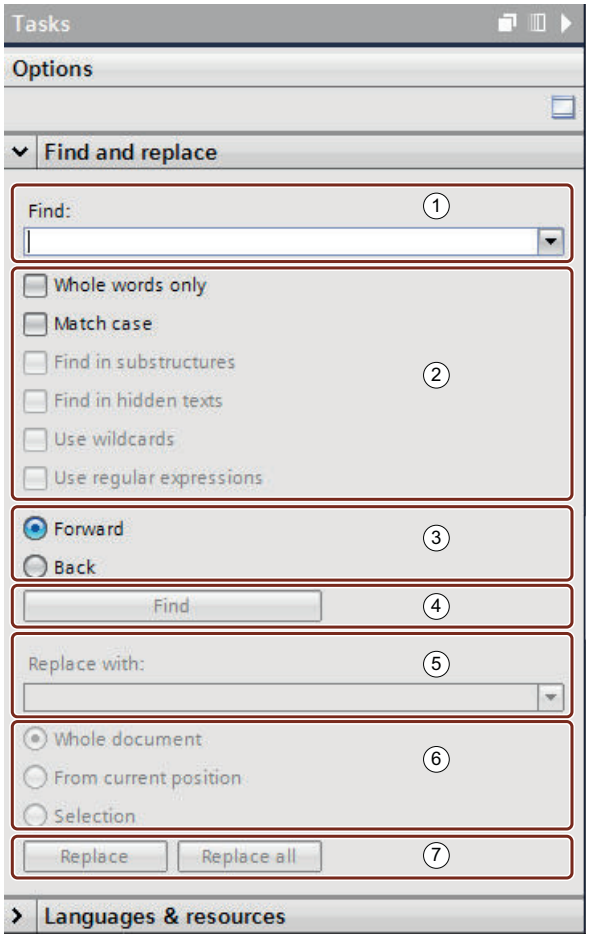
6.4.3.2 Overview of the "Find and replace" palette

Function of the "Find and replace" palette

The "Find and replace" palette in the "Tasks" task card lets you search in an open editor. It includes all options that you need for an efficient search.

Layout of the "Find and replace" palette

The following figure shows the components of the "Find and replace" palette:



① Enter search terms

② Additional options

③ Search direction

④ Start search

⑤ Enter replacement term

⑥ Replace area

⑦ Replace

6.4.3.3 Searching and replacing in the editor

You have the option to search for text in an editor or to replace it.

Start search

Follow these steps to start the "Find and replace" function:

1. Select the "Find and replace" command in the "Edit" menu or open the "Find and replace" pane in the "Tasks" task card.
   The "Find and replace" pane opens.
2. Enter a term in the "Find" drop-down list.
   As an alternative, you can select the most recent search key from the drop-down list.
3. Select the options desired for the search.
4. Using the option buttons, select the starting point for the search and the search direction.
5. Click "Find".
   The first hit is marked in the editor.
6. Click "Find" again to display the next hit.
   The next hit is marked in the editor. Repeat this step as needed until you reach the last hit.

Replacing the search key

You have the option of replacing hits individually or automatically replacing all the found texts, if the respective editor supports this function. Follow these steps to replace terms:

1. Enter a term in the "Find" drop-down list.
   As an alternative, you can select the most recent search key from the drop-down list.
2. Select the options desired for the search.
3. Click the "Find" button to start a search for the specified search key.
   The first hit is displayed in the editor.
4. In the "Replace" drop-down list, enter the text you wish to use to replace the search key.
   As an alternative, you can select one of the last entered texts from the drop-down list.
5. Click the "Replace" button to replace the selected hit with the specified text.
   The found text is replaced and the next hit is marked in the editor.
   Repeat this procedure until you have replaced all the desired hits. To skip to the next hit without replacing the marked word, click the "Find" button instead of "Replace".
6. Click "Replace all" to automatically replace all hits at once.

See also

Basics of searching (Page 303)

Search in entire project (Page 307)

6.5 Undoing and redoing actions

6.5.1 Basics of undoing and redoing actions

Function

You can undo performed actions at any time. For this purpose, every action you perform is saved in an action stack. When undoing actions, the stack is processed from top to bottom. In other words, if you undo an action that lies further down in the stack, all actions located above it in the stack will also be undone automatically.

You can redo previously undone actions until you execute a new action. Once you execute a new action, it is no longer possible to redo previously undone actions.

Particularities for undoing

There are a few actions that empty the action stack. You cannot undo these actions or the actions performed before these actions. The following actions empty the action stack:

- Saving
- Project management (creating a new project, opening project, closing a project, deleting a project)
- Deleting unused GSD files
- Establishing an online connection

Keep in mind that the action stack is emptied every time you establish an online connection. This means that the actions executed offline cannot be undone once the online connection is established. You can undo all actions which you subsequently execute online, even after the online connection has been terminated.

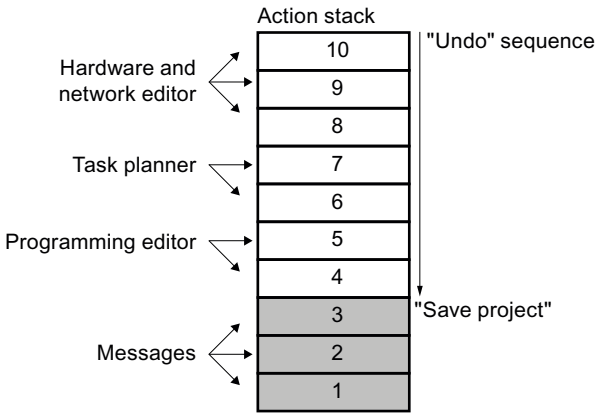
Displaying the action stack

The "Undo" button in the toolbar is enabled as soon as you perform an action that can be undone. This button is split; you can use the arrow down portion to open a drop-down list. containing all actions of the action stack that you can undo. If you had performed actions in an editor other than the currently displayed editor, the corresponding editor is also displayed as a subheading. This allows you to always identify the point at which the undo operation will be applied. The subheadings are removed from the list when the editor responsible can no longer undo actions.

Actions you have undone are entered in the action stack from where they can be redone. Here, you can redo actions you have undone. The display of actions you can redo it is analogous to the display of the actions that you can undo.

Example of undoing actions

The figure below shows how actions performed in various editors and tables are undone:



In this example, you cannot undo actions 1 to 3 because the project was saved. You can undo actions 4 to 10 in the order indicated by the direction of the arrow. In other words, you must undo action 10 first. Once you have undone action 8, you cannot then undo action 5. You must first undo actions 7 and 6. As the final step in the sequence, you can then undo action 4. You also have the option of undoing several actions in a single step by undoing an action located further down in the action stack. All actions located above it in the stack will be undone automatically.

The same principle also applies to redoing of actions.

See also

Undoing an action (Page 317)

Redoing an action (Page 318)

6.5.2 Undoing an action

The following options are available for undoing actions:

- Undoing the last action only
  Only the last action performed is undone.
- Undoing as many actions as required
  Multiple actions in the action stack are undone in a single step.

Undoing the last action only

To undo the last action performed, follow the steps below:

1. Click the "Undo" button in the toolbar.
   - If the action was not performed in the currently displayed editor, a confirmation prompt appears.
   - If the undo operation requires an editor containing a protected object to be opened, you must enter the password for the object.
2. Click "Yes" to confirm.
3. Enter the password, if necessary.
   The editor in which the action was performed is displayed and the action is undone.

Undoing as many actions as required

To undo multiple actions in the action stack in a single step, follow these steps:

1. Click the Down arrow next to the "Undo" button in the toolbar.
   This opens a drop-down list containing all actions you can undo. Actions performed in other editors are identified by the editor name in the subheading.
2. Click the action you want to undo.
   The chosen action and all actions in the stack located above the chosen action are undone. If the undo operation requires an editor containing a protected object to be opened, you must enter the password for the object.
3. Enter the required passwords, if necessary.
   The editors in which the actions were performed are displayed and the actions are undone.

See also

Basics of undoing and redoing actions (Page 316)

Redoing an action (Page 318)

6.5.3 Redoing an action

You have the option of redoing an action that has been undone, so that you can return to the status present before "Undo" was performed. However, this is only possible until you perform a new action. The following options are available for redoing actions:

- Redoing the last undone action only
  Only the last undone action is redone.
- Redoing as many undone actions as required
  Multiple undone actions in the action stack are redone in a single step.

Redoing the last undone action only

To redo the last undone action, follow the steps below:

1. Click the "Redo" button in the toolbar.
   - If the action is not being redone in the currently displayed editor, a confirmation prompt appears.
   - If the redo operation requires an editor containing a protected object to be opened, you must enter the password for the object.
2. Click "Yes" to confirm.
3. Enter the password, if necessary.
   The editor in which the action was undone is displayed and the action is redone.

Redoing as many undone actions as required

To redo multiple undone actions in the action stack in a single step, follow these steps:

1. Click the Down arrow next to the "Redo" button in the toolbar.
   This opens a drop-down list containing all actions you can redo. Actions performed in other editors are identified by the editor name in the subheading.
2. Click the action you want to redo.
   The chosen action and all actions in the stack located above the chosen action are redone. If the redo operation requires an editor containing a protected object to be opened, you must enter the password for the object.
3. Enter the required passwords, if necessary.
   The editors in which the actions were undone are displayed and the actions are redone.

See also

Basics of undoing and redoing actions (Page 316)

Undoing an action (Page 317)

Editing projects 7

7.1 Creating and managing projects

7.1.1 The basics of projects

Projects contain data and programs resulting from the creation of an automation solution. The data assembled in a project include, for example:

- Configuration data on the hardware structure and parameter assignment data for modules
- Project engineering data for communication over networks
- Project engineering data for the devices
- Logs for important events in the life cycle of the project

Project version

The project version defines the version of the TIA Portal for which a project is compatible. A project that is compatible with the current version of the TIA Portal is provided in project version V14 SP1 (or V14 in compatibility mode) of the TIA Portal.

You can view the project version of an already opened project in the project properties. To do so, select the "Software products in the project" entry in the "General" tab of the properties dialog. In the table, you can see all products and their versions, which are required for editing the open project.

Project hierarchy

Data is stored in a project in the form of objects. Within the project, the objects are arranged in a tree structure (project hierarchy).

The project hierarchy is based on the devices and stations along with the configuration data and programs belonging to them.

Common data of the project and online access, for example, are also displayed in the project tree.

Basic integrity check

The engineering software features a base-protective mechanism to detect changes in the essential custom project data (e.g. blocks, hardware configuration). Causes for changes to the project data, for example, can be tampering outside the engineering software or defects in the data storage medium.

The test of the configuration data is optional upon request, because, depending on the scope and application, it as can be time consuming, especially if you use multiuser engineering. You change the default for the test in the settings of the engineering software.

If a change is detected during a test, contact Siemens Customer Support. You can continue to work with the project data. Continue work at your own risk.

Note

Reduced working speed when basic integrity check is activated

If you enable the basic integrity check in the settings of the engineering software, the operating speed of the engineering software is reduced.

See also

Overview of the program settings (Page 230)

Changing the settings (Page 236)

Overview of settings for basic integrity check (Page 235)

Using logs (Page 329)

Creating a project (Page 322)

Using projects created with optional software (Page 334)

Opening projects (Page 323)

Upgrading projects (Page 336)

Displaying properties of the project (Page 328)

Saving projects (Page 325)

Closing projects (Page 326)

Removing projects from the project list (Page 326)

Deleting projects (Page 327)

7.1.2 Creating a project

Procedure

To create a new project, follow these steps:

1. Select the "New" command in the "Project" menu.
   The "Create a new project" dialog opens.
2. Enter a project name and path or accept the suggested settings.
3. In the "Version" drop-down list, select the project in which you want to create the project version.
   If you select "V14", the project is compatible with the TIA Portal V14 without installed Service Pack 1.
4. Click the "Create" button.

Result

The new project is created and displayed in the project tree.

See also

Compatibility between TIA Portal versions (Page 333)

The basics of projects (Page 321)

Using projects created with optional software (Page 334)

Opening projects (Page 323)

Upgrading projects (Page 336)

Displaying properties of the project (Page 328)

Saving projects (Page 325)

Closing projects (Page 326)

Removing projects from the project list (Page 326)

Deleting projects (Page 327)

7.1.3 Opening projects

You recognize projects of the TIA Portal by their file name extension ".ap[version number]". Projects of current version have the file name extension ".ap14". Projects with the version V14 and V14 SP1 can be opened immediately. Upgrade projects with the project version V13 SP1 before opening.

Procedure

To open an existing project, follow these steps:

1. Select the "Open" command in the "Project" menu.
   The "Open project" dialog box opens and the list of recently used projects is displayed.
2. Select a project from the list and click "Open".
3. If the project you require is not included in the list, click the "Browse" button. Navigate to the desired project folder and open the project file.
4. To check whether the project has been changed without permission, select the check box "Activate basic integrity check". The basic integrity check may take some time to complete.
5. Click the "Open" button.
   The project is opened. If you have selected a project from V13 SP1 version of the TIA Portal, the "Upgrade project" dialog opens. You can find more information on upgrading projects under "Upgrading projects (Page 336)".

Checking for missing software components

When opening existing projects, an automatic check is made to determine if the appropriate software is installed for all modules used within the project. If you try to open a project with modules that are not supported by the current scope of installation of the TIA portal, a message appears on opening the project informing you of the missing software components. If the software components are not absolutely required to open the project, the project can still be opened.

Reaction to missing software components

Unsupported modules of a project are handled as follows:

- Display of the modules on the user interface
  - The unsupported modules are displayed in the project tree with all of their lower-level objects. However, the modules themselves cannot be processed in editors or in the inspector window. When possible, a replacement module is used that best matches the original module. Replacement modules are indicated by an exclamation mark.
  - Display of properties in tables is limited. This applies in particular to the display of network parameters, such as the IP address.
- Functional limitations
  - Unsupported modules cannot be printed out or compiled.
  - An online connection cannot be established to the modules. It is therefore also impossible to download.
  - To change the device type, the device must first be deleted and then re-inserted. The "Change device type" function is not supported.
  - Copying and inserting lower-level objects, such as blocks, is possible although the device itself cannot be copied and inserted.
  - The network configuration cannot be changed with replacement modules within the network view.
  - Cross-references can be displayed. However, the cross-references only reflect the status last saved within the project because on online comparison with the original module cannot be made.

See also

The basics of projects (Page 321)

Creating a project (Page 322)

Using projects created with optional software (Page 334)

Upgrading projects (Page 336)

Displaying properties of the project (Page 328)

Saving projects (Page 325)

Closing projects (Page 326)

Removing projects from the project list (Page 326)

Deleting projects (Page 327)

Compiling project data (Page 348)

Opening a global library (Page 459)

7.1.4 Saving projects

You can save the project at any time either under the same or a different name. You can even save a project when it still contains elements with errors.

Saving a project

To save a project, follow these steps:

1. Select the "Save" command in the "Project" menu.
   All changes to the project are saved under the current project name. If you are editing a project from an earlier version of the TIA Portal, the file extension of the project is also retained and you can continue to edit the project in the earlier version of the TIA Portal.

Project Save as

To save a project under another name, follow these steps:

1. Select the "Save as" command in the "Project" menu.
   The "Save current project as" dialog opens.
2. Select the project folder in the "Save in" box.
3. Enter the new project name in the "File name" box.
4. Confirm your entry with "Save".
   The project is saved under the new name and opened.

Note

Undoing actions

Keep in mind that you cannot undo actions once you have saved the project.

See also

The basics of projects (Page 321)

Creating a project (Page 322)

Using projects created with optional software (Page 334)

Opening projects (Page 323)

Upgrading projects (Page 336)

Displaying properties of the project (Page 328)

Closing projects (Page 326)

Removing projects from the project list (Page 326)

Deleting projects (Page 327)

7.1.5 Closing projects

Procedure

To close a project, follow these steps:

1. Select the "Close" command in the "Project" menu.
   If you have made changes to the project since the last time you saved it, a message is displayed.
2. Decide whether or not you want to save the changes.

See also

The basics of projects (Page 321)

Creating a project (Page 322)

Using projects created with optional software (Page 334)

Opening projects (Page 323)

Upgrading projects (Page 336)

Displaying properties of the project (Page 328)

Saving projects (Page 325)

Removing projects from the project list (Page 326)

Deleting projects (Page 327)

7.1.6 Removing projects from the project list

You can remove projects from the list of recently used projects. The project data is retained on the storage medium.

Procedure

To remove a project from the list of recently used projects, follow these steps:

1. Select the "Delete project" command in the "Project" menu.
   The "Delete project" dialog opens and includes the list of most recently used projects.
2. Select a project from the list.
3. Click the "Remove" button.
4. Click "Yes" to confirm the prompt in order to remove the project from the list.

Result

The project is no longer displayed in the list of recently used projects. If you open the project again, it will be added to the list again.

See also

The basics of projects (Page 321)

Creating a project (Page 322)

Using projects created with optional software (Page 334)

Opening projects (Page 323)

Upgrading projects (Page 336)

Displaying properties of the project (Page 328)

Saving projects (Page 325)

Closing projects (Page 326)

Deleting projects (Page 327)

7.1.7 Deleting projects

Deleting removes the entire project data from the storage medium.

Requirement

The project you want to delete is not open.

Procedure

Follow the steps below to delete an existing project:

1. Select the "Delete project" command in the "Project" menu.
   The "Delete project" dialog opens and includes the list of most recently used projects.
2. Select a project from the list.
   If the project you require is not included in the list, click the "Browse" button. Navigate to the desired project folder, and open the project file.
3. Click the "Delete" button.
4. Click "Yes" to confirm. This starts the deletion of the project.

Result

The entire project folder is deleted from the file system.

See also

The basics of projects (Page 321)

Creating a project (Page 322)

Using projects created with optional software (Page 334)

Opening projects (Page 323)

Upgrading projects (Page 336)

Displaying properties of the project (Page 328)

Saving projects (Page 325)

Closing projects (Page 326)

Removing projects from the project list (Page 326)

7.1.8 Displaying properties of the project

You can display the properties of a project. In the "General" tab you find the following project properties in the area navigation:

- Metadata for the project
  This includes the following information: creation time, author, file path, project size, copyright, project languages, etc. Many of the properties can be changed.
- Project history
  The project history contains an overview with important events in the project life cycle. Here, for example, you can see the version of the TIA Portal used to create a project and whether it has been upgraded in the meantime. If a project was created during a migration, for example, this is also indicated in the project history table with the date and time of the migration. If a log was created for an event, you can also call the log directly.
- Support packages in the project
  An overview of the add-on software needed to work with all devices in the project is displayed. In addition, installed GSD files are listed (device description files for other devices in the hardware catalog).
- Software products in the project
  You can display an overview of all installed software products needed for the project. The "Version" column shows the project version of the respective software with which the project is stored.

In the "Protection" tab you specify that blocks can be simulated after the compilation with SIMATIC S7-PLCSIM Advanced. To do this, select the check box "Support simulation during block compilation". By default, the function is disabled.

Procedure

To display the project properties, follow these steps:

1. Select the open project in the project tree.
2. Select "Properties" in the shortcut menu of the project.
   The dialog with the properties of the project opens.
3. Select the project properties in the area navigation that you want to have displayed.

See also

The basics of projects (Page 321)

Creating a project (Page 322)

Using projects created with optional software (Page 334)

Opening projects (Page 323)

Upgrading projects (Page 336)

Saving projects (Page 325)

Closing projects (Page 326)

Removing projects from the project list (Page 326)

Deleting projects (Page 327)

7.1.9 Using logs

For some operations within the TIA Portal, logs are created automatically in the background. These logs document changes in the project. Logs are created automatically, for example, when you migrate projects and programs or when you update instances from the library.

Logs are displayed in the "Common data" folder in the project tree. They are stored together with the project in the project folder and can therefore be read independently of the programming device/PC used as soon as you have opened the project. The log can be filtered for errors, warnings, and information.

In addition to displaying them in the TIA Portal, logs can also be printed.

Displaying logs

To open a log, follow these steps:

1. Open the "Common data > Logs" folder in the project tree.
2. Double-click the desired log in the list.
   The contents of the log are displayed in the work area.
3. Optional: To show or hide a particular category of alarms, activate or deactivate the button for "Errors", "Warnings", or "Information" in the toolbar.

Deleting logs

To delete a log, follow these steps:

1. Select the log in the project tree.
2. Press <Del>.
   The selected log is deleted from the project directory and removed from the project tree.

7.1.10 Security of industrial plants

In addition to numerous advantages, the ongoing digitization of industrial automation is accompanied by increased vulnerability of industrial plants. The TIA Portal and the SIMATIC products already have several features that ensure secure operation of your plant, for example:

- Know-how protection of blocks
- Basic integrity protection for projects to protect projects against manipulation
- The protection and security functions of CPUs

Further information and the possibility for reporting security problems can be found online at the following link:

Whitepapers and additional information (https://www.industry.siemens.com/topics/global/en/industrial-security/support/Pages/white-papers.aspx)

Industrial Security

Industrial Security goes beyond the security regulations and measures in office environments. Industrial Security is a comprehensive approach to protecting industrial plants. The approach is designed to protect against unauthorized access, sabotage, espionage and malicious tampering. Industrial security ensures the following factors:

- Availability
  Availability has highest priority in industrial plants. Downtime must be kept as low as possible.
- System and data integrity
  A third party should not be able to unlawfully modify the plant and any of its data.
- Confidentiality
  Information on the industrial plant and data must not fall into the hands of unauthorized third parties.

Comprehensive approach

In addition to the security features integrated in the TIA Portal, Industrial Security requires more comprehensive measures. Observe the following security aspects:

- Plant safety
  - Access protection against unauthorized persons
  - Physical access protection for critical components
- Network security
  - Controlled interfaces between an office and plant network, for example, through firewalls
  - Segmentation of the plant network
- System integrity
  - Use of anti-virus and whitelisting software
  - Compliance with maintenance and update processes
  - User authentication for the machine or plant operator
  - Integrated access protection in automation components

7.1.11 Importing and exporting CAx data

CAx files are XML files that you create in the TIA Portal and that you can swap between different installations or with other tools such as ECAD systems. The files contain the following information:

- Hardware configuration of a single device or an entire project (without parameter assignment)
- Folder hierarchies (groups)
  The folder hierarchy is restored for the import.
- Association of a device to a subnet
  When importing, the devices stored in the CAx file are reconnected to the same subnet.
- Addresses of the devices
- Address areas of inputs and outputs
- Profinet I/O systems and PROFIBUS DP master systems

Requirements

- The "Openness" add-on package is installed.
  If you have not installed the add-on package, install it from the TIA Portal installation DVD. The add-on package is stored in the "\support" folder on the DVD.
- A project is open.

Exporting the CAx file of a project

1. In the "Tools" menu, click on "Export CAx data...".
2. In the "CAx Export" dialog, select a directory for the file to be exported.
3. If necessary, change the file name and confirm with "Save".
   The exported file receives the file extension ".aml".
   You see the status of the export in the "Info > General" tab of the Inspector window.

Exporting the CAx file of a device

1. Highlight the exported devices in the project tree.
2. Select "Export CAx data ..." from the shortcut menu.
3. In the "CAx Export" dialog, select a directory for the file to be exported.
4. If necessary, change the file name and confirm with "Save".
   The exported file receives the file extension ".aml".
   You see the status of the export in the "Info > General" tab of the Inspector window.

Import CAx file of a device or a project

1. In the "Tools" menu, click on "Import CAx data...".
2. Select a CAx file in a directory in the "CAx Import" dialog.
3. Click the "Open" button.
   You receive feedback about the status of the import of devices, for example error messages, in the inspector window in the "Info > General" tab.

Import options with name conflicts of stations

If you want to import a CAx file and the currently open TIA Portal project included stations with identical names, a conflict arises. A dialog for setting the import options opens. Before you import the file, you select how the stations with the same name are to be merged. The stations from the CAx file can be optionally renamed and grouped together in a new folder.

Note

Default name of the folder

The folder for the stations to be imported are named "ParkingLot" by default.

You can specify the name of the new folder yourself. To do this, open the settings and navigate to the "CAx" item. Enter the name for the folder with "Settings for conflict resolution" in the "Name of store for devices that already exist in the project" field.

Optionally, you can also specify that the stations triggering a conflict are either not to be imported or replaced by stations of the same name in the currently open TIA Portal project.

7.2 Compatibility of projects

7.2.1 Compatibility between TIA Portal versions

With the TIA Portal V14 SP1, you can also open projects that were created with the version V14 of the TIA Portal. You need to upgrade projects from previous versions of the TIA Portal to the current version before opening them.

Opening projects from previous versions of the product

The following table shows the response of the TIA Portal when opening projects from a previous version of the product:

Version of the TIA Portal (file extension of the respective version)	Response when opened with the current version of the TIA Portal
V10.5 (.ap10) V11.x (.ap11) V12 (.ap12) V12 SP1 (.ap12) V13 (.ap13)	Projects from previous versions of V13 SP1 of the TIA Portal cannot be directly opened with TIA Portal V14 SP1. When you attempt to open the project, a message prompts you to open the project with the version V13 SP1. If you open the project with TIA Portal V13 SP1 version, the project is upgraded to the project version V13 SP1 after your confirmation and is assigned the file extension ".ap13".
V13 SP1 (.ap13)	Once you give your confirmation, the project is upgraded project version V14 SP1 upon opening and obtains the file extension ".ap14".
V14 (.ap14)	The project is opened in compatibility mode.

Backward compatibility of the current version of the TIA Portal

Projects that you save in the current project version V14 SP1 are not backward compatible due to the expanded functionality over previous versions. Projects in version V14 SP1 can be only be opened with TIA Portal V14 SP1 or higher.

7.2.2 Using projects created with optional software

A wide range of optional software is available for the TIA Portal. This includes hardware support packages, for example. When you open a project that was created with optional software but the optional software is not installed, the following will occur:

- Software components which are not absolutely required are missing:
  A dialog appears listing the missing software components. After the project is opened, its properties are displayed. You now have the opportunity to install missing components. All the devices contained in the project are available even if you do not install the missing components. However, you can only work with the devices that are supported by the currently installed software.
  Devices which are not supported because software components are missing are identified by the following symbol in the project tree:
  
- Essential software components are missing:
  A dialog appears listing the missing software components. The essential software components are marked. The project can only be opened if you install the missing software components.

See also

The basics of projects (Page 321)

Creating a project (Page 322)

Opening projects (Page 323)

Upgrading projects (Page 336)

Displaying properties of the project (Page 328)

Saving projects (Page 325)

Closing projects (Page 326)

Removing projects from the project list (Page 326)

Deleting projects (Page 327)

Compatibility of global libraries (Page 459)

7.2.3 Compatibility mode

When you create or open a project that was created with V14 project version and do not upgrade it, the project is in compatibility mode. As long as you do not use functions that have been newly added to TIA Portal V14 SP1, you work in compatibility mode. The project remains backward compatible and can be continue to be opened and edited with TIA Portal V14. You only have to upgrade the project when you use features newly added with Service Pack 1. The backward compatibility is lost in this case.

Global libraries are always created in the most recent format and are not backwards compatible, even if the project is opened in compatibility mode.

Libraries in compatibility mode

If you use global libraries in compatibility mode, observe the following:

- System libraries
  In compatibility mode, only system libraries that can be used with the project version of the opened project are displayed. Once you upgrade the project, the system libraries that can be used with the current version of the TIA Portal are also displayed.
- Custom global libraries
  You can load global libraries created with the TIA Portal V14 SP1 in compatibility mode. However, once you use elements from the library that are not compatible with the version of the TIA Portal V14, you must upgrade the project to project version V14 SP1.

Adding supported devices from hardware support packages

To projects in compatibility mode, components can be added that were supplied at a later date for TIA Portal V14 as part of a Hardware Support Package (HSP). To continue working with the project in TIA Portal V14, install the appropriate hardware support package in TIA Portal V14.

Loading hardware configuration and software in compatibility mode

When you load the hardware configuration and software in compatibility mode on a device, the device remains compatible with the TIA Portal V14. The device can be continued to be loaded with the TIA Portal V14 and all online functions are available.

Upgrading projects

You need to upgrade the project to use the full range of functions of the current version. You automatically receive a corresponding message when you use a function that was newly added with Service Pack 1. Upgrading converts the project to the current project version and exits compatibility mode. The full range of functions of the current version becomes available.

7.2.4 Upgrading projects

7.2.4.1 General information on upgrading projects

A project from the version V13 SP1 of the TIA Portal can be opened only after they have upgraded the project to the current version of the TIA Portal. When you open the project, you are prompted to upgrade the project.

To learn about the options available with other product versions of the project, refer to the section "Compatibility between TIA Portal versions (Page 333)".

Upgrading modules with know-how protection

Blocks with know-how protection are not automatically upgraded with the project. Remove the know-how protection of the blocks before you upgrade the project:

- For V13 SP1 projects
  Remove the know-how protection with TIA Portal V13 SP1.
- For V14 projects
  Remove the know-how protection with TIA Portal V14 or the TIA Portal V14 SP1 in compatibility mode.

Set up know-how protection again afterwards with the latest version of the TIA Portal. You can find additional information in the section "Upgrading know-how protected blocks (Page 338)".

Upgrading global libraries

Global libraries are not upgraded automatically together with the project because they are independent of projects. If you want to continue using global libraries from older versions of the TIA Portal, upgrade the global libraries as well. You can find more information on upgrading global libraries in the section "Compatibility of global libraries (Page 459)".

Instruction version

The instructions that you use to create your user program may be in different versions. Different product versions of the TIA Portal can contain different instruction versions. You have the option to upgrade the project and instructions to the latest version. You can then use the new instruction versions. After upgrading, the new instruction versions are not automatically used in your program. You can update your program to use the new versions. You can find more about "Instruction versions" in the "Programming editor" section.

Multiuser Engineering

To upgrade multiuser engineering projects, read about the prescribed procedure in the section "Upgrading multiuser engineering projects (Page 338)".

See also

Using projects created with optional software (Page 334)

Upgrading projects (Page 336)

7.2.4.2 Upgrading projects

The following will show you how to upgrade projects from earlier versions of the TIA Portal.

Upgrading projects from TIA Portal V13 SP1

To upgrade a project from TIA Portal V13 SP1, follow these steps:

1. Select the "Open" command in the "Project" menu.
   The "Open project" dialog box opens and the list of recently used projects is displayed.
2. Select a project from the list and click "Open".
3. If the project you require is not included in the list, click the "Browse" button. Navigate to the desired project folder and open the project file.
   The "Upgrade project" dialog box opens.
4. Confirm the upgrade of the project with "OK".
   The project will be upgraded and opened in the current version of the project.
5. Compile the hardware and software of all devices in the project.

Upgrading TIA Portal V13 SP1 projects with master copies from TIA Portal V10.5

Before upgrading a project, check if the master copies created with TIA Portal V10.5 are contained in its project library. This is necessary, for example, if you receive a message during the upgrade about an S7-1200 CPU with firmware version V1.0 used in the project, even though you have not configured such a CPU. The CPU is a master copy in the project library in this case.

Before upgrading a project with master copies from TIA Portal V10.5, therefore, do the following:

1. Use each master copy created with the TIA Portal V10.5 once in the project.
2. Delete the affected master copies from the project library.
3. Create the master copy again with the objects from the project.
4. Save the project in TIA Portal V13 SP1.

You can then upgrade the project to the latest version.

Upgrading projects from the TIA Portal V13 or earlier

Projects of product version V13 or earlier cannot be upgraded and opened with the TIA Portal. First upgrade projects from TIA Portal V13 or earlier with the product version V13 SP1. Then upgrade the projects with the current version of the TIA Portal again to the project version V14 SP1.

You can find additional information on upgrading projects from earlier product versions in the Siemens Industry online support (https://support.industry.siemens.com/cs/ww/en/view/109476392).

See also

Upgrading know-how protected blocks (Page 338)

General information on upgrading projects (Page 335)

Upgrading multiuser engineering projects (Page 338)

Compatibility of global libraries (Page 459)

The basics of projects (Page 321)

Creating a project (Page 322)

Using projects created with optional software (Page 334)

Opening projects (Page 323)

Displaying properties of the project (Page 328)

Saving projects (Page 325)

Closing projects (Page 326)

Removing projects from the project list (Page 326)

Deleting projects (Page 327)

Compiling project data (Page 348)

7.2.4.3 Upgrading multiuser engineering projects

The following will show you how to upgrade multiuser engineering projects.

Procedure

To upgrade multiuser engineering projects, follow these steps:

1. Before upgrading, save all existing local sessions in the multi-user server project.
2. Save the multiuser server project as a single-user project.
3. Upgrade the single-user project.
4. After the upgrade, create a multiuser project again from the single-user project.
5. Create the local sessions again from the upgraded multiuser project for all workers.

See also

Upgrading projects (Page 336)

Introduction to Multiuser Engineering (Page 6687)

7.2.4.4 Upgrading know-how protected blocks

For security reasons, the know-how protection of blocks is tied to the version of the TIA Portal with which the know-how protection was set up. This means blocks that were assigned know-how protection in an earlier product version cannot be opened and edited with the previous password after the upgrade. However, the blocks can be downloaded to the controller and are executable. If you want to open and edit the block after the upgrade, remove the know-how protection of the blocks and set it up again.

Procedure

To upgrade know-how protected blocks, follow these steps:

1. Remove the know-how protection of the blocks from the previous version in the "Edit" menu with the "Know-how protection" command.
   For more detailed information, refer to section "Removing know-how protection for blocks".
2. Upgrade the project as described in the section "Upgrading projects (Page 336)".
3. Set up the know-how protection of the blocks from the current product version again in the "Edit" menu with the "Know-how protection" command.
   For more detailed information, refer to section "Setting up know-how protection for blocks".

7.2.5 Compatibility with the plant configuration

To be able to use all the online functions, the plant configuration with TIA Portal V14 SP1 must be compatible to the project version of the opened project. The following factors for the compatibility are especially relevant when designing the plant:

- Hardware and its firmware version
  The individual devices and their firmware version must be supported by the TIA Portal V14 SP1 and within the opened project.
- The user program on the respective devices
  All blocks and instructions on the devices must be available in TIA Portal V14 SP1 and ready for using with the project version of the opened project.

Influence of the project version on compatibility

With TIA Portal V14 SP1, you can edit projects in compatibility mode for project version V14. If you have opened a project in compatibility mode, you can only load the hardware configuration and software of the system which is compatible with the TIA Portal V14. If the hardware configuration or the software in the plant uses the functionality of a newer version of the TIA Portal, it cannot be loaded with a project in compatibility mode. To load the hardware configuration or software of the plant in this case, upgrade the open project to the current version of the TIA Portal.

With CPUs of the S7-1200 and S7-1500 product series, you see the project version with which the blocks on the CPU are compatible on the display, on the Web server and in the online and diagnostics view of the TIA Portal. Note, however, that a compatible version of the project alone is not enough to work with the CPU. The CPU itself and its firmware version must be compatible with the installed version of the TIA Portal and the various installed products.

You can see the project version of the currently open project in the project properties.

Incompatible blocks

An online/offline comparison is performed for the configured device with an existing online connection. If the modules available online are incompatible with the project version of one project, incompatible blocks in the project tree are indicated as such by the following symbol:



Loading hardware configuration and software on a device

If you want to load the hardware configuration or software on a device that was last loaded with TIA Portal V13 SP1 or an earlier version, then load the entire hardware configuration and software to the device again. It is not possible in this case to only load the changes. On devices that were last loaded TIA Portal V14 or higher, you can only load the changes to the hardware configuration and software.

See also

General information on loading (Page 348)

The basics of projects (Page 321)

General information on upgrading projects (Page 335)

Upgrading projects (Page 336)

Basics of project data comparison (Page 358)

Displaying properties of the project (Page 328)

7.3 Archiving and retrieving projects

7.3.1 Working with project archives

Archiving and transferring projects

If you work for a long time with a project, large files may result, especially with extensive hardware configurations. Therefore, you may want to reduce the size of the project, for example, when you archive it to an external hard drive, or when you send it via e-mail and require a smaller file size.

You can create a project archive to reduce the size of the project. TIA Portal project archives are compressed or uncompressed files, each containing an entire project including the entire folder structure of the project. Before the project directory is compressed into the archive file, all files are reduced to their essential components to further decrease the size of the project.

Project archives have the file extension ".zap[version number of the TIA Portal]". Projects created with TIA Portal V14 have the file extension ". zap14".

To open a project archive, retrieve the project archive. By retrieving it, the archive file with the included project files is extracted to the original project directory structure.

See also

Retrieving compressed project (Page 342)

Creating a project archive (Page 341)

7.3.2 Creating a project archive

You can reduce the required storage space of the currently open project by archiving the project in a compressed file.

Note

The most recently saved state of the project is used for archiving. Therefore, save the project before using the archiving function. This will ensure that your most recent changes are included in the archived project.

Procedure

To archive a project, follow these steps:

1. Select the "Archive ..." command from the "Project" menu.
   The "Archive project" dialog box opens.
2. To create a compressed archive file, select the "TIA Portal project archive" option.

3. If you do not want to archive the search index and the HMI compile result, select the option "Discard recoverable data".
   You can restore the discarded data if needed.
4. To add a date and time automatically, select the "Add date and time to the file name".
   The extension is shown in the "Resulting file name" box behind the selected file name.
5. Enter a file name in the "File name" box.
6. Select the path to the archive file.
   You can set the default directory under "Options > General > Storage location for project archives".
7. Click "Archive".

Result

A compressed file with the extension ".zap14" is generated.

The archive file contains the complete project directory. The individual files of the project are also reduced to the essential components in order to save space.

See also

Working with project archives (Page 341)

Retrieving compressed project (Page 342)

7.3.3 Retrieving compressed project

You extract project archives of the TIA Portal with the "Retrieve" function. This restores the project directory structure including all project files.

Requirement

No project is open.

Procedure

To extract a project archive, follow these steps:

1. Select the "Retrieve" command in the "Project" menu.
   The "Retrieve archived project" dialog opens.
2. Select the project archive.
3. Click "Open".
4. The "Find folder" dialog opens.
5. Select the target directory to which the archived project should be extracted.
6. Click "OK".

Result

The project is extracted to the selected directory and opened immediately. If you extract a project archive that includes a project from the product version V13 SP1, you may have to upgrade the project. You will automatically receive the corresponding prompt as soon as you open the project. The same rules apply that are described in the chapter "Using projects created with optional software (Page 334)". The search index and HMI compile result are automatically restored in the background.

See also

Working with project archives (Page 341)

Opening projects (Page 323)

Using projects created with optional software (Page 334)

Upgrading projects (Page 336)

7.4 Using reference projects

7.4.1 Basics of reference projects

Introduction

You can open other projects as a reference in addition to the current project. You can use these reference projects as follows:

- You can drag individual objects from a reference project into the current project and then edit them.
- You can open specific objects, for example, code blocks from a reference project as read-only. But this is not possible for all elements.
- You can use an offline/offline comparison to compare devices of the reference project to devices from the current project.

Note that reference projects are read-only. You cannot change the objects of a reference project.

Projects that were created with an older version of the TIA Portal or with a different installation package can also be opened as reference projects. The same compatibility rules apply here as for normal opening of a project from an older version of the TIA Portal.

See also: Compatibility of projects (Page 334)

See also

Comparing reference projects (Page 345)

Opening and closing a reference project (Page 344)

Reference projects (Page 259)

7.4.2 Opening and closing a reference project

Opening a reference project

To open a reference project, follow these steps:

1. In the "View" menu, select the "Reference projects" check box.
   The "Reference projects" palette is displayed in the project tree.
2. In the "Reference projects" palette of the project tree, click on "Open reference project" in the toolbar.
   The "Open reference project" dialog box opens.

3. Navigate to the desired project folder and open the project file. TIA Portal V14 projects have the extension ".ap14". Older projects of the TIA Portal have the extension ".ap[version number]".
4. Click "Open".
   The selected project is opened as a read-only reference project.

Closing a reference project

To close a reference project, follow these steps:

1. In the "Reference projects" palette of the project tree, select the reference project you want to close.
2. Click on "Close reference project" in the toolbar.
   The selected reference project is closed.

See also

Basics of reference projects (Page 344)

Comparing reference projects (Page 345)

Reference projects (Page 259)

7.4.3 Comparing reference projects

Introduction

You can compare devices from reference projects with devices from both the current project as well as from the same or another reference project or from a library.

Note

Please note the following:

- You cannot specify actions for the comparison objects, since the reference projects are write-protected.
- You can perform a detailed comparison for the comparison objects, if the type of comparison object generally allows a detailed comparison.
- When comparing reference projects, you can always switch between automatic and manual comparison.

Procedure

To compare the objects of a reference project to the device data of the current project, follow these steps:

1. In the project tree, select the device whose data you want to compare to the data of a reference project and which allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Open the "Reference projects" palette in the project tree.
4. Select the device of a reference project that you want to compare to the device data from the current project.
5. Drag the device from the reference project into the right drop area of the compare editor.
   You can identify the status of the objects based on the symbols in the status and action area. When you select an object, the object properties and the corresponding object of the assigned device are clearly shown in the property comparison.
   You can drag other devices into the drop areas from the current project, a library or from a reference point at any time to start a new comparison. It does not matter which device you drag into the drop area.
6. If needed, perform a detailed comparison of objects for which this type of comparison is possible.

Note

You can start the detailed comparison directly from the reference project for blocks, PLC tags and PLC data types. You can find additional information in the section "Comparing PLC programs".

See also

Running a detailed comparison (Page 371)

Basics of reference projects (Page 344)

Reference projects (Page 259)

Opening and closing a reference project (Page 344)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Editing project data

8

8.1 Compiling and loading project data

8.1.1 Compiling project data

8.1.1.1 General information on compiling project data

Compiling project data

During compilation, project data is converted so that it can be read by the device. Hardware configuration data and program data can be compiled separately or together. You can compile the project data for one or more target systems at the same time.

The following project data must be compiled prior to loading:

- Hardware project data, for example, configuration data of the devices or networks and connections
- Software project data, for example, program blocks or process screens

Note

While a device is being compiled, no additional compiling process can be started. Note in this regard that you can not only perform a compiling process manually, but you can also trigger it automatically for operator control and monitoring devices.

Scope of the compilation

When you compile project data, you have the following options depending on the device involved:

- Hardware and software (only changes)
- Hardware (only changes)
- Hardware (rebuild completely)
- Software (only changes)
- Software (rebuild all)
- Software (reset memory reserve)

See also

Compiling project data (Page 348)

8.1.1.2 Compiling project data

The following section describes the general procedure for compiling project data in the project tree. You will find details of how certain objects are compiled and any special points to note in the online help of the product.

Procedure

To compile project data, follow these steps:

1. In the project tree, select the devices whose project data you want to compile.
2. Select the option you require in "Compile" submenu of the shortcut menu.

Note

Note that the options available to you depend on the selected device.

The project data is compiled. You can check whether or not the compilation was successful in the Inspector window with "Info > Compile".

See also

General information on compiling project data (Page 347)

8.1.2 Loading project data

8.1.2.1 General information on loading

Introduction

In order to set up your automation system, you need to load the project data you generated offline on the connected devices. This project data is generated, for example when configuring hardware, networks, and connections or when programming the user program or when creating recipes.

The first time you download, the entire project data is downloaded. During later loading operations, only changes are downloaded.

You can download the project data to devices and memory cards.

Note

While a device is being compiled, no additional download process can be started. Note in this regard that you can not only perform a compiling process manually, but you can also trigger it automatically for operator control and monitoring devices.

Possible options for downloading

Depending on the object you want to download, you have the following options:

- Hardware and software (only changes)
  Both the hardware configuration and software are downloaded to the destination if differences exist between the online and offline versions.
- Hardware configuration
  Only the hardware configuration is downloaded to the destination.
- Software (only changes)
  Only the objects that differ online and offline are downloaded to the destination.
- Load PLC program to device and reset
  All the blocks are loaded to the destination and all values are reset to their initial state. Be aware that this also applies to retentive values.

You can also upload project data already contained in a device back to your project. You have the following options:

- Upload entire device as new station
  The hardware configuration of the device and the software on the device are loaded in the project.
  All relevant data of the device is uploaded to the project.
- Upload software of a device
  Only the blocks and parameters from the device are uploaded to an existing CPU in the project.

In both cases, during loading all instances of library types are connected again with the corresponding version of the type in the project library. If no suitable type is yet available for a loaded instance or the correct version of the type does not exist in the project library, the type or the version is added to the project library.

Downloading with synchronization

In team engineering, it is possible for several users to work on one project with several engineering systems at the same time and to access one S7-1500 CPU. To ensure consistency within the shared project, it is necessary to synchronize the changed data prior to loading so that nothing gets overwritten unintentionally.

If differences are detected between the online and offline data management within the shared project that were caused by a different engineering system, automatic synchronization of the data to be loaded is offered when loading.

In this case, the "Synchronization" dialog displays the data to be synchronized with the current status (online-offline comparison) and the possible actions.

Use case	Recommendation	Synchronization
One or more blocks on the CPU (online) are more recent than in the engineering system (offline).	These blocks should be uploaded from the CPU to the engineering system before downloading.	Automatic synchronization is possible: The blocks in the engineering system are updated prior to loading.
One or more new blocks have been created and exist only on the CPU (online).	These blocks should be uploaded from the CPU to the engineering system before downloading.	Automatic synchronization is possible: The new blocks are added to the engineering system prior to the download.

Use case	Recommendation	Synchronization
One or more blocks on the CPU have been deleted.	The blocks should also be deleted prior to the download in the engineering system.	Automatic synchronization is not possible. The blocks deleted on the CPU should be manually deleted in the offline project in the engineering system.
One or more blocks on the CPU and in the engineering system are different. This is the case when a different user has changed blocks to which you have also made corrections and has already downloaded them to the CPU.	These blocks with competing changes must be adapted manually. You decide in this case which changes you are going to accept. If the blocks on the CPU are to be retained, you should adopt these blocks from the CPU in your engineering system before downloading to the CPU. If the blocks that you have changed are to be applied, you can continue with the download without synchronization.	Automatic synchronization is not possible: The affected blocks on the CPU or in the engineering system must be adapted manually. One of the existing block versions (online or offline) will be overwritten in the process.
There are differences in the hardware configuration on the CPU (online) and in the engineering system (offline).	Differences in the hardware configuration must be adapted manually. You decide in this case which hardware configuration you are going to accept. If the existing hardware configuration on the CPU is to be retained, you should adopt this in your engineering system prior to downloading. If you want to apply the hardware configuration you changed, you can continue downloading without synchronization.	Automatic synchronization is not possible: The hardware configuration must be adapted manually. One of the existing hardware configurations (online or offline) will be overwritten.

If required, you can use the "Force download to device" command to download blocks without synchronization.

See also

Downloading project data to a device (Page 350)

Downloading project data to a memory card (Page 352)

Uploading project data from a device (Page 354)

8.1.2.2 Downloading project data to a device

The following section describes the general procedure for downloading project data to a device. You will find details of how certain objects are downloaded and any special points to note in the online help of the product.

Requirement

- The project data is consistent.
- Each device to which you want to download is accessible via an online access.

Procedure

To download the project data to the selected devices, follow these steps:

1. Select one or more devices systems in the project tree.
2. Right-click on a selected element.
   The shortcut menu opens.
3. Select the option you require in the shortcut menu of the "Download to device" submenu.

   Note

   Note that the options available to you depend on the selected device.

   When necessary, the project data is compiled.

   – If you had previously established an online connection, the "Load preview" dialog opens. This dialog displays messages and proposes actions necessary for downloading.
   – If you had not previously established an online connection, the "Extended download to device" dialog opens, and you must first select the interfaces via which you want to establish the online connection to the device. You have the option of showing all compatible devices by selecting the corresponding option and clicking the "Start search" command.
     See also: AUTOHOTSPOT
4. Check the messages in the "Load preview" dialog, and select the actions in the "Action" column, if necessary.

   ⚠ WARNING

   Preventing personal injury and material damage

   Performing the proposed actions while the plant is in operation can cause serious bodily injury and property damage in the event of malfunctions or program errors.

   Make sure that no dangerous situations can arise before you start the actions.

   As soon as loading becomes possible, the "Load" button is enabled.
5. Click the "Load" button.
   The loading operation is performed. If there is a need for synchronization, the system automatically displays the "Synchronization" dialog. This dialog displays messages and suggests actions that are required for the synchronization. You have the option of performing these actions or forcing the download without synchronization by clicking "Force download to device". If you have performed the suggested actions, you will be asked whether you want to continue with the download. The "Load results" dialog then opens. In this dialog, you can check whether or not the loading operation was successful and take any further action that may be necessary.
6. Click the "Finish" button.

Result

The selected project data was downloaded to the devices.

See also

General information on loading (Page 348)

Downloading project data to a memory card (Page 352)

Uploading project data from a device (Page 354)

8.1.2.3 Downloading project data to a memory card

You have the option of loading project data to a memory card. For CPUs of the S7-300/400 series you can also explicitly download the user program to a memory card inserted in the CPU.

To download project data to a memory card, you have the following options:

- Dragging project data to a memory card
- Writing project data to a memory card
- Downloading user program to a memory card that is inserted in a CPU of the S7-300/400 series

Requirement

A memory card is displayed.

See also: Accessing memory cards (Page 422)

Downloading project data to a memory card

To download project data to a memory card, follow these steps:

1. In the project tree, drag the project data you want to download to the memory card.
   If necessary, the project data is compiled. The "Load preview" dialog then opens. This dialog displays alarms and recommends actions needed for the loading operation.
2. Check the alarms and select the actions in the "Action" column, if necessary.
   As soon as downloading becomes possible, the "Load" button is enabled.
3. Click the "Load" button.
   The loading operation is performed.

Or:

1. In the project tree, select the project data that you want to download.
2. To do this, right-click the selection and select the "Copy" command from the shortcut menu. You can also use the key combination <Ctrl+C>.
3. Right-click the memory card and select the "Paste" command from the shortcut menu. You can also use the key combination <Ctrl+V>.
   If necessary, the project data is compiled. The "Load preview" dialog then opens. This dialog displays alarms and recommends actions needed for the loading operation.

4. Check the alarms and select the actions in the "Action" column, if necessary.
   As soon as downloading becomes possible, the "Load" button is enabled.
5. Click the "Load" button.
   The loading operation is performed.

Or:

1. In the project tree, select the project data that you want to download.
2. Select the "Card Reader/USB memory > Write to memory card" command in the "Project" menu.
   The "Select memory card" dialog opens.
3. Select a memory card that is compatible with the CPU.
   A button with a green check mark is activated at the bottom of the dialog.
4. Click the button with the green check mark.
   If necessary, the project data is compiled. The "Load preview" dialog then opens. This dialog displays alarms and recommends actions needed for the loading operation.
5. Check the alarms and select the actions in the "Action" column, if necessary.
   As soon as downloading becomes possible, the "Load" button is enabled.
6. Click the "Load" button.
   The loading operation is performed.

Downloading the user program to a memory card in the CPU (S7-300/400 only)

To download the user program to a memory card in a CPU of the S7-300/400 series, follow the steps below:

1. Select a CPU of the S7-300/400 series in the project tree.
2. Select the "Download user program to memory card" command in the "Online" menu.
   The "Load preview" dialog opens. This dialog displays alarms and recommends actions needed for the loading operation.
3. Check the alarms and select the actions in the "Action" column, if necessary.
   As soon as downloading becomes possible, the "Load" button is enabled.
4. Click the "Load" button.
   The loading operation is performed and the "Load results" dialog is displayed. This dialog displays alarms and suggests possible actions.
5. Check the alarms and select the actions in the "Action" column, if necessary.
6. Click the "Finish" button.

See also

General information on loading (Page 348)

Downloading project data to a device (Page 350)

Uploading project data from a device (Page 354)

8.1.2.4 Uploading project data from a device

The following section describes the general procedure for uploading project data from a device. Which project data you can upload from a device depends on the products installed.

You have the following options for uploading project data from a device to your project:

- Uploading as a new station
  With this option you upload existing project data of a device to your project as a new station.
- Uploading project data of a device
  With this option you upload project data from the device to an existing CPU in the project. You will find the project data that can be loaded in the online help of the product.

In both cases, all instances of library types are connected again with the corresponding version of the type in the project library during uploading. If no suitable type is yet available for an uploaded instance or the correct version of the type does not exist in the project library, the type or the version is added to the project library. You can only upload library types in the state "in test" that are saved on the device to the project, if the released type version on which the test version is based is available in the project library.

Requirement

- A project is open.
- The hardware configuration and software to be uploaded must be compatible with the TIA Portal. If the data on the device was created with a previous program version or with a different configuration software, please make sure they are compatible.

Uploading as a new station

To upload the complete device to your project, follow these steps:

1. Select the project name in the project tree.
2. Select "Upload device as new station (hardware and software)" in the "Online" menu.
   The "Upload device to PG/PC" dialog opens.
3. Select the type of interface you want to use for the uploading operation in the "Type of the PG/PC interface" drop-down list.
4. Select the interface to be used from the "PG/PC interface" drop-down list.
5. Click the "Configure interface" button to the right of the "PG/PC interface" drop-down list to adapt the settings for the selected interface.
   See also: AUTOHOTSPOT
6. Display all compatible devices by selecting the relevant option and clicking the "Start search" command. In the accessible devices table, select the device from which you want to upload project data.
7. Click on "Load".
   Depending on the selected device, a dialog appears in which you have to enter additional information, such as the position of the module rack.
   The project data of the device is uploaded to the project. You can edit it offline and then download it to the device again.

Uploading project data of a device

To upload only project data from one device to your project, follow these steps:

1. Establish an online connection to the device from which you want to download the project data.
   See also: AUTOHOTSPOT
2. Select the device in the project tree.
   The "Upload from device (software)" command in the "Online" menu is enabled.
3. Select the "Upload from device (software)" command in the "Online" menu.
   The "Upload preview" dialog box opens.
4. Check the alarms in the "Upload preview" dialog, and select the necessary actions in the "Action" column.
   As soon as uploading becomes possible, the "Upload from device" button is enabled.
5. Click the "Upload from device" button.
   The loading operation is performed.

Loading of project data from a different project version from a device

When the commands "Upload from device (software)" and "Loading the device as new station (hardware and software)" are executed in the "Online" menu, it is checked whether the project data on the device was created with a different version than the opened version of the TIA Portal.

If the project data was created with a different project version, the "Upload preview" dialog provides you with information on whether all upload requirements are met.

Note the information on the requirements for loading and, if necessary, select a command in the selection menu in the "Action" column.

The "Upload from device" button is enabled as soon as the view has been updated and uploading is possible.

Special features when using data blocks from other product versions

Data blocks that were created with a different product version of the TIA Portal cannot be opened and edited with the current product version. You can find additional information in the section "AUTOHOTSPOT".

See also

General information on loading (Page 348)

Downloading project data to a device (Page 350)

Downloading project data to a memory card (Page 352)

8.1.2.5 Loading project data from a memory card

You have the following options for uploading project data from a memory card to your project:

- Upload project data of the memory card as a new station
  With this option you upload the project data of a memory card to your project as a new station.
- Upload project data of the memory card to an existing device
  With this option you upload project data of a memory card to an existing device in your project. You will find the project data that can be loaded in the online help of the product.

In both cases, all instances of library types are connected again with the corresponding version of the type in the project library during loading. If no suitable type is yet available for a loaded instance or the correct version of the type does not exist in the project library, the type or the version is added to the project library.

Requirement

- A project is open.
- The memory card is displayed.
  See also: Accessing memory cards (Page 422)
- The hardware configuration and software to be uploaded must be compatible with the TIA Portal. If the data on the memory card was created with a previous program version or with different configuration software, please make sure they are compatible.

Uploading project data as a new station

To upload project data from a memory card to your project, follow these steps:

1. In the project tree, select the project data you want to upload.
2. Select "Upload device as new station (hardware and software)" in the "Online" menu.

Or:

1. In the project tree, drag the memory card folder to the project.

Or:

1. Right-click on the memory card.
2. Select "Copy" in the shortcut menu.
3. Right-click on the project.
4. Select the "Paste" command in the shortcut menu.

Uploading project data to an existing device

To upload the project data of a memory card to an existing device, follow these steps:

1. In the project tree, drag the folder of the memory card to a device in the project or copy the memory card and paste the data into a device.
   The "Upload preview" dialog box opens.
2. Check the alarms in the "Upload preview" dialog, and select the necessary actions in the "Action" column.
   As soon as uploading becomes possible, the "Upload from device" button is enabled.
3. Click the "Upload from device" button.
   The loading operation is performed.

8.2 Comparing project data

8.2.1 Basics of project data comparison

Function

You can compare project data of the same type in order to determine possible differences. In principle, the following comparison methods are available:

- Online/offline comparison
  With this comparison method you can compare the software of objects of a device with the objects of a project. This is only possible when you establish an online connection to the device.
- Offline/offline comparison
  With this comparison method you can either compare the software or the hardware. When you compare the software, you can compare objects from projects or libraries. The hardware comparison is available for devices from the currently open project or from reference projects. You can decide for the software as well as the hardware comparison whether the comparison should be performed automatically for all objects or whether you want to compare individual objects manually.
- Detailed comparison
  For some objects, for example, blocks, you can also perform a detailed comparison in addition to the online/offline and offline/offline comparison. This involves opening the blocks to be compared beside each other and highlighting the differences.

A simple online/offline comparison is performed as soon as you establish an online connection. During this process, comparable objects in the project tree are marked with icons that represent the result of the comparison. You can also run a more comprehensive online/offline and offline/offline comparison in the compare editor. When you compare software, you can also select actions for non-identical objects in the comparison.

Note

- Not all objects allow all types of comparison. Which comparison method you can use for which project data depends on the products installed.
- Compile your user program before you start a comparison or detailed comparison. After each change of the program during a comparison, repeat this step before you update the result of the comparison. This ensures that the comparison shows the current status.
- When monitoring a module, note that a difference of the time stamp can also exist with an online/offline comparison if the block is recompiled with code change. The checksum remains unchanged. In this case download the block to the device and restart the monitoring.

Comparison using checksums

The online/offline and offline/offline comparison of software objects uses checksums that are generated for certain data of the objects. The objects are only the same if their checksums are identical.

Note

Time stamps continue to be used for online/offline comparison for CPUs of the S7-300/400 series.

The data of an object are divided into the following two categories:

- Source data
  Source data are all the data of an object which you can directly influence, such as the name of the object, source code, comments or programming language. The properties that you can edit here may differ depending on the object.
- Target data
  All compiler and runtime data are the target data. These data may have no direct influence, since they are produced by the system based on their source data. Once again here, the characteristics differ depending on the object.

The following table provides an overview of the areas of the property comparison:

Category	Area	Description
Source data	Safety	Checksum for safety objects You can find additional information about the comparison of safety objects in the "Comparing safety programs" section of the online help for SIMATIC Safety.
	Interface without comments	Checksum of all the tags of the block interface of the first layer. Comments of the tags are not part of the checksum.
	Code without comments	Checksum of the source code. Block and network comments are not included in the checksum.
	Comments (multi-language)	Checksum of all block comments and network comments
	Language configuration	Online/offline comparison: Displays the languages that are loaded to the device during the loading process. Offline/offline comparison: Displays the enabled languages in the project.
	Event	Checksum of all event-relevant data of an organization block (OB).
	Properties	Checksum of all configurable properties of an object.
	Configuration of technology objects	Checksum of the technology-specific settings of a technology object.
	Alarm configuration	Checksum on all alarm-specific settings of an object.
	Supervision configuration	Checksum of all supervision settings of an object.
Target data	Compilation and runtime data	Checksum of data for compiling and relevant for loading.
	Time stamp	Time stamps for S7-300/400 CPUs that are created by the system for compiling and downloading. Only available when an online connection is available.

Note

- Note that the object determines which checksums are actually used for the comparison.
- The value settings of technology objects are also transferred to the checksum of the block interface.
- For know-how protected blocks that were created in an older version, the checksums "Code without comments" and "Comments (multi-language)" are not displayed under "Source data".

See also

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Running a detailed comparison (Page 371)

Using the compare editor (Page 362)

8.2.2 Carrying out an online/offline comparison

The online/offline comparison lets you compare objects of a device with objects of a project.

Requirement

The project tree is open.

Procedure

To perform an online/offline comparison, follow these steps:

1. Select a device in the project tree that allows online/offline comparison.
2. Select the "Compare > Offline/online" command in the shortcut menu.
3. If you have not already established an online connection to this device, the "Go online" dialog opens. In this case, set all the necessary parameters for the connection and click "Connect".
   The online connection is established and compare editor opens.

Result

All objects that exist online and offline are displayed. The symbols in the compare editor and in the project tree show you the status of the objects. In the compare editor, you can now define certain actions for the objects, depending on their status.

See also

Basics of project data comparison (Page 358)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Running a detailed comparison (Page 371)

8.2.3 Carrying out offline/offline comparisons

With offline/offline comparison you have the option to compare the project data of two devices. You can carry out a software as well as a hardware comparison. When you compare the software, you can compare objects from projects or libraries. The hardware comparison is available for devices from the currently open project or from reference projects. You can decide whether the comparison should be performed automatically for all objects or whether you want to compare individual objects manually. For a software comparison, the following options are available as well:

- For objects that are not identical, you can specify actions to eliminate the differences.
- You can specify the criteria that should be used for the comparison.

You can drag any other device to the drop area at any time to perform further comparisons.

Requirement

The project tree is open.

Performing an offline/offline comparison for software

To perform an offline/offline comparison for software, follow these steps:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Drag-and-drop an additional device to the drop area of the right pane.
   All existing objects of the selected devices are displayed depending on the settings of the compare editor in the "Software" tab and an automatic comparison is carried out. You can identify the status of the objects based on the symbols in the compare editor. If needed, specify an action for non-identical objects to eliminate the differences.
4. If needed, define the comparison criteria.
5. Select an object to display the details of the property comparison.
6. If you want to carry out a manual comparison, click on the button for switching between automatic and manual comparison in the status and action area. Then select the objects that you want to compare.
   The properties comparison is displayed. You can identify the status of the objects based on the symbols.

Performing an offline/offline comparison for hardware

To perform an offline/offline comparison for hardware, follow these steps:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Drag-and-drop an additional device to the drop area of the right pane.
4. Open the "Hardware" tab.
5. If you want to carry out a manual comparison, click on the button for switching between automatic and manual comparison in the status and action area. Then select the objects that you want to compare.
   The properties comparison is displayed. You can identify the status of the objects based on the symbols.

See also

Basics of project data comparison (Page 358)

Carrying out an online/offline comparison (Page 360)

Using the compare editor (Page 362)

Running a detailed comparison (Page 371)

8.2.4 Using the compare editor

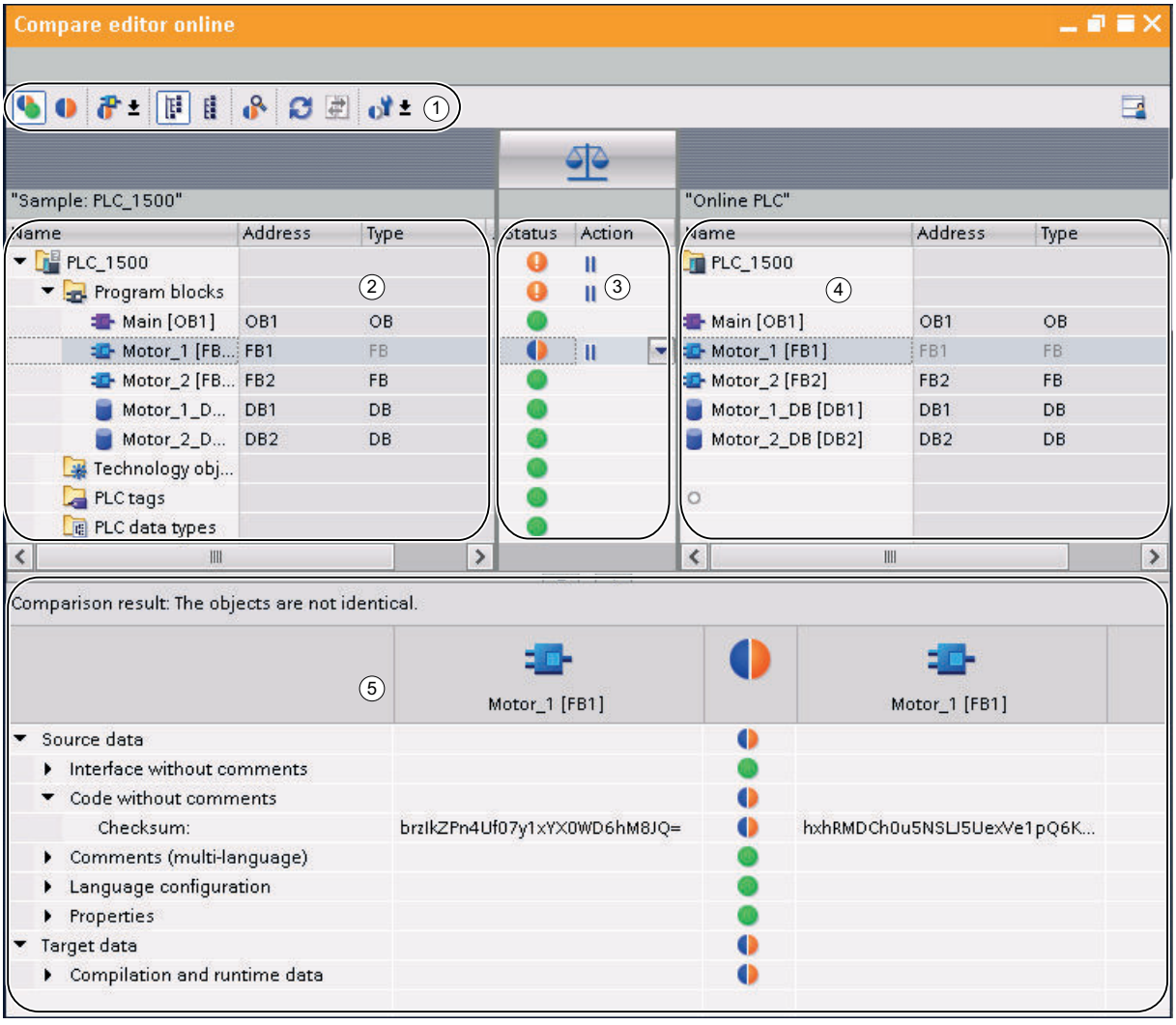
8.2.4.1 Overview of the compare editor

Function

The compare editor gives an overview of the results of a comparison in a table. The appearance varies slightly depending on whether it is an online/offline comparison or a hardware/software comparison.

Layout of the compare editor

The following figure shows the layout of the compare editor for an online/offline comparison:



① Compare editor toolbar

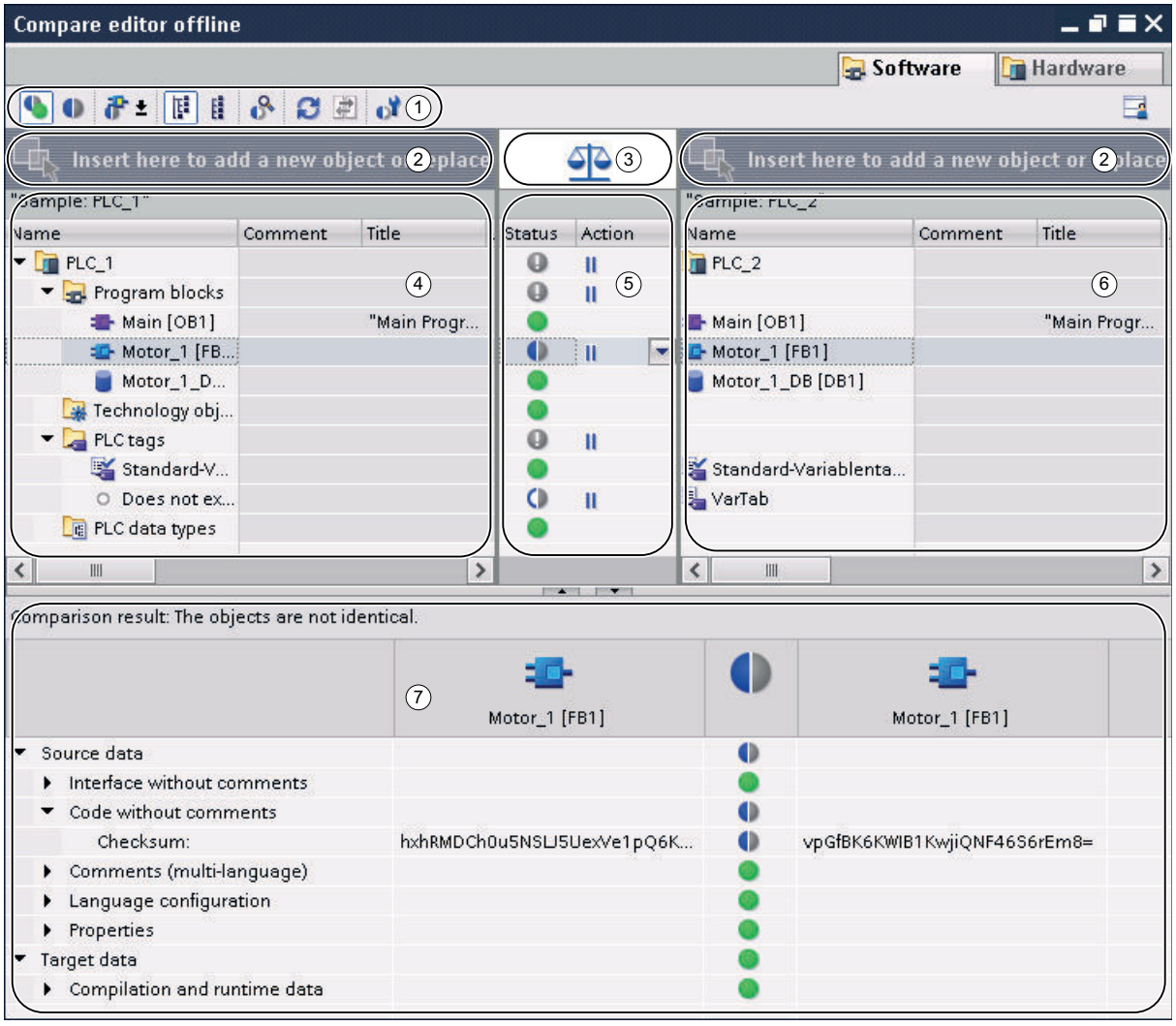
② Left comparison table

③ Status and action area

④ Right comparison table

⑤ Property comparison

The following figure shows the layout of the compare editor for an offline/offline comparison (software):



① Compare editor toolbar

② Drop areas

③ Button to toggle between automatic and manual comparison

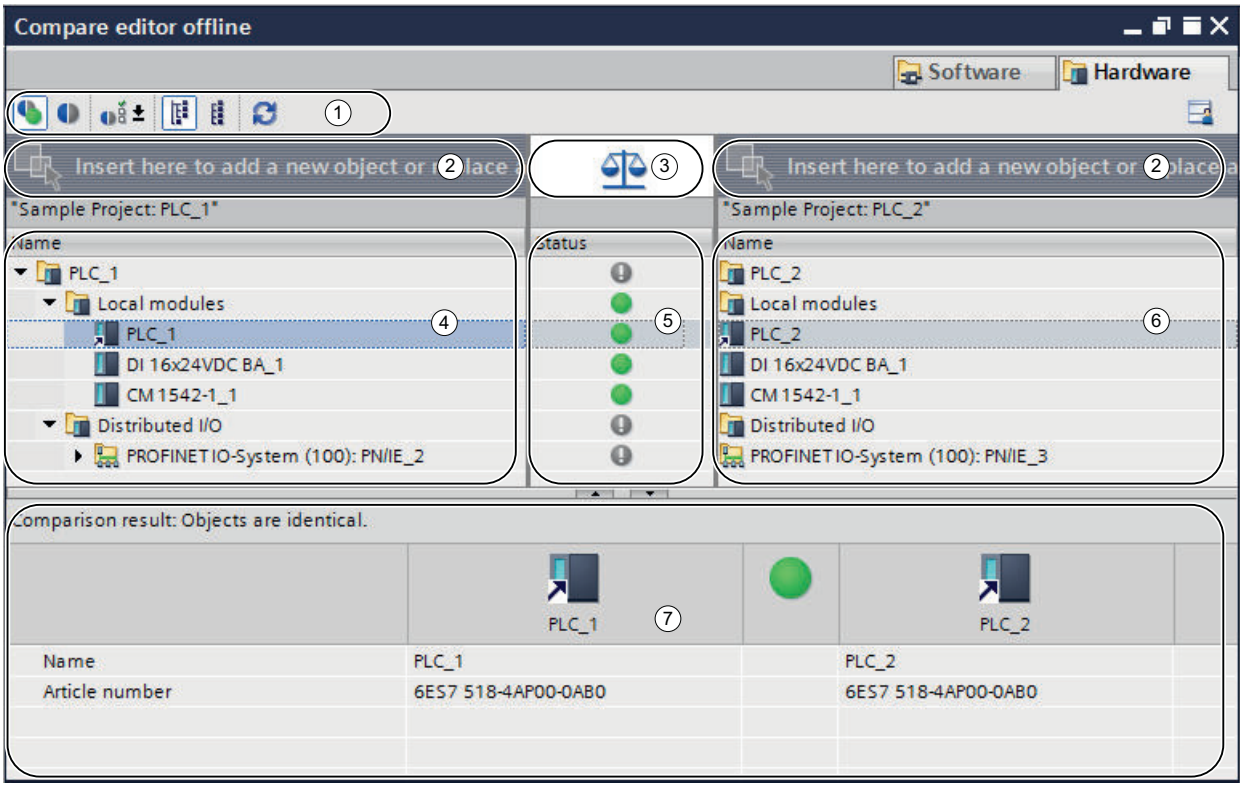
④ Left comparison table

⑤ Status and action area

⑥ Right comparison table

⑦ Property comparison

The following figure shows the layout of the compare editor for an offline/offline comparison (hardware):



① Compare editor toolbar

② Drop areas

③ Button to toggle between automatic and manual comparison

④ Left comparison table

⑤ Status area

⑥ Right comparison table

⑦ Property comparison

Compare editor toolbar

With the toolbar, you can access the following compare editor functions:

- Show identical and different objects
  You can show identical objects if you want to view the comparison in full.
- Show only objects with differences
  You can hide identical objects to make the comparison easier to follow.
- Show additional filters available (only online/offline comparison and offline/offline comparison for software)
  You can define which objects are to be compared.
- Display available assignment criteria (only offline/offline comparison for hardware)
  You can specify the criterion according to which the modules are to be assigned to each other for the comparison.

- Changing the view
  You can choose between a hierarchical and a flat view. In the hierarchical view, the devices are shown in their structure; in the flat view, the objects of the devices are listed without structure.
- Start detail comparison (only online/offline comparison and offline/offline comparison for software)
  You can start a detailed comparison for objects to show the individual differences. This function is, however, not available for every object.
- Refresh the view
  After you have modified objects, you can update the comparison results using this function.
- Execute actions (only online/offline comparison and offline/offline comparison for software)
  You can synchronize non-identical objects using specific actions.
- Set comparison criteria (only online/offline comparison and offline/offline comparison for software)
  You can specify which criteria are to be used in the comparison. Disabled criteria do not affect the overall result of the comparison.

Drop areas

In the case of an offline/offline comparison, you can drag the devices you want to compare into the drop areas. In the case of a software comparison, the devices to be compared can originate from the opened project, from reference projects, from the project library or from global libraries. However, note that you can only drop complete libraries into the right drop area. In the case of a hardware comparison, you can compare devices from the opened project or from reference projects.

Button to toggle between automatic and manual comparison

With an offline/offline comparison, you can switch between automatic and manual comparison. In the case of automatic comparison, the objects to be compared are assigned automatically to each other. In the case of a manual comparison, you can select the objects that are to be compared.

Comparison tables

Comparison tables show the objects of the devices being compared to one another.

The following table shows the meaning of the columns of the comparison table:

Column	Description
Name	Name of the compare object
Comment	Comment on the compare object
Title	Title of the compare object
Address	Address of the compare object
Numbering	Type of numbering for the comparison object
Type	Type of compare object
Language	Programming language set for the compare object.

Column	Description
Time stamp interface	Time of the last modification to the block interface
Time stamp code	Time of the last modification to the source code
Author	Name of the author of the compare object
Version	Version of the compare object
Family	Name of the object family
Load memory	Memory usage of the load memory of the compare object
Work memory	Memory usage of the work memory of the compare object
Modified on	Time of last modification
Optimized block access	Indicates whether "Optimized block access" is enabled for a block.
Signature	Signature of the comparison object (SIMATIC Safety*)
Interface signature	Signature of the block interface of the comparison object (SIMATIC Safety *)

* You can find additional information about the comparison of safety objects in the "Comparing safety programs" section of the online help for SIMATIC Safety.

Not all columns are available in every comparison type. For the hardware comparison, for example, the comparison tables contain only the "Name" column.

Not all columns are shown in the default setting. However, as in all table editors, you can show or hide the columns as required and sort according to individual columns.

Status and action area

The status and action area offers the following options:

- You can view the results of automatic comparison. The results are displayed with symbols.
- In an online/offline comparison and an offline/offline comparison for software, you can specify actions for non-identical objects.

Status and action symbols

The following table shows the comparison results symbols for an online/offline comparison:

Symbol	Description
	Folder contains objects whose online and offline versions differ
	Result of comparison is either not known or cannot be displayed for the following reasons: • No rights available for access to a protected CPU. • The loading process on the CPU was executed with a TIA Portal version earlier than V14.
	Online and offline versions of the object are identical
	Online and offline versions of the object are different
	Object only exists offline

Symbol	Description
	Object only exists online
	This comparison criterion is disabled and the associated checksum is not incorporated into the comparison result.

The following table shows the comparison results symbols for an offline/offline comparison:

Symbol	Description
	Reference program
	Version compared
	Folder contains objects of which the versions compared differ
	Results of the offline/offline comparison are not known
	The versions of the object compared are identical
	The versions of the object compared differ
	Object only exists in the reference program
	Object only exists in the version compared
	Hardware comparison only: Although the lower-level objects of the container are identical, there are differences between the containers themselves. Such a container may be a rack, for example.
	Hardware comparison only: The lower-level objects of the container are different. There are also differences between the containers. Such a container may be a rack, for example.
	This comparison criterion is disabled and the associated checksum is not incorporated into the comparison result.

The following table shows the symbols for possible actions in a software comparison:

Symbol	Description
	No action
	Overwrite the object of the compared version with the object from the reference program
	Overwrite the object of the output program with the object from the compared version
	Different actions for the compare objects in the folder

Property comparison

The property comparison compares the properties of the selected compare objects. The result is displayed with symbols. Only the property comparison is made with a manual comparison so that the status and action area remains empty. With automatic comparison, you can perform the property comparison in addition to the comparison in the comparison tables.

See also

Basics of project data comparison (Page 358)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Show and hide table columns (Page 369)

Filtering the compare editor view (Page 370)

Running a detailed comparison (Page 371)

Updating the comparison results (Page 372)

Set comparison criteria (Page 373)

Synchronizing non-identical objects (Page 374)

Changing the view (Page 376)

8.2.4.2 Show and hide table columns

In a software comparison, you can show or hide the columns of comparison tables as required.

Procedure

To show or hide table columns, follow these steps:

1. Click a column header.
2. Select the "Show/Hide" command in the shortcut menu.
   The selection of available columns is displayed.
3. To show a column, select the column's check box.
4. To hide a column, clear the column's check box.
5. In order to show or hide multiple columns, click "More" and select or clear the check boxes of the corresponding columns in the "Show/Hide" dialog.

Result

The columns in both the left and the right comparison table are shown or hidden.

See also

Basics of project data comparison (Page 358)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Overview of the compare editor (Page 362)

Filtering the compare editor view (Page 370)

Running a detailed comparison (Page 371)

Updating the comparison results (Page 372)

Set comparison criteria (Page 373)

Synchronizing non-identical objects (Page 374)

Changing the view (Page 376)

8.2.4.3 Filtering the compare editor view

You can improve the clarity of the compare editor using the following filters:

- Hiding identical comparison objects
  You can hide comparison objects which have identical online/offline or offline/offline versions. Any such comparison objects you have hidden can also be shown again at any time.
- Selecting displayed objects
  In an online/offline comparison or an offline/offline comparison for software, you can specify the objects for which the comparison results are to be shown.

Requirement

The compare editor is open.

Hiding identical comparison objects

To hide identical objects, follow these steps:

1. Click on the "Show only objects with differences" button in the toolbar.
   Only the elements that differ online and offline are displayed.

Showing identical comparison objects

To show identical objects again, follow these steps:

1. Click on the "Show identical and different objects" button in the toolbar.
   All elements will be displayed.

Selecting displayed objects

To select the objects for which comparison results should be displayed, follow these steps:

1. Perform an online/offline or an offline/offline comparison for software.
2. Click the arrow of the button "Show additional filters available" in the toolbar.
3. Select the required filter.

See also

Basics of project data comparison (Page 358)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Overview of the compare editor (Page 362)
Show and hide table columns (Page 369)
Running a detailed comparison (Page 371)
Updating the comparison results (Page 372)
Set comparison criteria (Page 373)
Synchronizing non-identical objects (Page 374)
Changing the view (Page 376)

8.2.4.4 Running a detailed comparison

Note

Please note the following:

- Not all objects allow a detailed comparison. The project data for which you can perform a detailed comparison depends on the products installed. A detailed comparison of the hardware components is not available for hardware comparison.
- For blocks, PLC tags and PLC data types, you can start the detail comparison directly from the project tree. You can find additional information in the section "Comparing PLC programs".

Procedure

Proceed as follows to perform a detailed comparison:

1. First, perform an online/offline or an offline/offline comparison for software.
   The compare editor opens.

   Note

   You can only perform a detailed comparison for objects that are listed in the left as well as the right comparison table.

2. In the compare editor, select the object for which you want to perform a detailed comparison.
3. Click the "Start detailed comparison" button in the toolbar.

Result

The object and its associated comparison partners are opened in their appropriate editors. The editors are arranged next to each other to provide you with a quick overview of the differences. You can find more information with the description of the respective data type.

See also

Basics of project data comparison (Page 358)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)
Overview of the compare editor (Page 362)
Show and hide table columns (Page 369)
Filtering the compare editor view (Page 370)
Updating the comparison results (Page 372)
Set comparison criteria (Page 373)
Synchronizing non-identical objects (Page 374)
Changing the view (Page 376)

8.2.4.5 Updating the comparison results

As soon as you change an object, the comparison results are no longer valid and must be updated.

Note

For online/offline comparisons, you should note that changes in the device may result in the system automatically updating the comparison editor if objects in the comparison are affected by the change. This can have the following results:

- Some of the actions you have defined may become invalid, for example if the device no longer contains the object in question. Objects with such invalid actions will be highlighted so you can define new, valid actions.
- The selection you made before the automatic update may also be cancelled.

Requirement

The comparison editor is open.

Procedure

To update the comparison results, follow these steps:

1. Click the "Refresh view" button in the toolbar.
   The comparison results are updated.

   Note

   Please note that the "Refresh view" button will not be available while the comparison editor is loading or synchronizing content.

See also

Basics of project data comparison (Page 358)
Carrying out an online/offline comparison (Page 360)
Carrying out offline/offline comparisons (Page 361)

Overview of the compare editor (Page 362)
Show and hide table columns (Page 369)
Filtering the compare editor view (Page 370)
Running a detailed comparison (Page 371)
Set comparison criteria (Page 373)
Synchronizing non-identical objects (Page 374)
Changing the view (Page 376)

8.2.4.6 Set comparison criteria

You have the option to set to criteria to be taken into account for a comparison. The specified criteria are applied to all objects of a comparison, if they are available for an object. If a criterion for an object is not available, for example, "Code without comments" for data blocks, the comparison result for this checksum is considered to "match".

For a criterion that you disabled, you can still view the checksums for the overall comparison result although it is not used. Disabled comparison criteria for the property comparison are indicated by the filter icon. The filter symbol can also be displayed in the upper area of the comparison editor when no checksums should be used in the comparison result based on the selection of comparison criteria.

Note

With an online/offline comparison, the comparison criteria have no influence on the result of comparison in the project tree. Note that there may be different results in the project tree and comparison editor.

Procedure

To define the criteria to be taken into account for the comparison, follow these steps:

1. In the toolbar, click on the arrow of the "Display further comparison criteria" button.
2. Select all the criteria that should be considered for the comparison.
3. Clear all the criteria that should be considered for the comparison.
4. Click the button with the green check mark at the bottom of the drop-down list.
5. Click on the button with the x to close the drop-down list without applying the changes.

Result

Only the enabled criteria are taken into account for the comparison.

See also

Basics of project data comparison (Page 358)
Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)
Overview of the compare editor (Page 362)
Show and hide table columns (Page 369)
Filtering the compare editor view (Page 370)
Running a detailed comparison (Page 371)
Updating the comparison results (Page 372)
Synchronizing non-identical objects (Page 374)
Changing the view (Page 376)

8.2.4.7 Synchronizing non-identical objects

Specifying actions

If you have performed a comparison, you can specify the actions to be performed for non-identical objects in the compare editor. You cannot select any actions for identical objects. Note that you cannot execute any actions during hardware comparison.

In the case of an online/offline comparison, only synchronization actions in one direction are permitted, in order to retain program consistency. Thus, for example, you can load multiple blocks to a device or from a device, but you cannot perform a combination of loading actions in one synchronization action. In this case, the first action you set in the compare editor determines the synchronization direction. For example, if you specify for a block that the offline block is to be loaded to the device, then the other objects can also only be loaded to the device via a synchronization action. To load objects from the device again, first select the "No action" option. You can then specify the action settings again as required. Or, you can perform a new comparison.

Note

Note the following CPU-specific aspects when defining actions:

- S7-300/400: You can define actions for the "Program blocks" folder, for folders you have created yourself or for individual blocks.
- S7-1200/1500: You can define actions for the "Program blocks" folder, for folders you have created yourself or for individual blocks. If you have performed an online/offline comparison and select download to the device as action, a consistent download is executed. If you upload the object from the device to the project, however, you can also upload individual blocks.

Requirement

The compare editor is open.

Procedure

To select an action for a non-identical object, follow these steps:

1. In the status and action area, click in the "Action" column on the cell of the object for which you want to define an action.
   The cell changes to a drop-down list.
2. Click on the drop-down list.
3. Select the action you want.
   The action set will be carried out for the object in question the next time synchronization is performed.
   If you have accidentally changed the action you had selected, you can undo the change before the next synchronization.
4. To restore the previously set action selection, right-click the action in the status and action area that you want to restore.
5. Select the "Restore last selection" command in the shortcut menu.

See also

Basics of project data comparison (Page 358)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Overview of the compare editor (Page 362)

Show and hide table columns (Page 369)

Filtering the compare editor view (Page 370)

Updating the comparison results (Page 372)

Synchronizing objects (Page 375)

Synchronizing objects

Synchronization executes the actions you have specified for non-identical objects. Note, however, that in the case of an online/offline comparison you can only perform actions in one direction in one synchronization action.

Requirement

- The compare editor is open.
- The desired actions have been selected.

Procedure

To synchronize objects, follow these steps:

1. Click the "Execute actions" button in the toolbar.

Result

The actions you specified for the objects are performed.

See also

Basics of project data comparison (Page 358)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Overview of the compare editor (Page 362)

Show and hide table columns (Page 369)

Filtering the compare editor view (Page 370)

Updating the comparison results (Page 372)

Specifying actions (Page 374)

8.2.4.8 Changing the view

You can choose between a hierarchical and a flat view for the left comparison table. In the hierarchical view, the devices are shown in their structure; in the flat view, the objects of the devices are listed without structure. In the right comparison table, the objects are always displayed flat.

Setting the hierarchical view

To set the hierarchical view, follow these steps:

1. Click the "Display in hierarchical view" button in the toolbar of the compare editor.

Setting the flat view

To set the flat view, follow these steps:

1. Click the "Display in flat view" button in the toolbar of the compare editor.

See also

Basics of project data comparison (Page 358)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Overview of the compare editor (Page 362)

Show and hide table columns (Page 369)

Filtering the compare editor view (Page 370)

Running a detailed comparison (Page 371)

Updating the comparison results (Page 372)

Set comparison criteria (Page 373)
Synchronizing non-identical objects (Page 374)

8.3 Protecting project data

8.3.1 Protection concept for project data

Introduction

You can protect your project data from unauthorized access. These include, for example:

- Access protection for devices
- Copy and display protection of objects
- Restrictions for printouts of know-how-protected objects

For objects with know-how protection, this protection is also retained after the object is pasted into a library. Note that every protection mechanism is not available for all objects. How to protect specific objects is described in the online help of the product.

Companies can also specify the use of a password provider to protect blocks. To do so, the central settings must be adjusted and all users must integrate the corresponding password providers in the TIA Portal. The passwords provided by a password provider can subsequently be used. You can find additional information in the section "Protecting blocks".

Revoking access rights for devices

If you want to execute a function that is password-protected by means of the device protection level, you are prompted to enter a password. When the password is entered correctly, you can execute the required function. The access right is retained on the device until you close the TIA Portal.

If you want to reactivate password protection while the TIA Portal is open, you can explicitly revoke the access rights for a device. As a result, certain functions for the protected device cannot be executed until the correct password is entered again. You specify the functions for which a password must be entered when you assign the device protection level.

See also

Printing project data (Page 397)

8.3.2 Revoking access rights for devices

Requirement

- A protection level has been set for the device.
- A protected function for the device has been enabled by entering the password.

Procedure

To revoke the access rights for the device, follow these steps:

1. Select the device for which you want to revoke access rights in the project tree.
2. Select the "Delete access rights" command in the "Online" menu.

Result

The access rights are revoked, and starting from now the user will be prompted to enter the password again to execute a password-protected function on the device. The function can only be executed if the correct password is entered.

If the device has an online connection, it will be disconnected.

See also

Protection concept for project data (Page 378)

8.4 Printing project contents

8.4.1 Printing project documentation

8.4.1.1 Documentation settings

Introduction

Once a project is created, the contents can be printed in an easy-to-read format. You may print the entire project or individual objects within the project. A well-structured printout is helpful when editing the project or performing service work. The printout can also be used for your customer presentations or as full system documentation.

You can prepare the project in the form of standardized circuit manuals and print it in a uniform layout. You can limit the scope of the printout. You have the option to print to the entire project, individual objects along with their properties, or a compact overview of the project. In addition, you can print the contents of an open editor.

Improving the printout with frames and cover pages

You can design the appearance of the printed pages according to your own requirements, for example, to add your own company logo or the corporate design of your company in the project documentation. You can create any number of design variants as frames and cover pages. The frames and cover pages are stored in the project tree under the item "Documentation settings" and are part of the project. You can insert placeholders for data from previously entered document information within the frames and cover pages. These will be filled automatically with the appropriate metadata during printing.

If you want to avoid designing your own template, there are ready-made frames and covers pages available. These include templates complying with the ISO standard for technical documentation.

Modular structure of a printout

An printout generally consists of the following components:

- Cover page (only when printing from the project tree)
- Table of contents (only when printing from the project tree)
- Name and path of an object within the project tree
- Object data

Printout of the cover page or the table of contents can be deactivated in the "Print" dialog.

See also

Creating frames (Page 386)

Creating a cover page (Page 386)

Editing cover pages and frames (Page 388)
Entering document information (Page 384)
Print function for module labels (Page 399)

8.4.1.2 Printout of project contents

Availability of print function

The following contents can be printed:

- An entire project in the project tree
- One or more project-related objects in the project tree
- Contents of an editor
- Tables
- Libraries
- Diagnostics view of the Inspector window

It is not possible to print in the following areas:

- Portal view
- Detailed view
- Overview window
- Compare editor
- All tabs of the Inspector window, except the diagnostics view
- All task cards, except the libraries
- Most of the dialogs
- Properties and devices of the programming device/PC not related to the project, for example online portals and connected card readers.

Scope of printout

To be able to print, at least one printable element has to be selected.

If a selected object is printed, all subordinate objects are also printed. For example, if a device is selected in the project tree, all of its data is also printed. If you select the entire project in the project tree for printing, all project contents are printed with the exception of the graphical views. These have to be printed separately. Items in the project tree that are not part of the project cannot be printed. For example, this includes online portals and connected card readers and USB memory devices.

When table contents are printed, all lines in the table in which a cell is selected are printed. In order to print one or more table columns, the desired columns must be selected. If no individual cells or columns are selected, the entire table is printed.

Limitations when printing

In general, it is possible to print all objects that can be visualized on the user interface. Conversely, this means that you cannot print objects that you do not have access to. If a printout fails, possible reasons may include the following:

- A valid license does not exist for displaying an object.
- There is no device description for an object.
- A software component needed to display an object is not installed.

See also

Printing project data (Page 397)

8.4.1.3 Changing the print settings

Changing the print settings

You can specify general print settings that are retained even after the TIA Portal is closed and re-opened. Some settings are dependent on the products installed. The following settings are possible in every case:

Always print table data as pairs of values

If this option is selected, tables are not printed in tabular format but rather as a pairs of key and value.

Example:

Object name	Property 1	Property 2
Object A	Value A1	Value A2
Object B	Value B1	Value B2

In this case, the printout has the following appearance:

Object A

Property 1: Value A1

Property 2: Value A2

Object B

Property 1: Value B1

Property 2: Value B2

Printing mask editors

- Always print data in tables
  All parameters of technology objects are printed in tabular format.
- Print mask graphics if possible
  If the utilized editor supports this function, the contents of the editor are not printed as a table but rather as a complete graphic as it appears on the screen.

Procedure

To change the print settings, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General" group.
3. Select the desired default settings in the "Print settings" area.
   The changes are applied immediately and are retained for all projects, even after the TIA Portal is closed.

See also

Overview of the print settings (Page 233)

8.4.1.4 Specifying the print layout

Specifying the print layout

If you do not want to rely on ready-made print templates, you can specify your own cover page or your own layout for the individual pages. Your designs are saved together with the respective project.

Your designs for the cover page and your templates for the page layout can be found in the project tree under the "Documentation information" group. You will also find metadata on the project there under the entry "Document information". For subsequent print operations, you can customize the appearance of the printout in the "Print" dialog using the saved cover pages and page layout templates and the available metadata.

Designing the cover page

The cover page can be customized. You can insert a background graphic and provide placeholders for text on the page. The placeholders are automatically filled with data from a documentation information during printing.

Cover pages are located in the project tree under the "Documentation information > Cover pages" group.

Designing the content page

The regular pages of a printout can contain the following elements:

- Frame with static content, such as a company logo
- Placeholders for text, such as the name of the project, the page number, and the time the printout was started
  Several different values for the individual placeholders can be specified in the document information. Other values, such as the project name, are preassigned and are inserted automatically during printing.
- Footnote
  The footnote is always output below the content area.
- Content area
  You can specify an area where the printed content is to be embedded.

The design of the content pages is saved in Frames. The individual frames are located in the project tree under the "Documentation information > Frames" group.

8.4.1.5 Entering document information

You can enter metadata in the document information for every project. In addition, a print frame and a cover page are specified in the document information. You can create different information, if required, to enable you to quickly switch between different document information containing different information, frames, cover pages, page sizes, and page orientations when printing. For example, this is useful if you want to generate printouts in different languages and different document information is provided for each language.

In the documentation editor, you can specify placeholders on the cover page or in the frame of the regular pages. These placeholders can be automatically replaced with metadata from the documentation information during printing.

The various document information are therefore part of the printing function and specify the print layout and print content.

Procedure

To add metadata, follow these steps:

1. To create new document information, double-click "Add new document information" under "Documentation information > Document information" in the project tree.
   The new document information is created and opened immediately.
2. Enter a name for the set in the "Name" field.
3. Fill in the individual fields with the metadata for the project.

8.4.1.6 Managing cover pages and frames

Using cover pages and frames

Uses for cover pages

You can give your plant documentation printouts a professional appearance by adding a cover page. You can design your own cover page or use ready-made cover pages. Ready-made cover pages can be adapted and stored again as a template.

Cover pages can be saved in global libraries where they are available for use across projects.

Cover pages are designed for use as a right printed page only.

Uses of frames

You can embed the regular pages of your plant documentation inside a consistently uniform page frame. The frame can contain placeholders for project metadata, which is stored in the document information. It can also contain graphic elements that you design yourself.

You can create your own frames or rely on ready-made page frames. You can adapt a ready-made page frame and then store it again as a new frame.

Like cover pages, frames can be saved in global libraries where they are available for use across projects.

Frames are designed for use on right printed pages only.

Cover pages and templates in the project tree

Cover pages and frames associated with the project are stored in the project tree under the entry "Documentation information". There are separate folders here for frames and cover pages.

The following actions are available in the project tree for cover pages and frames.

- Creating your own subfolders
- Copying and pasting
- Inserting cover pages and frames from the "Documentation templates" system library
- Copying cover pages and templates to a global library

Cover pages and templates in libraries

The "Documentation templates" system library contains a few cover pages and templates that are available in every project. The cover pages and templates can be moved from there to the project tree using a drag-and-drop operation. You can then adapt the cover pages and templates in the project tree according to the requirements of your project.

Cover pages and templates can be moved from the project tree to a global library. Afterwards, these are available in every project.

See also

Library basics (Page 439)

Overview of the "Libraries" task card (Page 443)

Using ready-made frames and cover pages (Page 387)

Designing cover pages and frames (Page 388)

Creating frames

You can create any number of frames for each project. The frames are stored in the project tree below the "Documentation information > Frames" group. You can assign a frame to all document information. When you select document information for printing, its associated frame is used.

Procedure

To create a new frame, follow these steps:

1. Double-click the entry "Add new frame" below the "Documentation information > Frames" group in the project tree.
   The "Creating frames" dialog opens.
2. Enter a name for the frame in the "Name" field.
3. Choose the paper size from the "Paper type" drop-down list.
4. Choose whether the page is to be created in portrait or landscape format in the "Orientation" drop-down list.

Click the "Add" button.

Result

A new frame is created. The frame is then opened automatically in the documentation editor where it can be edited.

See also

Editing cover pages and frames (Page 388)

Creating a cover page (Page 386)

Creating a cover page

You can create any number of cover pages for the printout for each project. The cover pages are stored in the project tree below the the "Documentation information > Cover pages" group. You can assign a cover page to all document information. When you select specific document information for printing, its associated cover page is used.

Procedure

To create a new cover page, follow these steps:

1. Double-click the entry "Add new cover page" below the "Documentation information > Cover pages" group in the project tree.
   The "Add new cover page" dialog box opens.
2. Enter a name for the cover page in the "Name" field.
3. Choose the paper size from the "Paper type" drop-down list.
4. Choose whether the page is to be created in portrait or landscape format in the "Orientation" drop-down list.

Click the "Add" button.

Result

A new cover page is created. The cover page is then opened automatically in the documentation editor where it can be edited.

See also

Editing cover pages and frames (Page 388)

Creating frames (Page 386)

Using ready-made frames and cover pages

The TIA Portal comes with some ready-made frames and cover pages. These can change according to your wishes.

Procedure

To create and edit the ready-made frames and cover pages, follow these steps:

1. Open the "Global libraries" pane in the "Libraries" task card.
2. In the "Templates" folder, open the "Cover Pages" or "Frames" folder.
3. Drag a cover page or a frame from one of the folders into the project tree and drop it into one of the following folders:
   - For frames: "Document information > Frames"
   - For cover pages: "Document information > Cover pages".

   The ready-made frame or cover page can now be used in the project.
4. Double-click on the new entry in the project tree click to edit the frame or the cover page.

See also

Using cover pages and frames (Page 385)

Editing cover pages and frames (Page 388)

8.4.1.7 Designing cover pages and frames

Editing cover pages and frames

The documentation editor is a graphical editor which allows you to design frames and cover pages for your plant documentation. You can place images or text elements on the frame and the cover pages in the document editor. The text elements are either static or they are automatically filled during printing with the data from the document information that you have selected in the print dialog.

Procedure

To edit a cover page or a frame in the documentation editor, follow these steps:

1. In the project tree, double-click on the entry for an existing cover page or frame under the "Documentation information > Frames " or "Documentation information > Cover pages" group.
   The documentation editor opens.
2. Design the cover page or frame as desired.
3. Close the documentation editor.
   The changes to the cover page or frame are applied automatically.

See also

Creating a cover page (Page 386)

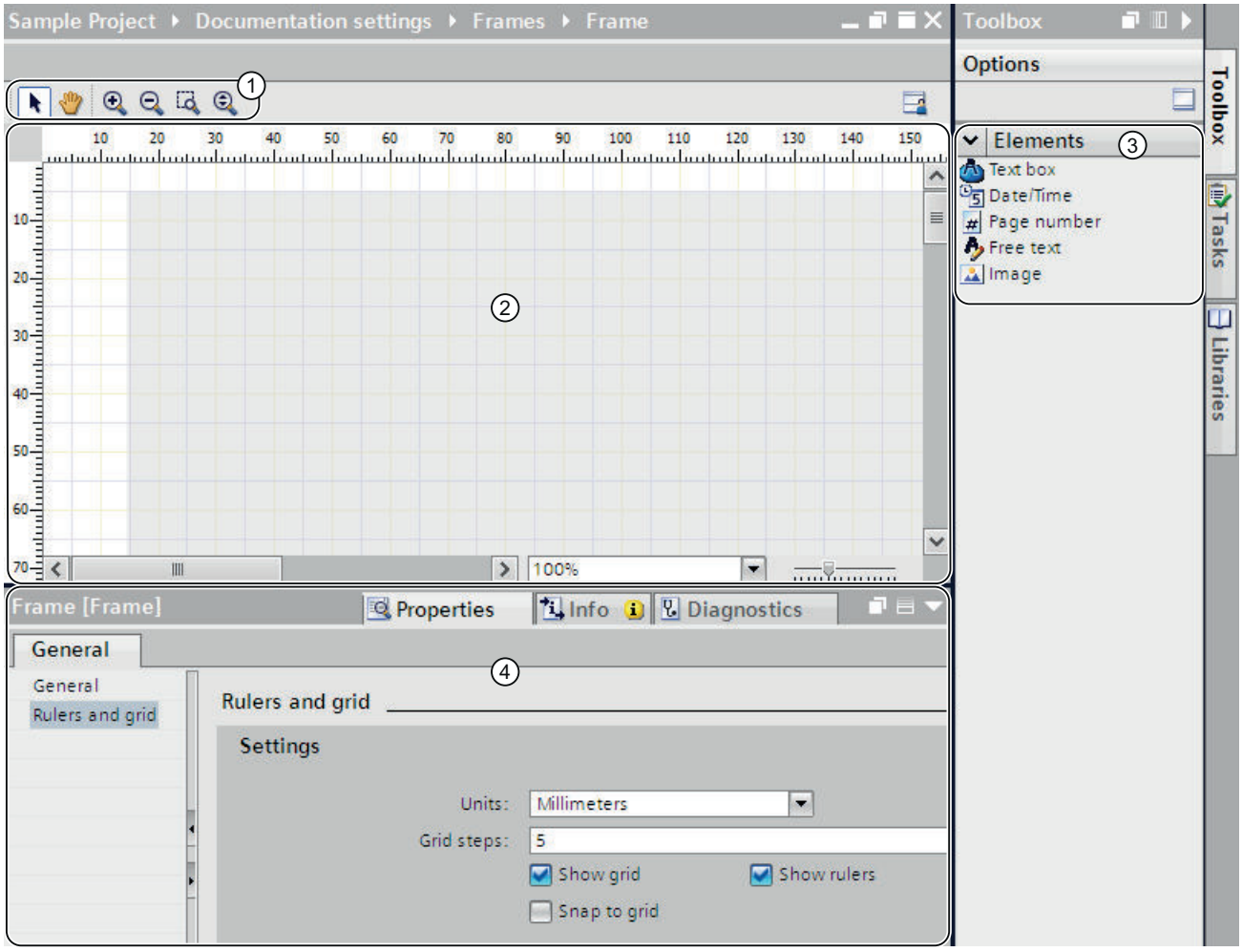
Creating frames (Page 386)

General operation of the documentation editor (Page 389)

General operation of the documentation editor

Components of the documentation editor

The following figure provides an overview of the components of the documentation editor:



① Toolbar

The toolbar provides the following tools (from left to right):

- Arrow tool
  Enables object selection.
- Navigation tool
  Allows shifting of the partial page.
- Zoom-in button
  Magnifies the page display incrementally.
- Zoom-out button
  Reduces the page display incrementally.
- Zoom with selection
  Adapts the page size to a selected work area.
- Dynamic zoom
  Adapts the page width to the work area.

② Work area

You can design the cover page or frame in the work area.

③ "Toolbox" task card

The "Toolbox" task card contains various types of placeholders that you can use on the cover sheet or frame. The placeholders can be placed in the work place using a drag-and-drop operation.

④ Properties in the Inspector window

You can display and modify the properties of the currently selected object in the "Properties" tab of the Inspector window. For example, you can modify the properties of the page, format text, specify the position of objects on the page, etc.

Operation in the documentation editor

The following basic functions are available in the documentation editor:

- Drag-and-drop functionality
  The documentation editor is a graphic editor. which means you can place objects anywhere with the mouse. An image of the page is displayed in the work area. This image corresponds to the ultimate print layout.
  If you want to select objects on the page in order to move them or modify their properties, the arrow tool must be activated in the toolbar.
- Zoom function
  You can use the zoom function to change the size of the page display. You have two options for changing the page size:
  - Via the buttons in the toolbar
    Select the "Zoom in" or "Zoom out" magnifying glass button in the toolbar of the documentation editor. Then click on the page in order to magnify (zoom in) or reduce (zoom out) the page incrementally.
    To zoom in on a particular area, select the "Zoom with selection" tool and use the mouse to drag an outline around the area you want to focus on.
    To continuously zoom in or zoom out of the work area, use the "Dynamic zoom" tool. To magnify the page display, click anywhere on the work area, and then hold down the mouse button while dragging the mouse toward the top of the page. To reduce the page display, drag the mouse toward the bottom of the page.
  - Via the zoom bar
    You can also use the zoom bar (located in the bottom right corner of the work area) to change the display size. Select a percentage value from the drop-down list or enter a percentage value. Alternatively, you control the display size using the slider.
- Navigation over the page
  In addition to scrolling, you the option of changing the partial page with the navigation tool. To change the partial page with the navigation tool, select the Hand button in the toolbar. Then, click anywhere on the page and hold the mouse button down while moving the page to the desired position.

Using and adapting the positioning aids

You have various aids at your disposal to help you position elements on the page:

- Rulers
  Rulers are affixed to the page margins in the work area.
- Page grid
  A grid is placed underneath the page in the work area.

You can display, hide or adapt the positioning aids in the Inspector window under "Properties > Rulers and grid". You can make the following settings:

- Units:
  Specify the unit of measurement for the grid and the rulers.
- Grid steps:
  Specify the width of the grid.
- Show grid:
  Specify whether the grid is to be displayed or hidden.

- Snap to grid:
  Specify whether objects are to be aligned automatically to the grid. If the option is selected, the grid lines function like a "magnet".
- Show rulers:
  Specify whether the rulers are to be displayed.

See also

Editing cover pages and frames (Page 388)

Specifying the print area (Page 392)

Inserting placeholders for metadata (Page 392)

Specifying the print area

An area within the frame is provided for the actual printed contents. The project data is then always inserted inside this defined and uniformly consistent area within the frame. You can adjust the size of the print area.

Requirement

A frame is open in the documentation editor.

Procedure

To define an area for the printed contents, follow these steps:

1. Click on the slightly darker area within the page display in the documentation editor to select the area for the print content.
   This opens the properties of the area to be printed in the Inspector window.
2. Enter the position of the print area on the X and Y axes in the Inspector window.
3. Specify the width and height of the print area in cm in the Inspector window.

Alternatively, you can change the width and position of the print area in the graphic display of the page. To do so, use the mouse to drag the margins of the print area until the desired size and position are achieved.

See also

Creating frames (Page 386)

General operation of the documentation editor (Page 389)

Inserting placeholders for metadata

You can provide placeholders on the cover page and in a frame. The placeholders are automatically filled with metadata from documentation information during printing, if they are placeholders for text. Alternatively, you can add non-modifiable data, such as free text or an image.

All elements are arranged in numbered Z-Orders. If objects overlap, you can determine in which sequence these are to be arranged.

Types of placeholders

The following types of placeholders are available to you:

- Text field
  The text field stands as a placeholder for a text element from a document information. In the properties of the text field, you set which text from a document information should be automatically inserted during printing.
- Field for date and time
  A date and time are inserted instead of the placeholder when printing. This can be the date of creation or the point in time when the last change was made to the project. In the properties of the Inspector window, you specify which date or time is printed.
- Page number
  The correct page number is automatically applied when printing.
- Free text
  You can enter freely selectable text in the properties of the text field. The text is static and is not influenced by the document information selected at the time of printing.
- Image
  Select the image file in the properties of the placeholder in the Inspector window. Images in the formats BMP, JPEG, PNG, EMF or GIF are possible.

Requirement

An cover page or frame is open in the documentation editor.

Procedure

To insert placeholders for metadata on the cover sheet or in a frame, follow these steps:

1. Drag a field from the "Toolbox > Elements" task card to the work area of the documentation editor.
   The placeholder is inserted. The placeholder properties are shown in the Inspector window and can be edited there.
2. Select the metadata to be inserted during printing from the "Text" drop-down list in the Inspector window under "Properties > General > Text box". Alternatively, you have the option of entering free text or selecting an image depending on the type of placeholder.
3. In the Inspector window under "Properties > General > Position and size", specify the position of the placeholder on the X and Y axis and enter the width and height of the text box in cm. You specify the sequence of the objects in the "Z-Order" field, if these overlap. The smaller the value, the further down an object is located.
4. In the Inspector window, go to "Properties > View" and select the font formatting and the orientation of the text as well as the alignment of the text. You cannot make this setting for images.

See also

General operation of the documentation editor (Page 389)

8.4.1.8 Displaying print preview

Creating a print preview

Creating a print preview

You can create a preview of the printout. Document information can be chosen for this, in the same way as for the actual printout. In this way, you preview the selected frame and, if applicable, the cover sheet. The settings are retained for later printing.

Procedure

To create a print preview and to set the scope of the later printout, follow these steps:

1. Select the "Print preview" command in the "Project" menu.
   The "Print preview" dialog opens.
2. Select the frame layout you want to use for the printout.
   - In the "Document information" drop-down list, select the documentation information you want to use later for the printout.
   - Select the "Print cover page" check box to print the cover page, which is specified in the selected document information.
   - Select the "Print table of contents" check box to add a table of contents to the printout.

   The check boxes for printing the cover page and the table of contents can only be selected if you have started the printout in the project tree.
3. Under "Print objects/area", select what is to be printed. The selection is only possible if you have started the printout from an editor that supports this function.
   - Select "All" to print out the entire content of the editor.
   - Choose "Selection" to print only the objects currently selected in the editor.

4. Select the print scope under "Properties".
   - Select "All" to print all configuration data of the selected objects.
   - Select "Visible" to print the information of an editor that is currently visible on the screen. This option can only be chosen if you have started the printout from an editor that supports this function.
   - Select "Compact" to print out an abbreviated version of the project data.
5. Click "Preview" to generate the preview.
   A print preview is created in the work area.

Note

Wait time for extensive documents

It can take several minutes to generate the print preview in the case of very extensive projects. You can continue working normally in the meantime on systems with adequate resources. The progress of the print preview is shown in the status bar.

See also

Operation within the print preview (Page 396)

Operation within the print preview

Functions within the print preview

The print preview shows an exact image of the subsequent printout. You can use the buttons in the toolbar to modify the print preview display. The following functions are available (from left to right):

- Navigation mode
  Allows shifting of the partial page.
  To change the partial page with the navigation tool, select the arrow button in the toolbar. Then, click anywhere on the page and hold the mouse button down while moving the page to the desired position.
- Zoom function
  - "Zoom in" and "Zoom out"
    Magnifies or reduces the page display.
    To zoom in or zoom out the display incrementally, select the corresponding button. Then click on the page in order to magnify (zoom in) or reduce (zoom out) the page incrementally.
    To zoom in on a particular area, enable the "Zoom with selection" icon and use the mouse to drag an outline around the area you want to focus on.
    To zoom dynamically through the page, select the button "Zoom in / zoom out dynamically". With pressed mouse button, scroll down over the page to zoom in. Scroll up to zoom out.
  - Percentage value in the drop-down list
    Specifies the display size of the page in percent.
    Enter a percentage value or select a percentage value from the drop-down list. Alternatively, choose the "Fit to page" option from the drop-down list to adapt the page size to the work area. Or, choose "Fit to width" to adapt the page width to the work area.
- "Forward" and "Backward"
  Each change in the partial page, the page count, or the display size is saved in a history in the background. You can use the "Forward" or "Backward" button to return to the previous view or the next view.
- Page navigation
  - "First page"
    Jumps to the first page
  - "Previous page"
    Goes to the previous page.
  - "Page number" input field
    Shows the current page. To jump directly to a page, enter the page number of the page you want to view.
  - "Next page"
    Goes to the next page.
  - "Last page"
    Jumps to the last page.

See also

Creating a print preview (Page 394)

8.4.1.9 Printing project data

You have two options for printing out project data:

- Print immediately using default settings by means of the "Print" button in the toolbar.
  The button is only active if a printable object is selected.
- Printout with additional setting options with the "Project > Print" menu command.
  For example, you can choose a different printer or specific documentation information or you can specify whether a cover page and table of contents are to be printed. In addition, you can specify the print scope or display a print preview prior to printing.

Requirement

- At least one printer is configured.
- The objects to be printed are not protected.
  The print scope for protected objects is limited. Disable the know-how protection to print the objects in full.

Printing project data

To print out data from the current project or the entire project with additional setting options, follow these steps:

1. Select the entire project in the project tree in order to print out the entire project. To print only individual elements within a project, select them in the project tree.
2. Select the "Print" command in the "Project" menu.
   The "Print" dialog opens.
3. Select the printer in the "Name" box.
4. Click "Advanced" to modify the Windows printer settings.
5. Select the frame layout you want to use for the printout.
   - Select the documentation information in the "Document information" drop-down list.
     The frame stored in the document information is used for the printout. All placeholders within the chosen frame are filled with the metadata from the selected document information.
   - Select the "Print cover page" check box to print the cover page, which is stored in the selected document information.
   - Select the "Print table of contents" check box to add a table of contents to the printout.

   The check boxes for printing the cover page and the table of contents can only be selected if you have started the printout in the project tree.

6. Under "Print objects/area", select what is to be printed. The selection is only possible if you have started the printout from an editor that supports this function.
   - Select "All" to print out the entire content of the editor.
   - Select "Selection" to print only the objects currently selected in the editor.
7. Select the print scope under "Properties".
   - Select "All" to print all configuration data of the selected objects.
   - Select "Visible" to print the information of an editor that is currently visible on the screen. This option can only be chosen if you have started the printout from an editor.
   - Select "Compact" to print out an abbreviated version of the project data.
8. Click "Preview" to generate a print preview in advance.
   A print preview is created in the work area.
9. Click "Print" to start the printout.

Note

Scope of the "Print" dialog

The options available in the "Print" dialog vary depending on the elements to be printed.

Result

The project data is prepared in the background for printing and then printed on the selected printer. The status bar shows the progress of the print operation. You can continue working normally while data is being prepared for printing.

The print results and any errors or warnings are listed in the Inspector window under "Info" at the conclusion of the print job.

Canceling a print job

To cancel an active print job, follow these steps:

1. Click the blue cross in the status bar next to the progress display for the printout.
   The printout is aborted.

See also

Protection concept for project data (Page 378)

Revoking access rights for devices (Page 378)

Printout of project contents (Page 381)

Designing cover pages and frames (Page 388)

8.4.2 Printing module labels

8.4.2.1 Print function for module labels

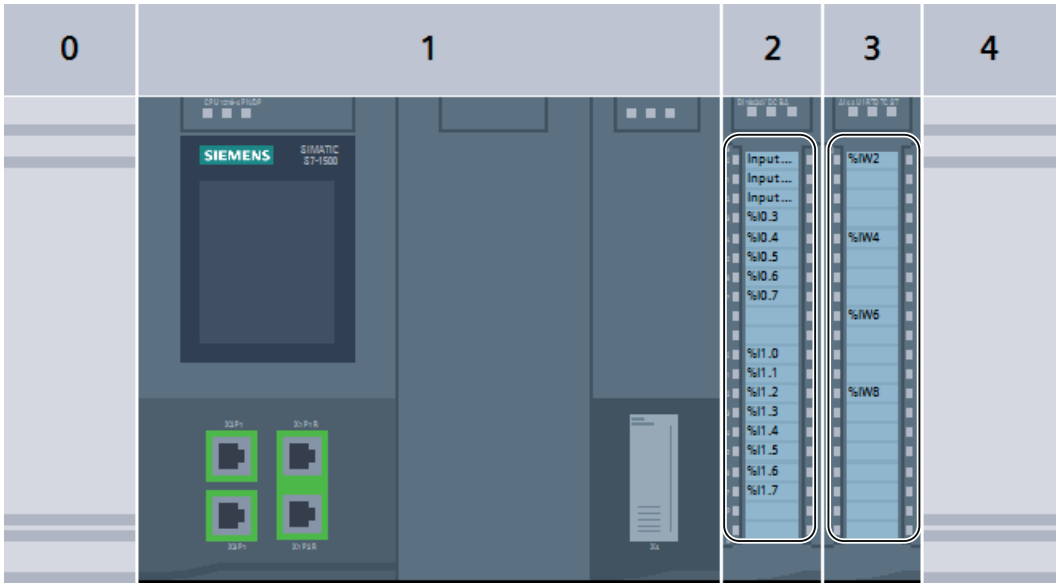
Printing of module labeling strips for hardware modules

You can print labeling strips for the modules in your project with the help of the TIA Portal. The labeling strips are custom-fit to each module and can contain the following printed information:

- Symbolic name of the input or output
- Absolute address of the input or output
- Symbolic name and additionally the absolute address of the input or output. The order of the information can be specified.

The modules are displayed graphically in the device view. If you set the zoom level in the device view to at least 200%, the labels for the individual modules will be visible. The printout on the labeling strip corresponds to the representation of the labeling in the device view.

The following figure shows an example of two modules in the device view on which the labeling of the inputs and outputs is visible:



Export and further editing as Microsoft Word file

The labeling strips are first exported as a Microsoft Word file (.docx) before they are printed. The file can be further edited with commonly available word processing programs such as Microsoft Word. The individual labeling strips are represented as a table in the .docx file.

The character spacing of the text within the table is adapted by default, so that texts are not truncated. If you want to prevent this from stretching or compressing the text too much, change the character spacing of the text in the table cell properties.

Printing the labeling data to an XML file

As an alternative to printing the labeling strips, you can output the addresses of the inputs and outputs of a module to an XML file. You can use the export to an XML file for devices for which no ready-made labeling strips are available. You can also use the export to an XML file to create labeling strips with another program. The program must be able to transform the data of the XML file into an input format that is suitable for the labeling system. The schema of the XML file is available in the section "XML schema of the export file (Page 403)".

Print media

You can print the labeling strips either on ready-made print sheets or on standard DIN A4 paper. You can separate the individual labeling strips from the ready-made print sheets and insert them in the designated labeling areas of your modules. If you print on standard paper, the individual labeling strips must be cut out. Cut marks are automatically included on the printout and serve as aids.

Because the paper feeds of printers differ slightly, the printout may be slightly offset on the paper. When the labeling strips are printed on ready-made sheets, printing that is accurate to the millimeter is important. Otherwise, the text will not be fit exactly inside the stamped area. In addition, if the printing is imprecise, the labeling of an input or output may no longer be congruent with the channel status displays of the module. To ensure precise printing, you can enter an offset value for your printer in the TIA Portal. For information on how to determine the proper offset value for your printer, refer to Chapter "Determining the print area offset (Page 406)".

See also

Exporting labeling data as XML (Page 402)

XML schema of the export file (Page 403)

Printing labels (Page 400)

Determining the print area offset (Page 406)

Documentation settings (Page 380)

8.4.2.2 Printing labels

You can print labeling strips for the modules in your project if provision has been made for attaching labels to the utilized modules. The labels are first exported to a Microsoft Word file (.docx). A separate .docx file is created for each module family (for example, for all selected S7-1500 modules). The labels are always printed from the word processing program.

Requirement

The following requirements apply to printing of labeling strips:

- The chosen modules must support the printing of labels. Otherwise, the data can only be output to an XML file.
- A word processing program that supports Microsoft Word DOCX files must be installed, e.g., Microsoft Word 2010 or later.
- You need the ready-made labels for your modules or commercially available DIN A4 paper.

Procedure

To print labeling strips for hardware modules, follow these steps:

1. In the project tree, select the modules for which you want to print labeling strips.
   - You can select one or more stations in order to print out labeling strips for all modules plugged into these stations.
   - Alternatively, select the desired modules below the stations in the "Local modules" folder.
2. Right-click one of the devices, and select the "Export labeling strips" command from the shortcut menu.
   The "Export labeling strips" dialog opens.
3. In the "Content of the labeling strip" area, select the data to be printed on the labeling strip:
   - Select "Symbolic name" in order to print the symbolic name of the input or output (corresponds to the contents of the "Name" column in the IO tag table).
   - Select "Absolute address" in order to print the absolute address of the input or output (corresponds to the contents of the "Address" column in the IO tag table).
   - Select "Absolute and symbolic address" or "Symbolic and absolute address" in order to print both addresses. Printing takes place according to the specified order.
4. In the "Export format" area, define how the labeling data will be output.
   - Select "Print on SIEMENS labeling sheet" if you are printing on a ready-made label sheet for your modules.
   - Select "Print on plain DIN A4 page" if you are printing on standard DIN A4 paper.
5. Select correction values for your printer in the "Offset print area", if required. The correction values are used for correct alignment of the print area. Correction values are only necessary if you are printing on ready-made labeling strips.
   - Enter a correction value, in millimeters, in the "Vertical offset" field. A negative value shifts the print area upward. A positive value shifts the print area downward.
   - Enter a correction value, in millimeters, in the "Horizontal offset" field. A negative value shifts the print area to the left. A positive value shifts the print area to the right.
6. In the "Path" field, select the path where the exported files will be stored.
7. Click the "Export" button to start the export.
   The export files are created.
8. Open the DOCX files with a conventional word processing program, such as Microsoft Word, and change the design of the labeling strips if necessary.

9. Print out the labeling strips from your word processing program. Use the paper that you specified in the Export dialog for this.
10. If you are using ready-made sheets, separate the labeling strips at the stamped lines provided for that purpose. When standard DIN A4 paper is used, you must cut out the labeling strips.

See also

Determining the print area offset (Page 406)

Exporting labeling data as XML (Page 402)

8.4.2.3 Exporting labeling data as XML

The TIA Portal supports a large number of different modules and can be continually expanded using Hardware Support Packages, for example. Ready-made labeling strips are not available for every supported module. However, you can still use the TIA Portal to label inputs and outputs of modules that are not supported. First, you export the absolute and symbolic addresses of the inputs and outputs to a standardized XML file. Then, you import the XML file to an external program for printing the labels. In this program, you prepare the data to suit your modules and print out the labels.

Procedure

To export labeling data for hardware modules as XML, follow these steps:

1. In the project tree or network view, select the modules for which you need labeling strips.
   - You can select one or more stations in order to export the input and output addresses of all modules plugged into these stations.
   - Alternatively, select the desired modules below the stations in the "Local modules" folder.
2. Right-click one of the devices and select the "Export module labeling strips" command from the shortcut menu.
   The "Export labeling strips" dialog opens.
3. In the "Export format" area, select the "Export to XML file" option.
4. In the "Path" field, select the path where the XML file will be stored.
5. Click the "Export" button to start the export to an XML file.
   The XML file is created with the name "<Project name>_IO_Channels.xml".

See also

XML schema of the export file (Page 403)

Print function for module labels (Page 399)

Printing labels (Page 400)

8.4.2.4 XML schema of the export file

XML schema of an export file

The XML file for module labeling strips is structured according to the following schema:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://tempuri.org/XMLSchema.xsd"
elementFormDefault="qualified"  xmlns:mstns="http://tempuri.org/
XMLSchema.xsd"  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Stations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Station" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
        <xs:element name="Rack" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
          <xs:element name="Module" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
            <xs:element name="IOChannel" maxOccurs="unbounded">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Address" type="xs:string"></
xs:element>
               <xs:element name ="Tag" type="xs:string"></xs:element>
              </xs:sequence>
            <xs:attribute name="Number" type="xs:int"></xs:attribute>
            </xs:complexType>
            </xs:element>
            </xs:sequence>
            <xs:attribute name="Name"></xs:attribute>
          </xs:complexType>
          </xs:element>
          </xs:sequence>
          <xs:attribute name="Name" type="xs:string"></xs:attribute>
        </xs:complexType>
        </xs:element>
        </xs:sequence>
        <xs:attribute name="Name" type="xs:string"></xs:attribute>
      </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  </xs:element>
</xs:schema>
```

Example of an XML file

The following example shows an XML file that contains the labeling data for an S7-1500 CPU with a digital input module and an analog input module:

```
<?xml version="1.0" encoding="UTF-8"?>
<Stations>
  <!-- The CPU is specified first -->
  <Station Name="S71500/ET200MP-Station_1">
    <Rack Name="Rack_0"> <!-- Name of the rack -->
      <Module Name="Sample S7-1500" /> <!-- Name of the CPU -->
     <Module Name="DI 16x24VDC BA_1"> <!-- Name of the digital input
module -->
       <!-- The individual channels of the digital input module are
specified -->
        <IOChannel Number="0">
          <Address>%I0.0</Address>
          <Tag>Input Value 1</Tag> <!-- Symbolic address of input 0
-->
        </IOChannel>
        <IOChannel Number="1">
          <Address>%I0.1</Address>
          <Tag>Input Value 2</Tag>
        </IOChannel>
        <IOChannel Number="2">
          <Address>%I0.2</Address>
          <Tag>Input Value 3</Tag>
        </IOChannel>
        <!-- All other channels follow -->
      </Module>
      <Module Name="AI 4xU/I/RTD/TC ST_1"> <!-- Name of the analog
input module -->
        <!-- The individual channels of the analog input module are
specified -->
        <IOChannel Number="0">
          <Address>%IW2</Address>
          <Tag> <!-- Symbolic addresses are not specified for the
analog input module. -->
          </Tag>
        </IOChannel>
        <IOChannel Number="1">
          <Address>%IW4</Address>
          <Tag>
          </Tag>
        </IOChannel>
        <IOChannel Number="2">
          <Address>%IW6</Address>
          <Tag>
          </Tag>
        </IOChannel>
        <IOChannel Number="3">
          <Address>%IW8</Address>
          <Tag>
          </Tag>
```

```
      </IOChannel>
    </Module>
    <Module Name="Sample S7-1500" />
    <Module Name="DI 16x24VDC BA_1">
      <IOChannel Number="0">
        <Address>%I0.0</Address>
        <Tag>Input Value 1</Tag>
      </IOChannel>
      <IOChannel Number="1">
        <Address>%I0.1</Address>
        <Tag>Input Value 2</Tag>
      </IOChannel>
      <IOChannel Number="2">
        <Address>%I0.2</Address>
        <Tag>Input Value 3</Tag>
      </IOChannel>
      <!-- All other channels follow -->
    </Module>
    <Module Name="AI 4xU/I/RTD/TC ST_1">
      <IOChannel Number="0">
        <Address>%IW2</Address>
        <Tag>
        </Tag>
      </IOChannel>
      <IOChannel Number="1">
        <Address>%IW4</Address>
        <Tag>
        </Tag>
      </IOChannel>
      <IOChannel Number="2">
        <Address>%IW6</Address>
        <Tag>
        </Tag>
      </IOChannel>
      <IOChannel Number="3">
        <Address>%IW8</Address>
        <Tag>
        </Tag>
      </IOChannel>
    </Module>
  </Rack>
 </Station>
</Stations>
```

See also

Exporting labeling data as XML (Page 402)

8.4.2.5 Determining the print area offset

If you are using a ready-made label sheet, the printing on it must be applied precisely so that the text is properly oriented on the prestamped labels and will have the proper fit relative to the channel status displays of the module. However, the paper feeds vary slightly from one printer to another. For this reason, you must enter a suitable correction value for your printer in the TIA Portal, if necessary. The print area is then shifted in the exported .docx file in such a way that the printing fits precisely on the ready-made label sheet.

The settings for shifting the print area are stored for the specific Windows user. If you log on to Windows using a different user name, you have to enter the correction values again.

The procedure for determining the correction value for your printer is described below.

Requirement

- You require a ready-made label sheet.
- You must have access to the actual printer that you will use subsequently for the printout. The printer must be made ready for printing on standard DIN A4 paper.

Procedure

To determine the correction value for your printer, follow these steps:

1. Print out a label sheet on standard DIN A4 paper, as described in Chapter "Printing labels (Page 400)".
2. Compare the printout on the DIN A4 paper with the ready-made label sheet.
3. If the print area is offset, you must use correction values.
   - Using a ruler, measure the horizontal offset relative to the ready-made label sheet. This will be entered later in the "Horizontal offset" field of the Export dialog box for the printing. If the print area is offset to the right, a negative correction value must be entered. If the print area is offset to the left , a positive correction value must be entered.
   - Using a ruler, measure the vertical offset relative to the ready-made label sheet. This will be entered later in the "Vertical offset" field of the Export dialog box for the printing. If the print area is offset downward, a negative correction value must be entered. If the print area is offset upward, a positive correction value must be entered.

8.5 Working with multi-language projects

8.5.1 Project text basics

Texts in different languages in the project

When you enter texts while working on a project, you would normally do this in your own language. If you then pass on the project to someone else who does not know this language, this person will require a translation of the relevant texts to a language they know. This is why all texts can be translated. In this way, you can ensure that anyone who is subsequently confronted with the texts sees the texts in his/her language of choice.

Project language

Project languages are all languages in which a project will later be used. Based on the editing language, all the texts can be translated to the various project languages. You specify the languages that will be available in the project tree under "Languages & Resources > Project languages".

Editing language

Every project has an editing language. When you enter texts, these are always created in the editing language. You should therefore make sure that the editing language set is the language in which you enter the texts. This avoids problems if you translate the texts later.

The editing language does not depend on the language of the user interface. You could, for example, set English as the user interface language, but use Italian as the editing language. If you enter texts, these will be created in the project language "Italian" in this case, although the user interface of the TIA Portal is displayed in English.

You set the editing language in the project tree under "Languages & Resources > Project languages > Editing language".

Reference language

The reference language is used as a template for translation. The text is displayed in the reference language for each text box in the "Tasks > Languages and resources" task card. You therefore know which text that belongs in a text box, even when no text is entered in the currently selected editing language.

User texts and system texts

For clarification purposes, a distinction is made between user texts and system texts:

- User texts are texts that the user created.
- System texts are texts that are created automatically according to the configuration in the project.

You manage the project texts in the project tree under "Languages & Resources > Project texts".

Examples of multilingual project texts

You can, for example, manage the following project texts in more than one language:

- Block titles and block comments
- Network titles and network comments
- Comments in tables
- Alarm texts
- Operator-relevant texts
- Text lists
- Labels of buttons
- Display names of recipes

Translating texts

The following procedures are available to translate texts.

- Translate all texts used in the project in tabular form
  You can enter the translations for the individual project languages directly in the "Project texts" table. You can find the table in the project tree under "Languages & Resources > Project texts".
- Specify text assigned to individual objects in the Inspector window
  In the Inspector window, you can translate the texts that are assigned to the currently selected objects. Columns are displayed in a table for all available project languages. You can enter the translations for each text in the columns.
- Translating texts using reference texts
  You can change the editing language for shorter texts. All the text cells are filled again with the default values and can be filled in the current language. As orientation, you can display what you last entered in the box in the reference language. To do this, select the "Tasks" task card and open the "Languages & resources".
- Exporting texts and translating them externally
  With larger volumes of text, you can export the texts to an Office Open XML file and translate them in a conventional table calculation program. You then import the translated list again into the TIA Portal.

Note

Using East Asian project languages

You need Microsoft Windows at least in the Professional version or higher to display East Asian project languages. Microsoft Windows in the Professional version must be installed in the local language. With the "Ultimate" or "Enterprise" versions, it is sufficient to install the appropriate language pack.

See also

Overview of the program settings (Page 230)

Changing the settings (Page 236)

Application examples for multilanguage projects (Page 415)

8.5.2 Categories of user texts

Overview of text categories

Below you will find an overview of the categories of user texts that you can select for export.

Text categories	Usage
Formatted alarm text	
HMI scree	Text that appears in an HMI screen
HMI comments	Text that appears in a comment in Runtime
HMI Runtime	
Category for text lists	
Multilingual text category	
Alarm class text	
Alarm text	Text that appears in a message display
SiVArc configuration	
Other text categories	

8.5.3 Select project languages

All the texts can be displayed within a project in the same language that you selected for your software user interface. This means that all project texts must exist in the corresponding language. You can select the available project languages yourself.

Procedure

To select the project languages, follow these steps:

1. Click on the arrow symbol to the left of "Languages & Resources" in the project tree.
   The elements below this are displayed.
2. Double-click on "Project languages".
   In the work area, you will see a list of languages that you can select.
3. Select the required languages.

Result

All texts can be displayed in the activated languages if there is already a translation for these languages.

8.5.4 Setting the editing language

All the texts in the project are created in the editing language when they are entered. If you change the editing language, all future text input will be stored in the new editing language.

Procedure

To change the editing language, follow these steps:

1. Click on the arrow symbol to the left of "Languages & Resources" in the project tree. The lower-level elements are displayed.
2. Double-click on "Project languages". The possible settings for the project languages are displayed in the work area.
3. Select the editing language in "General > Editing language".

8.5.5 Translating all project texts in tabular form

You can display and edit all project text used in the currently open project in a list. User and system texts are separated into two different lists for clarity. Both lists contain a column for each project language. Enter the translations of the texts in the respective columns.

Requirement

You have selected at least one further project language.

Procedure

To translate text in the project-wide list, follow these steps:

1. Click on the arrow symbol to the left of "Languages & Resources" in the project tree. The elements below this are displayed.
2. Double-click "Project texts". A list with the user texts in the project is displayed in the work area.
3. Click on "System texts" if you you want to edit the list of system texts rather than the user texts.
4. You can improve the clarity of the lists if you have a lot of texts.
   - To group identical texts and to translate them all at once, click the "Switch on/off grouping" button in the toolbar.
   - To hide texts that do not have a translation, click the "Filter for empty texts on/off" button in the toolbar.
   - To further limit the displayed project texts to certain devices, select the devices for which you want to display project texts in the drop-down list.
5. Enter the translation of the project texts in the relevant column.

8.5.6 Translating project texts

If you want to edit the text of individual objects, it would be too difficult to locate the matching text in the table with all project texts. For this reason, there is a table in the Inspector window in which only the texts assigned to the currently selected objects are displayed. In the table, you can add missing translations for individual project languages or change existing texts.

Requirement

Enter a text in at least one project language for the texts to be translated.

Procedure

To edit the text of the currently selected object, follow these steps:

1. Select the object whose text you want to edit.
2. Open the "Properties" tab in the Inspector window.
3. Open the lower-level "Texts" tab in the inspector window.
   A table with all the texts that belong to the selected objects is displayed. The table contains one column for the currently selected editing language and the reference language, as well as additional columns for the other project languages.
4. Add or change the entries in the table for each project language.

See also

Application examples for multilanguage projects (Page 415)

8.5.7 Translating project texts using reference texts

Introduction

After changing the editing language, all texts are shown in input boxes in the new editing language. If there is not yet a translation available for the newly set language, the input boxes are empty or filled with default values.

If you enter text in an input box, this is saved in the current editing language. Following this, the texts exist in two project languages for this input field, in the previous editing language and in the current editing language. This makes it possible to create texts in several project languages.

You can display existing translations for an input box in other project languages. These serve as a comparison for text input in the current editing language and they are known as the reference language.

Note

The display of reference texts depends on the installed products and is not supported by every editor.

Requirement

There is at least one translation into a different project language for an input field.

Procedure

To display the translation of an input cell in a reference language, follow these steps:

1. In the "Tasks" task card, select the "Languages & Resources" pane.
2. Select a reference language from the "Reference language" drop-down list.

Result

The reference language is preset. If you click in a text box, translations that already exist in other project languages are shown in the "Tasks > Languages & Resources" task card.

8.5.8 Exporting project texts

You can export project texts for translation and then reimport them. The texts are exported to an Office Open XML file with the extension ".xlsx". This can be edited in Microsoft Excel or a number of other spreadsheet programs.

The following export options are available:

- Exporting individual project texts
  You can select individual texts in the project texts editor and then export the selected texts.
- Exporting project texts of a device
  When you have selected a device, the "Properties > Texts" tab of the Inspector window includes all texts that are part of the respective device. Here you can export all texts that are part of the respective device.
- Exporting all user texts or system texts at once
  You can either export all texts in the project or further limit the export by categories.

Note

Row limit in Microsoft Excel

Note that spreadsheet programs may be able to process only a certain number of rows. Microsoft Excel 2003 supports a maximum of 65536 rows, for example. Later versions of Microsoft Excel support significantly more rows.

Exporting individual project texts

To export individual project texts, follow these steps:

1. Open the "Languages & Resources" folder in the project tree.
   The lower-level elements are displayed.
2. Double-click "Project texts".
   The project texts editor opens.
3. Choose the "User texts" or "System texts" tab in the editor, depending on which texts you want to export.
4. Select the project texts you want to export.
5. Click "Export project texts" on the editor toolbar.
   The "Export" dialog box opens.
6. In the "Source language" drop-down list, select the reference language from which you want to translate.
7. In the "Target language" drop-down list, select the editing language into which you want to translate the texts. The drop-down list contains the project languages you specified previously. If the required language is missing, you must first specify it in the project languages editor.
8. Specify a file path and a file name for the export file in the "Select file for export" input box.
9. Click "Export".

Exporting project texts of a device

1. To export all project texts that are part of a specific device, follow these steps:
2. Select the device and open the device properties in the Inspector window.
3. Open the "Texts" tab in the Inspector window.
4. On the toolbar, click the "Export project texts" icon.
   The "Export" dialog box opens.
5. In the "Source language" drop-down list, select the reference language from which you want to translate.
6. In the "Target language" drop-down list, select the editing language into which you want to translate the texts. The drop-down list contains the project languages you specified previously. If the required language is missing, you must first specify it in the project languages editor.

7. Specify a file path and a file name for the export file in the "Select file for export" input box.
8. Click "Export".

Exporting all system or user texts

To export all project texts, follow these steps:

1. Select the "Export project texts" command in the "Tools" menu.
   The "Export" dialog box opens.
2. In the "Source language" drop-down list, select the reference language from which you want to translate.
3. In the "Target language" drop-down list, select the editing language into which you want to translate the texts. The drop-down list contains the project languages you specified previously. If the required language is missing, you must first specify it in the project languages editor.
4. In "Select content", select the check box "User texts" to export user texts. To export system texts, select "System texts". To export both user texts and system texts, select both check boxes.
5. In "Select content", select the required text categories for the user texts or the system texts.
6. In the "Export file" input field, specify a file name for the export file.
7. In the "Path" input field, select a path in the data system to which the export file is to be saved.
8. Click "Export".

See also

Application examples for multilanguage projects (Page 415)

Importing project texts (Page 414)

8.5.9 Importing project texts

After external compilation in a table calculation program, you import the project texts into the TIA Portal. You can import the project texts at the following locations:

- The "Tools" menu
- In the toolbar of the project texts editor
- In the properties of a device
  When you have selected a device, the "Properties > Texts" tab of the Inspector window includes all texts that are part of the respective device. You can also import the texts of the device at this location.

Importing project texts

To import a file containing project texts, follow these steps:

1. Select the "Import project texts" command in the "Tools" menu.
   Alternatives:
   - Click the "Import project texts" icon in the toolbar of the project texts editor.
   - Select a device and open its properties in the Inspector window. Open the "Texts" tab and click the "Import project texts" icon in the toolbar.
2. The "Import" dialog box opens.
3. Select the path and the file name of the import file from the "Select file for import" field.
4. Select the "Import source language" check box if you have made changes to the source language in the export file and you want to overwrite the entries in the project with the changes.
5. Click "Import".

See also

Exporting project texts (Page 412)

8.5.10 Application examples for multilanguage projects

Let us assume you are working in a team with colleagues some of whom speak English, some French and some German. You have created a project with the TIA Portal and have already created a functioning configuration.

To allow your other colleagues to be able to keep track of the project, you would like all devices being used to have comments in English and German. First, you would like to enter the comments in German. Following this, to save time and costs, you want to have the texts translated into English in a spreadsheet program by an external translation office.

In addition to this, you also want a single comment for a particular device in French so that your French-speaking colleague can continue working on this device.

The section below describes an example of how you can achieve this with the tools of the TIA Portal.

Translating the project into English

To enter the comments in German and to have them translated into English later, follow these steps:

1. Set the editing language to "German" and fill all the comment boxes with the relevant texts in German.
   On the device selected from the French-speaking colleague, enter, for example "Unser neues Gerät" in German first.
   All the comments are now stored in German.
2. Export all user texts to an Office Open XML file with the extension ".xlsx".

3. Have the user texts contained in the file translated into English in a spreadsheet program such as Microsoft Excel.
4. Import the file into the TIA Portal after it has been translated.
   All texts are now available in German and English.

Translating a single comment field to French

To translate an individual comment field to French, follow these steps:

1. Open the comment box for the device on which the French-speaking colleague will be working.
2. Open the "Languages & resources" pane in the "Tasks" task card.
3. Set "French" as the editing language in the "Languages & Resources" pane. As the reference language, set, for example, "English".
   Since no translation has yet been installed in French, the comment box is empty. In the "Languages & Resources" pane, the English translation "Our new device" is displayed as a reference.
4. Orientating yourself on the English reference text enter "Notre nouvel appareil" in the comment box.
   The comment for this device is now available in the languages German, English and French.

See also

Project text basics (Page 407)

Exporting project texts (Page 412)

Translating project texts (Page 411)

8.6 Working with text lists

8.6.1 Text lists

Introduction

You can centrally manage texts to be referenced in alarms. All the texts are stored in text lists. Each text list has a unique name with which you can call up its content. A range of values is assigned to each text in a text list. If a value from a range of values occurs, the corresponding text is called up.

All the texts can be translated to all project languages. Here, you have two options available:

- You can enter the translation of the texts in a list. You will find the list in the project tree under "Languages & Resources > Project texts".
- You can export all texts to a file in Office Open XML format and enter the translation in a spreadsheet program. The translations can then be imported again. Only export the data to areas that are protected with appropriate access mechanisms. Only import files that originate from trusted sources.

The texts are translated into the other project languages within the framework of the project texts. In the text lists editor, you only have to manage the assignment of individual texts to a text list.

Each device in a project can have its own text lists. Device-specific text lists relate to only one device in the project and are therefore only valid for this device. For this reason, they are arranged under a device in the project tree. Device-specific text lists can be user-defined or created by the system.

Text lists editor

The text lists editor is divided into a bottom and top area. The top area displays the individual text lists. As soon as you select a text list, the included texts and the associated value ranges are displayed in the lower area. You can sort the table columns in the text lists editor in ascending and descending order by clicking the table header of the respective column.

User-defined and system-defined text lists

There are two types of text lists:

- User-defined text lists
  You can create user-defined text lists yourself and fill them with texts; in other words, you can specify value ranges and the corresponding texts yourself. With user-defined text lists, the name of the text list begins with "USER" as default. You can change this name to any suitable name.
- System-defined text lists
  System-defined text lists are created by the system. These always involve texts relating to devices. They are automatically created as soon as you insert a device in the project. With system alarms, the name of the text list begins with "SYSTEM". The name of the text list and the ranges of values it contains cannot be modified. You can only edit texts assigned to individual value ranges.

User-defined text lists	System-defined text lists
You can create new text lists and delete existing text lists.	You cannot create new text lists or delete text lists.
You can add and delete value ranges in the text lists.	You cannot add or delete value ranges in the text lists.
You can specify both the value ranges as well as the associated texts.	You can only edit the text associated with one value range.

See also

Exporting project texts (Page 412)

8.6.2 Creating user-defined text lists

You can create user-defined text lists for individual devices.

Requirement

- You are in the project view.
- A project is open.
- The project includes a least one device.

Procedure

To create user-defined text lists, follow these steps:

1. Click on the arrow to the left of a device in the project tree.
   The elements arranged below the device are displayed.
2. Double-click on "Text lists".
   All the text lists assigned to the device are displayed in the work area listed in a table.

3. Double-click on the first free row in the table.
   A new user-defined text list is created.
4. Enter a name for your new text list in the "Name" column.
5. Select if you want to specify the value ranges in decimal, binary or in bits from the drop-down list in the "Selection" column. Depending on the device, there may be further options available at this point.
6. Enter a comment in the "Comment" column.
   A new user-defined text list is created and you can now enter the value ranges and texts.

8.6.3 Editing user-defined text lists

You can enter value ranges and the corresponding texts in user-defined text lists. User-defined text lists are always located below a device in the project tree.

Requirement

- You are in the project view.
- A project is open.
- The project includes a least one device.

Procedure

To add to user-defined text lists with value ranges and texts, follow these steps:

1. Click on the arrow to the left of a device in the project tree.
   The elements arranged below are displayed.
2. Double-click on "Text lists".
   All the text lists assigned to the device are displayed in the work area listed in a table.
3. Select a text list in the table.
   The contents of the selected text list are displayed in the work area. There, you can enter a value range and assign texts to the individual value ranges.
4. Enter the value ranges you require in the "Range from" and " Range to" columns. The entry must be made in the numeric format selected for the text list.
5. Enter a text for each value range in the "Entry" column.

8.6.4 Editing system-defined text lists

In system-defined text lists, you can modify the individual texts assigned to a value range.

System-defined text lists are always located below a device in the project tree.

Requirement

- You are in the project view.
- A project is open.
- The project includes a least one device.

Procedure

To edit texts in system-defined text lists that are assigned to a value range, follow these steps:

1. Click on the arrow to the left of a device in the project tree.
   The elements arranged below are displayed.
2. Double-click on "Text lists".
   All the text lists assigned to the device are displayed in the work area listed in a table.
3. Select a text list in the table.
   The contents of the selected text list are displayed in the work area. Here, you can add to or edit the texts assigned to a value range.
4. Enter a text for each value range in the "Entry" column.

8.7 Using memory cards

8.7.1 Basics about memory cards

Introduction

Memory cards are plug-in cards that come in a variety of types and can be used for a variety of purposes. Depending on the device type or device family, memory cards can be used for purposes, such as:

- Load memory of a CPU
- Storage medium for projects, firmware backups, or any other files
- Storage medium for performing a firmware update
- Storage medium for the PROFINET device name

For information regarding the technical variants of the respective memory cards and general information on their handling, refer to the respective documentation for the device. For information on handling memory cards in the TIA Portal, refer to the online help under the keyword "Memory Card".

NOTICE
Memory card is unusable for SIMATIC devices If you use a SIMATIC memory card for non-SIMATIC purposes or you format it incorrectly, you will overwrite the internal structure of the SIMATIC memory card. The structure is not recoverable and the SIMATIC memory card becomes unusable for SIMATIC devices. Do not use memory cards for non-SIMATIC-related purposes, and do not format SIMATIC memory cards with third-party devices or Windows tools.

See also

Adding a user-defined card reader (Page 421)

Accessing memory cards (Page 422)

Displaying properties of memory cards (Page 423)

8.7.2 Adding a user-defined card reader

Introduction

If your card reader is not detected automatically, you can add it manually.

Requirement

The project view is open.

Procedure

To add a card reader, follow these steps:

1. Open the project tree.
2. Select the "Card Reader / USB memory > Add user-defined Card Reader" command in the "Project" menu.
   The "Add user-defined Card Reader" dialog opens.
3. In the drop-down list box, select the path for the card reader.
4. Confirm your entries with "OK".

See also

Basics about memory cards (Page 421)

Accessing memory cards (Page 422)

Displaying properties of memory cards (Page 423)

8.7.3 Accessing memory cards

Requirement

- A memory card is inserted in the card reader.
- The project view is open.

Note

You cannot work with multiple memory cards at one time. Only insert one memory card into the card reader.

Procedure

To access memory cards, follow these steps:

1. Open the project tree.
2. Select the "Card Reader / USB memory > Card Reader / Show USB memory" command in the "Project" menu.
   The "Card reader / USB memory" folder is displayed in the project tree.
3. Open the "Card Reader / USB memory" folder.
   You can now access the memory card.

Note

If data from a non-installed product is stored on the memory card, the folders that contain these data are shown in gray. You receive an error message when you attempt to access such a folder. Install the corresponding product if needed.

See also

Basics about memory cards (Page 421)

Adding a user-defined card reader (Page 421)

Displaying properties of memory cards (Page 423)

8.7.4 Displaying properties of memory cards

You can display the properties for the utilized memory cards. Note that different memory cards with different properties must be used, depending on the device.

Requirement

- A memory card is inserted in the card reader.
- The project view is open.

Procedure

To display the properties of a memory card, follow these steps:

1. Right-click on the memory card for which you want to display the properties.
2. Select the "Properties" command in the shortcut menu.
   The "Memory Card <name of the memory card>" dialog opens. The properties are displayed in this dialog.

See also

Basics about memory cards (Page 421)
Adding a user-defined card reader (Page 421)
Accessing memory cards (Page 422)

8.8 Using cross-references

8.8.1 Using cross-references

Introduction to cross-references

The cross-reference list provides an overview of the use of objects within the project. You can see which objects are interdependent and where the individual objects are located. Cross-references are therefore part of the project documentation.

You can also jump directly to the point of use of an object.

Which objects you can display and localize in the cross-reference list depends on the installed products.

See also

Displaying cross references of an instance (Page 456)

8.9 Providing user-defined documentation

8.9.1 Using user-defined documentation

User-defined documentation for project or library contents

Over time, you create your own contents in a project or a library. Your own contents include, for example, blocks, tags or library types. While the functionality of the TIA Portal is described in the supplied help system, there is no help for the contents you have created yourself. You can create your own user-defined documentation to explain to other employees how your project works or how to use individual library types.

You can provide user-defined documentation in the available user interface languages. The default user interface languages in the TIA Portal are English, German, French, Spanish, Italian, and Chinese. You need to observe a few conventions when you create user-defined documentation so that the help matching an object can be opened.

You create the user-defined documentation either in one of the supported Office formats or as compiled HTML help in CHM format.

Possible areas for user-defined documentation

You can offer user-defined documentation in the following areas of the TIA Portal, for example:

- Project tree
- "Libraries" task card and library view
- Some editors, depending on the products installed, for example:
  - Program editor
    The programming languages LAD, FBD, STL, SCL and GRAPH are supported. Block calls also support the connection of user-defined documentation.
  - "Screens" editor
    Uses of screens and instances of HMI faceplates are supported in the "Screens" editor.

Directories for user-defined documentation

Save the user-defined documentation in one of the following directories:

- Project folder
  If you create user-defined documentation for objects within a project, save this help in the project folder. The user-defined documentation is also included when you pass on the project.
- Directory of a global library
  If you create user-defined documentation for objects within a global library, save the user-defined documentation in the directory of the global library. The user-defined documentation is also included when you pass on the global library.
- Central directory on the hard drive or a network drive
  You can store the user-defined documentation in a central directory on the hard disk or on a network drive. In this way, you have access to the user-defined documentation in each project or you use the documentation on a network drive together in the team. You specify the central file directory for the user-defined documentation using an XML file or in the settings of the TIA Portal.

Homepage for the user-defined documentation

You can create a separate homepage for each language version of the user-defined documentation. The homepage for the user-defined documentation can contain general help for a project or for a library. The homepage must be saved in the central storage directory for user-defined documentation.

Calling the user-defined documentation

If user-defined documentation is available for an object, you call it with the keyboard shortcut <Shift+F1>. The user-defined documentation is always opened with the standard program specified for the respective file format in Microsoft Windows.

Once you have pressed <Shift+F1>, certain directories are searched in a fixed order for user-defined documentation. The search order is given below:

1. Search in the central directory for user-defined documentation
   1.1 Search for a CHM file
   1.2 Search for documentation in other file formats
2. Search in the project or library directory
   2.1 Search for a CHM file
   2.2 Search for documentation in other file formats

The search is initially performed in the language directory for the currently set user interface language of the TIA Portal. If no help is contained in this language directory, the search for user-defined documentation is performed in the same order in the English language directory.

As soon as user-defined documentation is found in one location, the user-defined documentation is opened and the search is canceled. If no user-defined documentation is found in any of the directories, a search for a homepage for user-defined documentation is performed in the order presented above. The search for a homepage is again initially performed

in the language directory for the currently set user interface language. If no homepage is found there, the English language directory is searched.

Call log

You can display a call log for the user-defined documentation for easier connection of the user-defined documentation. The alarms within the log indicate the directories in which documentation is searched for and whether the call of the user-defined documentation is successful. In addition, the file name that is expected for the file is indicated. This allows you to identify how you must name your documentation and the directories in which you must save the user-defined documentation. The call log has the same sequence as the one used to search for user-defined documentation or a homepage.

The log is displayed in the Inspector window in the "Info" tab. Before you can display the call log, you must first enable the call log in the settings of the TIA Portal or using an XML file.

See also

Conventions for the creation (Page 432)

Specifying settings with an XML file (Page 429)

Creating a homepage (Page 431)

Creating user-defined documentation (Page 436)

Calling user-defined documentation (Page 435)

General remarks on the information system (Page 283)

8.9.2 Specifying settings in the TIA Portal

You specify the following settings for user-defined documentation in the settings of the TIA Portal:

- Displaying the call log in the Inspector window
  A log of the call of user-defined documentation is displayed on the 'Info > General' tab of the Inspector window. The log helps you to adhere to the conventions for calling user-defined documentation.
- Search for user-defined documentation in a central file directory
  You can save user-defined documentation in a directory outside the current project directory in order, for example, to make documentation available across projects.
- Central directory for user-defined documentation
  You store cross-project documentation in the central file directory for user-defined documentation.

Note

XML configuration file takes precedence over the settings of the TIA Portal

If you use an XML configuration file and have specified settings for user-defined documentation there, the settings in the XML file take precedence. As soon as you refresh the XML configuration file or restart the TIA Portal, the settings from the XML file are applied. The settings that you have made in the TIA Portal lose their validity.

Procedure

To specify a central storage location for user help, follow these steps:

1. Select the "Settings" command in the "Options" menu.
2. Open the "General > General" area.
3. Navigate to the "User documentation" section.
4. Select the "Display call log for user-defined documentation" check box in order to display a log of the call of the user-defined documentation in the Inspector window.
5. Select the "Search for user-defined documentation in a central directory" check box in order to store user-defined documentation in a cross-project directory.
6. Specify the path to where you save the cross-project documentation in the "Central directory for user-defined documentation" field.

See also

Specifying settings with an XML file (Page 429)

8.9.3 Specifying settings with an XML file

As an alternative to settings in the TIA Portal, you can make settings for the user-defined documentation in an XML file. The XML file is the same file you use for integrating corporate libraries.

If you use an XML configuration file and have specified settings for user-defined documentation there, the settings in the XML file take precedence. As soon as you refresh the XML configuration file or restart the TIA Portal, the settings from the XML file are applied. The settings that you have made in the TIA Portal lose their validity.

You can set the following options in the XML configuration file:

- Displaying the call log in the Inspector window
  A log of the call of user-defined documentation is displayed on the 'Info > General' tab of the Inspector window. The log helps you to adhere to the conventions for calling user-defined documentation.
- Search for user-defined documentation in a central file directory
  You can save user-defined documentation in a directory outside the current project directory in order, for example, to make documentation available across projects.
- Central directory for user-defined documentation
  You store cross-project documentation in the central file directory for user-defined documentation.

Procedure

To specify settings for the user-defined documentation, follow these steps:

1. Create an XML file named "CorporateSettings.xml", if you are not yet using an XML configuration file for the integration of company libraries. If you are already using a configuration file, proceed with step 3.
   The configuration file must be saved with "UTF-8" coding.
2. Save the file in the following directory on your computer:
   C:\ProgramData\Siemens\Automation\Portal V14\CorporateSettings\
3. Enter the content listed below into the XML configuration file.
4. Adapt the attributes for display of the user-defined documentation. The meaning of the individual elements is available in the comments in the XML configuration file. Use the value "true" to activate a function. Use the value "false" to deactivate a function.

Content of the XML configuration file

The XML configuration file must have the following content:

XML

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
 <Settings.Settings ID="0">
  <ObjectList>
   <Settings.General ID="1" AggregationName="General">
    <!-- Here you find the settings for global company libraries, if available. -->
    <ObjectList>
     <Settings.UserDocumentation ID="2" AggregationeName="UserDocumentation">
      <!-- In the following section, you specify the values for display of the user-defined
documentation. -->
      <AttributeList>
       <!-- Activates or deactivates the display of the access log. -->
       <DisplayLogInformation>
        <Value>true</Value>
       </DisplayLogInformation>
      <!-- Activates or deactivates the search for user-defined documentation in a central
directory. -->
       <EnableLookupFromCentralStorageLocation>
        <Value>true</Value>
       </EnableLookupFromCentralStorageLocation>
       <!-- Specifies the central directory for user-defined documentation. -->
       <CentralStorageLocation>
        <Value>D:\CorporateDocumentation\UserDocumentation\</Value>
       </CentralStorageLocation>
      </AttributeList>
     </Settings.UserDocumentation>
    </ObjectList>
   </Settings.General>
  </ObjectList>
 </Settings.Settings>
</Document>
```

See also

Using user-defined documentation (Page 426)

Specifying settings in the TIA Portal (Page 428)

Calling user-defined documentation (Page 435)

Creating a homepage (Page 431)

Creating a configuration file for corporate libraries (Page 470)

8.9.4 Creating a homepage

You can design a homepage for user-defined documentation. The homepage can be an HTML page that you save either within a CHM file or in the directory of the relevant language. You

can also use other file formats approved for user-defined documentation. You design the homepage of the user-defined documentation outside the TIA Portal.

Procedure

To create a homepage, follow these steps:

1. Design a file in HTML format or in any other file format approved for user-defined help.
2. Name the file "Home".
3. Copy the file to the central directory for user-defined documentation on the hard disk or on a network drive:
   <Central directory for user-defined documentation>\<Folder for the respective language>
4. If the respective language folder does not exist yet, create the folder now.
   Alternative: If you are creating the homepage for a CHM file, place the homepage in the main directory of the CHM file.

Sample configuration for the homepage

Below you see the correct path for the following conditions:

- The user-defined documentation is in Spanish.
- The homepage is an HTML file.

The path for these conditions is as follows:

<Central directory for user-defined documentation>\es-ES\Home.html

See also

Specifying settings with an XML file (Page 429)

Using user-defined documentation (Page 426)

Conventions for the creation (Page 432)

Calling user-defined documentation (Page 435)

Displaying the call log (Page 436)

Creating user-defined documentation (Page 436)

8.9.5 Conventions for the creation

You must observe some conventions to ensure that user-defined documentation is called at the correct location:

- the user-defined documentation must be saved in the correct directory.
- The file name must be exactly the same as the object name in the TIA Portal.

To prevent malicious code from being executed on your computer, only file formats that are considered as relatively safe are permitted.

Supported file formats

Create the user-defined documentation in one of the following file formats:

- Microsoft Word (.docx)
- Microsoft Excel (.xlsx)
- Microsoft PowerPoint (.pptx and .ppsx)
- HTML pages (.htm or .html)
- Microsoft XPS (.xps)
- Rich Text Format (.rtf)
- Text documents (.txt)
- Compiled HTML help (.chm)
- PDF documents (.pdf)

You save the homepage of the user-defined documentation in HTML format or save the homepage within a CHM file.

NOTICE
Infection of the computer with malicious code If the user-defined documentation contains malicious code, it can infect your computer. Especially HTML pages and CHM files can contain malicious code. Make sure that the user-defined documentation comes from a trustworthy source. You should also use the standard security measures, such as the use of a firewall and an up-to-date virus scanner.

Directories for user-defined documentation

Save the user-defined documentation in one of the following directories:

- Project folder:
  UserFiles\UserDocumentation\<Folder for the respective language>\<Object category>
- Directory of a global library:
  UserFiles\UserDocumentation\<Folder for the respective language>\<Object category>
- Central directory on the hard drive or a network drive:
  <Central directory for user-defined documentation>\<Folder for the respective language>
  \<Object category>\

The user-defined documentation must be located in the suitable subfolder for the respective language. The table below shows the respective language folders for the user languages installed as default:

Language	Subfolder
German	\de-DE
English	\en-US
Spanish	\es-ES

Language	Subfolder
French	\fr-FR
Italian	\it-IT
Chinese	\zh-CN

The language folder must contain a separate subfolder for each object category. Create the corresponding subfolders for objects for which you are providing user-defined documentation. Always use the English designation of the object category. The table below shows the English designations of the most important object categories in the TIA Portal:

Object category	English designation
HMI screens	Screens
Organization blocks (OB)	Organization Blocks
Function blocks (FBs)	Function Blocks
Functions (FCs)	Functions
Data blocks	Data Blocks
Types in the library	Library Types
Master copies in the library	Master Copies
The project node in the project tree	Projects
All types of folders in the project tree, in the project library or in global libraries	Folders
All types of links in the project tree, for example, "Add new block", "Add new device", etc.	ShortCut
Libraries in the "Libraries" task card or in the library view	Libraries

If you are not sure of the English designation for an object category, change the user interface language of the TIA Portal to English. Alternatively, open the user-defined documentation for an object with <Shift+F1> and check in the call log which designation is expected for the object category.

Permitted file names

The file name must be exactly the same as the object name in the TIA Portal.

There are, however, restrictions for file names under Microsoft Windows. The same restrictions apply to the file system used to format the hard drive. The file name may only include certain characters and must not exceed a specific length. The restrictions for file names differ depending on the Windows version and the file system used for the hard drive.

To ensure that the help call works, read up on possible restrictions in the Microsoft Windows documentation.

Special features of CHM files

You store CHM files directly in the respective language folder. The folder for the respective object category must be included in the actual CHM file. Within the compiled CHM file, the names of the individual HTML files must also be exactly the same as the object names in the TIA Portal.

Note

Opening CHM files on network drives

If CHM files are saved on a network drive, the CHM files are not displayed correctly in more recent versions of Microsoft Windows. This behavior is determined by the security guidelines of the operating system. All versions of Microsoft Windows as of Windows Server 2003 SP1 are affected.

You can bypass the security guidelines by changing the registry database in Microsoft Windows.

To not compromise the security of your computer, save the CHM files only locally on your computer and do not change the registry database.

See also

Using user-defined documentation (Page 426)

Creating a homepage (Page 431)

8.9.6 Calling user-defined documentation

The user-defined documentation is opened in the language that is currently set as the user interface language. If there is no user-defined documentation available in the currently set user interface language, the English version of the user-defined documentation opens. If no user-defined documentation exists, a homepage is searched for.

Requirement

You have already saved user-defined documentation or a homepage according to the conventions.

Procedure

To open the user-defined documentation, follow these steps:

1. Select the object for which you want to display the user-defined documentation.
2. Press <Shift+F1>.
   The suitable user-defined documentation or the homepage opens.

See also

Using user-defined documentation (Page 426)

Opening the information system (Page 288)

Creating a homepage (Page 431)

Specifying settings with an XML file (Page 429)

Displaying the call log (Page 436)

8.9.7 Displaying the call log

Use the call log to check whether the user-defined documentation is connected correctly. The call log shows the directories in which the search is performed for user-defined documentation or a homepage. The call log also displays the names that the individual files must have in order to call the user-defined documentation.

Requirement

The call log is enabled in the settings of the TIA Portal or using an XML configuration file.

Procedure

To display the call log, follow these steps:

1. Open the "Info" tab in the Inspector window.
2. Open the "General" tab.
3. Select the object for which you want to call the help.
4. Press <Shift+F1>.
   If possible, the matching user-defined documentation or the homepage of the user-defined documentation is opened. In any case, you will be informed in the Inspector window about which user-defined documentation is opened. You may be shown the directories in which no user-defined documentation was found.

See also

Calling user-defined documentation (Page 435)

Creating a homepage (Page 431)

8.9.8 Creating user-defined documentation

You create user-defined documentation for individual elements within a project or global library outside the TIA Portal. You can create the user-defined documentation in all available user interface languages.

If you create the user-defined documentation as CHM file, the procedure for creating the help is somewhat different to the creation process for other file formats.

Note the information provided in chapter "Conventions for the creation (Page 432)".

Creating user-defined documentation as single file

To create user-defined documentation as a single file, follow these steps:

1. Create a file in a valid file format.
2. Name the file identically to the object for which you want to call the user-defined documentation.
   If you are offering help for a library type, for example, name the help file identical to the type.
3. Depending on whether you are creating the user-defined documentation for project contents or for contents of a global library, copy the file to one of the following storage locations:
   - project folder under "UserFiles\UserDocumentation\<Folder for the respective language>\<Object category>"
   - Directory of a global library under "UserFiles\UserDocumentation\<Folder for the respective language>\<Object category>"
   - Central directory on the hard drive or a network drive:
     <Central directory for user-defined documentation>\<Folder for the respective language>\<Object category>\

   If the respective language folder or the folder for the object category does not exist yet, create the required folders before copying the file.

Sample configuration for user-defined documentation

Below you see the correct path for the following conditions:

- The user-defined documentation is intended for a type in a global library.
- The user-defined documentation is in French.
- The type is called "commande de moteur".
- The user-defined documentation is supplied with the global library.
- The user-defined documentation is created in Microsoft PowerPoint format.

The path for these conditions is as follows:

<Folder of the global library>\UserFiles\UserDocumentation\fr-FR\Library Types\commande de moteur.pptx

Creating user-defined documentation as CHM file

To provide user-defined documentation in CHM format, follow these steps:

1. Create a folder in Windows Explorer for each object category for which you want to create user-defined documentation. Use the English designation for the object category.
2. Create an HTML file for each object for which you want to provide user-defined documentation. Name the HTML file identically to the object for which you want to call the user-defined documentation. If you want to provide user-defined documentation for a library type, for example, name the HTML file identically to the type.
3. Store the HTML files in the corresponding folders of the respective object category.
4. Use the Microsoft HTML Help Workshop to create the CHM file. Use the prepared folder structure with the HTML files.
5. Copy the CHM file to one of the following storage locations:
   - Project folder under "UserFiles\UserDocumentation\<Folder for the respective language>"
   - Directory of a global library under "UserFiles\UserDocumentation\<Folder for the respective language>"
   - Central directory on the hard drive or a network drive:
     <Central directory for user-defined documentation>\<Folder for the respective language>

   If the respective language folder does not exist yet, create the language folder before you copy the CHM file.

See also

Creating a homepage (Page 431)

Conventions for the creation (Page 432)

Using libraries

9

9.1 Library basics

Introduction

You can store library elements you want to reuse in libraries. For each project there is a project library that is connected to the project. In addition to the project library, you can create any number of global libraries that can be used over several projects. Since the libraries are compatible with each other, library elements can be copied and moved from one library to another. Libraries are used, for example, to create templates for blocks that you first paste into the project library and then further develop there. Finally, you copy the blocks from the project library to a global library. You make the global libraries available to other employees working on your project. The other employees continue to use the blocks and adapt them to their individual requirements as needed.

Both the project library and global libraries distinguish between two different types of library elements:

- Types
  Objects that are required to run user programs, for example, blocks, PLC data types (UDT), user-defined data types or faceplates are suitable as types. Types can be versioned and therefore support professional further development. Projects using the types can be updated as soon as new versions of the types are available.
- Master copies
  Almost every object can be saved as a master copy and pasted into the project again later. You can, for example, save entire devices with their contents or cover sheets for plant documentation as master copies.

Project library

Each project has its own library, the project library. In the project library, you store the objects you want to use more than once in the project. The project library is always opened, saved, and closed together with the current project.

Global libraries

In addition to the project library, you use global libraries if you want to use libraries over several projects. Global libraries exist in three versions:

- System libraries
  Siemens supplies global libraries for its own software products. These include off-the-peg functions and function blocks that you can use within your project. The supplied libraries cannot be changed. The supplied libraries are loaded automatically matching the project. If you edit the project in compatibility mode, only libraries that are compatible with the TIA Portal V14 are loaded.
- Corporate libraries
  Corporate libraries are made available centrally by your organization, for example, in a central folder on a network drive. The TIA Portal manages the corporate libraries automatically. As soon as a more recent version of an existing corporate library becomes available, you receive a prompt to update the corresponding corporate library to the more recent version.
- User libraries
  Global user libraries are independent of a specific project and can therefore be passed on to other users. Shared access to global user libraries is also possible, for example on a network drive, if all users open the global user library with write protection.
  Global user libraries from older versions of the TIA Portal that you created yourself can still be used. To continue using global user libraries from older versions of the TIA Portal, they must first be upgraded.

Comparing library elements

You can compare blocks and PLC data types (UDT) with the objects of a device. This allows you to determine, for example, whether certain blocks or PLC data types (UDT) have been used in a project and whether they have been modified.

See also

Overview of the "Libraries" task card (Page 443)

Overview of the library view (Page 447)

Overview of the library management (Page 451)

Basics on master copies (Page 472)

Basics on types (Page 476)

9.2 Using libraries in different TIA products

Description

You can use different TIA products (e.g. STEP 7 Professional, Safety, WinCC Advanced, etc.) within a TIA Portal version (e.g. V14 SP1). You can also open projects with project libraries and global libraries that you have created with one TIA product with a different TIA product at any time. However, the library elements can only be edited and used in the project if they are supported in the current project and when all necessary licenses and support packages are available. You cannot, for example, edit STL program blocks, with STEP 7 Basic.

If a master copy or a type or its versions are not supported, the block icon in the library and in the project tree is also marked with the icon for library elements that are not supported.

If a master copy or a type is marked as unsupported, the following functions are then available to you:

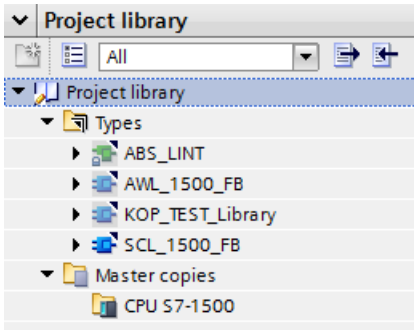
- rename
- delete
- move within the library
- copy to a different library, as long as this library also supports the library element

Even if a type is marked as unsupported, the following functions are also available to you:

- The "Go to type" function
- The library management
- "Clean up library" function
- Upgrade library as long as it does not contain a type version that is not supported
- Assign type version as long as it does not contain a type version that is not supported

Example

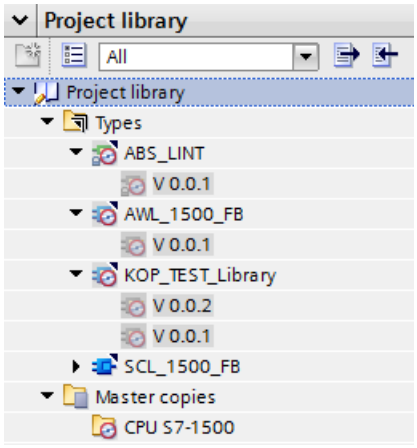
The example below shows you what it looks like when you open a project library from a more comprehensive product in a less comprehensive product:



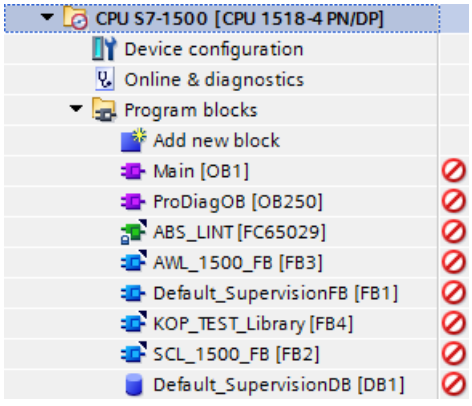
The project library was created with STEP 7 Professional and includes types that are specific to Professional.

When you open the STEP 7 Professional project with STEP 7 Basic you will see a note that the contained libraries cannot be edited without restrictions.

- Representation in the project library:



- Representation in the project tree:



All STEP 7 Professional specific types are marked accordingly as not supported and can only be edited with restrictions.

9.3 Using the "Libraries" task card

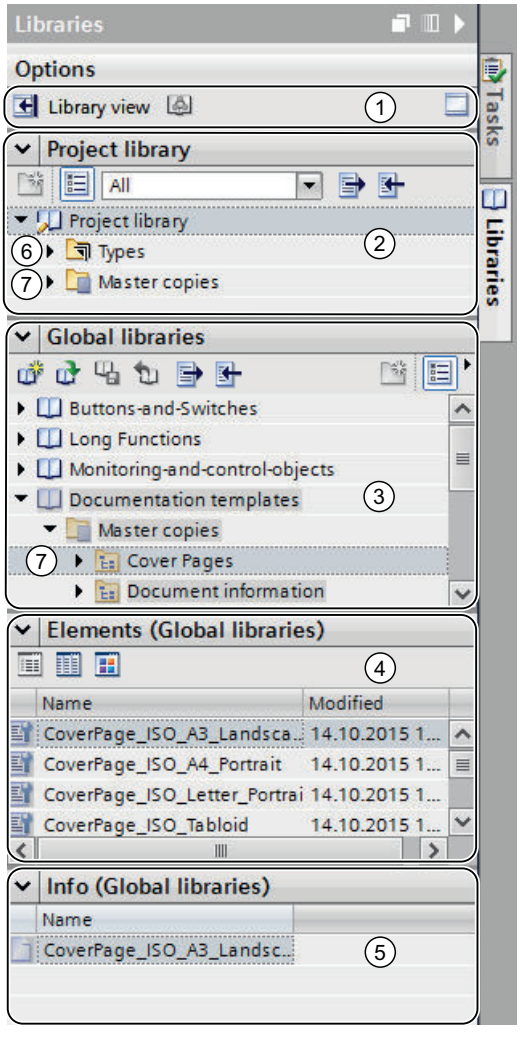
9.3.1 Overview of the "Libraries" task card

Function of the "Libraries" task card

The "Libraries" task card enables you to work efficiently with the project library and the global libraries.

Layout of the "Libraries" task card

The "Libraries" task card consists of the following components:



① "Library view" and "Library management" buttons
② "Project library" pane

③ "Global libraries" pane
④ "Elements" pane
⑤ "Info" pane
⑥ "Types" folder
⑦ "Master copies" folder

"Library view" button

The "Library view" button is used to change to the library view. The "Libraries" task card and the project tree are hidden with this action.

See also: Using the library view (Page 447)

"Project library" pane

In the "Project library" pane, you can store the library elements that you want to use more than once in the project.

"Global libraries" pane

In the "Global libraries" pane, you manage the global libraries whose library elements you want to reuse over several projects.

The "Global libraries" pane also lists libraries that were shipped together with the products you purchased. These libraries provide you with ready-made functions and function blocks, for example. The supplied global libraries cannot be edited.

"Elements" pane

In this pane, you can display the contents of folders in the library. The "Elements" pane is not displayed by default. If you want to display the "Elements" pane, you have to enable it first. Three view modes are available in the "Elements" palette:

- Details mode
  The properties of folders, master copies and types are shown in table form in details mode.
- List mode
  In list mode, the contents of folders are listed.
- Overview mode
  In overview mode, the contents of folders are displayed with large symbols.

See also: Using the element view (Page 445)

"Info" pane

You can display the contents of the library elements in the "Info" pane. The individual versions of types and the last revision date of the version are also displayed.

"Types" folder

In the "Types" directories, you can manage types and type versions of library elements that you use as instances in the project.

See also: Using types (Page 476)

"Master copies" folder

In the "Master copies" directories, you can manage master copies of library elements that you can use as copies in the project.

See also: Using master copies (Page 472)

See also

Library basics (Page 439)

Comparing library elements (Page 520)

9.3.2 Using the element view

Introduction

When you open the "Libraries" task card the first time, the "Project library" and "Global libraries" palettes are opened and the "Info" palette is closed. You can display the "Elements" palette if needed.

The elements view shows the elements of the selected library. Three view modes are available in the elements view:

- Details
  The properties of folders, master copies and types are shown in table form in details mode.
- List
  In list mode, the contents of folders are listed.
- Overview
  In overview mode, the contents of folders are displayed with large symbols.

The "Info" palette shows the contents of the selected library element. If you select a type in the elements view, for example, the type versions are displayed in the "Info" palette.

Requirement

The "Libraries" task card is displayed.

Procedure

To use the element view, follow these steps:

1. Click "Open or close the element view" in the "Project library" or "Global libraries" pane.
2. To change the view mode from the details view to the list mode or overview mode, click the corresponding icon on the toolbar.

See also

Library basics (Page 439)

Overview of the "Libraries" task card (Page 443)

Comparing library elements (Page 520)

Using global libraries (Page 458)

9.4 Using the library view

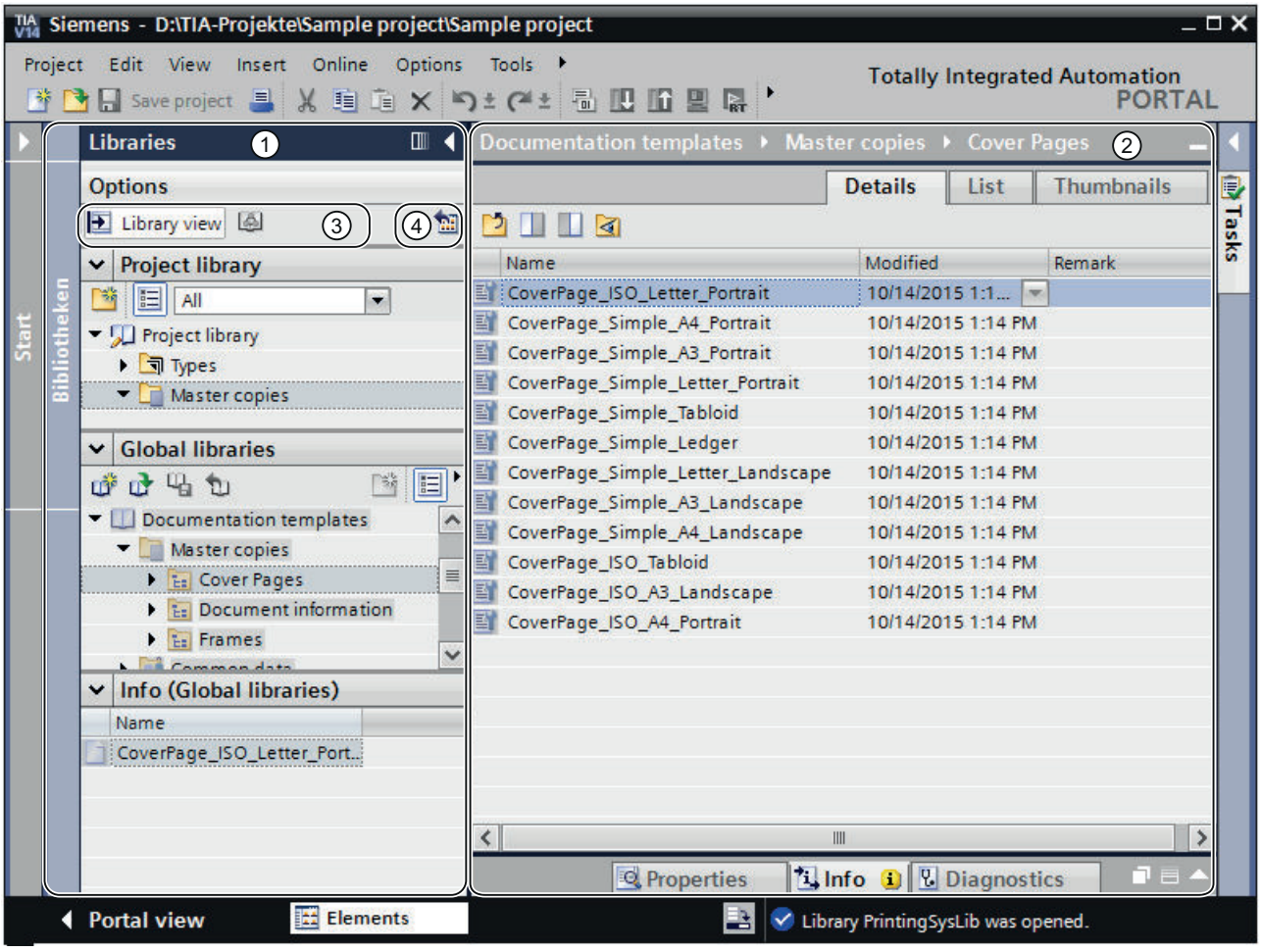
9.4.1 Overview of the library view

Function of the library view

The library view combines the functionality of the "Libraries" task card and the overview window. In the library view, the elements of a library are displayed in various views. In the details view, for example, you see additional properties of the individual elements. You can also edit and version the types in the library view.

Layout of the library view

The following figure shows the components of the library view:



① Library tree
② Library overview
③ "Open/close the Library view" button and "Open library management"
④ "Open or close library overview" button

Library tree

The library tree is similar to the "Libraries" task card, apart from a few minor differences. In contrast to the task card, there is no "Elements" palette, because the elements are displayed in the library overview. In addition, you can close the library view in the library tree, or open and close the library overview.

See also: "Libraries" task card (Page 443)

Library overview

The library overview corresponds to the overview window and displays the elements of the currently selected object in the library tree. You can display the elements in three different views:

- Details view
  The objects are displayed in a list with additional information, such as the date of the last change.
- List view
  The objects are displayed in a simple list.
- Icon view
  The objects are displayed as icons according to category.

In addition, you can perform the following actions in the library overview:

- Renaming elements
- Deleting elements
- Copying elements
- Moving elements
- Editing type instances
- Versioning types
- WinCC only: Editing faceplates and HMI user data types

See also: Overview window (Page 262)

See also

Basics on master copies (Page 472)

Basics on types (Page 476)

Opening and closing the library view (Page 449)

Library basics (Page 439)

Comparing library elements (Page 520)

9.4.2 Opening and closing the library view

The library view is opened automatically in some cases, for example, when you edit the test instance of a type or when you edit faceplates and HMI user data types. You can also open the library view manually.

Opening the library view

To open the library view manually, follow these steps:

1. Open the "Libraries" task card.
2. Click the "Open/close library view" button in the "Libraries" task card.
   The library tree opens. The "Library" task card and the project tree are closed.
3. If the library overview is not displayed, click the "Maximizes/minimizes the elements" button in the library tree.
   The library overview opens.

Exit library view

To exit the library view, follow these steps:

1. Click the "Close/open library view" button in the library tree.
   The library tree closes. The "Libraries" task card and the project tree are opened.

See also

Overview of the library view (Page 447)

Library basics (Page 439)

Comparing library elements (Page 520)

Using the "Libraries" task card (Page 443)

Using global libraries (Page 458)

9.5 Using library management

9.5.1 Overview of the library management

Function of the library management

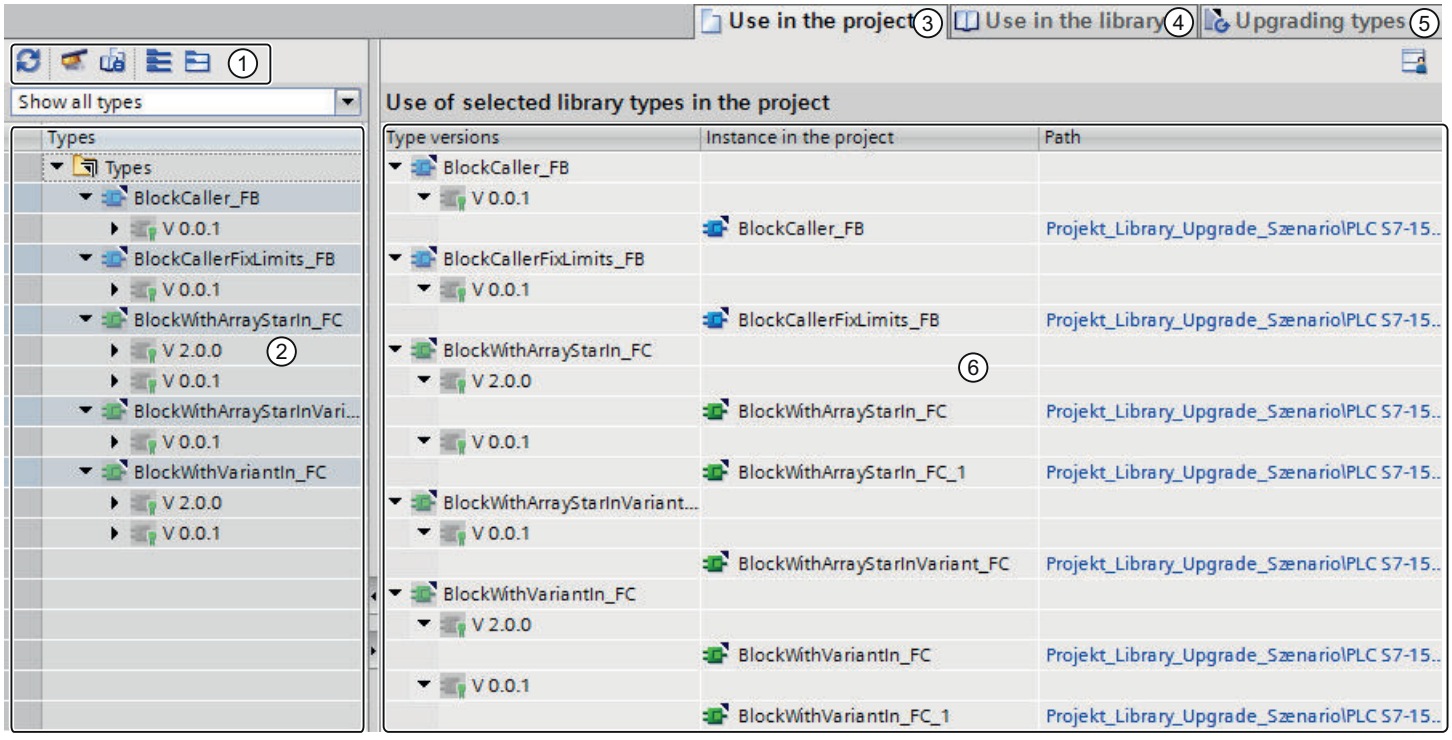
Master copies and types with dependencies to other library elements are subject to some functional restrictions. They cannot be deleted, for example, as long as dependencies still exist. This prevents other library elements from becoming useless. The library management is used to identify the dependencies and to create an overview of the work progress.

The library management offers the following functions:

- Display of the correlations of types and master copies
  If a type is referenced in other types or master copies, the correlations are displayed in the library management. You will also be able to see which library elements reference a type or a master copy.
- Display of points of use of types in the project
- Upgrading types
- Using filters to narrow down the displayed types

Layout of the library management

The following figure shows the components of the library management:



① Toolbar of the library management
② "Types" area
③ "Use in the project" tab
④ "Use in the library" tab
⑤ "Upgrading types"
⑥ "Uses" window

Toolbar of the library management

You can perform the following tasks in the toolbar of the library management:

- Update uses
  If the project was changed, you can update the view of the library management.
- Clean up library
  By cleaning up a library, you delete all types and type versions that are not linked to an instance in the project.
- Harmonize project
  By harmonizing a project, you adapt the names and the path structures of type uses in the project to the corresponding names and path structures of the types within a library.
- Collapse all
  Hides all subentries of the uppermost node in the "Types" area. The lower-level elements, such as types and individual type versions, are no longer displayed.
- Expand all
  Expands all lower-level elements in the "Types" area. The lower-level elements, such as types and type versions, are displayed in full.

"Types" area

The "Types" area displays the contents of the folder that you have selected in the library view. For each type, the types that it references are displayed. You can expand or collapse all type nodes by using the buttons in the toolbar of the "Types" area.

You can also filter the view with the "Filter" drop-down list. You will find additional information on the specific filter settings here: Filtering the display of types (Page 454)

"Use in the project" tab

The "Use in the project" tab is used to show the instances of type versions and their respective point of use in the project. When you select an instance, you can show the cross references of the instance in the project in the Inspector window.

You can find additional information here:

Displaying cross references of an instance (Page 456)

Displaying instances in the project (Page 455)

"Use in the library" tab

The "Use in the library" tab is used to show all points within the library at which a type or a master copy is used.

You can find additional information here: Displaying relations to other library objects (Page 457)

"Upgrading types"

In the "Upgrading types" tab, you can identify all types, reference changed types, and upgrade.

You can find more information on upgrading projects here:

Upgrading types in an existing project (Page 506)

Example of the updating and upgrading of types (Page 509)

See also

Opening library management (Page 454)

Library basics (Page 439)

9.5.2 Opening library management

Procedure

To open the library management, follow these steps:

1. Open the library view.
2. Select a type or any folder that contains types.
3. Select the "Library management" command from the shortcut menu.

Or:

1. Click the "Types" folder in the project library.
2. Select the "Library management" command from the shortcut menu.

Result

The library management opens and the types are displayed with their versions.

See also

Overview of the library management (Page 451)

9.5.3 Filtering the display of types

You can use filters to obtain a better overview of types in extensive libraries. A filter provides you with the option of limiting the displayed types. The following filters are available:

- Types with pending changes
  The filter displays all types that have a version "In test" or "In work".
- Released types
  The filter displays all released types that do not have versions "In test" or "In work".
- Types with multiple versions
  The filter displays all types that have more than one version.

- Types not used in the project
  The filter displays all types which have no instance in the project.
- Types with new versions for upgrading other types
  The filter displays all types that are relevant for upgrading other types. The types must also meet the following criteria:
  - The types have more than one version.
  - All versions of the respective type have been released.

Requirement

The library management is open.

Procedure

To filter the displayed types, follow these steps:

1. In the "Types" area, select the folder whose contents you want to filter.
2. Select the required filter from the drop-down list.
   In the "Types" area, types are displayed that correspond to the selected filter criteria.

See also

Overview of the library management (Page 451)

State of type versions (Page 478)

Creating a test version of a type (Page 487)

Editing test version of a type (Page 489)

Creating an editing version of a type (Page 489)

9.5.4 Displaying instances in the project

In the library management, you can have the instances of all versions of a type or an individual type version displayed. You can jump directly to each instance in the project.

Requirement

The library management is open.

Procedure

To display the instances of a type or its versions, follow these steps:

1. In the "Types" area select the required folders, types or versions.
2. Open the "Use in the project" tab.
   The instances in the project are displayed for each type version. The "Path" column shows the path at which the respective instance is located in the project.
3. Click the path to jump directly to the respective instance in the project tree.
   The library management is hidden and the instance is selected in the project tree.

See also

Displaying cross references of an instance (Page 456)

Using types of the project library (Page 485)

Overview of the library management (Page 451)

Displaying relations to other library objects (Page 457)

9.5.5 Displaying cross references of an instance

You can display the cross references of an instance without exiting the library management.

Requirement

The library management is open.

Procedure

To display the cross references of an instance in the project, follow these steps:

1. In the "Types" area, select a type version whose instances you want to display.
2. Select the instance of the required type version in the "Uses in the project" tab.
3. Open the "Info > Cross-references" tab in the inspector window.
   The cross reference of the instance are displayed in the project.

See also

Using cross-references (Page 425)

Overview of the library management (Page 451)

Displaying instances in the project (Page 455)

9.5.6 Displaying relations to other library objects

You can display relations between individual library objects in the library management. The references of the individual type versions to the other library objects are automatically displayed in the "Types" area. In the "Uses in the library" tab, you can also view the other library objects in which the respective type version is referenced.

Requirement

The library management is open.

Procedure

To display the other library objects from which a type version is referenced, follow these steps:

1. In the "Types" area select the required folders, types or versions.
2. Open the "Uses in the library" tab.
   In the "Uses" window, you can now see which other library objects are referenced by the individual type versions.
3. To jump to the referenced library object, click on the respective path in the "Path" column.

See also

Displaying instances in the project (Page 455)

Overview of the library management (Page 451)

Referencing of objects within a block (Page 483)

9.6 Using global libraries

9.6.1 Creating a global library

Requirement

The "Libraries" task card is displayed or the library view is open.

Procedure

To create a new global library, follow these steps:

1. Click the "Create new global library" icon in the toolbar of the "Global libraries" palette or select the command "Global libraries > Create new library" in the "Options" menu.
   The "Create new global library" dialog opens.
2. Specify the name and the storage location for the new global library.
3. In the "Version" drop-down list, select the library in which you want to create the library version.
   If you select "V14", the library is compatible with the TIA Portal V14 without installed Service Pack 1.
4. Confirm your entries with "Create".

Result

The new global library is generated and pasted into the "Global libraries" palette. A folder with the name of the global library is created in the file system at the storage location of the global library. The actual library file is given the file name extension ".al14".

See also

Library basics (Page 439)

Opening a global library (Page 459)

Displaying properties of global libraries (Page 462)

Saving a global library (Page 463)

Closing a global library (Page 464)

Deleting a global library (Page 465)

9.6.2 Compatibility of global libraries

Libraries from the TIA Portal V13 or earlier

Global libraries that were created with TIA Portal V13 or earlier cannot be used in the current version of the TIA Portal. First, upgrade the libraries with version V13 SP1 of the TIA Portal to the current library version. You can then open the library and upgrade once again.

Libraries from the TIA Portal V13 SP1

You can use the TIA Portal to open global libraries that were created with TIA Portal V13 SP1. When you open the library, you are prompted to upgrade the library to the current version of the TIA Portal. Once the global library is upgraded, it cannot be opened with earlier versions of the TIA Portal.

Libraries from the TIA Portal V14

It is possible to open global libraries from the TIA Portal V14 in compatibility mode. The library remains compatible with the TIA Portal V14 until you save an element that can only be used with the TIA Portal V14 SP1. As soon as you save an element with functionality from the TIA Portal V14 SP1 in a global library with the library version V14, a prompt appears with the request to upgrade the library.

Backward compatibility of the current library version

Global libraries saved with the library format of the current TIA Portal product version are not backward compatible with older versions due to their enhanced functionality. Global libraries in the current library format can only be used in connection with TIA Portal V14 SP1 projects.

See also

Opening a global library (Page 459)

9.6.3 Opening a global library

Global libraries can be further developed centrally and used over several projects. Several persons can open a global library simultaneously from a central storage location, provided all users open the global library in write-protected mode.

Requirement

The "Libraries" task card is displayed or the library view is open.

Procedure

To open a global library, follow these steps:

1. Click the "Open global library" icon in the toolbar of the "Global libraries" pane or select the command "Global libraries > Open library" in the "Options" menu.
   The "Open global library" dialog box opens.
2. Select the global library you want to open. Library files are identified by the file name extension ".al[version number]". This means that global libraries that were saved with the current TIA Portal product version have the file name extension ".al14".
3. Write protection is activated for the library. If you want to modify the global library, disable the "Open as read-only".
4. Click "Open".
   If the library is saved in the library version for TIA Portal V14 or V14SP1, the "Global library" is opened and pasted into the "Global libraries" pane. The "Upgrade global library" dialog opens if you have selected a global library from TIA Portal V13 SP1. Upgrade the library to the current version of the TIA Portal in this case.

Result

The global library is displayed in the "Global libraries" pane.

If the global library was opened write-protected, it and all its subfolders are grayed out.

See also

Retrieving global libraries (Page 467)

Compatibility of global libraries (Page 459)

Library basics (Page 439)

Creating a global library (Page 458)

Displaying properties of global libraries (Page 462)

Saving a global library (Page 463)

Closing a global library (Page 464)

Deleting a global library (Page 465)

9.6.4 Upgrading global libraries

Where necessary, global libraries in compatibility mode can be upgraded manually for TIA Portal V14.

Requirement

A global library is open in compatibility mode for TIA Portal V14.

Procedure

To manually upgrade global libraries to the current library version, follow these steps:

1. Right-click on the global library you want to upgrade.
2. Select the "Upgrade library" command in the shortcut menu.
   The "Upgrade" dialog opens.
3. Confirm with "Yes".
   The library is upgraded.

9.6.5 Defining library languages

You can define various library languages for global libraries and select an editing language for all comments from them. The selection of the library languages is valid for all library elements (types, type versions and master copies) within the global library.

The comments of the types and their type versions can be translated into multiple library languages and managed in the global library. The selected editing language specifies the languages in which the comments are displayed. The comments of a type can be edited at any time. As soon as the version of a type is released, it comments can no longer be edited and are displayed in ready-only mode.

If you change the editing language of a global library, the comments of all types are displayed in the newly selected editing language, provided translations are available.

Requirement

The "Libraries" task card is displayed or the library view is open.

Procedure

To define the library languages of a global library, follow these steps:

1. Click the "Open global library" icon in the toolbar of the "Global libraries" pane or select the command "Global libraries > Open library" in the "Options" menu.
   The "Open global library" dialog box opens.
2. Select the global library you want to open.
3. Disable the write protection of the global library by removing the check mark for "Open write-protected".
   If you do not disable the "Open write-protected" option, you can see the selection of library languages in the global library, but you cannot change these.
4. Click "Open".
   The global library is displayed.
5. Open the "Languages & Resources" folder.
6. Double-click the "Library languages" entry or select "Open" from the shortcut menu.

Result

The library languages are displayed in the Library view.

9.6.6 Displaying properties of global libraries

Global libraries contain properties for describing the respective library in more detail. Properties include the following:

- General information about the library
  This includes the following information: creation time, author, file path, file size, copyright, etc. Many of the attributes can be changed.
- Library history
  The library history contains an overview of the migrations performed. Here you can also call the log file for the migrations. The library history also contains information on updates of the global library.
- Support packages in the library
  You can display an overview of additional software. The additional software is needed to process all devices of the project.
- Software products in the library
  You can display an overview of all installed software products needed for the project.

Requirement

The "Libraries" task card is displayed or the library view is open.

Procedure

To display the properties of a global library, follow these steps:

1. Right-click the global library whose properties you want to display.
2. Select the "Properties" command in the shortcut menu.
   A dialog containing the properties of the global libraries opens.
3. Select the properties in the area navigation that you want to have displayed.

See also

Opening a global library (Page 459)
Library basics (Page 439)
Creating a global library (Page 458)
Saving a global library (Page 463)
Closing a global library (Page 464)
Deleting a global library (Page 465)

9.6.7 Displaying logs of global libraries

Logs are created when you update or clean up global libraries or assign a shared version to several types. The logs list all changes you have made to the global library. The logs are stored together with the global library and are always available once you have opened the global library.

Procedure

To open the logs of a global library, follow these steps:

1. Open the global library in the "Libraries" task card or in the library view.
2. Open "Common data > Logs" in the lower-level folder.
3. Double-click the required log.
   The log opens in the work area.

See also

Updating library with the types of another library (Page 505)

9.6.8 Saving a global library

After you have changed a global library, you need to save it. You can save a global library under another name using the "Save library as" command.

Note

Backward compatibility with older versions of the TIA Portal

Note that global libraries can no longer be opened in older versions of the TIA Portal once they have been saved in the current version.

Requirement

The "Libraries" task card is displayed or the library view is open.

Saving changes

To save a global library, follow these steps:

1. Right-click on the global library you want to save.
2. Select the "Save library" command in the shortcut menu.

Saving a global library under another name

To save a global library under another name, follow these steps:

1. Right-click the global library that you want to save under a different name.
2. Select the "Save library as" command in the shortcut menu.
   The "Save global library as" dialog opens.
3. Select the storage location and enter the file name.
4. Confirm your entries with "Save".
   The library is saved in the specified location under the new name. The original library is retained.

See also

Working with archives of global libraries (Page 466)

Archiving global libraries (Page 466)

Library basics (Page 439)

Creating a global library (Page 458)

Opening a global library (Page 459)

Displaying properties of global libraries (Page 462)

Closing a global library (Page 464)

Deleting a global library (Page 465)

9.6.9 Closing a global library

Global libraries are independent of projects. This means that global libraries are not closed together with your project. You must therefore close global libraries explicitly.

Requirement

The "Libraries" task card is displayed or the library view is open.

Procedure

To close a global library, follow these steps:

1. Right-click on the global library you want to close.
2. Select the "Close library" command in the shortcut menu.
   If you have made changes to the global library, select whether or not you want to save the changes.
   The global library is closed.

See also

Creating a global library (Page 458)

Opening a global library (Page 459)

Displaying properties of global libraries (Page 462)

Saving a global library (Page 463)

Library basics (Page 439)

Deleting a global library (Page 465)

9.6.10 Deleting a global library

If you no longer require a global library, you can delete it. Libraries supplied by Siemens cannot be deleted.

Requirement

The "Libraries" task card is displayed or the library view is open.

Procedure

To delete a global library, follow these steps:

1. Right-click the global library you want to delete.
2. Select the "Delete" command in the shortcut menu.
3. Click "Yes" to confirm.

Result

The global library is removed from the "Global libraries" palette. The entire library for the global library is deleted from the file system.

See also

Library basics (Page 439)

Creating a global library (Page 458)

Opening a global library (Page 459)

Displaying properties of global libraries (Page 462)

Saving a global library (Page 463)

Closing a global library (Page 464)

9.6.11 Archiving and disabling global libraries

9.6.11.1 Working with archives of global libraries

If you want to back up global libraries on an external hard drive or send them via e-mail, for example, use the archiving function to reduce the storage space of the library.

Options for reducing the size of the project

There are two ways to reduce the space required to store global libraries:

- Creating a compressed archive of global libraries
  Archives of global libraries are compressed files, each containing an entire global library, including the entire folder structure of the library. Before the directory with the global library is compressed into the archive file, all files are reduced to their essential components to further decrease the storage space. Compressed archives of global libraries are therefore well suited for sending via e-mail.
  Compressed archives of global libraries of the current product version have the file extension ".zal14". Archives from the TIA Portal V13.x have the file extension ".zal13".
  To open a compressed archive of a global library, retrieve the archive. The archive file is extracted to a location you have selected with the entire folder structure and all files.
- Minimizing global libraries
  You can skip additional compression in an archive file, and instead create a copy of the global library directory. The included files can be reduced to the essential elements. This minimizes the required storage space. The full functionality of the global library is retained and the global library can be loaded as usual.
  Minimized global libraries are especially well suited for archiving, for example, on an external medium.

See also

Archiving global libraries (Page 466)

Retrieving global libraries (Page 467)

9.6.11.2 Archiving global libraries

You reduce the storage space of a global library by packing it into a compressed file. You can also reduce the storage space by saving the global library reduced to the essential elements. You can do both of these with the archiving function for global libraries.

Requirement

The global library is loaded.

Procedure

To archive a global library, follow these steps:

1. Select the global library that you want to archive.
2. Right-click the global library and select the "Archive" command from the shortcut menu. The "Archive global library as..." dialog opens.
3. Select the directory where you want to save the archive file or the new directory of the global library.
   The directory may not be located in a project directory or within the directory of a global library.
4. Select the file type from the "File type" drop-down list:
   - Global libraries archive, if you want to create a compressed file of the library.
   - Minimized global library, if you only want to create a copy of the library directory with minimal storage requirement.
5. Enter a file name in the "File name" field if you are creating an archive file. If you are creating a minimized global library, enter the name of the new library directory to be created in the "File name" box.
6. Click "Save".

Result

A compressed file with the extension ".zal14" is generated.

The file includes the complete directory of the global library. The individual files of the global library are also reduced to the essential components in order to save space.

If you have minimized the global library, only a copy of the original directory of the global library is created at the required location. The files contained within it were reduced to their essential components in order to save space.

See also

Working with archives of global libraries (Page 466)

Retrieving global libraries (Page 467)

9.6.11.3 Retrieving global libraries

Before you can use an archived global library, you have to retrieve it. The global library is extracted and then opened in the TIA Portal.

Procedure

To extract the archive of a global library, follow these steps:

1. Select the "Global libraries > Retrieve library" command in the "Options" menu. The "Retrieve archived global library" dialog opens.
2. Select the archive file.

3. Select the check box "Open read-only" if you want to load the global library write-protected.
4. Click "Open".
5. The "Find folder" dialog opens.
6. Select the target directory to which the archived global library should be extracted.
7. Click "OK".

Result

The global library is extracted to the selected directory and opened immediately.

See also

Working with archives of global libraries (Page 466)
Archiving global libraries (Page 466)
Opening a global library (Page 459)
Compatibility of global libraries (Page 459)

9.6.12 Using global corporate libraries

9.6.12.1 Basics of corporate libraries

Introduction

Corporate libraries are global libraries made available by an administrator and assigned to the TIA Portal. The administrator can assign new libraries or change libraries at any time. New libraries are automatically loaded in the TIA Portal following your confirmation. As soon as more recent versions of corporate libraries are available, the existing corporate libraries are automatically updated to the more recent version, also following your confirmation.

The corporate libraries are located just like normal global libraries in the "Global libraries" palette of the "Libraries" task card.

Providing corporate libraries

Corporate libraries can be saved in any directory on the hard drive of the computer or on a network drive. To include corporate libraries in the TIA Portal, use an XML file. The XML file includes the directories and names of the assigned corporate libraries. The XML file must be saved in the following directory on the hard drive of the computer:

`C:\ProgramData\Siemens\Automation\Portal V14\CorporateSettings\`

The XML file must be named "`CorporateSettings.xml`".

You either copy the configuration file yourself to the corresponding directory or the configuration file is assigned to you through the company network. The valid configuration is

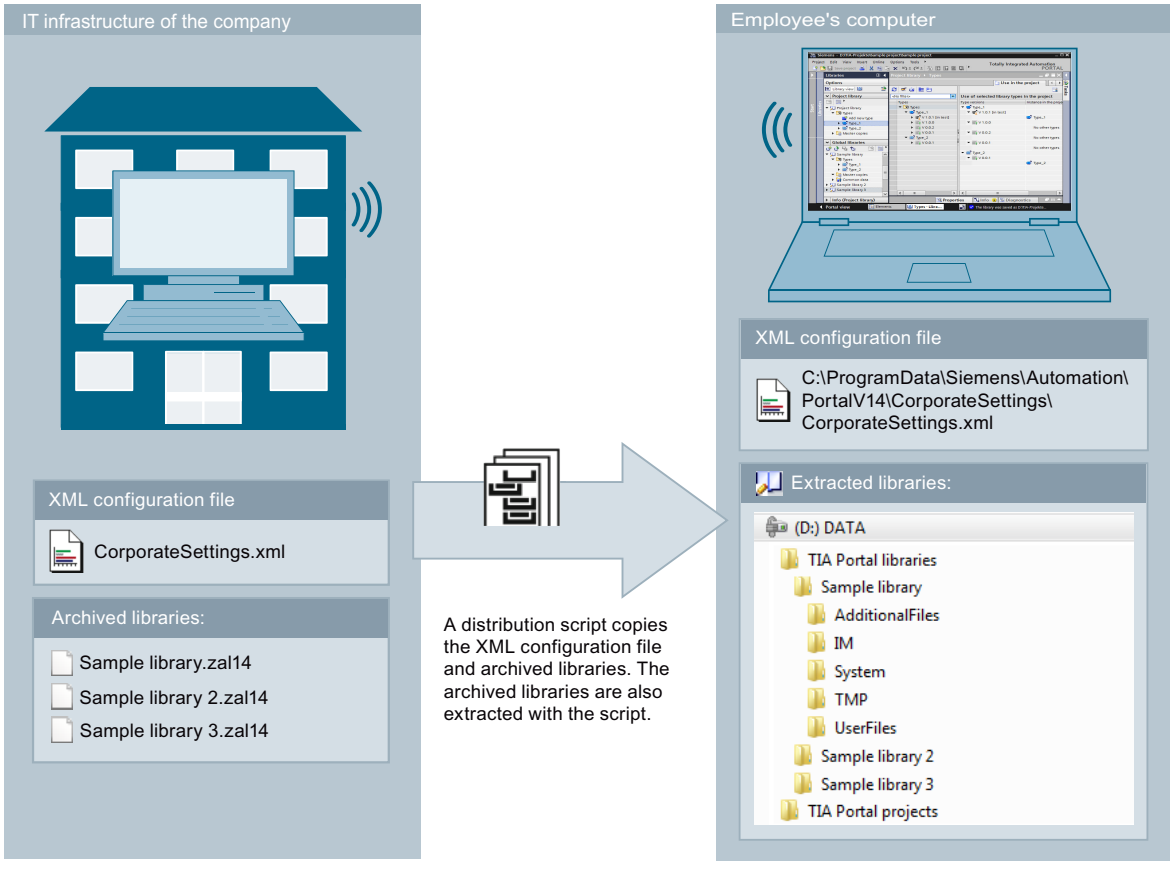
automatically applied when the TIA Portal is started. When the TIA Portal is started, it continuously monitors the directory for configuration files. If the configuration file has changed, you receive a prompt to apply the new configuration. You can reject this prompt twice. You will always receive the next prompt three hours later. You must apply the new configuration at the third prompt. You receive a changed configuration file from the project administrator, for example, if corporate libraries have been added or deleted.

Options as project administrator

You can automatically assign the configuration file or the corporate libraries to the computers of the team members or distribute updates to the team members. This function is not part of the TIA Portal and requires a corresponding IT infrastructure in your company. If you want to administer the configuration file centrally, discuss this approach with the IT managers at your company.

The archive function for global libraries can be used to pass corporate libraries to employees of the project team. However, you must retrieve corporate libraries prior to their use in the TIA Portal.

The following figure shows an example of how you can distribute corporate libraries to individual employees within your organization:



See also

Working with archives of global libraries (Page 466)

Archiving global libraries (Page 466)

Retrieving global libraries (Page 467)

9.6.12.2 Creating a configuration file for corporate libraries

You use a configuration file in XML format to make the corporate libraries available in the TIA Portal. The configuration file includes the directories and file names of the libraries to be loaded. Below you will learn how you create the XML configuration file and where to save it.

Procedure

To provide a configuration file for corporate libraries, follow these steps:

1. Create an XML configuration file with the content listed below. Use the coding "UTF-8".
2. Replace the paths mentioned in the example in the "<CorporateLibraryPaths>" section below with the paths where your libraries are stored.
3. Save the XML file under the name "CorporateSettings.xml".
4. Save the file in the following directory on your computer:
   C:\ProgramData\Siemens\Automation\Portal V14\CorporateSettings\

Content of the XML configuration file

The XML configuration file must have the following content:

XML

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
 <Settings.Settings ID="0">
  <ObjectList>
   <Settings.General ID="1" AggregationName="General">
    <AttributeList>
     <CorporateLibraryPaths>
     <!-- Example for two entries -->
     <Item>D:\CorporateLibraries\Corporate_Library_1.al14</Item>
     <Item>D:\CorporateLibraries\Corporate_Library_2.al14</Item>
     <!-- Here you enter additional global libraries, if any. -->
     </CorporateLibraryPaths>
    </AttributeList>
   </Settings.General>
  </ObjectList>
 </Settings.Settings>
</Document>
```

Result

As soon as you have saved the XML configuration file in the respective directory, you will receive a prompt in the TIA Portal to load the corporate libraries.

See also

Specifying settings with an XML file (Page 429)

9.7 Using master copies

9.7.1 Basics on master copies

Master copies are used to create standardized copies of frequently used elements. You can create as many elements as needed and insert them into the project based on a master copy. The elements inherit the properties of the master copy.

You store master copies either in the project library or in a global library. Master copies in the project library can only be used within the project. When you create the master copy in a global library, it can be used in different projects.

The following elements can be created as master copies in the library, for example:

- Devices with their device configuration
- Tag tables or individual tags
- Instruction profiles
- Watch tables
- Elements from the documentation settings, for example, cover sheets and frames
- Blocks and groups containing multiple blocks
- PLC data types (UDTs) and groups which contain multiple PLC data types
- Text lists
- Alarm classes
- Technology objects

In many cases, the objects you add as master copy contain additional elements. A CPU, for example, can contain blocks. If the included elements are uses of a type version, the used versions of the types are automatically created in the library. The elements contained therein are then used as an instance and linked to the type.

See also

Adding master copies (Page 473)

Using master copies (Page 475)

Basics on types (Page 476)

Filtering master copies (Page 474)

9.7.2 Adding master copies

If you want to use objects multiple times, copy them as master copies in the project library or in a global library. You can choose from the following methods for creating master copies:

- You select one or more elements and create individual master copies from them
- You select multiple elements and create a single master copy from them that contains all selected elements.

Requirement

- The "Libraries" task card is displayed.
- If you add a device as a master copy, this device meets the following requirements:
  - The device is compiled and consistent.
  - The device contains no test instance of a type.
- If you add the master copy to a global library, the global library is opened with write permission.

Creating master copies from one or more elements

To create a master copy for one or more elements, follow these steps:

1. Open the library in the "Libraries" task card.
2. Select the desired elements.
3. Using a drag-and-drop operation, move the elements to the "Master copies" folder or any subfolder of "Master copies".

Alternative:

1. Select the desired elements.
2. Copy the elements to the clipboard and paste the elements at the required location. Each element is pasted in the library as a master copy. A type is created automatically in each case from any objects contained (such as referenced blocks).

Creating a single master copy from multiple elements

To create a single master copy for all elements from multiple elements, follow these steps:

1. Open the library in the "Libraries" task card.
2. Copy to the clipboard the elements that you want to create as master copies.
3. Right-click on the "Master copies" folder or any of its subfolders in the library.
4. In the shortcut menu, select "Paste as a single master copy" command.

Alternative:

1. Select the desired elements.
2. Using a drag-and-drop operation, move the elements to the "Master copies" folder or any subfolder of "Master copies". Meanwhile, keep the <Alt> key pressed.
   The elements are pasted in the library as a single master copy. The single master copy contains all selected elements. A type is created automatically in each case from any objects contained (such as referenced blocks).

Note

Avoiding complex structures of master copies

To prevent name conflicts and conflicts regarding the folder structure during subsequent use of master copies, avoid complex master copies. Complex master copies are, for example, master copies that consist of multiple elements and nested folders.

See also

Basics on master copies (Page 472)

Using master copies (Page 475)

Library basics (Page 439)

Adding types to the project library (Page 481)

9.7.3 Filtering master copies

To make it easier to keep track of a large number of master copies, you can filter the display according to the type of master copy.

Requirement

The "Libraries" task card is displayed or the library view is open.

Procedure

To filter the view, follow these steps:

1. Open the "Master copies" folder in the project library or a global library.
2. In the drop-down list of the toolbar, select the type of objects you want to display under "Master copies".

Result

Only the selected type of master copies is displayed. You can set the filter to "All" to return to an unfiltered view.

See also

Library basics (Page 439)

Creating folders in a library (Page 515)

Basics on master copies (Page 472)

Using master copies (Page 475)

Using the element view (Page 445)

9.7.4 Using master copies

Master copies are contained either in the project library or in a global library. You can paste one or more master copies in the project at the same time. If you insert multiple master copies at the same time, ensure that all master copies are compatible with the desired point of use.

Requirement

The "Libraries" task card is displayed.

Procedure

To paste a master copy into the project, follow these steps:

1. In a library, open the "Master copies" folder or any subfolder of "Master copies".
2. Using a drag-and-drop operation, move the desired master copies or complete folders to the point of use.

Or:

1. Open the element view.
2. Using a drag-and-drop operation, move the desired master copies or whole folders from the "Elements" pane to the point of use.

Result

A copy of the master copies is inserted. If incompatible master copies were included in a multiple selection, these are omitted and a copy is not created in the project.

See also

Basics on master copies (Page 472)

Adding master copies (Page 473)

Filtering master copies (Page 474)

Library basics (Page 439)

Using the element view (Page 445)

9.8 Using types and their versions

9.8.1 Basics on types

Using types

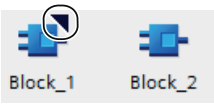
Types are elements that are required for the execution of user programs. Types can be versioned and further developed from a central location.

The following elements can be stored as type in the project library or the global library:

- Functions (FCs)
- Function blocks (FBs)
- PLC data types (UDTs)
- Faceplates
- Screens
- Styles
- User-defined functions

You can use the same version of a type only once within a CPU. If various versions of a type are available, these can also be used within a CPU. Depending on the number of CPUs used, any number of instances can be derived within a project. The instances are then connected with the version of the type in the library. If you are using types from a global library, the type is also created in the project library. If the type already exists in the project library, missing type versions are added if necessary. The instance is then linked only to the respective type version in the project library.

Types and their instances are marked with a black triangle. The following figure shows an instance, marked with a black triangle, and an ordinary program block:



Basics on versioning of types

Type versioning provides you with the means to develop types centrally and then roll out the most recent version to the individual projects as an update. In this way, error corrections and added functions can be easily integrated into existing projects. If you have created a new version of a global library, you can update existing projects in an automatic process. This minimizes errors and reduces the amount of maintenance work for large automation solutions that contain numerous individual projects.

Versioning allows you to trace the development process of individual types. Before you release a version, you can try it out in a test environment to determine whether changes made to a type integrate smoothly into an existing project. You only release a version for productive use after you have made sure that everything operates without errors. You can view the history of

individual instances in the project at any time and determine the version from which the instance was derived.

The TIA Portal also checks automatically whether there are associated objects with individual versions of a type. Associated objects can be, for example, PLC data types (UDTs) or other blocks referenced in a block. All associated objects are already taken into consideration during the creation of a type or during copying between libraries. Before being released, versions of types are checked for their consistency to ensure that no inconsistencies arise in the project.

Versions of types

Versions are assigned to each type. The version number is displayed in both the "Libraries" task card as well as the library view next to the respective type. You can also display the version of a type in the project tree next to the instances. To do this, select the "Version" column in the project tree. This allows you to see at all times which version of an instance is used in the project.

The version number consists of three numbers separated by periods. You can randomly assign the first two digits. Numbers from 1 to 999 are permitted as the first two numbers. The third digit is the build number. It is automatically incremented by one when you edit an instance related to the version. The build number is reset to 1 when you release the version.

The versions of types can have three states:

- In work (faceplates and HMI user data types)
- In test (all sorts of types except faceplates and HMI user data types)
- Released

The following figure shows a type with two versions. One version is "in test" and one version is released:



If you use functions (FCs) or function blocks (FBs) as types in the project library, observe the following when compiling the block if you use different instruction versions:

- You can recompile know-how-protected blocks without password if the interfaces of the block versions are identical.
- You can recompile blocks that are not know-how-protected if the supply is not lost for any actual parameters and if no new actual parameters that you need to supply are added by the new version.
- If it is not possible to compile the type with a different instruction version, a new version of the type is required that contains the required adjustments.

See also

State of type versions (Page 478)

Basics on master copies (Page 472)

Adding types to the project library (Page 481)

Using types of the project library (Page 485)

Editing library elements (Page 516)

Duplicating types (Page 484)

9.8.2 State of type versions

The versions of types can be in three different states. The states can be determined from the instance or in the library.

"In work" status

Only versions of faceplates and HMI user data types have the "In work" status. If a version is in progress, "In work" appears next to the version in the library.

When you create a new type or a new version of a released type, the type is assigned the status "In work".

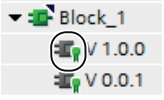
Types with the status "In work" can be edited in the library view. A reference to an instance in the project does not have to exist. Upon release, the compatibility of the type is tested by a consistency check.

"In test" status

All types except for faceplates and HMI user data types can have the "in test" status. If a version is in test, "in test" appears next to the instance and in the library. A version in the test state is linked to a test instance in the project. This allows you try out the effects of your changes in a test environment including all online functions before you release a type for use in actual operations.

"Released" status

The "released" status is available for all types, regardless of the point of use. If a version has been released, the symbol of the version is marked with a seal in the library:



Released versions can be opened with write protection in their instance. If you want to edit a released version, you must first create a new "In work" or "In test" version.

See also

Basics on types (Page 476)

Using types of the project library (Page 485)

Creating a test version of a type (Page 487)

Editing test version of a type (Page 489)

Creating an editing version of a type (Page 489)
Performing a consistency check for a version (Page 490)
Discarding versions (Page 490)
Release versions (Page 491)
Assigning a version (Page 500)
Updating the project to the latest versions (Page 494)
Remove the link between instance and type (Page 495)
Filtering the display of types (Page 454)

9.8.3 Displaying a released type version

If you display a released version but do not want to edit it, open the instance with write protection. All types except faceplates and HMI user data types can be opened directly at the instance. Faceplates and HMI user data types can only be opened in the "Libraries" task card or in the library view.

Requirement

The released version has an instance in the project, except the type is a faceplate or an HMI user data type.

Opening a type version at an instance

To open a released version of a type with write protection starting from an instance, follow these steps:

1. Select the released version at the instance in the project tree.
2. To do this, right-click the instance and select the "Open" command from the shortcut menu. The instance is opened with write protection.

Opening a type version in the "Libraries" task card or library view

To open a released version of a type in the "Libraries" task card or in the library view, follow these steps:

1. Select the version.
2. To do this, right-click the version and select the "Open" command from the shortcut menu. If it is a faceplate or HMI user data type, it is opened directly in the library view. In this case, skip the remaining steps. If it is any other type, the "Open type" dialog opens.
3. Select the instance with the version that you want to display from the list of instances.
4. Click "OK" to confirm.
   The instance is opened with write protection.

9.8.4 Displaying properties of a type or version

You can display the properties of a type or version.

Procedure

To display the properties of a type or a version and to enter a comment, follow these steps:

1. Select a type or the version of a type in the "Libraries" task card or in the library view.
2. To do this, right-click on the type or one of its versions and select the "Properties" command from the shortcut menu.
   The "Properties" dialog box opens.
3. If needed, enter a comment on the type in the "Comment" field or edit an existing comment.

Visible and changeable properties

The following table shows the properties that you can see or change for a type or version:

Property	Description	Version	Types
Name	Name of the type	-	Visible and changeable
Version	Version number	Visible	-
Last change	If you create, release, duplicate or assign a version, this is recorded as a change to the type. The date and time of the change are entered in the "Last change" field.	Visible	-
Author	The creator of the version is indicated as the author.	Visible	-
Original library	The project and the library from which the current version of the type was generated are indicated. This information is important, for example, for finding the original of the type after it has been copied from another library.	Visible	-
Version GUID or Type GUID	The GUID is used to uniquely identify the type or the version of the type, even if, for example, there are types or versions with an identical name within the project library or the global library. The GUID cannot be changed and is assigned automatically.	Visible	Visible
Comment	Comments on the type or version	Visible and, for versions whose status is "in test" or "in work", changeable	Visible and changeable

9.8.5 Working with types in the project library

9.8.5.1 Adding types to the project library

Types for reuse in the project can be created in the project library for various elements. You can create the following elements as types, for example:

- Program blocks (FCs/FBs)
- PLC data types (UDTs)
- Faceplates
- HMI user data type
- Screens
- Styles

If you add an element as a type to the project library and this element has dependencies to other elements, the dependent elements are automatically created as a type as well.

After a type has been added to the project library, the type is linked to the added element from the project. The element from the project is thus the instance of the type.

Requirement

- The "Libraries" task card is displayed.
- The elements that you want to add as type are compiled.
- The elements are consistent.
- For blocks, all other requirements are met as described in section "Block requirements (Page 482)

Procedure

To add an existing element to the project library as a type, follow these steps:

1. Open the project library in the "Libraries" task card.
2. Drag-and-drop one or more elements into the "Types" folder or any subfolder of "Types".
   Alternative: Copy the elements in the project tree to the clipboard and add the elements in the desired project library folder.
   The "Generate type" dialog opens.
3. Enter the properties of the new type:
   - Enter a name for the new type in the "Name of the type" field.
   - Enter a version number for the new type in the "Version" field.
   - Enter the name of the editor who is responsible for the type in the "User" field.
   - Enter a comment on the type in the "Comment" field.
4. Click "OK" to confirm.
   The new type is generated with a released version. The version is linked to the element (instance) that has been added.

See also

Basics on types (Page 476)

Duplicating types (Page 484)

Block requirements (Page 482)

Referencing of objects within a block (Page 483)

Library basics (Page 439)

Adding master copies (Page 473)

9.8.5.2 Block requirements

Permitted blocks for creating a type

You can create types of the following blocks in the project library:

- Function blocks (FBs)
- Functions (FCs)
- PLC data types (UDTs)

Consistency and compilation

To create a type from a block, the block must be consistent and compiled. When you create a type, the consistency of the block is checked automatically and the block is compiled if necessary.

Block requirements

To create a type from a block, the block must meet the following requirements:

- The block must match the type of CPU.
  A block which does not match a CPU can be identified by an incompatibility symbol to the right of the block in the project tree. The case occurs, for example, if you copy a block from a S7-1500 CPU to an S7-300 CPU. A block of an S7-1500 is not compatible with an S7-300 CPU and cannot be compiled. This means that you cannot create a type from the block.
- The block is not a system data block.
- The block does not include global data access or single instance call of an instance data block.
  This also applies, for example, to the call of a data block within an STL block with the "OPN" command.

Special aspects for know-how-protected blocks

You can also create types of know-how-protected blocks. However, keep in mind the following additional requirements for know-how-protected blocks:

- Release of the block for use as type
  Know-how-protected blocks must be released to be used in a library. You make the necessary settings in the block properties.
- The block does not include access to data blocks, PLC tags or PLC constants.
  Because you cannot create types from data blocks, PLC tags or PLC constants, you also cannot create a type from know-how-protected blocks with access to the objects listed above. The instance of a know-how-protected block does not work without the associated data blocks, PLC tags or PLC constants.

See also

Adding types to the project library (Page 481)

Referencing of objects within a block (Page 483)

9.8.5.3 Referencing of objects within a block

In many cases you reference other objects within a type. For example, call a tag or a data block within block. Accordingly, there is a dependence between the block and the referenced object. The block is only consistent if all dependent objects are also available.

Referencing other types

If you call other program blocks, faceplates, PLC data types (UDTs) or HMI user data types in a block these dependent objects are also stored automatically as type in the project library. When the referenced block is reused later as instance in the project, an instance of the referenced object is also created. The TIA Portal thus automatically ensures that the block state is consistent at all times and that the user program can be executed.

Referencing additional objects

Access to data blocks, PLC tags or PLC constants is permitted in blocks without know-how protection, Therefore, types can be created from the blocks. While function blocks referenced in a block, for example, are also automatically created as type in the project library, this is not the case for access to data blocks, PLC tags and PLC constants. If data blocks, PLC tags or PLC constants are referenced in a block, only the block itself is created as type. The referenced objects are not created as type. If you reference objects which are not stored as type, note the following:

- Use the block at a different point of use
  If you create an instance from the type at another point of use (such as a different CPU), the referenced objects are missing at the later point of use. This means that you should create referenced objects at the later point of use again (for example, data blocks or tag tables). Make sure that the referenced objects have the same name at the later point of use.
- Subsequent name changes to referenced objects
  If you subsequently change the name of the referenced objects, the block is no longer consistent. The referenced object (for example, a tag) is no longer found. In the case of already released versions of the type, you can no longer correct the call of the object. In this case create a new version of the type and correct the name of the called object. Then update all the instances of the type in the project to the latest version.

See also

Block requirements (Page 482)

Adding types to the project library (Page 481)

Performing a consistency check for a version (Page 490)

Updating the project to the latest versions (Page 494)

Displaying relations to other library objects (Page 457)

9.8.5.4 Duplicating types

Types in the project library can be duplicated. If you duplicate a type, the following applies to the duplicate:

- The duplicate is created in the same folder.
- The duplicate is created from the highest version of the type.
- The duplicate does not have an instance in the project.

Requirement

The type is released.

Procedure

To duplicate a type in the project library, follow these steps:

1. Right-click on a type.
2. Select the "Duplicate type" command in the shortcut menu.
   The "Duplicate type" dialog opens.
3. Enter the properties of the new type:
   - Enter a name for the new type in the "Name of the type" field.
   - Enter a version number for the new type in the "Version" field.
   - Enter the name of the editor who is responsible for the type in the "User" field.
   - Enter a comment on the type in the "Comment" field.
4. Click "OK" to confirm.
   The new type is generated with a released version.

See also

Adding types to the project library (Page 481)

Editing library elements (Page 516)

9.8.5.5 Using types of the project library

Types from the project library can be used any number of times within a project. The use of types is always linked to a version of the respective type in the project library. If the type contains dependent elements, these are also created as a type of use at a suitable point in the project. Examples of dependent elements are PLC data types referenced in a block.

You can only assign a version of the same type to a device. If necessary, you can create uses of several types at the same time.

Uses of a type are called instances in the project tree.

Possible points of use for type versions

To use a type, create an instance of a particular version of the type at a suitable point in the project. Suitable points of use are:

- Folder in the project tree
  An instance of a type can be created in a folder in the project tree. The folder must be suitable for the particular type. If the type is a user data type, for example, you can only create the instance in the "PLC data types" folder.
- Editor
  An instance can be created from a type in a suitable editor. For example, you can create an instance from the type of a function block in another block. The type of the function block is called from another block in this way.

Options for creating an instance

You have two options for creating type instances:

- Drag the required type version from the project library and drop it at the location where you want to use it.
  Instances of the types and their dependent elements are generated and pasted at the location where you want to use them. The instances are connected to the respective type version in the project library. If you have created the instances in an editor, instances of the types are also created at the appropriate points in the project tree. By default, the folder structure from the library is reproduced in the project tree. If you want to select a different folder in the project tree, you will find the instances in the same folders as in the library.
- Copying and pasting type instances
  You can copy type instances to the clipboard and paste them at another point. This way you create another instance of the type version. The instance is still connected to the same type version in the project library. When you copy the instance of a type to the clipboard and paste it in another project, all required type versions are accepted in the project library of the other project.

Requirement

- The desired versions have been released.
- A device which supports the category of desired type is already available in the project
- The device is not yet assigned any further instance of the same type.

Procedure

To create an instance of a type, follow these steps:

1. In the project library, select the versions from which you want to create an instance.
2. Using a drag-and-drop operation, move the versions from the project library to the point of use in the project tree or an editor.
   Alternative: To automatically use the newest version, move the types themselves from the project library to the point of use using a drag-and-drop operation.
   For example, use a drag-and-drop operation to move the type of a function block to the block folder of the CPU in the project tree. To call the type directly from another block, for example, move the type from the project library to the point of use in the program editor using a drag-and-drop operation.

Alternative:

1. Copy one or more instances to the clipboard.
2. Paste the instances at a suitable point in the same project or in a different project.

See also

Basics on types (Page 476)

State of type versions (Page 478)

Displaying types for an instance (Page 487)

Library basics (Page 439)

Using master copies (Page 475)

9.8.5.6 Displaying types for an instance

You can jump from the project tree to the type in the library that belongs to instance in the project tree.

Procedure

To jump to the type in the project library, follow these steps:

1. Right-click the instance of the type in the project tree.
2. Select the "Go to type" command from the shortcut menu.
   The referenced type version is displayed in the project library.

See also

Using types of the project library (Page 485)

9.8.5.7 Creating a test version of a type

Before you release a type for productive use, you need to test the type within a project and on the automation system. The test is conducted in a specific test environment. This test environment can be a CPU, for example.

Create a version with "in test" status for the test. The creation of a version "in test" is suitable for all sorts of types except for faceplates and HMI user data types. On the other hand, versions "In work" can be created from faceplates and HMI user data types.

There are two ways to create a test version of a type and define the test environment:

- In the "Libraries" task card or library view
  Generate the new version with "in test" status in the "Libraries" task card or in the library view. You can generate the new version either directly from the type or from a specific version of the type.
- At an instance in the project tree
  You can also create the test version directly at the instance in the project tree. Since the instance is always used in a specific version in the project, a new version of the type is generated from the version used at the instance.

You can also create test versions from several types at the same time.

The following rules apply to a version "in test":

- You can set only one version to "in test" for each type at a given time.
- A version in testing may only be linked to a single instance in the project. Therefore, it is not possible to copy an instance to the clipboard, to duplicate it or to create an additional type from the instance as long as it has "in test" status.

Requirement

- There is at least one instance of the type within the project in a given version.
- If you want to create the new version from a particular version of the type, the instance must be used in this version in the project.

Procedure

To generate a new test version of a type or the version of a type, follow these steps:

1. Select the type, a version of the type, or the instance.
   When you create the test version directly at the instance, you can select several elements or folders with multiple selection. You can skip steps 3 and 4 because the test environment is already defined by the selected instance.
2. Right-click the selected element and select the "Edit type" command from the shortcut menu.
   If you have started the editing in the "Libraries" task card or in the library view, the "Edit type" dialog opens. If you have started the editing at the instance in the project tree, the test instance is immediately opened for editing in the library view.
3. Select an instance of the type from the list in the project.
   If you have started editing at the type itself, the following applies:
   – The location at which the instance is used (for example, the CPU) is also used as test environment for the subsequent editing of the type.
   – Selecting the test instance also defines the version to be edited.

   The following applies to editing a specific version:
   If your starting point is a specific version, you can only select from the list instances that are used in the same version.
4. Click "OK" to confirm.

Result

A new version of the type is created. The new version is "in test" and is identified as such in the user interface.

See also

Basics on types (Page 476)

State of type versions (Page 478)

Library basics (Page 439)

9.8.5.8 Editing test version of a type

When you edit a version "in test", a new version is not created. You can start the editing of the test version at the instance in the project tree, in the "Libraries" task card or in the library view.

Note

Deleting and renaming interface parameters

You can add new parameters. However, if you rename or delete existing parameters, these parameters will not be supplied when the block is called.

Procedure

To edit the test version of a type, follow these steps:

1. Right-click the test version or the instance.
2. Select the "Edit type" command from the shortcut menu.
   The test instance is opened and can be edited.

See also

Basics on types (Page 476)

State of type versions (Page 478)

Library basics (Page 439)

Performing a consistency check for a version (Page 490)

Discarding versions (Page 490)

Release versions (Page 491)

9.8.5.9 Creating an editing version of a type

If you want to edit a type with faceplates or HMI user data types, create a new "In work" version of the type. The new version is edited in the library view. To check the compatibility of the changes, a consistency check is automatically performed for the type prior to the release.

Requirement

The project library is opened in the "Libraries" task card or in the library view.

Procedure

To create a new version of a type in work, follow these steps:

1. Right-click the type or the version of the type.
2. Select the "Edit type" command from the shortcut menu.
   A new "In work" version is created and opened for editing in the library view.

See also

Basics on types (Page 476)

State of type versions (Page 478)

Library basics (Page 439)

9.8.5.10 Performing a consistency check for a version

A type version can unintentionally obtain an inconsistent state during editing. To notice errors in the development process in good time, you can regularly perform a consistency check. However, the consistency check always takes place automatically as soon as you release a version.

Details of how to start the consistency check manually for the version of a type are provided below.

Requirement

- The project library is opened in the "Libraries" task card or in the library view.
- The version is "In work" or "In test".

Procedure

To perform a consistency check for the version of a type, follow these steps:

1. Right-click on the version that you want to check for consistency.
2. Select the "Consistency check" command in the shortcut menu.
   The consistency check is carried out. You receive a message with the result of the consistency check.

See also

Release versions (Page 491)

Discarding versions (Page 490)

Editing test version of a type (Page 489)

Basics on types (Page 476)

State of type versions (Page 478)

Library basics (Page 439)

9.8.5.11 Discarding versions

Discard versions of a type "In test" or "In work" when you no longer need the version. You can also select several types or folders and discard all test or working versions contained therein. All uses of the deleted versions are reset to the last released status.

Requirement

- The version that you want to discard is in the state "In test" or "In work".
- Your are in the library view of the "Libraries" task card is open.

Procedure

To discard one version, follow these steps:

1. Right-click the version that you want to discard.
2. Select the "Discard changes and delete version" command from the shortcut menu.
   The version is deleted.

Alternative in the library view:

1. Click the "Discard changes and delete version" button in the toolbar while a version is opened for editing.
   The version is deleted.

See also

Basics on types (Page 476)

State of type versions (Page 478)

Library basics (Page 439)

Performing a consistency check for a version (Page 490)

9.8.5.12 Discarding all versions within a folder

You can discard all versions with the "In test" or "In work" status at the same time within a folder. All uses of the deleted version are reset to the last released status.

Requirement

Your are in the library view of the "Libraries" task card is open.

Discarding a version of a single type

To discard all versions within a folder, follow these steps:

1. Right-click the folder.
2. Select the "Discard all" command from the shortcut menu.
   All "In test" or "In work" versions are deleted.

9.8.5.13 Release versions

When you are finished editing a type version, release the version for productive use. Assign a version number for the release. You can also use a multiple selection to release several versions at the same time.

Requirement

- The "Libraries" task card is open or you are in the library view.
- The versions that you want to release are "In test" or "In work" status.
- The versions are consistent.
  A consistency check is performed as soon as you start the release. If errors that prevent a release occur during the consistency check, a message is displayed indicating how you can correct the errors.

Procedure

To release type versions, follow these steps:

1. Select the versions you want to release.
2. Right-click your selection.
3. Select the "Release version" command from the shortcut menu.
   The "Release type version" dialog box opens.
4. If necessary, change the properties of the version:
   - Enter a name for the type in the "Name" field. If you have selected several versions for release, the "Name" field cannot be changed.
   - In the "Version" field, define a main and an intermediate version number for the version to be released. If you have selected several versions for release, the "Version" field cannot be changed and the last version number is used for the release.
   - In the "Author" field, enter the editor of the version to be released.
   - In the "Comment" field, enter a comment on the version to be released.
5. Optional: Select the "Delete unused type versions from the library" check box to delete all versions from the library that are not connected to any instance in the project. Versions with dependencies on other types or master copies are not deleted.
6. Click "OK" to confirm.

Alternative in the library view:

1. Click the "Release version" icon in the toolbar while a version is opened for editing.
2. Continue with steps 3 to 5 of the description above.

Result

The selected versions are released.

The properties are applied for the types themselves, the versions to be released, and for all future versions. Versions already released remain unaffected by the changes.

If needed, all instances with the same original version are updated to the most recent version and the unused versions of the type are deleted.

See also

Releasing all versions within a folder (Page 493)

Basics on types (Page 476)

State of type versions (Page 478)

Library basics (Page 439)

Performing a consistency check for a version (Page 490)

Assigning a version (Page 500)

Adding types to a global library (Page 496)

Updating the project to the latest versions (Page 494)

9.8.5.14 Releasing all versions within a folder

When you are finished editing all types within a folder, release all versions simultaneously.

Requirement

- Your are in the "Libraries" task card or in the library view.
- The folder includes versions "In test" or "In work" status.
- All versions "In test" or "In work" are consistent.
  A consistency check is performed as soon as you start the release. If errors that prevent release occur during the consistency check, a message is displayed indicating how to correct the errors.

Procedure

To release all type versions within a folder, follow these steps:

1. Right-click the required folder.
2. Select the "Release all" command from the shortcut menu.
   The "Release type version" dialog box opens.
3. If necessary, change the properties of the version:
   - In the "Author" field, enter the editor of the versions to be released.
   - In the "Comment" field, enter a comment on the versions to be released.
4. Select the "Delete unused type versions from the library" check box to delete all versions from the library that are not connected to any instance in the project. Versions with dependencies on other types or master copies are not deleted.
5. Click "OK" to confirm.

Result

All type versions "In test" or "In work" status within the selected folder are released.

The properties are applied for the versions to be released and for all future versions. Versions already released remain unaffected by the changes.

Versions of types not used in the project may be deleted.

See also

Release versions (Page 491)

9.8.5.15 Updating the project to the latest versions

After you have updated several types in the project library, update all instances in the project to the most recent version of the types from the project library. If you do not want to apply the changes to the entire project, restrict the update to individual devices in the project.

Each of the following elements can be selected as source for the update:

- The entire project library
- Individual folders within the project library
- Individual types
  You can select multiple types.

Requirement

Your are in the "Libraries" task card or in the library view.

Procedure

To update instances in a project with the contents from the project library, follow these steps:

1. Select the entire project library or elements from it.
2. Right-click the required elements and select the "Update > Project" command from the shortcut menu.
   The "Update project" dialog box opens.
3. Select either the entire project or individual devices for the update.
4. Select the options for the update process:
   – The "Update instances in the project" check box is always selected during this process.
   – Select the "Delete all unused type versions from the library" check box to delete all older versions of the updated types from the project library.
5. Click "OK" to confirm.
   The update is performed.

Result

The following changes were made to the project:

- All older versions are deleted from the project library as needed.
- All instances within the selected devices are updated to the most recent version of the linked type.
- If a CPU does not include block instances that are called in the user program, the block instances are removed from the CPU. The types still exist in the library and can be used in the project again as needed.
- You can find a log of the update process in the projection tree under "Common data".

See also

Updating the project to the latest type versions (Page 498)

Updating library with the types of another library (Page 505)

Basics on types (Page 476)

Library basics (Page 439)

State of type versions (Page 478)

9.8.5.16 Remove the link between instance and type

Instances of types are always connected to the version of the corresponding types. They cannot be edited like an ordinary object. If you edit the instance, a new version of the type is created automatically in the project library and the changes therefore affect the entire project.

If you remove the link between the instance and its type, you then edit the object like an ordinary object in the project tree.

Note

Disable "Terminate link to the type"

If you want to prevent instances of your types getting separated in the project, you can also deactivate the "Terminate link to the type" option.

Activate the option "Shortcut menu entry 'Terminate link to the type'" under "Options > Settings > General > Library settings".

Requirement

The instance must not be in the "in test" state.

Procedure

To remove the link between instances and their type versions, follow these steps:

1. Select one or more instances in the project tree.
2. Right-click the selection and select the "Terminate link to type" command from the shortcut menu.
3. The link to the corresponding type versions is removed.

See also

Basics on types (Page 476)
Library basics (Page 439)
State of type versions (Page 478)

9.8.6 Working with types in global libraries

9.8.6.1 Adding types to a global library

Global libraries are used as a central resource when working on multiple projects. Among the types, only the types in the project library can be edited directly. Therefore, use the project library if you want to work on types. When you are finished editing a type in the project library, you can add the type to a global library. Adding types from the project library corresponds to a normal copy process from the project library.

Requirement

- The "Libraries" task card is displayed or the library view is open.
- The global library to which you want to add types is opened and can be written to.

Procedure

To add types to a global library, follow these steps:

1. Open the required folder in the global library in the "Libraries" task card or in the library view.
2. Drag one or more types from the project library to the "Types" folder or any subfolder of the global library.

Alternative:

1. Copy the required types from the project library to the clipboard.
2. Open the required global library in the "Global library" pane of the "Libraries" task card.
3. Right-click the "Types" folder or any subfolder of "Types".
4. Select "Paste" in the shortcut menu.

Result

The types are inserted in the global library. Dependent types, such as types of HMI user data types or tags, are also copied to the global library, provided they do not already exist there. This ensures that all necessary elements for generating an instance are present in the global library.

If a type already exists in the global library, the described action corresponds to an update of the global library. In this case, the most recent released versions of the types are added to the global library.

See also

Basics on types (Page 476)

Release versions (Page 491)

Assigning a version (Page 500)

Updating the project to the latest type versions (Page 498)

Library basics (Page 439)

9.8.6.2 Using types from a global library

To use types from the global library, create an instance of a particular version of the type at a suitable point in the project. If necessary, you can create uses of several types at the same time. Uses of a type are called instances in the project tree.

Possible points of use for type versions

Suitable points of use for types from the global libraries are:

- Folder in the project tree
  An instance of a type can be created in a folder in the project tree. The folder must be suitable for the particular type. If the type is a user data type, for example, you can only create the instance in the "PLC data types" folder.
- Editor
  An instance can be created from a type in a suitable editor. For example, you can create an instance from the type of a function block in another block. The type of the function block is called from another block in this way.

Linking the instance with the project library

In the project, instances of types from a global library are not linked to the type in the global library. Instead, a copy of the type and its dependent elements is generated in the project library when an instance is created. Dependent elements, for example, can be PLC data types that are referenced in a block. In each case, the copy of the type and the dependent elements in the project library contains the version that you have linked to the instances. If the type or a dependent element already exists in the project library, only the missing version is added in the project library, if necessary.

The instance is finally linked to the copy of the type in the project library. You can only assign a type to a device once irrespective of the version.

Requirement

- A device which supports the category of the type is already available in the project.
- The device is not assigned to any further instances of the same type.

Procedure

To use the version of a type in the project, follow these steps:

1. In the global library, select the versions from which you want to create an instance.
2. Using a drag-and-drop operation, move the desired versions of the types to the point of use.
   Alternative: To automatically use the newest version, move the types themselves from the library to the point of use using a drag-and-drop operation.
   For example, use a drag-and-drop operation to move the type of a function block to the block folder of the CPU in the project tree. To call the type directly from another block, for example, move the type from the library to the point of use in the program editor using a drag-and-drop operation.

Result

Missing types or individual versions are added in the project library. If a type is not yet present in the project library, it is stored in the same folder as before in the global library. An instance is created from the types and their dependent elements and inserted at the point of use. The instances are connected to the respective type version in the project library.

If you have created the instances in an editor, instances of the types are also created at the appropriate points in the project tree. The folder structure from the library is reproduced in the project tree. You will therefore find the instances in the same folders as in the global library.

See also

Basics on types (Page 476)

Updating the project to the latest type versions (Page 498)

Library basics (Page 439)

Using the element view (Page 445)

9.8.6.3 Updating the project to the latest type versions

In large enterprises with numerous automation projects. the global libraries are frequently edited from a central location. The updated global libraries of the individual projects are made available after the completion of a new version. If you have received a more recent version of a global library, replace the outdated instances in your project with the most recent version. If you do not want to apply the changes to the entire project, restrict the update to individual devices in the project.

The project library is also updated with the new versions of the types in the global library during the updating of the project or individual devices.

The following elements can be selected as source for the updating:

- Individual folders within a global library
- Individual types
  You can select multiple types.

Requirement

- Your are in the "Libraries" task card or in the library view.
- The updated global library is open.

Procedure

To update instances in a project with the contents from a global library, follow these steps:

1. Select the updated global library or the individual elements from it.
2. Right-click the global library or the required elements and select the "Update > Project" command from the shortcut menu.
   The "Update project" dialog box opens.
3. Select either the entire project or individual devices for the update.
4. Select the options for the update process:
   - The "Update instances in the project" check box is always selected during this process.
   - Select the "Delete all unused type versions from the library" check box to delete all older versions of the updated types from the project library.
5. Click "OK" to confirm.
   The update is performed.

Result

The following changes were made to the project:

- The most recent version of the select types is present in the project library. All older versions were deleted if necessary.
- All instances within the selected devices were updated to the most recent version of the linked types.
- You can find a log of the update process in the projection tree under "Common data".

See also

Updating the project to the latest versions (Page 494)

Updating library with the types of another library (Page 505)

Basics on types (Page 476)

Library basics (Page 439)

Adding types to a global library (Page 496)

9.8.7 Assigning a version

A library is more clearly structured if types related by content have the same version number. The identical version number reflects the work progress. When you have completed the work on multiple associated types, you can assign the same version number to these types.

You have the following options to assign a common version to types:

- One or more folders in a library
- One or more types

Requirement

- The "Libraries" task card or the library view is open.
- Your selection must not contain types with "In test" or "In work" status.

Procedure

To assign several types the same version, follow these steps:

1. Select the types to which you want to assign a common version.
2. Select the "Assign version" command from the shortcut menu.
   The "Assign version" dialog box opens.
3. If necessary, change the properties of the version:
   - In the "Version" field, determine the new version number. The version number must be higher than the highest version number of all selected types.
   - In the "Author" field, enter the person responsible for the version to be released.
   - In the "Comment" field, enter a comment on the version to be released.
4. Click "OK" to confirm.

Result

The selected type versions are changed as follows:

- A new version of all selected types is created with the specified version number.
- The properties are applied to all selected types, the new version and to all future versions. Lower versions remain unaffected by the changes. If you make no changes to the properties, the properties of the last released version of each type are applied.
- The build number of dependent types is incremented to the next free build number as long as the dependent types were not in your selection. If you had selected a dependent type as well, the version number you specified will be assigned.

A log of the changes is created. If you have versioned the types in the project library, you find the log in the project tree under "Common data > Logs". If you have versioned types in a global library, you find the log in the "Common data > Logs" folder in the level below the global library.

See also

Basics on types (Page 476)
State of type versions (Page 478)
Library basics (Page 439)
Release versions (Page 491)
Adding types to a global library (Page 496)
Displaying logs of global libraries (Page 463)

9.9 Multiuser Engineering in libraries

9.9.1 Multiuser Engineering with library elements

Library elements in the project library can be used in Multiuser Engineering.

More information on working with Multiuser Engineering can be found in section "Introduction to Multiuser Engineering (Page 6687)".

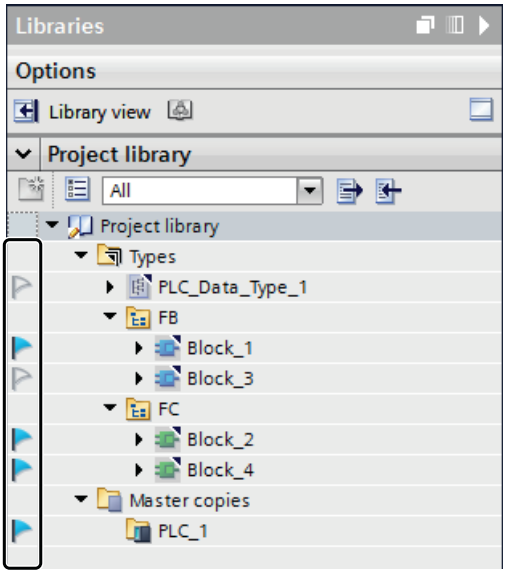
Markings for the check-in

Individual library elements within the project library are marked for Multiuser Engineering in the same way as in the project tree. You can mark library elements in the following areas of the TIA Portal:

- "Libraries" task card
- Library view
- Library management
- At the respective point of use, e.g. at type instances

Always mark the library elements manually.

The figure below shows the "Libraries" task card with the marking column and marked library elements for the multiuser function:



Library elements that can be marked

The following table shows which types can be marked for check-in in the project library and in the instance in the project:

	Type can be marked in the library	Instance can be marked
Function blocks (FB)	Yes	Yes
PLC Block Library Type (FC)	Yes	Yes
PLC data types (UDT)	Yes	Yes
HMI faceplates	Yes	By marking the screen in which the instance of the block is located
HMI user-defined data type (UDT)	Yes	By marking the object (e.g. screen or HMI tag), in which the UDT is used.
HMI style	Yes	No
HMI style sheet	Yes	Not available
HMI screen	Yes	Yes
HMI C script	Yes	Yes
HMI VB script	Yes	Yes

Deleting types in the local session

If you want to delete types in the local session, note the following:

- Release the type if a version is still being tested or edited.
- To delete the type in the assigned server project, mark the type in the local session before deletion.

Individual versions can be deleted in the local session, but these are retained in the server project. Because deleting is an administrative activity, you delete the types and their versions in the server project.

Updating the project

If you want to update the project, note the following:

- Mark the types before you update the project.
- Mark the HMI user data types at the point of use before you update the project.
- Changes to HMI styles are not applied in the local session when you update the project. Update the HMI style in the server project view.

See also

Introduction to Multiuser Engineering (Page 6687)

9.9.2 Checking in types and master copies

Check changes to types and master copies into the multiuser server project in the same way as all other objects within the local session. When you check in a type, all versions of the type are always checked in.

Update the server project

Types and master copies are identified and synchronized based on the unique GUID in the server project. When types and master copies are checked in, their changes from the local session are applied in the server project:

- The properties of the type or the master copy, e.g. the name, are applied in the server project
- Missing versions of types are added.
- The folder structure within the library is applied in the server project. However, empty folders are not deleted.

Conflicts can occur through updating the server project. Any conflicts are resolved as follows:

- Naming conflicts
  Two types with identical names cannot be checked in. The names of the respective types automatically receive an extension (<_1>, etc.).
  Two types instances with identical names are not permitted and result in an error when checking in.
- A type is "In test" or "In work" status.
  If a type is in one of these two states in the server project or in the local session, you cannot update the server project. First release all marked types before you update the server project.
- Version conflicts
  A message will notify you if both the server project and local session have versions with the identical version number but different content. Change the version numbers so that these exist only once.

More information on Multiuser Engineering can be found in section "Introduction to Multiuser Engineering (Page 6687)".

See also

Introduction to Multiuser Engineering (Page 6687)

9.10 Updating and upgrading types of a library

9.10.1 Updating library with the types of another library

An existing library can be updated with the contents of another library. The following options are available for updating libraries:

- Updating a global library with types from another global library or from the project library
- Updating the project library with types from a global library

Each of the following elements can be selected as source for the update:

- The "Types" folder
- Individual folders within the "Types" folder
- Individual types
  You can select multiple types.

During the update, new versions are added to the types that already exist in the target library. Types that do not yet exist in the target library are copied together with all their versions to the target library.

Note

User-defined documentation when copying types

User-defined documentation is not copied together with a type to another library. You need to copy the user-defined documentation for types to the corresponding directory.

For additional help on using the user-defined documentation, refer to the section "Using user-defined documentation (Page 426)".

Requirement

If you want to update a global library, you have to open it with write permission.

Procedure

To update a library with the contents of a different library, follow these steps:

1. Select the "Types" folder or individual elements from this folder as source for the update.
2. Right-click the source and select the "Update > Library" command from the shortcut menu. The "Update library" dialog box opens.
3. Select the type of library you want to update:
   - Select "Update the project library" to update the project library with types from a global library.
   - Select "Update a global library" if you want to update a global library.
4. Optional: In the drop-down list, select the global library that you want to update, if you want to update a global library.

5. Select the options for the update:
   – The option "Update instances in the project" is always deactivated in this process.
   – Select the "Delete all unused type versions from the library" check box if you want to delete all versions of types from the library if these are not assigned to any instance in the project and if there are no dependencies to other types. This option cannot be selected for the update of a global library, since types of a global library never have a point of use in the project.
6. Click "OK" to confirm.
   The update is performed.

Result

The following changes were made to the target library:

- Types not yet available in the target library were copied together with all their type versions. More recent type versions were added to the types that already existed in the target library. If a more recent version of a type already existed in the target library, the latest version was nevertheless copied and automatically assigned a newer version number.
- If needed, all versions of types were deleted from the project library if these were not used in any instance in the project.
- A log listing all performed changes to the target library was created for the update process. If you have updated the project library, you can find the log in the project tree under "Common data > Logs".
  If you have updated a global library, you can find the log in the "Common data > Logs" folder in the level below the global library.

See also

Updating the project to the latest versions (Page 494)

Updating the project to the latest type versions (Page 498)

Displaying logs of global libraries (Page 463)

Library basics (Page 439)

Using user-defined documentation (Page 426)

Conventions for the creation (Page 432)

9.10.2 Upgrading types in an existing project

Using the function "Upgrading types", you can upgrade the type dependencies to the latest available type versions in one step throughout the entire project.

If you edit a type that references another type within your project library and create a new type version, the referenced type is automatically set to the state "in test" or "in work". This ensures that all types within the project are up to date.

If you edit a type outside of the project and create a new type version, the referenced types are not automatically updated. To ensure that these dependencies to the individual referenced

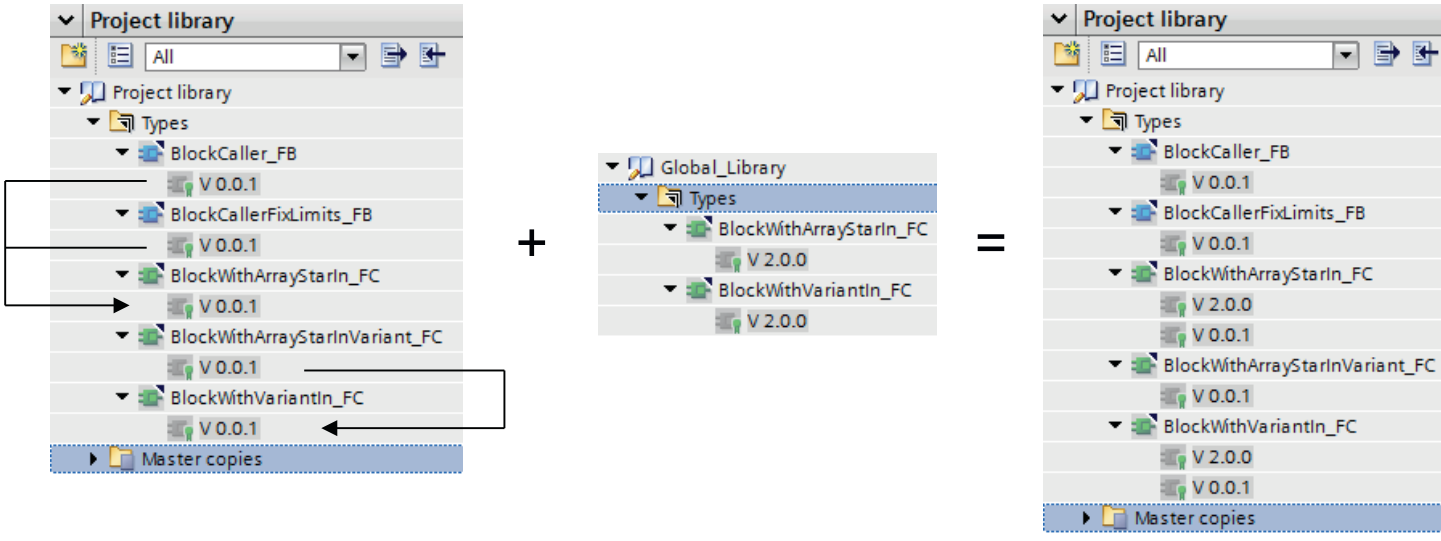
types do not have to be manually upgraded, the "Upgrading of types" function is available in the library management.

Using this function you can identify all referenced types that are dependent on the changed type and automatically upgrade these types. The "in test" or "in work" extension is added to all upgraded types in the project library. After the upgrading you can see at a glance which types are also affected by the changes and can check these individually and release them once again.

Requirement

You have created a project that, for example, contains program blocks that call other program blocks or objects, such as PLC data types (UDTs). Dependencies thus exist between the individual objects.

You have created a project library and stored the types to be edited in a global library. The types were further developed outside of your project and are now to be re-integrated with the new type versions in your existing project library. Update the project library for this purpose.



To integrate the new type versions into the existing project library, follow these steps:

1. Open the global library without write protection.
   The types to edited have a higher type version than the types that are contained in the project library.
2. In the global library, click the folder "Types" and select "Update > Project" from the shortcut menu.
   The project library is updated and the new type versions are included.

Additional information on updating libraries is available here: Updating library with the types of another library (Page 505).

At this time the new type versions are only available in the project library, but not yet referenced in the project.

Procedure

To automatically reference new type versions in the project, these must be upgraded in the library management.

Proceed as follows:

1. In the project library, click the folder "Types" and open the library management.
2. In the "Types" area, select the entry "Types with new versions for upgrading other types" from the drop-down list.
   All types are displayed for which a new type version is available in the project library.
3. Right-click the "Types" folder and select the entry "Display upgradable types" from the shortcut menu.
   All referenced types are displayed in the "Upgrading of types" tab.
4. Select one of two entries in "Upgrading of types":
   - Only with test environment
     All types that reference an instance in the test environment are displayed.
   - Compatible with selected test environment
     All types that reference an instance in the test environment are displayed. However, the instance was not yet created at this time.
5. Select a test environment
   With the selection of a test environment, in which test environment the types are to be upgraded.
6. All types that have to be upgraded are now displayed.

   Note

   Display of types

   The display of types depends on which folder you selected in the library management and which settings you made in the "Upgrading types" tab.

7. Click "Start".
   The "Upgrading of types" dialog opens after the successful upgrading.

   Note

   "Error" status

   You cannot start the upgrading of types if one the types to be upgraded has the "Error" status. You have two options:
   1. Correct the error and restart the upgrading.
   2. Refresh the view of the possible types to be upgraded, by selecting the required entry from the drop-down list under "Types to be upgraded".

   All referenced type are now in the project tree and in the "in test" state in the project library.
8. Check the types "in test" and then release them. You can also use the management library for this purpose.
9. To do so, select the "Types with pending changes" from the drop-down list and release all displayed types.

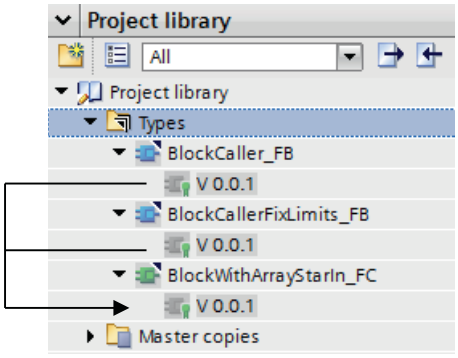
Result

All types within your project now reference the current type version.

9.10.3 Example of the updating and upgrading of types

Description

You have a project library whose types reference one other in different ways.



The function "BlockWithArrayStarIn_FC" is called by each of the two function blocks.

The function "BlockWithArrayStarIn_FC" should be further developed independently outside of the project library.

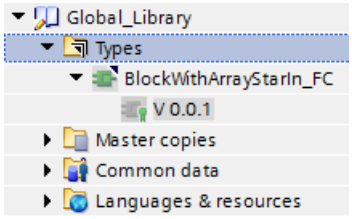
Requirement

You have created a project library with all relevant types.

Procedure

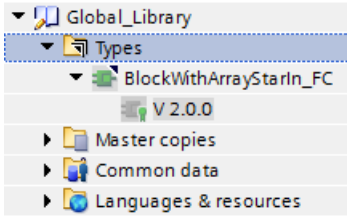
Proceed as follows to further develop the type "BlockWithArrayStarIn_FC" independently outside of the project:

1. Copy the type from the project library to a global library in order to be able to transfer it.



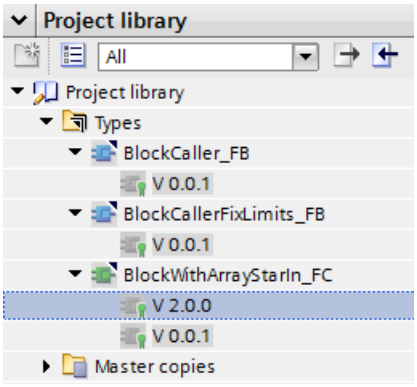
   You transfer version V0.0.1 for further development.

2. The type "BlockWithArrayStarIn_FC" was edited and released. The provider has created a new version V2.0.0 and returned the global library with the new type version:



To integrate the new versions of the function into your existing project library, follow these steps:

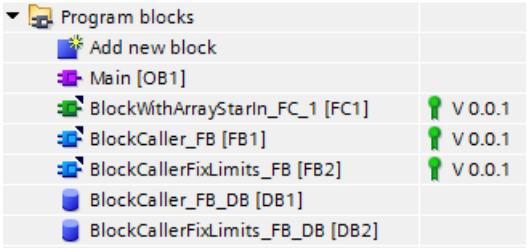
1. Click the "Types" folder of the global library "Global_Library".
2. Select the entry "Update > Project" from the shortcut menu.
   The "Update project" dialog box opens.
3. Click "OK".
   The new type version V2.0.0 has been integrated in the project library.



Note

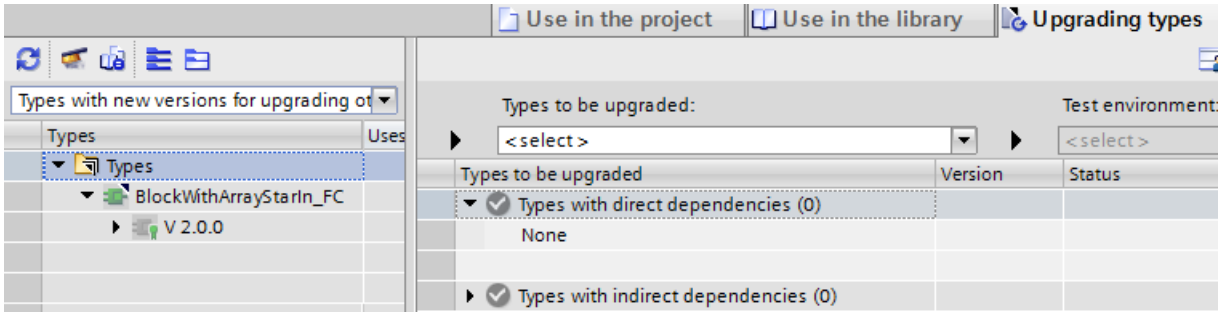
Referencing the latest type version

At this time the new type version is only available in the project library. The instances in the project tree still reference the obsolete type version V0.0.1.

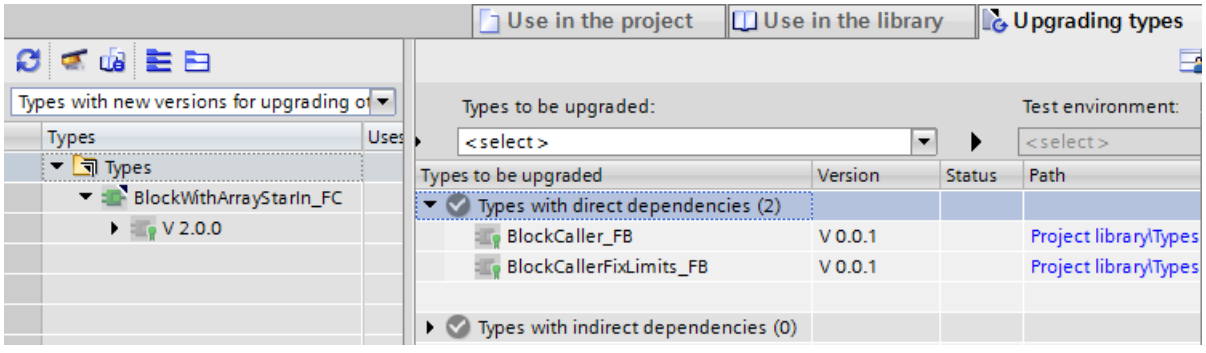


To upgrade the instances in the project tree to the new type version in one step, follow these steps:

1. Click the "Types" folder in the project library and open the library management.
2. In the "Types" area, select the entry "Types with new versions for upgrading other types" from the drop-down list.
   The function that a new type version has that is not yet used in the project tree is displayed.

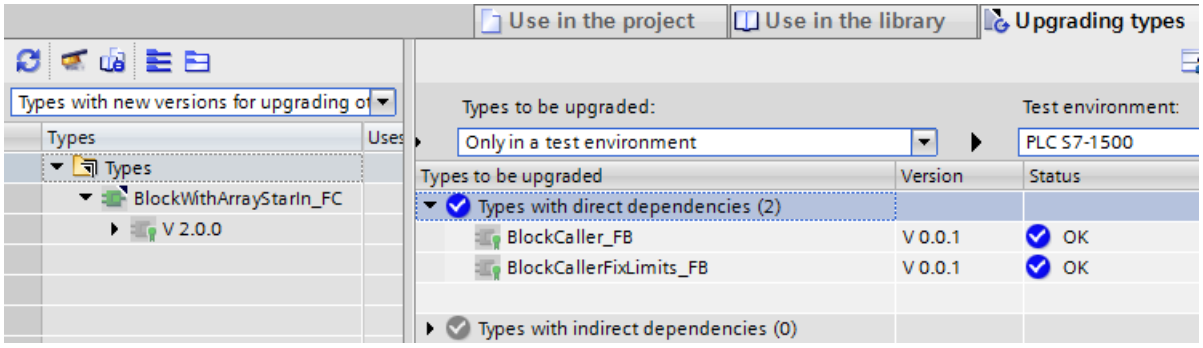


3. Click the "Types" folder and select the entry "Display upgradable type" from the shortcut menu.
   The "Upgrading types" tab opens and all types are displayed which have a direct dependency to the changed type.



4. To filter the upgraded types for a test environment, select the entry "Only with test environment" under "Types to be upgraded".

5. Select the test environment "PLC S7-1500".

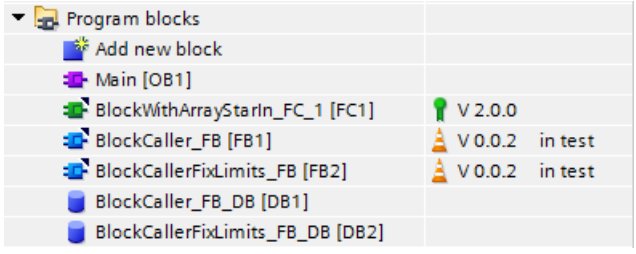


All types that can be upgraded have the status "OK".
If one or more types have the status "Error", you cannot start the upgrading. First correct the error and then repeat the steps 1 - 5.

6. Click "Start".

Result

All referenced types are upgraded to a new version in the project library and in the project tree and the extension "in test" is added.



Check the new versions and then release these.

9.11 Defining the project languages in the project library

All defined project languages are also available in the project library.

You can define and change the project languages centrally for the entire project library. In this case, all changes to the project languages affect all existing and newly created types and master copies in the project library.

However, you can also change the project languages for individual type versions without affecting the rest of the project library.

Procedure

To define the project languages for the entire project library, follow these steps:

1. Open or edit a type or master copy.
2. Open the "Tasks" card.
3. Under "Languages & resources" either select another project language from the existing editing languages or click "Add new editing language".
4. If you click "Add new editing language", the list of all available project languages will be displayed.
5. Select the editing language you want to use.

Procedure

To define the project languages for an individual type version, follow these steps:

1. Select the appropriate type version from the project library.
2. Select the command "Tools > Project languages" from the menu bar.
   Shows the list of all available project languages.
3. Select the editing language you want to use.

Result

The project library or the type version has the requested project or editing languages.

9.12 Creating folders in a library

The library elements are stored in the libraries according to their type in the "Types" and "Master copies" folders. Create additional folders below "Types" and "Master copies" to further organize master copies and types.

Requirements

- The "Libraries" task card is displayed or the library view is open.
- If you want to create new folders within a global library, you have to open the global library with write permission.

Procedure

To create a new folder, follow these steps:

1. Right-click any folder within the library.
2. Select "Add folder" from the shortcut menu.
   A new folder is created.
3. Enter a name for the new folder.

See also

Filtering master copies (Page 474)

Working with types in the project library (Page 481)

9.13 Editing library elements

Types, master copies and folder can be cut, copied, pasted, moved, renamed or deleted in the usual way within the "Libraries" task card or the library view. Global libraries must be opened with write permission for each of the above-described processes.

Note

User-defined documentation for types and master copies

User-defined documentation is not affected by any of the operations within the library. If you move a master copy or a type to a different location, you also move the associated user-defined documentation in the file system to the corresponding location.

For additional help on using the user-defined documentation, refer to the chapter "Using user-defined documentation (Page 426)".

Copying types

The following rules apply when you copy types to the clipboard:

- Types are always copied to the clipboard with all associated versions. However, only versions that have previously been released are copied.
- Types are always copied to the clipboard with all dependent elements.
- Master copies are always copied to the clipboard with all type versions used in them.

Copying and pasting type versions

When you copy type versions to another library, the types must already exist in the target library.

Cutting elements

You can only paste previously cut library elements into the same library. In so doing, you can only paste master copies into the "Master copies" folder or any of its subfolders. Likewise, you can only paste types into the "Types" folder or any of its subfolders.

Pasting types

Pasting types in a different library corresponds to updating the target library.

The following rules apply when you have pasted a type into a different library:

- A type is always pasted with all its versions.
- If the type already exists in the target library, all versions that are not yet available are added to the corresponding types in the target library.
- If there is already a version with released status in the target library, this version is not pasted again.

- If the same version exists with "In test" or "In work" status in the target library, it is replaced by the released version.
- If a type needs other types, these are also added at the respective location.

Pasting master copies

When you paste master copies, all type versions contained in these copies are also pasted. If the corresponding types already exist in the library, only the missing versions are added to the individual types. If one of the types used does not yet exist, it is pasted at the highest level in the library. The type includes the type version that was used in the master copy.

Moving elements

When you move an element from one library to another, the element is copied and not moved. The same rules apply as described under "Pasting types" and "Pasting master copies".

Deleting of types and type versions

Note the following when you delete types or type versions:

- A type or a type version can only be deleted if there are no dependencies to other types.
- When you delete a type, all versions of the type are deleted.
- If you delete all versions of a type, the type is also deleted.
- If you delete a version that has instances in the project, the instances are also deleted from the project.
- If you delete a type that is also stored at the same time as a master copy, the master copy is also deleted.

Deleting instances

If you delete an instance that has dependencies to other instances, this instance is restored during the next compilation. The instance is then linked again to the original type version. This restores the consistency of the project.

See also

Library basics (Page 439)

Remove the link between instance and type (Page 495)

Updating library with the types of another library (Page 505)

Conventions for the creation (Page 432)

Using user-defined documentation (Page 426)

Duplicating types (Page 484)

9.14 Harmonizing names and path structure

You can harmonize the project with a library. This helps you correct the following items:

- Names of the instances:
  Instances can be created during the development phase of a library, the names of which are appended by "_1", "_2", etc. due to an automatic correction. This extension is used to prevent duplicate names in the project. During harmonization, the instances once again receive the names of their associated types.
- Path structure:
  The original path structure can be lost due to parallel development or copying of dependent instances. This affects the clarity of the project. During harmonization, the path structure within the project is adapted to the path structure of the library.

Procedure

To harmonize the names and the path structure, follow these steps:

1. Open the library management.
2. Click "Harmonize project" in the toolbar.
   The "Harmonize project" dialog box opens.
3. Select the device with which you wish to harmonize the library.
4. Select the "Harmonize paths between project and library" check box if you want to restore the path structure.
5. Select the "Harmonize names between project and library" check box if you want to have the names corrected.
6. Confirm your entries with "OK".

Result

Depending on your settings, the names and the path structure in the project are harmonized with the library.

The changes to the project are logged. The log is available under "Common data > Logs" in the project navigation.

See also

Library basics (Page 439)

Overview of the library view (Page 447)

Overview of the library management (Page 451)

9.15 Clean up library

You can clean up the project library and the global libraries to remove types or versions that are not connected to any instance in the project. This step provides more clarity within the libraries and decreases the size of the library.

Cleaning up the project library

To clean up the project library, follow these steps:

1. Open the library management.
2. Click on "Clean up library" in the toolbar.
   The "Clean up project library" dialog box opens.
3. Select the scope in which types or types versions are to be deleted:
   - To retain the version with the highest version number, even if this has no instance, select the option "Delete old type versions and retain the newest type version".
   - Select the option "Delete complete types" to delete the complete type if no version is connected to an instance.
4. Confirm your entry with "OK".
   Depending on your selection, either unused type versions or types are removed from the project library.
   The changes are logged. The log is available under "Common data > Logs" in the project navigation.

Clean up global library

To clean up a global library, follow these steps:

1. Open the global library in not write-protected mode.
2. Open the library management.
3. Click on "Clean up library" in the toolbar.
   The "Clean up global library" dialog box opens.
4. Click "OK".
   Unused type versions are deleted. The latest version of a type is always retained.
   The changes are logged. The log is available in the "Common data > Logs" folder in the level below the global library.

See also

Library basics (Page 439)

Overview of the library view (Page 447)

Overview of the library management (Page 451)

Displaying logs of global libraries (Page 463)

9.16 Comparing library elements

Introduction

You can compare devices from libraries with devices from both the current project as well as from the same or another libraries or reference projects. Note, however, that reference projects are write-protected. You can also compare instances in a device with their type version in a library. Not all actions are available for the comparison with types. You cannot, for example, overwrite an instance of a newer version with an older type version from the library.

When comparing library elements, you can always switch between automatic and manual comparison.

Procedure

To compare library elements with the device data of a project, follow these steps:

1. In the project tree, select the device whose data you want to compare to a library element and which allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Open the "Libraries" task card.
4. Select the library element that you want to compare to the device data.
5. Drag the library element into the right drop area of the compare editor.
   You can identify the status of the objects based on the symbols in the status and action area. When you select an object, the object properties and the corresponding object of the assigned device are clearly shown in the property comparison.
   You can drag other devices into the drop areas from the current project, a library or from a reference point at any time to start a new comparison. It does not matter which device you drag into the drop area.

See also

Using the compare editor (Page 362)

Carrying out offline/offline comparisons (Page 361)

Using the library view (Page 447)

Library basics (Page 439)

Overview of the "Libraries" task card (Page 443)

Overview of the library view (Page 447)

Using the element view (Page 445)

Using global libraries (Page 458)

9.17 Exporting library texts

Description

You can export library element texts for translation and then reimport them. The texts are exported to an Office Open XML file with the extension ".xlsx". This can be edited in Microsoft Excel or a number of other spreadsheet programs.

The following export options are available:

- Exporting texts of selected types and master copies
  You can select individual types or master copies, and use the shortcut menu to export only the texts of the selected object.
  Or you can select and export multiple or all types or master copies by making multiple selections.
- Exporting texts of all library elements of the project library or a global library
  You can use the buttons in the toolbar of the "Project library" or "Global libraries" pane to export all texts of a project library or a global library.

If multiple versions of one type exist, the texts of the latest version are exported.

Note

No exporting of library texts

If the current version of a type has not been released yet and has the extension "In test" or "In work", no library texts are exported.

Note

Row limit in Microsoft Excel

Note that spreadsheet programs may be able to process only a certain number of rows. Microsoft Excel 2003 supports a maximum of 65536 rows, for example. Later versions of Microsoft Excel support more rows.

Defining the source and target languages in a project library

The selection of possible source and target languages in the project library depends on the languages you have defined as project languages.

You can find additional information here: Select project languages (Page 409)

Defining the source and target languages in global libraries

In a global library you can define the library languages yourself unless the library has write protection.

You can display the selection of possible languages by double-clicking on the "Library languages" item in the "Languages & Resources" subfolder. For a write-protected library the library languages are displayed as read-only in the library view.

You can find additional information here: Defining library languages (Page 461)

Requirement

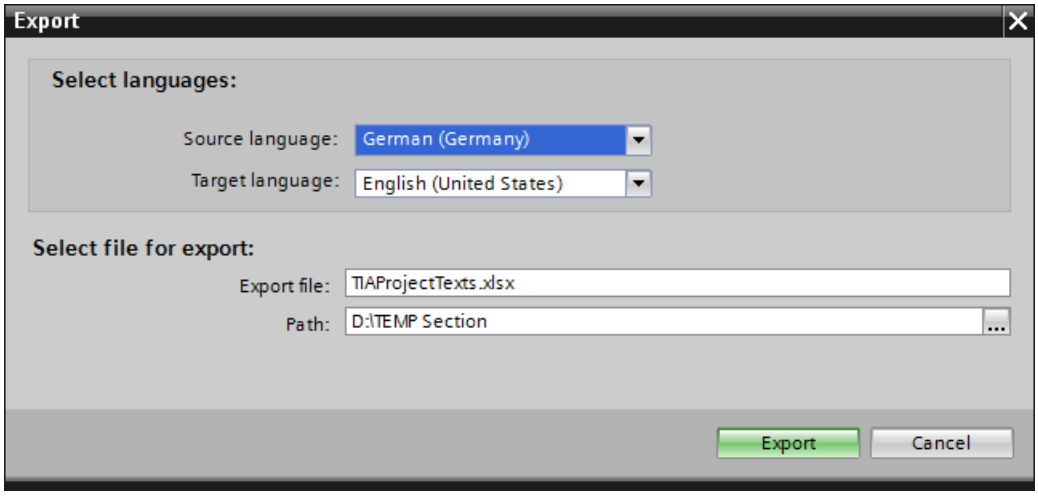
Activate the required target language:

- For a project library in the project languages
- For a global library in the library languages

Exporting texts of library elements

To export the texts of the selected library elements, follow these steps:

1. Select the required library elements (individual library element, multiple library elements, a folder or a complete library).
2. Either right-click and select "Export library texts" from the shortcut menu or click the "Export library texts" button in the toolbar.
   The "Export" dialog box opens.



3. Select the required source language and the required target language.
4. Give the export file the required name.
5. Set the required path to the storage location of the export file.
6. Click "Export".

Result

The export was successfully completed and the ".xlsx" file was created in the storage location.

	A	B	C	D	E	F
1	Category	ViewPath	Internal ID	Substitutions	de-DE*	en-US
2	<BlockCommentCategoryData>	Program blocks\	a5exBpTAQ6Af2xKwOQN99f		Start Motor	
3	<BlockCommentCategoryData>	Program blocks\	a5exBpTAQ6Af2xKwOQN99f		Stop Motor	

Write the English translations in the "en-US" column, in order to be able to then import them.

9.18 Importing library texts

Description

After external translation in a spreadsheet program, you import the texts into the TIA Portal. You can import the texts in the project library or a global library at the following locations:

- Using the "Import library texts" shortcut menu command
- Using the "Import library texts" button.

Regardless of how many library elements you have selected, all texts from the import file are imported for the entire library. The library elements from the import file must match the library elements of the library into which the texts are to be imported. The target languages you want to import must be enabled in the project or in the global library.

When you import texts for master copies, the existing texts are overwritten with the new texts. When you import texts for types, only the texts of the latest version are overwritten with the new texts.

Note

No import of library texts possible

No import of library texts possible if one of the following options applies:

- The library texts belong to a type instance which is contained in a master copy.
- A version of a type has not yet been released within the project library and has the extension "in test" or "in work".
- The global library is write-protected.
- It is a system library.

Note

Importing library texts for a multiuser project.

To also import library texts from objects that are not marked, the library texts in a multiuser project must be imported in the server project view.

Requirement

You have entered the translations in the target language in the export file.

	A	B	C	D	E	F
1	Category	ViewPath	Internal ID	Substitution	de-DE*	en-US
2	<BlockCommentCategoryData>	Program blo	a5exBpTAQ6Af2xKwOQN		Start Motor	Start the motor
3	<BlockCommentCategoryData>	Program blo	a5exBpTAQ6Af2xKwOQN		Stop Motor	Stop the motor

If you want to import library texts into a project library, the required target languages must be activated in the project languages.

If you want to import library texts into a global library, the required target languages must be activated in the global library.

Importing all texts of a project library or a global library

To import all translated library texts, follow these steps:

1. Select the project library or a global library.
2. Either right-click and select "Import library texts" from the shortcut menu or click the "Import library texts" button in the toolbar.
   The "Import" dialog box opens.
3. Select the file to be imported.
4. You have the option to also import the source language by enabling the option Import source language".
5. Click "Import".
   The "Import completed" dialog opens after the import.

Result

- Import successfully completed:
  The latest type version contains the translated texts after a successful import.
- Import completed with warnings:
  The reason for the error is either indicated directly in the dialog or you can click the link "Click here to view the log file" to view the causes of the error.
- Import failed:
  To find the log with the causes of the error, click the link "Click here to view the log file".

Editing devices and networks

10

10.1 Configuring devices and networks

10.1.1 Hardware and network editor

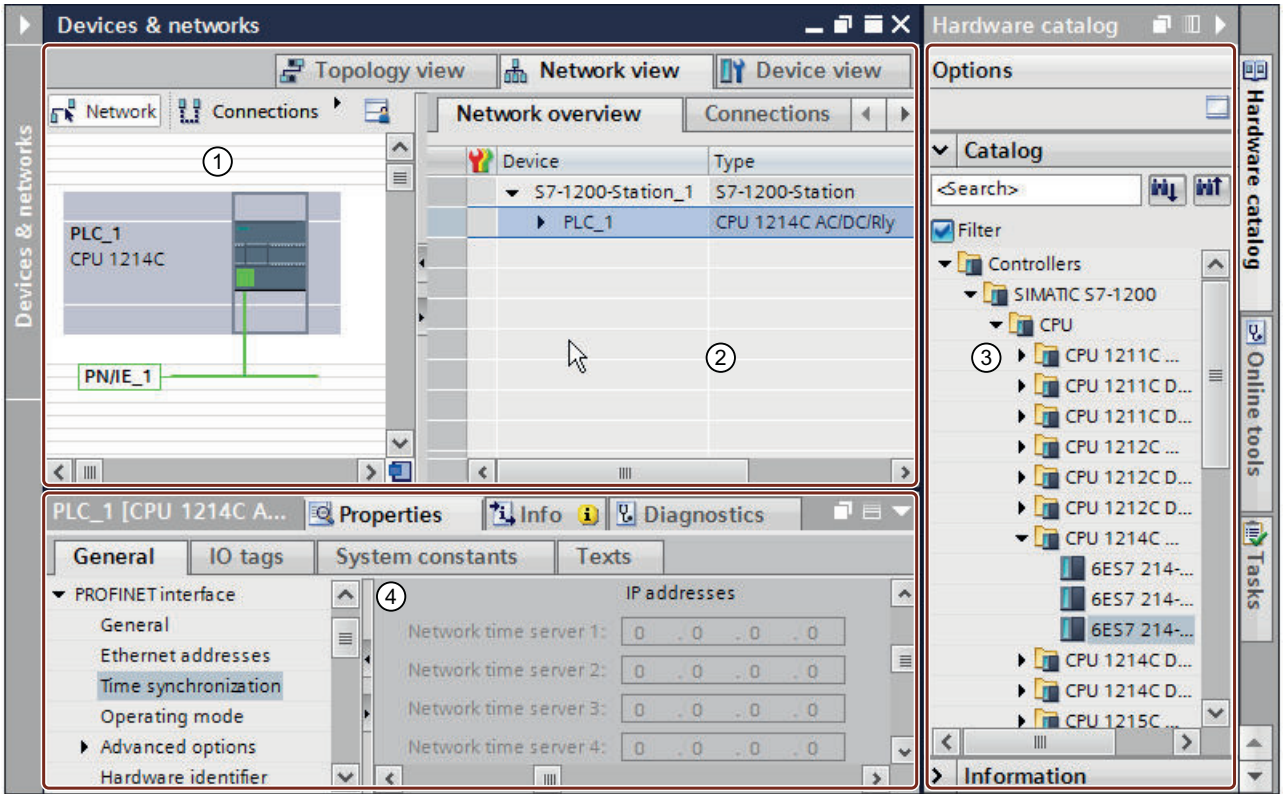
10.1.1.1 Overview of hardware and network editor

Function of the hardware and network editor

The hardware and network editor opens when you double-click on the "Devices and Networks" entry in the project tree. The hardware and network editor is the integrated development environment for configuring, networking and assigning parameters to devices and modules. It provides maximum support for the realization of the automation project.

Structure of the hardware and network editor

The hardware and network editor consists of the following components:



① Device view (Page 530), Network view (Page 527), Topology view (Page 536): Graphic area

② Device view (Page 530), Network view (Page 527), Topology view (Page 536): Table area

③ Hardware catalog (Page 544)

④ Inspector window (Page 539)

The hardware and network editor provides you with three views of your project. You can switch between these three views at any time depending on whether you want to produce and edit individual devices and modules, entire networks and device configurations or the topological structure of your project.

Drag the devices and modules you need for your automation system from the hardware catalog to the network, device ot topology view.

The inspector window contains information on the object currently marked. Here you can change the settings for the object marked.

10.1.1.2 Overview of settings for hardware configuration

In "Options > Settings > Hardware configuration", you can make various settings relating to the hardware configuration.

Overview

The following table provides an overview of the settings for the hardware configuration:

Group	Setting	Description
Information on product support	Deactivated	Prevents access to the Siemens Industry Online Support
	Via Internet	Enables access to product information about individual devices in the hardware catalog via the Internet.
Topology overview	Temporarily assigning an IP address	Assigns a temporary IP address for topology discovery if a device does not have a valid IP address. Topology information (LLDP) cannot be read from a device without valid IP address.
	Show a warning if the option is activated	Shows a warning when a temporary IP address is assigned to a device during topology discovery.
Compiling and downloading	Download module comment	Transfers any existing comments in addition to the hardware configuration when loading the hardware configuration to the device. The comment is available after the device is loaded to a programming device.
	Download I&M data	This option must be enabled if you want to be able to also download I&M data to the actual hardware when downloading the hardware configuration, the software or both to a device.

See also

Changing the settings (Page 236)

Enabling product support (Page 548)

Displaying product support for hardware components (Page 549)

Topology view (Page 536)

10.1.1.3 Network view

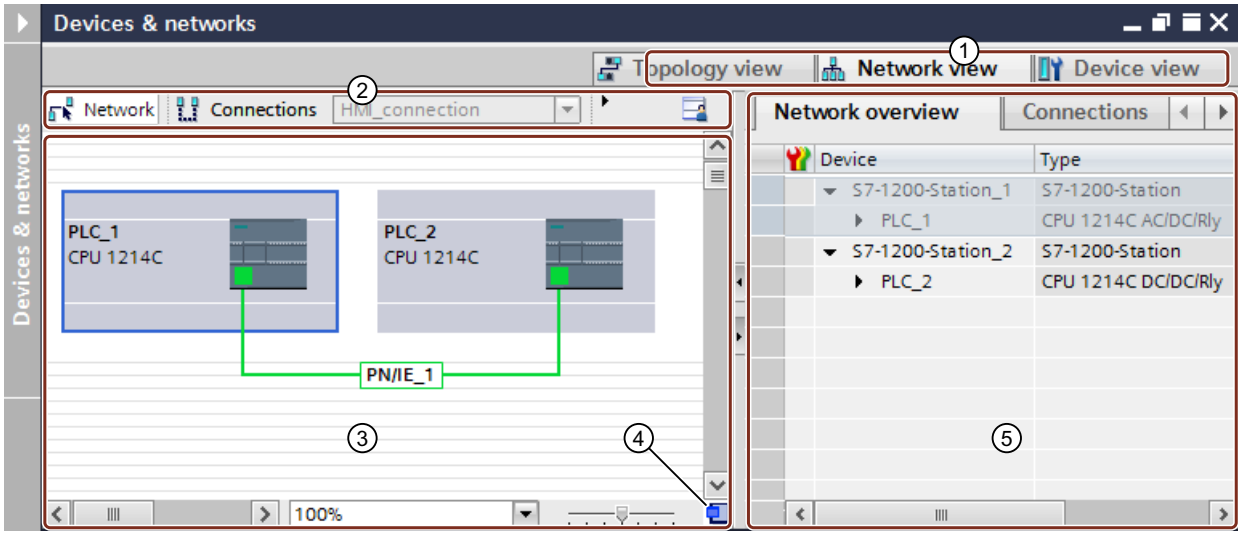
Introduction

The network view is one of three working areas of the hardware and network editor. You undertake the following tasks here:

- Configuring and assign device parameters
- Networking devices with one another
- Editing the device names

Structure

The following diagram shows the components of the network view:



① Changeover switch: device view / network view / topology view
② Toolbar of network view
③ Graphic area of network view
④ Overview navigation
⑤ Table area of network view

You can use your mouse to change the division between the graphic and table areas of the network view. The division can be changed between horizontal and vertical using a button in the toolbar. To change the position of the divider, click between the graphic and the table areas and change the spacing by moving the divider to the left or right while keeping the mouse button pressed. The Speedy Splitter (the two small arrow keys) allows you to use a single click to minimize the table view, maximize the table view or restore the last selected split.

Toolbar

The toolbar provides the following functions:

Icon	Meaning
	Mode to network devices.
	Mode to create connections. You can use the adjacent drop-down list to set the connection type.
	Mode to create relations.
	Opens the dialog for manual name assignment for PROFINET devices. For this purpose the IO device must be inserted and connected online with the IO system.
	Show interface addresses. You can edit the addresses for the MPI, PROFIBUS and Ethernet interfaces in the network view itself: Select the required address and then click on the selected address field or press [F2].
	Enables page break preview. Dotted lines are displayed at the positions where the pages break when printed.

Icon	Meaning
	You can zoom in (+) or zoom out (-) the view in steps using the zoom symbol or draw a frame around an area to be zoomed in.
	Changes the division of the graphical area and table area of the editor view between horizontal and vertical.
	Saves the current table view. The layout, width and visibility of columns in the table view is stored.

Graphic area

The graphic area of the network view displays any network-related devices, networks, connections and relations. In this area, you add devices from the hardware catalog, connect them with each other via their interfaces and configure the communication settings.

The operator controls for view control are located at the bottom edge of the graphic area:

- Select the zoom leveling using the drop-down list. You can also enter a value directly into the field of the drop-down list.
- You can also set the zoom level using the slider.
- You can re-focus the window of the graphic area using the icon in the bottom right corner.

Overview navigation

Click in the overview navigation to obtain an overview of the created objects in the graphic area. By holding down the mouse button, you can quickly navigate in the overview navigation to the desired objects and display them in the graphic area.

Table area

The table area of the network view includes various tables for the devices, connections and communication settings present:

- Network overview
- Connections
- Relations
- IO Communication
- VPN

You can use the shortcut menu of the title bar of the table to adjust the tabular display.

See also

Adding a device to the hardware configuration (Page 556)

Networking devices in the network view (Page 586)

Tabular network overview (Page 589)

Layout of the user interface (Page 238)

10.1.1.4 Device view

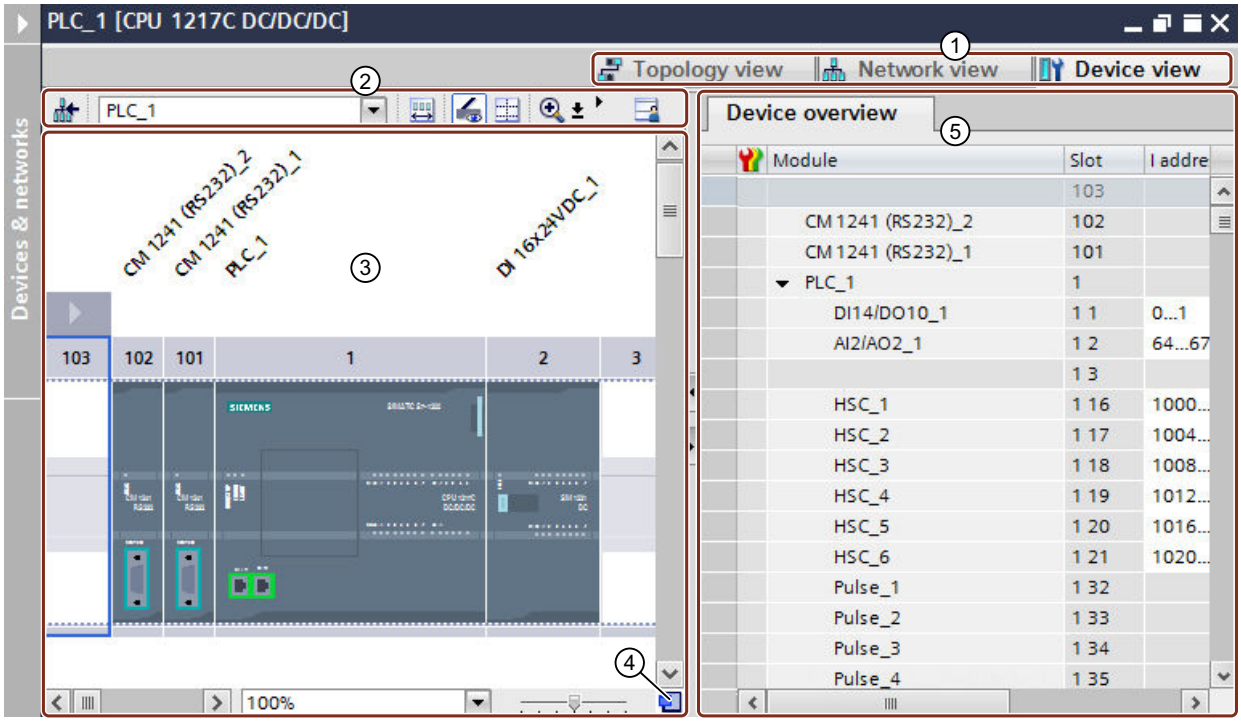
Introduction

The device view is one of three working areas of the hardware and network editor. You undertake the following tasks here:

- Configuring and assign device parameters
- Configuring and assign module parameters
- Editing the names of devices and modules

Structure

The following diagram shows the components of the device view:



① Changeover switch: device view / network view / topology view
② Toolbar of device view
③ Graphic area of the device view
④ Overview navigation
⑤ Table area of device view

You can use your mouse to change the spacing between the graphic and table areas of the device view. The division can be changed between horizontal and vertical using a button in the toolbar. To change the position of the divider, click between the graphic and the table areas and change the spacing by moving the divider to the left or right while keeping the mouse button pressed. The Speedy Splitter (the two small arrow keys) allows you to use a single click to minimize the table view, maximize the table view or restore the last selected split.

Toolbar

You can switch between the devices in your project in the device view using the drop-down list in the toolbar. The following functions are also available for device selection in addition to the drop-down list:

Icon	Meaning
	Switches to the network view. The device view can switch between the existing devices using the drop-down list.
	Show the area of unplugged modules.
	Opens the dialog for manual name assignment for PROFINET devices. For this purpose the IO device must be inserted and connected online with the IO system.
	Show module labels. You can edit the labeling in the device view itself: Select the required labeling and then click on the selected text field or press [F2].
	Enables page break preview. Dotted lines are displayed at the positions where the pages break when printed.
	You can use the Zoom icon to zoom in (+) or out (-) incrementally or to drag a frame around an area to be enlarged. With signal modules, you can recognize the address labels of the I/O channels from a zoom level of 200% or higher.
	Changes the division of the graphical area and table area of the editor view between horizontal and vertical.
	Saves the current table view. The layout, width and visibility of columns in the table view is stored.

Graphic area

The graphic area of the device view displays hardware components and if necessary the associated modules that are assigned to each other via one or more racks. In the case of devices with racks, you have the option of installing additional hardware objects from the hardware catalog into the slots on the racks.

The operator controls for view control are located at the bottom edge of the graphic area:

- Select the zoom leveling using the drop-down list. You can also enter a value directly into the field of the drop-down list.
- You can also set the zoom level using the slider.
- You can re-focus the window of the graphic area using the icon in the bottom right corner.

Overview navigation

Click in the overview navigation to obtain an overview of the created objects in the graphic area. By holding down the mouse button, you can quickly navigate in the overview navigation to the desired objects and display them in the graphic area.

Table area

The table area of the device view gives you an overview of the modules used and the most important technical and organizational data.

You can use the shortcut menu of the title bar of the table to adjust the tabular display.

See also

Rack: Basics (Page 561)

Network view (Page 527)

Device view: Area for unplugged modules (Page 535)

Rack: Inserting a module (Page 564)

Device view: Objects in the device view (Page 532)

Layout of the user interface (Page 238)

10.1.1.5 Device view: Objects in the device view

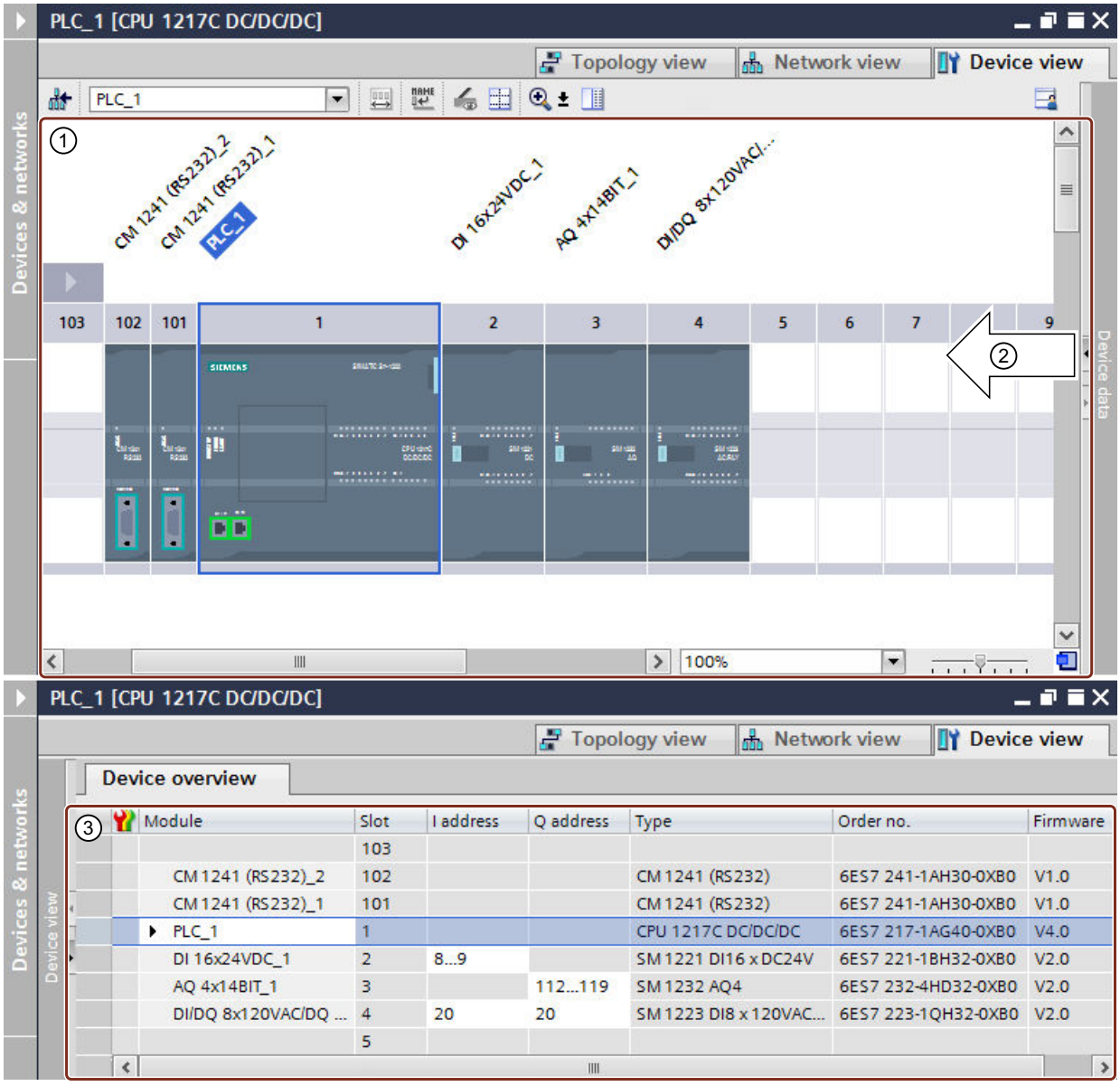
A graphic display of the rack and the devices plugged into it is shown in the left area of the device view. You can see the device overview in the right area of the device view. The device overview is a table showing you the most important information about the modules inserted in the rack. Both areas are displayed in a window. The size of both areas is adjusted using a divider. You can also use it to display one or the other area in full or to hide them.

Structure and content of device view

The offline configuration of the devices in the rack are displayed in the graphic device view. This is a symbolic representation of the configuration on the real rack.

The rack configuration is displayed as a table in the device view. Each line in the table contains the information for assigning a slot.

The following screenshot shows the device view with the configuration of a SIMATIC S7-1200 CPU.



① Graphic view of the allocation of the rack by the CPU and various modules in slots 1 to 4, as well as 101 and 102.

② You can use the divider to adjust the proportion of the device view between the left area (graphic view) and the right area (device overview). If you click on the arrows, you can quickly switch the division of the separate areas.

③ Device view with the tabular representation of the rack's slots and the inserted components. You can show additional columns and hide displayed columns using the shortcut menu of the column title.

Each line in the device overview represents one slot. The key information for each slot is displayed in the various columns:

Column	Meaning
Online status	Symbolic representation of the online status
Fail-safe	Symbolic representation for fail-safe module
Module	Name of module, can be edited in any way

Column	Meaning
Rack	Number of the rack
Slot	Slot number
I address	Input address area, can be edited in any way
Q address	Output address area, can be edited in any way
F-source address	F-source address when using fail-safe I/O
F-destination address	F-destination address when using fail-safe I/O
Type	Catalog name of module
Article no.	Article number of the module
Firmware	Firmware version of the module
Comment	Optional comment

Display in I/O channels

If you set the zoom level in the device view to at least 200%, you can see the individual I/O channels for I/O modules. If you have defined PLC tags for the channels, the names of the PLC tags are displayed.

The figure below shows the input channels of the digital input module with the two PLC tags "Test1" and "Test2" at a zoom level of 400 %.



You can select the individual I/O channels and have the following options for channels with PLC tags:

- You see the general properties of the selected PLC tag in the inspector window under "Properties".
- In the inspector window under "Info > Cross-references" you find the cross-reference information on the selected PLC tag. If you have selected the PLC tag, you can also open the cross-reference information using the shortcut menu.

See also

Device view (Page 530)

Overview of PLC tag tables (Page 4837)

10.1.1.6 Device view: Area for unplugged modules

In some cases, the modules for a hardware configuration are not assigned a slot for short periods. Such unplugged modules are moved to the area of unplugged modules, a special area in the device view.

Adding modules to the storage area

The modules that, for example, should be assigned to a device using a copy action but for which the corresponding rack does not have a free compatible slot, are moved automatically into the area of unplugged modules.

Under the following conditions, modules are automatically added to the area of unplugged modules:

- In the network view, a module is moved to a device but the rack does not have a compatible free slot.
- In the device view, a module is moved or copied from the rack, the hardware catalog or the project tree straight into the storage area.

CPs and FMs that occupy a network resource can be moved into the area of unplugged modules but will lose the network resources they have been assigned.

You can add modules to the area of unplugged modules by means of drag-and-drop, for example. To do this, the area must be opened.

Using the area of unplugged modules

Use the corresponding button to open the area of unplugged modules.

You can find the area of unplugged modules in the device view.



You open the area of unplugged modules with the respective icon in the toolbar of the device view (Page 530).

Icon	Meaning
	Open area of unplugged modules

Note

To free up slots, move modules from your configuration into the storage area and plug the modules you want from the storage area into the freed up slots.

You can use this approach to temporarily move modules whose parameters have already been assigned out of the configuration without deleting them.

Treatment of modules in the storage area

The following rules apply to modules in the storage area:

- The modules appear in the project tree under the corresponding device in the "Local modules" folder.
- The modules retain all settings and parameters previously assigned.
- The modules are not taken into account during loading to a target system. This means that a consistency check is not undertaken for modules in the area of unplugged modules.
- Using the context menu, the modules can be copied, cut, or deleted, for example.

10.1.1.7 Topology view

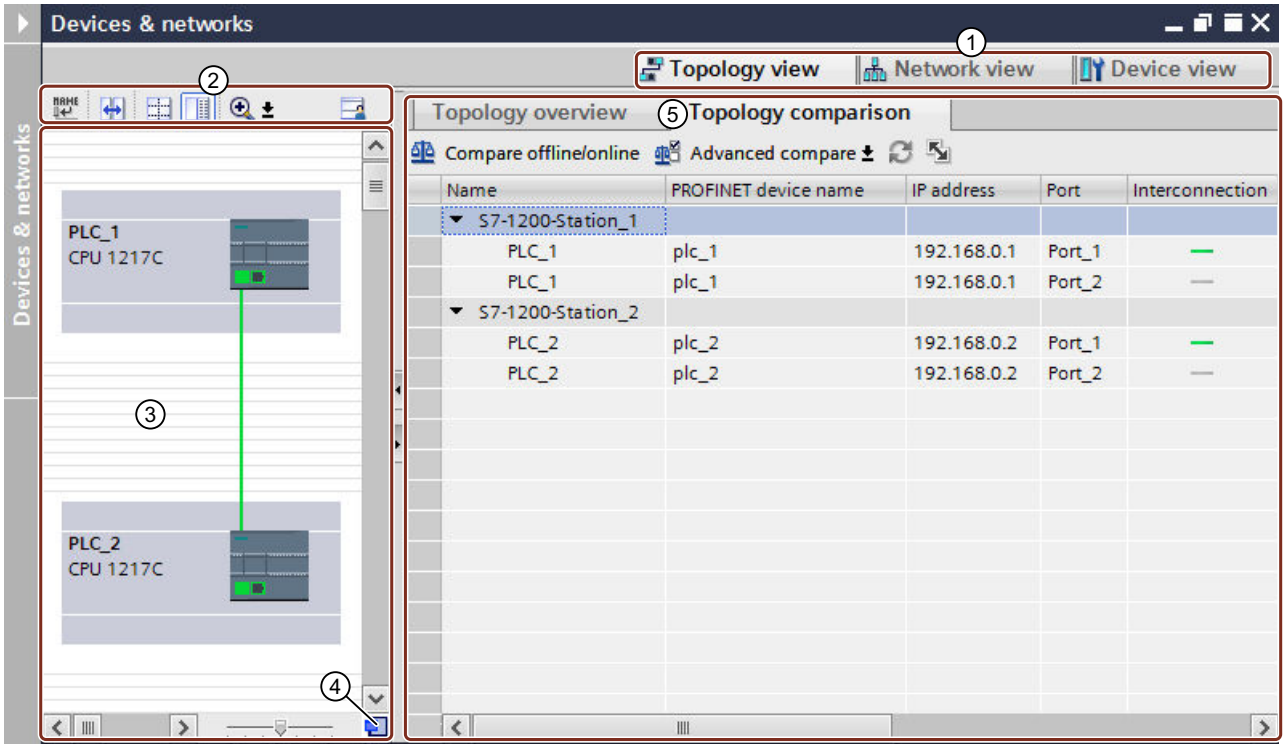
Introduction

The topology view is one of three working areas of the hardware and network editor. You undertake the following tasks here:

- Displaying the Ethernet topology
- Configuring the Ethernet topology
- Identifying and minimizing differences between the desired and actual topology
- Editing the device names

Structure

The following figure provides an overview of the topology view.



① Changeover switch: device view / network view / topology view
② Topology view toolbar
③ Graphic area of the topology view
④ Overview navigation
⑤ Table area of the topology view (topology overview)

You can use your mouse to change the spacing between the graphic and table areas of the topology view. The division can be changed between horizontal and vertical using a button in the toolbar. To change the position of the divider, click between the graphic and the table areas and change the spacing by moving the divider to the left or right while keeping the mouse button pressed. The Speedy Splitter (the two small arrow keys) allows you to use a single click to minimize the table view, maximize the table view or restore the last selected split.

Toolbar

The toolbar provides the following functions:

Icon	Meaning
	Opens the dialog for manual name assignment for PROFINET devices. For this purpose the IO device must be inserted and connected online with the IO system.
	Changes the position of the devices in the topology view so that the new position is as close as possible to the position in the network view.
	Enables page break preview. Dotted lines are displayed at the positions where the pages break when printed.

Icon	Meaning
	You can zoom in (+) or zoom out (-) the view in steps using the zoom symbol or draw a frame around an area to be zoomed in.
	Changes the division of the graphical area and table area of the editor view between horizontal and vertical.
	Saves the current table view. The layout, width and visibility of columns in the table view is stored.

Graphic area

The graphic area of the topology view displays Ethernet modules with their appropriate ports and port interconnections. Here you can add additional hardware objects with Ethernet interfaces. See: Adding a device to the hardware configuration (Page 556)

The operator controls for view control are located at the bottom edge of the graphic area:

- Select the zoom leveling using the drop-down list. You can also enter a value directly into the field of the drop-down list.
- You can also set the zoom level using the slider.
- You can re-focus the window of the graphic area using the icon in the bottom right corner.

Overview navigation

Click in the overview navigation to obtain an overview of the created objects in the graphic area. By holding down the mouse button, you can quickly navigate in the overview navigation to the desired objects and display them in the graphic area.

Table area

The table area consists of the following two tabs:

- "Topology overview" tab
- "Topology comparison" tab

The "Topology overview" tab displays the Ethernet or PROFINET modules, their ports and port interconnections in a table. This table corresponds to the network overview table in the network view.

In the "Topology comparison" tab, you identify and minimize differences between the desired and actual topology.

See also

Overview of settings for hardware configuration (Page 526)

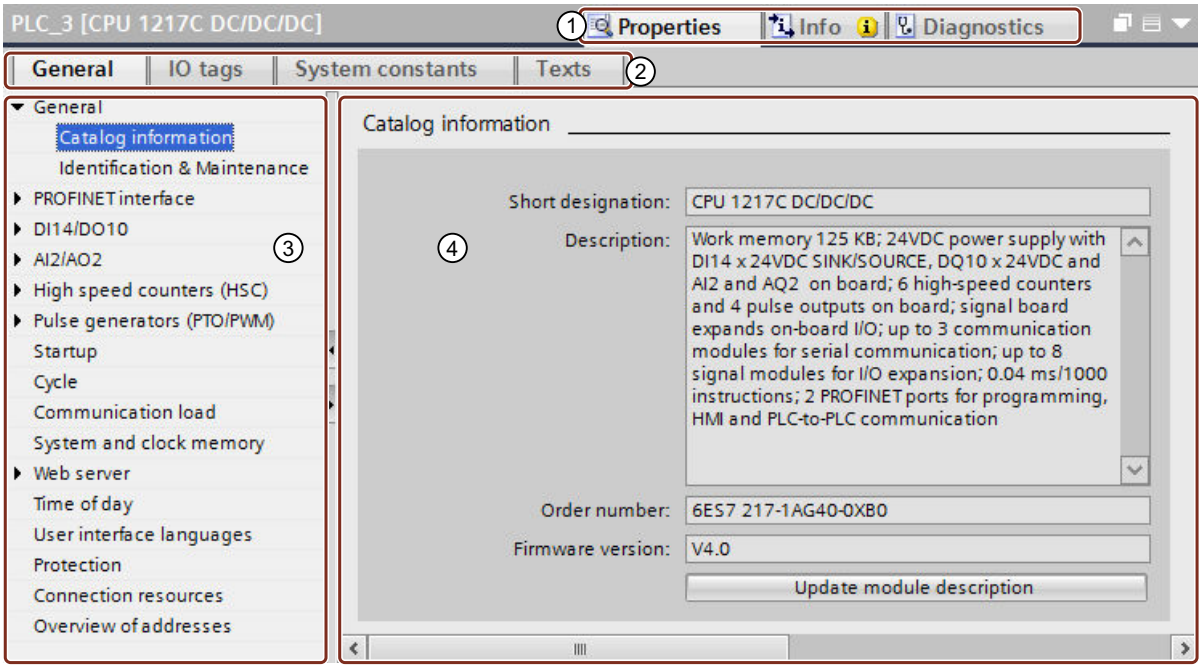
Layout of the user interface (Page 238)

10.1.1.8 Inspector window

The properties and parameters shown for the object selected can be edited in the Inspector window.

Structure

The Inspector window consists of the following components:



① Switch between various information and work areas

② Switch between various tabs of the information and work areas

③ Navigation between various pieces of information and parameters

④ Display showing the selected information and parameters

Description of functions

The information and parameters in the Inspector window are split into different types of information with different tabs:

- Properties
  This is where you can configure the device. You can make the settings for the hardware, edit tags and system constants and manage project texts. The following tabs are available here:
  - General (see Properties and parameters (Page 575))
  - IO tags (see Overview of PLC tag tables (Page 4837))
  - System constants (see Rules for global system constants (Page 4853))
  - Texts (see Project text basics (Page 407))
- Info
  In this area, you will find general information on the device in the project, a cross-reference list and messages and notes on compiling the device. The following tabs are available here:
  - General
  - Cross-reference list (see Cross-references in the Inspector window (Page 5172))
  - Compiling (see Information on compiling (Page 347))
- Diagnostics
  When you go online with the device, you will find the diagnostic information for the device and its connections here. The relevant diagnostic messages are also displayed. The following tabs are available here:
  - Device information (see Hardware diagnostics)
  - Connection information (see Connection diagnostics)
  - Message display (see Message display)

To display the corresponding information and parameters, click in the relevant area.

Display of device properties

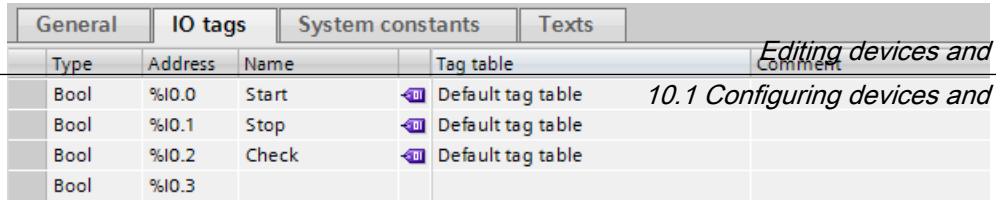
The "Properties" area is the most important one for configuring an automation system. The editing options for this area are therefore described in detail below. For information on the other areas, please see the links above to other sections of the information system.

"General" tab

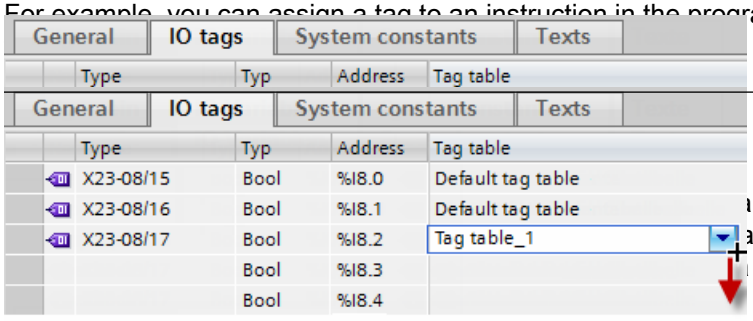
Display the properties and settings of the device or module. Here you can edit the settings and parameters. The left pane of the Inspector window is used for area navigation. Information and parameters are arranged there in groups. If you click on the arrow symbol to the left of the group name, you can expand the group if sub-groups are available. If you select a group or sub-group, the corresponding information and parameters are displayed in the right pane of the Inspector window and can be edited there too.

"IO tags" tab

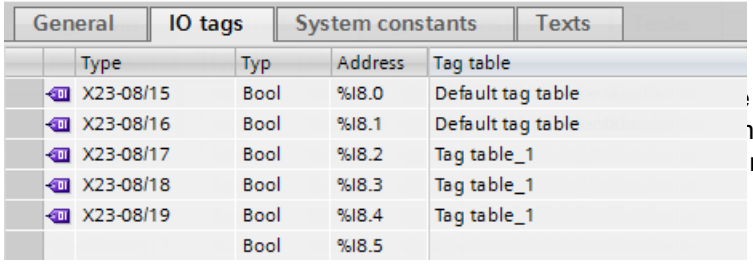
Display the IO tags of the device, the PLC and the individual modules. You can assign names for the tags, assign the tags to the user-defined tag tables via a drop-down list, and enter comments for the tags. The IO tags are also shown in the PLC tag table.



You can click on the tag symbols in the table to drag and drop tags for use in other editors. For example, you can assign a tag to an instruction in the programming editor or copy a tag

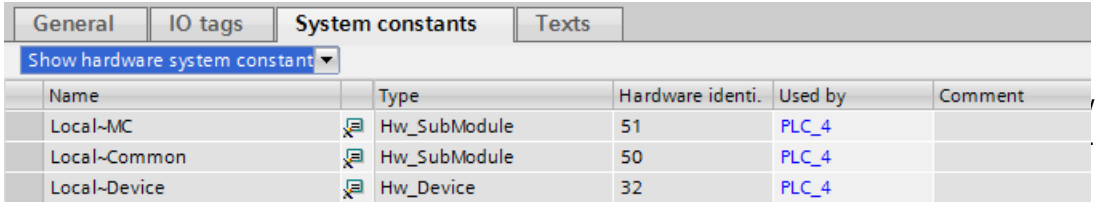


lso click on the name of a tag
above or below. A new tag is
number is added to the name
a number, the number is



new tags for the tag table or
the drop-down list and drag
new tags in the corresponding

If existing tags are affected by these actions, a dialog appears to ask whether the affected tags should be overwritten or whether new tags should only be inserted in empty fields. In the latter case, this means that only three tags are added to an area with nine fields and six existing tags; the existing tags are not changed.



With I-slaves and I-devices, the system constants of the DP master or IO controller relating to the I-slave or I-device are also displayed.

If you click on the link in the column in which the use of the constant is displayed, the device view of the corresponding device opens. The system constants are also shown in the PLC tag table.

You can click on the constant symbols in the table to drag and drop system constants to use in other editors. For example, you can assign a system constant to an instruction in the programming editor or create a system constant as a user constant in another device.

"Texts" tab

Display the reference language and specify the text source for the project texts. Please see the rest of the information system.

See also

Overview of hardware and network editor (Page 525)

Translating project texts (Page 411)

Addressing modules (Page 1045)

Inspector window: Displaying UDTs (Page 542)

10.1.1.9 Inspector window: Displaying UDTs

Under "Properties > IO tags", the Inspector window displays self-defined data structures comprising multiple elements. These UDT (User Defined Structure) PLC data types are displayed differently from simple IO tags.

Display of UDTs in the I/O tag table

Structured PLC tags that occupy the address space of a module are displayed along with the data items they contain in combined columns. Occupied address areas are indicated by vertical lines and individual addresses by small diamonds.

The following symbols are used for display in line with IO module channel address assignment by the address area of the UDTs:

Symbol	Meaning
	Start of a data item address area covering several channels. The address area of the entire channel is used.
	Part of a data item address area covering several channels. The address area of the entire channel is used.
	End of a data item address area covering several channels. The address area of the entire channel is used.
	Complete address area of a data item over the address area of a channel used in full. Example: Only the input word of a channel is completely used for a data item.
	Start of a data item address area covering several channels. Only part of the address area of the channel is used.
	End of a data item address area covering several channels. Only part of the address area of the channel is used.
	Complete address area of a data item. Only the first part of the channel address is used. Example: Only the first byte of a word is assigned.
	Complete address area of a data item. Only the second part of the channel address is used. Example: Only the second byte of a word is assigned.
	Single address of a data item for a channel that only has one address. Example: Individual bit access to the channel address of a digital input.
	Single address of a data item for a channel that has multiple addresses. Example: Individual bit access to the channel address of an analog input.

Application example

The figure below shows a tag table with a UDT "Motor" PLC data type and its data items "Start", "Stop" and "Dummy":

	Name	Data type	Address
1	Start	Bool	%I2.0
2	Stop	Bool	%I2.1
3	ErrorCode	Byte	%IB3
4	Motor	"MotorUDT"	%I0.0
5	Dummy	Word	%IW0
6	Start	Bool	%I2.0
7	Stop	Bool	%I2.1
8	Status	Array[0..7] of B...	%I4.0
9	Status[0]	Bool	%I4.0
10	Status[1]	Bool	%I4.1

For a DI32 input module with input addresses I0.0 to I3.7, you see the combined column "Motor ("MotorUDT")" under the IO tags in the Inspector window. This combined column contains a column for the address area of the UDT "Motor" and a column for the data items contained in the UDT: "Motor.Dummy", "Motor.Start" and "Motor.Stop". The "Status" data item only starts at address I4.0 and is not displayed in the IO tags for this input module, because the address area of the 32 digital channels only runs from I0.0 to I3.7. The tag "ErrorCode (Byte)" is not part of the UDT. It is therefore not displayed in the combined column under UDT "Motor", but is instead shown in a separate column:

General | IO tags | System constants | Texts

Type	Address	Name	Tag table	Motor ("MotorUDT")		ErrorCode (Byte)
	Bool	%I0.0		Motor	Motor.Dummy	
	Bool	%I0.1				
	Bool	%I0.2				
	⋮	⋮		⋮	⋮	
	Bool	%I1.7				
Start	Bool	%I2.0	Default tag table		Motor.Start	
Stop	Bool	%I2.1	Default tag table		Motor.Stop	
	Bool	%I2.2				
	⋮	⋮		⋮		
	Bool	%I2.7				
	Bool	%I3.0				
	Bool	%I3.1				
	Bool	%I3.2				
	⋮	⋮		⋮		⋮
	Bool	%I3.7				

The UDT "Motor" in this module occupies an address area from I0.0 to I3.7. Within this address area, the "Dummy" tag occupies the address area from I0.0 to I1.7, and the two tags "Start"

and "Stop" occupy the single addresses I2.0 and I2.1. The "ErrorCode (Byte)" tag that is not part of the UDToccupies addresses I3.0 to I3.7 in its own column.

Note

The table of IO tags only shows the data in the address area of the selected device; in this case the address area of the input module DI32. If another input module with at least 16 channels is connected, the display of the address assignment is continued by the UDT with the extended address area: The address area of the UDT "Motor" runs to address I5.7, and the status data item it contains covers addresses I4.0 to I5.7. The "Status" data item occupies single addresses for individual status bits from I4.0 to I4.7. An additional column is added to the combined column of the UDT to display the status bits.

See also

Other useful information regarding structured PLC tags (Page 4849)

Creating structured PLC tags (Page 4851)

Using PLC data types (UDT) (Page 2229)

Inspector window (Page 539)

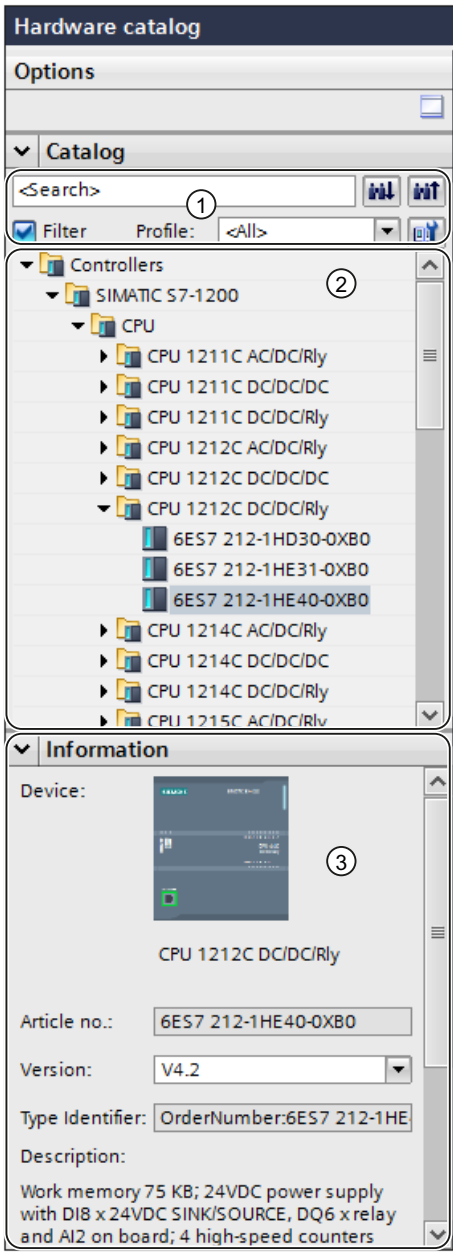
Details view (Page 260)

10.1.1.10 Hardware catalog

The "Hardware catalog" task card gives you easy access to a wide range of hardware components.

Structure

The "Hardware catalog" task card consists of the following panes:



① "Catalog" pane, search and filter function
② "Catalog" pane, component selection
③ "Information" pane

Search and filter function

The search and filter functions of the "Catalog" pane make it easy to search for particular hardware components. After entering the search text, you can start a search in the upwards or downwards direction from the current catalog position.

You can limit the display of the hardware components to certain criteria using the filter function. For example, you can limit the display to objects that you can also place within the current context or which contain certain functions. Objects that can be used in the current context include, for example, interconnectable objects in the network view or only modules compatible with the device in the device view.

You can create hardware profiles with which the objects displayed in the hardware catalog are limited. For example, you could create a hardware profile in which only certain controllers can be seen and another profile in which only certain devices of the distributed I/O are included. You can switch between the created hardware profiles.

Component selection

The component selection in the "Catalog" pane contains the installed hardware components in a tree structure. You can move the devices or modules you want from the catalog to the graphic work area of the device or network view.

Installed hardware components without a license are grayed out. You cannot use non-licensed hardware components.

Hardware components belonging to various components groups thematically are partially implemented as linked objects. When you click on such linked hardware components, a catalog tree opens in which you can find the appropriate hardware components.

Information

The "Information" pane contains detailed information on the object selected from the catalog:

- Schematic representation
- Name
- Article number
- Version number
- Description
- Type Identifier (only when activated)

Note

Type Identifier

You can activate the display of the "Type Identifier" attribute in the "Options > Settings > Hardware configuration" menu. If the Type Identifier is activated, hardware objects for use in external development environments can be uniquely identified. The Type Identifier is displayed in the "Information" pane in the hardware catalog.

See also

Browsing the hardware catalog (Page 547)

Overview of hardware and network editor (Page 525)

10.1.1.11 Browsing the hardware catalog

Introduction

Use the "Hardware catalog" task card to select the hardware components you want for a hardware configuration. Use the hardware catalog to select the interconnectable hardware components in the network and topology view and to select the modules you want in the device view.

Context filter

You can use the "Filter" option of the hardware catalog to restrict the number of displayed hardware components and the number of hardware components that can be found by searching.

When you enable the filter, components available in the hardware catalog are displayed depending on the selected view and the selected device. If you switch between the various views, the view of the filter objects is adapted to the current context. If the filter is not enabled, the entire hardware catalog is displayed in all views.

Example: With the filter enabled in the device view of an S7-1200 CPU, only S7-1200 CPUs and matching modules are displayed in the hardware catalog.

Search options

You can use the search function to search for specific entries in the hardware catalog. Note the following rules when entering search terms:

- No distinction is made between upper and lower case text.
- Dashes and blanks are ignored during the search.
- The search function considers parts of a search term.
- Multiple search terms must be separated by a space.
- The asterisk "*" is used as a wildcard.

Note

Use of wildcards in the hardware catalog

In search queries, the asterisk "*" is used as a wildcard. A search for "1A**0", for example, also return hits with "1AD30" or "1AE40". The hardware catalog also contains modules with wildcards in their article numbers. The wildcards in the various article numbers can be either asterisks or the lowercase letters "x", "y" or "z". If you search for the specific article number "3RW4 422-1BC34", for example, your search will also return articles with the number "3RW4 422-*BC**" and "3RW4 4xx-1By3z".

You start the search from an object highlighted in the hardware catalog and either search upwards or downwards.

Icon	Meaning
	Search down
	Search up

Browsing the hardware catalog

If you want to browse the hardware catalog, proceed as follows:

1. Click in the entry field of the search function
2. Enter a search term. The search includes the following elements:
   - Name of device or module
   - Article number (MLFB)
   - Description in "Information" pane
3. Click on either the "Search down" or "Search up" button.

Note

To ensure the right search direction, note which point you have marked in the hardware catalog. To browse the entire catalog, click on the topmost object of the hardware catalog and start the search once you have entered the search term by clicking "Search down".

The first match with the search term found is displayed as the result. For more search results, again click on the "Search down" or "Search up" button.

Observe the context filter of the hardware catalog. If this is selected, the search in the hardware catalog is restricted to the displayed inserted hardware components.

See also

Hardware catalog (Page 544)

10.1.1.12 Enabling product support

For each device in the hardware catalog, you have the option of displaying additional information that is stored in the Siemens Industry Online Support. By default, the function is disabled. Instructions for enabling the function are given below.

Requirement

The TIA Portal has access to the Internet.

Procedure

To enable access to Siemens Industry Online Support, follow these steps:

1. In the "Options" menu, select the "Settings" command.
2. Open the "Hardware configuration" group in the area navigation.
3. Select the "Via Internet" check box.

Result

You can access product support, FAQs and manuals in the hardware catalog via the shortcut menu for the module.

See also

Displaying product support for hardware components (Page 549)

Overview of settings for hardware configuration (Page 526)

10.1.1.13 Displaying product support for hardware components

In the hardware catalog, you have direct access to information that is stored for each module in Siemens Industry Online Support. You can jump directly to the following pages in Siemens Industry Online Support:

- Information on product support
- FAQs
- Manuals

Requirement

- You have access to the Internet.
- Access to Product Support is enabled in the settings of the TIA Portal.
  For information on how to enable the function, refer to section "Enabling product support (Page 548)".

Procedure

To display the information for a particular module in Siemens Industry Online Support, follow these steps:

1. Navigate to the required module in the hardware catalog.
2. Right-click the module.
3. Select one of the following entries in the shortcut menu:
   - Information on product support
   - FAQs
   - Manuals

Result

The default browser set in the operating system is opened and the relevant page in the Siemens Industry Online Support is loaded.

See also

Enabling product support (Page 548)

Overview of settings for hardware configuration (Page 526)

10.1.1.14 Printing hardware and network configurations

Printout of hardware and network configurations

You can print out the following elements of the hardware and network view as part of the project documentation:

- Graphic network view
- Network overview table
- Graphic device view
- The device overview table
- The parameters of the object currently selected in the editor

Printout of editor content

If you start a printout within an opened editor and no module is selected, the content of the editor is always printed. This includes a graphic representation of the editor as well as the table for the editor. You can adapt the scope of the printout. You can specify whether only the graphic view, the table or both together are to be printed. Read section "Printing: Changing settings (Page 552)" for more on this.

If the graphic is larger than the page layout you have selected, the printout is continued on the next page. No content is lost this way. Alternatively, you can change the zoom level of the graphic representation to fit the printout on one page. The printout is always made in the currently selected zoom setting.

To check that all content fits on one page, you can either use the print preview or activate the page break preview. When page break preview is activated, dashed lines are displayed within the graphic editor at the location where the page break is later made.

Printing very large tables

If a table is larger than the print area and therefore cannot be fully printed, the content of the table is not printed as a table, but instead as pairs between value and key.

Example:

Object name	Property 1	Property 2
Object A	Value A1	Value A2
Object B	Value B1	Value B2

In this case, the printout has the following appearance:

Object A

Property 1: Value A1

Property 2: Value A2

Object B

Property 1: Value B1

Property 2: Value B2

You can also preset this as a template so that tables are always printed as pairs between the key and the value. Read section "Changing the print settings (Page 382)" for more on this.

Printing module parameters

Parameters of selected modules are printed out along with the current value settings in text form. All parameters from corresponding modules are also printed. For example, if you have selected a CPU, the parameters of an inserted signal board, if present, are printed as well.

You can determine the scope of the module parameters to be printed. In the "Print" dialog, select whether all properties and parameters of a module are to be printed or whether to use the compact printout. If you select the compact form, only the entries in the "General" area of the module properties are printed. Comments on modules, as well as the author and module description, are excluded. In compact mode, the following module parameters are therefore printed, for example:

- Module specifications
  Name, module slot, short description, article number, firmware version
- Name of the PROFINET interface
- Subnet specifications
  Name of the subnet, ID of the S7 subnet

See also

Printing: Changing settings (Page 552)

Documentation settings (Page 380)

Creating a print preview (Page 394)

Printing project data (Page 397)

Printing: Enabling page break preview (Page 552)

10.1.1.15 Printing: Changing settings

Changing the scope of the printout

When printing from an editor, you can specify whether both graphics and tables are to be printed or just one of the two. Both are printed by default.

Procedure

To change the scope of the printout, proceed as follows:

1. In the "Options" menu, select the "Settings" command.
2. In the area navigation, open the "Print settings" parameter group under "General".
3. Scroll to the "Hardware configuration" group.
4. Select or clear the "Active graphic view" check box, depending on whether you want to print the graphics of the network and device view as well.
5. Select or clear the "Active table" check box, depending on whether you want to print the table for the editor as well.

See also

Printing hardware and network configurations (Page 550)

10.1.1.16 Printing: Enabling page break preview

You can activate the page break preview for the printout in the graphic editor. If this option is activated, dashed lines are shown within the graphic editor at the locations where page breaks are later made during printout.

Procedure

Proceed as follows to activate the page break preview:

1. Select the graphic area of the corresponding view.
2. Click on the "Show page break" symbol in the toolbar of the graphic editor.
   Dashed lines are displayed within the graphic editor at the location a page break is later made.
3. To modify the frame layout, select the "Print" command in the "Project" menu.
4. To disable page break preview, click again on the "Show page break" symbol in the toolbar of the graphic editor.

10.1.1.17 Keyboard operation: Navigation in the editor

You can use shortcut keys in the network and device view to navigate between the components of the hardware and network editor and its objects.

Navigating between elements and functions

Function	Shortcut keys
Switch to the next lower selection level You can for example, use <Return> to switch from a selected rack to the lower selection level of the devices and modules that are snapped onto it. If a device is selected, you can use <Return> to switch to the lower selection level of the interfaces that are displayed on the device.	<Return>
Switch to the next higher selection level You can use <Esc>, for example, to switch from a selected interface to the higher selection level of the devices and modules. If a device is selected, you can use <Esc> to switch to the higher selection level of the rack.	<Esc>
Navigation between objects in the current selection level You can use the arrow keys to switch between the objects within a current selection level. To change the selection level, use the <Return> or <Esc> keys.	<Up arrow> <Down arrow> <Right arrow> <Left arrow>
Switches to the device view	<Ctrl+Shift+D>
Switches to the network view	<Ctrl+Shift+N>
Switches to the topology view	<Ctrl+Shift+T>
Switch between editor elements Use the <Tab> key to switch from one editor element to the next element. Use <Shift +Tab> to switch to the previous element. You can switch, for example, between the graphical view, Speedy Splitter, table view or underlying tabs.	<Tab> <Shift+Tab>
Switch between tabs Use the <Ctrl+Tab> keys to switch from one tab to the next tab on the right. Use <Ctrl +Shift+Tab> to switch to the next tab to the left. You can use these keys, for example, to switch between the device view, the network view and the topology view.	<Ctrl+Tab> <Ctrl+Shift+Tab>

Opening elements and functions

Function	Shortcut keys
Opening the online and diagnostics view When a device is selected, <Ctrl+D> opens the online and diagnostics view for the selected device.	<Ctrl+D>
Opening the download to device dialog When a device is selected, <Ctrl+L> opens the advanced download dialog.	<Ctrl+L>
Add new device <Ctrl+N> opens the dialog for adding a new device.	<Ctrl+N>
Opens the "Hardware catalog" task card	<Ctrl+Shift+C>
Opens "Online Tools" task card	<Ctrl+Shift+O>

See also

Keyboard operation in the TIA Portal (Page 270)

10.1.1.18 Keyboard operation: Editing objects

You can execute some of the functions of the network and device view directly with a combination of keyboard and mouse in the hardware and network editor. The keyboard operation in tables (Page 270) corresponds to standard characteristics. Here you find the keyboard operation for the graphic work area of the network and device view.

General keyboard operation

Function	Shortcut keys
Zoom in on view in frame Drag a frame in the graphical view in order to correspondingly change the size of the view.	<Ctrl+Space> + pressed mouse button
Move view Move the mouse pointer in order to move the view.	<Space> + pressed mouse button
Cancel current operation	<Esc>
Separate connection Use <Esc> or a double-click to exit connection mode when dragging a connection.	<Esc> or double-click
Zoom in graphic view The enlargement or reduction depends on the direction of rotation.	<Ctrl> + turn mouse wheel

Selected objects

Function	Shortcut keys
Select object	Mouse click
Cut an object The selected object is copied to the clipboard and deleted from the graphical view.	<Ctrl+X>
Copy object The selected object is copied to the clipboard.	<Ctrl+C>
Paste object The object from the clipboard is inserted into the selection.	<Ctrl+V>
Delete selected object	<Del>
Select several objects 1 You can add several objects to the selected objects by clicking on them individually. Alternatively, you can use <Shift> + pressed mouse key to drag a frame around the objects that are to be selected.	<Shift> + click
Select several objects 2 You can add several objects to the selected objects by clicking on them individually. Alternatively, you can use <Shift> + pressed mouse key to drag a frame around the objects that are to be selected. When holding down the <Ctrl> key, you can use a mouse click to deselect selected objects.	<Ctrl> + click

Function	Shortcut keys
Move selection When the mouse button is pressed, you can drag devices or modules to allowed slots on a rack.	Mouse button pressed
Copy selection Using <Ctrl> + pressed mouse button you can drag devices and modules to allowed slots on a rack. This copies the devices or modules.	<Ctrl> + pressed mouse button

10.1.2 Configuring devices

10.1.2.1 Introduction to configuring hardware

To set up an automation system, you will need to configure, assign parameters and interlink the individual hardware components. The work needed for this is undertaken in the device and network view.

Configuring

"Configuring" is understood to mean arranging, positioning, and networking devices and modules within the device or network view. Racks are represented symbolically. Just like "real" racks, they allow you to plug in a defined number of modules.

An address is automatically assigned to each module. The addresses can be subsequently modified.

When the automation system is started, the CPU compares the configuration that is preset by the software with the actual configuration of the system. Possible errors can be detected and reported straight away.

Assigning parameters

"Assigning parameters" is understood to mean the setting of properties of the components used. Parameter assignment is carried out for hardware components and for data exchange settings:

- Properties of modules with assignable parameters
- Settings for data exchange between components

The parameters are loaded to the CPU and transferred to the corresponding modules when the CPU starts up. Modules can be replaced with ease since the assigned parameters are automatically loaded into the new module during startup.

Adjusting the hardware to the project requirements

You need to configure hardware if you want to set up, expand or change an automation project. To do this, add hardware components to your configuration, connect them to existing components, and adapt the hardware properties to the tasks.

The properties of the automation systems and modules are preset in such a way that in many cases there is no need for additional parameter assignment. Parameter assignment is however needed in the following cases:

- You want to change the default parameter settings of a module.
- You want to use special functions.
- You want to configure communication connections.

See also

Changing properties of the modules (Page 980)

10.1.2.2 Adding a device to the hardware configuration

Introduction

There are various ways of adding a connectable device from the hardware configuration in the network view and the topology view:

- Command "Add new device" in the project tree
- Double-click device in hardware catalog
- Drag-and-drop from the hardware catalog in network view or in topology view:
  - Text entry from the "Catalog" pane
  - Preview graphic from the "Information" pane
- "Add > Device" command from menu bar in network view or topology view
- Shortcut menu of a device in the hardware catalog for copying and pasting

A suitable rack is created along with the new device. The selected device is inserted at the first permitted slot of the rack.

Regardless of the method selected, the added device is visible in the project tree and the network view of the hardware and network editor.

Adding device using the project tree

To use the project tree to add a device to the hardware configuration, proceed as follows:

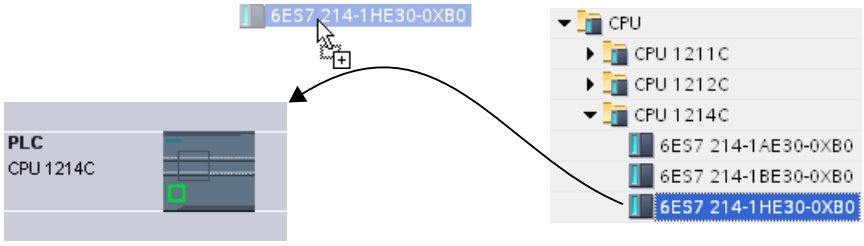
1. Click on the command "Add new device" in the project tree.
   The "Add new device" dialog box opens.
2. Display the required device in the tree structure:
   - Go to required device in the tree structure.
   - Enter a device name in the entry field.
3. Select the required device from the tree.
   More information about the device presently selected is displayed on the right-side of the dialog box.
4. If necessary, set the firmware version using the drop-down list in the dialog box.

5. Select the "Open device view" check box if you want to change to the device view immediately after adding the device.
   There you can immediately continue with device configuration and equipping the rack.
6. Click on "OK" to add the device selected.
   The dialog box closes.

Adding device from the hardware catalog

To add a device to the hardware configuration using the hardware catalog, proceed as follows:

1. Open the network view or the topology view.
2. Open the hardware catalog.
3. Go to the required device in the hardware catalog.
4. Click on the chosen device to select it.
5. If necessary, set the firmware version using the drop-down list in the hardware catalog.
6. Drag the device to the network view or the topology view.



You have now placed the device in the network view or in the topology view. The displayed rectangle (in other words "Station") symbolizes the plugged device together with its rack and any lower-level modules. Double-click on the device or station to open the device view and view the new rack and inserted device. In the next steps, you can configure the device in the device view and equip the rack with modules.

See also

Network view (Page 527)

Adding an unspecified CPU (Page 557)

Topology view (Page 536)

10.1.2.3 Adding an unspecified CPU

Introduction

If you have not yet selected a CPU but have already started programming or want to use an existing program, you have the option of using an unspecified CPU. You can also adjust some settings with unspecified CPUs. The setting options are restricted to parameters that all CPUs of the same CPU family have in common.

Adding an unspecified CPU in the portal view

To add an unspecified CPU in the portal view, follow these steps:

1. Now, click one of the following options:
   – "Devices & networks > Add new device"
   – "PLC programming" > "Device" button
2. For a device family, select an unspecified CPU from the tree structure of the "Add new device" dialog.
3. Click on "Add".

An unspecified CPU is added and the device view for this CPU is opened.

Further options for adding unspecified CPUs

In the project view, you can add unspecified CPUs like specified CPUs:

- Using the "Add new device" button in the project tree
- In the "Hardware catalog" task card

You can also use these methods to add multiple unspecified CPUs.

Specifying unspecified CPUs

You have two options for specifying unspecified CPUs:

- Use drag-and-drop to assign an existing CPU from the hardware catalog to an unspecified CPU by means of module replacement (Page 572).
- Select a selected, unspecified CPU and then the menu command "Online > Hardware detection" and assign a CPU identified online. For this purpose, you assign an IP address using the "Add address for PG/PC" button.

Note

If you want to go online after the hardware detection, you have to first download the detected configuration to your project; otherwise, an error may occur due to inconsistent configurations. After hardware detection, the article numbers of the CPU in the project and the actually existing CPU are identical, but their parameters are not. The parameters of the CPU in the project have the default values; the parameters of the actually existing CPU have the values set by you.

After module replacement, F-CPUs behave as a CPU without failsafe functionality. In contrast to F-CPUs from the hardware catalog, after module replacement with an unspecified CPU or a hardware recognition, the F capability is turned off and the protection level is set to "Full access (no protection)".

See also

Selecting a CPU (Page 559)

Adding a device to the hardware configuration (Page 556)

10.1.2.4 Selecting a CPU

Introduction

Select a CPU from the hardware catalog and place it, together with a rack, in the network view. On this device drag the desired modules from the hardware catalog; they are arranged automatically on the rack.

Selecting the components in the hardware catalog

Each hardware component is displayed as a folder in the hardware catalog. When you open this folder, you see the different versions of the selected hardware component with its respective article numbers.

There will be an example of how to set up a CPU with a rack in network view.

Requirement

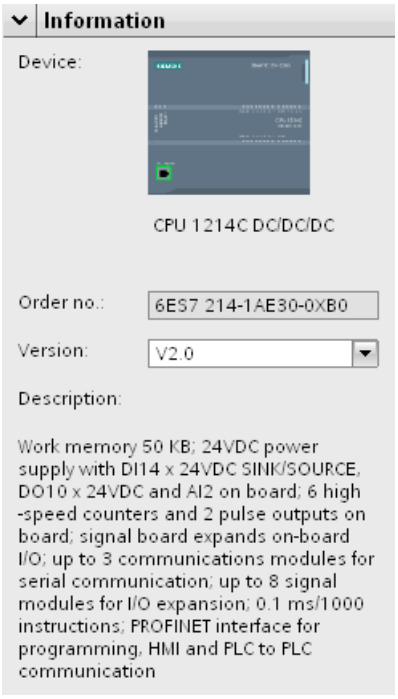
- The hardware catalog is open.
- You must be in the network view.

Procedure

To select a CPU from the hardware catalog, proceed as follows:

1. In the hardware catalog navigate to the folder with the desired CPUs.
2. Open the folder with the desired CPU type; you will see all article numbers for the selected CPU type.

3. Click on a CPU article number to get information about the selected CPU in the "Information" pane.



4. Set up the CPU and a rack. You have the following options:
   - Use drag-and-drop to drag the CPU from the hardware catalog into network view.
   - Use Copy & Paste to copy the CPU to the network view.
   - Double-click the CPU entry in the hardware catalog.

See also

Browsing the hardware catalog (Page 547)

Adding a device to the hardware configuration (Page 556)

Rack: Inserting a module (Page 564)

Rack: Basics (Page 561)

Adding an unspecified CPU (Page 557)

10.1.2.5 Selecting fail-safe modules

Introduction

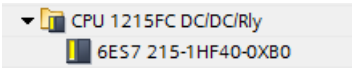
Fail-safe automation systems (F systems) are used to control processes with a safe state that can be achieved immediately on shutdown. F systems control processes that can be shut down immediately without endangering human beings or the environment.

Selecting the F components in the hardware catalog

Each hardware component is displayed as a folder in the hardware catalog. Fail-safe hardware components are displayed with yellow symbols in the hardware catalog in contrast to the blue hardware components that are not fail-safe.

The following figure shows an excerpt from the hardware catalog with an F-CPU (yellow).

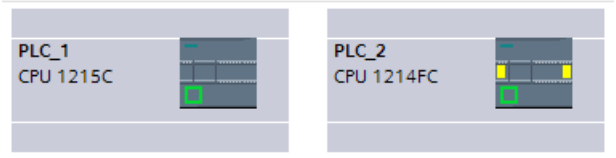
Based on this color coding, you can identify F components in the hardware catalog simply and quickly.



Display of the F components in the hardware and network editor

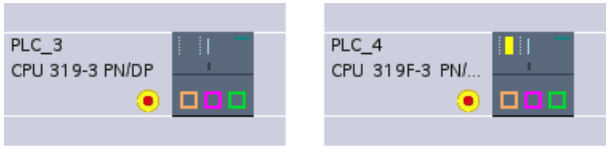
If you use fail-safe devices and "normal" devices in the graphic views of the hardware and network editor, you can recognize them simply by the yellow color-coding of the devices.

The following figure shows two CPUs. The F CPU is shown in yellow:



Fail-safe modules in racks or devices can also be recognized in the device view due to their yellow color coding. Since modules that are not capable of networking are not displayed in the network or topology view, devices with F components without networking capability are represented by a special symbol in the network and topology view.

The following figure shows a "normal" CPU and an F-CPU in the network view that both contain F components:



Based on the color coding of the fail-safe devices and modules, you can also identify the F components quickly in the graphic views of the hardware and network editor.

10.1.2.6 Rack: Basics

Introduction

To assign modules to a device, you need a rack, for example a mounting rail. Secure the modules on the rack and connect these via the backplane bus with the CPU, a power supply or other modules.

Creating a rack

If you insert a device in the network view, a station and a rack suitable for the device selected are created automatically. The rack and slots available are displayed in the device view. The number of slots available again depends on the type of device used.

Rack structure

A rack always contains the device that has been inserted in the network view. The device is permanently assigned a slot which will depend on the type of device in question. There are additional slots on the right of the device and, if necessary, on left of the device; slot numbers are located above slots in which devices are plugged.

A corresponding short description is displayed above the plugged devices and modules. You show or hide this short description via the toolbar under "View" with the command "Display module titles" or the corresponding symbol in the toolbar of the device view (Page 530).

Symbol	Meaning
	Show module titles

When modules are selected in the hardware catalog, all the slots permitted for this module are marked. This allows you to see immediately the slot into which the selected module can be inserted.

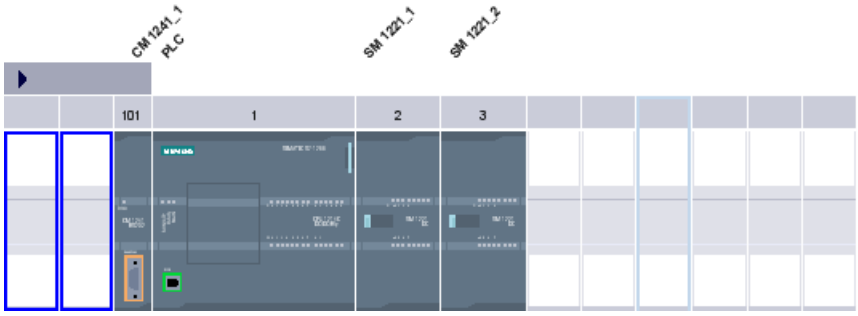
In the following screenshot, a signal module has been selected in the hardware catalog for a partially filled S7-1200 rack:



Since slots 101-103 are reserved for communications modules, only the other free slots are shown as available slots.

You can expand and collapse the front group of slots using an arrow symbol above the expandable slot. When the group of slots is collapsed, the first and last of the group's slot numbers are displayed.

The following figure shows the expanded slot group:



Groups of slots into which modules have already been plugged cannot be collapsed.

Multiple selection of modules and slots

There are various ways of selecting several modules or slots:

- By pressing <Shift> or <Ctrl>, you can select several modules or slots at the same time.
- Click outside the rack and then hold the mouse button and drag a frame to include the modules or slots you want to select.

10.1.2.7 Rack: General slot rules

Introduction

Specific slot rules apply to each automation system and module.

If you select a module from the hardware catalog in the device view, all possible slots for the module selected are marked in the rack. You can only drag modules to marked slots.

If you insert, move or swap a module, the slot rules are also applied.

Consistency

Some slot rules depend on how the environment is configured. This means that you can sometimes plug modules into the rack although this would result in inconsistencies at the current time. If you change the configuration, for example by selecting different modules or module parameter settings, you can make the configuration consistent again.

In cases where inserting a module results in an inconsistency that can be corrected, this will be permitted. A consistency check is run when the configuration is compiled. Inconsistencies are displayed as alarms in the Inspector window under "Info". Based on the result of the consistency check, you can revise your configuration and make it consistent again.

Rules for arranging modules

The following rules apply generally to modules in racks:

- You can only plug modules into a rack.
- You can only plug interface modules into a module.
- You can only use modules of the same product or system family in one rack.

There are also other special rules for some modules:

- Can only be inserted in certain slots
- Insertion depends on other modules, CPUs or settings
- Limited number of uses in a rack

10.1.2.8 Rack: Inserting a module

Introduction

Once you have added devices from the hardware catalog to your configuration in network view, you can add modules to the devices. There are various ways of adding a module to a rack in the device view:

- If there is an available valid slot, double-click a module in the hardware catalog.
- Use drag-and-drop to move the module from the hardware catalog to an available valid slot in the graphic or table area:
  - Text entry from the "Catalog" pane
  - Preview graphic from the "Information" pane
- Select "Copy" in the shortcut menu for a module in the hardware catalog and then select "Paste" in the shortcut menu on an available valid slot in the graphic or table area.

To access the device view from the network view, double-click a device or station in the network view or select the Device view tab. The device view contains an illustration of the device selected within a rack. The graphic illustration of the rack in the software corresponds to the real structure, i.e. you can see the same number of slots as exist in the real structure.

Note

You can also move a module to a rack in the network view. The filter function for the hardware catalog must be deactivated in this instance. The module is automatically plugged into a free and permitted slot. If there are no slots available, the module will be moved to the area of unplugged modules (Page 535).

Equipping a rack

Arrange the modules on a rack according to the applicable slot rules.

After a module has been inserted in a rack with an already inserted CPU, the address areas are checked automatically so that addresses are not assigned twice. After it has been inserted,

each module then has one valid address range. To do so, DP slaves and IO devices must be networked with a CPU via the corresponding DP master or IO system.

Requirements

- You are in the device view.
- The hardware catalog is open.

Adding module from the hardware catalog

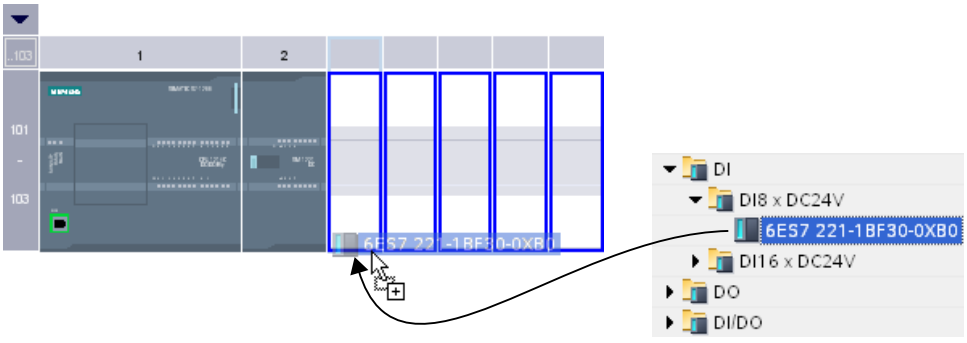
How to insert a module from the hardware catalog into a rack is illustrated based on the example of a signal module. To do this, follow these steps:

1. Go to the required module board in the hardware catalog.

   Note

   If you activate the filter function of the hardware catalog, only those modules which match the selected device type will be displayed.

2. Select the chosen module.
3. If necessary, set the firmware version using the drop-down list in the hardware catalog.
4. Drag the signal module to a free slot in the rack.

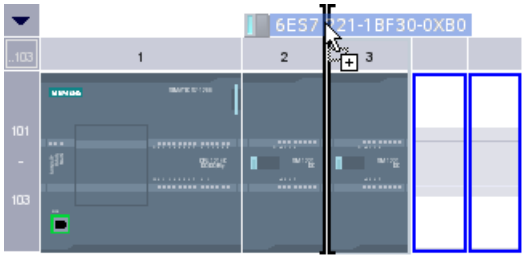


You have now inserted the digital signal module in a slot in the rack. Repeat these steps with the other modules.

The name of the module is displayed above the inserted modules. You can activate or deactivate module labeling in the menu bar with "View > Show module labels".

Inserting module

You can also drag modules and drop them between modules that have already been inserted. To do this, drag a module above and between the two existing modules while holding down the mouse button.



A mouse pointer appears. When you release the mouse button, all modules plugged to the right of the pointer are moved one slot to the right. Any redundant modules are moved to the area of unplugged modules. The new module is plugged at the point of the freed up slot.

See also

Device view (Page 530)

Device view: Area for unplugged modules (Page 535)

Rack: General slot rules (Page 563)

10.1.2.9 Select hardware component

You can select hardware components and additional objects of the graphic views by clicking the left mouse button. You can also select multiple objects by dragging the mouse while keeping the left mouse button pressed or with the combination of left-click and the <Shift> or <Ctrl> keys.

You can select the following objects in the graphic views: Devices (with CPU, rack, modules), CPUs, CPs, modules, subnets, connections and nodes.

Rules

You can select objects in the graphic views as follows:

- Left-click on object: The object is selected.
- Left-click and <Ctrl> on an object: The object is added to the current selection.
- Left-click and <Shift> on an object: All objects between the currently selected object and the previously selected objects are added to the selection.
- Drag selection frame around multiple objects while pressing left mouse button: All objects completely enclosed by the frame are added to the selection.

Multiple selection is only possible for objects of the same category:

- Category 1: Devices, subnets, connections, port interconnections
- Category 2: CPUs, CPs, interface modules / module
- Category 3: Interfaces, ports, nodes

Behavior for multiple selection with objects from different category:

- Selection frame: Only objects of the respective highest category will become part of the multiple selection
- Left-click with <Shift> or <Ctrl>: The object of a new category that was selected last remains selected; the previously selected objects are dropped from the selection.

Note

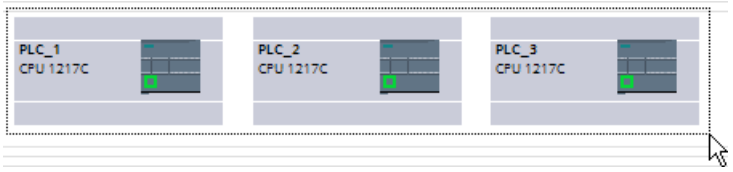
Multiple selection of different types of objects

Only the same object categories can be selected in case of multiple selection. For example, multiple CPUs or multiple networks can be selected. However, it is not possible to first select a device in a multiple selection and then only add the CPU of a second device or the interface of a third device to the multiple selection.

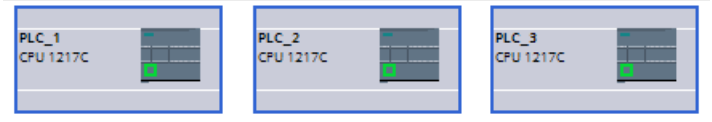
Multiple selection of devices with the help of the selection frame

To select multiple devices in the network view by dragging a selection frame, follow these steps:

1. Press the left mouse button and draw a frame around all devices that you want to include in your selection:



2. Release the mouse button. All devices within the drawn frame are now part of the multiple selection.



In the example above, the selection frame was drawn around entire devices. If you draw the frame so that it does not enclose entire devices but only the CPUs contained therein, only the CPUs and not the entire device is included in the multiple selection. If you draw the selection frame so that it partially encloses entire devices and partly only CPUs, the highest category of the selected objects is decisive and only the entire devices are added to the multiple selection.

Multiple selection of CPUs with the help of the <Shift> key

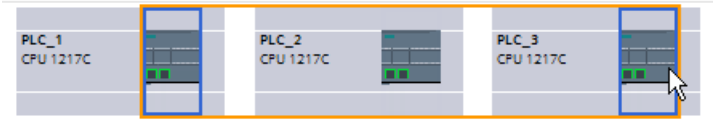
Instead of the multiple selection with a selection frame, you can also make the multiple selection with the <Shift> key. You select two objects with this key and all objects in between are included in the multiple selection.

To select multiple CPUs in the topology view with the <Shift> key, follow these steps:

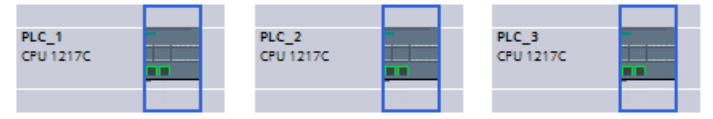
1. Select the first CPU of your multiple selection with the left mouse button:



2. Select the last CPU of your multiple selection with the left mouse button while pressing the <Shift> key. The two selected CPUs behave like the margins of a selection frame indicated here by the orange line:



3. All CPUs between the first and the last CPU selected are included in the multiple selection:



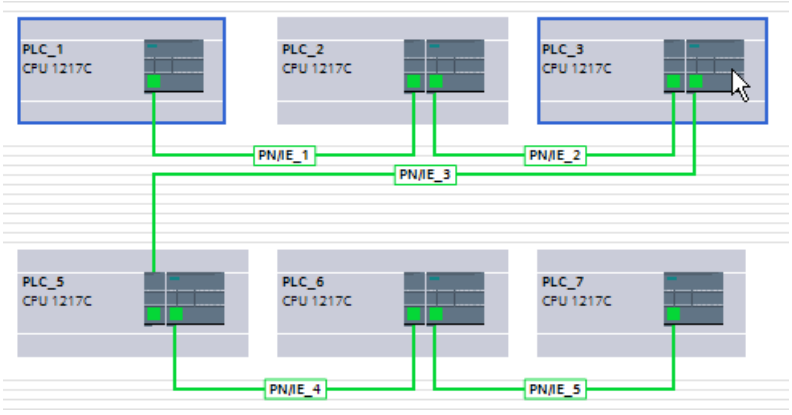
In the example above, only CPUs were selected. If you select entire devices instead, the devices between the first and the second device selected are included in the multiple selection. If you select an entire device as well as a CPU, the category of the object selected last is decisive for the category of all selected objects.

Multiple selection of devices with the help of the <Ctrl> and <Shift> keys

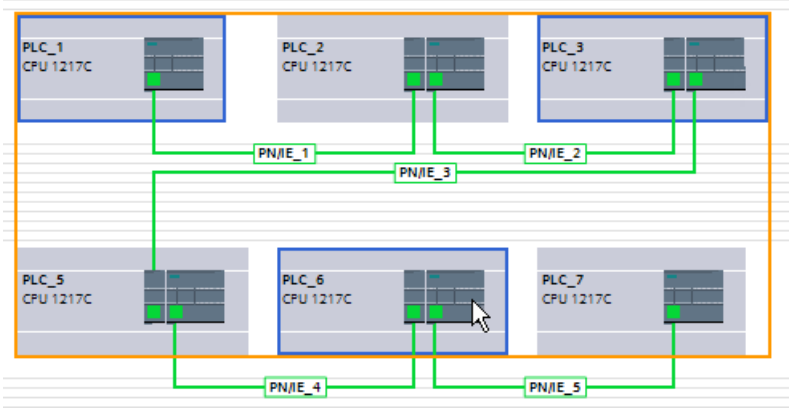
You can also make a multiple selection with the two keys <Ctrl> and <Shift>. You select multiple objects with a left mouse click and the <Ctrl> key. If you then select an object with a left mouse click and the <Shift> key, all objects between the previously selected objects and the last selected object are included in the multiple selection.

To select multiple devices in the device view with the <Shift> and <Ctrl> keys, follow these steps:

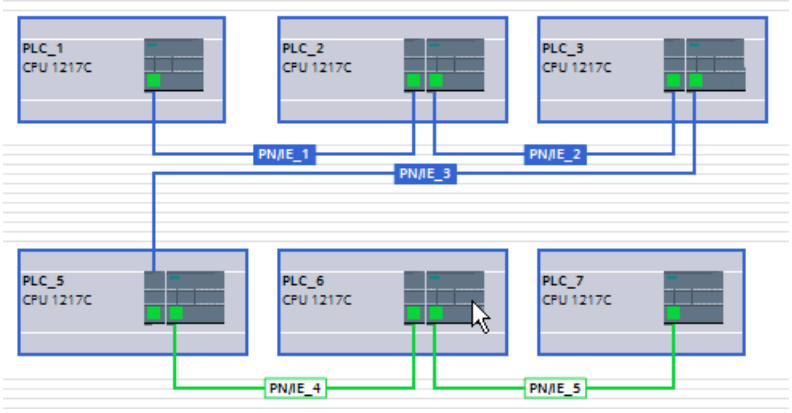
1. Use the left mouse button and press the <Ctrl> key to select multiple individual devices:



2. Select the last device of your multiple selection with the left mouse button while pressing the <Shift> key. The already selected devices behave like the margins of a selection frame indicated here by the orange line:



3. All devices of the virtual selection frame and the subnets in the selection frame are now included in the multiple selection.



Only the same object categories are selected. In the example above this includes the subnets in addition to the corresponding devices because they were located within the virtual selection

frame and belong to the same category as the devices. The subnets PN/IE_4 and PN/IE_5 were not located in the virtual selection frame and therefore are not part of the multiple selection.

10.1.2.10 Copying a hardware component

You can copy hardware components in the device or network view. Copied hardware components are stored on a clipboard and can be pasted at another point from this clipboard. Copied stations are pasted as new stations in the network view. Copied devices and modules can be pasted into existing racks in the network and device view.

Rules

- Single objects as well as several objects can be copied at the same time.
- Modules inserted in the rack and in the area of unplugged modules can be copied.
- You can only copy devices and modules to free and valid slots in keeping with the slot rules.
- Racks with a CPU inserted cannot be copied individually, but only as complete units along with all inserted hardware components.

Procedure

Proceed as follows to copy a hardware component:

1. Select the hardware components you want to copy.
   - Device view: Select the module in a rack or put it in the area of unplugged modules.
   - Network or topology view: Select the station or the relevant hardware component from the network view.
   - Project tree: Select the station or module.
2. Select "Copy" from the shortcut menu or press <Ctrl+C>.
   If the "Copy" menu item is unavailable, your selection contains at least one component that cannot be copied.
3. Select the location at which the content of the clipboard is to be pasted.
   - In the device view select a free slot in the rack or area of unplugged modules.
   - In the network or topology view select a station where you want to insert devices or modules.
4. Select "Paste" from the shortcut menu or press <Ctrl+V>.
   - If you paste the copied object without selection in the network or topology view with <Ctrl +V>, the copied object is pasted to the right of the original object.
   - If you paste the copied object without selection in the network or topology view using the shortcut menu, it is placed at the location of the mouse pointer.

   If the "Paste" menu item is unavailable, the clipboard is empty or contains at least one component that cannot be pasted at this point.

The copied object is pasted at the chosen point.

If you have selected a station in the network or topology view to paste a module, the module is inserted into the first free and valid slot. If no free, valid slots are available, the object is inserted in the area of unplugged modules.

Note

You can also copy a module from one device to another:

To do so, copy a module in the hardware and network editor, select a different device in the network or topology view or the drop down list of the device view and insert the module.

You can insert the copied object directly in a slot or place it in the area of unplugged modules in the device view. If you add the copied object in the network view of a device or a station, it will be inserted in the first available slot.

If there is no slot available for the object, it is automatically placed in the area of unplugged modules (Page 535).

Note

You can use <Ctrl> and drag-and-drop to directly copy a selected hardware component.

See also

Keyboard operation: Editing objects (Page 554)

10.1.2.11 Moving a hardware component

You can move hardware components in the device or network view.

Rules

- You can move devices and modules from the rack and the area for unplugged modules taking the slot rules into consideration.
- CPs can be moved in the network view. The CP is plugged in a free and valid slot in the target device. If there are no free slots available, the CP to be inserted is moved to the area for unplugged modules.
- In the network view, CPU and slave head modules can be moved between the devices; depending on CPU type also within the rack.

Note

Moved CPs are disconnected from their network but keep their network parameters and address. If you reconnect the CP to the network and its address has been assigned, use a dialog to assign a new unique address to the CP.

Procedure

Proceed as follows to move a hardware component:

1. Select the hardware component you want to move.
   - Device view: Select the module in a rack or put it in the area of unplugged modules.
   - Network view: Select the hardware component of relevance to the network view.
2. Select "Cut" from the shortcut menu or press <Ctrl+X>.
   If the "Cut" menu item is unavailable, your selection contains at least one component that cannot be cut.
3. Select the location to which the cut object is to be moved.
   - Device view: Select a free slot in the rack or area of unplugged modules.
   - Network view: Select a station where you want to insert devices or modules.
4. Select "Paste" from the shortcut menu or press <Ctrl+V>.
   If the "Paste" menu item is unavailable, the clipboard is empty or contains at least one component that cannot be pasted at this point.

The selected hardware component is moved to the target. If the hardware component being moved is a networked object, it is disconnected from the network.

Note

You can use drag-and-drop to directly move a selected hardware component.

See also

Keyboard operation: Editing objects (Page 554)

10.1.2.12 Replacing a hardware component

You can replace hardware components with others. This, for example, allows you to replace unspecified CPUs (Page 557) with available CPUs from the hardware catalog.

Rules

You can only replace hardware components if they support module replacement and if the two components are compatible.

Procedure

To replace one module with another, proceed as follows:

1. Select the module you want to replace.
2. Open the shortcut menu:
   - If the "Replace device" entry is enabled, the module can be replaced.
   - If the "Replace device" entry is disabled, a module cannot be replaced.

3. Click on "Replace device" in the shortcut menu. The "Replace device" dialog box appears.
4. Under "New device" in the tree structure, select the module with which you want to replace your current module.
5. Click "OK".
6. Compile the hardware and software.

The existing module is replaced by the new one.

As an alternative, you can take a module by dragging it from the hardware catalog to the module you are replacing. If the module can be replaced by the selected module, this is indicated by the mouse pointer symbol.

10.1.2.13 Renaming hardware component

You can conveniently edit the names of hardware components in all three views (network view, device view, topology view) directly in the graphic area as well as in the corresponding table areas or in the Inspector window.

Rules

You can rename the hardware components as follows:

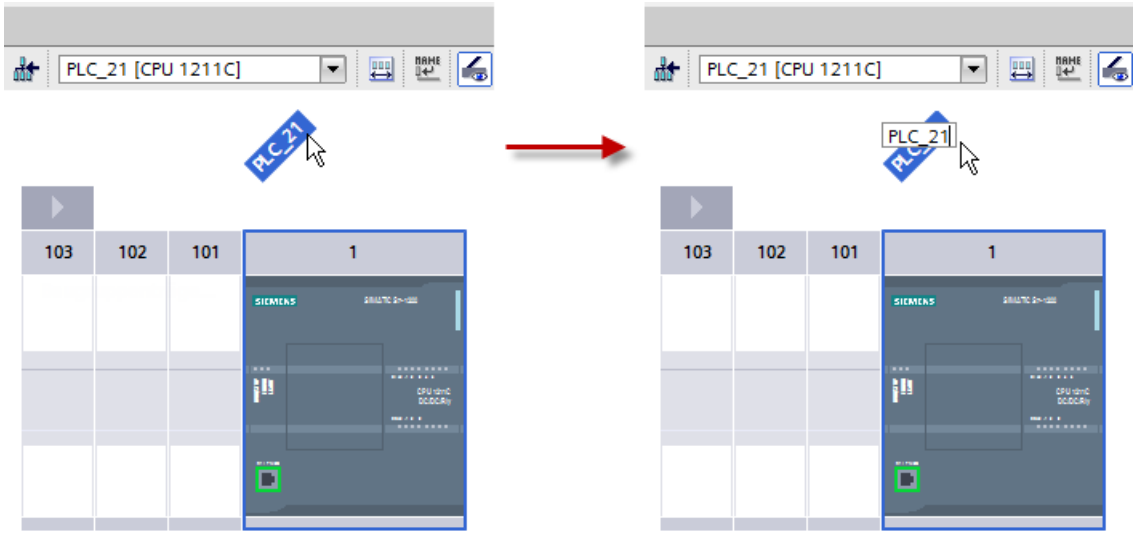
- In the graphic view after selection of the displayed name.
- In the table view in the table cell with the name that is to be changed.
- In the Inspector window under "Properties > General" for the selected hardware component.

Procedure

To rename a hardware component in the graphic area of a view, follow these steps:

For the device view:

1. Activate the display of the module names with the corresponding button in the toolbar.
2. Select the hardware component or its name.
3. You have two options:
   – Press <F2>.
   – Click on the previously selected name.
4. Edit the name.



For the network and topology view:

1. Select the hardware component or its name.
2. You have three options:
   - Press <F2>.
   - Click on the previously selected name.
   - Select "Rename" from the shortcut menu.



For editing names in the table area, select the cell with the name you want to edit in the table. Then click on the name once again so that you can edit the names directly in the cell.

For editing names in the Inspector window, select the hardware component whose name you want to edit in the graphic area. The relevant parameters are then displayed in the Inspector window. Now you can edit the names under "Properties > General".

See also

Network view (Page 527)

Device view (Page 530)

Topology view (Page 536)

Inspector window (Page 539)

10.1.2.14 Deleting a hardware component

There are various ways of deleting hardware components. Deleted hardware components are removed from the system and assigned addresses made available again.

Rules

- CPUs or modules from the rack and from the area of unplugged modules can be deleted.
- When a rack is deleted in the device view, the plugged hardware components are moved to the area of unplugged modules.

Procedure

Proceed as follows to delete a hardware component:

1. Select the hardware components you want to delete.
   - Network view: Select devices or network relevant hardware components in the graphic view or in the network view.
   - Device view: In the graphic view or device overview, select racks or modules in racks or in the area of unplugged components.
   - Topology view: Select devices or hardware components with Ethernet interfaces in the graphic view or in the topology view.
   - Project tree: Select devices or individual hardware components from the tree structure.
2. Select "Delete" from the shortcut menu or press <Del>.
   If the "Delete" menu item is unavailable, your selection contains at least one component that cannot be deleted.

The selected hardware components are deleted.

Note

Deleting hardware components may result in inconsistencies in the project, such as violation of slot rules. Inconsistencies are reported during the consistency check. Correct the inconsistencies by taking appropriate action, for example, ensuring compliance with the slot rules.

See also

Keyboard operation: Editing objects (Page 554)

10.1.2.15 Editing properties and parameters

Once you have inserted hardware components in your rack, you can edit their default properties, such as names, parameters or addresses, in the network view, device view or topology view.

Requirement

You are in the network view, device view or topology view.

Note

You can edit the relevant properties and parameters in the different views in the graphic view as well as in the table view. It depends on the view which parameters you can edit. In the graphic network view, you have access to the network-related hardware components and the station. You can access modules and hardware components not displayed in the graphic network view using the table network view. In the topology view and in the device view, you have access to additional parameters regarding the topology or device access.

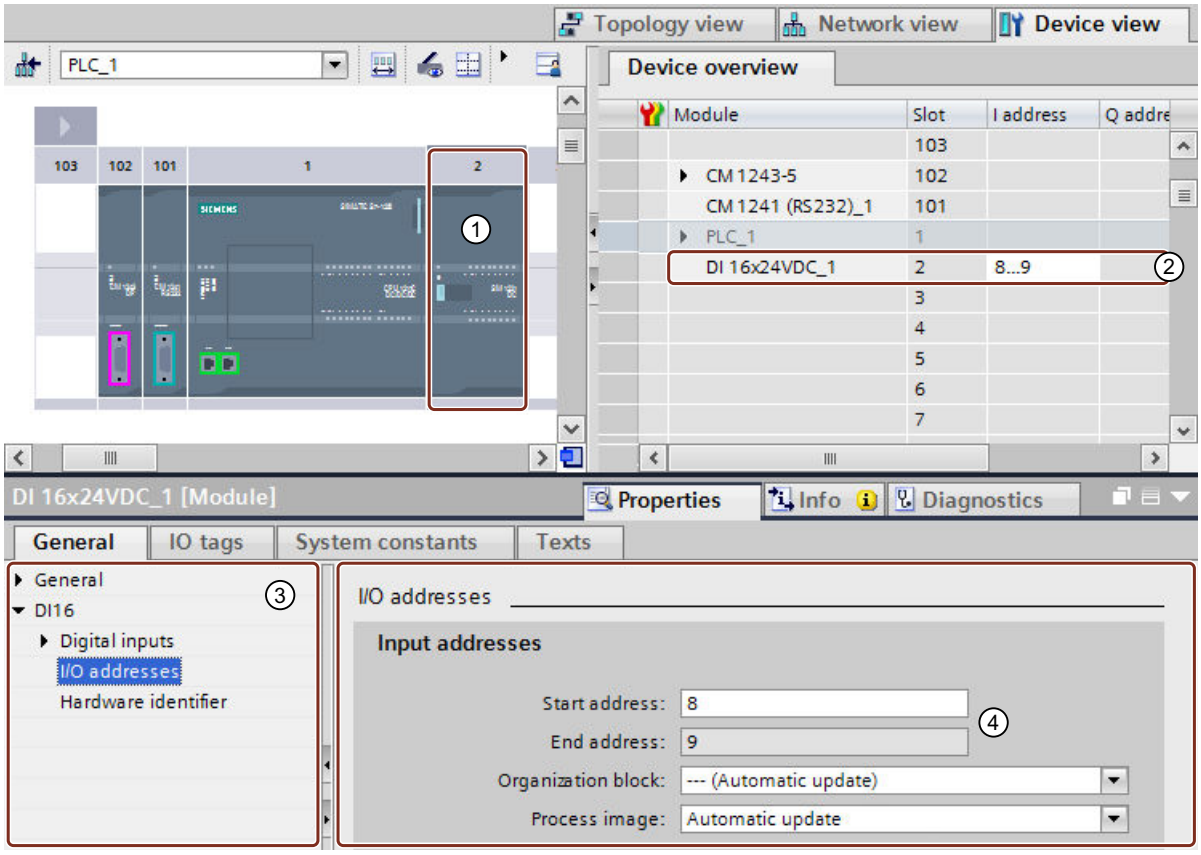
Procedure

To change the properties and parameters of the hardware components in the device view, follow these steps:

1. In the graphic view, select the CPU, module, rack or interface you want to edit.
2. Edit the settings for the selected object:
   - In the graphic area you can, for example, change the object names with activated module labeling.
   - Use the table area to edit addresses and names, for example.
   - In the Inspector window additional setting possibilities are available in "Properties".

Note that modules can only be fully parameterized if they are assigned to a CPU. Therefore, PROFIBUS or PROFINET interfaces modules must first be networked with the CPU or a centrally inserted communication module so that they form a master system or IO system. Only then is it possible, for example, to edit the addresses of the distributed components that are inserted.

Example of changing settings



① Selection of a module
② Editing option for addresses in the device overview
③ Selection options in the inspector window
④ Editing option for addresses in the inspector window

See also

Inspector window (Page 539)

10.1.2.16 Input and output addresses in the address overview

Introduction

The currently used input and output addresses can be displayed in the address overview in a table form. The address overview can be found in the Inspector window under "Properties" of the CPU.

Design of the address overview

With the different check boxes, you can set which objects should be displayed in the address overview:

- Inputs: Display of the input addresses
- Outputs: Display of the output addresses
- Address gaps: Display of open address spaces
- Slot: Display of the slot number

The following information is typically shown in the address overview:

Table header	Meaning
Type	Type of the address range. Indicates whether the area is an input address area or an output address area. • I = input address of module with input I/O address space (with length > 0) • I* = input or diagnostics address of modules without input I/O address space (with length = 0) No data (signal statuses or measured values) is exchanged via these addresses in the user program. Used for reporting diagnostics events of the operating system (e.g. in the start information of error OBs). • Q = output address
Addr. from	Start address of the address range. Indicates the first assigned address of the input or output.
Addr. to	End address in the address range. Indicates the last assigned address of the input or output.
Size	Address length of the address range occupied by the module for the input or output. Examples: • With a start address of 0 and an end address of 1, the address length is 2 bytes. • With a start address of 0 and an end address of 3, the address length is 4 bytes. • With a start address of 0.1 and an end address of 0.2, the address length is 2 bits. • If the start address and end address are identical, 0 bytes is displayed.
Module	Module using the address area.
Rack	Number of the rack on which the hardware component is inserted.
Slot	Number of the slot in the rack in which the hardware component is inserted.
Device name	Name and type of device in which the module is inserted. Examples: • The module is inserted in a PLC with the name "PLC_1" of the type CPU 1516-3 PN/DP, "PLC_1 [CPU 1516-3]" is displayed. • The module is inserted in a head module with the name "IO Device_1" of the type IM 151-3 PN; "IO Device_1 [IM 151-3 PN]" is displayed

Table header	Meaning
Device number	PROFINET device number or PROFIBUS address if the module is connected to a PLC via PROFINET or PROFIBUS. If the module is inserted directly in a PLC, "-" is displayed. Examples: • The module is inserted in a head module that is connected to a PLC via a PROFINET interface; the device number of the Ethernet address is displayed. • The module is inserted in a DP slave that is connected to a PLC via a PROFIBUS interface, the address parameter of the PROFIBUS address is displayed.
Master I/O system	Name of the DP master system or IO system via which the DP slave or IO device is connected to the PLC. The corresponding number of the DP master system or IO system is displayed in square brackets. If the module is inserted directly in a PLC, "-" is displayed. Examples: • The module is inserted in an IO device and is connected to a PLC via a PROFINET IO system with the name "PROFINET IO-System_1" and the number 100. "PROFINET IO-System_1 [100]" is displayed. • The module is inserted in a DP slave and is connected to a PLC via a PROFIBUS DP master system with the name "DP-Mastersystem" and the number 1. "DP-Mastersystem [1]" is displayed.
OB	Organization block that is assigned to the process image. The "Main" cycle OB always refers to the automatic update of the process image. This column is not available for every CPU.
PIP	Process image partition. Indicates whether the address is updated automatically via the cyclic process image or is located in a process image partition that must be triggered explicitly for an update. Refer to the further description of the "Process image partition PIP2" below.

Process image partition PIP

The "PIP" table column shows the assignment of the address to the cyclic process image or to a process image partition (PIP).

For S7-300/400:

- "OB1-PA": The address is assigned to the cyclic process image. The operating system updates this address automatically in each program cycle.
- "PIP x": The address is assigned to the process image partition x (for example PIP 1, no cyclic process image). The operating system updates this PIP when the assigned OB is executed. If this PIP is not assigned to an OB, the operating system does not update this PIP. You have the option to update the PIP yourself with the instructions "UPDAT_PI" and "UPDAT_PO" (for S7-400 and some S7-300 CPUs).

For S7-1200:

- "Automatic update": The address is assigned to the cyclic process image (PIP 0). The operating system updates this address automatically in each program cycle.
- "None": The address not assigned to any process image partition. You access this address directly in the user program (direct I/O access, no process image).

- "PIP x": The operating system updates this PIP when the assigned OB is executed. If this PIP is not assigned to an OB, the operating system does not update this PIP. You have the option to read inputs or write outputs in the user program via direct I/O access. The instructions "UPDAT_PI", "UPDAT_PO", "SYNC_PI" and "SYNC_PO" are not supported for S7-1200.
- "PIP OB servo": The process image partition "PIP OB Servo" is not assigned to an organization block (fixed setting for organization block: "---(None)"). The operating system does not update this PIP or any of the addresses contained in it: You access the addresses directly in the user program (direct I/O access).

For S7-1500:

- "Automatic update": The address is assigned to the cyclic process image (PIP 0). The operating system updates this address automatically in each program cycle.
- "None": The address not assigned to any process image partition. You access this address directly in the user program (direct I/O access, no process image).
- "PIP x" (PIP 1 to PIP 31): The operating system updates PIP x when the assigned OB is executed. If this PIP is not assigned to an OB, the operating system does not update PIP x. You have the option to update the PIP x in the user program with the instructions "UPDAT_PI" and "UPDAT_PO". If PIP x is assigned to an isochronous mode interrupt OB (OB 61 to OB 64), the operating system does not update PIP x. You have the option to update PIP x in your user program with the instructions "SYNC_PI" and "SYNC_PO".
- "PIP OB servo": The process image partition "PIP OB servo" is permanently assigned to the OB "MC-Servo". STEP 7 generates this OB automatically when you create a technology object in the Motion Control area. When the OB is executed, the PIP OB Servo is updated isochronously. All drives and encoders used by Motion Control are assigned to this process image partition.

See also

Specifying input and output addresses (Page 1046)

10.1.2.17 Update module version

Explanation of terms

The terms "Module version" and "Firmware version" are explained in more detail in the following section.

- Module version: The specific version of the configuration software from which the module description stems.
  Example: V11.0.0.0
- Firmware version: The version of the firmware of the module whose parameters are assigned offline
  Example: V2.0

Requirement

- You have created a device configuration.
- You have installed an update or an optional package at a later date. As a result of this installation, the module version of at least one module type was updated in the hardware catalog, whereby the new version is incompatible with the previous version.
- You have used such modules in your device configuration and want to use the modified or added properties.

Procedure

Perform the following step for each affected module type.

1. Select the affected module in the device view.
2. In the Inspector window, go to "Properties > General > Catalog Information". Click the "Update module version" button there.
3. In the query that then appears, specify whether you want to update the module version only for the selected module or for all modules of this type in the current project.

Result

The selected modules are replaced by the same modules with updated module version in the current project.

In which cases is it unnecessary to update the module version?

An updating of the module version is not necessary in the following cases:

- You do not want to use the modified or added properties of the modules.
- When you open an existing project with a version of the configuration software that is more recent than the version with which you created the project, the system automatically performs a project conversion, for example, from TIA Portal V12 to V13. In this case, all older module versions are automatically adapted.

10.1.2.18 Distributed I/O in the project tree

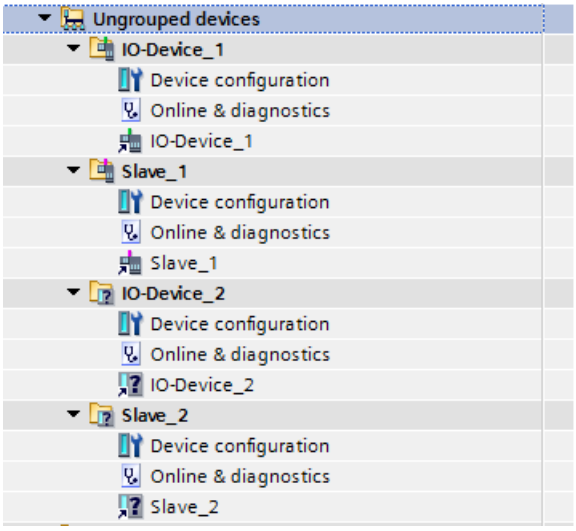
Distributed I/O devices in the project tree shown in separate folders. Depending on whether a device was assigned to a PROFINET IO system, a DP master system or no system, the device is displayed as a node or link to the device in one of the folders described below.

"Ungrouped devices" folder in the project

All distributed I/O devices in the project are collected in the "Ungrouped devices" folder.

The icons of the folders and links show the status "assigned" with colored identifiers: PROFINET IO devices are marked green, PROFIBUS DP slaves are marked magenta, other distributed I/O devices are not marked.

The icons of the folders and links show the status "not assigned" with question marks.



If you have established a connection to an IO device or a DP slave with the "Go online" function, the same diagnostics information (e.g. green check mark) is displayed for the link of a device as for the device itself in the project navigation. For more information, refer to the section "View in the online mode".

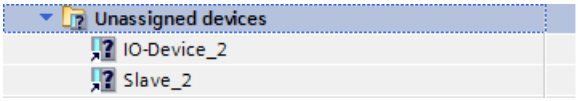
It is possible to copy or move distributed I/O devices from the "Ungrouped devices" folder, but not the folder itself, to another position in the project tree. Possible destinations are the project folder or device folder. You can also copy or move distributed I/O devices back to the "Ungrouped devices" folder.

By double-clicking on the link, you go to the device view and can configure the distributed I/O device.

"Unassigned devices" folder in the project

Devices that are not assigned to a distributed I/O system appear as links in the "Unassigned devices" folder.

The icon for the folder and the link of the respective device in the project tree indicate with a question mark that they are not connected to a distributed I/O system. The link is marked by a small arrow.



The display of the project tree as a link allows IO devices or DP slaves to be selected individually, for example to select the destination of the link. It remains possible to select device groups.

By double-clicking on the link, you go to the device view and can configure the distributed I/O device.

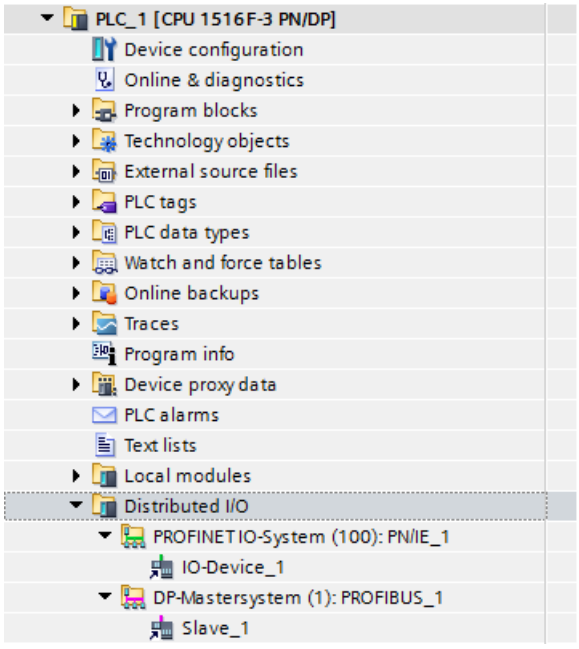
"Distributed I/O" folder under a device

If you assign an IO device to a PROFINET IO system, this is displayed as a link below the node "Distributed I/O > PROFINET IO system".

Accordingly, a DP slave that is assigned to a DP master system is shown as a link below the node "Distributed I/O > DP master system".

The link references the corresponding IO device or the DP slave in the project tree.

The icon for the node and the link of the respective device in the project tree indicate whether the device is connected via PROFINET (green) or via PROFIBUS (magenta). The link is marked by a small arrow.



Gateways are shown as a link below the DP master/IO node to which they were assigned.

By double-clicking on the link, you go to the device view and can configure the distributed I/O device.

10.1.2.19 Comparing devices

Basics of device comparison

Function

You have the option of comparing the hardware components of two devices allowing you to identify any differences. You can perform an offline/offline comparison for this purpose. The devices to be compared can be from one project or different projects.

You can compare the central as well as the distributed I/O. The devices to be compared can be assigned either automatically or manually. The automatic assignment of central I/O is based

on the slot number. With the distributed I/O, the automatic assignment can be made according to the following criteria:

- Assignment using the address/HW ID: The assignment is made using the addresses or IDs of the devices. This criterion is suitable for comparing devices within a project.
- Assignment using the name: The assignment is made based on the device names. This criterion is suitable for comparing devices in different projects.

You can either specify the assignment yourself or let the system decide. In the latter case, the system selects the assignment itself depending on the context.

To compare the modules of the distributed I/O, start an offline/offline comparison for the affected distributed stations.

See also

Basics of project data comparison (Page 358)

Overview of the compare editor (Page 362)

Making a device comparison (Page 584)

Making a device comparison

Procedure

Follow the steps below to compare devices:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The comparison editor opens and the selected device is displayed in the left area.
3. Open the "Hardware" tab.
4. Drag an additional device to the drop area in the right-hand pane.
   All existing objects of the selected devices are displayed depending on the settings of the comparison editor in the "Hardware" tab and an automatic comparison is made. You can identify the status of the comparison objects based on the symbols in the compare editor. For submodules, you can find detailed results in the "Property comparison" area.
5. If you want to change the matching criterion, click on the arrow of the "Available matching criteria" button in the toolbar. Then, select the matching criterion you want to use.
6. If you want to make a manual comparison, click the button for switching between automatic and manual comparison above the status area. Then select the objects you want to compare.
   The properties comparison is displayed. You can see the status of the objects based on the symbols.

See also

Basics of device comparison (Page 583)

Overview of the compare editor (Page 362)

10.1.3 Configure networks

10.1.3.1 Networking devices

Communication and networks

Communication between devices

The basis of all types of communication is always a previously configured network. The network configuration provides the necessary requirements for communication:

- All the devices in a network are provided with a unique address
- Communication of the devices with consistent transmission properties

Definition

Via the network, the various computers and end devices (PC, PG, PLC, AS) are connected together with interface modules, physical cables and suitable software. Networked devices can exchange data with each other. Networking can be implemented via wiring or a wireless network, e.g. via WLAN.

The subnet is part of a network whose parameters must be synchronized with the devices (e.g., with PROFIBUS). It includes the bus components and all connected devices. Subnets can also be linked by gateways to form a network. In practice, the terms "subnet" and "networked devices" are often used synonymously.

Network configuration

The following steps are necessary when configuring networks:

- Connect devices to subnet
- Specify the properties/parameters for each subnet
- Specify the device properties for every networked module
- Download configuration data to the devices to supply interfaces with the settings resulting from the network configuration
- Document the network configuration

For Open User Communication, creating and configuring a subnet is supported by the assignment of connection parameters.

Relation between network configuration and project

Within a project, subnets and their properties are managed. Properties result mainly from adjustable network parameters and the quantity and communication properties of the connected devices.

The devices to be networked must be in the same project.

Subnet name and subnet ID

Within the project, subnets are clearly identified by a subnet name and ID. The subnet ID is saved in all components along with interconnectable interfaces. Components can then be clearly assigned to a subnet even after uploading into a project.

Networking options

In the project, you can create and network devices with components capable of communication. The following basic options are available for networking the devices:

- You link the interfaces of the components capable of communication with one another. A new subnet is created suitable for the type of interface.
- You connect the interface of the devices capable of communication with a new or existing subnet.
- You create an Open User Communication connection. When you assign parameters to the connection for Open User Communication, a subnet is created automatically between the communication partners.
- You use the graphic connection configuration to configure connections; missing networks are hereby recognized and are created either automatically or via dialog.

Due to the different tasks of the devices or the span of the plant, you may need to use several subnets. These subnets are managed in a project.

Networking devices in the network view

Options

In the graphic network view, you have an overview of the subnets of the entire system in the project. You can use the tabular network overview for additional support.

Depending on the starting situation, there are various ways of undertaking configuration to network the interface for a component capable of communication. The procedures are described in the following section:

- Creating an individual subnet
- Creating several subnets at one time
- Connecting two target devices via a new subnet
- Connecting devices to existing subnet
- Selecting an existing subnet from a list
- Automatic networking during the configuration of the connection;
  See also: AUTOHOTSPOT

Possible starting situations are:

- A suitable subnet is not yet available.
- The subnet with which you want to connect the component already exists.

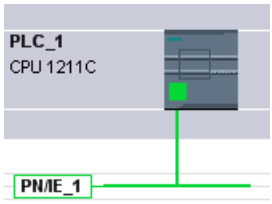
Procedure - creating a single subnet

To create a subnet and to connect it to an interface, proceed as follows:

1. Select the interface of a CPU / a CP.
2. Select the "Create subnet" command in the shortcut menu of the interface.

The selected interface is connected to a new subnet. The inconsistencies possibly resulting from the modified configuration are clarified automatically or per query. In the case of consistency problems, the next free address is preferably used.

The following schematic shows an interface with outgoing line connecting to a subnet:



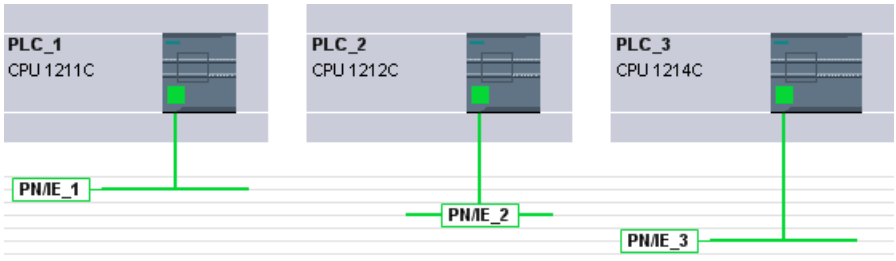
Procedure - creating several subnets at one time

To create several subnets at one time, proceed as follows:

1. Select several interfaces by clicking on them while pressing the <Ctrl> button.
2. Select the "Create subnet" command in the shortcut menu of the interface.

Each selected interface is connected to a new subnet. The inconsistencies possibly resulting from the modified configuration are clarified automatically or per query. In the case of consistency problems, the next free address is preferably used.

The following figure shows multiple subnets created by selecting multiple interfaces:

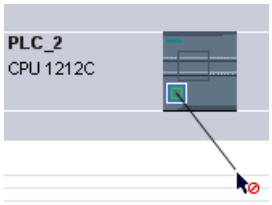


Procedure – Connecting two target devices via a new subnet

To connect an interface with another device via a subnet that does not yet exist, proceed as follows:

1. Position the mouse pointer over the interface for a component capable of communication requiring networking.
2. Click with the left mouse button and hold the button down.

3. Move the mouse pointer.
   The pointer now uses the networking symbol to indicate "Networking" mode. At the same time, the mouse pointer shows the lock symbol that will only disappear when the pointer is on a valid target.



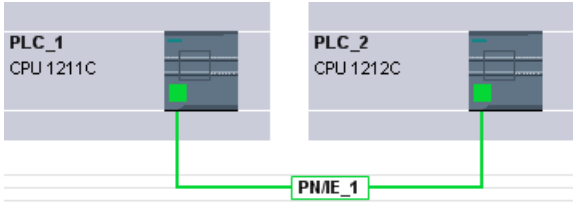
4. Now move the pointer in networking mode onto the interface of the target device. You can either keep the mouse button pressed or release it.
5. Now release the left mouse button or press it again (depending on your previous action).

Note

If you want to exit networking mode beforehand, press <Esc>, right-click or double-click in the background of the network view.

A new subnet is created. The interfaces are now connected via the new subnet. The inconsistencies possibly resulting from the modified configuration are clarified automatically or per query. In the case of consistency problems, the next free address is preferably used.

The following schematic shows two networked devices:

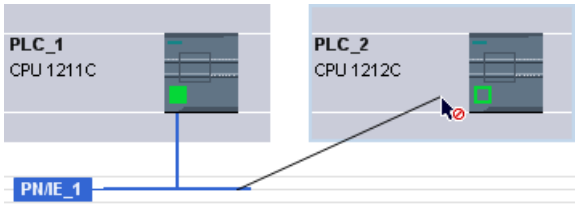


Procedure - Connecting devices to existing subnet

To connect an interface to an existing subnet, proceed as follows:

1. Position the mouse pointer on the interface of a communications-compliant component you want to network or on the existing subnet.
2. Click with the left mouse button and hold the button down.
3. Move the mouse pointer.
   The pointer now uses the networking symbol to indicate "Networking" mode. At the same time, the mouse pointer shows the lock symbol that will only disappear once the pointer is moved to a valid target.

4. Now move the mouse pointer to the existing subnet or to the interface to be networked. You can either keep the mouse button pressed or release it.



5. Now release the left mouse button or press it again.
   If you want to exit networking mode beforehand, press <Esc> or right-click.

Result:

The interface and selected subnet are now connected. The inconsistencies possibly resulting from the modified configuration are clarified automatically or per query. In the case of consistency problems, the next free address is preferably used.

Procedure - selecting an existing subnet from a list

To link an interface with a subnet that has already been created, proceed as follows:

1. Select the interface of a CPU.
2. Select the "Assign to new subnet" command in the shortcut menu of the interface.
   A list box containing the available subnets appears.
3. Select a subnet from the list.

The interface and selected subnet are now connected. The inconsistencies possibly resulting from the modified configuration are clarified automatically or per query. In the case of consistency problems, the next free address is preferably used.

Tabular network overview

Table area of network view

The table area of the network view consists of several tables:

- Network overview: The table for the network overview supplements the graphical network view with the following functions:
  - You obtain detailed information on the structure and parameter settings of the devices.
  - Using the "Subnet" column, you can connect components capable of communication with created subnets.
- Connections: Communication connections are displayed in this table. Some aspects of the connections displayed here can be edited directly in the table. Selected connections are displayed in the Inspector window and can be edited there.
- Relations: Logical connections of the HMI client-server relation are displayed here.

- I/O communication: Device communication via PROFIBUS DP and PROFINET IO is configured and displayed in this table. In the I/O communication table, only those interfaces of a device are displayed that can be used for communication via PROFIBUS DP or PROFINET IO.
- VPN: Properties of the VPN user groups and their assigned security modules are displayed here.

Basic functions for tables

The network overview supports the following basic functions for editing a table:

- Displaying and hiding table columns
  Note: The columns relevant for configuration cannot be hidden.
- Optimizing column width
- Sorting table
- Displaying the meaning of a column, a row or cell using tooltips.

Networking devices in the device view

Networking in the device view

In the device view, you can check and set all the parameters of the components belonging to a device and the interfaces in detail. Here you can also assign the interfaces to the subnets created in the project.

Requirements

- The subnet with which you want to connect an interface has already been created.
- If the subnet has not yet been created, change to the network view and perform the networking.

Procedure - connecting to an existing subnet

To connect an interface to an existing subnet, proceed in the device view as follows:

1. Select the entire communications-compliant component or the interface to be networked. The properties of the selected interface or component are displayed in the Inspector window.
2. In the Inspector window, select the parameter group for the selected interface; for example, the "Ethernet addresses" parameter group for a PROFINET interface.
3. Select the subnet to be connected from the "Interface connected with" drop-down list.

The interface and selected subnet are now connected. The inconsistencies possibly resulting from the modified configuration are clarified automatically or per query. In the case of consistency problems, the next free address is preferably used.

Procedure - creating a new subnet

To create a subnet and to connect it to the interface, proceed as follows in the device view:

1. Select the entire communications-compliant component or the interface to be networked. The properties of the selected interface or component are displayed in the Inspector window.
2. In the Inspector window, select the parameter group belonging to the selected interface, for example, the "Ethernet addresses" parameter group for a PROFINET interface.
3. In "Interface connected with", click the "Add new subnet" button.

The interface is connected to a new subnet of the appropriate subnet type. The inconsistencies possibly resulting from the modified configuration are clarified automatically or per query. In the case of consistency problems, the next free address is preferably used.

Checking or changing network parameters and interface parameters

Introduction

Communication between networked devices requires the following parameters to be configured:

- Network parameters
  Network parameters identify the network within the system configuration, for example, using a name.
- Interface parameters
  Interface parameters define specific properties of a component capable of communication. Addresses and transmission characteristics are set automatically and are consistent with the network parameters.

Note

Network parameters and interface parameters are usually set during networking such that communication can take place for numerous applications without the parameters having to be changed. The inconsistencies possibly resulting from a modified configuration are clarified automatically or per query. In the case of consistency problems, the next free address is preferably used.

Procedure - checking or changing network parameters

Proceed as follows to check or change network parameters:

1. Enter the network view.
2. Select the subnet from the network view.
   You can see the network parameters in the "Properties" tab in the inspector window.
3. If necessary, check or modify the network parameters in the relevant group of parameters.

Procedure - checking or changing interface parameters

You can check and modify interface parameters in the network and device view.

Proceed as follows to check or change interface parameters:

1. Enter the network view or device view.
2. Select the interface.
   You can see the interface parameters in the "Properties" tab in the inspector window.
3. If necessary, check or modify the interface parameters in the relevant group of parameters.

Procedure - checking or changing interface addresses

You can edit the MPI, PROFIBUS and Ethernet addresses in the graphical network view itself.

Proceed as follows to edit MPI, PROFIBUS and Ethernet addresses in the network view:

1. Enable the display of interface addresses in the network view.
2. Select an MPI, PROFIBUS or Ethernet address to be edited.
3. Click on the selected address field or press [F2] to edit the address in the network view.

Changing networkings

Introduction

You can cancel an interface's network connection or assign it to another subnet of the same subnet type.

Consequences

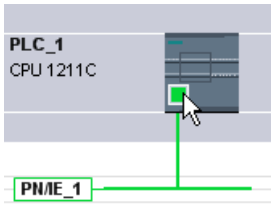
Depending on the version, a distinction should be made between:

- Canceling a network connection for an interface
  The configured parameters for the interface remain unchanged.
- Assigning a network connection to another subnet
  If the addresses in the assigned subnet are not unique, in other words, they already exist, they will be changed automatically to make them unique.

Procedure - disconnecting from a network

Proceed as follows to cancel the network connection for an interface:

1. Select the networked interface.



2. Select the "Disconnect from subnet" command in the shortcut menu of the interface.

The network connection is deleted, the interface addresses are, however, not changed.

Configured connections are retained; however these connections are marked red in the connection table because they are not networked. Specified connections remain specified.

See also

Networking devices in the network view (Page 586)

Copying, cutting or deleting subnets

Introduction

You can copy subnets as individual objects or copy them along with networked devices or other networks.

For example, you can create complex configurations to be used more than once in different variants within the project with no additional effort.

Effects on the copied subnet

Properties that have to be assigned explicitly within a project are re-assigned appropriately when the copied objects are copied.

For subnets, this means: The subnet ID and name are re-assigned to the copied subnet.

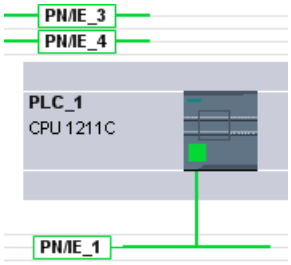
The configured properties are adopted in the copied subnet.

Procedure - copying a subnet

Proceed as follows to copy one or more subnets:

1. Select one or more subnets.
2. In the shortcut menu, select the "Copy" command.
3. Select the "Paste" command in the shortcut menu.

The copied subnets are shown as "orphaned" subnets in the top part of the network view.



Procedure - copying a subnet with connected devices

To copy one or more subnets with networked devices, proceed as follows:

1. Select one or more subnets with the connected devices, for example by drawing a lasso around them.
2. In the shortcut menu, select the "Copy" command.
3. Select the "Paste" command in the shortcut menu.

Complete copies of the subnets and connected devices are created.

Configured connections are adopted and remain within the copied devices. Connections to devices that have not been copied are interrupted and become unspecified.

MPI network configuration (S7-300, S7-400, PC)

Allocating MPI addresses

Note the following for devices with an MPI interface: All devices of a subnet must have a different device address.

CPUs with MPI address ship with the default MPI address 2. Since you can only use this address once in the MPI subnet, you will have to change the default address in all other CPUs.

The following applies to devices with the article no. 6ES7 3xx-xxxxx-0AB0:

When planning the MPI addresses for several CPUs, you have to fill "MPI address gaps" for FMs and CPs with separate MPI addresses to prevent addresses being assigned twice.

Only when all the modules in a subnet have different addresses and your actual structure matches that of the network configuration produced, should you load the settings across the network.

Rules for assigning the MPI address

- Allocate the MPI addresses in ascending order.
- Reserve MPI address 0 for a programming device.
- You can link up to 126 (addressable) devices with one another in an MPI subnet; up to 8 devices at a transfer speed of 19.2 KB/s.
- All MPI addresses in an MPI subnet must be different.

You will find other rules relating to the structure of a network in the manuals for setting up automation systems.

PROFIBUS network configuration

PROFIBUS addresses

Rules for the network configuration

All the devices in a subnet must have a different PROFIBUS address.

Only when all the modules in a subnet have different addresses and your actual structure matches that of the network configuration produced, should you load the settings across the network.

You can connect devices to the PROFIBUS subnet that communicate via configured connections or that belong to a PROFIBUS DP master system.

You can find more information on configuring a DP master system in the following sections.

Requirements

The 121xC CPU is PROFIBUS compatible as of firmware version 2.0.

Rules for assigning PROFIBUS addresses

- Allocate the PROFIBUS addresses in ascending order.
- Reserve the PROFIBUS address "0" for a programming device.
- Allocate a unique PROFIBUS address between 0 and 126 for each device on the PROFIBUS network and/or for each DP master and each DP slave in the PROFIBUS network.
- There are modules with which the smallest address that can be set has to be greater than 1.
- All PROFIBUS addresses of a PROFIBUS subnet must be different.

You will find additional rules relating to the structure of a network in the manuals for setting up automation systems, for example SIMATIC S7-1200.

Note

PROFIBUS address "0"

Reserve PROFIBUS address "0" for a programming device that you will briefly connect up to the PROFIBUS network at a later date for servicing.

See also

What you need to know about PROFIBUS bus parameters (Page 596)

What you need to know about PROFIBUS bus parameters

Matching parameters to one another

The PROFIBUS subnet will only function without problem if the parameters for the bus profile are matched to one another. You should therefore only change the default values if you are familiar with how to configure the bus profile for PROFIBUS.

Note

It may be possible for the bus parameters to be adjusted depending on the bus profile. If the bus parameters cannot be adjusted, they are grayed out. The offline values of the bus parameters are always shown even if you are online and linked to the target system.

The parameters shown apply to the entire PROFIBUS subnet and are briefly explained below.

Activating cyclic distribution of the bus parameters

If the "Enable cyclic distribution of the bus parameters" check box is selected under "Bus parameters" while PROFIBUS subnet is selected in the Inspector window, the bus parameters are transferred cyclically during operation by the modules that support this function. This allows a programming device, for example, to be easily connected to the PROFIBUS in runtime.

You should deactivate this function:

- For a heterogeneous PROFIBUS subnet (or more accurately: when external devices are connected whose protocol uses the DSAP 63 for Multicast)
- When in constant bus cycle time mode (minimize bus cycle)

Bus parameters for the bus profile of PROFIBUS subnets

Bus parameter	Adjustable?	Limit values
Tslot_Init	Yes	Max. Tsdr + 15 <= Tslot_init <= 16.383 t_bit
Max. Tsdr	Yes	35 + 2*Tset + Tqui <= Max. Tsdr <= 1.023 t_bit
Min. Tsdr	Yes	11 t_bit <= Min. Tsdr <= MIN(255 t_bit, ... Max. Tsdr - 1, 34 + 2*Tset + Tqui)
Tset	Yes	1 t_bit <= Tset <= 494 t_bit
Tqui	Yes	0 t_bit <= Tqui <= MIN(31 t_bit, Min. Tsdr - 1)
GAP factor	Yes	1 <= GAP factor <= 100
Retry limit	Yes	1 <= Retry limit <= 15
Tslot	No	---
Tid2	No	Tid2 = Max. Tsdr
Trdy	No	Trdy = Min. Tsdr
Tid1	No	Tid1 = 35 + 2*Tset + Tqui
Ttr	Yes	256 t_bit <= Ttr <= 16.777.960 t_bit

Bus parameter	Adjustable?	Limit values
Ttr typical	No	This time is provided for information only and is not transferred to the nodes.
Response monitoring		10 ms <= response monitoring (watchdog) <= 650 s

If you want to create a customized bus profile, we recommend the following settings:

- Minimum target rotation time (Ttr) = 5000 x HSA (highest PROFIBUS address)
- Minimum response monitoring (watchdog) = 6250 x HSA

Recalculating

You can use the "Recalculate" button to recalculate the parameters.

See also

PROFIBUS addresses (Page 595)

Description of the bus parameters (Page 597)

Description of the bus parameters

Detailed description of PROFIBUS bus parameters

Bus parameter	Meaning
Tslot_Init	The wait-for-reception (slot time) defines the maximum time the sender will wait to receive a response from the addressed partner. If the influence of the line components on message frame run times is entered in the "Cable Configuration" parameter group, these components must also be taken into account. The component is added to the specified Tslot_Init and the total used as Tslot.
Max. Tsdr	The maximum protocol processing time defines the latest time by which the responding node should have replied.
Min. Tsdr	The minimum protocol processing time defines the earliest time by which the responding node may reply.
Tset	The trigger time is the time which may lapse between the reception of a data message frame and the response to it in the node.
Tqui	The modulator quiet time is the time which a sending node needs after the end of the message frame to switch from sending to receiving.
GAP factor	The GAP update factor defines the number of token rotations after which a newly added, active node can be added to the logical token ring.
Retry limit	This parameter defines the maximum number of attempts (message frame repeats) made to reach a node.
Tslot	The wait-for-reception time (slot time) defines the maximum time the sender will wait to receive a response from the addressed partner. If the influence of the bus design components on message frame run times is entered in the "Cable Configuration" parameter group, these components must also be taken into account. The component is added to the specified Tslot_Init and the total used as Tslot.

Bus parameter	Meaning
Tid2	Idle time 2 defines the earliest time by which a sending node may send the next message frame after sending a message frame that has not been acknowledged.
Trdy	The ready time specifies the earliest time by which a sending node may receive a response message frame.
Tid1	Idle time 1 defines the earliest time by which a sending node may send the next message frame after receiving a response.
Ttr	The target rotation time is the maximum time made available for a token rotation. During this time, all active nodes (DP masters etc.) have the right to send (token) once. The difference between the target rotation time and the actual token holding time of a node determines how much time is left over for the other active nodes (programming device, other DP masters etc.) to send message frames.
Ttr typical	The typical data cycle time is the average response time on the bus if all configured slaves are exchanging data with the DP master. None of the slaves report diagnostics and there is no extra message frame traffic with programming devices or other active nodes etc. on the bus.
Response monitoring	The response monitoring time is only needed for PROFIBUS-DP bus systems. It defines the latest time by which a DP slave has to be addressed by its DP master with a new data message frame. If this does not happen, the DP slave assumes that the DP master has failed and resets its outputs to a secure mode.

See also

What you need to know about PROFIBUS bus parameters (Page 596)

Bus profiles with PROFIBUS

Introduction

Depending on the device types connected and protocols used on the PROFIBUS, different profiles are available. The profiles differ in terms of the setting options and calculation of bus parameters. The profiles are explained below.

Devices with different profiles on the same PROFIBUS subnet

The PROFIBUS subnet only functions without problem if the bus parameters of all devices have the same values. For example, if both DP and FMS services are being used on a subnet, the "slower" sets of bus parameters must always be set for all devices, i.e. even the "Universal (DP/FMS)" profile for DP devices.

Profiles and transmission rates

Profiles	Supported transmission speeds in Kbits/s
DP	9.6 19.2 45.45 93.75 187.5 500 1500 3000 6000 12000
Standard	9.6 19.2 45.45 93.75 187.5 500 1500 3000 6000 12000
Universal (DP-FMS)	9.6 19.2 93.75 187.5 500 1500
Customized	9.6 19.2 45.45 93.75 187.5 500 1500 3000 6000 12000

Meaning of profiles

Profile	Meaning
DP	Select the "DP" bus profile when the only devices connected to the PROFIBUS subnet are those which satisfy the requirements of standard EN 50170 Volume 2/3, Part 8-2 PROFIBUS. The bus parameter setting is optimized on these devices. This includes devices with DP master and DP slave interfaces of the SIMATIC S7 and distributed I/Os of other manufacturers.
Standard	Compared to the "DP" profile, the "Standard" profile also offers scope for devices of another project or devices which have not been configured here to be taken into account when calculating the bus parameters. The bus parameters are then calculated following a simple, non-optimized algorithm.
Universal (DP/FMS)	Select the "Universal (DP/FMS)" bus profile when individual devices on the PROFIBUS subnet use the PROFIBUS-FMS service. This includes the following devices for example: • CP 343-5 (SIMATIC S7) • PROFIBUS-FMS devices of other manufacturers As with the "Standard" profile, this profile allows you to take other devices into account when calculating the bus parameters.
Customized	The PROFIBUS subnet will only function without problem if the parameters for the bus profile are matched to one another. Select the "Customized" profile when none of the usual profiles "match" a PROFIBUS device and you need to adapt the bus parameters to your special structure. Information on this can be found in the documentation for the PROFIBUS device. You should only change the default values if you are familiar with how to configure the bus profile for PROFIBUS. Not all combinations that can be theoretically set can be used even with this bus profile. The PROFIBUS standard specifies several parameter limits that depend on other parameters. For example, a responder must not respond (Min Tsdr) before the initiator can receive the message frame (Trdy). These standard specifications are also checked in the "Customized" profile. Tip: The bus parameters last valid on the PROFIBUS subnet are always automatically set as customized. For example, if the "DP" bus profile was valid for the subnet, then the bus parameters for "DP" are set in the "Customized" bus profile. The parameters can be modified on this basis. The monitoring times are not automatically recalculated so that you do not put at risk the consistency of set values, for example with configurations in other configuration tools without realizing that you have done so. You can also have the Ttr monitoring times and target rotation time calculated on the basis of parameters you have set: Click here on the "Recalculate" button.

Note

Both mono-master mode and multi-master mode are possible with all PROFIBUS profiles.

What you need to know about PROFIBUS line configuration

Cable configuration and bus parameters

Information regarding the cable configuration can be taken into consideration when calculating the bus parameters. For this purpose, you must select the "Consider cable configuration" check box in the properties for the PROFIBUS subnet.

The remaining information then depends on the type of cable used; the following settings are available:

- Copper cable
- Fiber-optic cable/optical ring

PROFIBUS line configuration, optical ring

The calculation depends on the OLM types used. The selection is made by activating the check box (multiple activation is possible, at least one OLM type must be selected):

- OLM/P12
- OLM/G12
- OLM/G12-EEC
- OLM/G12-1300

The following bus parameter adjustments are made:

- Configuration of a node not present

  Note

  The following restrictions apply to optical rings, even for passive nodes (DP slaves for example):

  A maximum of HSA-1 nodes may be connected to the PROFIBUS network. With an HSA of 126, addresses 126 and 125 must not be used. A maximum of 125 nodes are possible on the bus (No. 0 to 124).

  If an HSA is less than or equal to 125, the addresses HSA and greater may not be used. The address HSA-1 may not be used.

- Increase the retry value to 3
- Setting of minimum slot time needed for ring mode

  Note

  Short slot time values are needed for OLM/P12, average ones for OLM/G12 and OLM/G12-EEC and long ones for OLM/G12-1300. This results in high performance for small network lengths and average to low performance for average to large network lengths.

PROFIBUS communication load

Communication load - allowing for additional network stations

The bus parameters depend on the volume of communication between the active network nodes. There are differences between cyclic communication (DP) and connection-based, acyclic communication (S7 communication, Send/Receive (FDL), FMS). Unlike DP, the volume and size of communication tasks (communication load) depends on the user program. For this reason, the communication load cannot always be calculated automatically.

To calculate the bus times you can define a network configuration in the "Additional network stations" parameter group that differs from the network configuration.

Taking the profile into account

The network configuration can be defined for the "Standard", "Universal (DP/FMS)", and "User-defined" profiles. Parameters cannot be entered in the "Additional network stations" parameter group for the "DP" profile.

Quantification of the communication load

The following settings are available in order to make allowance for the communication load.

- Information regarding the number of unconfigured network stations;
- Information on the communication load resulting from the user programs for FDL or S7 communication. Here you can selected from the following settings:
  - Low
    Typically used for DP, no great data communication apart from DP.
  - Medium
    Typically used for mixed operations featuring DP and other communication services (such as for S7 communication), when DP has strict time requirements and for average acyclic volumes of communication.
  - High
    For mixed operations featuring DP and other communication services (such as for S7 communication), when DP has loose time requirements and for high acyclic volumes of communication.

Configuring Industrial Ethernet

Rules for the network configuration

The Ethernet interfaces of the devices have a default IP address that you can change.

IP address

The IP parameters are visible if the module capable of communication supports the TCP/IP protocol. This is usually the case for all Ethernet modules.

The IP address consists of 4 decimal figures in the range of 0 to 255. The decimal figures are separated from one another by a dot.

Example: 140.80.0.2

The IP address consists of:

- Address of IP subnet
- Address of the device (generally also called host or network node)

Subnet mask

The subnet mask splits these two addresses. It determines which part of the IP address addresses the network and which part of the IP address addresses the device.

The set bits of the subnet mask determine the network part of the IP address.

Example:

Subnet mask: 255.255.0.0 = 11111111.11111111.00000000.00000000

In the example given for the above IP address, the subnet mask shown here has the following meaning:

The first 2 bytes of the IP address identify the subnet - i.e. 140.80. The last two bytes address the device, i.e. 0.2.

It is generally true that:

- The network address results from AND logic operation of the IP address and subnet mask.
- The device address results from the AND NOT logic operation of the IP address and subnet mask.

Relation between IP address and default subnet mask

An agreement exists relating to the assignment of IP address ranges and so-called "Default subnet masks". The first decimal number (from the left) in the IP address determines the structure of the default subnet mask. It determines the number of "1" values (binary) as follows:

IP address (decimal)	IP address (binary)	Address class	Default subnet mask
0 to 126	0xxxxxxx.xxxxxxxx....	A	255.0.0.0
128 to 191	10xxxxxx.xxxxxxxx...	B	255.255.0.0
192 to 223	110xxxxx.xxxxxxxx...	C	255.255.255.0

Note

Range of values for the first decimal point

A value of between 224 and 255 is also possible for the first decimal number of the IP address (address class D, etc). This is not recommended, however, because there is no address check for these values.

Masking other subnets

You can use the subnet mask to add further structures and form "private" subnets for a subnet that is assigned one of the address classes A, B or C. This is done by setting other lower points of the subnet mask to "1". For each bit set to "1", the number of "private" networks doubles and the number of devices they contain is halved. Externally, the network functions like an individual network as it did previously.

Example:

You have a subnet of address class B (for example IP address 129.80.xxx.xxx) and change the default subnet mask as follows:

Masks	Decimal	Binary
Default subnet mask	255.255.0.0	11111111.11111111.00000000.00000000
Subnet mask	255.255.128.0	11111111.11111111.10000000.00000000

Result:

All devices with addresses between 129.80.001.xxx and 129.80.127.xxx are on one IP subnet, all devices with addresses between 129.80.128.xxx and 129.80.255.xxx are on another IP subnet.

Router

The task of the routers is to connect the IP subnets. If an IP datagram is to be sent to another network, it first has to be conveyed to a router. To make this possible, you have to enter the address of the router for each device in the IP subnet.

The IP address of a device in the subnet and the IP address of the router can only differ at the points at which there is a "0" in the subnet mask.

See also

Settings for interconnecting Ethernet devices (Page 603)

Settings for interconnecting Ethernet devices

The following paragraph describes the behavior of STEP 7 during interconnection of PROFINET devices and the effects of the port interconnection on the network view.

Relation between port interconnection and network view

Topology view

In the topology view, you specify the physical interconnection of Ethernet ports. You specifically determine which Ethernet port of a device is to be connected with a specific Ethernet port of another device by means of an Ethernet cable (preset topology).

Example:

You specify that port 1 of the PROFINET interface of the CPU is to be connected to port 2 of the PROFINET interface of device A by means of an Ethernet cable. You also specify the interface for devices with several PROFINET interfaces.

Example:

You specify that port 1 of PROFINET interface X2 of the CPU is to be connected to port 2 of the PROFINET interface of device A by means of an Ethernet cable. The Ethernet ports can be interconnected in a table or graphically.

Network view

In the network view, you specify which devices are to be connected with each other via an Ethernet subnet. You do not specify the Ethernet ports by which the devices are interconnected with each other (that is the task of the port interconnection).

The port interconnection has effects on the network view: When you interconnect Ethernet ports of devices with each other in the topology view, STEP 7 connects the interconnected PROFINET interfaces of the devices in the network view with an Ethernet subnet (green line). However, the course of the green line does not reflect the actual cable routing. You specify the actual wiring in the topology view.

An Ethernet subnet always has a name and an S7 subnet ID. You can set these two values in the subnet properties.

Which Ethernet subnet is used to link interconnected devices?

STEP 7 distinguishes between the following cases:

- Ethernet subnet specified (default subnet).
- No Ethernet subnet specified (no default subnet).

Ethernet subnet specified (default subnet)

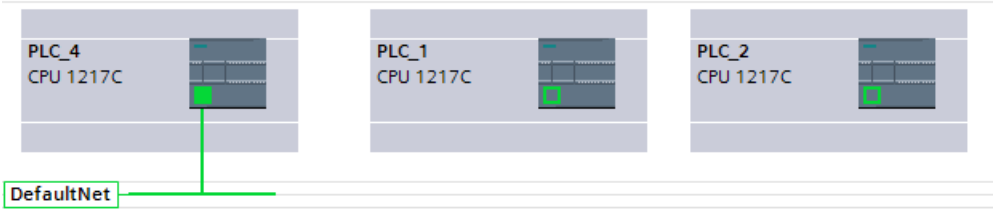
The option "Connect devices that are not linked with this subnet in case of port connection" is activated in the properties of an Ethernet subnet (default). This option can be activated for exactly one Ethernet subnet. Activate the option, if necessary, for the Ethernet subnet that is to be continued when interconnecting devices that are not linked. This subnet is referred to as "default subnet" below.

Response of STEP 7:

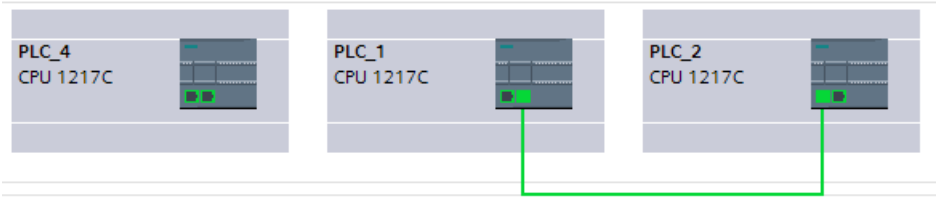
When you interconnect the ports of two devices that are not linked in the topology view, STEP 7 links these devices with the default subnet.

Example:

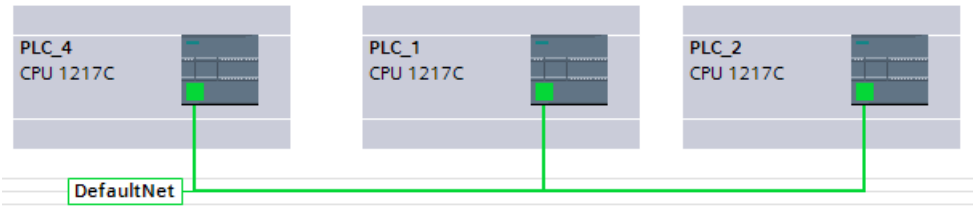
Step 1: Create a subnet in the network view at PLC_4, "Connect devices that are not linked with this subnet in case of port connection" option is **enabled**.



Step 2: Interconnect PLC_1 with PLC_2 (topology view).



Result: All PLCs are now connected to the same default subnet (network view).



No Ethernet subnet specified (no default subnet).

This is the case when the following conditions are fulfilled:

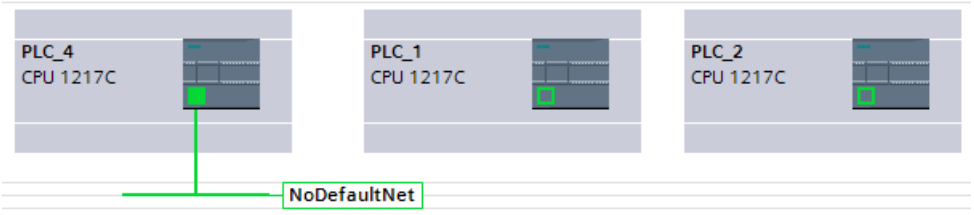
- A subnet has been added to a PROFINET interface (this interface is referred to as "Interface A" below).
- The option "Connect devices that are not linked with this subnet in case of port connection" is deactivated in the properties of this subnet (no default subnet).
- There is no other Ethernet subnet for which this option is activated.

Response of STEP 7:

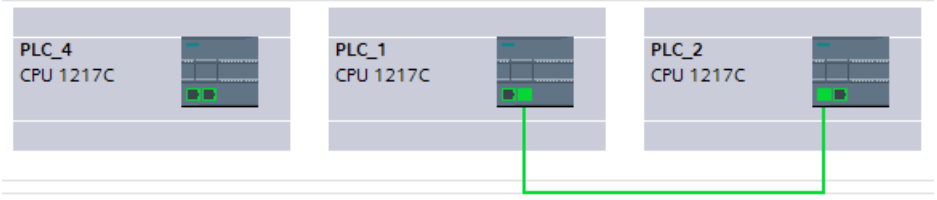
- The Ethernet subnet of interface A is only continued when you interconnect a port of interface A with a port of another device.
- When you interconnect ports of other devices that are not linked with each other, STEP 7 creates a new Ethernet subnet.

Example:

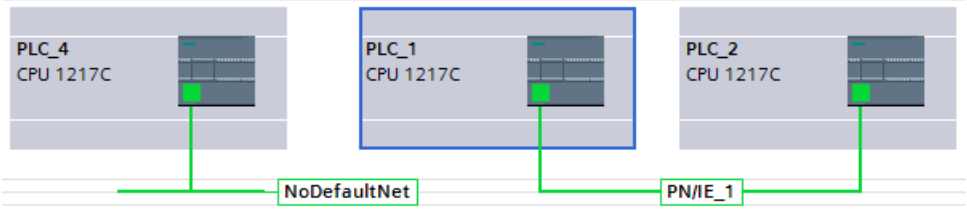
Step 1: Create a subnet in the network view at PLC_4, "Connect devices that are not linked with this subnet in case of port connection" option is **disabled**.



Step 2: Interconnect PLC_1 with PLC_2 (topology view).



Result: PLC_1 and PLC_2 are connected to a new subnet (network view).



Manual adaptation of the IP addresses

It can occur that STEP 7 does not adapt the IP addresses of the interconnected devices in such a way that the resulting network can be correctly compiled, for example, for devices with multiple PROFINET interfaces. In these situations, you need to adapt the IP addresses of the devices manually.

The following rules apply:

- Devices that are to communicate with each other without a router cannot belong to different IP subnets.
- For devices with multiple PROFINET interfaces, the interfaces must be in different IP subnets.

To change the IP address of a PROFINET interface, perform the following steps:

1. Change to the network view (if not selected already).
2. Left-click on the icon of the PROFINET interface that is not to be part of the IP subnet.
3. Change the subnet part of the IP address in the properties of the PROFINET interface (in the "Ethernet addresses" area).

Example: Manual conversion of the IP subnet part for devices with multiple PROFINET interfaces in a subnet

The IP address is: "192.168.0.1".

The subnet mask is "255.255.255.0".

The first three numbers "192.168.0" form the IP subnet part of the IP address "192.168.0.1".

Change the IP subnet part, for example, to "192.168.1"

See also

Interconnecting ports in the graphic view (Page 713)

Interconnecting ports in the table view (Page 714)

Configuring Industrial Ethernet (Page 601)

10.1.3.2 Communication via connections

Working with connections

Using Open User Communication (S7-1200, S7-1500)

Basics of Open User Communication (S7-1200, S7-1500)

Introduction

Open User Communication (OUC) is the name given to a program-controlled communication process for communicating via the integrated PN/IE interface of S7-1200/1500 and S7-300/400 CPUs. Different connection types are available for this communication process.

The main feature of Open User Communication is its high degree of flexibility in terms of the data structures transferred. This allows open data exchange with any communicating devices providing they support the connection types available here. Since this communication is controlled solely by instructions in the user program, event-driven connection establishment and termination is possible. Connections can also be modified by the user program during runtime.

For CPUs with an integrated PN/IE interface, the TCP, UDP, and ISO-on-TCP connection types are available for Open User Communication. The communication partners can be two SIMATIC PLCs or a SIMATIC PLC and a suitable third-party device.

Instructions for Open User Communication

To create the connections, you have various instructions available after opening in the program editor in the "Instructions > Communication > Open User Communication" task card:

- Compact instructions for sending or receiving data via the integrated functions for establishing and terminating the connection (S7-1200/1500 only):
  - TSEND_C (Page 4419) (connection establishment/termination, sending)
  - TRCV_C (Page 4432) (connection establishment/termination, receiving)
- Individual instructions for sending and receiving data or for establishing or terminating connections:
  - TCON (Page 4474) (connection establishment)
  - TDISCON (Page 4484) (connection termination)
  - TSEND (Page 4490) (TCP or ISO-on-TCP: Sending)
  - TRCV (Page 4493) (TCP or ISO-on-TCP: Receiving)
  - TUSEND (Page 4516) (UDP: Sending)
  - TURCV (Page 4520) (UDP: Receiving)

Connection establishment

For Open User Communication, instructions for establishing and terminating the connection must exist for both communication partners. One communication partner sends its data using TSEND, TUSEND or TSEND_C while the other communication partner receives the data using TRCV, TURCV or TRCV_C.

One of the communication partners starts the connection establishment as the active partner. The other communication partner reacts by starting its connection establishment as the passive partner. If both communication partners have initiated their connection establishment, the communication connection is fully established.

Connection configuration

You can specify establishment of the connection via a connection description DB with the TCON_Param, TCON_IP_v4, or TCON_IP_RFC structure by means of parameter assignment as follows:

- Manually create, assign parameters and write directly to the instruction.
- Supported by connection configuration.

Connection configuration supports the establishment of the connection and should, therefore, be given preference over the other methods.

You specify the following in the connection configuration:

- Connection partner
- Connection type
- Connection ID

- Connection description DB
- Address details according to selected connection type

In addition, you specify here which communication partner activates the connection establishment and which partner establishes a connection passively in response to a request from its communication partner.

See also

Principle of operation of connection-oriented protocols (Page 620)

Connection configuration (S7-1200, S7-1500)

Overview of connection configuration (S7-1200, S7-1500)

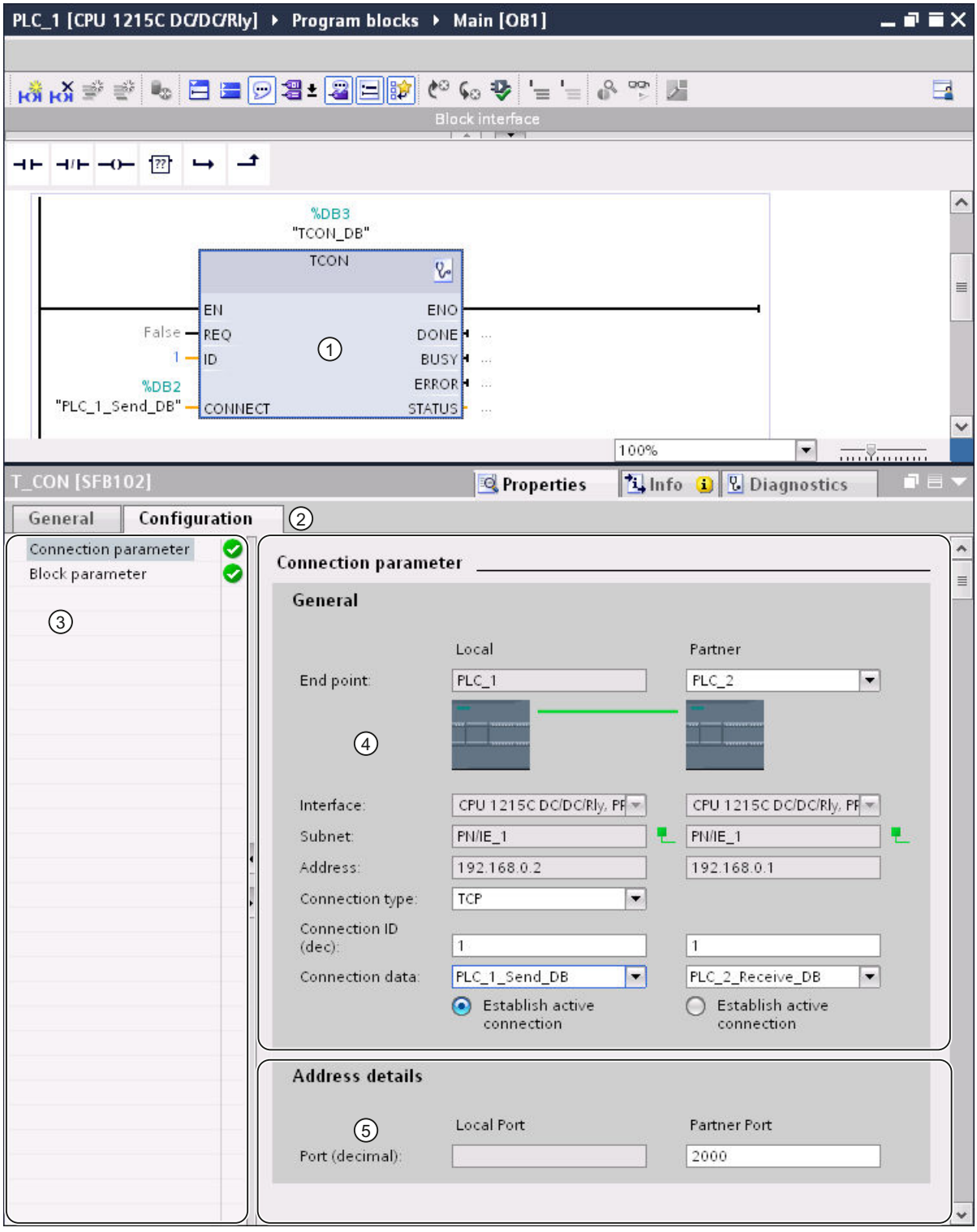
Introduction

You can find the connection configuration in the inspector window of the program editor if you want to program Open User Communication with the communication instructions TSEND_C, TRCV_C or TCON.

Connection configuration supports the flexible functionality of communication programming: The parameters entered for the connection configuration are stored in an automatically generated global DB derived from the TCON_Param, TCON_IP_v4 or TCON_IP_RFC structure. You can modify the connection parameters in this connection description DB.

Structure of the connection configuration

The connection configuration is made up of the following components:



① Communication instruction for TCON, TSEND_C or TRCV_C

② "Configuration" tab in the "Properties" tab
③ Area navigation of the "Configuration" tab
④ General properties of the connection parameters
⑤ Address details of the connection parameters (for selected connection DBs)

"Configuration" tab

Enter the desired connection parameters in the "Configuration" tab. The area navigation of the "Configuration" tab includes the "Connection parameters" group. This group contains the connection configuration. Here, you can enter the parameters for the connections and the address details with system support. Here, you also connect the CONNECT (TCON, TSEND_C, TRCV_C) or ID (TCON, TSEND, TRCV, TUSEND, TURCV) block parameters of the selected communication instructions.

When all the required parameters are assigned, a check mark is set in front of the "Connection parameters" group in the area navigation.

Note

The connection parameter assignment does not check whether the connection IDs and port numbers (TCP, UDP) or TSAPs (ISO-on-TCP, ISO) are unique. When you configure Open User Communication, you should, therefore, make sure that the parameter settings are unique within a device.

See also

Connection parameters with structure according to TCON_Param (Page 622)

Connection parameters with structure according to TCON_IP_v4 (Page 625)

Connection parameters with structure according to TCON_IP_RFC (Page 631)

TSEND_C: Send data via Ethernet (Page 4419)

TSEND_C: Establishing a connection and sending data (Page 4423)

TRCV_C: Receive data via Ethernet (Page 4432)

TRCV_C: Establishing a connection and receiving data (Page 4437)

TCON: Establish communication connection (Page 4474)

TCON: Establish communication connection (Page 4478)

Description of the connection parameters (S7-1200, S7-1500)

Overview

The following table shows the general connection parameters:

Parameter	Description
End point	The names of the local end point and the partner end point are shown. The local end point is the CPU for which TCON, TSEND_C or TRCV_C is programmed. The local end point is, therefore, always known. The partner end point is selected from the drop-down list. The drop-down list shows all available possible connection partners including unspecified connection partners for devices whose data is unknown in the project. For S7-1500, broadcast can be selected as the partner end point (message to all subnet devices). For S7-1500 CPs/CMs, multicast can also be selected as the partner end point (message to a group within the subnet). The connection type is converted automatically to UDP in this case. As long as no connection partner is selected, all other parameters in the mask are disabled except for the "Connection data" parameter.
Interface	The interface of the local end point is displayed. If multiple interfaces are available, e.g., by means of CPs or CMs, the interface can be selected from the drop-down list. To display or select the partner interface, a specified partner end point must first be selected.
Subnet	The subnet of the local end point is displayed, provided this exists. The partner subnet is displayed only after the partner end point has been selected. If at least one of the two connection partners is not connected with a subnet, the two connection partners are connected with each other. A connection between partners in different subnets is only possible with IP routing. The IP routing settings can be edited in the properties of the relevant Ethernet interfaces.
Address	The IP address or the PROFIBUS address of the local end point is displayed, depending on the subnet used. The corresponding address of the partner is displayed only after the partner end point has been selected. If you have selected an unspecified connection partner, the input box is empty and has a red background. In this case, you must specify a valid IP address/PROFIBUS address. The address type (IP or PROFIBUS) depends on the type of subnet that is set for the local partner. Broadcast (S7-1500 only): If "Broadcast" is set as the partner end point, a non-editable IP address with host address 255 is entered automatically for the connection partner. The network allocation corresponds to that of the sender. Example: Local IP address 192.168.0.1, partner IP address 192.168.0.255. Multicast (S7-1500 CPs/CMs only): If "Multicast" is set as the partner end point, the editable IP address 224.0.1.0 is entered automatically for the connection partner.
Connection type	Select the connection type you want to use from the "Connection type" drop-down list: • TCP • ISO-on-TCP • UDP With the S7-1500, you can also select the ISO connection type at the configuration type of the configured connections for TSEND_C and TRCV_C or TCON. The connection types can only be used for partners that support the corresponding protocol.

Parameter	Description
Connection type (for S7-1500 only)	With the S7-1500, two different configuration types can be set for TSEND_C and TRCV_C: • Programmed connections use program blocks for the connection description. • Configured connections are saved for the configuration and are only created after download to the device in runtime. You can also use the configured connection to select the connection type ISO. The specified configuration method depends on the selected connection type. If both configuration methods are possible, the programmed connection is preset. The same configuration method must be set for both connection partners.
Connection ID	Enter the connection ID in the input box. You can change the connection ID in the input boxes or enter it directly in TCON. Ensure that the connection ID assigned is unique within the device.
Connection data	The names of the connection description DBs for the connection description structured according to TCON_IP_v4, TCON_IP_RFC or TCON_Param are displayed in the drop-down lists. You can use the drop-down list to generate a new data block or to select an existing data block. This data block is filled automatically with the values from the connection configuration. The name of the selected data blocks is entered automatically in the CONNECT block parameter of the selected TSEND_C, TRCV_C or TCON instruction. From the drop-down list, you can also reference another valid data block. If a DB is referenced using the CONNECT input parameter of the TSEND_C, TRCV_C or TCON extended instructions and this does not correspond to the structure of a TCON_IP_v4, TCON_IP_RFC or TCON_Param, the drop-down list is shown with no content on a red background.
Connection name (for S7-1500 only)	If the connection type of the configured connections is set for TSEND_C and TRCV_C for the S7-1500, the "Connection data" parameter is replaced with the "Connection name" parameter. The name of the configured connection serves here as the connection data. The drop-down list is still empty after selection of a connection partner. You can use the drop-down list to generate a new connection or to select an existing connection. If needed, a data block is created and automatically filled with the values from the connection configuration. The name of the data block is entered automatically in the CONNECT block parameter of the TSEND_C or TRCV_C instruction. You can also reference an existing connection from the drop-down list.
Active connection establishment	Use the "Active connection establishment" check box to specify the active partner of the Open User Communication (only with TCP and ISO-on-TCP).
Port (only with TCP and UDP)	Address component for a TCP or UDP connection. The default after creating a new TCP connection is 2000. You can change the port numbers. The port numbers must be unique on the device!
TSAP (ISO-on-TCP only)	Address component for an ISO-on-TCP connection. The default value after creating a new ISO-on-TCP connection is E0.01.49.53.4F.6F.6E.54.43.50.2D.31 (S7-1200/1500) or E0.02.49.53.4F.6F.6E.54.43.50.2D.31 (S7-300/400). You can enter the TSAP-ID with an extension or as an ASCII TSAP. The TSAPs must be unique on the device!

Note

UDP connection for the "Broadcast" setting (S7-300/400/1200)

The parameters of the UDP connection for the "Broadcast" setting for the partner end point are stored in a connection description DB TCON_IP_v4 : With respect to UDP communication with TCON and TUSEND/TURCV , the TCON_IP_v4 is not filled with any partner parameters (value=0). However, the partner address and the partner port are necessary for sending the data and must be entered by the user in the TADDR_Param . The TADDR_Param for UDP communication is referenced by the TUSEND-/TURCV block parameter ADDR . The values for both parameters can be taken from the connection configuration.

The configuration must also be adapted for the other recipients of UDP communication. In order to receive broadcast frames, the partner port must be configured at the receiver end. For this purpose, the RemotePort parameter of the TADDR_Param must be filled at the ADDR block.

Note

Communication via TSEND_C and TRCV_C (S7-1500)

When TSEND_C and TRCV_C are used, a separate TSEND_C and TRCV_C block pair with a configured connection is required for each communication. Multiple TSEND_C and TRCV_C block pairs cannot simultaneously use the same configured connection for communication.

Additional connections for a TSEND_C or TRCV_C instruction can be created in the inspector window for the connection parameters using the appropriate button next to the connection data.

The connections configured using TSEND_C and TRCV_C are displayed in a connection table in the inspector window under "Properties > Configuration > Overview of configured connections" when the TSEND_C or TRCV_C block is selected.

See also

Assignment of port numbers (Page 636)

TSAP structure (Page 639)

Examples of TSAP assignment (Page 641)

Ability to read back connection description parameters (Page 638)

Creating and assigning parameters to connections (Page 616)

Connection parameters with structure according to TCON_Param (Page 622)

Connection parameters with structure according to TCON_IP_v4 (Page 625)

Connection parameters with structure according to TCON_IP_RFC (Page 631)

Starting connection parameter assignment (S7-1200, S7-1500)

The connection configuration for Open User Communication is enabled as soon as a TCON, TSEND_C or TRCV_C instruction for communication is selected in a program block.

Requirement

- Your project must contain at least one S7-CPU.
- The program editor is open.
- A network is available.

Procedure

To insert the extended instructions for Open User Communication, proceed as follows:

1. Open the task card, pane and folder "Instructions > Communication > Open User Communication".
2. Drag one of the following instructions to a network:
   - TSEND_C
   - TRCV_C
   - TCON

   The "Call options" dialog opens.
3. Edit the properties of the instance DB in the "Call properties" dialog. You have the following options:
   - Change the default name.
   - Select the "Manual" check box to assign your own number.
   - You can also execute the DB as a multi-instance for function blocks.
4. Click "OK" to complete your entry.

Result

A corresponding instance DB is created at a single instance for the inserted instruction TSEND_C, TRCV_C or TCON. In the case of a multi-instance, the instance DB of the function block is used.

With TSEND_C, TRCV_C or TCON selected, you will see the "Configuration" tab under "Properties" in the Inspector window. The "Connection parameters" group in area navigation contains the connection parameter assignment that you can now make.

See also

Inserting FBD elements using the "Instructions" task card (Page 4949)

Inserting LAD elements using the "Instructions" task card (Page 4893)

Creating and assigning parameters to connections (Page 616)

TSEND_C: Send data via Ethernet (Page 4419)

TSEND_C: Establishing a connection and sending data (Page 4423)

TRCV_C: Receive data via Ethernet (Page 4432)

TRCV_C: Establishing a connection and receiving data (Page 4437)

TCON: Establish communication connection (Page 4474)

TCON: Establish communication connection (Page 4478)

Creating and assigning parameters to connections (S7-1200, S7-1500)

In the connection configuration for Open User Communication, you can create and configure connections of the TCP, UDP or ISO-on-TCP type.

Requirements

A CPU exists with a TCON, TSEND_C or TRCV_C communication instruction.

Procedure

To create a connection for Open User Communication, follow these steps:

1. Select a TCON, TSEND_C or TRCV_C block of Open User Communication in the program editor.
2. Open the "Properties > Configuration" tab in the inspector window.
3. Select the "Connection parameters" group. Until you select a connection partner, only the drop-down lists for the partner end point and the local connection data are enabled. All other input options are disabled.
   The connection parameters already known are displayed:
   - Name of the local end point
   - Interface of the local end point
   - IP address (for Ethernet subnet) or PROFIBUS address (for PROFIBUS subnet) of the local end point.
4. If you want to use an existing connection description DB, go to step 6. If you want to configure the connection with a new remote connection partner, continue here:
   In the drop-down list box of the partner end point, select a connection partner. You can select an unspecified device or a CPU in the project as the communication partner. Certain connection parameters are then entered automatically.
   The following parameters are set:
   - Name of the partner end point
   - Interface of the partner end point
   - IP address (for Ethernet subnet) or PROFIBUS address (for PROFIBUS subnet) of the partner end point.

   If the connection partners are networked, the name of the subnet is displayed.
5. With the S7-1500, in the "Configuration type" drop-down list, you choose between using program blocks or configured connections.

6. Select an existing connection description DB in the "Connection data" drop-down list or for configured connections select an existing connection under "Connection name". You can also create a new connection description DB or a new configured connection. When you select an existing connection description DB, the available configuration data is applied and entered in the connection description. Any data of a remote partner that was entered previously is overwritten. Later, you can still select other configured connections or change the names of the connection description DBs in order to create new data blocks:
   - You can also see the selected data block at the interconnection of the CONNECT input parameter of the selected TCON, TSEND_C or TRCV_C instruction.
   - If you have already specified a connection description DB for the connection partner using the CONNECT parameter of the TCON, TSEND_C or TRCV_C instruction, you can either use this DB or create a new DB.
   - If you edit the name of the displayed data block in the drop-down list, a new data block with the changed name but with the same structure and content is generated and used for the connection.
   - Changed names of a data block must be unique in the context of the communication partner.
   - A connection description DB must have the structure TCON_Param, TCON_IP_v4 or TCON_IP_RFC, depending on CPU type and connection.
   - A data block cannot be selected for an unspecified partner.

   Additional values are determined and entered after the selection or creation of the connection description DB or configured connection.
   The following is valid for specified connection partners:
   - ISO-on-TCP connection type
   - Connection ID with default of 1
   - Active connection establishment by local partner
   - TSAP ID
     for S7-1200/1500: E0.01.49.53.4F.6F.6E.54.43.50.2D.31
     for S7-300/400: E0.02.49.53.4F.6F.6E.54.43.50.2D.31

   The following is valid for unspecified connection partners:
   - TCP connection type
   - Partner port 2000

   The following applies for a configured connection with a specified connection partner:
   - TCP connection type
   - Connection ID with default of 257
   - Active connection establishment by local partner
   - Partner port 2000

   The following applies for a configured connection with an unspecified connection partner:
   - TCP connection type
   - Local port 2000

7. Enter a connection ID as needed for the connection partner. No connection ID can be assigned to an unspecified partner.

   Note

   You must enter a unique value for the connection ID at a known connection partner. The uniqueness of the connection ID is not be checked by the connection parameter settings and there is no default value entered for the connection ID when you create a new connection.

8. Select the desired connection type in the relevant drop-down list. Default values are set for the address details depending on the connection type. You can choose between the following:
   - TCP
   - ISO-on-TCP
   - UDP

   For configured connections with S7-1500, ISO applies in addition.
9. You can edit the input boxes in the address details. Depending on the selected protocol, you can edit the ports (for TCP and UDP) or the TSAPs (for ISO-on-TCP and ISO).
10. Use the "Active connection establishment" check box to set the connection establishment characteristics for TCP, ISO and ISO-on-TCP. You can decide which communication partner establishes the connection actively.

Changed values are checked immediately for input errors by the connection configuration and entered in the data block for the connection description.

Note

Open User Communication between two communication partners can only work when the program section for the partner end point has been downloaded to the hardware. To achieve fully functional communication, make sure that you load not only the connection description of the local CPU on the device but also that of the partner CPU as well.

See also

Description of the connection parameters (Page 612)

Starting connection parameter assignment (Page 614)

TSAP structure (Page 639)

Assignment of port numbers (Page 636)

Connection parameters with structure according to TCON_Param (Page 622)

Connection parameters with structure according to TCON_IP_v4 (Page 625)

Connection parameters with structure according to TCON_IP_RFC (Page 631)

TSEND_C: Send data via Ethernet (Page 4419)

TSEND_C: Establishing a connection and sending data (Page 4423)

TRCV_C: Receive data via Ethernet (Page 4432)

TRCV_C: Establishing a connection and receiving data (Page 4437)

TCON: Establish communication connection (Page 4474)

TCON: Establish communication connection (Page 4478)

Deleting connections (S7-1200, S7-1500)

Introduction

The data of a created connection for Open User Communication is stored in a connection description DB. You can delete the connection by deleting the data block containing the connection description.

Requirement

You have created an Open User Communication connection.

Procedure

To delete a connection, follow these steps:

1. Select a communication partner for Open User Communication in the project tree.
2. Open the "Program blocks > System blocks > Program resources" folder below the selected communication partner.
3. Select the "Delete" command from the shortcut menu of the data block with the connection parameter assignment.

Note

If you are not certain which block to delete, open the extended instruction TCON, TSEND_C or TRCV_C. You will find the name of the data block as the CONNECT input parameter or in the connection parameter assignment as the "Connection data" parameter.

If you only delete the instance DBs of the extended instructions TCON, TSEND_C or TRCV_C, the assigned connections are not deleted as well.

Note

If the connection DB is still being used by other blocks of the extended instructions, then the corresponding calls, their instance DBs, and, if present, the combination blocks TSEND_C and TRCV_C must also be deleted from the block folder, provided they are not used elsewhere.

This action prevents the program from being inconsistent.

Result

You have deleted the connection.

Note

Insert an extended instruction TCON, TSEND_C, or TRCV_C again in order to reference an existing connection description with the TCON_Param, TCON_IP_v4, or TCON_IP_RFC structure again via the "Connection data" parameter.

Connection parameters (S7-1200, S7-1500)

Principle of operation of connection-oriented protocols (S7-1200, S7-1500)

Introduction

Connection-oriented protocols establish a logical connection to the communication partner before data transmission is started. After the data transmission is complete, they then terminate the connection, if necessary. Connection-oriented protocols are used especially when reliable data transmission is important. Several logical connections can exist over one physical line.

Open User Communication supports the following connection types:

- TCP
- ISO-on-TCP
- ISO (S7-1500 only)
- UDP

Both communication partners must support the same connection type for a connection. If a communication partner does not support a connection of the type ISO-on-TCP, for example, use the connection type TCP instead, if it is supported.

For communication partners that cannot be configured in the TIA Portal, such as third-party devices or PCs, enter "unspecified" for the partner end point during connection parameter assignment. The required connection type for unspecified devices is listed in the respective documentation.

Note

Connections with ISO

For S7-1500 CPUs, configured connections of the type ISO can be created using the TSEND_C and TRCV_C instructions. For additional information on these connection types, refer to the general connection descriptions.

Characteristics of TCP

TCP is a streaming protocol in which the length of the data stream is transmitted to the receiver so that it can receive the data stream as individual TCP segments. This means no information about the start and end of a message is transmitted during data transmission via a TCP connection. The receiver cannot determine by the received segments of the data stream where one message in the data stream ends and the next one begins. It is therefore recommended that the number bytes to be received (parameter LEN, instruction TRCV/TRCV_C) be assigned the same value as the number of bytes to be sent (parameter LEN, instruction TSEND/ TSEND_C).

If the length of the sent data and the length of the expected data do not match, the following occurs:

- Length of data to be received (parameter LEN, instruction TRCV/TRCV_C) greater than length of data to be sent (parameter LEN, instruction TSEND/TSEND_C): TRCV/TRCV_C copies the received data to the specified receive area (parameter DATA) only after the assigned length is reached. When the assigned length is reached, data of the subsequent job are already being received. As a result, the receive area contains data from two different send jobs. If you do not know the exact length of the first message, you are unable to recognize the end of the first message and the start of the second message.
- Length of data to be received (parameter LEN, instruction TRCV/TRCV_C) less than length of data to be sent (parameter LEN, instruction TSEND/TSEND_C): TRCV/TRCV_C copies the number of bytes you specified in the LEN parameter to the receive data area (parameter DATA). Then, it sets the NDR status parameter to TRUE (job completed successfully) and assigns RCVD_LEN (amount of data actually received) the value of LEN. With each subsequent call, you receive a further block of the sent data.

A receive area with fixed data length can be specified in the TRCV/TRCV_C instructions with the protocol version of the Ad-hoc mode.

Characteristics of ISO-on-TCP

ISO-on-TCP is a message-oriented protocol which detects the end of the message at the receiver end and indicates the data that belongs to the message to the user. This does not depend on the specified reception length of the message. This means that information regarding the length and the end of a message is included during data transmission via an ISO-on-TCP connection.

If the length of the sent data and the length of the expected data do not match, the following occurs:

- Length of data to be received (parameter LEN, instruction TRCV/TRCV_C) greater than length of data to be sent (parameter LEN, instruction TSEND/TSEND_C): TRCV/TRCV_C copies all the sent data to the receive data area (parameter DATA). Then, it sets the NDR status parameter to TRUE (job completed successfully) and assigns RCVD_LEN (amount of data actually received) the length of the data sent.
- Length of data to be received (parameter LEN, instruction TRCV/TRCV_C) less than length of data to be sent (parameter LEN, instruction TSEND/TSEND_C): TRCV/TRCV_C does not copy any data to the receive data area (parameter DATA), but instead supplies the following error information: ERROR=1, STATUS=W#16#8088 (destination buffer too small).

Characteristics of UDP

UDP is a message-oriented protocol which detects the end of the message at the receiver end and indicates the data that belongs to the message to the user. This does not depend on the specified reception length of the message. This means that information on the length and the end of a message is included during data transmission via a UDP connection.

If the length of the sent data and the length of the expected data do not match, the following occurs:

- Length of data to be received (parameter LEN, instruction TRCV/TRCV_C) greater than length of data to be sent (parameter LEN, instruction TUSEND/TSEND_C): TURCV/TRCV_C copies all the sent data to the receive data area (DATA parameter). Then, it sets the NDR status parameter to TRUE (job completed successfully) and assigns RCVD_LEN (amount of data actually received) the length of the data sent.
- Length of data to be received (parameter LEN, instruction TRCV/TRCV_C) less than length of data to be sent (parameter LEN, instruction TUSEND/TSEND_C): TRCV/TRCV_C copies as much data to the receive data area (parameter DATA) as the LEN parameter requests. No further error message is generated. In this case, the user has to call a T_URCV again in order to receive the remaining bytes.

See also

Basics of Open User Communication (Page 607)

TSEND_C: Send data via Ethernet (Page 4419)

TRCV_C: Receive data via Ethernet (Page 4432)

TSEND: Send data via communication connection (Page 4490)

TRCV: Receive data via communication connection (Page 4493)

TUSEND: Sending data (Page 4516)

TURCV: Receiving data (Page 4520)

Connection parameters with structure according to TCON_Param (S7-1200, S7-1500)

Data block for connection description

A connection description DB with a structure according to TCON_Param is used for some S7-1200 CPUs when it comes to the assignment of parameters for TCP, UDP and ISO-on-TCP communication connections. The fixed data structure of the TCON_Param contains all the parameters that are needed to establish the connection. The connection description DB is automatically created for a new connection by the connection configuration for Open User Communication when the TSEND_C, TRCV_C or TCON instruction is used.

The CONNECT connection parameter of the instance DBs for TSEND_C, TRCV_C or TCON contains a reference to the data block used.

Structure of the connection description according to TCON_Param

Byte	Parameter	Data type	Start value	Description
0 … 1	block_length	UINT	64	Length: 64 bytes (fixed)
2 … 3	id	CONN_OUC	1	Reference to this connection (value range: 1 to 4095). You must specify the value of this parameter for the TSEND_C, TRCV_C or TCON instruction under ID.
4	connection_type	USINT	17	Connection type: • 17: TCP (17 dec = 0x11 hex) • 18: ISO-on-TCP (18 dec = 0x12 hex) • 19: UDP (19 dec = 0x13 hex)
5	active_est	BOOL	TRUE	Identifier for the type of connection establishment. FALSE always applies to UDP, since data can be sent and received via local ID. The following is valid for TCP and ISO-on-TCP: • FALSE: Passive connection establishment • TRUE: Active connection establishment
6	local_device_id	USINT	1	ID for the local PN/IE interface.
7	local_tsap_id_len	USINT	0	Length of parameter local_tsap_id used, in bytes; possible values: • 0 or 2, if connection type = 17 (TCP) Only the value 0 is permissible for the active side. • 2 to 16, if connection type = 18 (ISO-on-TCP) • 2, if connection type = 19 (UDP)
8	rem_subnet_id_len	USINT	0	This parameter is not used.
9	rem_staddr_len	USINT	4	Length of address of partner end point, in bytes: • 0: unspecified, in other words, parameter rem_staddr is irrelevant. • 4: valid IP address in the parameter rem_staddr (TCP and ISO-on-TCP only)
10	rem_tsap_id_len	USINT	2	Length of parameter rem_tsap_id used, in bytes; possible values: • 0 or 2, if connection type = 17 (TCP) Only the value 0 is permissible for the passive side. • 2 to 16, if connection type = 18 (ISO-on-TCP) • 0, if connection type = 19 (UDP)
11	next_staddr_len	USINT	0	This parameter is not used.

Byte	Parameter	Data type	Start value	Description
12 … 27	local_tsap_id	ARRAY [1..16] of BYTE	-	Local address component of connection: • TCP and UDP: local port no. (possible values: 1...49151; recommended values: 2000...5000); local_tsap_id[1] = high byte of port no. in hexadecimal notation; local_tsap_id[2] = low byte of port no. in hexadecimal notation; local_tsap_id[3-16] = irrelevant • ISO-on-TCP: local TSAP-ID: local_tsap_id[1] = B#16#E0; local_tsap_id[2] = rack and slot of local end points (bits 0 to 4: Slot number, bits 5 to 7: rack number); local_tsap_id[3-16] = TSAP extension, optional Note: Make sure that every value of local_tsap_id is unique within the CPU.
28 … 33	rem_subnet_id	ARRAY [1..6] of USINT	-	This parameter is not used.
34 … 39	rem_staddr	ARRAY [1..6] of USINT	-	TCP and ISO-on-TCP only: IP address of the partner end point, for example, for 192.168.002.003: • rem_staddr[1] = 192 • rem_staddr[2] = 168 • rem_staddr[3] = 002 • rem_staddr[4] = 003 • rem_staddr[5-6]= irrelevant
40 … 55	rem_tsap_id	ARRAY [1..16] of BYTE	-	Partner address component of connection • TCP: partner port no. (possible values: 1...49151; recommended values: 2000...5000); rem_tsap_id[1] = high byte of port no. in hexadecimal notation; rem_tsap_id[2] = low byte of port no. in hexadecimal notation; rem_tsap_id[3-16] = irrelevant • ISO-on-TCP: partner TSAP-ID: rem_tsap_id[1] = B#16#E0; rem_tsap_id[2] = rack and slot of partner end point (bits 0 to 4: Slot number, bits 5 to 7: rack number); rem_tsap_id[3-16] = TSAP extension, optional • UDP: This parameter is not used.
56 … 61	next_staddr	ARRAY [1..6] of BYTE	-	This parameter is not used.
62 … 63	spare	WORD	W#16#0000	Reserved.

Note

TCON_Param for S7-1500 CPU

The connection description DB with the structure according to TCON_Param is also supported by S7-1500 CPUs for migration reasons. However, we recommend that you use the new structures TCON_IP_v4 and TCON_IP_RFC.

See also

Principle of operation of connection-oriented protocols (Page 620)

Description of the connection parameters (Page 612)

Ability to read back connection description parameters (Page 638)

Basic principles for programming of data blocks (Page 4795)

Overview of connection configuration (Page 609)

TSAP structure (Page 639)

Assignment of port numbers (Page 636)

Connection parameters with structure according to TCON_IP_v4 (S7-1200, S7-1500)

Data block for connection description

A connection description DB with a structure according to TCON_IP_v4 is used for CPUs of S7-1200 V4.0 and higher and S7-1500 to assign parameters for TCP and UDP communication connections. The fixed data structure of the TCON_IP_v4 contains all parameters that are required to establish the connection. The connection description DB is automatically created for a new connection by the connection configuration for Open User Communication when the TSEND_C, TRCV_C or TCON instruction is used.

The CONNECT connection parameter of the instance DBs for TSEND_C, TRCV_C or TCON contains a reference to the data block used.

Structure of the connection description according to TCON_IP_v4

Byte	Parameter	Data type	Start value	Description
0 … 1	interface_id	HW_ANY	64	Hardware identifier of the local interface (value range: 0 to 65535).
2 … 3	id	CONN_OUC	1	Reference to this connection (value range: 1 to 4095). You must specify the value of this parameter for the TSEND_C, TRCV_C or TCON instruction under ID.

Byte	Parameter	Data type	Start value	Description
4	connection_type	BYTE	11	Connection type: • 11: TCP (11 dec = 0x0B hex) • 19: UDP (19 dec = 0x13 hex) For compatibility and migration reasons, the CPU S7-1500 also allows the values of the connection description DB with a structure according to TCON_Param. For the connection type TCP, the entry 17 is therefore also valid (17 dec = 0x11 hex).
5	active_established	BOOL	TRUE	Identifier for the type of connection establishment: • FALSE: Passive connection establishment • TRUE: Active connection establishment
6 … 9	remote_address	ARRAY [1..4] of BYTE	-	• IP address of the partner end point, for example, for 192.168.0.1: – addr[1] = 192 – addr[2] = 168 – addr[3] = 0 – addr[4] = 1 • Multicast address of an IPv4 multicast group (for S7-1500-CPUs as of firmware version V2.0 or V2.1 and higher when using TSEND_C/TRCV_C)
10 … 11	remote_port	UINT	2000	Port address of the remote connection partner (value range: 1 to 49151).
12 … 13	local_port	UINT	2000	Port address of the local connection partner (value range: 1 to 49151).

See also

Principle of operation of connection-oriented protocols (Page 620)
Description of the connection parameters (Page 612)
Ability to read back connection description parameters (Page 638)
Basic principles for programming of data blocks (Page 4795)
Overview of connection configuration (Page 609)
Assignment of port numbers (Page 636)

Connection parameters according to TCON_IP_V4_SEC (S7-1500)

Data block for connection description

To configure the communication connections for TCP over IPv4 with Secure Communication, use the following modules:

- S7-1500 as of firmware version V2.0
  With connection description DB with the structure of the SDT TCON_IP_V4_SEC
- Additionally optional: CP 1543-1 as of firmware version V2.0 or CP1543SP-1 as of firmware version V1.0

Note

Non-secure TCP or UDP connection over IPv4

You can also use SDT TCON_IP_V4_SEC for a non-secure TCP or UDP connection over IPv4.

The connection parameter CONNECT of the instance DBs for TCON contains a reference to the data block used.

Structure of the connection description according to TCON_IP_V4_SEC

Byte	Parameter	Data type	Start value	Description
0 … 15	ConnPara	TCON_IP_v4	-	SDT for the connection parameters Information about the interface_id: • If you leave the interface_id at the preset value of 0, the operating system of the CPU evaluates the remote IP address and the IP routes existing locally and then specifies an Industrial Ethernet interface of the CPU for establishing the secure OUC connection. In this case, the diagnostics data is always assigned to the first Industrial Ethernet interface of the CPU. • If you specify the hardware identifier of an Industrial Ethernet interface of the CPU or of a CP as interface_id, the secure OUC connection is established via the associated Industrial Ethernet interface.
16	ActivateSecure-Conn	BOOL	FALSE	Activation of Secure Communication for this connection If this parameter has the value FALSE, the subsequent security parameters are irrelevant, meaning that the connection is non-secure. You can set up a non-secure TCP or UDP connection in this case.
17	TLSServerReq-ClientCert	BOOL	FALSE	Only for the server side: Request for an X.509-V3 certificate from the TLS client

Byte	Parameter	Data type	Start value	Description
18 ... 19	ExtTLSCapabilities	WORD	16#0	• Bit 0: Only for the client side. A set bit means that the client validates the alternative name of the certificate subject (subjectAlternateName) in the X.509-V3 certificate of the server to check the identity of the server. The certificates are checked when the connection is established. • Bit 1 to 15: reserved for future upgrades
20 ... 23	TLSServerCertRef	UDINT	0	• Server side: ID of its own X.509-V3 certificate • Client side: ID of the X.509-V3 certificate (usually a CA certificate) that is used by the TLS client to validate the TLS server authentication. If this parameter is 0, the TLS client uses all the (CA) certificates currently loaded in the client certificate store to validate the server authentication.
24 … 27	TLSClientCertRef	UDINT	0	• Client side: ID of its own X.509-V3 certificate • Server side: ID of the X.509-V3 certificate (or a group of X.509-V3 certificates) that is used by the TLS server to validate the TLS client. If this parameter is 0, the TLS server uses all (CA) certificates currently loaded in the server certificate store to validate the client authentication.

See also

Connection parameters with structure according to TCON_IP_v4 (Page 625)

Secure OUC between two S7-1500 CPUs (Page 942)

Connection parameters in accordance with TCON_QDN (S7-1500)

Data block for connection description

To configure the communication connections for TCP and UDP over the fully qualified domain name, use a connection description DB with a structure according to TCON_QDN for S7-1500 CPUs as of firmware version V2.0.

The connection parameter CONNECT of the instance DBs for TCON contains a reference to the data block used.

Note

TCON_DNS structure

TCON_QDN replaces TCON_DNS. Therefore use only connection description DBs with a structure according to TCON_QDN.

Structure of the connection description in accordance with TCON_QDN

Byte	Parameter	Data type	Start value	Description
0 … 1	interface_id	HW_ANY	0	irrelevant Note: The first Industrial Ethernet interface is always used in the firmware version V2.0 of the S7-1500 CPUs.
2 … 3	id	CONN_OUC	1	Reference to this connection (value range: 1 to 4095). You must specify the value of this parameter for the TCON instruction under ID.
4	connection_type	BYTE	11	Connection type: • 11: TCP (11 dec = 0x0B hex) • 19: UDP (19 dec = 0x13 hex) For compatibility and migration reasons, the CPU S7-1500 also allows the values of the connection description DB with a structure according to TCON_Param. For the connection type TCP, the entry 17 is therefore also valid (17 dec = 0x11 hex).
5	active_established	BOOL	TRUE	Identifier for the type of connection establishment: • FALSE: Passive connection establishment • TRUE: Active connection establishment
6 … 261	remote_qdn	Array of STRING [1..254]	-	Fully qualified domain name of the partner end point, which must finish with "." Please note that in a SIMATIC network, the name including the concluding dot must not exceed 254 characters.
262 … 263	remote_port	UINT	2000	Port address of the remote connection partner (value range: 1 to 49151).
264 … 265	local_port	UINT	2000	Port address of the local connection partner (value range: 1 to 49151).

Connection parameters in accordance with TCON_QDN_SEC (S7-1500)

Data block for connection description

To configure the communication connections for TCP over the fully qualified domain name with secure communication, use a connection description DB with the structure of the SDT TCON_QDN_SEC for S7-1500 CPUs with firmware version V2.0 or higher.

Note

Non-secure TCP or UDP connection over the fully qualified domain name

You can also use the TCON_QDN_SEC SDT for a non-secure TCP or UDP connection over the fully qualified domain name.

The connection parameter CONNECT of the instance DBs for TCON contains a reference to the data block used.

Structure of the connection description according to TCON_QDN_SEC

Byte	Parameter	Data type	Start value	Description
0 … 271	ConnPara	TCON_QDN	-	Connection parameter
272	ActivateSecure-Conn	BOOL	FALSE	Activation of secure communication for this connection If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
273	TLSServerReq-ClientCert	BOOL	FALSE	Only for the server side: Request for an X.509-V3 certificate from the TLS client
274 .. . 275	ExtTLSCapabilities	WORD	16#0	• Bit 0: Only for the client side. A set bit means that the client validates the subjectAlternateName in the X.509-V3 certificate of the server to check the identity of the server. The certificates are checked when the connection is established. • Bit 1 to 15: reserved for future upgrades
276 .. . 279	TLSServerCertRef	UDINT	0	• Server side: ID of its own X.509-V3 certificate • Client side: ID of the X.509-V3 certificate (usually a CA certificate) that is used by the TLS client to validate the TLS server authentication. If this parameter is 0, the TLS client uses all (CA) certificates currently loaded in the client certificate store to validate the server authentication.
280 … 283	TLSClientCertRef	UDINT	0	• Client side: ID of its own X.509-V3 certificate • Server side: ID of the X.509-V3 certificate (or a group of X.509-V3 certificates) that is used by the TLS server to validate TLS client authentication. If this parameter is 0, the TLS server uses all (CA) certificates currently loaded in the server certificate store to validate the client authentication.

See also

Connection parameters in accordance with TCON_QDN (Page 628)

Secure OUC between two S7-1500 CPUs (Page 942)

Secure OUC from an S7-1500 CPU as a TLS client to an external PLC (TLS server) (Page 946)

Secure OUC from an S7-1500 CPU as TLS server to an external PLC (TLS client) (Page 949)

Secure OUC via e-mail (Page 955)

Connection parameters with structure according to TCON_IP_RFC (S7-1200, S7-1500)

Data block for connection description

A connection description DB with a structure according to TCON_IP_RFC is used for CPUs of S7-1200 V4.0 and higher and S7-1500 to assign parameters to ISO-on-TCP communication connections. The fixed data structure of the TCON_IP_RFC contains all parameters that are required to establish the connection. The connection description DB is automatically created for a new connection by the connection configuration for Open User Communication when the TSEND_C, TRCV_C or TCON instruction is used.

The CONNECT connection parameter of the instance DBs for TSEND_C, TRCV_C or TCON contains a reference to the data block used.

Structure of the connection description according to TCON_IP_RFC

Byte	Parameter	Data type	Start value	Description
0 … 1	interface_id	HW_ANY	64	Hardware identifier of the local interface (value range: 0 to 65535).
2 … 3	id	CONN_OUC	1	Reference to this connection (value range: 1 to 4095). You must specify the value of this parameter for the TSEND_C, TRCV_C or TCON instruction under ID.
4	connection_type	BYTE	12	Connection type 12: ISO-on-TCP (12 dec = 0x0C hex) For compatibility and migration reasons, the CPU S7-1500 also allows the values of the connection description DB with a structure according to TCON_Param. For the connection type ISO-on-TCP, the entry 18 is therefore also valid (18 dec = 0x12 hex).
5	active_established	BOOL	TRUE	Identifier for the type of connection establishment: • FALSE: Passive connection establishment • TRUE: Active connection establishment
8 … 11	remote_address	ARRAY [1..4] of BYTE	-	IP address of the partner end point, for example, for 192.168.0.1: • addr[1] = 192 • addr[2] = 168 • addr[3] = 0 • addr[4] = 1
12 … 45	remote_tselector	TSelector	-	TSelector of the remote connection partner: • TSelLength = Value range 0 to 32 as UINT • TSel[1-32] = Value range each 0 to 255 in bytes
46 … 79	local_tselector	TSelector	-	TSelector of the local connection partner: • TSelLength = Value range 0 to 32 as UINT • TSel[1-32] = Value range each 0 to 255 in bytes

See also

Principle of operation of connection-oriented protocols (Page 620)

Description of the connection parameters (Page 612)

Ability to read back connection description parameters (Page 638)

Basic principles for programming of data blocks (Page 4795)

Overview of connection configuration (Page 609)

TSAP structure (Page 639)

Connection parameters to TCON_Phone (S7-1200, S7-1500)

System data types TCON_... for telecontrol connections

TCON_PHONE is used as a connection description for the following types of connection:

- In mobile wireless CPs of the S7-1200 for connections of Open User Communication to transfer SMS over TCON / TDISCON / TSEND or TSEND_C.
- With the CP 1242-7 for telecontrol connections to SMS clients with TC_CON

To configure establishment of a connection using one of the previously mentioned instructions, the CONNECT parameter of the instruction is used for the connection description.

The connection description is defined by the structure of the system data type (SDT) TCON_Phone. The structure of the relevant SDT contains the parameters necessary to establish the connection with the remote communications partner.

Creating a DB of the type TCON_Phone

You need to type in the data type of the DB with the keyboard. It is not displayed in a selection list. Upper or lower case is unimportant when entering the data type.

Follow these steps to create the DB:

1. Create a data block of the type "Global DB" with block access "Standard".
   To do so, deactivate the property "Optimized access" in the block properties (shortcut menu) in the "Attributes" tab.
2. Create an SDT in the parameter configuration table of the DB.
3. Enter the following name in the declaration table of the DB in the cell of the data type:
   TCON_Phone
   The SDT and its parameters are created (see below).
4. Configure the parameters described below.

Reserved bits are not displayed.

System data type TCON_Phone for SMS connections

Note

SMS text

- Programmed SMS texts for SMS messages to be sent are accessed using the DATA parameter of the TC_SEND instruction.
- The text of a received SMS message is assigned to the address area of the CPU by the DATA parameter of the TC_RECV instruction.

Table 10-1 Parameters of TCON_Phone

Byte	Parameter	Data type	Initial value	Description
0 ... 1	InterfaceId	HW_ANY		Reference to the local interface (telecontrol interface or Ethernet interface of the CP > "Hardware identifier")
2 ... 3	ID	CONN_OUC	1...07FF$_h$	Reference to the local mobile wireless connection. The ID is assigned and must be unique within the CPU. Note on telecontrol connections of the CP 1242-7: The same value as that of the parameter ID of the TC_CON instruction must be used here.
4	ConnectionType	Byte	W#16#0E	Protocol variant 14 (E$_h$): SMS connection
5	ActiveEstablished	Bool		Identifier for the type of connection establishment (not relevant for the CP 1242-7): • 0: Passive connection establishment (not relevant here) • 1: Active connection establishment
6...7	-	-	-	- reserviert -
8 ... 31	PhoneNumber	String[22]		Call number of the connection partner Permitted values: Plus character (+) and numbers Note the exact notation of the international dialing code of the relevant phone number assigned by the network provider ("+" character or zeros). Note on the telecontrol connections of the CP 1242-7: if the parameter PhoneNumber is not entered, no connection partner is specified and SMS messages can be received by all authorized connection partners. Without an entry, TC_RECV initially delivers the oldest received SMS messages during startup.

Connection parameters to TCON_FDL (S7-1200, S7-1500)

System data block (SDT) for FDL connection description

In order to parameterize an FDL connection for the CM 1542-5, a connection description DB with an SDT to TCON_FDL is used. The fixed data structure of the STD contains all parameters that are required to establish the connection. The connection description SDT has to be created in a created DB.

The connection parameter CONNECT of the instance DBs for TSEND_C or TRCV_C contains a reference to the data block used.

The following types of FDL connections can be established:

- **Configured FDL connections**
  These connection types are configured in the STEP 7 program editor, see AUTOHOTSPOT. TSEND_C and TRCV_C are used as blocks.
  - Specified connection
    Fully configured connection between two masters
  - Unspecified connection
    Configured connection with an unspecified partner
- **Programmed FDL connections**
  These connection types cannot be configured in the STEP 7 program editor. Instead these connection types require the calling the TCON instruction together with TSEND/TRCV or TUSEND/TURCV, respectively.
  - Unspecified Layer 2 connection
    Programmed FDL connection with an unspecified partner with free Layer 2 access
  - Broadcast connection
    Connection with all connected partners
  - Multicast connection
    Connection with several defined partners

If specific settings for the individual connection types are required, this is mentioned at the subsequent parameters of the SDT.

Structure of the connection description "TCON_FDL" for configured connections

The table below describes the connection parameters for the following configured FDL connections:

- Specified connection
- Unspecified connection

Byte	Parameter	Data type	Value range / Start value	Description
0 … 1	InterfaceId	HW_ANY	0 .. 65535	Hardware identifier of the local interface
2 … 3	ID	CONN_OUC	0 .. 4095	Reference to this connection in accordance with connection configuration
4	ConnectionType	Byte	15	Connection type: 0x15 (dec. 21) for FDL
5	ActiveEstablished	Bool	FALSE	ID for the manner in which the connection is established • FALSE: Passive connection establishment • TRUE: Active connection establishment The parameter value is not processed further.

Byte	Parameter	Data type	Value range / Start value	Description
5	ServiceId	Byte	0 .. 2	Identifier of the data transfer service • 0: Automatic default with selection of the service in accordance with the receiver configured in the connection description Only SDA is used in this setting for configured connections. • 1: Send Data Acknowledged (SDA) - for active connections between two masters Use this setting for unspecified connections. • 2: Send Data Non-Acknowledged (SDN) - for multicast / broadcast
6 … 9	RemotePBAddress	Byte	0 .. 126	PROFIBUS address of the connection partner • Is used for specified connections by the connection configuration. • Has to be stated explicitly for unspecified connections.
6 … 9	LocalPBAddress	Byte	0 .. 126	Own (local) PROFIBUS address Is assigned automatically by the interface configuration.
10 … 11	RemoteLSAP	Byte	2 .. 32	LSAP (link layer service access point) of the connection partner Has to be stated explicitly for configured connections.
12 … 13	LocalLSAP	Byte	2 .. 32	Own (local) LSAP Has to be stated explicitly for configured connections.

Structure of the connection description "TCON_FDL" for programmed connections

The table below describes the connection parameters for the following programmed FDL connections:

- Unspecified Layer 2 connection
- Broadcast connection
- Multicast connection

Byte	Parameter	Data type	Value range / Start value	Description
0 … 1	InterfaceId	HW_ANY	0 .. 65535	Hardware identifier of the local interface
2 … 3	ID	CONN_OUC	0 .. 4095	Reference to this connection Reference to the value of the ID generated by TCON. Specify the value for TSEND / TRCV or TUSEND / TURCV at the parameter ID.
4	ConnectionType	Byte	15	Connection type: 0x15 (dec. 21) for FDL
5	ActiveEstablished	Bool	FALSE	ID for the manner in which the connection is established • FALSE: Passive connection establishment • TRUE: Active connection establishment The parameter value is not processed further.

Byte	Parameter	Data type	Value range / Start value	Description
5	ServiceId	Byte	0 .. 2	Identifier of the data transfer service • 0: Automatic selection of the service. Not to be used for programmed connections. • 1: Send Data Acknowledged (SDA) - for active connections between two masters • 2: Send Data Non-Acknowledged (SDN) - for multicast / broadcast "1" or "2" has to be programmed for programmed connections.
6 … 9	RemotePBAddress	Byte	0 .. 127	PROFIBUS address of the connection partner Specify the address if TSEND or TUSEND is used. The value is set by the user program for TRCV or TURCV. • For broadcast / multicast: 127
6 … 9	LocalPBAddress	Byte	0 .. 126	Own (local) PROFIBUS address Is assigned automatically by the interface configuration.
10 … 11	RemoteLSAP	Byte	2 .. 32, 63	LSAP (link layer service access point) of the connection partner Specify the address if TSEND or TUSEND is used: • For unspecified Layer 2 connections: 2 .. 32 • For broadcast: 63 • For multicast: Use the same values as "LocalLSAP". The value is set by the user program for TRCV or TURCV.
12 … 13	LocalLSAP	Byte	2 .. 32	Own (local) LSAP Is to be assigned by the program.

Assignment of port numbers (S7-1200, S7-1500)

Introduction

When an Open User Communication is created, the value 2000 is automatically assigned as the port number.

Permissible values for port numbers are 1 to 49151. You can assign any port number within this range. However, because some ports may already be used depending on the system, port numbers within the range from 2000 to 5000 are recommended.

Note

Port numbers must be unique. The connection configuration or a corresponding block call is rejected with an error if the port numbers are assigned twice.

Overview of port numbers

The following table summarizes the system reactions to different port numbers.

Port no.	Description	System reaction
2000 … 5000	Recommended range	No warning, no error message on entry Port number is permitted and accepted
1 … 1999, 5001 … 49151	Can be used, but is outside the recommended range	Warning message on entry Port number is permitted and accepted
0, 20, 21, 25, 80, 102, 135, 161, 443, 465, 989, 990, 8080, 34962 … 34964	Can be used conditionally*	
53, 80, 102, 135, 161, 162, 443, 520, 9001, 34962 … 34964	Can be used conditionally**	

* Ports specified for specific functions ("well-known Ports"):
0: ANY - Port number is automatically assigned by S7-1500 CPU as value (>49151)
20: FTP data transmission
21: FTP control
25: SMTP (Simple Mail Transfer Protocol) / TMAIL_C
80: HTTP (Hypertext Transfer Protocol) / Web server
102: ISO-on-TCP (RFC1006)
135: DCE Endpoint Mapper for PROFINET
161: SNMP (Simple Network Management Protocol)
443: HTTPS (Hypertext Transfer Protocol Secure)
465: SMTPS (Simple Mail Transfer Protocol Secure)
989: FTPS data transfer
990: FTPS control
8080: HTTP alternative / internal Web
34962 ... 34964: PROFINET

Note

The user usually specifies the value 0 for the local port on the active connection end point for UDP/TCP. In this case, the CPU operating system selects the next available port above 49151. The partner port usually has the default 0 with the passive connection end point. The corresponding parameter is disabled in the connection configuration.

** These ports are disabled depending on the function scope of the CPU in use. The documentation of the respective CPUs provides the assignment of these ports.

See also

Description of the connection parameters (Page 612)

Creating and assigning parameters to connections (Page 616)

Ability to read back connection description parameters (S7-1200, S7-1500)

Changing parameter values in the connection description

The connection description for exactly one connection of the Open User Communication is entered from the connection configuration in the connection description DB.

You can change the parameter values of the connection description DB outside of the connection configuration in the user program. Connection description DBs containing values you changed subsequently can be read back from the connection configuration. Under "Properties > Configuration > Connection parameters", the inspector window displays only the connection parameters stored in the connection description DB.

Note

You can only change the values in the running user program if the instructions TCON, TSEND_C or TRCV_C are not being processed and the referenced connection is not established.

The connection configuration does not support nested entries of connection descriptions in DB types that can only be found via offset referencing (for example, Global-DB).

The structure of the connection description cannot be changed.

Ability to read back individual connection parameters

For the "Address" parameter of the communication partner in a TCP or ISO-on-TCP connection, its IP address is displayed from the "rem_staddr" parameter of the connection description.

The following values can also be reloaded from the connection description:

- Connection type
- Local connection ID
- Active/passive connection establishment (only with UDP)
- Local TSAP (ISO-on-TCP only)
- Partner TSAP (ISO-on-TCP only)
- Local port (only with TCP and UDP)
- Partner port (only with TCP)

The values of the connection ID parameters of the communication partner, the connection data, as well as the connection establishment, are not included in the connection description in the local connection description DB. Consequently, these parameters cannot be displayed when the connection configuration is reopened. The connection establishment of the partner results from the local connection establishment and is therefore also displayed.

A new communication partner can be selected at any time in the "Partners" drop-down list box.

When a CPU recognized in the project is selected as a specified communication partner, the entry options for the connection ID and the connection data are shown again.

See also

Connection parameters with structure according to TCON_Param (Page 622)

Description of the connection parameters (Page 612)

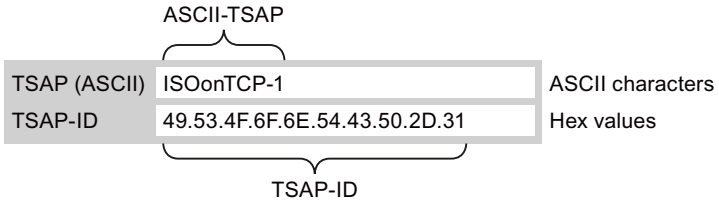
TSAP structure (S7-1200, S7-1500)

Introduction

For an ISO-on-TCP connection, Transport Service Access Points (TSAPs) must be assigned for both communication partners. TSAP IDs are assigned automatically after an ISO-on-TCP connection is created. To ensure the uniqueness of TSAP IDs within a device, you can change the preassigned TSAPs in the connection parameter assignment.

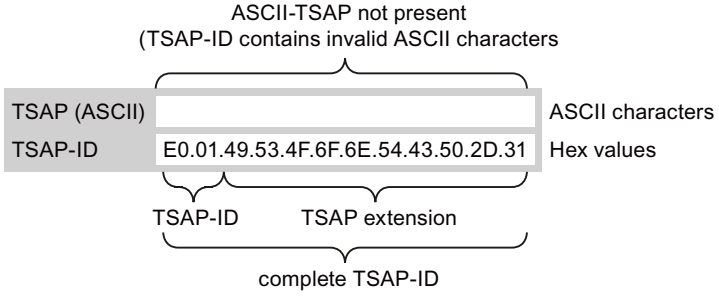
Structure of TSAPs

You must comply with certain rules when assigning TSAPs. A TSAP must contain a certain number of bytes, which are able to be displayed and entered as hexadecimal values (TSAP-ID) or as ASCII characters (ASCII-TSAP):



Entries or changes of the TSAP-ID or the ASCII-TSAP in the corresponding entry fields always take effect in the other display format as well.

If a TSAP contains no valid ASCII characters, the TSAP is displayed only as TSAP-ID and not as ASCII-TSAP. This is the case after a connection is created. The first two hex characters as TSAP-ID identify the communication type and the rack/slot. Because these characters are not valid ASCII characters for a CPU, the ASCII-TSAP is not displayed in this case.



In addition to the rules for length and structure of TSAPs, you must also ensure the uniqueness of the TSAP-ID. The assigned TSAPs are not automatically unique.

Length and content of TSAPs

Structure of the TSAP ID with TSAP extension:

- Valid for CPU S7-1200 firmware V1
  Length = 2 to 16 bytes
  x_tsap_id[0] = 0xE0 (Open User Communication)
  x_tsap_id[1] (bits 0 to 4) = slot number of CPU
  x_tsap_id[1] (bits 5 to 7) = rack number of CPU
  x_tsap_id[2...15] = any characters (TSAP extension, optional)
  (x = loc (local) or x = rem (partner))
- Valid for CPU S7-1500 and S7-1200 firmware as of V2
  Length = 2 to 16 bytes
  x_tsap_id[0] = 0xE0 (Open User Communication)
  x_tsap_id[1] = 0x00 to 0xFF
  x_tsap_id[2...15] = any characters (TSAP extension, optional)
  (x = loc (local) or x = rem (partner))

Structure of the TSAP ID as ASCII TSAP:

- Valid for CPU S7-1200 firmware V1
  Length = 3 to 16 bytes
  x_tsap_id[0...2] = 3 ASCII characters (0x20 to 0x7E)
  x_tsap_id[3...15] = any characters (optional)
  (x = loc (local) or x = rem (partner))
- Valid for CPU S7-1200 firmware as of V2
  Length = 3 to 16 bytes
  x_tsap_id[0...2] for active connection = 3 ASCII characters (0x00 to 0xFF) or any bit string*
  x_tsap_id[0...2] for passive connection = 3 ASCII characters (0x20 to 0x7E) or any bit string*
  x_tsap_id[3...15] = any characters (optional)
  (x = loc (local) or x = rem (partner))
- Valid for S7-1500 CPU
  Length = 3 to 16 bytes
  x_tsap_id[0...2] = 3 ASCII characters (0x00 to 0xFF) or any bit string*
  x_tsap_id[3...15] = any characters (optional)
  (x = loc (local) or x = rem (partner))

* ASCII character strings must not start with "SIMATIC-"

The following table shows the schematic structure of various TSAP IDs:

TSAP-ID	tsap_id_len	tsap_id[0]	tsap_id[1]	tsap_id[2]	tsap_id[3..15]
...with extension (CPU S7-1200 firmware V1)	2...16 bytes	0xE0	0x01 or 0x02 or 0x00*	Extension (optional)	Extension (optional)
...with extension (CPU S7-1500, S7-1200 firmware as of V2)	2...16 bytes	0xE0	0x00...0xFF	Extension (optional)	Extension (optional)
...as ASCII-TSAP (CPU S7-1200 firmware V1)	3...16 bytes	0x20...0x7E	0x20...0x7E	0x20...0x7	Any (optional)

*An S7-1200 CPU is normally inserted on rack 0 and slot 1, and an S7-300/400 CPU on rack 0 and slot 2. For this reason, hex value 01 or 02 is valid for the second position of the TSAP ID with extension. If the connection partner is an unspecified CPU, for example, a third-party device, the hex value 00 is also permissible for the slot address.

Note

For unspecified communication partners, the local TSAP-ID and the partner TSAP-ID can have a length of 0 to 16 bytes, in which all hex values from 00 to FF are permitted.

ASCII code table for entry of ASCII TSAPs

When entering ASCII TSAPs with the hexadecimal values from 20 to 7E, only the following characters are permitted:

Code	..0	..1	..2	..3	..4	..5	..6	..7	..8	..9	..A	..B	..C	..D	..E	..F
2..		!	"	#	$	%	&	’	(	)	*	+	,	-	.	/
3..	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4..	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5..	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6..	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7..	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	

See also

Examples of TSAP assignment (Page 641)

Description of the connection parameters (Page 612)

Creating and assigning parameters to connections (Page 616)

Examples of TSAP assignment (S7-1200, S7-1500)

The following examples show the processing of the TSAPs for CPUs of the S7-1200/1500 (CPU on slot 1) under various points of view:

- Example 1: Creating a new connection for PLC-PLC communication
- Example 2: Entry of a local ASCII-TSAP
- Example 3: Entry of a TSAP extension in the TSAP-ID
- Example 4: Incorrect editing of the TSAP-ID
- Example 5: Entry of an ASCII-TSAP via the "TSAP-ID" entry field

Example 1: Creating a new connection for PLC-PLC communication

Once you have created a new connection with two PLCs for the Open User Communication, the TSAP extension "ISOonTCP-1" is assigned automatically.

This TSAP extension produces the TSAP-ID E0.01.49.53.4F.6F.6E.54.43.50.2D.31, which is entered automatically in the connection description DB and in the entry fields of the local and the partner TSAP. The entry fields of the ASCII-TSAPs remain empty:

	Local TSAP	Partner TSAP
TSAP (ASCII)		
TSAP-ID	E0.01.49.53.4F.6F.6E.54.43.50.2D.31	E0.01.49.53.4F.6F.6E.54.43.50.2D.31

You can change the values in the entry fields of the TSAP-ID and the ASCII-TSAP at any time.

The entry field of the TSAP-ID shows the complete TSAP stored in the data block of the connection description. The TSAP-ID with TSAP extension, which is limited to 16 characters, is not displayed in the "TSAP (ASCII)" entry field because the character E0 does not represent a valid character for the ASCII-TSAP.

If the displayed TSAP-ID is a valid ASCII-TSAP, it is displayed in the "TSAP (ASCII)" entry field.

Changes in the entry fields for TSAP-ID and ASCII-TSAP affect the other field.

Example 2: Entry of a local ASCII-TSAP

If you have created a new connection and assigned an ASCII value for the local TSAP in the "TSAP (ASCII)" entry field, for example, "ISOonTCP-1", the resulting TSAP-ID is created automatically.

When you exit the "TSAP (ASCII)" entry field, the number of ASCII characters is checked automatically for compliance with the limit (3 to 16 characters) and the resulting TSAP-ID is entered into the corresponding entry field:

	Local TSAP	Partner TSAP
TSAP (ASCII)	ISOonTCP-1	
TSAP-ID	49.53.4F.6F.6E.54.43.50.2D.31	E0.01.49.53.4F.6F.6E.54.43.50.2D.31

Example 3: Entry of a TSAP extension in the TSAP-ID

If, following creation of a connection and entry of an ASCII-TSAP (see examples 1 and 2) in the entry field of the local TSAP-ID, you add the prefix "E0.01" to the TSAP value, the ASCII-TSAP will no longer be displayed when the entry field is exited.

	Local TSAP	Partner TSAP
TSAP (ASCII)		
TSAP-ID	E0.01.49.53.4F.6F.6E.54.43.50.2D.31	E0.01.49.53.4F.6F.6E.54.43.50.2D.31

Once you have exited the entry field of the TSAP-ID, a check is performed automatically to determine whether the first character of the TSAP-ID is a valid ASCII character. Since the character "E0" now present in the TSAP-ID is not a valid character for the ASCII-TSAP, the "TSAP (ASCII)" entry field no longer displays an ASCII-TSAP.

If a valid ASCII character is used, the check for compliance with the length specification of 2 to 16 characters follows.

Example 4: Incorrect editing of the TSAP-ID

If you remove the hex value "E0" from a TSAP-ID beginning with "E0.01", the TSAP-ID now begins with "01" and therefore no longer complies with the rules and is invalid:

	Local TSAP	Partner TSAP
TSAP (ASCII)		
TSAP-ID	01.49.53.4F.6F.6E.54.43.50.2D.31	E0.01.49.53.4F.6F.6E.54.43.50.2D.31

After the entry field is exited, a message is output because the TSAP-ID is neither a valid ASCII-TSAP (this would have to have a hex value in the range from 20 to 7E as the first value) or a valid TSAP-ID (this would have to have the identifier "E0" as the first value).

Example 5: Entry of an ASCII-TSAP via the "TSAP-ID" entry field

If you remove the value "01" in addition to the value "E0" from the incorrect TSAP-ID in example 4, the TSAP-ID begins with the hex value 49. This value is within the permissible range for ASCII-TSAPs:

	Local TSAP	Partner TSAP
TSAP (ASCII)		
TSAP-ID	49.53.4F.6F.6E.54.43.50.2D.31	E0.01.49.53.4F.6F.6E.54.43.50.2D.31

When you exit the entry field, the TSAP-ID is recognized as a valid ASCII-TSAP and the resulting ASCII-TSAP "ISOonTCP-1" is written to the "TSAP (ASCII)" entry field.

See also

TSAP structure (Page 639)

Description of the connection parameters (Page 612)

Configuring HMI connections

Introduction to configuring connections

Definition

A connection defines a logical assignment of two communication partners in order to undertake communication services. A connection defines the following:

- Communication partner involved
- Type of connection (e.g., HMI connection)
- Special properties (e.g., whether a connection is established permanently or whether it is established and terminated dynamically in the user program, and whether status messages are to be sent)
- Connection path

What you need to know about connection configuration

During connection configuration, a local connection name is assigned for an HMI connection as a unique local identification.

In the network view, a "Connections" tab is displayed in addition to the "Network overview" tab. This tab contains the connection table. A line in this connection table represents a configured connection, e.g., between an HMI device and PLC, along with its properties.

What you need to know about using connection resources

Introduction

Each connection requires connection resources for the end point and/or transition point on the devices involved. The number of connection resources is device-specific.

If all the connection resources of a communication partner are assigned, no new connections can be established. This situation is apparent when a newly created connection in the connection table has a red background. The configuration is then inconsistent and cannot be compiled.

HMI connections

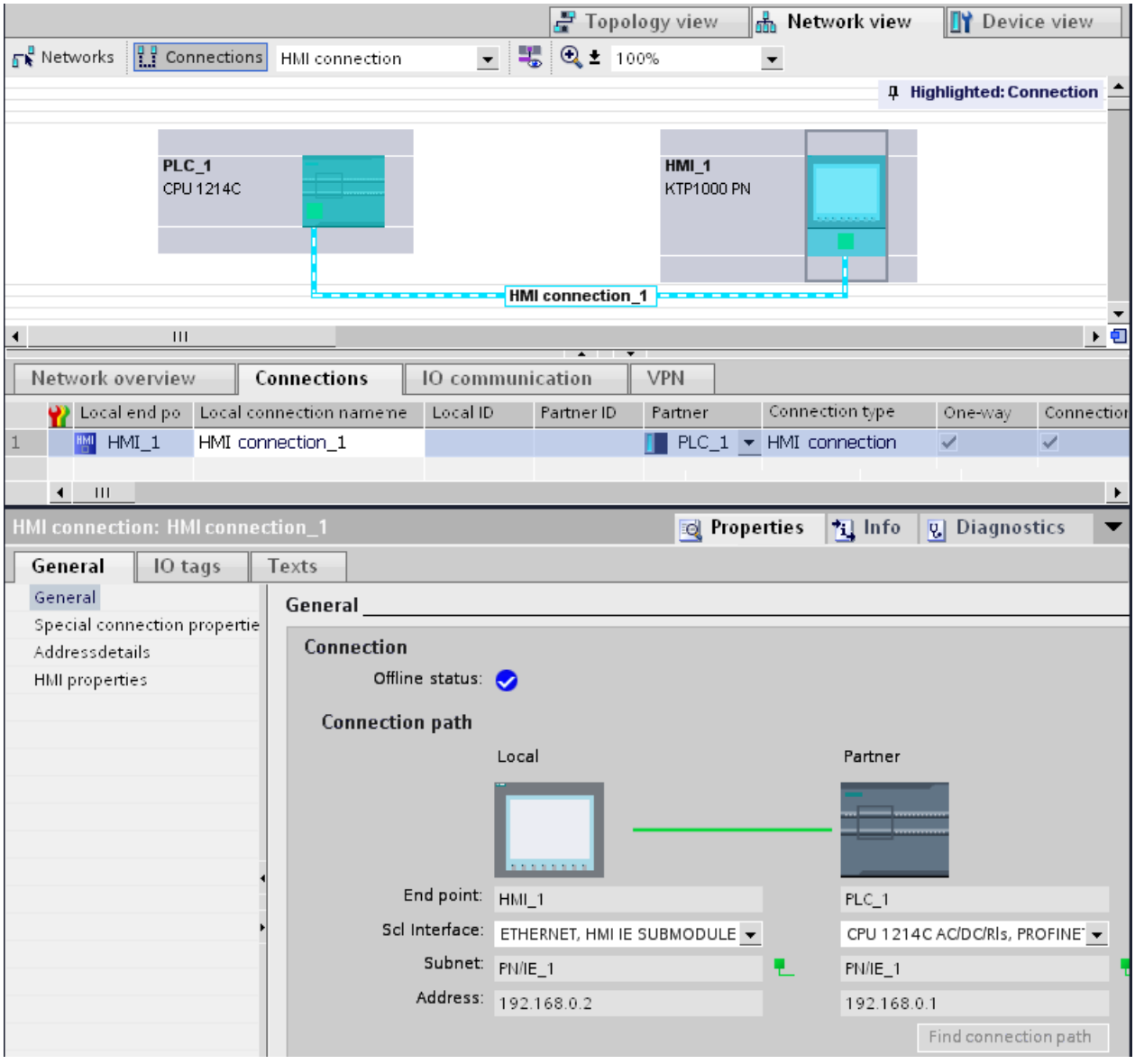
For HMI connections via the integrated PN interface, one connection resource for the endpoint per HMI connection is occupied for the HMI device.

One connection resource is also required for the connection partner (PLC).

Views containing information about the configured connections

The views described below will provide you with comprehensive access to all the information and functions regarding configuring and checking communication connections.

- Connection display in the network view
- Connection table
- "Properties" tab for a connection in the inspector window



Benefits

The information shown in these views are always up-to-date in terms of the current user actions. This means:

- The connection table displays all connections created.
- If you have selected a connection in the connection table:
  - You will graphically see the connection path in the network view.
  - The "Properties" tab in the Inspector window displays the parameters of this connection.

The connection table

The connection table offers the following functions:

- List of all connections in the project
- Selection of a connection and display of connections associated with it in the network view
- Changing of connection partners
- Display showing status information

"Properties" tab for a connection in the inspector window

The properties dialog has the following meaning:

- Display for connection parameters
- Display of connection path
- Subsequent specification of connections using the "Find connection path" button

Creating a new connection

Creating a connection - alternatives

You have the following options for creating a connection in the network view:

- Graphic connection configuration
- Interactive connection configuration

You'll find the individual steps for this in the following chapters.

Requirement and result

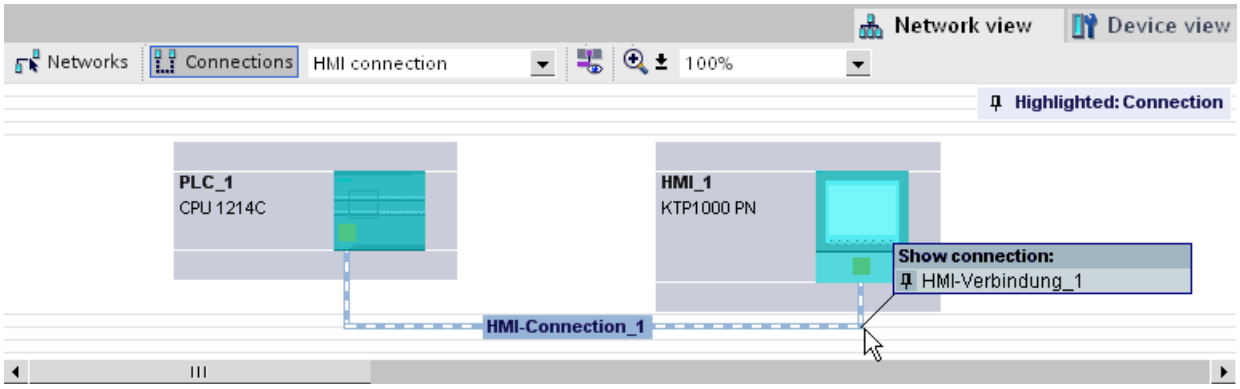
You have created the devices with CPUs and HMI devices between which you want to configure connections in the network view.

Specifying a connection

If both partners for the connection type selected are networked on the same network, use the graphic or interactive selection of both communication partners to create a fully specified connection.

This connection is entered automatically into the connection table of the HMI device. A local connection name is assigned for this connection.

The following schematic shows a configured connection with a networked device:



Creating a new connection graphically

Graphically configuring connections

When using the graphic connection configuration, if necessary the system asks you to define the connection path. Select the devices to be connected in the current configuration.

Automatically determining connection path

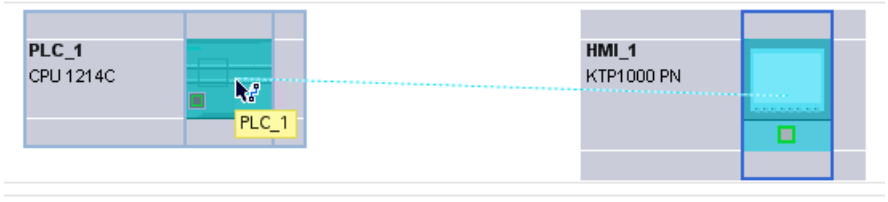
To create a connection graphically, follow these steps:

1. Click the "Connections" button.

   The connection mode for the connection type you have selected is then activated.
   You will see this from the following:
   The devices that can be used for the connection type selected in your project are color-highlighted in the network view.

2. Hold down the mouse button and drag the mouse pointer from the device from which the connection will originate to the device at which the connection ends.

3. Release the mouse button over the destination device to create the connection between the two devices.

Result

- A specified connection is created.
- The connection path is highlighted.
- The connection is entered in the connection table.

Interactively creating a new connection

Interactively configuring connections

Define the local device and its connection partners.

Procedure

Proceed as follows to interactively create a connection:

1. Select the "Create new connection" command in the shortcut menu of a connection partner for which you want to create a connection.
   The "Create new connection" dialog is opened.
2. Select the partner endpoint.
   In the right pane of the dialog, a possible connection path fitting the selected endpoint is displayed, if available. Incomplete paths, for example, for a non-specified CPU, are marked by an exclamation mark on a red background.
3. To close the dialog, click "OK".
   To accept the configured connection and to configure additional connections to other endpoints, click "Apply".

Working in the network view

Highlighting connection path and partner in the network view

To display the connection partners for all or certain connection types in the network view, proceed as follows:

1. Click the "Connections" button.

2. Select the "Highlight connection partners" command in the shortcut menu for the HMI device whose connection partners you want to display in the network view.
3. Select "All connection partners" in the following menu.
   The local device and the CPUs of the target devices are selected. The local connection partner shows an arrow pointing right and the remote connection partners show an arrow pointing left.
4. To open a list with information on the target devices, click the arrow of the local device. This additional function is useful in complex network configurations in which some devices are not visible.



Note

You can display one of the connection partners which cannot be seen in the current display range of the network view. Click on the communication partner in the list that appears. Result: The display is moved such that the connection partner becomes visible.

See also

Creating a new connection graphically (Page 647)

Working with the connection table

Basic functions for tables

The connection table supports the following basic functions for editing a table:

- Changing column width
- Displaying the meaning of a column, a row or cell using tooltips.

Changing column width

To adjust the width of a column to the content so that all texts in the lines are legible, follow these steps:

1. Position the cursor in the header of the connection table to the right of the column that you want to optimize until the cursor changes its shape to two parallel lines (as if you wanted to change the width of the column by dragging it with the cursor).
2. Double click on this point.

or

1. Open the shortcut menu on the header of the table.
2. Click on
   - "Optimize column width" or
   - "Optimize width of all columns".

For columns that are too narrow, the complete content of specific fields is shown when you pause with the cursor on the respective field.

Show / hide columns

You can use the shortcut menu of the header of the connection table to control the display of the various table columns. The shortcut menu entry "Show/hide columns" provides you with an overview of the available columns. Use the check box to control whether columns are shown or hidden.

Using cursor keys to move within the connection table

You can use the UP and DOWN cursor keys to select a connection from the connection table; the selected connection is marked and is shown highlighted in the network view.

Changing properties of connection

You can directly edit the parameters displayed in the connection table in some cases. To change the name of a connection, you do not have to navigate to the Inspector window.

Changing connection partners

You can change the connection partner of a connection as follows:

1. Select the connection.
2. Select the new connection partner from the open drop-down list in the "Partner" column.

Deleting connections

You can delete configured connections using the network view or the connection table.

In the network view you can delete one highlighted connection per action. In the connection table you can delete one or several connections per action.

Procedure

To delete a connection, follow these steps:

1. Select the connection to be deleted:
   - In the network view: Select the connection to be deleted.
   - In the connection table: Select the rows of the connections to be deleted (multiple selection possible).
2. Open the shortcut menu with a right mouse click.
3. Select the "Delete" command.

Result

The selected connection is removed completely.

Copying connections

Introduction

Connections are not copied singly but always in context along with the project or the device.

You can copy:

- Entire projects
- One or more devices within a project or from one project to another

Copying a project

When you copy a project all configured connections will also be copied. No settings whatsoever are required for the copied connections because the connections remain consistent.

Copying devices

If you copy devices for which connections have been configured (HMI devices), the connections are copied as well. To complete the connection path, you must still finalize the networking.

An S7-1200 CPU with a V.10 firmware is only a server for connections and has no connection configuration itself. Consequently, no connections are copied along with it when an S7-1200 CPU with a V1.0 firmware is copied.

Inconsistent connections - connections without assignment

With an inconsistent connection the structure of the connection data is destroyed or the connection is not functional in the project context.

Inconsistent connections cannot be compiled and loaded - operation is not possible with such a connection.

In the connection table inconsistent connections are marked in red.

Possible causes for inconsistent connections

- Deletion or change of the hardware configuration.
- Missing interface network links in the project, which are necessary for a connection.
- Connection resources are exceeded
- Errors when backing up data due to insufficient memory
- Connections to an unspecified connection partner without partner address information.

Detailed information regarding the reasons for the inconsistency can be found in the "Compile" tab following compilation (Edit > Compile).

Remedies

If the connection cannot be repaired by opening the connection properties, changing them or undoing them in the configuration, then it may be necessary to delete the connection and re-create it.

HMI connection general settings

General connection parameters

General connection parameters are displayed in the "General" parameter group under the properties of the connection; these connection parameters identify the local connection end point.

Here, you can also assign the connection path and specify all aspects of the connection partner.

Special connection properties

Display of the connection properties (cannot be changed):

- Active connection establishment
  The connection establishment always starts from the HMI device. This option is selected by default if the address of the partner is specified.
- One-way
  One-way means that the connection partner functions as a server on this connection and cannot send or receive actively.
- Sending operating mode messages
  Not relevant for HMI devices.

Address details

Displaying address details of the HMI connection. With an unspecified partner, the values for the rack and slot can be changed. All other values are obtained from the current configuration and cannot be changed.

Miscellaneous

Display of the access points for the online connection between HMI device and connection partner.

Configuring S7 connections (S7-1200)

Introduction to configuring connections (S7-1200)

Definition

A connection defines a logical assignment of two communication partners in order to undertake communication services. A connection defines the following:

- Communication partner involved
- Type of connection (for example, S7 connection)
- Special properties (e.g., whether a connection is established permanently or whether it is established and terminated dynamically in the user program, and whether status messages are to be sent)
- Connection path

What you need to know about connection configuration

During connection configuration, a local connection name is assigned for an S7 connection as a unique local identification.

In the network view, a "Connections" tab is displayed in addition to the "Network overview" tab. This tab contains the connection table. A row in this connection table represents a

configured connection from the viewpoint of the local communication partner with its properties, for example, between two S7-1200 CPUs.

What you need to know about using connection resources (S7-1200)

Introduction

Each connection requires connection resources for the end point and/or transition point on the devices involved. The number of connection resources is device-specific.

If all the connection resources of a communication partner are assigned, no new connections can be established. This situation is apparent when a newly created connection in the connection table has a red background. The configuration is then inconsistent and cannot be compiled.

S7 connections

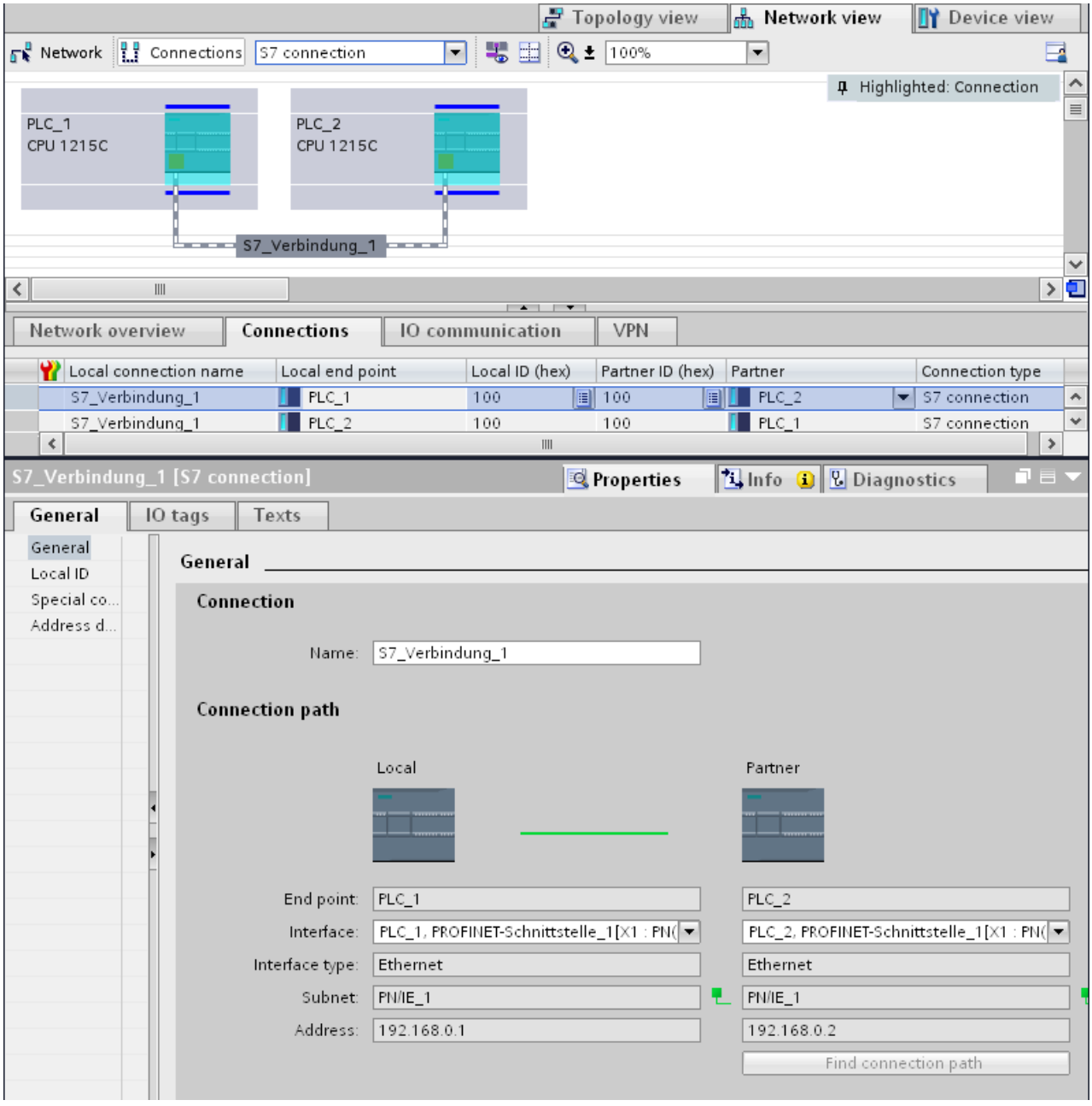
In the case of S7 connections via the PN interface, one connection resource per S7 connection is assigned for the endpoint for the S7-1200 CPU. One connection resource is also required for the connection partner.

You can find an overview of available and assigned connection resources for selected S7-1200 CPU in the Inspector window at "Properties > Connection Resources"

Views containing information about the configured connections (S7-1200)

The views described below will provide you with comprehensive access to all the information and functions regarding configuring and checking communication connections.

- Connection display in the network view
- Connection table
- "Properties" tab for a connection in the inspector window



Benefits

The information shown in these views are always up-to-date in terms of the current user actions. This means:

- The connection table displays all connections created.
- If you have selected a connection in the connection table:
  - When connection mode is enabled, the connection path is highlighted in the network view.
  - The "Properties" tab in the Inspector window displays the parameters of this connection.

The connection table

The connection table offers the following functions:

- List of all connections in the project
- Selection of a connection and display of connections associated with it in the network view (when connection mode is enabled)
- Changing of connection partners
- Display showing status information

"Properties" tab for a connection in the inspector window

The properties dialog has the following meaning:

- Display for connection parameters
- Display of connection path
- Subsequent specification of connections using the "Find connection path" button

Creating a new connection (S7-1200)

Creating a connection - alternatives

You have the following options for creating a connection in the network view:

- Graphic connection configuration
- Interactive connection configuration

You'll find the individual steps for this in the following chapters.

Requirement and result

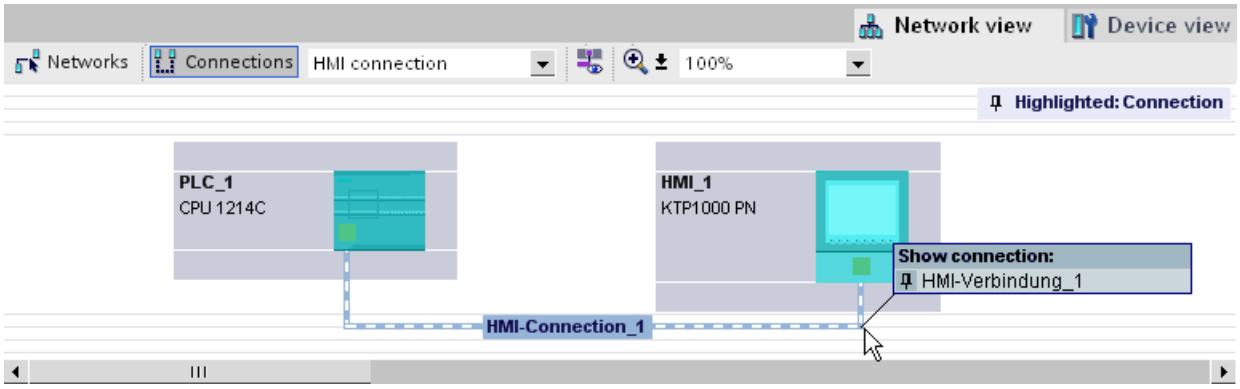
In the network view, you have add devices between which the connections should be configured.

Specifying a connection

If both partners for the connection type selected are networked on the same network, use the graphic or interactive selection of both communication partners to create a fully specified connection.

This connection is entered automatically in the connection table of the S7-1200 CPU. A local connection name is assigned for this connection.

The following schematic shows a configured connection with a networked device:



Creating a new connection graphically (S7-1200)

Graphically configuring connections

In the case of graphic connection configuration, the connection path is automatically specified provided interfaces and resources are available. Select the devices to be connected in the current configuration.

Automatically determining connection path

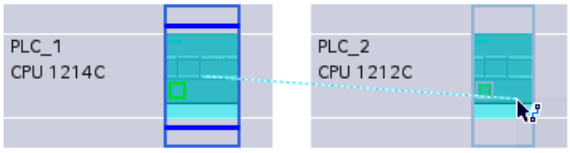
To create a connection graphically, follow these steps:

1. Click the "Connections" button.

   This step activates the connection mode: You can now select the connection type you want. You will see this from the following:
   The devices that can be used for the connection type selected in your project are color-highlighted in the network view.

2. Hold down the mouse button and drag the mouse pointer from the device from which the connection will originate to the device at which the connection ends.

3. Release the mouse button over the destination device to create the connection between the two devices.

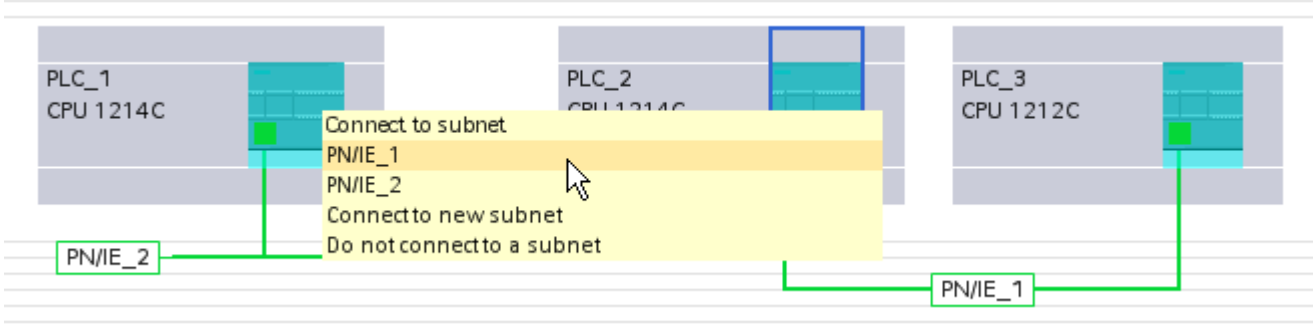
Result

- A specified connection is created.
- The connection path is highlighted.
- The connection is entered in the connection table.

Configuring a connection when there is no or no clear network assignment

Any networking not in place will if possible be created automatically when a connection is created. A query will be made upon completion of connection configuration if unique network assignment is not possible. You will be able to choose from the existing subnets.

Example below: A query is made upon creation of a connection between stations PLC_1 and PLC_2, which are not yet networked.



Interactively creating a new connection (S7-1200)

Interactively configuring connections

Define the local device and its connection partners.

Procedure

Proceed as follows to interactively create a connection:

1. Select the "Add new connection" command in the shortcut menu of a connection partner for which you want to create a connection.
   The "Create new connection" dialog appears.
2. Select the partner endpoint.
   In the right pane of the dialog, a possible connection path fitting the selected endpoint is displayed, if available. Incomplete paths, for example, for a non-specified CPU, are marked by an exclamation mark on a red background.
3. To accept the configured connection and to configure additional connections to other endpoints, click "Add".
   To close the dialog, click "OK".

Working in the network view (S7-1200)

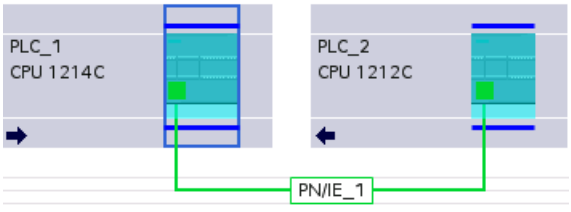
Highlighting connection path and partner in the network view

To display the connection partners for all or certain connection types in the network view, proceed as follows:

1. Click the "Connections" button.

2. Select the S7-CPU for which you want to display the connection partners in the network view and then select the "Highlight connection partners" command in the shortcut menu.
3. Select "All connection partners" in the following menu.
   The local device and the CPUs of the target devices are selected. The local connection partner shows an arrow pointing right and the remote connection partners show an arrow pointing left.
4. To open a list with information on the target devices, click the arrow of the local device. This additional function is useful in complex network configurations in which some devices are not visible.



Note

You can display one of the connection partners which cannot be seen in the current display range of the network view. Click on the communication partner in the list that appears. Result: The display is moved such that the connection partner becomes visible.

Working with the connection table (S7-1200)

Basic functions for tables

The connection table supports the following basic functions for editing a table:

- Changing column width
- Displaying the meaning of a column, a row or cell using tooltips.

Changing column width

To adjust the width of a column to the content so that all texts in the lines are legible, follow these steps:

1. Position the cursor in the header of the connection table to the right of the column that you want to optimize until the cursor changes its shape to two parallel lines (as if you wanted to change the width of the column by dragging it with the cursor).
2. Double click on this point.

or

1. Open the shortcut menu on the header of the table.
2. Click on
   - "Optimize column width" or
   - "Optimize width of all columns".

For columns that are too narrow, the complete content of specific fields is shown when you pause with the cursor on the respective field.

Show / hide columns

You can use the shortcut menu of the header of the connection table to control the display of the various table columns. The shortcut menu entry "Show/hide columns" provides you with an overview of the available columns. Use the check box to control whether columns are shown or hidden.

If you want to store the layout, width and visibility of the table columns, click on the "Remember layout" function in the top right-hand of the network view.

Symbol	Meaning
	Remember layout Saves the current table view. The layout, width and visibility of columns in the table view is stored.

Using cursor keys to move within the connection table

You can use the UP and DOWN cursor keys to select a connection from the connection table; the selected connection is marked and is shown highlighted in the network view.

Changing properties of connection

You can directly edit some of the parameters displayed in the connection table. The name of the connection can, for example, only be changed in the connection table.

Changing connection partners

You can change the connection partner of a connection as follows:

1. Select the connection.
2. Select the new connection partner from the open drop-down list in the "Partner" column.

Deleting connections (S7-1200)

You can delete configured connections using the network view or the connection table.

In the network view you can delete one highlighted connection per action. In the connection table you can delete one or several connections per action.

Procedure

To delete a connection, follow these steps:

1. Select the connection to be deleted:
   - In the network view: Select the connection to be deleted.
   - In the connection table: Select the rows of the connections to be deleted (multiple selection possible).
2. Open the shortcut menu with a right mouse click.
3. Select the "Delete" command.

Result

The selected connection is removed completely.

Copying connections (S7-1200)

Introduction

Connections are not copied singly but always in context along with the project or the device.

You can copy:

- Entire projects
- One or more devices within a project or from one project to another

Copying a project

When you copy a project all configured connections will also be copied. No settings whatsoever are required for the copied connections because the connections remain consistent.

Copying devices

If you copy devices for which connections have been configured, the connections are copied as well. To complete the connection path, you must still finalize the networking.

An S7-1200 CPU with a V.10 firmware is merely a server for connections and has no connection configuration itself. Consequently, no connections are copied along with it when an S7-1200 CPU with a V1.0 firmware is copied.

Inconsistent connections - connections without assignment (S7-1200)

With an inconsistent connection the structure of the connection data is destroyed or the connection is not functional in the project context.

Inconsistent connections cannot be compiled and loaded - operation is not possible with such a connection.

In the connection table inconsistent connections are marked in red.

Possible causes for inconsistent connections

- Deletion or change of the hardware configuration.
- Missing interface network links in the project, which are necessary for a connection.
- Connection resources are exceeded
- Connections to an unspecified connection partner without partner address information.

Detailed information regarding the reasons for the inconsistency can be found in the "Compile" tab following compilation (Edit > Compile).

Remedies

To assign a closed connection path to an existing open connection path, expand the device configuration in such a way that the interfaces required for the connection type are available for both partners. At "Properties > General > Interface" in the Inspector window, you can use the "Find connection path" button to create a connection to an existing partner.

S7 connection - general settings (S7-1200)

General connection parameters

General connection parameters are displayed in the "General" parameter group under the properties of the connection; these connection parameters identify the local connection end point.

Here, you can assign the connection path and specify all aspects of the connection partner.

Local ID

The local ID of the module from which the connection is viewed is displayed here (local partner). You can change the local ID. You may need to do this if you have already programmed communication function blocks, and you want to use the local ID specified in those function blocks for the connection.

Special connection properties

Display of connection properties (can be modified depending on the components used):

- One-way
  One-way means that the connection partner functions as a server on this connection and cannot send or receive actively.
- Active connection establishment
  In the case of one-way connection, for example with a S7-1200 CPU (firmware version V1.0), a connection partner can only be provided for the active connection establishment. In the case of a two-way connection you can set which connection partner will assume the active role.
- Sending operating mode messages
  Indicates whether or not the local partner sends operating mode messages to the connection partner.

Address details

Displaying address details of the S7 connection. With an unspecified partner, the values for the rack and slot can be changed. All other values are obtained from the current configuration and cannot be changed.

S7 connection - address details (S7-1200)

Meaning

The address details show the end points of the connection and can localize these via the specification of rack and slot.

When a connection is established, the connection-specific resources of a module are assigned permanently to this connection. This assignment requires that the connection resource can be addressed. The TSAP (Transport Service Access Point) is, as it were, the address of the resource that is formed with the help of the connection resource or, in the case of S7-1200 CPUs (firmware V2.0 or higher) with the SIMATIC-ACC (SIMATIC Application Controlled Communication).

Configuration of TSAP for S7-1200

- For S7-1200 CPU (firmare V2.0 or higher):
  "SIMATIC-ACC"<nnn><mm>
  nnn = Local ID
  mm = any value
- For S7-1200 CPU (firmare V1.0):
  <xx>.<yz>
  xx = Number of the connection resource
  y = Rack number
  z = Slot number

TSAP structure, dependent on partner

The configuration of the TSAP for S7-1200 CPUs is dependent on the respective firmware and on the remote connection partner. When a S7-1200 CPU is connected with a S7-300/400 CPU, a S7-1200 CPU also uses a TSAP configuration that includes the connection resource.

See the following examples of TSAPs of various connection configurations

- Connection between two S7-1200 CPUs (both with firmware V2.0):
  - S7-1200 CPU "A" with firmware V2.0 and local ID 100:
    TSAP: SIMATIC-ACC10001
  - S7-1200 CPU "B" with firmware V2.0 and local ID 5AE:
    TSAP: SIMATIC-ACC5AE01
- Connection between two S7-1200 CPUs (firmware V2.0 and V1.0):
  - S7-1200 CPU with firmware V2.0 and local ID 1FF:
    TSAP: SIMATIC-ACC1FF01
  - S7-1200 CPU with firmware V1.0 (rack 0, slot 1, connection resource 03):
    TSAP: 03.01
- Connection between S7-1200 CPUs (firmware V2.0) and S7-300/400 CPU:
  - S7-1200 CPU with firmware V2.0 (rack 0, slot 1, connection resource 12):
    TSAP: 12.01
  - S7-300/400 CPU (rack 0, slot 2, connection resource 11):
    TSAP: 11.02

S7 connections via CM/CP (S7-1200)

Introduction

S7-1200 CPUs with a firmware version of V2.0 or higher support one-way and two-way S7 connections via CM/CP interfaces. This increases the number of Ethernet ports and networks that can be used for S7 connections. Even though the connection is then guided via the CM/CP, the associated S7-1200 CPU is an end point of the connection. The other end point can be any other device in the case of two-way connections. This other device must also support S7 connections.

Data volumes and configuration limit

The number of communication connections which are supported by CM/CP can be found in the manual accompanying every CM/CP. The number of connections per device can be further increased by adding other CM/CPs.

If several CM/CPs are fitted in a device, an automatic switch is made to the next CM/CP when this limit is exceeded. Specifically assign the connections using the path selection as required.

Note

Data transfer > 240 byte transfer is supported by the current CPs.

CPs with an older product version support the transfer of data with a data length of up to 240 bytes.

For more information, refer to the details provided in the Ethernet CP equipment manual.

Tasks of the Ethernet CM/CP in online mode

During data transfer via a connection, the Ethernet CM/CP takes on the following tasks:

- Receiving
  Receiving data from the Ethernet and forwarding to the user data area in the CPU
- Sending
  Transferring data from the user data area of the CPU and sending data via the Ethernet

The connection is established automatically as soon as the partner can be reached.

S7 connections using PUT and GET instructions

Basic information on communication via the PUT/GET instruction

Basic information on PUT/GET instructions

Use PUT and GET instructions to exchange data between two CPUs via an S7 connection.

The GET instruction is used to read data from a partner CPU. The PUT instruction is used to control the writing of tags by the communication partner via the user program. Apart from the PUT and GET instructions, no additional communication functions are provided for reading and writing tags.

To simplify the use of the two instructions, specify all required parameters for the connection and all block parameters in the Inspector window of the program editor.

Requirement

To be able to use the PUT and GET instructions, the following requirements must be satisfied:

- At least one S7-1200/1500 CPU or S7-300/400 CPU must be created in the project. Firmware 2.0 or higher must be installed on an S7-1200 CPU. If you have not yet created a second CPU in the project, you can initially establish the connection to an unspecified partner.
- An S7 connection must exist between the two CPUs. If you have not yet established a connection between two CPUs, a connection is automatically established during the configuration of the instructions.
- For both instructions, an instance data block is required in which all data used by the instruction is stored. The instance data block is created automatically as soon as you drag a PUT or GET instruction to a network in the program editor. For the correct execution of the program, it is essential that the instance data blocks are not changed; consequently, these data blocks are know-how protected. You only have read access to the instance data blocks.

See also

Overview of connection configuration (Page 667)

Assigning parameters to start request (Page 672)

PUT: Set parameters for write and send area (Page 673)

GET: Set parameters for read and memory area (Page 674)

Connection configuration

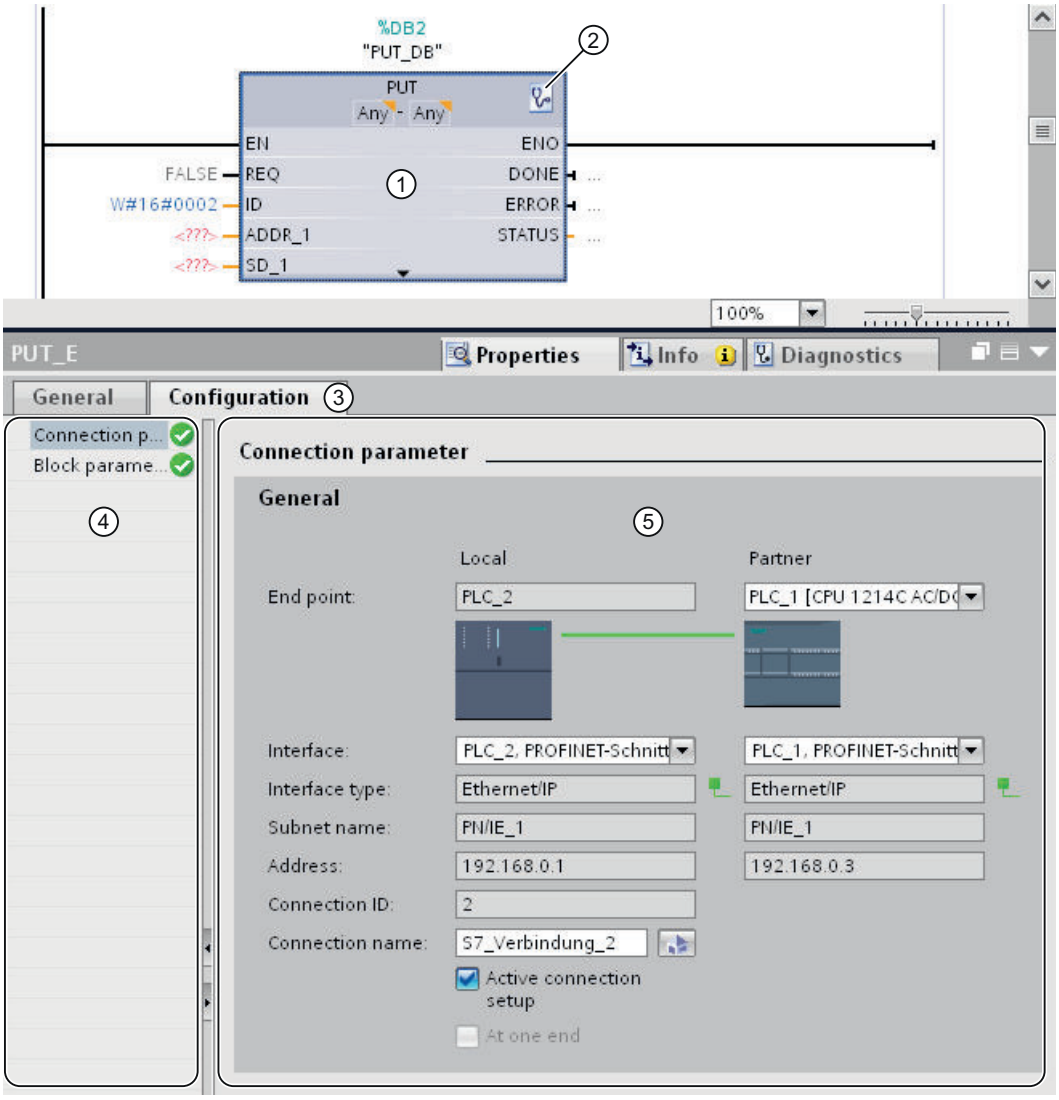
Overview of connection configuration

Introduction

The connection parameters for the PUT and GET instructions are assigned in the inspector window of the program editor. All parameters are saved in the corresponding instance data block.

Structure of the connection configuration

The connection configuration is made up of the following components:



① Communication instruction for PUT or GET
② Call of online and diagnostic functions
③ "Configuration" tab in the "Properties" tab
④ Area navigation of the "Configuration" tab
⑤ General properties of the connection parameters

Display of online and diagnostic functions

If you click the icon for starting the online and diagnostic functions, the associated CPU goes online automatically. The connection table in the network view is opened. In addition, the "Diagnostics" tab and the connection information are displayed in the inspector window.

Entering the connection parameters

Enter the desired connection parameters in the "Configuration" tab. The area navigation of the "Configuration" tab includes the "Connection parameters" group. This group contains the connection configuration. Here, you can enter the parameters for the connections using system functions. When all the required parameters are assigned, a check mark is set behind the "Connection parameters" group in the area navigation.

See also

Assigning parameters to start request (Page 672)

PUT: Set parameters for write and send area (Page 673)

GET: Set parameters for read and memory area (Page 674)

Description of the connection parameters

Overview

The following table shows the general connection parameters:

Parameter	Description
End point	The names of the local end point and the partner end point are shown. • Local end point The local end point is the CPU in which the PUT or GET instruction is programmed. • Partner end point The partner end point is selected from the drop-down list. The drop-down list shows all available possible connection partners including unspecified connection partners for devices whose data is unknown in the project. As long as no connection partner is set, all other parameters in the mask are disabled.
Interface	The interface of the partner CPU is displayed. The partner interface is not displayed until a specified partner CPU has been selected.
Interface type	The type of interface via which communication is handled is displayed.
Subnet name	The subnet of the local end point is displayed, provided this exists. The partner subnet is displayed only after the partner end point has been selected. If at least one of the two connection partners is not connected with a subnet, the two connection partners are automatically connected with each other. The partner which is not connected to a network is hereby connected to the same subnet via which the other partner is already connected to a network. A connection of connection partners to different subnets is only possible with IP or S7 routing. The IP routing settings can be edited in the properties of the relevant Ethernet interfaces.
Address	The IP address of the local end point is displayed. The IP address of the partner is displayed only after the partner end point has been selected. If you have selected an unspecified connection partner, the input box is empty and has a red background. In this case, you will need to specify a valid IP address for the connection partner.
Connection ID	The currently set connection ID is displayed. You can change the connection ID in the connection table in the network view. You can also directly access the connection table while you are setting the connection parameters. To do this, click the "Create new connection" icon.

Parameter	Description
Connection name	The name of the connection which was automatically created when the PUT/GET instruction was inserted is displayed. You can change the name of the connection by entered a different name in the field. You can also create a new connection or edit existing connections by clicking the "Create new connection" icon.
Active connection establishment	Use the "Active connection establishment" option button to specify which partner starts the communication. When the connection is created, the local partner is initially specified by default for the establishment of the connection. If a device does not support active connection establishment, you have to activate active connection establishment on the other partner.
Configured at one end	If this check box is selected, the connection partner is the server for this connection. It cannot actively send or receive. This corresponds to the behavior of the PUT/GET instructions. In this case, other instructions are not possible. If the check box is not selected, other instructions can also be used for the communication.

Starting connection parameter assignment

You can assign the connection parameters for PUT and GET in the inspector window as soon as you have inserted and selected a PUT or GET instruction in a program block.

Procedure

To insert PUT/GET instructions, follow these steps:

1. Open the "Instructions" task card in the "Communication > S7 Communication" folder.
2. Drag a PUT or GET instruction to a network.
   The "Call options" dialog opens.
3. Optional: Edit the properties of the instance DB in the "Call properties" dialog. You have the following options:
   - Change the default name.
   - Select the "Manual" check box to assign your own number.
4. Click "OK".

Result

A corresponding instance data block is created for the inserted PUT or GET instruction. For S7-300 CPUs, a function block is also created in the program resources.

When PUT or GET instruction is selected, you will see the "Configuration" tab under "Properties" in the inspector window. The "Connection parameters" group in area navigation contains the connection parameter assignment that you can now make.

See also

Creating and assigning parameters to connections (Page 671)

Deleting connections (Page 672)

Creating and assigning parameters to connections

You can create S7 connections and assign the parameters for these in the connection parameter assignment of the PUT/GET instructions. Changed values are checked immediately by the connection parameter assignment for input errors.

Requirement

A CPU exists with a PUT or GET communication instruction.

Procedure

To configure an S7 connection using PUT/GET instructions, follow these steps:

1. In the program editor, select the call of the PUT or GET instruction.
2. Open the "Properties > Configuration" tab in the inspector window.
3. Select the "Connection parameters" group. Until you select a connection partner, only the empty drop-down list for the partner end point is enabled. All other input options are disabled.
   The connection parameters already known are displayed:
   - Name of the local end point
   - Interface of the local end point
   - IP address of the local end point
4. In the drop-down list box of the partner end point, select a connection partner. You can select an unspecified device or a CPU in the project as the communications partner.
   The following parameters are automatically entered as soon as you have selected the connection partner:
   - Interface of the partner end point
   - Interface of the partner end point. If several interfaces are available, you can change the interface as required.
   - Interface type of the partner end point
   - Subnet name of both end points
   - IP address of the partner end point
   - Name of the connection which is used for the communication. If no connection exists yet, it is automatically established.
5. If required, change the connection name in the "Connection name" input box. If you want to create a new connection or edit an existing connection, click on the "Create new connection" icon.

Note

The PUT and GET instructions between two communication partners can only run if both the hardware configuration and the program part for the partner end point have been loaded into the hardware. To achieve fully functional communication, make sure that you load not only the connection description of the local CPU on the device but also that of the partner CPU as well.

Deleting connections

A connection which was automatically created during the insertion of a PUT or GET instruction appears in the connection table of the network view like every standard connection. As a result, it can be deleted in the connection table.

Procedure

To delete a connection, follow these steps:

1. Open the connection table in the network view.
2. In the connection table, select the connection that you want to delete.
3. To do this, right-click the connection and select the "Delete" command from the shortcut menu.

Result

The connection is deleted. The PUT or GET instruction and the associated instance data blocks are retained and must be manually deleted if necessary.

To continue using the PUT or GET instruction, you must configure the connection again in the inspector window of the program editor, since all connection parameters were also deleted when the connection was deleted. In this case, specify a new communication partner and a suitable connection.

Block parameter assignment

Assigning parameters to start request

To start communication via the PUT or GET instruction, you have to specify an event which activates the instruction. This event is referred to as control parameter (REQ). The communication job is activated as soon as there is a positive edge at the control parameter REQ.

Please note that the control parameter REQ is assigned the default FALSE at first call.

Requirement

- The program editor is open.
- You have already inserted a PUT or GET instruction.
- A connection has been established between two communication partners.

Procedure

To define the REQ control parameter, follow these steps:

1. Select the PUT or GET instruction in the program editor.
2. Open the "Configuration" tab in the inspector window.

3. Select the "Block parameter assignment" entry in the area navigation.
4. In the "REQ" field, select a tag of the "BOOL" data type to initialize the execution of the instruction. Alternatively, you can also interconnect a previous instruction in the program editor.

See also

Data consistency (Page 4383)

PUT: Set parameters for write and send area (Page 673)

GET: Set parameters for read and memory area (Page 674)

PUT: Set parameters for write and send area

For communication via the PUT instruction, you must specify the memory area of the partner CPU to which the data should be written. In addition, you must specify the memory area of the local CPU from which the data is to be read.

Requirement

- The program editor is open.
- You have already inserted a PUT instruction.
- A connection has been established between two communication partners.

Procedure

To specify the read and the memory area for the instruction, follow these steps:

1. Select the PUT instruction in the program editor.
2. Open the "Configuration" tab in the inspector window.
3. Select the "Block parameter assignment" entry in the area navigation.
4. In the "In/Outputs > Write area (ADDR_1) > Start" field, select a "REMOTE" data type pointer to the area of the partner CPU which is to be written.
   Only absolute addressing is permitted.
   Example: P#DB10.DBX5.0 Byte 10
5. In the "Length" field, enter the length of the write area and select the data type of the memory area from the drop-down list.
6. In the "In/Outputs > Send area (SD_1) > Start" field, select a pointer to the area in the local CPU which contains the data to be sent.
7. In the Length field, enter the length of the memory area to be read and select the data type from the drop-down list.
   Only the data types BOOL (for a bit array, "0" must be used as address and an integer multiple of byte must be used as length), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, COUNTER, TIMER are permitted.
   If the VARIANT pointer accesses a DB, the DB must always be specified (for example: P#DB10.DBX5.0 Byte 10).

See also

GET: Set parameters for read and memory area (Page 674)

GET: Set parameters for read and memory area

For communication via the GET instruction, you must specify the memory area of the local CPU to which the data should be written. In addition, you must specify the memory area of the partner CPU from which the data is to be read.

Requirement

- The program editor is open.
- You have already inserted a GET instruction.
- A connection has been established between two communication partners.

Procedure

To specify the read and the memory area for the instruction, follow these steps:

1. Select the GET instruction in the program editor.
2. Open the "Configuration" tab in the inspector window.
3. Select the "Block parameter assignment" entry in the area navigation.
4. In the "In/Outputs > Read area (ADDR_1) > Start" field, select a "REMOTE" data type pointer to the area of the partner CPU which is to be read.
   Only absolute addressing is permitted.
   Example: P#DB10.DBX5.0 Byte 10
5. In the "Length" field, enter the length of the read area and select the data type of the memory area from the drop-down list.
6. In the "In/Outputs > Memory area (RD_1) > Start" field, select a pointer to the area in the local CPU in which the read data is to be stored.
7. In the Length field, enter the length of the memory area and select the data type from the drop-down list.
   Only the data types BOOL (for a bit array, "0" must be used as address and an integer multiple of byte must be used as length), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, COUNTER, TIMER are permitted.

See also

PUT: Set parameters for write and send area (Page 673)

Connection resources

Configuration limits in the connection configuration

Connection resources

Some communications services require connections. During operation, communication connections utilize memory and program resources of the CPUs, CPs, and CMs involved in the communication path (e.g., memory areas in the CPU operating system). Within a controller (PLC), this applies, for example, to the CPU as connection end point and to the extensions with CPs as interfaces to the subnets. In most cases, one resource per CPU/CP/CM is assigned for a connection. In HMI communication, up to 3 connection resources are required for each HMI connection.

Available connection resources

The maximum configuration limit within the automation system is always defined by the CPU, either through the high system limit defined by the CPU (S7-1200/1500) or through the sum of usable resources of the CPU and CP (S7-300/400). In most cases, the available connection resources of a CPU/CP/CM are automatically distributed to their interfaces, as required.

Each CPU brings along reserved connection resources for programming device, HMI, and Web server communication in its configuration. In addition, there are available resources that can be used for any HMI, S7 and open communication (S7-1200/1500).

You can determine which resources are available to you based on the configuration of the devices as follows:

- Selection of the device type at connection end point (CPU)
- Number and type of additional communication modules (CP, CM)

The number of connection resources available differs for the various automation systems:

- S7-300: The used CPU defines the number of available resources for connections. The maximum number of resources for connections can be expanded by added CPs.
- S7-400: The used CPU defines the number of possible resources for S7 connections. The resources can also be used for other connection types, in which case the maximum number of resources for other connection types can be expanded by adding CPs.
- S7-1200: The used CPU defines the number of possible resources for S7 connections. This high limit can be extended by adding CPs or CMs.
- S7-1500: The CPU that is used defines the applicable high limit of the connection resources for the whole automation system. The CPU itself accounts for some of these connection resources, and additional resources are accrued via CPs and CMs. However, regardless of how many CPs and CMs are added, the total number of connection resources cannot exceed the high limit of connection resources defined by the CPU type.

Time of occupation of connection resources

Which resources are actually used in operation, depends on which communication functions are used at any specific time and which communication connections are enabled at any specific time. Therefore, only limited checks of compliance with specific configuration limits and available communication resources can be performed at the time of configuration. The connection configuration offers the greatest possible support for this.

Factors influencing the configuration limits

You determine the communication resources with the device configuration:

- Number of available S7 connection resources in the CPU (S7-300/400)
- Number of connection resources for S7 communication and open communication in the station (S7-1500)
- Number of protocol-specific connection resources in the CPs/CMs
- Number of available CPs/CMs
- Automatic, programmed, or configured connection setup

Behavior of the connection configuration

During input, the connection configuration checks the following circumstances and behaves as follows:

- During the creation of S7 connections
  Is the maximum number of available S7 connections assigned? If yes, it is not possible to configure another S7 connection via the selected interface.
  For the S7-300, it is possible to increase the configuration limit by adding CPs.
- During the creation of any connection types
  Is the maximum number of protocol-specific connection resources in the interface (CP) assigned? If yes, it is not possible to configure another connection of the selected type via the selected interface.
  For the S7-300/400, it is possible to increase the configuration limit by adding CPs.
- During the creation of any connection types with OPC (PC stations)
  In the case of connections with OPC as connection end point, more connections can generally be configured than can be operated at any one time in the CP used in the PC station. That is because the connection properties differentiate between "Maintain permanently" and "Set up on demand".

When the connection configuration is compiled, a warning or, in the case of a non-assignable interface, an error is output, if appropriate.

Note

S7 connection resources are exhausted

If the maximum number of S7 connection resources is reserved, no S7 connection resources additionally required for data transmission between CPU and CP are in operation or possibly available for connections. The corresponding communication jobs are then refused.

You should therefore make sure that there is a sufficient number of non-assigned S7 connection resources, depending on the total assigned communication functions.

You can find the number of reserved S7 connection resources for the selected S7-300 or S7-1200/1500 CPU in the area navigation under "Connection resources". For the S7-300, only the connection resources of the CPU are shown. For the S7-1200/1500, you also see the resources of the CPs and CMs.

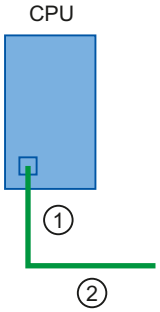
Connection resources and communication type

Every connection needs connection resources for the endpoint and transition point (e.g., CP, CM) on the devices involved. The number of available connection resources is CPU/CP/CM-specific.

If all the connection resources of a communication partner are assigned, no new connections can be established. Each type of communication is considered in detail below. Any number of combinations are possible however as long as the available connection resources are taken into account.

S7 connections via integrated interface

With S7 connections via the integrated MPI/PROFIBUS DP/PN interface, one connection resource is assigned for the endpoint on the CPU per S7 connection. This applies to all S7-300/400/1200/1500 CPUs.



① Assigned S7 connection resource on the CPU

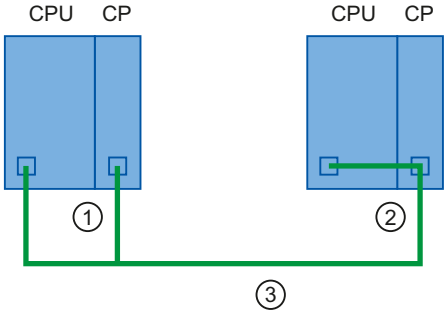
② Subnet (e.g., MPI, PROFIBUS-DP, or Industrial Ethernet/PROFINET)

- S7-300/400 CPU: One S7 connection resource is assigned on the CPU.
- S7-1200/1500 CPU: One S7 connection resource is assigned on the CPU and thus also one S7 connection resource on the station.

S7 connections via external interfaces

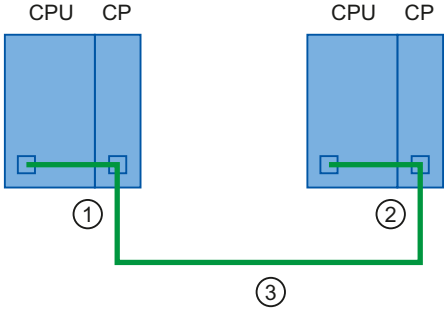
With S7 connections via an external CP interface, a distinction should be made between the following scenarios:

- S7-300 CPU
  - S7 connection configured at one end: One S7 connection resource is assigned on the CPU (for the endpoint) and one S7 connection resource on the CP (transition point) for each S7 connection.



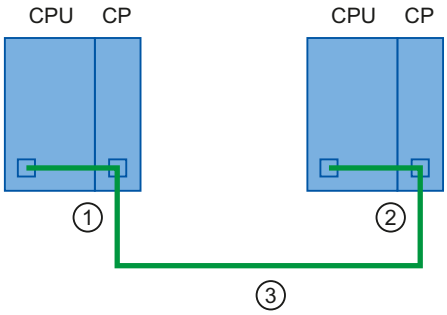
① One S7 connection resource in the CPU and one in the CP of the local end point
② Connection partner
③ Subnet, e.g., Industrial Ethernet

  - S7 connection configured at both ends: One S7 connection resource is assigned on the CP for each S7 connection.



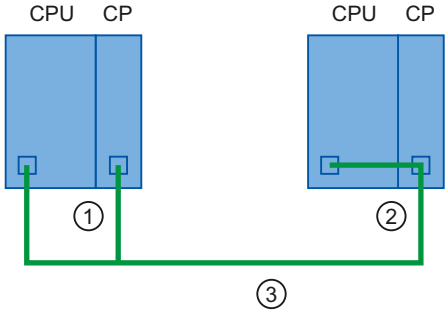
① One S7 connection resource in the CP of the local end point
② Connection partner
③ Subnet, e.g., Industrial Ethernet

- S7-400 CPU
  One S7 connection resource is assigned on the CPU (for the endpoint) and one S7 connection resource on the CP (transition point) for each S7 connection.



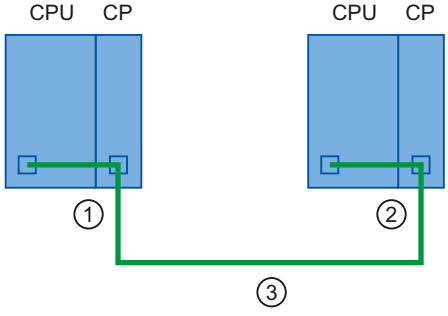
① One S7 connection resource in the CPU and one in the CP of the local end point
② Connection partner
③ Subnet, e.g., Industrial Ethernet

- S7-1200/1500 CPU
  - S7 connection configured at one end: One S7 connection resource is assigned on the CPU and thus also one S7 connection resource on the station for each S7 connection.



① One S7 connection resource in the CPU and one in the CP of the local end point
② Connection partner
③ Subnet, e.g., Industrial Ethernet

  - S7 connection configured at both ends: One S7 connection resource is assigned on the CPU and thus also one S7 connection resource on the station for each S7 connection.



① One S7 connection resource in the CP and, thus, also one in the station
② Connection partner
③ Subnet, e.g., Industrial Ethernet

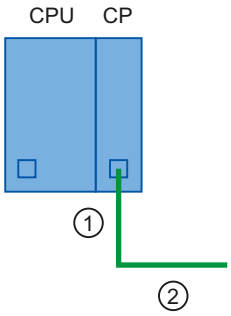
Note

The number of reserved S7 connection resources for the selected CPU for certain CPU types of the S7-300, and for the selected CPU and its connected communication modules for the S7-1200/1500, can be seen in the area navigation under "Connection resources". S7 connection resources for OP/PG and S7 basic communication can be reserved here.

Connections for open communication services (SEND/RECEIVE interface)

Communication via the SEND/RECEIVE interface is solely via CPs. For this, one protocol-specific connection resource is allocated for the endpoint per connection (i.e., FDL, ISO transport ISO on TCP, UDP and TCP connection).

Here, no connection resources are required for the connection on the CPU.



① 1 assigned connection resource in the CP

② Subnet, e.g., Industrial Ethernet

Exception: on the S7-400 CPU, connections with the property "SPEED SEND/RECEIVE" require an additional S7 connection resource for each "SPEED SEND/RECEIVE" connection on the CPU.

Connection resources and connection type

Setting up the connection

Often, you can decide between a configured setup or a programmed setup. The programmed setup makes it easier to release connection resources following data transfer. With configured setup, you receive help in managing connection resources in STEP 7.

Table 10-2 Setting up the communication service

Communication service	Automatic setup	Programmed setup	Configured setup
PG communication	X	-	-
HMI communication	X	-	X
S7 communication	-	X	X
Open communication via TCP/IP	-	X	X

Communication service	Automatic setup	Programmed setup	Configured setup
Open communication via ISO-on-TCP	-	X	X
Open communication via UDP	-	X	X
Open communication via ISO	-	X	X
Open communication via e-mail	-	X	-
Open communication via FTP	-	X	-
Open communication via FDL	-	X	X
S7 routing	-	-	X

Assignment of connection resources according to the connection setup

Connection resources are assigned in different ways, depending on the type of connection setup:

- Configured setup
  In the case of configured connections, the connection resource is assigned as soon as the hardware configuration has been downloaded to the CPU. For OPC servers, the resources are assigned, as required.
- Programmed setup
  The call of the appropriate instructions for establishing the connection (using TSEND_C or T_CON) triggers assignment of a connection resource.
- Automatic setup
  As soon as you have connected the programming device or HMI device to a CPU physically and online, the connection resources will be assigned.

Connection resources for configured setup

You set up the connection in the network view of the hardware and network editor in the context of a CPU.

Following the data transfer, the connection is not terminated, i.e., the connection resources remain permanently assigned. To terminate the connection you must delete the connection configuration and download the modified configuration to the CPU.

OPC connections are automatically established and terminated as needed.

If you have **configured** the connections, compliance with the maximum possible number of connection resources in the automation system will be monitored.

Connection resources for programmed setup

You set up the connection in the program editor in the context of a CPU by parameter assignment of instructions for communication, e.g., TSEND_C.

Once the connection is established by the TCON or TSEND_C instruction, the connection will be terminated through appropriate parameter assignment of the TSEND_C instruction or by calling the TDISCON instruction after the data transfer. If the connection is terminated, the connection resources in the CPU/CP/CM are available again. If the connection is retained and

another data transfer is performed, the processing time of the instructions will be shorter because the connection does not have to be reestablished.

If the establishment and termination of connections are **programmed** in the user program, you are responsible for ensuring compliance with the automation system limits. If necessary, you will receive a communication error at the "STATUS" output parameter of the TSEND_C or T_CON instruction and can obtain more detailed information from the connection diagnostics.

Connection resources for automatic setup

The connection will be set up automatically (e.g., programming device or HMI connection) as soon as the PG/PC interface has been physically connected to an interface of the CPU and the assignment to the interface has been made in the "Go online" dialog.

For **automatically** set up connections, compliance with the connection resources reserved in the automation system for these connections is monitored.

See also

TCON: Establish communication connection (Page 4474)

TDISCON: Terminate communication connection (Page 4484)

TSEND_C: Send data via Ethernet (Page 4419)

TSEND_C: Establishing a connection and sending data (Page 4423)

TCON: Establish communication connection (Page 4478)

S7-1200 connection resources: (S7-1200)

Display of connection resources (S7-1200)

Connection resources in the Inspector window

You will find an overview of the reserved and available connection resources of an S7-1200 automation system in the inspector window of the hardware and network editor under the CPU properties.

Overview of connection resources

The CPU supports at most the following number of simultaneous asynchronous communication connections for PROFINET and PROFIBUS. The maximum number of connection resources assigned to each category is predefined. You cannot modify the values. The 6 freely available connections can, however, be used to increase the number of connections in each category as required by your application.

The display of connection resources distinguishes between the offline and online view. The following figure shows you the offline display of the reserved and available connection resources of a CPU 1217C:

Connection resources

	① Station resources			② Module resources
	Reserved		Dynamic ⚠	PLC_11 [CPU 1217C DC/DC/DC]
Maximum number of resources:	62		6	68
	Maximum	Configured	Configured	Configured
PG communication:	4	-	-	-
HMI communication:	12	0	0	0
S7 communication:	8	0	0	0
Open user communication:	8	0	0	0
Web communication:	30	-	-	-
Other communication:	-	-	0	0
Total resources used:		0	0	0
Available resources:		62	6	68

① Station-specific connection resources

② Module-specific connection resources

Station-specific connection resources

The columns of the station-specific connection resources provide information about the reserved and dynamically available connection resources of the station. In the example, a maximum of 68 station-specific connection resources are available for the automation system.

- 62 connection resources are reserved for specific connection types
- 6 connection resources can be used freely for connection types

The warning triangle in the column of the dynamic station resources is displayed because the sum of the maximum available connection resources of the CPU has reached the station limit of 68.

Note

Exceeding the available connection resources

STEP 7 signals the reaching of the station-specific connection resources with a warning. If you require more connection resources, you either have to use a CPU with a higher maximum number of available station-specific connection resources or reduce the number of communications connections.

Module-specific connection resources

The columns of the module-specific connection resources provide information about the allocation of resources on the CPUs, CPs and CMs of an automation system: The display is per module and not per interface. In the example, the CPU provides a maximum of 68 connection resources.

Note

If you add CM/CP modules, the total number of S7-1200 communication resources is not increased.

Online view

If you are connected to the CPU online, you can also see how many resources are currently being used in the "Connection information" tab of the "Diagnostics" section in the Inspector window.

The online view of the "Connection resources" table not only displays the reserved and dynamic connection resources of the configured connections, but also includes columns containing the connection resources of the programmed and automatically set-up connections that are currently allocated in the station. Thus, the online view displays **all** the assigned connection resources in the automation system, regardless of how the connection was set up. The "Other communication" row displays connection resources assigned for communication with external devices and communication via data record routing.

The table is updated automatically. If there are inconsistencies between the configured hardware configuration and the hardware configuration available online, a corresponding message will be output.

Resource requirements of the connections

The communication connections of open user communication, S7 connection, HMI, programming device and Web server (HTTP) can utilize several connection resources on the basis of the functions used.

The following number of connections are available per device on the basis of the assigned connection resources:

	Programming device (PG)	HMI devices	GET/PUT client/ server	Open user communication	Web browser
Maximum number of connection resources of a CPU 1217C	4 (guaranteed support of a programming device connection)	12 (guaranteed support of four HMI connections)	8	8	30 (guaranteed support of three HTTP connections)

Programming device communication

For example, four connection resources are available for programming device communication. Depending on the currently used programming device functions, the programming device occupies up to three connection resources for a connection. With the S7-1200, at least one programming device connection is guaranteed at all times.

HMI communication

A further example is the number of HMI devices as shown in the following figure. 12 connection resources are available for HMI devices. Depending on the type or model of your HMI device and the HMI functions used, the HMI device can require up to three connection resources for an HMI connection. For example, a Basic Panel requires up to two connection resources in the CPU for an HMI connection, while a Comfort Panel requires up to three connection resources in the CPU. Depending on the number of required connection resources, at least four HMI connections can be configured simultaneously.

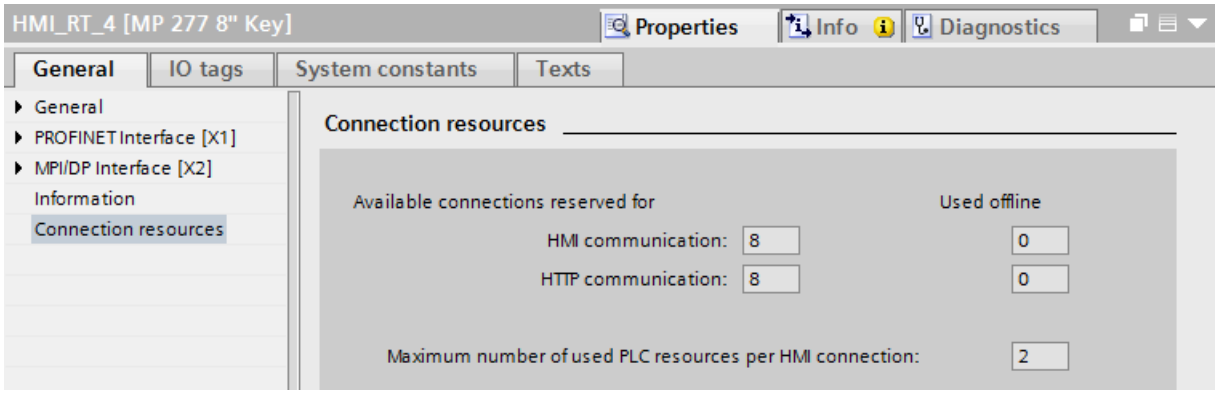
An HMI device can use its available connection resources for the following functions:

- Reading
- Writing
- Interrupts and diagnostics

This is only an example. The actual number of connections can differ depending on the type and version of the HMI device.

Example	HMI 1	HMI 2	HMI 3	HMI 4	HMI 5	Total number of available connection resources
Used connection resources	2	2	2	3	3	12

Information regarding the availability and assignment of connection resources for HMI connections is available in the offline view in the HMI device context (under "Properties > General > Connection resources" in the Inspector window):



The following is displayed in the connection resources area:

- Number of available connections on the HMI reserved for HMI communication and HTTP communication
- Number of connection resources for HMI communication and HTTP communications used offline in the HMI
  If the maximum number of available connection resources for an HMI device is exceeded, a corresponding alarm is output by STEP 7.
- Maximum number of used PLC resources per HMI connection. This parameter is a factor that is to be multiplied by the number of HMI connections used offline. The product is the number of HMI resources occupied on the CPU.

If the maximum number of available connection resources for an HMI device is exceeded, a corresponding alarm is output.

Web communication

The CPU offers Web server (HTTP) connections for several Web browsers. The number of browsers that can be supported simultaneously by the CPU depends on the number of connections that a Web browser requests/utilizes.

Connection resources S7-1500 (S7-1500)

Connection resources in the automation system (S7-1500)

Introduction

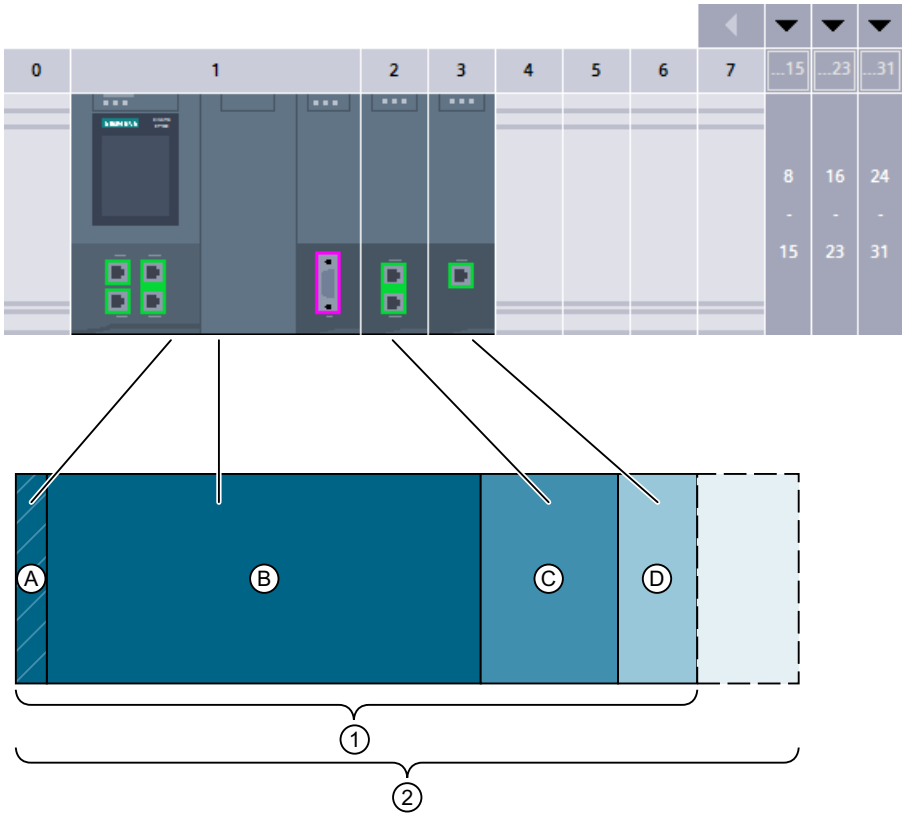
Some communications services require connections. Connections occupy resources in the automation system. The connection resources are made available by the CPUs, communications processors (CPs) and communications modules (CMs).

Connection resources of a station

The connection resources available depend on the CPUs, CPs and CMs and may not exceed a maximum number per station. The maximum number of resources of a station is determined by the CPU.

Each CPU has reserved connection resources for PG, HMI and Web server communication. This ensures, for example, that a PG can always establish at least one online connection with the CPU regardless of how many other communications services are already using connection resources. In addition, there are dynamic resources. The difference between the maximum number of connection resources and the number of reserved connection resources is the maximum number of dynamic connection resources. The communication services PG communication, HMI communication, S7 communication, Open User Communication, Web communication and other communication (such as OPC UA) use the dynamic connection resources from the pool.

The figure below shows an example of how individual components make connection resources available to an S7-1500 station.



① Available connection resources of the station, of which

- A Reserved connection resources of the station
- A + B Connection resources of CPU 1518
- C Connection resources of communications module CM 1542-1
- D Connection resources of communications processor CP 1543-1

② Maximum communications resources of the station using the example of a configuration from CPU 1518, CM 1542-1 and CP 1543-1

Number of connection resources of a station

The following table lists the maximum number of available connection resources for some CPU types:

Connection resources of a station	1511 1511C	1512C 1513	1515	1516	1517	1518
Maximum connection resources of the station	96	128	192	256	320	384
of which reserved	10					
of which dynamic	86	118	182	246	310	374
Connection resources of the CPU	64	88	108	128	160	192
Max. additionally usable connection resources by plugging in CMs/CPs	32	40	84	128	160	192

Connection resources of a station	1511 1511C	1512C 1513	1515	1516	1517	1518
Additional connection resources CM 1542-1	64					
Additional connection resources CP 1543-1	118					
Additional connection resources CM 1542-5	40					
Additional connection resources CP 1542-5	16					

The number of connection resources supported by a CPU or a communication module is described in the technical specifications section of the manuals.

Example

You have configured a CPU 1518-4PN/DP with a communications module CM 1542-1 and a communications processor CP 1542-5.

- Maximum connection resources of the station: **384**
- Available connection resources:
  - CPU 1518-4 PN/DP: 192
  - CM 1542-1: 64
  - CP 1542-5: 16
  - Total: **272**

The setup provides 272 connection resources. By adding further communication modules the station can support a maximum of 112 additional connection resources.

Reserved connection resources

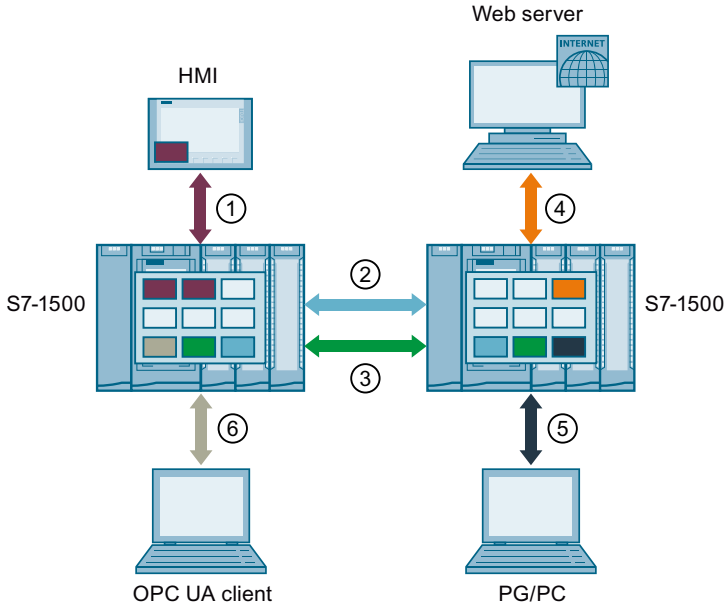
10 connection resources are reserved for stations with S7-1500 CPU, ET 200SP CPU and ET 200pro CPU based on S7-1500:

- 4 for PG communication required by STEP 7, for example, for test and diagnostics functions or downloading to the CPU
- 4 for HMI communication which are occupied by the first HMI connections configured in STEP 7
- 2 for communication with the Web server

Allocation of connection resources (S7-1500)

Overview - occupation of connection resources

The following figure shows how different connections occupy the resources of the S7-1500:



① HMI communication: See below.

② Open User Communication: Connections of Open User Communication occupy a connection resource in every end point.

③ S7 communication: Connections of S7 communication occupy a connection resource in every end point.

④ Web communication: The Web server connection occupies at least one connection resource in the station. The number of occupied connections depends on the browser.

⑤ PG communication: The PG connection occupies one connection resource in the station.

⑥ OPC UA communication: Each session that the OPC UA server of the CPU establishes with an OPC UA client usually occupies one connection resource (other communication) in the station.

Connection resource for HMI communication

Connection resource for Open User Communication

Connection resource for S7 communication

Connection resource for Web communication

Connection resource for PG communication

Connection resource for other communication (for example OPC UA)

Connection resources for HMI communication

With HMI communication, the occupation of connection resources in the station depends on the HMI device being used. The following table lists the maximum number of occupied connection resources for an HMI connection depending on the various HMI devices:

HMI device	Maximum number of allocated connection resources of the station for each HMI connection
Basic Panel	1
Comfort Panel	2[1]
RT Advanced	2[1]
RT Professional	3

[1] If you do not use system diagnostics or alarm configuration, the station occupies only one connection resource per HMI connection.

Example: You have configured the following HMI connections for a CPU 1516-3 PN/DP:

- Two HMI connections to an HMI TP700 Comfort (2 connection resources each)
- One HMI connection to an HMI KTP1000 Basic (1 connection resource)

A total of 5 connection resources are allocated for HMI communication in the CPU.

Connection resources for routing

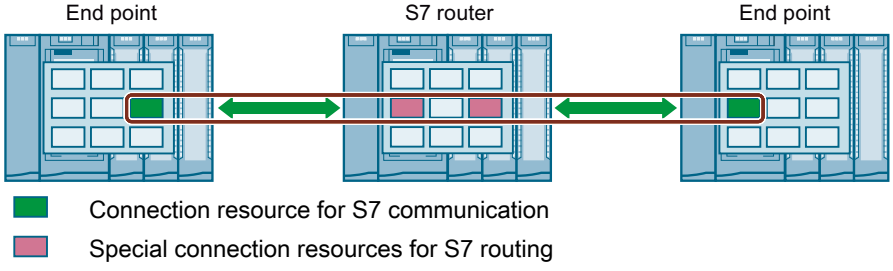
S7 connections beyond network boundaries (S7 routing)

To transfer data beyond S7 subnets ("S7 routing"), an S7 connection is established between two CPUs. The S7 subnets are connected via gateways known as S7 routers. The router is the transition point of an S7 connection and is capable of establishing S7 connections. CPUs, CMs and CPs in S7-1500 are S7 routers.

The following applies for a routed S7 connection:

- A routed connection occupies one connection resource each in both end points. STEP 7 shows these connection resources in the "Connection resources" table.
- On the S7 router, two special connection resources are occupied for S7 routing. STEP 7 does not display the special connection resources for S7 routing in the "Connection resources" table. The number of resources for S7 routing depends on the CPU. You will find the resources for S7 routing in the technical specifications of the CPU in "Number of S7 routing connections".

The following figure shows the allocation of the connection resources at S7 routing:



Data record routing

Data record routing also enables transfer of data beyond S7 subnets from an engineering station connected to PROFINET to various field devices via PROFIBUS.

With data record routing, as with S7 routing, two of the special connection resources for S7 routing are also occupied at the network transitions in every record router. At the end point of communication (CPU), connection resources for data record routing are displayed in the online view of the "Connection resources" table in the "Other communication" row.

Note

Connection resources with data record routing

With data record routing, on the data record router, two special connection resources for S7 routing are occupied. Neither the data record connection nor the allocated connection resources are displayed in the table of connection resources. You are responsible for observing the limits during configuration.

When are connection resources occupied?

The time for the occupation of connection resources depends on how the connection is set up.

- Programmed setup of a connection
  As soon as an instruction to establish a connection is called in the user program of the CPU (TSEND_C/TRCV_C or TCON), a connection resource is occupied. With suitable parameter assignment of the CONT parameter of the TSEND_C/TRCV_C instructions or by calling the TDISCON instruction, the connection can be terminated following data transfer and the connection resource is available again. If the connection is terminated, the connection resources on the CPU/CP/CM are available again.
- Configured connections (e.g. HMI connection)
  If you have configured a connection in STEP 7, the connection resource is occupied as soon as the hardware configuration is loaded to the CPU. After using a configured connection for data transfer, the connection is not terminated. The connection resource is permanently occupied. To release the connection resource again, you need to delete the configured connection in STEP 7 and download the modified configuration to the CPU.
- Programming device connection
  As soon as you have connected the programming device to a CPU online in STEP 7, connection resources are occupied.
- Web server
  As long as you have opened the Web server of the CPU in a browser, connection resources are occupied in the CPU.
- OPC UA server
  As long as a session exists between the OPC UA server of the CPU and an OPC UA client, a connection resource is occupied in the CPU.

Monitoring the maximum possible number of connection resources

Offline

During configuration of connections, STEP 7 monitors the occupation of the connection resources. If the maximum possible number of connection resources is exceeded, STEP 7 signals this with a suitable warning.

Online

The CPU monitors the use of connection resources in the automation system. If you establish more connections in the user program than those provided by the automation system, the CPU acknowledges the instruction to establish the connection with an error.

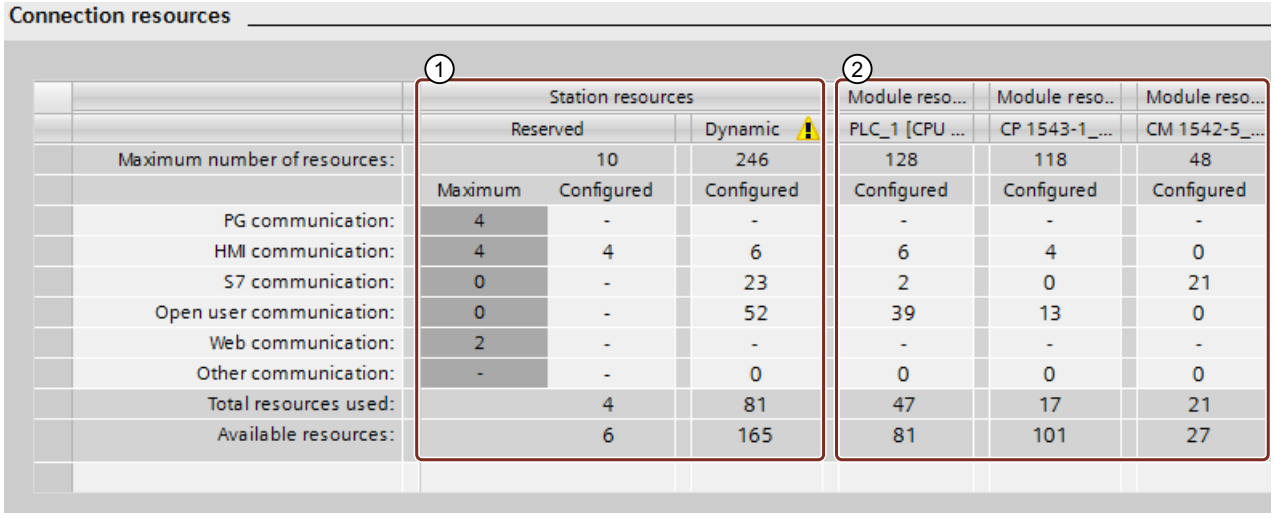
Display of the connection resources (S7-1500)

Connection resources in the Inspector window

You will find an overview of the reserved available connection resources of an S7-1500 automation system in the Inspector window of the hardware and network editor under the CPU properties.

Display of the connection resources in STEP 7 (offline view)

The display of connection resources distinguishes between the offline and online view. The following figure shows you the offline display of the reserved and available connection resources of an S7-1500 CPU with a CP and a CM:



	Station resources			Module reso...	Module reso..	Module reso...
	Reserved		Dynamic	PLC_1 [CPU ...	CP 1543-1_...	CM 1542-5_...
Maximum number of resources:		10	246	128	118	48
	Maximum	Configured	Configured	Configured	Configured	Configured
PG communication:	4	-	-	-	-	-
HMI communication:	4	4	6	6	4	0
S7 communication:	0	-	23	2	0	21
Open user communication:	0	-	52	39	13	0
Web communication:	2	-	-	-	-	-
Other communication:	-	-	0	0	0	0
Total resources used:		4	81	47	17	21
Available resources:		6	165	81	101	27

① Station-specific connection resources

② Module-specific connection resources

Station-specific connection resources

The columns of the station-specific connection resources provide information about the reserved and dynamically available connection resources of the station. In the example, a maximum of 256 station-specific connection resources are available for the automation system.

- 10 reserved connection resources, of which 4 are already in use and a further 6 still available.
  The occupied 4 resources are used for HMI communication.
- 246 dynamic connection resources, of which 81 are already in use and a further 165 still available.
  The configured resources are divided up as follows:
  - 6 resources for HMI communication
  - 23 resources for S7 communication
  - 52 resources for Open User Communication

The warning triangle in the column of the dynamic station resources is therefore displayed because the sum of the maximum available connection resources of CPU, CP and CM (= 294 connection resources) exceeds the station limit of 256.

Note

Exceeding the available connection resources

STEP 7 signals the exceeding of the station-specific connection resources with a warning. To make full use of the connection resources from the CPU, CP and CM, either use a CPU with a higher maximum number of available station-specific connection resources or reduce the number of communications connections.

Module-specific connection resources

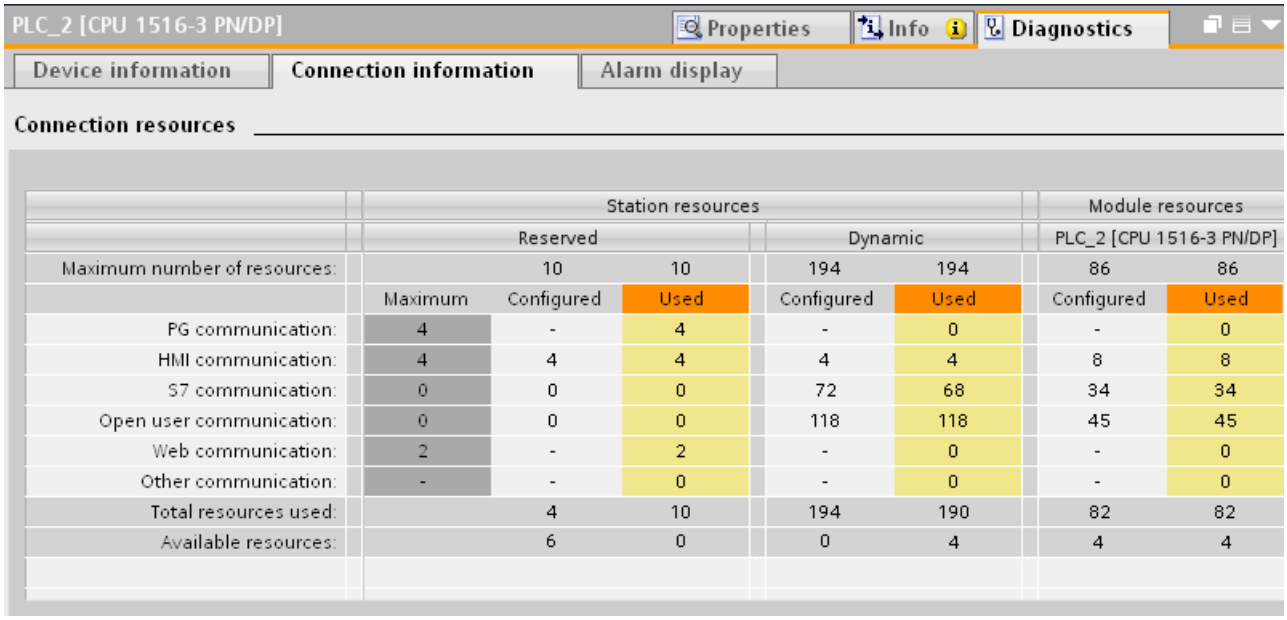
The columns of the module-specific connection resources provide information about the allocation of resources on the CPUs, CPs and CMs of an automation system: The display is per module and not per interface. In the example, the CPU makes a maximum of 128 connection resources available, of which 47 are already in use and a further 81 still available. The configured resources are divided up as follows:

- 6 resources for HMI communication
- 2 resources for S7 communication
- 39 resources for Open User Communication

The CP makes 118 resources and the CM 48 resources available, of which 17 resources and 21 resources respectively are used.

Display of the connection resources in STEP 7 (online view)

If you are connected to the CPU online, you can also see how many resources are currently being used in the "Connection information" tab of the "Diagnostics" section in the Inspector window. The following figure shows you the online display of the reserved, available and used connection resources of an S7-1500 CPU:



	Station resources					Module resources	
	Reserved			Dynamic		PLC_2 [CPU 1516-3 PN/DP]	
Maximum number of resources:		10	10	194	194	86	86
	Maximum	Configured	Used	Configured	Used	Configured	Used
PG communication:	4	-	4	-	0	-	0
HMI communication:	4	4	4	4	4	8	8
S7 communication:	0	0	0	72	68	34	34
Open user communication:	0	0	0	118	118	45	45
Web communication:	2	-	2	-	0	-	0
Other communication:	-	-	0	-	0	-	0
Total resources used:		4	10	194	190	82	82
Available resources:		6	0	0	4	4	4

The online view of the "Connection resources" table not only displays the reserved and dynamic connection resources of the configured connections, but also includes columns containing the connection resources of the programmed and automatically set-up connections that are currently allocated in the station. Thus, the online view displays **all** the assigned connection resources in the automation system, regardless of how the connection was set up. The "Other communication" row displays connection resources assigned for communication with external devices and communication via data record routing.

The table is updated automatically. If there are inconsistencies between the configured hardware configuration and the hardware configuration available online, a corresponding message will be output.

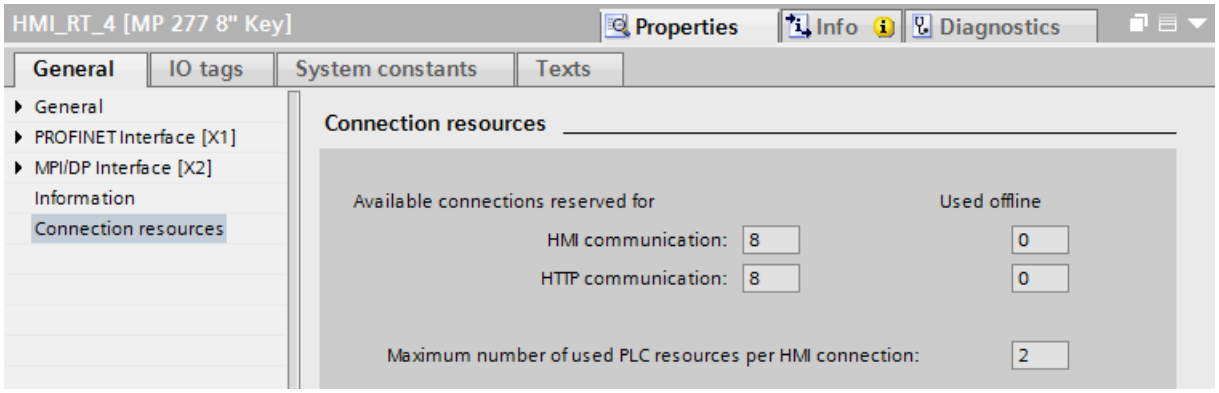
Note

If a routed S7 connection goes through a CPU, the required connection resources of the CPU do not appear in the table of connection resources!

Display of the connection resources for HMI

For HMI communication, connection resources in the CPU are assigned according to the HMI device used and the applications in which the HMI device is used. For example, a Basic Panel requires up to two connection resources in the CPU for an HMI connection, while a Comfort Panel requires up to three connection resources in the CPU.

Information regarding the availability and allocation of connection resources for HMI connections is available in the offline view in the HMI device context (under "Properties > General > Connection resources" in the Inspector window).



The following is displayed in the connection resources area:

- Number of available connections on the HMI reserved for HMI communication and HTTP communication
- Number of connection resources for HMI communication and HTTP communications used offline in the HMI
  If the maximum number of available connection resources for an HMI device is exceeded, a corresponding alarm is output by STEP 7.
- "Maximum number of used PLC resources per HMI connection". This parameter is a factor that is to be multiplied by the number of HMI connections used offline. The product is the number of HMI resources occupied on the CPU.

If the maximum number of available connection resources for an HMI device is exceeded, a corresponding alarm is output.

Displaying the connection resources in the Web server

You can display the connection resources not only in STEP 7, but also with a browser that displays the relevant page of the Web server. You will find information on displaying connection resources in the Web server in the function manual for the Web server.

Example for calculating the connection resources (S7-1500)

Introduction

This example for the table displaying connection resources shows you how the individual table entries are put together.

Requirement

- 2x CPU 1516-3 PN/DP with one CP 1543-1 and one CM 1542-1 each as connection partners for purposes of extending the configuration limit.
- 1x TFT display MP 377 12" Touch with WinAC MP 377 as a connection partner for HMI connections.
- The devices are networked together.

Creating the connections

Once the connection partners (CPUs, CPs, CMs, MP 377) have been created and networked in the network view of the hardware and network editor, follow these steps:

1. Switch to connection mode in the network view.
2. Create the following connections:
   - S7 connections: 2x via CPU, 21x via CM
   - TCP connections: 2x via CPU, 4x via CM
   - ISO connection: 6x via CP
   - ISO-on-TCP: 3x via CPU, 4x via CM
   - UDP connections: 4x via CPU, 106x via CP
   - HMI connections: 1x via CPU, 2x via CP

Result

The display of connection resources shows you not only the available resources but also the resources required for the created connections.

The available resources of devices and modules are highlighted in the following figure.

	Station resources			Module res...	Module res...	Module res...
	Reserved		Dynamic ③ ⚠	PLC_2 [CPU ..	CP 1543-1_...	CM 1542-5_...
Maximum number of resources:	① 10		246	② 128	118	118
	Maximum	Configured	Configured	Configured	Configured	Configured
PG communication:	4	-	-	-	-	-
HMI communication:	4	4	5	3	6	0
S7 communication:	0	-	23	2	0	21
Open user communication:	0	-	129	9	112	8
Web communication:	2	-	-	-	-	-
Other communication:	-	-	-	-	-	-
Total resources used:		4	157	14	118	29
Available resources:		6	89	114	0	89

① Depending on the CPU, at most 256 connections are available for this station. Ten (10) resources are reserved for particular connection types and 246 resources are dynamically available.

② The CPU provides 128 connection resources, and the CP and CM each provide another 118 resources.

③ CPU, CP, and CM altogether provide 128 + 118 + 118 = 364 resources. Because more connection resources are provided via the devices and modules than the station can operate with its maximum limit of 256 connection resources, a warning icon is displayed.

The following figure highlights the resources utilized by the created connections:

	Station resources			Module res...	Module res...	Module res...
	Reserved		Dynamic	PLC_2 [CPU ..	CP 1543-1_...	CM 1542-5_..
Maximum number of resources:		10	246	128	118	118
	Maximum	Configured	Configured	Configured	Configured	Configured
PG communication:	4	-	-	-	-	-
HMI communication:	4	4	5	3	6	0
S7 communication:	0	-	23	2	0	21
Open user communication:	0	-	129	9	112	8
Web communication:	2	-	-	-	-	-
Other communication:	-	-	-	-	-	-
Total resources used:		4	157	14	118	29
Available resources:		6	89	114	0	89

① These are the resources assigned via the HMI connections. In this example, one HMI connection utilizes 3 connection resources. For the whole station, 3 + 6 = 9 connection resources are required for the HMI communication.

② Four (4) resources are reserved for the HMI communication.

③ The 9 resources required for HMI communication are broken down as follows: For the first 4 resources, the resources already reserved for HMI communication will be used. The other 5 required resources will be taken from the dynamic resources.

④ These are the resources assigned via the S7 connections. An S7 connection occupies 1 connection resource. For the whole station, 2 + 21 = 23 connection resources are required.

⑤ For the S7 communication, there are no reserved resources. All 23 required resources come from the dynamically available resources.

⑥ These are the resources of the Open User Communication (for TCP, FDL, ISO, ISO-on-TCP, and UDP). For the whole station, 9 + 112 + 8 = 129 connection resources are required.

⑦ For the Open User Communication, there are no reserved resources. All 129 required resources come from the dynamically available resources.

Calculation of connection resources

Based on the available and utilized resources of individual devices and modules, the resources that are still free and the total resources used are calculated:

	Station resources			Module res...	Module res...	Module res...
	Reserved		Dynamic	PLC_2 [CPU ..	CP 1543-1_...	CM 1542-5_...
Maximum number of resources:	10		246	128	③ 118	118
	Maximum	Configured	Configured	Configured	Configured	Configured
PG communication:	4	-	-	-	-	-
HMI communication:	4	4	5	3	6	0
S7 communication:	0	-	23	① 2	0	21
Open user communication:	0	-	129	9	112	8
Web communication:	2	-	-	-	-	-
Other communication:	-	-	-	-	-	-
Total resources used:		4	157	② 14	④ 118	29
Available resources:		6	89	114	⑤ 0	89

① All required resources for the respective communication types of the CPU are displayed here. Three (3) resources are required for HMI communication, 2 for S7 communication, and 9 for the connection types of the Open User Communication.

② The total number of resources required is listed below the required resources for the communication types. For the CPU, therefore, 3 + 2 + 9 = 14 resources are required.

③ The CP 1543-1 has 118 available connection resources. If these are utilized, no more connections can be created for the CP.

④ For the CP 1543-1, 118 connection resources are already utilized by the 2 HMI connections (3 resources each = 6 resources) and 112 resources by connections of the open communication.

⑤ The CP 1543-1 has no more free resources. No more connections can be created for the CP 1543-1.

The number of resources still available and the total number of resources used is also displayed for the whole station:

	Station resources			Module res...	Module res...	Module res...
	Reserved		Dynamic	PLC_2 [CPU ..	CP 1543-1_...	CM 1542-5_...
Maximum number of resources:	10		246	128	118	118
	Maximum	Configured	Configured	Configured	Configured	Configured
PG communication:	4	-	-	-	-	-
HMI communication:	4	4	5	3	6	0
S7 communication:	0	① -	23	2	0	21
Open user communication:	0	-	129	9	112	8
Web communication:	2	-	-	-	-	-
Other communication:	-	-	-	-	-	-
Total resources used:		② 4	② 157	14	118	29
Available resources:		③ 6	89	114	0	89

① All connection resources used (configured), including both reserved and dynamic connection resources, for the individual communication types for the entire station. The values represent the sum of all resources utilized for devices and modules (CPU, CP, CM).

② All connection resources used (configured), including both reserved and dynamic connection resources, for all communication types for the entire station.

③ Total number of reserved and dynamic connection resources still available. The values are obtained by subtracting the number of resources used from the maximum number of resources that can be configured.

Data consistency (S7-1500)

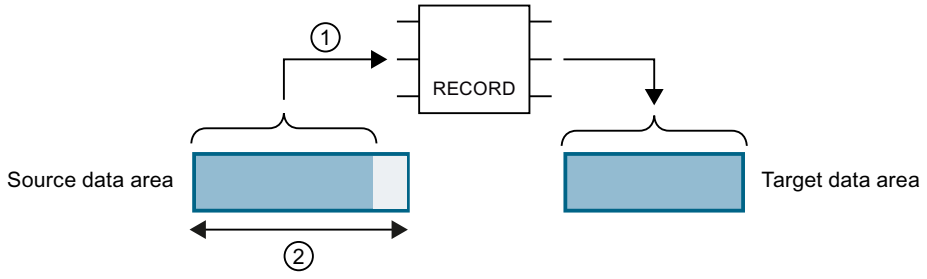
Definition

Data consistency is important for data transfer and you need to take this into account when configuring the communication task. Otherwise, malfunctions may occur.

A data area which cannot be modified by concurrent processes is called a consistent data area. This means that a data area which belongs together and which is larger than the maximum size of the consistent data area can consist in part of new and of old data at the same time.

An inconsistency can occur when an instruction for communication is interrupted, for example by a hardware interrupt OB with higher priority. This interrupts the transfer of the data area. If the user program in this OB now changes the data that has not yet been processed by the communication instruction, the transferred data originates from different times:

The following figure shows a data area that is smaller than the maximum size of the consistent data area. In this case, when transferring the data area, it is ensured that there is no interruption by the user program during data access so that the data is not changed.

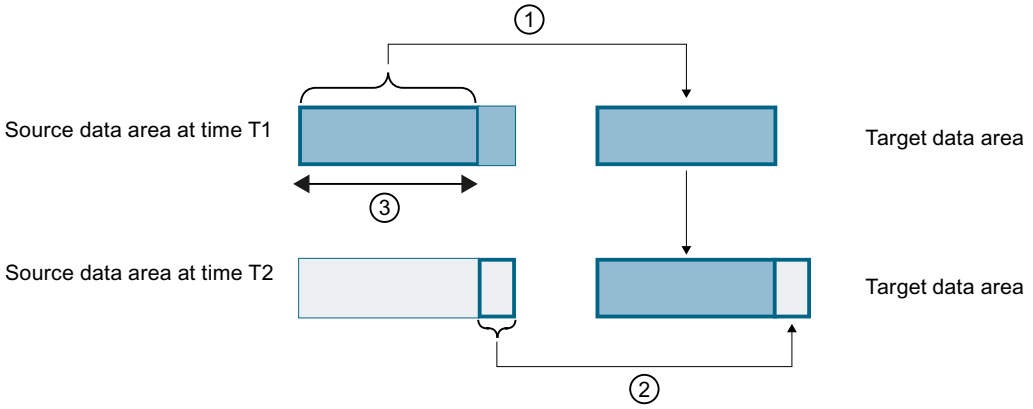


① The source data area is smaller than the maximum size of the consistent data area (②). The instruction transfers the data together to the destination data area.

② Maximum size of the consistent data area

Figure 10-1 Consistent transfer of data

The following figure shows a data area that is larger than the maximum size of the consistent data area. In this case, the data can be changed during an interruption of the data transfer. An interruption also occurs if, for example, the data area needs to be transferred in several parts. If the data is changed during the interruption, the transferred data originates from different times.



① The source data area is larger than the maximum size of the consistent data area (③). At time T1, the instruction only transfers as much data from the source data area to the destination data area as fit in the consistent data area.

② At time T2, the instruction transfers the rest of the source data area to the destination data area. After the transfer, data from different points in time exist in the destination data area. If the data in the source data area has changed in the meantime, an inconsistency may result.

③ Maximum size of the consistent data area

Figure 10-2 Transfer of data larger than the maximum consistency area

Example of an inconsistency

The figure below shows an example of changing data during the transfer. The destination data area contains data from different points in time.

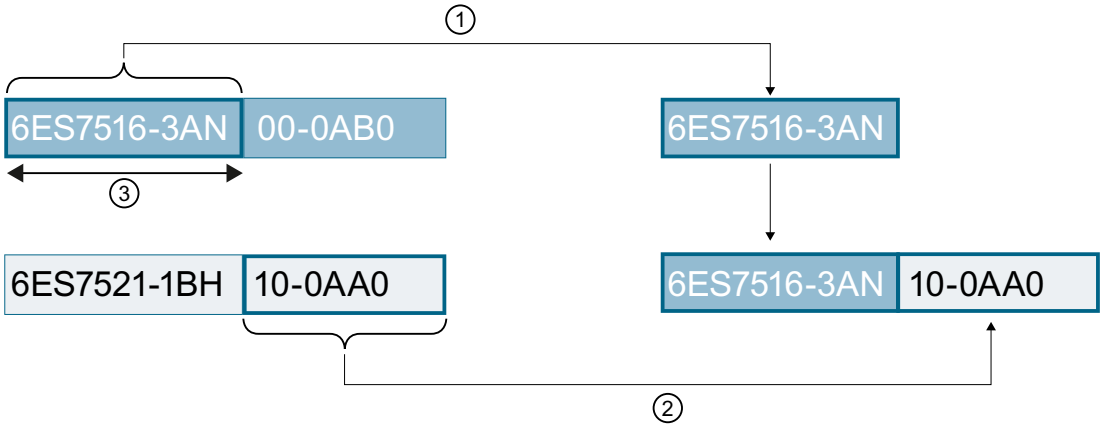


Figure 10-3 Example: Changing data during the transfer

System-specific maximum data consistency for S7-1500:

No inconsistency occurs if the system-specific maximum size of the consistent data is kept to. With an S7-1500, communication data is copied consistently into or out of the user memory in blocks of up to 512 bytes during the program cycle. Data consistency is not ensured for larger data areas. Where defined data consistency is required, the length of communication data in the user program of the CPU must not exceed 512 bytes. You can then access these data areas consistently, for example from an HMI device by reading/writing tags.

If more data than the system-specific maximum size needs to be transferred consistently, you yourself must ensure the data consistency with suitable measures in the user program.

Ensuring data consistency

Use of instructions for access to common data:

If the user program contains instructions for communication that access common data, for example TSEND/TRCV, you can coordinate access to this data area yourself, for example using the "DONE" parameter. The data consistency of the data areas that are transferred with an instruction for communication can therefore be ensured in the user program.

Note

Measures in the user program

To achieve data consistency, you can copy transferred data to a separate data area (for example global data block). While the user program continues to work with the original data, you can transfer the data saved in the separate data area consistently with the communication instruction.

For the copying, use uninterruptible instructions such as UMOVE_BLK or UFILL_BLK. These instructions ensure data consistency up to 16 KB.

Use of PUT/GET instructions or Write/Read via HMI communication:

In S7 communication with the PUT/GET instructions or Write/Read via HMI communication, you need to take into account the size of the consistent data areas during programming or configuration. In the user program of an S7-1500 as server, there is no instruction available that can coordinate the data transfer in the user program. The data exchanged using PUT/GET instructions updates the S7-1500 while the user program is running. There is no point in time within the processing of the cyclic user program at which the data is exchanged consistently. The length of the data area to be transferred should be smaller than 512 bytes.

Additional information

- You will also find the maximum amount of consistent data in the CPU, CM or CP manuals in the Technical Specifications.
- You will find further information on data consistency in the description of the instructions in the STEP 7 online help.

10.1.3.3 Displaying and configuring topology

Overview of the topology view

Functions of the topology view

The topology view is one of three working areas of the hardware and network editor. You undertake the following tasks here:

- Displaying the Ethernet topology
  - Displaying all the PROFINET devices and passive Ethernet components of the project along with their ports
  - Displaying interconnections between the ports
  - Displaying corresponding logical networks
  - Display diagnostic information of all ports
- Configuring the Ethernet topology
  - Creating, modifying and deleting interconnections of the ports
  - Renaming stations, devices, interfaces or ports
  - Adding PROFINET devices and passive Ethernet components to the project from the hardware catalog
- Identifying and minimizing differences between the desired and actual topology
  - Running an offline/online comparison of Ethernet modules, ports and port interconnections
  - Adopting existing online topology information in the offline project

Note

Devices without valid IP address

Topology information (LLDP) cannot be read from a device without valid IP address.

To prevent devices from having an invalid IP address, specify in the settings of the TIA Portal for hardware configuration that devices without valid IP address are temporarily assigned an IP address.

Note

Subnet membership

Before the topology determination, you have to set for the local interface of your PG/PC the network address that the Ethernet components that are to be determined also have in the real plant. If you do not meet this requirement, no further topology information (information about ports and neighborhood relationships) can be determined.

Differences between network view and topology view

- The network view shows all the logical subnets of the project.
  The topology view shows all Ethernet components of the project. These include passive components such as switches, media converters and cables.

  Note

  Stations with non-Ethernet components are also displayed if the station has a least one Ethernet component.

- The position of a device in the network view and its position in the topology view are initially independent of one other; in other words, the same device generally appears at different locations in the two views.
  If you want to align the devices in the topology view as in the network view, click on the following icon in the toolbar of the topology view.

Icon	Meaning
	Changes the position of the devices in the topology view so that the new position is as close as possible to the position in the network view.

- If you open the hardware catalog in the topology view, you only see devices with an Ethernet interface.

Structure of the topology view

The topology view (Page 536) essentially consists of a graphic area (called the graphic view below) and a table area (called the table view below).

Which functions are there in the graphic view and which functions are there in the table view?

- Displaying the Ethernet topology

Function	Graphic view	Table view
Displaying all the PROFINET devices and passive Ethernet components of the project along with their ports	yes	yes
Display interconnections between the ports (including type of medium)	yes	yes
Displaying corresponding logical networks	no	yes
Displaying properties of the cables between the ports	no	yes
Display diagnostic information of all ports	yes	yes

- Configuring the Ethernet topology

Function	Graphic view	Table view
Creating, modifying and deleting interconnections of the ports	• Create: yes • Modify: yes • Delete: yes	• Create: yes • Modify: yes • Delete: yes
Renaming stations, devices, interfaces or ports	no	yes
Adding PROFINET devices and passive Ethernet components to the project from the hardware catalog	yes	no

- Identifying and minimizing differences between the desired and actual topology

Function	Graphic view	Table view
Running an offline/online comparison of Ethernet modules, ports and port interconnections	no	yes
Adopting existing online topology information in the offline project	no	yes

Structure of the table view

The table view consists of the following two tabs:

- "Topology overview" tab
- "Topology comparison" tab

The "Topology overview" tab provides the following functions:

- Display of the configured topology with hierarchical structures (from the station to the port)
- Configuration of the Ethernet topology
- The display of the diagnostics status of hardware components

The "Topology comparison" tab provides the following functions:

- Identifying and minimizing differences between the desired and actual topology

Starting the topology view

Requirement

The device or network view is open in the hardware and network editor.

Procedure

To start the topology view of your project, follow these steps:

1. Click on the "Topology view" tab.

Or:

1. Open the network view of the hardware editor.
2. Select a PROFINET device or a PROFINET module.
3. Select the "Go to topology view" command in the shortcut menu.

Result

The topology view is started. If you opened the topology view using the shortcut menu, the selected component remains selected after the change of view.

Displaying topology

Displaying the graphic view of the configured topology

What is shown?

The graphic view of the configured topology shows the following:

- Configured PROFINET devices and passive Ethernet components along with their ports
- Configured stations with non-Ethernet components if there is at least one Ethernet component in the station
- Configured interconnections between the ports

Type of display

The way in which the graphic view of the topology view and the network view are displayed is very similar:

- Compared with the device view, components are shown in a simplified form.
- The interconnections between ports are shown as horizontal and vertical lines. These are dashed when an interconnection between a tool changer port and its possible partner ports is involved.

Displaying the table view of the configured topology

Requirement

- The table view of the topology is open.
- You are in the "Topology overview" tab

What is shown?

The table view of the configured topology shows exactly the same content as the graphic view except for the logical PROFINET subnets:

- All the configured PROFINET devices and passive Ethernet components along with their ports
- All the configured stations with non-Ethernet components if there is at least one Ethernet component in the station
- Configured interconnections between the ports
  For each port with the "Alternating partner port" property, there are as many completed rows as there are potential partner ports plus one empty row.

Type of display

As the name implies, the table view of the topology view consists of a table, the topology overview table. It displays the configured topology with hierarchical structures (from the station to the port) It is structured like the network overview table. It consists of the following columns:

- Device / port
  This is the most important column of the table. The entries in this column have a hierarchical structure with the PROFINET ports being the last element in the hierarchy. You can expand and collapse the hierarchical entries. For a CPU, for example, an entry consists of the following elements:
  - Station name
  - Device name
  - Name of the PROFINET interface
  - Names of the ports

  Note: All the other columns only have entries in the rows containing the port names.
- Slot
  Indicates the number of the slot in a rack in which the relevant module is installed. For interfaces, the designation of the interface is also displayed.
- Index (by default, this column is not displayed)
  This column is only relevant for PC stations. The index is the virtual slot address of the component.
- Device type (by default, this column is not displayed)
  Shows what type of station, device or interface the table row relates to or whether it belongs to a port.
- Article no. (by default, this column is not displayed)
  Article no. of the device
- Subnet (as default, this column is not displayed)
  Configured subnet to which the interface belongs
- Master/IO system (as default, this column is not displayed)
  Shows whether or not the interface belongs to a PROFIBUS DP master system or a PROFINET IO system.

- IP address (by default, this column is not displayed)
  Configured address of the interface in the subnet
- Partner station
  Name of the station that contains the partner port
- Partner device
  Name of the device that contains the partner port
- Partner interface
  Interface to which the partner port belongs
- Partner port
- Cable data
  Contains the cable length and the signal delay of the cable connecting the ports

Basic functions for tables

The topology overview table supports the following basic functions for editing a table:

- Displaying and hiding table columns
  Note: The columns relevant for configuration cannot be hidden.
- Optimizing column width
- Saving current representation
- Displaying the meaning of a column, a row or cell using tooltips.

Displaying the diagnostics status of ports and cables in the graphic view

Requirements

The graphic view of the topology view is open.

Procedure

To determine the diagnostics status of the port, follow these steps:

1. Go online with the required component or components.

Result

The following icons are displayed:

- The corresponding diagnostics icon is displayed for each device.
- If there is an error in at least one lower-level component, the diagnostics icon "Error in lower-level component" is also displayed in the left-hand lower corner of the diagnostics icon.
- The corresponding diagnostics icon is displayed for each port.
- Every cable between two ports that are online has the color that matches its diagnostics status.

You will find the possible diagnostics icons for ports and the color coding of Ethernet cables in the description of hardware diagnostics. See: AUTOHOTSPOT

Showing the diagnostics status of hardware components in the table view

Requirement

- The table view of the topology view is open.
- You are in the "Topology overview" tab

Procedure

To obtain the diagnostics status of hardware components of the topology overview table, follow these steps:

1. Go online with the required components.

Result

The following icons are displayed at the left-hand edge of the topology overview table in each row that belongs to the component involved:

- The diagnostics icon belonging to the hardware component is displayed.
- If the hardware component has lower-level components and if there is an error in at least one of the lower-level components, the diagnostics icon "Error in lower-level component" is also displayed in the left-hand lower corner of the diagnostics icon of the hardware component.

For the possible diagnostics icons for hardware components, refer to the description of hardware diagnostics. See: AUTOHOTSPOT

Note

The display of the diagnostics status of hardware components in the topology overview table and the network overview table is identical.

Running an offline/online comparison and displaying the results

Requirements

- The table view of the topology view is open.
- You are in the "Topology comparison" tab
- There may be an online connection to one or more devices, but this is not actually necessary.

Procedure

To find the differences between the configured and the actual topology, follow these steps:

1. Click the "Compare offline/online" button in the toolbar.

Result

The actual topology is identified and compared with the configured topology.

Devices identified online are automatically assigned to configured devices as far as this is possible. See Automatic assignment of devices by offline/online comparison (Page 717)

The columns on the right-hand side of the table are filled or updated:

- Columns for the result of the offline/online comparison: "Status" and "Action"
- Columns for the topology identified online

Note

Stopping a running online/offline comparison

If you click the "Compare offline/online" button while topology discovery is active, the current topology discovery is stopped.

The following buttons are enabled in the toolbar of the table:

Button	Name	Meaning
	Update	The detection of the existing online topology is started again.
	Synchronize	• Adopting the port interconnections identified online in the project (Page 718) • Adopt the devices identified online in the project (Page 718)

Note

If you have not configured your own programming device or your own PC offline, the gray filled circle is displayed as a diagnostic symbol.

You can also configure a second and third programming device or a second and third PC offline.

Columns for the result of the offline/online comparison

The following columns are displayed:

- "Status"
  The result of the offline/online comparison is shown here in the form of diagnostics icons. The following icons are possible:

Diagnostics icon	Meaning
	Differing topology information in at least one lower-level component
	Identical topology information
	Topology information only available offline or device is disabled
	Topology information only exists online
	Differing topology information
	No port interconnection configured, no port interconnection discovered online
	If a device does not support topology functions, the "Status" column remains empty.

- "Action"
  The possible actions are shown here. The following actions are possible:
  - No action
  - Apply

Note

Separating columns

To separate the offline and online columns of the topology comparison table optically, there is a separating column to the left of the "Status" column and to the right of the "Action" column. The separating columns cannot be hidden and their width cannot be changed.

Columns for the topology identified online

The following columns are displayed:

- "PROFINET device name"
- "Article no." (by default, this column is not displayed)
- "Device family" (by default, this column is not displayed)
- "IP address" (by default, this column is not displayed)
- "Port"
- "Interconnection"
  - Green line if the port is interconnected
  - Gray line if the port is not interconnected
- "Partner port"
- "Partner IP address"
- "Partner PROFINET device name"

- "Partner article number" (by default, this column is not displayed)
- "Partner device family" (by default, this column is not displayed)

Basic functions for tables

The topology comparison table supports the following basic functions for editing a table:

- Displaying and hiding table columns
  Note: The columns relevant for configuration cannot be hidden.
- Optimizing column width
- Saving current representation
- Displaying the meaning of a column, a row or cell using tooltips.

Running an advanced offline/online comparison and displaying the results

Overview

With the standard offline/online comparison, a search method based on the DCP protocol is used.

With the advanced offline/online comparison, ICMP is also used to be able to detect devices that do not support DCP. These include, for example, PG/PC stations.

Requirement

- The table view of the topology view is open.
- You are in the "Topology comparison" tab
- The devices to be detected with ICMP are located in the same subnet as the network adapter of your PG/PC you are currently using.

Procedure

Follow these steps:

1. In the toolbar, click on the arrow in the "Advanced comparison" button.
2. With the input boxes "Start address" and "End address", select an IP address range in which accessible Ethernet nodes will be searched for with ICMP. Confirm this, for example by clicking the button with the green check mark.
3. Click on the icon in the left-hand part of the "Advanced comparison" button.

Result

The Ethernet nodes detectable with DCP and ICMP are discovered.

The columns on the right-hand side of the table are filled or updated:

Configuring topology

Interconnecting ports

Overview

Interconnecting ports in the topology view

In the topology view, you have the following options for interconnecting ports:

- in the graphic view (Page 713)
- in the graphic view of a tool changer (Page 715)
- in the table view (Page 714)
- in the table view of a tool changer (Page 715)
- by adopting port interconnections identified online (Page 718)

Note

Interconnecting an electric with an optical port

If you want to interconnect an electric and an optical port, you have to decide between RT and IRT communication:

- With RT communication, it is not necessary to configure a media converter.
- With IRT communication, you have to make the interconnection via a media converter.

What effects does the interconnection of ports have on the network view?

Note

In the properties of a subnet in the network view, you can specify that this subnet is used when a port interconnection is created between two devices that are not networked.

When you create an interconnection between two ports, the following effects are possible in the network view:

- If the corresponding interfaces are disconnected: If you have specified a default subnet, this is used. Otherwise a new subnet is created to connect the two interfaces.
- If one (and only one) of the two interfaces involved is networked: The non-networked interface is connected to the same subnet as the already networked interface.
- In all other cases: The corresponding interfaces are not connected to a logical subnet.

See also

Interconnecting ports (Page 1787)

Settings for interconnecting Ethernet devices (Page 603)

Interconnecting ports in the graphic view

Requirement

You are in the graphic view of the topology view.

Procedure – Creating a new interconnection between two ports

To interconnect a port of a device with a port of another device, follow these steps:

1. Place the mouse cursor on the port you want to interconnect.
2. Click with the left mouse button and hold it down.
3. Move the mouse pointer.
   The pointer now shows the networking symbol to indicate "Interconnect" mode. At the same time, the mouse pointer shows the lock symbol that will only disappear when the pointer is on a valid target.
4. Now drag the mouse cursor to the target port. You can do this while holding down or after releasing the mouse button.
5. Now release the left mouse button or press it again (depending on your previous action).

Result: A new port interconnection is created.

Note

Creating a ring for S7-300, S7-400, and S7-1500 CPUs

If you create a ring using port interconnections for S7-300, S7-400, or S7-1500 CPUs, an MRP domain is created automatically.

Procedure – Changing an existing port interconnection without deleting it first

To do this, follow these steps:

1. Place the mouse cursor on the port of an existing interconnection that is to receive a new partner port.
2. Drag the port to the new partner port.

Result: The existing port interconnection is deleted. The new port interconnection is created.

Alternative procedure:

1. Place the mouse cursor on a port without an interconnection that is to be connected with an already interconnected port.
2. Drag the port to the already interconnected port.

Result: The existing port interconnection is deleted. The new port interconnection is created.

Procedure – Interconnecting two interconnected ports with each other without first deleting the two existing port interconnections

To do this, follow these steps:

1. Place the mouse cursor on the interconnected port that is to receive a new partner port.
2. Drag the port to the new partner port that has also already been interconnected.

Result: The two existing port interconnections are deleted. The new port interconnection is created.

Interconnecting ports in the table view

Which actions are possible with port interconnections in the table view?

The following actions are possible with port interconnections in the table view:

- Creating a new port interconnection
- Changing an existing port interconnection
- Deleting an existing port interconnection

Requirement

The row with the port whose interconnection you want to create, modify or delete is visible in the "Topology overview" tab.

Procedure

To create the interconnection of a port for the first time, to modify it or delete it, follow these steps:

1. Move the mouse pointer to the "Partner port" column in the row of the source port.
2. Click the drop-down list there.
3. Select the required partner port (when creating or changing a port interconnection) or the "Any partner" entry (when deleting a port interconnection).

Result

The required action is performed. The new partner port (after creating or modifying a port interconnection) or the "Select port" entry (after deleting a port interconnection) is displayed in the "Partner port" column.

Interconnecting a port with more than one partner port in the graphic view

Requirement

- You have configured a port of a PROFINET device with the "Alternative partners" property and have specified its possible partner ports.
- The graphic view of the topology view is open.

Procedure

1. Interconnect this port (referred to hereafter as source port) with one of the partner ports you have specified (referred to hereafter as target port).
2. Interconnect the source port with an additional target port.
   You can do this in several ways:
   - Drag the mouse pointer from a partner port that is already interconnected to a target port.
   - Drag the mouse pointer from an interconnection that has already been created to a target port.
   - Drag the mouse pointer from a target port to a partner port that is already interconnected.
   - Drag the mouse pointer from a target port to an already created interconnection.
3. If necessary, repeat the step above one or more times.

Result

An interconnection is created between the source port and the alternative partner ports. This is indicated by a dashed line.

Interconnecting a port with more than one partner port in the table view

Which actions are possible with port interconnections to several partner ports in the table view?

When working with a tool changer, the following actions can be performed with port interconnections to multiple partner ports in the table view:

- Creating a new port interconnection
- Changing an existing port interconnection
- Deleting an existing port interconnection

Requirement

- You have configured a port of a PROFINET device with the "Alternative partners" property and have specified its possible partner ports.
- The row with the port whose interconnection you want to create, modify or delete is visible in the "Topology overview" tab.

Procedure

To create the interconnection of a port to one or more partner ports for the first time, to modify it, or to delete it, follow these steps:

1. Move the mouse pointer to the "Partner port" column in the row of the source port.
2. Click the drop-down list there.
3. Select the required partner port (when creating or changing a port interconnection) or the "Any partner" entry (when deleting a port interconnection).

Result

The required action is performed:

- If you are creating an interconnection, a new row is inserted in the topology overview table. The new partner port is displayed there in the "Partner port" column.
- If you change an interconnection, the new partner port is displayed in the "Partner port" column.
- If you delete an interconnection, the row with the previous port interconnection is deleted.

Note

With a tool changer, there are normally several rows for a port with port interconnections to more than one partner port. The last row is always an empty row. The first row can be edited, all other rows are read-only.

Renaming stations, devices, interfaces or ports

Rename a station, a device, an interface or a port

Requirement

- The table view of the topology view is open.
- You are in the "Topology overview" tab

Procedure

To rename a station, a device, an interface or a port, proceed as follows:

1. Click twice in the relevant field of the topology overview table (the second click starts the editing mode).
2. Enter the new name and then press the ENTER key (this closes editing mode).

Result

The object is renamed.

Offline/online comparison

Automatic assignment of devices by offline/online comparison

Overview

During the offline/online comparison, the configured topology is compared with the actual existing topology. Devices identified online are automatically assigned to configured devices as far as this is possible.

Start of availability detection

You start the availability detection the first time by clicking the "Compare offline/online" button in the toolbar of the "Topology comparison" tab.

You restart availability detection by clicking the "Update" button.

Note

The availability detection can take several seconds. During this time, no user input is possible.

Automatic assignment of a PNIO device

A PNIO device identified online is automatically assigned to a configured device if the following properties of the two devices match up:

- Article no.
- Device type or device family
- PROFINET device name

No automatic assignment

In the following situations, no automatic assignment is possible:

- No device can be identified online to match a configured device. (This means that the corresponding columns in the "Online topology" area of the topology comparison table are empty.)
  In this case, you should add the already configured device to your system or delete the configured device from the configuration.
- A device identified online can be assigned to a configured device, but there are differences in the port interconnections.
  In this case, you can apply the port interconnections identified online manually to the project (Page 718).
- A device identified online cannot be assigned to a configured device. (In this case, the corresponding columns in the "Offline topology" area of the topology comparison table are empty.)
  In this case, you can include the device identified online manually in the project (Page 718).

Apply the port interconnections identified online manually to the project

Requirement

You have run an offline/online comparison in the topology view. The result of this is that at least one device identified online was automatically assigned to a configured device, but that there are differences relating to the interconnection.

Procedure

To adopt one more port interconnections identified online in the project manually, follow these steps:

1. Select the row belonging to the port interconnection.
2. If applicable, select further roles using multi-selection.
3. Select "Apply" > "Use selected" in the shortcut menu.
   The content of the corresponding table cells in the "Action" column changes to "Apply".
4. If you have mistakenly prepared too many port interconnections to be included in the project:
   Select the rows that belong to the port interconnections you have mistakenly prepared for inclusion in the project using multi-selection.
   Select "Reset" > "Reset selected" in the shortcut menu.
   The content of the corresponding table cells in the "Action" column change to "No action".
5. Click the "Synchronize" button.

Result

The port interconnections identified online for the corresponding devices are included in the project. Successful adoption is indicated by the diagnostics icon "Identical topology information" for each port.

Note

If other port interconnections are recognized for a device identified online and these differ from those that exist in the project, adopting these in the project means that the port interconnections that were previously in the project are replaced by those identified online. If no port interconnections are detected for a device identified online, adopting in the project means that all the port interconnections of this device are deleted in the project.

Include the devices identified online manually in the project

Requirements

You have run an offline/online comparison in the topology view. The result of this is that at least one device identified online could not be assigned to any configured device.

Procedure

To adopt one more devices identified online in the project manually, follow these steps:

1. For a configured device without an online partner, move the mouse pointer to the "Device/ port" column of the online topology.
2. Select the device you want to assign to the configured device from the drop-down list of this box.
3. Repeat the previous steps if necessary for other configured devices without an online partner.

Result

The selected device that was identified online is moved up from the end of the table. Following this, it is in the row of the configured device to which you have just assigned it.

Automatic assignment of devices by advanced offline/online comparison

Overview

With the advanced offline/online comparison, ICMP is also used alongside DCP to be able to detect devices that do not support DCP.

Automatic assignment of devices detected by ICMP

With devices detected by ICMP, no type is available.

With passive devices, no article number is available. For this reason, passive devices can only be assigned automatically if you have not assigned an article number in the configured data and the offline and online IP addresses match.

With switches, automatic assignment is possible if the offline and online article number, IP address and PROFINET device name match.

10.1.3.4 Industrial Ethernet Security

Configuring security

General

Supported devices

Supported devices

The security functions described in this help section are supported y the following products;

- SCALANCE S:
  - S602 V2*/V3/V4
  - S612 V2*/V3/V4
  - S613 V2*
  - S623 V3/V4
  - S627-2M V4
- SOFTNET Security Client:
  - SOFTNET Security Client V4
- S7 CPs / TIM: CP 343-1 GX31 Advanced, CP 443-1 GX30 Advanced, CP 443-1 OPC UA, CP 1543-1, CP 1543SP-1, CP 1243-1 BX30, CP 1243-1 PCC, CP 1242-7 KX31, CP 1243-7 LTE, CP 1243-8 IRC, TIM 4R-IE, TIM 4R-IE DNP3, TIM 1531 IRC
- PC CP: CP 1628
- Mobile wireless router: SCALANCE M875

* SCALANCE S V2 devices cannot be inserted in a STEP 7 project. SCALANCE S V2 devices from projects of older STEP 7 versions are simply displayed with their properties. The properties of these devices can, however, not be changed. Downloading and compilation and diagnostics of SCALANCE S V2 devices is also not possible. All the functions described for SCALANCE S devices relate to the firmware versions as of V3.

When opening a STEP 7 project with existing SCALANCE S V2 devices global firewall rules can be created from the existing local firewall rules for the device.

General terminology "security module"

In this section of the information system the name "security module" is used instead of the product names named above. You will find the functional differences between the products in the Siemens Industry Mall.

Use of the terms "interface" and "port"

In this documentation, the ports of SCALANCE S modules are named as follows:

- "External interface": The external port of the SCALANCE S602 / S612 / S613 / S623 or an external port of the SCALANCE S627-2M (marked red)
- "Internal interface": The internal port of the SCALANCE S602 / S612 / S613 / S623 or an internal port of the SCALANCE S627-2M (marked green)
- "DMZ interface": The DMZ port of the SCALANCE S623 / S627-2M (marked yellow)

The term "port" itself is used when the focus of interest is a special port of an interface.

Structure of this section of the help system

Topics that are relevant to several module types can be found in the "General" section. Information that is only relevant to a certain module type can be found in the sections relating specifically to these module types.

Overview - Scope of performance and method of operation

General use of the term "STEP 7"

Configuration of security functions is supported as of STEP 7 V12. For this reason, in this section of the information system, the name "STEP 7" will be used for all versions of STEP 7 V12 or higher.

Scope of performance

You can use the following security functions in STEP 7:

- Configuration of the security modules
- Creating VPN configuration data for SOFTNET Security Client V4
- Creating VPN configuration data for SCALANCE M875
- Creating VPN configuration files for VPN devices from third-party manufacturers
- Test and diagnostics functions, status displays

Offline configuration view and online diagnostics view

The security functions are configured in two views:

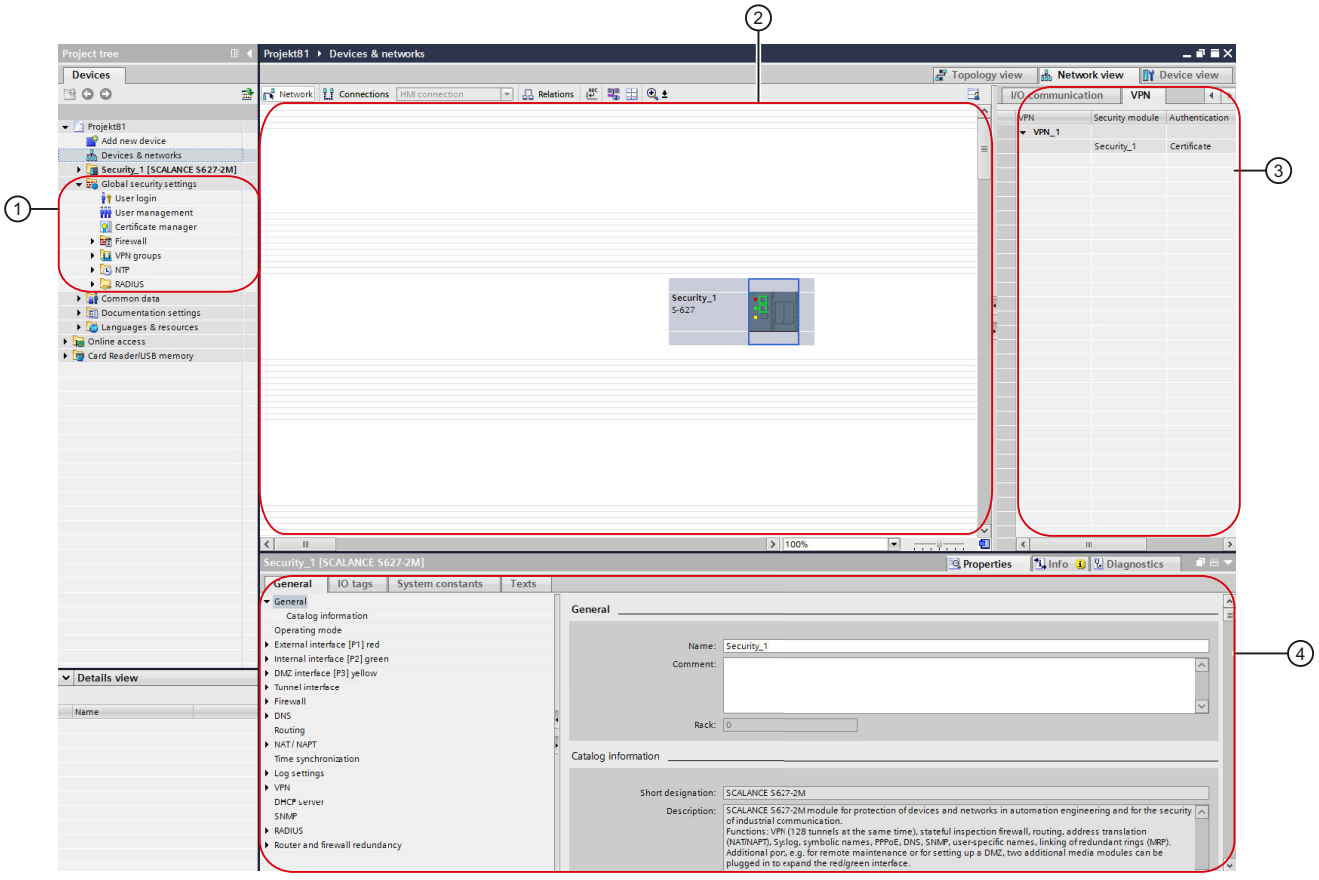
- Offline configuration view
  In the offline configuration view, you create the configuration data. Prior to downloading, there must already be a connection to the security modules.
- Online diagnostics view
  The online/diagnostics view is used for diagnostics of the security module and, among other things, allows you to run firmware updates.

How it works - security and consistency

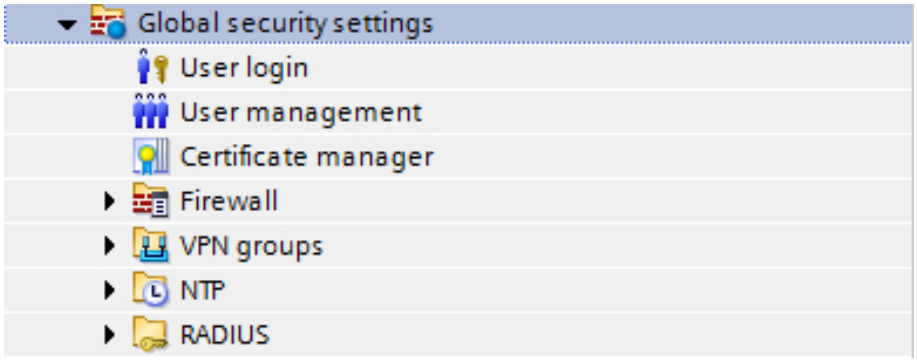
- Access only for authorized users
  The security functions of every project are protected from unauthorized access by assigning user names and passwords. With the help of password policies, project-specific rules for password assignment can be defined.
- Consistent project data
  Consistency checks are running even while you make the entries in the dialogs. In addition, cross-dialog, project-wide consistency checks are carried out.
  Only consistent project data can be downloaded to the security modules.
- Protecting project data by encryption
  The project and configuration data relevant to security are protected by encryption.
  Depending on the security module, data can be stored in the project and/or on the C-PLUG.

User interface - layout and menu commands

User interface for security functions in STEP 7



① Global security settings



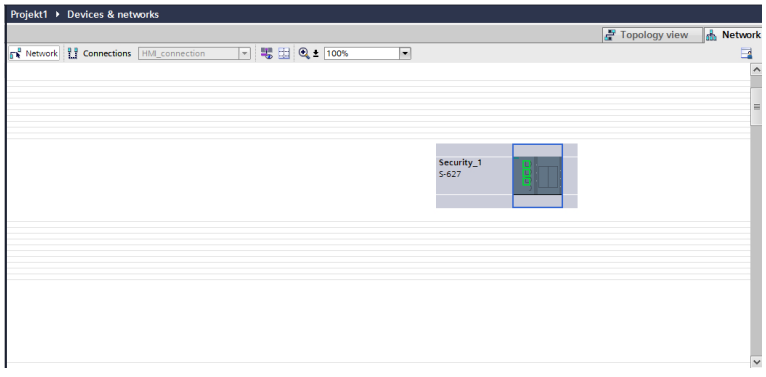
The global security settings are located in the project navigation. These security settings can be configured independently of the module and subsequently assigned to individual security modules as required. Changes to the global security settings must be loaded on all security modules involved. This also applies to the settings of redundancy relationships.

If the first security module to be configured is a CP, the global security settings are only displayed when the security functions have been enabled in the local security settings of the CP. If the first security module to be configured is

a SCALANCE S module, a SCALANCE M module or a security device the global security settings are displayed after logging in to the security project. The following main folders and entries are available in the global security settings:

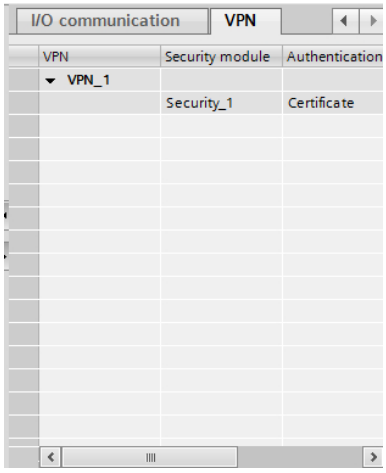
- User login
  For the security configuration within a project, there is a separate user management. Log in to the security configuration using the "User login" entry. The first time that there is a login to the security configuration, a user with the system-defined role "Administrator" is created automatically. You can create further users in the security configuration in the user management.
- User administration
  In user administration, you can create users, define rights for roles and assign these roles to users.
- Certificate manager
  In the certificate manager, you see an overview of all the certificates used in the project. You can, for example, import new certificates as well as export, renew or replace existing certificates.
- Firewall
  Under the "Firewall" entry, you can define global IP and MAC firewall rule sets and user-specific IP rule sets (SCALANCE S modules only) and assign security modules. Global firewall rule sets can also be exported from STEP 7 and imported into STEP 7. IP and MAC service definitions are used to define the IP and MAC firewall rules compactly and clearly.
- VPN groups
  All created VPN groups are contained in this folder. You can create new VPN groups here and assign security modules to these VPN groups. You can also adapt VPN group properties of VPN groups that have already been created.
- NTP
  Here, you can create NTP servers and assign them to one or more security modules. This ensures that time synchronization is performed through the assigned NTP server. Unsecured NTP servers can only be configured in the local security settings.
- RADIUS
  Here, you can create RADIUS servers and assign them to one or more security modules. With this, you ensure that authentication queries from users who log on to the selected security module to activate user-specific IP rule sets are forwarded to the assigned RADIUS server.

② Work area



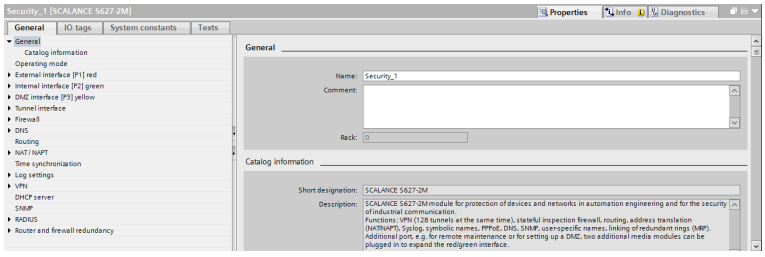
Once you have selected a SCALANCE M module or a security module in the work area, you can configure its local security settings in "Properties" > "General". If the selected object is in a VPN group, related information is displayed in the VPN tab.

③ VPN tab



This tab displays information about all the VPN groups to which the security module that was selected in the working area belongs. Information about the respective participants of a VPN group can be displayed and hidden.

④ Local security settings



Local security settings are configured for a specific security module / SCALANCE M module / security device. After one of these objects has been selected in the working area, its local security settings are available in the Inspector window under "Properties" > "General".

Note for CPs:

Before local security settings can be configured for CPs, these must first be enabled.

To do this, log in to your security project and then in the Inspector window, select the "Activate security features" check box in the "Properties" > "General" tab, "Security" entry. The local security settings are then displayed below the "Security" entry. When the check box is selected, the some of the CP settings (assuming they were enabled) are migrated automatically to the local security settings.

Depending on the particular security module, additional security functions are also available such as NTP (secure), SNMPv3, FTPS.

In addition, firewall rules that enable a connection to be established are created automatically for configured connections. Log settings are available to record blocked packets.

Secure and non-secure configuration areas

The user interface can be divided into secure and non-secure configuration areas.

The secure areas are areas in which configuration is possible only after logging in to the security configuration. These areas are encrypted and therefore only available to persons authorized in the user management even if the project is accessible to a wider circle of people.

Functions from the non-secure areas, on the other hand, can be configured without logging in to the security configuration. The correctness of the settings must be checked before downloading the project to the plant components if a wider circle of people can make modifications to the project.

Below, you will find a list of the configuration areas of the user interface showing which areas are secure and which are non-secure. To some extent, this depends on the security module for which the configuration is created.

- All settings from the global security settings are secure.
- Secure and non-secure configuration areas for SCALANCE S modules:
  - All the settings for the interfaces and ports, in particular IP addresses, are non-secure.
  - The settings under the entry "General" in the local security settings are non-secure.
  - Higher-level settings (e.g. MRP settings such as MRP manager etc.) that are not configured on the security module itself but may affect the security module are non-secure. This does not relate to the global security settings.
  - The other settings are protected.
- Secure and non-secure configuration areas for CPs:
  - All settings outside the "Security" entry are non-secure.
  - Higher-level settings (e.g. MRP settings such as MRP manager, PROFINET settings, connections etc.) that are not configured on the security module itself but may affect the security module are non-secure. This does not relate to the global security settings.
  - All the settings for the interfaces and ports, in particular IP addresses, are non-secure.
  - All settings below the "Security" entry are secure.

Restrictions in the use of libraries

If a master copy is created in the global library for a security module, the configuration of this security module is lost in the master copy. You need to log in again if you use a master copy.

If a master copy is created in the project library for a security module, the configuration of this security module is retained, apart from the cross-module information for VPN and redundancy relationships.

IP addresses for configurable servers

When configuring servers, the (alias) IP addresses of the security module must not be used. For all configurable servers, there is a consistency check when the IP address is assigned.

Running a consistency check

Overview

The following consistency checks are available:

- Local consistency checks
- Project-wide consistency checks

In the individual dialog descriptions in this help system, the rules you need to take into account for your entries are listed under the keyword "Consistency check".

Local consistency checks

A consistency check is local when it can be performed directly within a dialog. With the following actions, local consistency checks are made:

- After exiting a box
- After exiting a row in a table
- When you confirm a dialog with OK.

Project-wide consistency checks

Project-wide consistency checks provide you with information indicating whether or not project data is correctly configured. With the following actions, there is a consistency check through the entire project:

- When compiling a configuration
- When downloading a configuration

Note

You can only download configured data to a security module if the project-wide consistency check for the security module was successful.

Replacing a security module

Module-specific function

Only SCALANCE S modules as of V3 can be replaced by SCALANCE S modules as of V3, refer to the section:
Replacing a security module (Page 838) in the section "SCALANCE S".

Managing certificates

Overview of certificates

Certificate management

The certificate manager in the global security settings contains an overview of all the certificates used in the project, for example, CA certificates with information about the certificate owner, issuer, validity, use and the existence of a private key.

A CA certificate is a certificate issued by a certificate authority from which the device certificates are derived. The device certificates include SSL certificates. SSL certificates are required for authentication during secure communication between a network node and a security module. Further device certificates include the VPN group certificates of security modules located in VPN groups and OPC UA client/server certificates of CP 443-1 OPC UA modules.
Certificate authorities can be:

- The STEP 7 project itself. If the "certificate owner" and "issuer" are the same, this is a self-signed certificate; in other words, issued by the STEP 7 project.
- A higher ranking (commercial) certificate authority. These third-party certificates are external to the project and are imported and stored in the certificate store of STEP 7.

Certificates created by one of the two certificate authorities always have a private key so that the device certificates can be derived from them.

Every certificate is assigned an ID via which the certificate can be referenced in the program blocks. The IDs are generated consecutively when certificates are created. Due to the system some IDs are not used.

Note

Downloading the configuration

After replacing or renewing certificates, the configuration must be downloaded to the relevant security modules.

After replacing or renewing CA certificates, the configuration must be downloaded to all security modules.

Note

Current date and current time of day on the security modules

When using secure communication (for example, HTTPS, VPN...), make sure that the security modules involved have the current time of day and the current date. Otherwise the certificates used are not evaluated as valid and the secure communication does not work.

Note

Character set of certificates

Certificates are always stored in the ASCII character set. If a certificate contains characters that do not exist in the ASCII character set, the certificate is stored with the UTF-8 character set. This is not possible for SCALANCE S V2 modules.

Global certificate manager

The following functions are available in the certificate manager in the global security settings:

- Import of new certificates and certificate authorities.
- Import of SSL certificates (343-1 Adv. / CP 443-1 Adv. only), e.g. for FTP communication.
- Export of the certificates and certificate authorities used in the project.
- Renewal of expired certificates and certificate authorities.
- Replacing existing certificate authorities.
- Addition of trusted certificates and certificate authorities.
- Deleting manually imported certificates.

Double-click on the "Certificate manager" entry in the global security settings.

In the individual tabs, you have the following commands available in the shortcut menu:

Command	Meaning
Import / Export	Import / export of certificates. The following formats are permitted: *.cer (certificate only) *.crt (certificate only) *.pem (certificate only) *.p12 (certificate and corresponding private key)
Displays	Opens the certificate dialog of Windows where you see an overview of all certificate data. For the certificate to be displayed as valid, install it via the "Install certificate..." button of the "General" tab in the Windows certificate store. Appropriate Windows user rights are required to install the certificate.
Renew (only in the "CA" and "Device certificates" tabs)	Opens the "Renew certificate" dialog in which you can have a new certificate created by STEP 7 when necessary, for example with compromised certificates.
Replace (only in the "CA" tab)	Opens the "Replace certificate" dialog in which you can replace an existing certificate with a new one.
Delete	Deletes a certificate in the "Trusted certificates and root certification authorities" tab.

Local certificate manager

Certificates that were imported via the certificate manager in the global security settings are not automatically assigned to the corresponding modules. Imported certificates must be included in the list of trustworthy partner certificates manually via the entry "Certificate manager" in the local security settings. Certificates generated by STEP 7 such as SSL certificates or VPN group certificates are automatically assigned to the corresponding modules and do not need to be assigned using the local security settings.

Before certificates can be referenced in the program block for Secure Communication, these certificates must be assigned to the security module as device certificates via the local certificate manager. After selecting a table cell, you can either create a new certificate using the "Add" button or use an existing certificate of the project using the check mark symbol. For access to existing certificates of the project and for signing a new certificate by a certification authority, the check box "Use global security settings for the certificate manager" must be activated. To make the assigned certificate known to the partner module, this certificate must be entered in the list of trustworthy partner certificates after import.

Certificate authorities

"CA" tab

The following certification authorities are displayed

- CA certificates of a project: When you create a new project, CA certificates are generated for the project that use different hash algorithms for the certificate signature. CA certificates for projects are always loaded on the security modules belonging to them. The SSL certificates and OPC UA client/server certificates for the individual security modules are derived from the CA certificates.
- CA group certificates: When you create a new VPN group, a CA certificate is generated for the VPN group. The VPN group certificates of VPN group subscribers are derived from this certificate..

Device certificates

"Device certificates" tab

Display of the device-specific certificates generated by the certification authorities for modules. These include:

- SSL certificate of a module: An SSL certificate that is derived from a CA certificate of the project is generated for each module that you create. SSL certificates are used for authentication during secure communication between PG/PC and module when downloading the configuration.
- OPC UA client/server certificate of the module: Depending on the configured OPC UA client/server function of the CP 443-1 OPC UA, an OPC UA client/server certificate is generated that is used for authentication with the relevant communications partner.
- VPN group certificate of a module: A VPN group certificate is also generated for each module for each VPN group in which it is located.

Trusted certificates and root certification authorities

"Trusted root certification authorities" tab

Display of the third-party certificates imported into STEP 7. Server certificates can be imported for example from external FTP servers, project certificates from other STEP 7 projects and certificates required for the external services such as dyn.DNS.

After the import, certificates must be assigned to the corresponding modules using the local security settings. For further information, refer to the section Overview of certificates (Page 728).

Renewing certificates

Meaning

In this dialog, you renew CA certificates and device certificates.

How to access this function

1. Right-click on a list entry in the certificate manager.
2. Select the "Renew" entry in the shortcut menu.
3. Decide whether or not the new certificate will be self-signed or signed by a certificate authority.
4. If the certificate is to be signed by a certificate authority, select the certificate authority to be used with the "Select" button. Only certificate authorities stored in the certificate store of the current project can be selected.
5. Depending on the certificate, enter the following parameters in the corresponding input fields:

Parameter	Meaning
Common name of subject	Holder of the certificate. The common name of subject can contain no more than 35 characters. The displayed ID is not included in this count.
Signature	Algorithm used by the certification authority for signing the certificate. When a certificate with a SHA-256 signature from a V14 SP1 project is renewed in a V14 project, a certificate with an SHA-1 signature is always generated.
Valid from	The validity period of the certificate
Valid to	
Usage	Value that is entered in the "KeyUsage" and "ExtendedKeyUsage" extensions of the standard X.509 V3 for digital certificates.
Subject Alternative Name (SAN)	Depending on the certificate to be renewed, the alternative name of the certificate owner is displayed with an adaptable name type and value.

Replacing certificates

Meaning

In the "Replace certificate" dialog replace an existing certificate with a new one.

How to access this function

1. Right-click on the list entry of a certificate.
2. Select the "Replace" entry in the shortcut menu.
3. The "Replace certificate" dialog opens.

All certificates listed in the "Certificates involved" table are derived once again. This means that the CA group certificate of an already configured VPN group can be replaced in the project by the CA group certificate of a different project. The VPN group certificates for the VPN group members are therefore derived from the same CA group certificate in both projects.

After making changes in the certificate manager, download the configuration to all security modules affected.

Managing users and roles

Rules for user names, roles and passwords

Which rules apply to user names, role names and passwords?

When creating or modifying a user, a role or a password, remember the following rules:

Table 10-3 Rules for user management

Permitted characters	The following characters from the ANSI X 3.4-1986 character set are permitted: 0123456789 A...Z a...z !#$%&()*+,-./:;<=>?@ [\]_{|}~^
Characters not allowed	" ' `
Length of the user name (authentication method "password")	1 to 32 characters
Length of the user name (authentication method "RADIUS")	1 to 255 characters
Length of the password	8 to 32 characters
Length of the role name	1 to 32 characters
Maximum number of users per project	128

Maximum number of users on a security module (number of users that are assigned a role with module rights for the security module)	32 + 1 administrator when creating the project
Maximum number of roles per project	126
Maximum number of roles on a security module (number of roles that are assigned module rights for the security module and that are assigned to users)	37

Note

User names and passwords

As an important measure for increasing security, always make sure that user names and passwords are as long as possible and include special characters, upper and lowercase letters and numerals.

Using password policies, you can tighten the restrictions listed above for passwords even further. How to define password policies is described in the section:

Configuring password policies (Page 742)

Password strength

When a new password is entered, its password strength is checked. The following levels are distinguished for the password strength:

- Very weak
- Weak
- Medium
- Good
- Strong
- Very strong

Create users

Meaning

The security functions configured in STEP 7 are protected from unauthorized access by a separate user management. Before you can access the global and local security settings of security modules, you need to log in to the security configuration as a user.

Creating the first user in the project

After creating the first security module in the project, you first need to create a user. To do this, in the local security settings of the created security module, click the "User login" button under the "Security properties" entry and enter the login data of the user you want to create. You will then be logged in as the created user and the user is assigned the system-defined "Administrator" role. This role includes all configuration and module rights.

Upgrading a project to V14 SP1

If the STEP 7 project was created as a V14 project, the dialog for the user logon contains the check box "Upgrade project". If you select this check box, the project will be upgraded to V14 SP1 the next time a user logs on to the security configuration. V14 SP1 contains VPN settings with higher security, configurable VPN profiles and firewall settings for IP routing via the backplane bus. The user that upgrades the project to V14 SP1 by logging on to the security configuration must have the configuration right "Configure security". After upgrading, the project can no longer be opened with V14 and the check box can no longer be disabled. If a non-security relevant function is already being used in the project that is only available as of V14 SP1, the box "Upgrade project" is already enabled and cannot be disabled. The non-security relevant pert of the project was then already upgraded to V14 SP1. In this case, to be able to work on the security configuration the part of the project relevant for security must also be upgraded to V14 SP1. This requires a logon of a user with the configuration right "Configure security".

When adding the following modules available as of V14 SP1, the project must be upgraded to V14 SP1.

- CP 1543-1 V2.1
- CP 1243-1 BX30
- CP 1242-7 KX31
- CP 1243-7 LTE
- CP 1243-8 IRC
- TIM 1531 IRC
- TIM 3V-IE
- TIM 3V-IE Advanced
- TIM 3V-IE DNP3
- TIM 4R-IE
- TIM 4R-IE Standalone
- TIM 4R-IE DNP3
- TIM 4R-IE DNP3 Standalone

Creating users in the user management

When you are logged in as a user in the security configuration, you can create further users or delete users with the "User management" entry in the global security settings.

Note

Users with the "Administrator" role

There must always be at least one user with full configuration rights within a project. The administrator that is created automatically when you first enable the security functions in the project can only be deleted if at least one other user exists with the system-defined role "Administrator".

The following parameters are available in user management in the "User" tab:

Table 10-4 Information in the "User" tab

Parameter	Meaning
User name	Name of the user to be created. Click on the "Add new user" entry in the "User name" column to create a new user.
Password (only with the "Password" authentication method)	Entry of the password for the user. When it is entered, the password strength is checked. For more detailed information on password strength, refer to the following section: Rules for user names, roles and passwords (Page 732)
Authentication method	• **Password**: Use this authentication method for users that edit and download the STEP 7 Security project and that need to run diagnostics on the security module. The authentication of the user is performed by the security module when user-specific IP rule sets are activated. • **RADIUS** (only for SCALANCE S as of V4): The authentication of the user is performed by a RADIUS server when user-specific IP rule sets are activated. The password of the user is not configured in STEP 7 when using this authentication method but must be stored on the RADIUS server. Only use this authentication method for users that only need to log on to the Web page of a security module. A user with the "RADIUS" authentication method cannot log on to STEP 7 Security projects.
Role	Selecting a system-defined or user-defined role.
Maximum time of the session (only for SCALANCE S V3 or higher)	Entry of the time after which a user logged on to the Web page for user-specific IP rule sets of SCALANCE S modules is automatically logged off. The time entered here starts after the logon and after renewing the session on the Web page of the security module. • Default setting: 30 minutes • Minimum value: 5 minutes • Maximum value: 480 minutes
Comment	Entry of optional comments.

Creating roles

Overview

You can assign a system-defined or a user-defined role to a user. Specify the module rights of a user-defined role for each security module.

System-defined roles

The system-defined roles listed below are predefined. Certain rights are assigned to the roles that are the same on all modules and that the administrator can neither change nor delete.

- Administrator
  Default role when creating a security configuration.
  Unlimited access rights to all configuration data and all security modules.
- Standard
  Role with restricted access rights.
- Diagnose
  - Read access to configurations.
  - Read access to the security module in the "Online" mode for testing and diagnostics.
- Remote-Access
  No rights except for logging on to the Web page for user-specific firewall rule sets.
- administrator (radius)
  Role that can be used to activate user-specific IP rule sets with authentication using a RADIUS server.
  Access rights to all configuration data except SNMP MIBs.
- radius
  Role that can be used to activate user-specific IP rule sets with authentication using a RADIUS server.
  Read-only access.

You will find a detailed list of the configuration and module rights assigned to the system-defined roles "Administrator", "Standard" and "Diagnostics" in tables 1-3 to 1-7 of the section Managing rights (Page 737).

For more detailed information on user-specific IP rule sets, refer to the following section: AUTOHOTSPOT

For more detailed information on authentication using a RADIUS server, refer to the following section: AUTOHOTSPOT

User-defined role

In addition to the system-defined roles, you can create user-defined roles. For a user-defined role, select the configuration or module rights and specify the appropriate module rights for every security module used in the project. You manually assign the user-defined roles to the relevant user.

How to access this function

1. Double-click on the "User management" entry in the global security settings.
2. Select the "Roles" tab in User management.

Table 10-5 Information in the "Roles" tab

Parameter	Meaning
Role	Freely selectable role name. Double-click on the "Add new role" entry to create a new user-defined role. You can then set the rights for the created role.
Description	Specifying the system-defined role With user-defined roles, the "User-defined role" character string is displayed.
Maximum time of the session (only for SCALANCE S V3 or higher)	Entry of the time after which a user with the assigned role is automatically logged off from the Web page for user-specific IP rule sets of SCALANCE S modules. The time entered here starts after the logon and after renewing the session on the Web page of the security module. • Default setting: 30 minutes • Minimum value: 5 minutes • Maximum value: 480 minutes
Comment	Entry of additional, optional comments.

Note

Deleting roles

A user-defined role can only be deleted when it is no longer assigned to any user. If necessary, assign the user a different role.

System-defined roles cannot be deleted.

Managing rights

How to access this function

1. Double-click on the "User management" entry in the global security settings.
2. Select the "Roles" tab in User management.

Creating and assigning a user-defined role

1. Double-click on the "Add new role" entry.
2. Enter a name for the role and, if applicable, specify the maximum session time after which users with the assigned role are automatically logged off from the Web page for user-specific IP rule sets.

3. If necessary, select the system-defined role whose rights will be used as the template for the user-defined role from the drop-down list labeled "<Copy rights from>". User-defined roles cannot be selected from the drop-down list.
   Result: In the list of rights of the user roles, the rights are selected that are assigned to the selected system-defined role.
4. For each security module, enable or disable the rights to be assigned to the user-defined role.
5. Assign the role to a user in the "User" tab.

Configuration rights

Configuration rights are not module dependent and control the rights for configuration in STEP 7.

Depending on the user type, the following configuration rights are available for selection:

Table 10-6 Configuration rights

Configuration right	Administrator	Standard	Diagnose
Diagnose security	x	x	x
Configure security	x	x	-
Managing users and roles	x	-	-

Module rights

Module rights are configured per module. The "Service" column shows the service to which the particular right relates. With the "Copy rights" and "Paste rights" commands in the shortcut menu, you can transfer the rights from one module to another.

Depending on the user type, the following module rights are available for selection:

Table 10-7 Module rights CP 343-1 Advanced / CP 443-1 Advanced

Right within the service	Administrator	Standard	Diagnose	Service
Web: Format CP file system *	x	-	-	File system
FTP: Read files from the CP file system	x	x	x	
FTP: Write files to the CP file system	x	x	-	
FTP: Read files (DBs) from the S7 CPU **	x	x	x	PLC
FTP: Write files (DBs) to the S7 CPU ***	x	x	-	
Applet: Read tags using configured symbols *	x	x	x	
Applet: Write tags using configured symbols *	x	x	-	
Applet: Read tags using absolute addresses *	x	x	x	
Applet: Write tags using absolute addresses *	x	x	-	
Applet: Read status of the modules in the rack *	x	x	x	
Applet: Query order numbers of the modules in the rack *	x	x	x	

Right within the service	Administrator	Standard	Diagnose	Service
SNMP: Read MIB-II	x	x	x	SNMP
SNMP: Write MIB-II	x	x	-	
SNMP: Read automation MIB	x	x	x	
SNMP: Read LLDP-MIB	x	x	x	
SNMP: Read SNMPv2-MIB	x	x	x	
SNMP: Read MRP MIB	x	x	x	
SNMP: Write MRP MIB	x	x	-	
TIA Portal: Run diagnostics of the security module ****	x	x	x	Security
Web: Expand IP access control list *	x	-	-	Web
Web: Access Web diagnostics and CP file system	x	x	x	
Web: Send test e-mails *	x	x	x	
Web: Update firmware *	x	x	-	Maintenance
Web: Load diagnostics texts later *	x	x	-	

Table 10-8 Module rights CP 443-1 OPC UA

Right within the service	Administrator	Standard	Diagnose	Service
OPC UA: Read variables	x	x	x	OPC UA
OPC UA: Write variables	x	x	-	OPC UA
TIA Portal: Run diagnostics of the security module ****	x	x	x	Security
SNMP: Read automation MIB	x	x	x	SNMP
SNMP: Read LLDP MIB	x	x	x	SNMP
SNMP: Read MIB-II	x	x	x	SNMP
SNMP: Write MIB-II	x	x	-	SNMP
SNMP: Read SNMPv2-MIB	x	x	x	SNMP
Web: Update firmware	x	x	-	Maintenance
Web: Load diagnostics texts later	x	x	-	Maintenance
Web: Diagnostics	x	x	x	Web

Table 10-9 Module rights CP 1628

Right within the service	Administrator	Standard	Diagnose	Service
SNMP: Read MIB-II	x	x	x	SNMP
SNMP: Write MIB-II	x	x	-	
SNMP: Read automation MIB	x	x	x	
SNMP: Read SNMPv2-MIB	x	x	x	
TIA Portal: Run diagnostics of the security module ****	x	x	x	Security

Table 10-10 Module rights SCALANCE S

Right within the service	Administrator	Standard	Diagnose	Service
Download the configuration files	x	x	-	Security
TIA Portal: Run diagnostics of the security module ****	x	x	x	
SNMP: Read automation MIB	x	x	x	SNMP
SNMP: MIB-II read	x	x	x	
SNMP: MIB-II write	x	x	-	
SNMP: MRP-MIB read	x	x	x	
SNMP: MRP-MIB write	x	x	-	
SNMP: Read SNMPv2-MIB	x	x	x	
Web: Update firmware	x	x	-	Maintenance

Table 10-11 Module rights CP 1543-1

Right within the service	Administrator	Standard	Diagnose	Service
FTP: Read files from the CP file system	x	x	x	File system
FTP: Write files to the CP file system	x	x	-	
TIA Portal: Run diagnostics of the security module ****	x	x	x	Security
SNMP: Read automation MIB	x	x	x	SNMP
SNMP: Read IPv6 MIB	x	x	x	
SNMP: Read LLDP-MIB	x	x	x	
SNMP: Read MIB-II	x	x	x	
SNMP: Write MIB-II	x	x	-	
SNMP: Read SNMPv2-MIB	x	x	x	
FTP: Read files (DBs) from the S7 CPU **	x	x	x	PLC
FTP: Write files (DBs) to the S7 CPU ***	x	x	-	

Table 10-12 Module rights CP 1543SP-1

Right within the service	Administrator	Standard	Diagnose	Service
TIA Portal: Run diagnostics of the security module ****	x	x	x	Security
SNMP: Read automation MIB	x	x	x	SNMP
SNMP: Read IPv6 MIB	x	x	x	
SNMP: Read LLDP MIB	x	x	x	
SNMP: Read MIB-II	x	x	x	
SNMP: Write MIB-II	x	x	-	
SNMP: Read SNMPv2-MIB	x	x	x	

Table 10-13 Module rights CP 1243-1

Right within the service	Administrator	Standard	Diagnose	Service
TIA Portal: Run diagnostics of the security module ****	x	x	x	Security
SNMP: Read automation MIB	x	x	x	SNMP
SNMP: Read IPv6 MIB	x	x	x	
SNMP: Read MIB-II	x	x	x	
SNMP: Write MIB-II	x	x	-	
SNMP: Read SNMPv2-MIB	x	x	x	

Table 10-14 Module rights CP 1242-7

Right within the service	Administrator	Standard	Diagnose	Service
Use TeleService	x	-	-	TeleService

Table 10-15 Module rights CP 1242-7 LTE

Right within the service	Administrator	Standard	Diagnose	Service
TIA Portal: Run diagnostics of the security module ****	x	x	x	Security
Use TeleService	x	-	-	TeleService

Table 10-16 Module rights CP 1243-8 IRC

Right within the service	Administrator	Standard	Diagnose	Service
TIA Portal: Run diagnostics of the security module ****	x	x	x	Security
SNMP: Read automation MIB	x	x	x	SNMP
SNMP: Read IPv6 MIB	x	x	x	
SNMP: Read MIB-II	x	x	x	
SNMP: Write MIB-II	x	x	-	
SNMP: Read SNMPv2-MIB	x	x	x	

* To be able to use the function, the module right "Web: Access Web diagnostics and CP file system" must be enabled as well.

** To be able to use the function, the module right "FTP: Read files from CP file system" must be enabled as well.

*** To be able to use the function, the module right "FTP: Write files to CP file system" must be enabled as well.

**** To use the function, the configuration right "Security diagnostics" must also be enabled.

Setting module rights before and after creating the security modules

Within a user-defined role, the module rights for each security module are defined separately. If a security module was created before this role was added and module rights within a role need to be set, STEP 7 presets the module rights for the security module according to the selected rights template. The preset module rights can then be adapted when adding the role. If a security module was created after adding a role, no rights are preset for this security module. In this case, you will need to edit an existing role and set all the module rights yourself for the security module.

You can also transfer existing module rights to another module by copying and, if necessary, adapting them there. To do this, select a module in the shortcut menu in the module rights and select the "Copy rights" or "Paste rights" menu command.

Configuring password policies

Meaning

Using the password policies, specifications can be defined that need to be taken into account when assigning passwords to new users.

How to access this function

1. Double-click on the "User management" entry in the global security settings.
2. Select the "Password policies" tab in User management.

After selecting a check box, the corresponding policy is active and can, if necessary, be adapted using the relevant input box.

The minimum number of numbers and special characters together must not be longer than the minimum password length.

Parameter	Meaning
Minimum password length	Minimum number of characters that passwords are required to contain. The corresponding check box is enabled as default and cannot be disabled. • Minimum value: 8 characters • Maximum value: 32 characters
Minimum number of digits	Minimum number of digits that passwords are required to contain. • Minimum value: 1 digit • Maximum value: 32 digits
Minimum number of special characters	Minimum number of special characters that passwords are required to contain. A special character is any character that is neither a letter nor digit. • Minimum value: 1 special character • Maximum value: 32 special characters

Parameter	Meaning
Number of user passwords blocked for re-use	Number of the most recently used passwords that are not available for use as a new password if the password is changed. • Minimum value: 1 password • Maximum value: 10 passwords
At least one uppercase and lowercase character	If you select this check box, passwords must contain at least one uppercase and one lowercase letter.

Authentication using a RADIUS server

Module-specific function

This function is available only for SCALANCE S V4 modules or higher, refer to:

AUTOHOTSPOT in the section "SCALANCE S".

Generating configuration data for SCALANCE M modules

Meaning

You can generate the VPN information for the assignment of parameters to a SCALANCE M using STEP 7. For this to be possible, the module must be in at least one VPN group with a security module or a SOFTNET Security Client. With the generated files, you can then configure the SCALANCE M using the Web Based Management of the device.

Generated files

The following file types are generated:

- Export file with the configuration data
  - File type: *.txt file in ASCII format
  - Contains the exported configuration information for the SCALANCE M including information on the additionally generated certificates.
- VPN group certificates of the module
  - File type of the private key: *.p12 file
  - The file contains the module certificate and the key material.
  - Access is password protected.
- CA certificates of VPN groups
  - File type: *.cer file

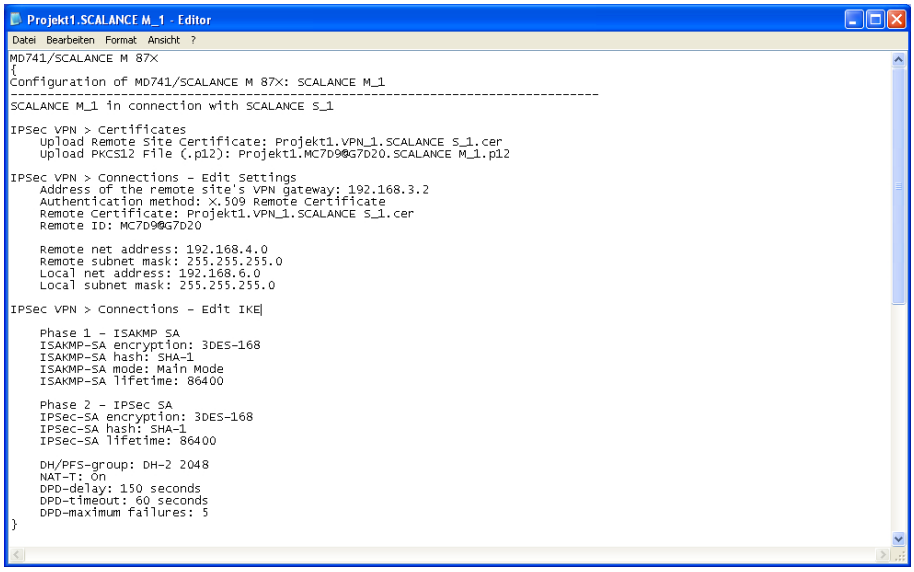


Figure 10-4 SCALANCE M configuration file

Note

No transfer to the SCALANCE M module

Configuration files are not transferred to the SCALANCE M module. An ASCII file is generated with which you can configure the VPN-relevant properties of the SCALANCE M. To allow this, the SCALANCE M must be located in at least one VPN group with another security module or a SOFTNET security client.

Note

Protecting exported configuration files from unauthorized access

Configuration files for SCALANCE M exported from STEP 7 can contain security related information. You should therefore make sure that these files are protected from unauthorized access. This is particularly important when passing on the files.

Follow the steps below

1. Select the module of the type "SCALANCE M".
2. In the local security settings, select the entry "SCALANCE M configuration".
3. Select the "Generate SCALANCE M files" check box and select a storage location for the configuration files.
4. Specify a password for the encryption of the VPN group certificate by using the project name as the password or assigning a password of your own.
5. Compile the configuration of the SCALANCE M module.

Result: The files (.txt file and certificates) are stored in the folder you specify.

Generating configuration data for security devices

Meaning

A security device serves as substitute for a device from another manufacturer.

A security device can have the function of a VPN device that serves as substitute for a VPN device from another manufacturer. You can generate the VPN information for the assignment of parameters to a VPN device using STEP 7. For this to be possible, the VPN device must be in at least one VPN group with a security module. With the generated files, you can then configure the VPN device.

Note

Protecting exported configuration files from unauthorized access

Configuration files for VPN devices exported from STEP 7 can contain security related information. You should therefore make sure that these files are protected from unauthorized access. This is particularly important when passing on the files.

Follow the steps below

1. Select the security device to be edited.
2. Select the "VPN device" entry in the local security settings.
3. Select the "Generate files of the VPN device" check box.
4. Select the mode of the VPN device from the "VPN mode" drop-down list. This selection is only possible when the VPN device is located in a VPN group.
   - VPN client: The VPN device can adopt the following role:
     Initiator The VPN device actively attempts to establish a VPN connection to the partner.
   - VPN server: The VPN device can adopt the following roles:
     Initiator/Responder: The VPN device actively attempts to establish a VPN connection to the partner. The receipt of requests for VPN connection establishment is also possible. This role is selected as default in the VPN connection properties.
     Responder: The receipt of requests for VPN connection establishment is possible.
   - NCP VPN client for Android (as of version 2.1): The VPN device with installed software NCP Secure VPN Client for Android actively attempts to establish a VPN connection to the partner.
5. Select a storage location for the configuration files.
6. Specify a password for the encryption of the VPN group certificate by using the project name as the password or assigning a password of your own.
7. If you have configured the VPN device as a VPN client:
   - Specify the corresponding file types in the "Exported certificates and keys" entry.

8. If you have configured the VPN device as a VPN server:
   - In the box "VPN server address (WAN IP address)" enter the WAN IP address at which the VPN server can be reached by the initiator of the VPN connection.
   - Specify the corresponding file types in the "Exported certificates and keys" entry.
   - In the entry "Subnets reachable through tunnel", enter the subnets and/or IP addresses for which tunnel communication will be released.
9. If you have configured the VPN device as an NCP secure VPN client:
   - In the entry "NCP Secure VPN client settings", enter the storage path on the NCP secure VPN client.

Note

Releasing subnets and subscribers with a VPN connection between NCP secure VPN client and SCALANCE S

The local internal subnet of SCALANCE S modules is released for tunnel communication as default. For this reason this is always entered as the first released subnet in the file generated for the NCP secure VPN client. Since NCP secure VPN client supports a maximum of 5 entries for released subnets / subscribers, as default the local internal network of the SCALANCE S module and the first 4 table entries from the entry "Nodes" in the NCP secure VPN client are released. To release other subnets and subscribers for tunnel communication, the file generated by STEP 7 must be suitably adapted.

Configuring interfaces for SCALANCE S modules

Overview

You will find information on configuration interfaces of SCALANCE S modules in AUTOHOTSPOT of the section "SCALANCE S". The information contained here deals with the following configuration options:

- Mode (bridge mode / routing mode / ghost mode (only for SCALANCE S602 as of V3.1)): Setting the operating mode (Page 841)
- IP address parameters: Configuring IP address parameters (Page 842)
- Port settings (only for SCALANCE S as of V3): Configuring port mode (Page 843)
- Settings for the Internet Service Provider (ISP) if one of the interfaces is operated using PPPoE (only for SCALANCE S as of V3): Configuring an Internet connection (Page 844)
- DNS (only for SCALANCE S as of V3): Configuring DNS (Page 846)
- Dynamic DNS (only for SCALANCE S as of V3): Configuring dynamic DNS (Page 846)
- LLDP (only for SCALANCE S as of V4 in routing mode): Configuring LLDP (Page 849)
- Media redundancy in ring topologies (e.g. MRP client), (only for SCALANCE S627-2M as of V4 in routing mode): AUTOHOTSPOT

For more information on special features of the Ghost mode, refer to the following section: Special features of the ghost mode (Page 852)

The configuration of the interfaces of CPs is described in the sections dealing with the CPs.

Setting up a firewall

Overview of the firewall

Meaning

The firewall functionality of the security modules is intended to protect networks and stations from third-party influence and interference. This means that only certain, previously specified communications relations are permitted. The firewall discards invalid frames without sending a response.

To filter the data traffic, IP addresses, IP subnets, services or MAC addresses can be used. You can also set a bandwidth limitation.

The firewall functionality can be configured for the following protocol levels:

- IP firewall with stateful packet inspection (layer 3 and 4)
- Firewall also for Ethernet "non-IP" frames according to IEEE 802.3 (layer 2)

With security modules capable of VPN, the firewall can also be used for the encrypted data traffic (IPsec tunnel). With the SCALANCE S602 Security module, the firewall can only be used for unencrypted data traffic.

Firewall rules

Firewall rules describe which packets in which direction are permitted or forbidden. IP rules affect all IP packets of layer 3 or higher. MAC rules only affect frames lower than layer 3.

Types of firewall rules

- Global firewall rule sets: Global firewall rule sets can be assigned to several security modules at the same time. Global firewall rule sets are configured in the global security settings.
- Local firewall rules: Local firewall rules are configured in the local security settings of a security module.
- User-specific IP rule sets (only for SCALANCE S as of V3): User-specific IP rule sets can be assigned to individual or several security modules at the same time. User-specific IP rule sets are configured in the global security settings and assigned there to one or more users.
  SCALANCE S as of V4 (RADIUS): User-specific IP rule sets can be assigned individual or multiple users as well as individual or multiple roles.

Service definitions

With the aid of service definitions, you can also define firewall rules clearly in a compact form. Service definitions are configured in the global security settings and can be used in the global, local and user-specific firewall rules.

Adapting standard rules for IP services

For SCALANCE S modules V3 or higher, you can adapt service-specific firewall rules that are set as default for the interfaces of the security modules. You will find information on configuring these firewall rules in Adapting standard rules for IP services (Page 868) of the section "SCALANCE S".

Automatically generated firewall rules for CP connections

For connections that were configured using CPs, STEP 7 automatically creates firewall rules that allow communication with the partner of the CP in the specified direction (CP active/ passive). The connection establishment directions are taken into account. To display these firewall rules, if the advanced firewall mode is enabled, the "Update connection rules" button needs to be clicked. The firewall rules are then displayed in advanced firewall mode.

Which firewall rules are generated automatically is described in the following sections.

- For S7-300 CPs/S7-400 CPs/PC CPs: Connection-related automatic firewall rules (Page 900) in the section "Security for S7-300 / S7-400 / PC CPs".
- For S7-1200-/S7-1500-CPs: Connection-related automatic firewall rules (Page 913) in the section "Security for S7-1200 / S7-1500 CPs".

Enabling the firewall

Firewall functionality for a specific security module is controlled in the local security settings using the "Enable firewall" check box. If the check box is enabled, the firewall can be configured and is effective following the download. For a security module that is a member of a VPN group, the "Enable firewall" check box is activated by default and cannot be deactivated. After switching to advanced firewall mode, it is not possible to switch back to standard mode. For more information about the standard and advanced firewall modes, refer to the section:

Overview of local firewall rules (Page 770).

Global firewall rule sets

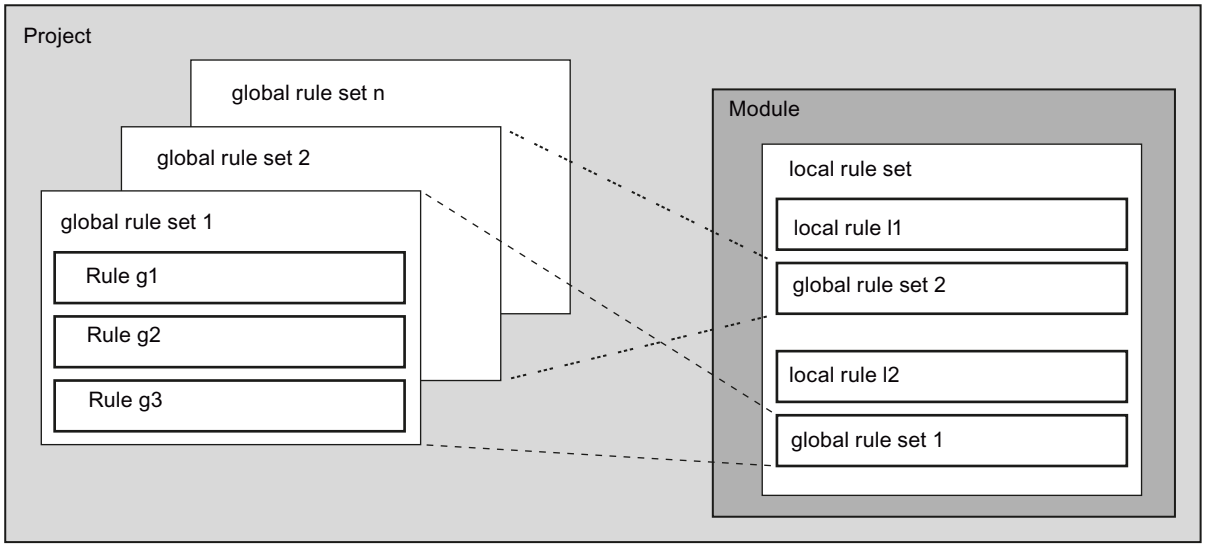
Application

Regardless of the module, global firewall rule sets are configured in the global security settings. A firewall rule set consists of one or more firewall rules and is assigned to the individual security modules.

In the global firewall rule sets, a distinction is made between the following:

- IP rule sets
- MAC rule sets

The following schematic illustrates the relationship between globally defined rule sets and locally used rule sets.



Configuring

When configuring global firewall rules, you can make detailed firewall settings. You can allow individual services for a single node or all services for the node for access to the station or network.

When are global IP and MAC firewall rules useful?

Global firewall rules are useful if you want to define identical filter criteria for communication with several security modules.

Note

Assigning firewall rule sets with incompatible firewall rules

For a security module, only the firewall rules from firewall rule sets are adopted correctly if the security module supports them. A firewall rule contained in a global firewall rule set with the direction "From: External" or "To: Any" is, for example not assigned to a CP 1628. The other firewall rules of the global firewall rule set are adopted if the CP 1628 supports them.

Global firewall rule sets - conventions

Global firewall rule sets are used locally

The following conventions apply when creating a global set of firewall rules and when assigning it to a module:

- Configuration view
  Global firewall rule sets are configured in the global security settings.
- Priority
  As default, locally defined rules have a higher priority than the global IP and MAC firewall rule sets. Newly assigned global IP and MAC firewall rule sets are therefore initially entered at the bottom of the local rule list.
  The priority can be changed by changing the position in the rule list.
- Entering, changing or deleting rule sets
  Global firewall rule sets cannot be edited in the local rule list of the firewall rules in the module properties. They can only be displayed there and positioned according to the required priority.
  It is not possible to delete a single rule from an assigned rule set in the local security settings. Only the entire rule set can be removed from the local list of rules. The firewall rule sets in the global security settings remain unaffected.

Creating global firewall rule sets

How to access this function

1. In the global security settings, select the entry "Firewall" > "Global firewall rule sets" > "IP rule sets" or "MAC rule sets".
   Result: The previously created IP rule sets or MAC rule sets are displayed under the selected entry.
2. Double-click on the entry "Add new IP rule set" or "Add new MAC rule set".
   Result: The created firewall rule set is displayed with an automatically assigned number below the entry "IP rule sets" or "MAC rule sets".
3. Double-click on the created firewall rule set.
   Result: In the working area the assignment dialog for firewall rule sets is displayed. The selected firewall rule set is selected in the "Rule set" drop-down list. The required security modules can be assigned to this, see section Assigning global firewall rule sets (Page 751). In the local security settings, the configurable properties of the firewall rule set are displayed.

4. Enter the following data in the "General" entry in the local security settings:
   - Name: Project-wide, unique name of the rule set. The name appears in the local rule list of the security module after the rule set is assigned.
   - Description (optional): Enter a description of the global rule set.
5. In the local security settings in the entry "Firewall rule set" enter the firewall rules in the list.
   Note the parameter description in the following sections:
   For IP rule sets: Defining IP packet filter rules (Page 771)
   For MAC rule sets: Defining MAC packet filter rules (Page 775)
   To switch over to the local security settings of another firewall rule set, select this from the "Rule set" drop-down list of the assignment dialog or double-click on the corresponding entry in the global security settings.

Result

You have created the global firewall rule set and can now assign this to the required security modules.
Note the descriptions in the following section:

Assigning global firewall rule sets (Page 751)

Assigning global firewall rule sets

Requirement

You have enabled the advanced firewall mode for the security modules you want to assign to a firewall rule set.

Procedure

1. In the global security settings, select the entry "Firewall" > "Global firewall rule sets" > "Assign module to a firewall rule set".
2. From the "Rule set" drop-down list, select the rule set to which you want to assign the security module.
   In the right-hand table, you will see the security modules that you can assign to the selected firewall rule set. In the left-hand table, you will see the security modules already assigned to the selected firewall rule set.
3. In the "Available modules" area, select the security modules you want to assign to the selected rule set.
4. Click the "<<" button to assign the selected modules to the selected rule set.

Result

The global rule set is used by the assigned security modules as a local rule set and appears automatically at the end of the list of firewall rules in the local security settings.

Exporting and importing global firewall rule sets

Meaning and function

Global IP rule sets and global MAC rule sets can be exported from STEP 7 in the XLSX format and imported into STEP 7. Per firewall rule set an XLSX file is created when you export.

The export and import provides the option of exchanging firewall rule sets between different STEP 7 projects. Exported firewall rule sets can also be edited in Microsoft Excel® and then imported into STEP 7 again. This simplifies the making of mass changes.

Firewall rule sets exported from STEP 7 are largely compatible with the Security Configuration Tool (SCT). Firewall rule sets exported from from the Security Configuration Tool are fully compatible with STEP 7.

Note

Restrictions when importing firewall rule sets from STEP 7 into the Security Configuration Tool

In SCT, IPv6 firewall rules and ICMPv6 services are not supported. During import firewall rules identified as IPv6 firewall rules and ICMPv6 services are ignored.

As the bandwidth limit a maximum of 100 Mbps are permitted in SCT. During import firewall rules with more than 100 Mbps are ignored.

Exporting firewall rule sets from STEP 7

1. In the global security settings, select the entry "Firewall" > "Global firewall rule sets".
2. Double-click on the "Overview" entry.
   Result: The IP rule sets and MAC rule sets that exist in STEP 7 are displayed in the working area.
3. If you do not want to export all firewall rule sets, enable the check box of the firewall rule sets to be exported.
4. Click the button.
5. In the "Export firewall rule sets" dialog, select whether all firewall rule sets, only the selected firewall rule sets or all firewall rule sets supported by SCT will be exported. With the third option firewall rule sets containing firewall rules containing communication directions not supported by SCT are excluded from the export.
6. Enter a path on which the firewall rule sets will be saved and click the "Export" button.
   Result: Per firewall rule set an XLSX file was created. The XLSX files were named with the name of the corresponding firewall rule set and according to the date and time of the export.

Importing firewall rule sets in STEP 7

Exported firewall rule sets can be edited prior to importing them. Refer to the information in the section "Requirements for firewall rule sets to be imported".

Follow the steps outlined below to import firewall rule sets in STEP 7:

1. In the global security settings, select the entry "Firewall" > "Global firewall rule sets".
2. Double-click on the "Overview" entry.
   Result: The IP rule sets and MAC rule sets that exist in STEP 7 are displayed in the working area.
3. Click the button.
4. If firewall rule sets of the same name are to be overwritten, select the "Overwrite existing firewall rule sets" check box.
   Note: For the comparison between the firewall rule sets existing in STEP 7 and the XLSX files, only the name of the firewall rule sets is used. The information about the date and time of the XLSX files is ignored.
5. Select the XLSX files to be imported. Multiple selection is possible.
6. Click the "Import" button.
   Result: The firewall rule sets of the selected XLSX files have been imported in STEP 7.

Structure of exported IP rule sets

For every IP rule set exported from STEP 7, an XLSX file is created with the following table sheets:

- IP Ruleset: Contains the IP packet filter rules of the IP rule set
- IP Services: Contains the IP services used in packet filter rules of the IP rule set. IP services in the IP service groups used are also displayed.
- ICMP Services: Contains the ICMP services used in packet filter rules of the IP rule set. ICMP services in the IP service groups used are also displayed.
- IP Service Groups: Contains the IP service groups used in IP packet filter rules of the IP rule set. The corresponding IP and ICMP services are displayed for every IP service group.

Structure of exported MAC rule sets

For every MAC rule set exported from STEP 7, an XLSX file is created with the following table sheets:

- MAC Ruleset: Contains the MAC packet filter rules of the MAC rule set
- MAC Services: Contains the MAC services used in MAC packet filter rules of the MAC rule set. MAC services in the MAC service groups used are also displayed.
- MAC Service Groups: Contains the MAC service groups used in MAC packet filter rules of the MAC rule set. The corresponding MAC services are displayed for every MAC service group.

Conventions for rule sets to be imported

The following conventions apply to firewall rule sets to be imported in STEP 7:

- The files have the file extension *.XLSX and the corresponding Office Open XML format.
- If the name of the XLSX file is longer than 128 characters, the rule set will be created in STEP 7 with a standard name.
- The names of the columns within the table sheets must not be changed, the order of the columns may, however, be changed.
- The names of the table sheets must not be changed, the order of the table sheets may, however, be changed.

Permitted parameter values for IP rule sets to be imported

The following values may be entered in the table columns of XLSX files for IP rule sets. Upper and lower case characters are not taken into account when the notation is checked.

Table 10-17 Permitted parameter values for table sheet "IP Ruleset"

Action	From	To	IPv6*	Source IP address	Destination IP Address	Service	Bandwidth	Logging	Stateful	Comment
• Allow • Drop	External	• Internal • DMZ • Tunnel • Any • Station • Backplane **	• True • False	[free text with max. 255 characters]	[free text with max. 255 characters]	[free text with max. 128 characters]	[Possible values: 0.001… 1000]	• True • False	• True • False	[free text with max. 255 characters]
	Internal	• External • Internal • DMZ • Tunnel • Any • Station								
	DMZ	• External • Internal • Tunnel • Any								
	Tunnel	• External • Internal • DMZ • Tunnel • Any • Station • Backplane **								

Action	From	To	IPv6*	Source IP address	Destination IP Address	Service	Bandwidth	Logging	Stateful	Comment
	Any	• External • Internal • DMZ • Backplane**								
	Station	• External • Internal • Tunnel • Backplane**								
	Backplane**	• Any • External • Station • Tunnel								

* This column is optional. If the column does not exist when importing into STEP 7, the default value "False" is used in STEP 7.

** Only possible with a V14, SP1 project

Table 10-18 Permitted parameter values for table sheet "IP Services"

Name	Protocol	Source Port	Destination Port
[free text with max. 128 characters]	• UDP • TCP • All	[free text with max. 32 characters]	[free text with max. 32 characters]

Table 10-19 Permitted parameter values for table sheet "ICMP Services"

Name	ICMPv6*	Type	Code
[free text with max. 128 characters]	True	• PacketToBig • EchoRequest • EchoReply • MulticastListenerQuery • MulticastListenerReport • V2MulticastListenerReport • MulticastListenerDone • RouterSolicitation • RouterAdvertisement • NeighbourSolicitation • NeighbourAdvertisement • RedirectMessage • InverseDiscoverySolicitation • InverseDiscoveryAdvertisement • HomeDiscoveryRequest • HomeDiscoveryReply • MobilePrefixSolicitation • MobilePrefixAdvertisement • CertificationPathSolicitation • CertificatePathAdvertisement • IcmpMessagesUtilizedBySeamoby • MulticastRouterAdvertisement • MulticastRouterSolicitation • MulticastRouterTermination • FmIpV6 • RplControl	NoCode
		RouterRenumbering	• RouterRenumberingCommand • RouterRenumberingResult • SequenceNumberReset
		IcmpNodeInformationQuery	• ContainsIpV6

Name	ICMPv6*	Type	Code
			• ContainsName • ContainsIpV4
		IcmpInformationResponse	• Successful • RefuseToSupplyAnswer • UnknownQtypeOfQuery
		TimeExceeded	• HopLimitExceededInTransit • FragmentReassemblyTimeExceeded
		ParameterProblem	• ErroneousHeaderFieldEncountered • UnrecognizedNextHeaderTypeEncountered • UnrecognizedIpV6OptionEncountered
	False	• EchoReply • SourceQuench • AlternateHostAddress • EchoRequest • RouterSolicitation • TimestampReply • TimestampRequest • InformationRequest • InformationReply • AddressMaskReply • AddressMaskRequest • MobileHostRedirect • IpV6WhereAreYou • IpV6IAmHere • MobileRegistrationReply • MobileRegistrationRequest • Skip • Star	NoCode
		DestinationUnreachable	• Star • NetUnreachable • HostUnreachable • ProtocolUnreachable • PortUnreachable • FragmentationNeededAndDontFragmentWasSet • SourceRouteFailed

Name	ICMPv6*	Type	Code
			• DestinationNetworkUnknown • DestinationHostUnknown • SourceHostIsolated • CommunicationWithDestinationNetworkIsAdministrativelyProhibited • CommunicationWithDestinationHostIsAdministrativelyProhibited • DestinationNetworkUnreachableForTypeOfService • DestinationHostUnreachableForTypeOfService • CommunicationAdministrativelyProhibited • HostPrecedenceViolation • PrecedenceCutOffInEffect
		Redirect	• Star • DatagramForNetwork • DatagramForTheHost • DatagramForTheTypeOfServiceAndNetwork • DatagramForTheTypeOfServiceandHost
		RouterAdvertisement	• Star • NormalRouterAdvertisement • DoesNotRouteCommonTraffic
		TimeExceeded	• Star • TimeToLiveExceededInTransit • FragmentReassemblyTimeExceeded
		ParameterProblem	• Star • TheIpHeaderIsInvalid • ArequiredOptionIsMissing
		Traceroute	• Star • OutboundPacketSuccessfullyForwarded

Name	ICMPv6*	Type	Code
			• NoRouteForOutboundPacketThePacketWasDiscarded
		ConversionError	• Star • UnknownOrUnspecifiedError • DontConvertOptionPresent • UnknownMandatoryOptionPresent • KnownUnsupportedOptionPresent • UnsupportedTransportProtocol • OverallLengthExceeded • IpHeaderLengthExceeded • TransportProtocolGreaterThen255 • PortConversionOutOfRange • TransportHeaderLengthExceeded • BitRolloverMissingAndAckSet • UnknownMandatoryTransportOptionPresent
		Photuris	• Star • BadSpi • AuthenticationFailed • DecompressionFailed • DecryptionFailed • NeedAuthentication • NeedAuthorization

* This column is optional. If the column does not exist when importing into STEP 7, the default value "False" is used in STEP 7.

Table 10-20 Permitted parameter values for table sheet "IP Service Groups"

Name	Description	IP Services
[free text with max. 128 characters]	[free text with max. 255 characters]	[free text, services are separated from each other with commas]

Permitted parameter values for MAC rule sets to be imported

The following values may be entered in the table columns of XLSX files for MAC rule sets. Upper and lower case characters are not taken into account when the notation is checked.

Table 10-21 Permitted parameter values for table sheet "MAC Ruleset"

Action	From	To	Source MAC Address	Destination MAC Address	Service	Bandwidth	Logging	Comment
• Allow • Drop	External	• Internal • Tunnel • Any • Station	[MAC address in the correct format: xx-xx-xx-xx-xx-xx]	[MAC address in the correct format: xx-xx-xx-xx-xx-xx]	[free text with max. 128 characters]	[Possible values: 0.001… 1000]	• True • False	[free text with max. 255 characters]
	Internal	• External • Tunnel • Any						
	Tunnel	• External • Internal • Station						
	Any	• External • Internal						
	Station	• External • Tunnel						

Table 10-22 Permitted parameter values for table sheet "MAC Services"

Name	Protocol	DSAP	SSAP	CTRL	OUI	OUI Type
[free text with max. 128 characters]	• SNAP • PROFINET IO • ISO • [hexadecimal values beginning with 0x]	[hexadecimal values with max. 2 characters]	[hexadecimal values with max. 2 characters]	[hexadecimal values with max. 2 characters]	[hexadecimal values with max. 6 characters]	[hexadecimal values with max. 4 characters]

Table 10-23 Permitted parameter values for table sheet "MAC Service Groups"

Name	Description	MAC Services
[free text with max. 128 characters]	[free text with max. 255 characters]	[free text, services are separated from each other with commas]

IP services

Defining IP services

How to access this function

In the global security settings, select the entry "Firewall" > "Services" > "Define services for IP rules".

Procedure

Using the definition of IP services, you can define succinct and clear firewall rules for specific services. You select a name and assign the service parameters to it.

These services defined in this way can also be grouped together under a group name.

When you configure the packet filter rules, you then use this name.

Parameters for IP services

You define the IP services using the following parameters:

Table 10-24 IP services: Parameter

Parameter	Meaning/comment	Available options / ranges of values
Name	Name for the service that is used as identification in the rule definition or in the group. The names of pre-defined services cannot be changed.	• Name must start with a letter. • Name must not contain any special characters. • Name must not be redundant.
Protocol	Selection of the protocol type	• TCP • UDP • TCP+UDP • All

Parameter	Meaning/comment	Available options / ranges of values
Source port	Filtering is based on the port number specified here; this defines the service access at the frame sender.	With the protocol selection "TCP+UDP", it is not possible to specify a port. Examples: *: Port is not checked 20 or 21: FTP service
Destination port	Filtering is based on the port number specified here; this defines the service access at the frame recipient.	With the protocol selection "TCP+UDP", it is not possible to specify a port. Examples: *: Port is not checked TCP 80: Web HTTP service TCP 102: S7 protocol

Defining ICMP services

How to access this function

1. In the global security settings, select the entry "Firewall" > "Services" > "Define services for IP rules".
2. Select the "ICMP" tab.

Procedure

Using the definition of ICMP services, you can define succinct and clear firewall rules for specific services. You select a name and assign the service parameters to it.

These services defined in this way can be grouped together under a group name.

When you configure the packet filter rules, you then use this name.

Parameters for ICMP services

Parameter	Meaning/comment	Available options / ranges of values
Name	User-definable name for the service that is used as identification in the rule definition or in the group. The names of predefined ICMPv6 services cannot be modified.	• Name must start with a letter. • Name must not contain any special characters. • Name must not contain more than 20 characters. • Name must not occur more than once
ICMPv6	If you enable this check box, the ICMP service is declared as an ICMPv6 service and you can select an ICMPv6-specific type and code for the service. An ICMPv6 service can only be used in the firewall rule of a security module that supports IPv6.	• Enabled • Disabled (default)
Type	Type of the ICMPv4 or ICMPv6 message.	If the "ICMPv6" check box is deselected, ICMPv4-specific types can be selected. If the check box is selected, ICMPv6-specific types can be selected.
Code	Code of the ICMP type.	Values depend on the selected type.

Creating service groups

How to access this function

1. In the global security settings, select the entry "Firewall" > "Services" > "Define services for IP rules".
2. Select the "Service groups" tab.

Forming service groups

You can put several services together by creating service groups. This allows you set up more complex services that you can then use in the packet filter rules simply by selecting the name. IPv4 and IPv6 services can be collected in the same service group.

Create groups in the open "Service groups" tab. You can then assign services to a group in the "Group management" tab.

Follow the steps below

1. First, create groups in this tab with names to suit your requirements and add a description if required.
2. Select the "Group management" tab. You can assign the previously specified IP services to the groups defined in this tab.

Managing service groups

How to access this function

1. In the global security settings, select the entry "Firewall" > "Services" > "Define services for IP rules".
2. Select the "Group management" tab.

Forming service groups

You can put several services together by creating service groups. This allows you set up more complex services that you can then use in the packet filter rules simply by selecting the name. IPv4 and IPv6 services can be collected in the same service group.

Use the "Group management" tab to assign services to a selected group you created previously in the "Service groups" tab.

Follow the steps below

1. Select a group created previously in the "Service groups" tab from the "Service groups" drop-down list in this tab.
2. Then assign the required services from the right-hand "Available services" list box to the group.

MAC services

Define MAC services

How to access this function

In the global security settings, select the entry "Firewall" > "Services" > "Define services for MAC rules".

Meaning

Using the definition of MAC services, you can define succinct and clear firewall rules for specific services. You select a name and assign the service parameters to it.

These services defined in this way can be grouped together under a group name.

When you configure the global or local packet filter rules, you use this name.

Parameters for MAC services

A MAC service definition is formed using protocol-specific MAC parameters:

Table 10-25 MAC services - parameters

Parameter	Meaning/comment	Available options / ranges of values
Name	User-definable name for the service that is used as identification in the rule definition or in the group.	• Name must start with a letter. • Name must not contain any special characters. • Name must not be redundant.
Protocol	Name of the protocol type: • ISO ISO identifies frames with the following properties: Lengthfield <= 05DC (hex), DSAP= userdefined SSAP= userdefined CTRL= userdefined • SNAP SNAP identifies frames with the following properties: Lengthfield <= 05DC (hex), DSAP=AA (hex), SSAP=AA (hex), CTRL=03 (hex), OUI=userdefined, OUI-Type=userdefined • PROFINET IO • As an alternative, a protocol number can be entered. The protocol entries 0800 (hex) and 0806 (hex) are not accepted since these values apply to IP or ARP frames.	• ISO • SNAP • PROFINET IO • 0x (entry of the protocol number)
DSAP	Destination Service Access Point: LLC recipient address	
SSAP	Source Service Access Point: LLC sender address	
CTRL	LLC control field	
OUI	Organizationally Unique Identifier (the first 3 bytes of the MAC address = vendor ID)	
OUI type	Protocol type/identification	

Note

Processing for S7-CPs

Only settings for ISO frames with DSAP=SSAP=FE (hex) are processed. Other frame types are not relevant for S7 CPs and are therefore discarded even before processing by the firewall.

Special settings for SIMATIC NET services

To filter special SIMATIC NET services, please use the following protocol settings:

- DCP:
  PROFINET IO
- SiClock :
  OUI= 08 00 06 (hex) , OUI-Type= 01 00 (hex)

Creating service groups

How to access this function

1. In the global security settings, select the entry "Firewall" > "Services" > "Define services for MAC rules".
2. Select the "Service groups" tab.

Forming service groups

You can put several services together by creating service groups. This allows you set up more complex services that you can then use in the packet filter rules simply by selecting the name.

Create groups in the open "Service groups" tab. You can then assign services to a group in the "Group management" tab.

Follow the steps below

1. First, create groups in this tab with names to suit your requirements and add a description if required.
2. Select the "Group management" tab. You can assign the previously specified MAC services to the groups defined in this tab.

Managing service groups

How to access this function

1. In the global security settings, select the entry "Firewall" > "Services" > "Define services for MAC rules".
2. Select the "Group management" tab.

Forming service groups

You can put several services together by creating service groups. This allows you set up more complex services that you can then use in the packet filter rules simply by selecting the name.

Use the "Group management" tab to assign services to a selected group you created previously in the "Service groups" tab.

Follow the steps below

1. Select a group created previously in the "Service groups" tab from the "Service groups" drop-down list in this tab.
2. Then assign the required services from the right-hand "Available services" list box to the group.

Overview of local firewall rules

Meaning

Local firewall rules are configured in the local security settings of a security module and apply only to this security module. After enabling the firewall functionality you can either use predefined firewall rules or define firewall rules in the advanced firewall mode.

Using predefined firewall rules

Here, you use simple, predefined rules. You can only enable service-specific rules. The enabled services are allowed for all nodes in the specified direction. You can find detailed information on the definition of firewall rules in this dialog in the following module-specific sections:

- For SCALANCE S: AUTOHOTSPOT
- For S7-300/S7-400/PC CPs: AUTOHOTSPOT
- For S7-1200/S7-1500 CPs: AUTOHOTSPOT

The communication direction "Station" that can be configured for CPs, includes access to the CP and access to the CPU via the CP.

Defining firewall rules in advanced firewall mode

In advanced firewall mode, you can make detailed firewall settings. You can allow individual services for a single node or all services for the node for access to the station or network. You enable the advanced firewall mode using the "Activate firewall in advanced mode" check box. In the local security settings, the firewall rules can then be configured in "Firewall" > "IP rules" or "MAC rules". The configuration options available here are described individually in the following section:

For IP packet filter rules: Defining IP packet filter rules (Page 771)

For MAC packet filter rules: Defining MAC packet filter rules (Page 775)

Note

Deactivating the advanced firewall mode not possible

Once you have activated the advanced firewall mode, you can no longer deactivate it.

Adopting predefined firewall rules in the advanced firewall mode

When you switch over to the advanced firewall mode, enabled predefined firewall rules and logging settings are adopted. If, for example, the check box “Allow S7 protocol” is selected in the direction “From external to internal”, in the advanced firewall mode an allow rule with this communication direction and the service “S7” is created automatically.

If predefined logging settings exist for S7-1200 / S7-1500 CPs, in advanced mode these are not shown as firewall rules but under the entry “Firewall > Log settings” where they can be edited.

Configuration limits

Number of firewall rules (advanced firewall mode)	
SCALANCE S as of V3	Maximum of 256
S7-1200/S7-1500 CPs	Maximum of 256
CP 343-1 Adv. / 443-1 Adv.	Maximum of 226
CP 1628	Maximum of 226

Defining IP packet filter rules

Meaning

With IP packet filter rules, you filter according to IP frames, such as TCP, UDP and ICMP.

Within a packet filter rule, you can also fall back on the definitions of the IP services.

Entering IP packet filter rules

Enter the firewall rules one by one in the list. Note the following parameter description.

Table 10-26 IP rules: Parameter

Parameter	Meaning/comment	Available options / ranges of values
Action	Allow/disallow (enable/block)	• Allow Allow frames according to definition. • Drop Block frames according to definition. For firewall rules generated automatically as result of configuring a connection and subsequently adapted manually: • Allow* • Drop* If you change automatically created connection rules, when the "*" option is selected they are not recreated and overwritten by STEP 7.
From / To	Selection of the communications directions for which the rule will apply.	Described in separate sections. • For SCALANCE S modules: IP packet filter directions SCALANCE S (Page 867) • For S7-300/S7-400/PC CPs: IP packet filter directions S7-300-/S7-400-/PC-CPs (Page 898) • For S7-1200/S7-1500 CPs: IP packet filter directions S7-1200 / S7-1500 CPs (Page 911) The communication direction "Station" includes access to the CP and access to the CPU via the CP. The communication direction "Any" includes all the firewall directions supported by the security module at the time of loading. The communication directions "Station", "Backplane bus" (only for CP 1543-1 as of V2.1 with the option "IP routing between communications modules" is classified as trustworthy as default. For communication between trustworthy directions no firewall rules need to be configured.
IPv6	If this check box is selected, you can use a previously defined ICMPv6 service in the firewall rule. For some security CPs, after selecting the check box you can enter additional IPv6 addresses in the "Source IP address" and "Destination IP address" input boxes.	The check box can only be selected and deselected if there are no entries in the "Source IP address" and "Destination IP address" input boxes. If IPv6 is disabled in the local settings of a CP capable of IPv6, you cannot select the "IPv6" check box in the local security settings of the CP and therefore cannot use ICMPv6 services or IPv6 addresses in firewall rules. Existing firewall rules that use IPv6 are shown grayed out if IPv6 is disabled.

Parameter	Meaning/comment	Available options / ranges of values
Source IP address	The firewall rule is applied to the frames whose sender has the IP address specified here. If you do not specify an IP address, the firewall rule applies to all nodes in the communications direction you selected in the "From" column.	You can find additional information about IP addresses in the IP addresses in IP packet filter rules (Page 777) section. **Configuration options in ghost mode (only for SCALANCE S S602 as of V3.1):** If ghost mode is activated, the IP address of the internal node is dynamically determined by the security module at runtime. Depending on the selected direction, you can select one of the following options in the column "Source IP address" (for direction "From internal to external) or in the column "Destination IP address" (for direction "From external to internal"): • IP address of the internal node: The IP address of the internal node is inserted in the firewall rule by the SCALANCE S. • Limited broadcast: The broadcast IP address 255.255.255.255 is inserted in the firewall rule by the SCALANCE S. • Directed broadcast: The broadcast IP address of the SCALANCE S network is inserted in the firewall rule by the SCALANCE S. A directed broadcast can also be forwarded into the destination network via routers. • Multicast: The multicast address band 224.0.0.0 /24 is inserted in the firewall rule by the SCALANCE S. After selecting this option, as an alternative a specific multicast IP address from the multicast address band can be specified.
Destination IP address	The firewall rule is applied to the frames whose recipient has the IP address specified here. If you do not specify an IP address, the firewall rules applies to all nodes in the communications direction you selected in the "To" column.	
Service	Selection of the IP/ICMP service or the service group to be used. Before you select an ICMPv6 service, you need to select the "IPv6" check box. Note: Before using a predefined service make sure that its predefined parameters meet your requirements. Pay particular attention to the predefined source and destination ports.	The drop-down list box displays the predefined and configured IP services and service groups you can select. These are displayed in the global security settings under "Firewall" > "Services" > "Define services for IP rules".
Bandwidth (Mbps)	Option for setting a bandwidth limitation Can only be entered if the "Allow" action is selected. A packet passes through the firewall if the allow rule matches and the permitted bandwidth for this rule has not yet been exceeded.	CP 343-1 Adv. / CP 443-1 Adv., S7-1200-/S7-1500-CPs: 0.001 ... 100 Mbps CP 1628 and SCALANCE S: 0.001 ... 1000 Mbps For global and user-specific rules: 0.001 ... 1000 Mbps For IP rules with the direction "Backplane bus": 0.001 ... 1 Mbps Note: If the direction "From tunnel to station" is configured in a firewall rule for the S7-1200/S7-1500 CPs, no bandwidth limitation can be specified. For CP 1543-1 V2.1 only one bandwidth limitation can be specified if "Any" is configured for one of the two directions. This does not apply to the combination of directions "Any" and "Station".
Logging	Enable and disable logging for this rule. If logging is enabled, the settings for the packet filter logging configured in the local security settings apply.	• Enabled • Disabled (default)

Parameter	Meaning/comment	Available options / ranges of values
Number	Automatically assigned number for the rule. The numbers are recalculated when rules are moved.	
Stateful	If this check box for an IP rule is disabled with the "Allow" action, no firewall states are generated by packets to which the allow rule applies. Firewall states automatically allow the responses to allowed packets. Can only be adapted if the "Allow" action is selected. Configuration of IP rules without firewall states is only possible for SCALANCE S modules as of firmware V3. If the responses to packets that have passed the firewall according to such IP rules should also be allowed, additional IP rules need to be configured for these responses.	For allow rules of SCALANCE S ≥ V3: • enabled (default) • disabled For allow rules of SCALANCE S < V3 and CPs: • enabled (default) Note: The check box is not present for CPs; however, the activated option is always used implicitly for allow rules of CPs. For drop rules: • Disabled (default)
Comment	Space for your own explanation of the rule	If a comment is marked with "AUTO", it was created for an automatic connection rule. For rules you have created, entry of a comment is optional.

Table 10-27 Meaning of the entries in the shortcut menu

Entry in the shortcut menu	Meaning
Delete	This deletes the selected rule or the selected rule set. Notes on removing a globally defined and locally assigned rule set: if you delete a rule set here, this only cancels the assignment to the security module.
Save as global rule set (only for local firewall rules)	Copies the selected firewall rule(s) and inserts it as a global firewall rule set in the global security settings. The firewall configuration currently configured for the security module remains unaffected by this procedure.
Move up	Use this button to move the selected rule or selected rule set up one position in the list. As an alternative, you can move the selected rule or the selected rule set by dragging it with the mouse. You can select multiple entries. The rule / rule set you moved is therefore handled with higher priority.
Move down	Use this button to move the selected rule or selected rule set down one position in the list. As an alternative, you can move the selected rule or the selected rule set by dragging it with the mouse. You can select multiple entries. The rule / rule set you moved is therefore handled with lower priority.
Define service for IP rules	This opens the dialog in which you can manage the IP services and service groups.

Defining MAC packet filter rules

Meaning

With the MAC packet filter rules, you filter according to MAC frames.

Within a packet filter rule, you can also fall back on the definitions of the MAC services.

Entering MAC packet filter rules

Enter the firewall rules one by one in the list. Note the following parameter description.

Table 10-28 MAC rules: Parameter

Parameter	Meaning/comment	Available options / ranges of values
Action	Allow/disallow (enable/block)	• Allow Allow frames according to definition. • Drop Block frames according to definition. For firewall rules generated automatically as result of configuring a connection and subsequently adapted manually: • Allow* • Drop* If you change automatically created connection rules, when the "*" option is selected they are not recreated and overwritten by STEP 7.
From / To	Selection of the communications directions for which the rule will apply.	Described in separate sections. • For SCALANCE S modules: MAC packet filter directions SCALANCE S (Page 867) • For S7-300/S7-400/PC CPs: MAC packet filter directions S7-300-/S7-400-/PC-CPs (Page 898) • For S7-1200/S7-1500 CPs: MAC packet filter directions S7-1200 / S7-1500 CPs (Page 912) The communication direction "Station" includes access to the CP and access to the CPU via the CP. As default, the communication directions "Station" and "Tunnel" are classified as being trustworthy. For communication between trustworthy directions no firewall rules need to be configured.

Parameter	Meaning/comment	Available options / ranges of values
Source MAC address	The firewall rule is applied to the frames whose sender has the MAC address specified here. If you do not specify a MAC address, the firewall rule applies to all nodes in the communications direction you selected in the "From" column.	MAC address in the correct format
Destination MAC address	The firewall rule is applied to the frames whose recipient has the MAC address specified here. If you do not specify a MAC address, the firewall rule applies to all nodes in the communications direction you selected in the "To" column.	
Service	Selection of the MAC service or the service group to be used. Note: Before using a predefined service make sure that its predefined parameters meet your requirements.	The drop-down list box displays the predefined and configured services and service groups you can select. These are displayed in the global security settings under "Firewall" > "Services" > "Define services for MAC rules".
Bandwidth (Mbps)	Option for setting a bandwidth limitation. Can only be entered if the "Allow" action is selected. A packet passes through the firewall if the allow rule matches and the permitted bandwidth for this rule has not yet been exceeded.	CP 343-1 Adv. / CP 443-1 Adv., S7-1200-/S7-1500-CPs: 0.001 ... 100 Mbps CP 1628 and SCALANCE S: 0.001 ... 1000 Mbps For global rules: 0.001 ... 1000 Mbps Note: If the direction "From tunnel to station" is configured in a firewall rule for the S7-1200/S7-1500 CPs, no bandwidth limitation can be specified.
Logging	Enable and disable logging for this rule. If logging is enabled, the settings for the packet filter logging configured in the local security settings apply.	• Enabled • Disabled (default)
Number	Automatically assigned number for the rule. The numbers are recalculated when rules are moved.	
Comment	Space for your own explanation of the rule	If a comment is marked with "AUTO", it was created for an automatic connection rule. For rules you have created, entry of a comment is optional.

Table 10-29 Meaning of the menu commands

Button	Meaning
Delete	This deletes the selected rule or the selected global rule set. Notes on removing a globally defined and locally assigned rule set: if you delete a rule set here, this only cancels the assignment to the security module.
Save as global rule set (only for local firewall rules)	Copies the selected firewall rule(s) and inserts it as a global firewall rule set in the global security settings. The firewall configuration currently configured for the security module remains unaffected by this procedure.

Button	Meaning
Move up	Use this button to move the selected rule or selected global rule set up one position in the list. As an alternative, you can move the selected rule or the selected rule set by dragging it with the mouse. You can select multiple entries. The rule / rule set you moved is therefore handled with higher priority.
Move down	Use this button to move the selected rule or selected global rule set down one position in the list. As an alternative, you can move the selected rule or the selected rule set by dragging it with the mouse. You can select multiple entries. The rule / rule set you moved is therefore handled with lower priority.
Define service for MAC rules	This opens the dialog in which you can manage the MAC services and service groups.

IP addresses in IP packet filter rules

Entering IP addresses in IP packet filter rules

In IP packet filter rules, you have the following options for entering IP addresses:

- Nothing specified
  The rule applies to all IP addresses.
- An IP address
  The rule applies specifically to the specified address.
- Address range
  The rule applies to all the IP addresses covered by the address range.
  An address range is defined by specifying the number of valid bit places in the IP address in the format: [IP address]/[number of bits to be taken into account]
  - [IP address]/24 therefore means that only the most significant 24 bits of the IP address are included in the filter rule. These are the first three octets of the IP address.
  - [IP address]/25 means that only the first three octets and the highest bit of the fourth octet of the IP address are included in the filter rule.
- Address area
  For the source IP address, an address range can be entered in the following format:
  [Start IP address]-[End IP address]

IPv4 addresses

An IPv4 address consists of 4 decimal numbers in the range from 0 to 255 separated from each other by a dot.

Table 10-30 Examples of address ranges in IPv4 addresses

Source IP address or destination IP address	Address range		Number of addresses*)
	from	to	
192.168.0.0**/16**	192.168.0.0	192.168.255.255	65.536
192.168.10.0**/24**	192.168.10.0	192.168.10.255	256
192.168.10.0**/25**	192.168.10.0	192.168.10.127	128

Source IP address or destination IP address	Address range		Number of addresses*)
	from	to	
192.168.10.0/**26**	192.168.10.0	192.168.10.63	64
192.168.10.0/**27**	192.168.10.0	192.168.10.31	32
192.168.10.0/**28**	192.168.10.0	192.168.10.15	16
192.168.10.0/**29**	192.168.10.0	192.168.10.7	8
192.168.10.0/**30**	192.168.10.0	192.168.10.3	4
*) Note: Note that the network address and the broadcast address are not available as IP addresses of network nodes in an address range.			

IPv6 addresses

IPv6 addresses consist of 8 fields each with four hexadecimal numbers (128 bits in total). The fields are separated by a colon.

Example: fd00:ffff:ffff:ffff:ffff:ffff:2f33:8f21

Rules / simplifications:

- Leading zeros within a field can be omitted.
  Example: Instead of 2001:0db8:2426:08d3:1457:8a2e:0070:7344 it is also possible to use the notation 2001:db8:2426:8d3:1457:8a2e:70:7344.
- If one or more fields have the value 0 (or 0000), a shortened notation is possible.
  Example: Instead of 2001:0db8:0:0:0:0:1428:57ab it is also possible to use the notation 2001:db8::1428:57ab.
  To ensure uniqueness, this shortened form can only be used once within the entire address.
- Decimal notation with periods
  The last 2 fields or 4 bytes can be written in the normal decimal notation with periods.
  Example: The IPv6 address fd00::ffff.125.1.0.1 is equivalent to fd00::ffff:7d01:1.
- Address range notation in IP packet filter rules: In the same way as IPv4 addresses, IPv6 addresses can also be noted in the form of address ranges.
  Example: The entry "2001:0db8:85a3:08d3:1319:8a2e:0:0/96" covers all IPv6 addresses from 2001:0db8:85a3:08d3:1319:8a2e:0:0 to 2001:0db8:85a3:08d3:1319:8a2e:ffff:ffff.

Carrying out module-specific log settings

Log settings - overview

Module-specific function

Some logging functions are only available for certain security modules.

Log settings in the configuration

The log settings made here are loaded on the module with the configuration and take effect when the security module starts up.

You can restrict the configured packet filter log settings in the online functions if necessary. For example, you can use the online functions to specify that merely IP logging is displayed if you have configured IP and MAC logging.

Logging procedures and event classes

Here, you can specify which data should be logged. As a result, you enable logging as soon as you download the configuration to the security module.

During configuration, you also select one or both of the possible logging procedures:

- Local logging
- Network Syslog

In both logging procedures, the security module recognizes the three following types of events:

- Packet filter events
- Audit events
- System events

Configuring local logging

How to access this function

1. Select the module to be edited.
2. Select the "Log settings" > "Local log memory" entry in the local security settings.

Configuring local logging

Table 10-31 Local logging - settings for log events

Log event	Meaning	Comments
Packet filter log (firewall)	The packet filter log records certain packets of the data traffic. Only those data packets for which a configured packet filter rule (firewall) applies or to which the basic protection reacts (corrupt or invalid packets) are logged. This is only possible when logging is enabled for the packet filter rule. In the "Packets to be logged" drop-down list, you can specify the data packets to be logged: • "All packets": The data packets that are logged are those to which a configured firewall rule applies. In addition to this, all the response packets to such packets are recorded that have passed the firewall according to a configured allow rule. • "Status generating packets" (only for SCALANCE S as of V3): The only data packets that are logged are those to which a configured firewall rule (standard mode or advanced mode) applies and that initially generate a status in the firewall. Data packets that pass through the firewall by using this firewall status are not logged.	Packet filter log data is not retentive The data is stored in volatile memory on the security module and is therefore no longer available after the power supply has been turned off. For retentive storage, you can also save the log data displayed in the "Online & diagnostics" dialog in a file.
Audit log	The logging of audit events is always enabled. The logged information is always stored in the ring buffer. The audit log automatically records security-relevant events, for example user actions such as activating or deactivating packet logging or downloading configurations to the security module.	Audit log data is retentive The data is stored in a retentive memory of the security module and is therefore still available after turning off the power supply. **Note for CPs:** The audit log files are not retentive on CPs. You should therefore use a Syslog server to backup this data.
System log	The system log automatically logs successive system events, for example the start of a process or the failed login attempt of a user. Select the "Log settings" > “System events" entry to configure the event filter and cable diagnostics functions.	System log data is not retentive The data is stored in volatile memory on the security module and is therefore no longer available after the power supply has been turned off. For retentive storage, you can also save the log data displayed in the "Online & diagnostics" dialog in a file.

Table 10-32 Local logging - storage of recorded data

Storage	Meaning
Ring buffer	At the end of the buffer, the recording continues at the start of the buffer and overwrites the oldest entries.
One-shot buffer	Recording stops when the buffer is full.

Configuring system events

How to access this function

1. Select the module to be edited.
2. Select the "Log settings" > "System events" entry in the local security settings.

Filtering of the system events

In this dialog, you set a filter level for the system events. As default, the following values are set:

- SCALANCE S: Level 3 (Error)
- CP: Level 3 (Error)

The system events are recorded that have the same priority as the selected filter level and that have a higher priority than the selected filter level. The lower the value of the filter level, the higher its priority and therefore less events are recorded.

The priority of the selected filter level must be the same or lower than the severity set for the cable diagnostics; see the "Setting parameters for line diagnostics" table (not for CPs).

Recommendation: Select "Error" as the filter level or a filter level with a higher priority to prevent recording of general non-critical events.

Note for CP

For CPs, select only level 3 or level 6 since only events of these levels are generated for CPs.

- Level 0 to level 3 error messages are output if level 3 is selected.
- If you select level 6, the error messages of levels 0 to 6 are output.

Properties of the system events - line diagnostics (only for SCALANCE S)

Line diagnostics generates a special system event. A system event is generated when a selectable percentage of bad frames is reached. This system event is assigned the severity and facility set in this dialog.

Table 10-33 Setting parameters for line diagnostics

Function / option / parameter	Meaning
Enable	Enabling and disabling logging
Limit	Selectable percentage of faulty frames representing the limit at which a system event is triggered.
Facility	Select a facility from the drop-down list that identifies the system event to be logged.
Severity	Using the severity, you weight the system events of line diagnostics relative to the severity of the other system events.

Note

Severity of the system events of line diagnostics

The system events of the line diagnostics must not have a lower severity than you set for the filter. At a lower severity, these events cannot pass through the filter and will not be logged.

Configuring the network Syslog

How to access this function

1. Select the module to be edited.
2. Select the "Log settings" > "Network Syslog" entry in the local security settings.

Configuring the network Syslog

Table 10-34 Network Syslog - basic settings

Option / parameter	
Enable network Syslog	Enables and disables the transfer of logging events to the Syslog server.
Syslog server	Here, enter the IP address of the Syslog server. As an alternative, an FQDN can be entered for SCALANCE S modules as of V4. The Syslog server must be accessible from the security module at the specified address and, if applicable, using the router configuration under the "Routing" entry of the local security settings. If the Syslog server cannot be accessed no Syslog messages are sent. You can recognize this operating state on the basis of corresponding system alarms. To activate the sending of the Syslog information again, you may have to update the routing information and initiate a restart of the security module.

Option / parameter	
Enabling event classes	Enable the classes of events to be transferred to the Syslog server. You can classify packet filter and audit events according to the Severity and using the Facility according to their origin.
Module name	The module name is displayed and cannot be changed at this point.

Note

Non-secure transfer of logging events

Logging events are transferred to Syslog servers in plain language. This should be taken into account when using Syslog servers.

Table 10-35 Network Syslog - settings for log events

Log event	Configuring	Comments
Packet filter events (firewall)	The packet filter log records certain packets of the data traffic. Only those data packets for which a configured packet filter rule (firewall) applies or to which the basic protection reacts (corrupt or invalid packets) are logged. This is only possible if logging is activated for the packet filter rule. Syslog alarms can be classified according to their origin and their severity level by setting Facility and Severity. This assignment is carried out with drop-down lists. The severity and facility that you set here is assigned to every event.	The value you select here depends on the evaluation in the Syslog server. If you retain the "default" value setting, the security module specifies the combination of facility and severity with which the event is displayed.
Audit events	The audit log automatically records security-relevant events, for example user actions such as activating or deactivating packet logging or downloading configurations to the security module. Assignment of the severity and facility is carried out with drop-down lists. The severity and facility that you set here is assigned to every event.	The value you select here for the severity and facility depends on the evaluation in the Syslog server. If you retain the "default" value setting, the security module specifies the combination of facility and severity with which the event is displayed.
System events	The system log automatically logs successive system events, for example the start of a process or the failed login attempt of a user.	Select the "Log settings" > “System events" entry in the local security settings to configure the event filter and cable diagnostics functions.

Security module as router

Overview of the routing settings

Meaning

If you operate the security module in routing mode, the networks connected to the internal and external interface are transformed into separate subnets. The DMZ interface (SCALANCE S623/S627-2M only) is connected in routing mode regardless of the mode. In routing mode, the frames intended for an existing IP address in the subnet are forwarded. The firewall rules for the direction of transmission also apply.

You have the following additional options:

- Setting specific routes - can be set in the local security settings under "Routing" (SCALANCE S only), see Specifying routes (Page 869) in the section "SCALANCE S".
- Use standard router - can be set in the local security settings with "External interface [P1] red", "Internal interface [P2] green" or "DMZ interface [P3] yellow" (SCALANCE S623/ S627-2M only) see Configuring IP address parameters (Page 842) in the section "SCALANCE S".
  A maximum of one standard router can be used per security module.
- NAT/NAPT routing - can be set in the local security settings in "NAT / NAPT" (only for SCALANCE S and CP 343-1 Adv. / CP 443-1 Adv.). To be able to use NAT/NAPT, the security module must be in routing mode.

Enabling routing mode (required for SCALANCE S modules only)

For this mode, you configure an internal IP address and an internal subnet mask for addressing the router in the internal subnet in the local security settings. All network requests that do not belong to a subnet are forwarded by the security module to a different subnet.

Note: In contrast to the bridge mode of the security module, VLAN tags are lost in routing mode.

1. Select the "Routing mode" option under "Mode" in the local security settings.
2. In the local security settings, enter an internal IP address and an internal subnet mask for addressing the router on the internal subnet in the input boxes under "Internal interface [P2] green" > Ethernet addresses".

Overview of NAT/NAPT

Requirements

- The security module is in routing mode or the DMZ interface is activated (SCALANCE S623 / S627-2M only).
- Since firewall rules that enable communication in the configured address translation direction are generated automatically for NAT / NAPT rules, the advanced firewall mode must be enabled for the security module. For more detailed information, refer to section Relationship between NAT/NAPT router and firewall (Page 792)

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "NAT / NAPT".
3. When required, enable address translation according to NAT (Network Address Translation) or NAPT (Network Address Port Translation).

Address translation with NAT (Network Address Translation)

NAT is a procedure for address translation between two address spaces.
The main task is to translate private addresses into public addresses; in other words into IP addresses that are used and even routed on the Internet. As a result, the IP addresses of the internal network are not known to the outside in the external network. The internal nodes are only visible to the external network by means of external IP address that is specified in the address translation list (NAT table). If the external IP address is not the address of the security module and if the internal IP address is unique, this is known as 1:1 NAT. With 1:1 NAT, the internal address is translated to this external address without port translation. Otherwise, n:1 NAT is being used.

Address translation with NAPT (Network Address Port Translation)

The address translation with NAPT changes the target address and the target port to a communication relation (port forwarding).

Frames coming from the external network or DMZ network and intended for the IP address of the security module are translated. If the destination port of the frame is identical to one of the values specified in the "Source port" column, the security module replaces the destination IP address and the destination port as specified in the corresponding row of the NAPT table. With the reply, the security module uses the values for the destination IP address and destination port as contained in the initial frame as the source IP address and the source port.

The difference to NAT is that with this protocol ports can also be translated. There is no 1:1 translation of the IP address. There is now only a public IP address that is translated to a series of private IP addresses with the addition of port numbers.

Address translation in VPN tunnels

Address translations with NAT/NAPT can also be performed for communications relations established via a VPN tunnel. This is supported for connection partners of the type SCALANCE S612 / S623 / S627-2M as of V4.

You will find further information on address translations in VPN tunnels in the following sections:

- NAT/NAPT routing (Page 786)
- Address translation with NAT/NAPT in VPN tunnels (Page 792)

Consistency check - these rules must be adhered to

Among other things, remember the following rules to obtain consistent entries:

- The IP address of the internal interface must not be used in the NAT / NAPT table.
- An IP address used in the NAT/NAPT address conversion list must not be a multicast or broadcast address.
- The external ports assigned for the NAPT translation are in the range > 0 and ≤ 65535. Port 443 (HTTPS), 161 (SNMP), 67+68 (DHCP) and 500+4500 (IPsec) are excluded if the relevant services are activated on the security module.
- The external IP address of the security module or the IP address of the DMZ interface may only be used in the NAT table for the action "Source NAT".
- Checking for duplicates in the NAT table
  An external IP address or an IP address in the DMZ network used in the direction "Destination NAT", "Source NAT +Destination NAT" or "Double NAT" may only be used once in each specified direction.
- Checking for duplicates in the NAPT table: A source port number may only be entered once for each interface.
- Internal NAPT ports can be in the range > 0 and ≤ 65535.

See also

Overview of the routing settings (Page 784)

NAT/NAPT routing

Enabling NAT

The input boxes for NAT are enabled. NAT address translations only take effect with the entries in the address translation list described below. After creating NAT rules, the corresponding firewall rules are generated and displayed in advanced firewall mode, see section:

Relationship between NAT/NAPT router and firewall (Page 792)

If PPPoE is activated for the external interface or the DMZ interface, the action "Destination NAT" cannot be configured. When configuring the action "Source NAT", the IP address cannot be entered in the "Source translation" input box because this is obtained dynamically during runtime.

Possible address translations for NAT

The following tables show the input options for address translation with NAT.

Action "Destination NAT" - "Redirect"

The action "Destination NAT" can be performed in the following direction:

- External to internal

If the DMZ interface of the security module (SCALANCE S623/S627-2M only) is activated, the action "Destination NAT" can also be performed in the following directions:

- External to DMZ
- DMZ to internal
- DMZ to external

If the SCALANCE S module (only SCALANCE S612/S623/S627-2M as of V4) is in a VPN group and the tunnel interface is enabled, the "Destination NAT" action can also be performed in the following directions:

- Tunnel to internal
- Tunnel to external
- Tunnel to DMZ (only if the DMZ interface is activated)

The following applies, for example for the direction "external to internal": The destination IP address of a frame coming from the external network is checked to see whether it matches the IP address specified in the "Destination IP address" input box. If it matches, the frame is forwarded into the internal network by replacing the destination IP address of the frame with the IP address specified in the "Destination translation" input box. Access from external to internal using the external address is possible.

The following table shows the input required for the action "Destination NAT".

Box	Possible entries	Meaning
Source IP address	Not relevant for this action.	-
Source translation	Not relevant for this action.	-
Destination IP address	IP address in the source network	Destination IP address in the source network with which an IP address in the destination network will be accessed. The destination IP address must not match the IP address of the security module in the source network. If the destination IP address in a frame matches the address entered, the address is replaced by the corresponding IP address in the destination network. The specified destination IP address becomes the alias address. This means that the specified IP address is also registered as an IP address on the selected interface. Make sure that the alias address does not cause an IP address conflict in the network. The alias IP addresses of a security module are displayed under the entry "Alias IP addresses" of the relevant interface.
Destination translation	IP address in the destination network	The destination IP address is replaced by the IP address specified here.
No.	-	Consecutive number assigned by STEP 7 used to reference the firewall rule generated by STEP 7 for the NAT rule.

Action "Source NAT" - "Masquerading"

The action "Source NAT" can be performed in the following direction:

- Internal to external

If the DMZ interface of the security module (SCALANCE S623/S627-2M only) is activated, the action "Source NAT" can also be performed in the following directions:

- Internal to DMZ
- External to DMZ
- DMZ to external

If the SCALANCE S module (only SCALANCE S612/S623/S627-2M as of V4) is in a VPN group and the tunnel interface is enabled, the "Source NAT" action can also be performed in the following directions:

- Internal to tunnel
- External to tunnel
- DMZ to tunnel (only if the DMZ interface is activated)

The following applies, for example for the direction "internal to external": The source IP address of a frame coming from the internal network is checked to see whether it matches the IP address specified in the "Source IP address" input box. If it matches, the frame with the external IP address specified in the "Source translation" input box is forwarded to the external network as a new source IP address. In the external network, the external IP address is effective.

The following table shows the input required for the action "Source NAT".

Box	Possible entries	Meaning
Source IP address	IP address in the source network	The source IP address of the specified node is replaced by the IP address specified in the "Source translation" input box.
	IP address range / IP address band in the source network	The IP addresses of the IP address range / IP address band are replaced by the IP address specified in the "Source translation" input box.
	*	The IP addresses of all nodes in the source network are replaced by the IP address specified in the "Source translation" input box.
Source translation	IP address in the destination network	Entry of the IP address that will be used as the new source IP address. If the IP address entered here is not the IP address of the security module, this becomes an alias address. This means that the specified address is also registered as an IP address on the selected interface. Make sure that the alias address does not cause an IP address conflict in the network. The alias IP addresses of a security module are displayed under the entry "Alias IP addresses" of the relevant interface.
Destination IP address	Not relevant for this action.	-
Destination translation	Not relevant for this action.	-
No.	-	Consecutive number assigned by STEP 7 used to reference the firewall rule generated by STEP 7 for the NAT rule.

Note

You can configure an address translation to the module IP address in the destination network for all frames going from a source network to a destination network. The security module also assigns a port number for each frame. This is an n:1 NAT address translation in which multiple IP addresses of the source network are translated to one IP address of the destination network.

Enter, for example, the following parameters for the direction "internal to external":

- Action: "Source NAT"
- From: "Internal"
- To "External"
- Source IP address: "*"
- Source translation: External IP address of the security module

Action "Source NAT + Destination NAT" - "1:1-NAT"

The action "Source NAT + Destination NAT" can be performed in the following direction:

- Internal to external

If the DMZ interface of the security module (SCALANCE S623/S627-2M only) is activated, the action "Source NAT + Destination" can also be performed in the following directions:

- Internal to DMZ
- External to DMZ
- DMZ to external

If the SCALANCE S module (only SCALANCE S612/S623/S627-2M as of V4) is in a VPN group and the tunnel interface is enabled, the "Source-NAT + Destination-NAT" action can also be performed in the following directions:

- External to tunnel
- Internal to tunnel
- DMZ to tunnel (only if the DMZ interface is activated)

The following applies, for example for the direction "internal to external": When accessing from internal to external, the action "Source NAT" is performed. When accessing from external to internal, the action "Destination NAT" is performed.

The following table shows the input required for the action "Source NAT + Destination NAT":

Box	Possible entries	Meaning
Source IP address	IP address in the source network	The configuration is always specified in the source NAT direction. The IP addresses of the destination NAT direction are then inserted automatically by STEP 7.
Source translation	IP address in the destination network	
Destination IP address	Not relevant for this action.	
Destination translation	Not relevant for this action.	
No.	-	Consecutive number assigned by STEP 7 used to reference the firewall rules generated by STEP 7 for the NAT rule.

Action "Double NAT"

The action "Double NAT" can be performed for SCALANCE S modules in the following directions:

- Internal to external
- External to internal

If the DMZ interface of the security module (SCALANCE S623/S627-2M only) is activated, the action "Double NAT" can also be performed in the following directions:

- Internal to DMZ
- External to DMZ
- DMZ to internal
- DMZ to external

In every direction, Source and Destination NAT always take place at the same time.
The following applies, for example for the direction "external to internal": When accessing from external to internal, the source IP address of the external node is replaced (Source NAT). Access to the internal network also uses the external IP address specified in the "Destination IP address" input box (Destination NAT).

You can, for example, use this action if a standard router other than the security module is entered for a device to be accessed using Destination NAT. Response frames from this device are then not sent to the entered standard router but to the corresponding interface of the security module.

The following table shows the input required for the action "Double NAT":

Box	Possible entries	Meaning
Source IP address	IP address in the source network	IP address of the node in the source network
Source translation	-	The Source NAT address translation is always to the IP address of the security module in the destination network. For this reason, the "Source translation" input box cannot be configured.

Box	Possible entries	Meaning
Destination IP address	IP address in the source network	Destination IP address in the source network with which an IP address in the destination network will be accessed. If the destination IP address in a frame matches the IP address entered, the IP address is replaced by the IP address specified in the "Destination translation" input box. If the IP address entered here is not the IP address of the security module, this becomes an alias address. This means that the specified address is also registered as an IP address on the selected interface. Make sure that the alias address does not cause an IP address conflict in the network. The alias IP addresses of a security module are displayed under the entry "Alias IP addresses" of the relevant interface.
Destination translation	IP address in the destination network	The destination IP address is replaced by the IP address specified here.
No.	-	Consecutive number assigned by STEP 7 used to reference the firewall rule generated by STEP 7 for the NAT rule.

Enabling NAPT

The input boxes for NAPT are enabled. NAPT translations only take effect with the entries in the list described below. After creating NAPT rules, the corresponding firewall rules are generated and displayed in advanced firewall mode, see section:
Relationship between NAT/NAPT router and firewall (Page 792)

The IP address translation with NAPT can be performed in the following direction:

- External to internal

If the DMZ interface of the security module (SCALANCE S623/S627-2M only) is activated, the IP address translation with NAPT can also be performed in the following directions:

- External to DMZ
- DMZ to internal
- DMZ to external

If the SCALANCE S module (only SCALANCE S612/S623/S627-2M as of V4) is in a VPN group and the tunnel interface is enabled, the IP address translation with NAPT can also be performed in the following directions:

- Tunnel to internal
- Tunnel to external
- Tunnel to DMZ (only if the DMZ interface is activated)

The following applies, for example for the direction "external to internal": Frames intended for the external IP address of the security module and for the port entered in the "Source port" column are forwarded to the specified destination IP address in the internal network and to the specified destination port.

The IP address from which the frames of forwarded into the target network is displayed in the column "Virtual IP address". With a direction from external this is the external IP address of the security module and with a direction from DMZ the IP address of the DMZ interface of the security module.

The following table shows the input required for address translation with NAPT:

Box	Possible entries	Meaning
Source port	TCP/UDP port or port range Example of entering a port range: 78:99	A node in the source network can send a frame to a node in the destination network by using this port number.
Destination IP address	IP address in the destination network	Frames intended for the IP address of the security module in the source network and the TCP/UDP port specified in the "Source port" box are forwarded to the IP address specified here.
Destination port	TCP/UDP port	Port number to which the frames coming from the source network are forwarded.
Protocol	• TCP+UDP • TCP • UDP	Selection of the protocol family for the specified port numbers
No.	-	Consecutive number assigned by STEP 7 used to reference the firewall rule generated by STEP 7 for the NAPT rule.

Address translation with NAT/NAPT in VPN tunnels

Module-specific function

Address translation with NAT/NAPT in VPN tunnels is only available for SCALANCE S612/ S623/S627-2M modules as of V4, refer to the section:
Address translation with NAT/NAPT in VPN tunnels (Page 870)

Relationship between NAT/NAPT router and firewall

Meaning

After creating NAT/NAPT rules, STEP 7 automatically generates firewall rules that enable communication in the configured address translation direction. The generated firewall rules are visible in advanced firewall mode and can, if necessary, be moved and expanded (additional IP address / IP address range / IP address band, services, bandwidth). In addition to this, the automatically generated firewall rules should be checked in terms of their priority and their position. If there are also manually configured firewall rules in the rule list that have higher priority than the automatically generated firewall rules, under certain circumstances no NAT / NAPT will be performed.

If there are several identical NAT / firewall pairs of rules, the priority in the firewall rule list decides which rule is used.

Firewall parameters generated by STEP 7 cannot be adapted. After deactivating NAT/NAPT, the firewall rules generated by STEP 7 are disabled.

To clarify the relationship between the NAT/NAPT rules and the corresponding firewall rules, they are identified by corresponding, consecutive numbers.

In firewall rules that were generated automatically for NAT/NAPT rules, the “Stateful” check box cannot be disabled.

The following table shows the system behind the firewall rules generated for NAT rules for SCALANCE S modules.

Table 10-36 NAT address translation and corresponding firewall rules for SCALANCE S modules

NAT action	Created firewall rule				
	Action	From	To	Source IP address	Destination IP address
Destination NAT	Allow	Source network	Destination network	-	IP address specified in the "Destination IP address" input box
Source NAT	Allow	Source network	Destination network	IP address of the node specified in the "Source IP address" input box	-
Source NAT + Destination NAT	Allow	Source network	Destination network	IP address of the node specified in the "Source IP address" input box	-
	Allow	Destination network	Source network	-	IP address that was inserted in the "Destination IP address" input box by STEP 7
Double NAT	Allow	Source network	Destination network	IP address of the node specified in the "Source IP address" input box	IP address specified in the "Destination IP address" input box
	Allow	Source network	Destination network	IP address of the node specified in the "Source IP address" input box	IP address of the node specified in the "Destination translation" input box

The following table shows the system behind the firewall rules generated for NAT rules for CP 343-1 Adv. / CP 443-1 Adv.

Table 10-37 NAT address translation and corresponding firewall rules for CP 343-1 Adv. / CP 443-1 Adv.

NAT action	Created firewall rule				
	Action	From	To	Source IP address	Destination IP address
Destination NAT	Drop	External	Station	-	IP address of the security module in the external network
	Allow	External	Any	-	IP address of the node specified in the "Destination translation" input box
Source NAT	Allow	Any	External	IP address specified in the "Source translation" input box	-
Source NAT + Destination NAT	Allow	Any	External	IP address specified in the "Source translation" input box	-
	Drop	External	Station	-	IP address of the security module in the external network
	Allow	External	Any	-	IP address of the node that was inserted in the "Destination translation" input box by STEP 7

The following table shows the system behind the firewall rules generated for NAT rules for SCALANCE S modules.

Table 10-38 NAPT translation and firewall rules created for SCALANCE S modules

Created firewall rule					
Action	From	To	Source IP address	Destination IP address	Service
Allow	Source network	Destination network	-	IP address of the security module in the source network	[Service_NAPT_rule]

The following table shows the system behind the firewall rules generated for NAPT rules for CP 343-1 Adv. / CP 443-1 Adv.

Table 10-39 NAPT translations and created firewall rules for CP 343-1 Adv. / CP 443-1 Adv.

Created firewall rules					
Action	**From**	**To**	**Source IP address**	**Destination IP address**	**Service**
Drop	External	Station	-	IP address of the security module in the external network	[Service_NAPT_rule]
Allow	External	Any	-	IP address of the security module in the external network	[Service_NAPT_rule]

Stateful packet inspection

The firewall and NAT/NAPT router supports the "Stateful Packet Inspection" mechanism. As a result, reply frames can pass through the NAT/NAPT router and firewall without it being necessary for their addresses to be included extra in the firewall rule and the NAT/NAPT address translation.

Relationship between NAT/NAPT router and user-specific firewall

Module-specific function

The configuration of NAT/NAPT rules in the user-specific firewall is only available for SCALANCE S modules as of V3, refer to the section:

Relationship between NAT/NAPT router and user-specific firewall (Page 872)

Configuring time-of-day synchronization

Procedure for time-of-day synchronization

Meaning

For the time synchronization of the transfer of productive data, the date and time are maintained on the security module to verify the validity (time) of a certificate and for time stamping log entries. For the Security functions the time of day of the security modules of a project must be synchronized.

Note

Configuring the firewall for communication with NTP servers

If the NTP server cannot be reached by the security module, you will need to allow the frames from the NTP server explicitly in the firewall (UDP, port 123).

Note

Time-of-day synchronization relates solely to the security module and cannot be used to synchronize devices in the internal network of the security module.

Some CPs can forward the time of day to other module of the station.

The CP 443-1 OPC UA can only receive the time-of-day in the SIMATIC mode from the station but cannot forward it.

Note

Before the security functions of a CP (time-of-day slave) are used, the CP must receive a valid time-of-day frame from the time master.

Configuring time-of-day synchronization for a security module

How to access this function

1. Select the module to be edited.
2. Select the "Time-of-day synchronization" entry in the local security settings.
3. Select the "Activate time-of-day synchronization" check box.

Procedure for time-of-day synchronization

Depending on the security module being used, the following methods can be configured:

Procedure	Meaning
Set time with each download	The module time is set automatically to the PC time when a configuration is downloaded.
SIMATIC	Time-of-day synchronization by received MMS time-of-day messages (MMS = Manufacturing Message Specification) For the CP select whether the CP is to adopt or forward the time of day. The CP 1628 always forwards the time of day. The CP 443-1 OPC UA uses the SIMATIC mode if no time-of-day synchronization using NTP is configured. It can only receive the time-of-day from the station but cannot forward it. Available directions: • Automatic The CP receives the time from the station or from the LAN and forwards it to the station or to the LAN. If several CPUs are being operated in the station, the default setting can lead to collisions. To avoid this, you should specifically define the forwarding direction. • From station The CP forwards the time of day to the LAN. • From LAN The CP forwards the time of day from the LAN to the station. If forwarding of the time of day is enabled, you can use the "Use corrected time" check box to specify whether or not a correction factor included in the time-of-day frame is used. For the CP 1628, this option is is the default and it cannot be disabled.

Procedure	Meaning
Time of day from partner	The time is obtained from the relevant communications partner. Synchronization cycle: Specifies the cycle for time-of-day synchronization. For the synchronization cycle of the CP, and individual hour or minute interval can be specified.
NTP / NTP (secure)	Time-of-day synchronization by an NTP server / NTP server of the type "NTP (secure)". • Time zone: In NTP mode, it is generally UTC (Universal Time Coordinated) that is transferred. This corresponds to GMT (Greenwich Mean Time). The time offset from UTC can be set by configuring the local time zone. • Update interval in seconds: Specifies the interval between the time queries in seconds. For SCALANCE S as of V3, the time interval for querying the NTP server is specified automatically. If the "Enable security functions" check box is enabled in the local security settings of a CP, the setting for the update interval is transferred from the CP's local settings into the CP's local security settings. • Time synchronization on the full minute: With this option, you specify that the time of day is forwarded to the communications bus on the full minute. This option is required for certain special applications. • Accept time from non-synchronized NTP servers Here you can specify whether the security module also accepts the time-of-day from non-synchronized NTP servers. • Forward time of day to station: Disable this option if the CPU requests the time separately from an NTP server. This prevents the time on the CPU obtained directly from the NTP server from being overwritten by the time detected in the CP. The accuracy may be reduced slightly due to forwarding via the CP. • NTP server: Creating NTP servers in the local security settings is described in the section Configuring NTP servers (Page 799).

Configuring NTP servers

Creating NTP servers of the type "NTP (secure)" in the global security settings

In the global security settings only NTP servers of the type "NTP (secure)" can be created and assigned to CPs or SCALANCE S modules as of V4.

1. In the global security settings, select the entry "NTP".
   Result: The previously created NTP servers (secure) are displayed under the selected entry.
2. Double-click on the "Add new NTP server" entry.
   Result: The created NTP server (secure) is displayed with an automatically assigned number below the entry "NTP".
3. Double-click on the created NTP server (secure).
   Result: In the working area the assignment dialog for NTP servers (secure) is displayed. The selected NTP server (secure) is selected in the "NTP server" drop-down list. The required security modules can be assigned to this, see section Assigning the security module to an NTP server (secure) (Page 800). In the local security settings, the configurable properties of the NTP server (secure) are displayed.
4. Enter the following data in the "General" entry in the local security settings:
   - Name: A maximum of 25 characters can be used for the name of the NTP server (secure).
   - IP address / FQDN: Address of the NTP server (secure).
5. Enter the following encryption parameters in the "NTP key" entry in the local security settings:

Property	Meaning
Key ID	Numeric value between 1 and 65534.
Authentication	Select the authentication algorithm.
Hex/ASCII	Select the format for the NTP key.
Key	Enter the NTP key with the following lengths: Hex: 22 ... 40 characters ASCII: 5 ... 20 characters

To switch over to the local security settings of another NTP server (secure), select this from the "NTP server" drop-down list of the assignment dialog or double-click on the corresponding entry in the global security settings.

Result

You have created the NTP server (secure) and can now assign this to the required security modules.
Note the descriptions in the following section:

Creating non-secure NTP servers in the local security settings

Non-secure NTP servers for SCALANCE S modules below V4 and CPs need to be created in the local security settings.

1. Select the module to be edited.
2. Select the "Time-of-day synchronization" entry in the local security settings.
3. Select the synchronization method..
4. Enter a name and the IP address of the NTP server. A maximum of 25 characters can be used for the name. If you have selected the "NTP (secure)" synchronization mode, you can select an NTP server (secure) that you created in the global security settings in the "Name" column.

Configuration limits for NTP servers

You can assign a maximum of 4 NTP servers to one security module.

Conventions for names of NTP servers

The characters 0-9, a-z, A-Z, -._ can be used for the names of NTP servers. A name must start with a letter and end with a letter or number. Before the '.' character only letters and numbers are permitted, after the '.' character only letters. Before the '_' or '-* character only letters and numbers are permitted.

Importing/exporting NTP servers (secure)

Using the "Import" or "Export" commands in the shortcut menu, you can export the key list of the currently selected NTP server (secure) and import the file into an NTP server (secure) or vice versa.

Assigning the security module to an NTP server (secure)

Requirement

- You have defined an NTP server (secure) in the global security settings.
- "NTP" or "NTP (secure)" is selected as the time-of-day synchronization mode in the local security settings of the security module that you want to assign to an NTP server (secure).

Procedure

1. Double-click the "NTP" entry in the global security settings.
2. Double-click the entry "Assign module to an NTP server".
3. From the "NTP Server" drop-down list, select the NTP server (secure) to which you want to assign a security module.

4. In the "Available modules" section, select the security module that you want to assign to the selected NTP server (secure).
5. Click the "<<" button to assign the selected security module to the selected NTP server (secure).

Result

You have assigned the security module to the NTP server (secure). The NTP server (secure) is displayed automatically in the local security settings in the list of NTP servers.

Security module as DHCP server

Module-specific function

The use of the security module as a DHCP server is only possible with SCALANCE S modules, see subsection:
AUTOHOTSPOT in the section "SCALANCE S".

Configuring SNMP

Overview of SNMP

What is SNMP?

The security module supports the transfer of management information using the Simple Network Management Protocol (SNMP). For this purpose, an SNMP agent that receives and responds to SNMP requests is installed on the security module. The information on the properties of SNMPcompliant devices is entered in MIB files (MIB = Management Information Base) for which the user must have the required rights.

In SNMPv1, the "community string" is also sent. The "community string" is like a password that is transmitted along with the SNMP request. If the community string is correct, the security module replies with the required information. If the string is incorrect, the security module discards the query and does not reply. The community string is transmitted via SNMPv1 without encryption.

SNMPv3 lets you transmit encrypted data.

Configuring SNMP - "SNMP" entry

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "SNMP".
3. Activate the "Activate SNMP" check box.

4. Select one of the following SNMP protocol versions:

   Note
   Encrypted data transmission with SNMPv3
   You should use SNMPv3 to transmit data in encrypted form in order to enhance security.

   – SNMPv1
     The security module uses the following default values for the community strings to control the access rights in the SNMP agent. These default values should be adapted to increase security.
     For read access: public
     For read and write access: private
     To enable write access using SNMP, select the "Allow write access" option.
   – SNMPv3
     Select either only an authentication algorithm or an authentication algorithm and an encryption algorithm.
     Authentication algorithm: none, MD5, SHA-1
     Encryption algorithm: none, AES-128, DES

     Note
     Preventing the use of DES
     DES is an insecure encryption algorithm. Therefore, it should only be used for reasons of down compatibility.

     Note
     When using SNMPv3 no RADIUS authentication is possible.

5. If SNMPv3 is to be used, assign a user a role with corresponding activated SNMP rights to enable access to the module via SNMP. An overview of SNMP rights is available in the section:
   Managing rights (Page 737).
6. In the "Advanced settings" area, for SCALANCE S modules configure module-specific information about the author, location and e-mail address overwriting the information from the project properties.
   The following applies to values written to the security module by an SNMP tool using an SNMP-SET command:
   If you select the "Keep values written by SNMP set" check box, the values are not overwritten when there is another download of a STEP 7 configuration to the security module.

Configuring proxy ARP

Module-specific function

This function is available only for SCALANCE S V3 modules or higher, see section Configuring proxy ARP (Page 876)

Activate Web server on security module

Module-specific function

This function is only available for CP 343-1 Advanced / CP 443-1 Advanced and CP 443-1 OPC UA see section:
Activating Web server on security module (Page 902)in the section "Security for S7-300 / S7-400 / PC CPs".

IPsec tunnel: Creating and assigning VPN groups

How to create an IPsec tunnel with VPN groups

Requirement

Note

Current date and current time of day on the security modules

When using secure communication (for example, HTTPS, VPN...), make sure that the security modules involved have the current time of day and the current date. Otherwise the certificates used are not evaluated as valid and the secure communication does not work.

Creating a VPN group

The security of the default settings of VPN groups was increased in V14 SP1. If in a V14 project a new VPN group should be given these default settings, the project needs to be upgraded to V14 SP1 when this VPN group is created. If the project is not upgraded the VPN group will be created with the default settings of V14. For further information on upgrading, refer to section Create users (Page 733)

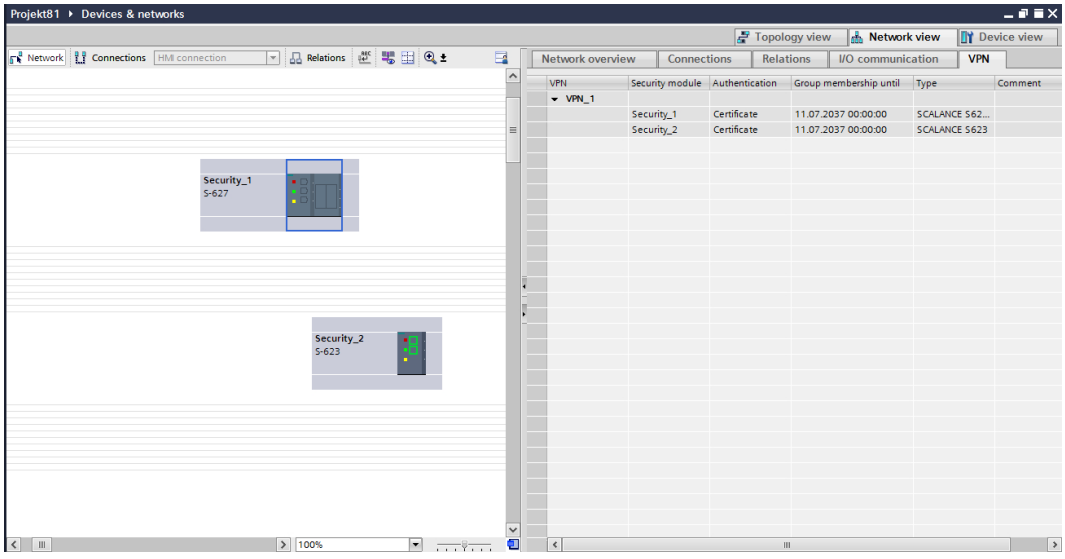
1. In the global security settings, select the entry "VPN groups".
   Result: The previously created VPN groups are displayed under the selected entry.
2. Double-click on the "Add new VPN group" entry.
   Result: The created VPN group is displayed with an automatically assigned number below the entry "VPN groups".
   As an alternative, in the network view you can right click on an interface of a module with VPN capability and create a VPN group with the shortcut menu command "Add new VPN group" (not possible for SOFTNET Security Client). The selected security module is automatically assigned to the created VPN group.
3. Double-click on the created VPN group.
   Result: In the working area the assignment dialog for VPN groups sets is displayed. The selected VPN group is selected in the "VPN" drop-down list. Security modules can be assigned to this in the relevant assignment dialog, see section Assigning a security module to a VPN group (Page 805). In the local security settings, the VPN group properties of the selected VPN group can be adapted, see section Configuring VPN group properties (Page 808).

Conventions for names of VPN groups

The characters 0-9, a-z, A-Z, -._ can be used for the names of VPN groups. A name must start with a letter and end with a letter or number. Before the '.' character only letters and numbers are permitted, after the '.' character only letters. Before the '_' or '-* character only letters and numbers are permitted.

Display of the VPN groups with their properties

If you select a security module that is in one or more VPN group(s), the properties of the VPN group(s) of which the security module is a member are shown in the "Network data" area.



The following properties of the VPN groups are displayed in columns in the "VPN" tab of the "Network data" area:

Property/column	Meaning
VPN	Names of the VPN groups of which the selected security module is member
Security module	Names of the assigned security modules
Authentication	Type of authentication: Pre-shared key or certificate
Group membership until	Date and time up to which the VPN group certificate of the security module is valid
Type	Model numbers of the assigned security modules
Comment	Comment

Setting the life of certificates

Open the dialog in which you can set the expiry date of the certificate as follows:

1. Select the VPN group you want to edit in the "VPN" tab.
2. In the "Properties" > "General" tab of the Inspector window, select the entry "Authentication".

Note

Expiry of a certificate

Communication through the VPN tunnel continues after the certificate has expired until the tunnel is terminated or the SA lifetime expires. For more information on certificates, refer to section:
AUTOHOTSPOT.

Configuration limits

Number of the VPN tunnel	
SCALANCE S612 V3/V4	Maximum of 128
SCALANCE S623 V3/V4	Maximum of 128
SCALANCE S627-2M V4	Maximum of 128
CP 343-1 Advanced / CP 443-1 Advanced	Maximum of 32
CP 1628	Maximum of 64
S7-1200/S7-1500 CPs	Maximum of 16

For the number of communications relations between subnets of the security module and subnets of other security modules that can be reached via VPN tunnel the following configuration limits apply:

Number of subnet relations	
SCALANCE S612 V3/V4	Maximum of 128
SCALANCE S623 V3/V4	Maximum of 128
SCALANCE S627-2M V4	Maximum of 128
CP 343-1 Advanced / CP 443-1 Advanced	Maximum of 32
CP 1628	Maximum of 64
S7-1200 / S7-1500 CPs	Maximum of 8

Assigning a security module to a VPN group

Requirement

You have created a VPN group in the global security settings. Before assigning security modules to VPN groups, note the rules for this in the section Modes of VPN groups (Page 815).

Procedure

1. Double-click on the "VPN groups" entry in the global security settings.
2. Double-click on the “Assign module to a VPN group" entry.
3. From the "VPN" drop-down list, select the VPN group to which you want to assign a security module.
4. In the "Available modules" section, select the security module that you want to assign to the selected VPN group.
5. Click the "<<" button to assign the selected security module to the selected VPN group.
   If the security module doe not support one or more properties of the VPN group box, the module symbol in the area of the assigned modules is displayed with an error symbol .
   As an alternative to the procedure , in the network view you can assign a security module to an existing VPN group by right clicking on an interface of a module with VPN capability and assign this to the VPN group using the shortcut menu "Assign module to a VPN group" (not possible for SOFTNET Security Client).

Result

You have assigned the security module to the VPN group.

Authentication methods

The following methods are available:

The authentication method is specified per VPN group and decides the type of authentication used.

The following key-based or certificate-based authentication methods are supported:

- Pre-shared key
  Authentication is achieved using a previously specified character string that is distributed to all modules in the group.
  Enter a pre-shared key in the "Key" field under "Authentication" > "General" in the VPN group properties dialog. If you click the "Renew preshared key" button, STEP 7 renews the pre-shared key to be used.
- Certificate
  Certificate-based authentication "Certificate" is the default setting. The procedure is as follows:
  - When a group is created, a CA certificate is generated as a root certificate.
  - Each security module that is a member of the VPN group also receives a group certificate signed with the key of the CA certificate.

  If you click the "Renew certificate " button, STEP 7 renews the CA certificate of the VPN group.
  All certificates are based on the ITU standard X.509v3 (ITU, International Telecommunications Union).
  The certificates are generated by a certificate authority contained in STEP 7.

Note

Restriction in VLAN operation

No VLAN tagging is transmitted in IP frames via the VPN tunnel of the security module. The VLAN tags included in IP frames are lost when they pass through the security modules because IPsec is used to transfer the IP frames.

As default, no IP broadcast or IP multicast frames can be transferred with IPsec through a layer 3 VPN tunnel. Through a layer 2 VPN tunnel of the security module, IP broadcast or IP multicast frames are packaged just like MAC packets including the Ethernet header in UDP and transferred. With these packets, the VLAN tagging is therefore retained.

VPN group properties

VPN connection establishment by Known Peers and Unknown Peers

The properties of a VPN group are made up of the selected authentication method and the configured parameters for IKE phase 1 and 2 and during loading are transferred to all VPN group nodes. On the responder the IP addresses of the security modules that may initiate the VPN connection establishment with the configured properties of the VPN group are stored in addition. When a security module attempts to establish a VPN connection to the responder, this first checks whether the IP address of the initiator is known to it. If this is the case, the initiator is a Known Peer. If the VPN group properties proposed by this Known Peer match those stored on the responder for the Known Peer, the responder allows the VPN connection establishment. In addition to the properties of a VPN group, fallback VPN profiles can be assigned to a VPN group. VPN profiles contain predefined properties for the authentication method, IKE phase 1 and 2. Fallback VPN profiles can be used as alternative configurations for VPN connection establishment. These come into play when VPN properties of the initiator do not match the VPN properties of the responder. In such cases, fallback VPN profiles ensure the VPN connection establishment. The configured fallback VPN profiles just as the properties of the VPN group are transferred to all VPN group nodes. The priority for the use of the fallback VPN profiles is specified by their order in the configuration. The fallback VPN profiles and the properties of the VPN group must not differ in the authentication method.

If the IP address of the initiator is not known to the responder because the initiator in productive mode for example is connected via a NAT translation, the initiator is an Unknown Peer (Road Warrior). As default the responder allows connection establishment by Unknown Peers. In the module-specific VPN properties of the responder you can specify whether and with which VPN group properties connection establishment by Unknown Peers is allowed by the responder, see section Configuring module- and connection-specific VPN settings (Page 819).

The configuration of fallback VPN profiles and responder setting for Road Warrior connections is only possible in V14 SP1 projects. For information on upgrading projects, refer to section Create users (Page 733)

Configuring VPN group properties

VPN group properties

Note

Knowledge of IPsec necessary

To be able to set these parameters, you require IPsec experience. If you do not make or modify any settings, the defaults apply.

The following settings can be configured in the properties of a VPN group:

- Authentication method (entry: "General")
- IKE settings (entry: "Advanced settings phase 1")
- IPsec settings (entry: "Advanced settings phase 2")
- Fallback VPN profiles

The configuration of all available settings of phase 1 and phase 2 and the configuration of fallback VPN profiles is only possible in V14 SP1 projects. For information on upgrading projects, refer to section Create users (Page 733).

If the VPN group properties you have configured are less secure than the recommended properties, STEP 7 makes this known to you with an appropriate message text.

How to access this function

1. In the global security settings select the VPN group whose properties you want to configure under the entry "VPN groups".
2. Select "Open" in the shortcut menu of this entry.
   Result: The VPN group properties appear in the area of the local security settings.
3. In the "Authentication" entry, select whether to use a pre-shared key or certificate for authentication. For more detailed information, refer to section:
   Authentication methods (Page 806).

Advanced settings phase 1

Phase 1: IKE negotiation of the Security Association (SA) for phase 2:

Here you set the parameters for negotiating the security parameters to be used in phase 2:

Parameter	Description
IKE mode	• Main (default setting) • Aggressive* Selection of the mode IKE phase 1. In Main mode, the connection partners exchange their identities encrypted, in Aggressive mode unencrypted. In Aggressive mode KE phase 1 runs faster but is less secure than in the alternative Main mode.
DH group	Groups selectable for the Diffie-Hellman Key exchange: • Group 15 • Group 14 (default setting) • Group 5* • Group 2* • Group 1*
SA lifetime type	Phase 1 Security Association (SA): • Time: Time limit in minutes The lifetime of the current key material is limited in time. When the time expires, the key material is renegotiated.
SA lifetime	Numeric value: Range of values for time: 1440 … 2500000 minutes (default setting: 2879)

Parameter	Description
Encryption	Encryption algorithm: • AES-256 (default setting), 192, 128: Advanced Encryption Standard (256 bit, 192 bit or 128 bit key length, mode CBC) • 3DES-168*: Triple DES (168-bit key length, mode CBC) • DES*: Data Encryption Standard (56 bit key length, mode CBC)
Authentication	Authentication algorithm: • SHA-512: Secure Hash Algorithm 512 • SHA-384: Secure Hash Algorithm 384 • SHA-256 (default setting): Secure Hash Algorithm 256 • SHA-1*: Secure Hash Algorithm 1 • MD5*: Message Digest Algorithm 5

* The setting is classified as less secure. It is recommended when possible that you use a more secure setting.

Advanced settings phase 2

Phase 2: IKE negotiation of the Security Association (SA) for IPsec data exchange:

Here, you set the parameters for negotiating the security parameters used for the IPsec data exchange with ESP (Encapsulating Security Payload) and AH (Authentication Header). Communication in phase 2 is already encrypted.

Parameter	Description
SA lifetime type	Phase 2 Security Association (SA): • Time (default setting): Time limit in minutes. The lifetime of the current key material is limited in time. When the time expires, the key material is renegotiated. • Limit: Limitation of the data volume in MB
SA lifetime	Numeric value: • Range of values for time: 60 … 16666666 minutes (default setting: 2879) • Range of values for limit: 2000 ... 500000 MB (default setting: 4000)
Encryption	Encryption algorithm: • AES-128 (default setting): Advanced Encryption Standard (128-bit key length, mode CBC) • 3DES-168*: Triple DES (168-bit key length, mode CBC) • DES*: Data Encryption Standard (56 bit key length, mode CBC)

Parameter	Description
Authentication	Authentication algorithm: • SHA-512: Secure Hash Algorithm 512 • SHA-384: Secure Hash Algorithm 384 • SHA-256 (default setting): Secure Hash Algorithm 256 • SHA-1*: Secure Hash Algorithm 1 • MD5*: Message Digest Algorithm 5
Perfect Forward Secrecy	• ON (default setting): Diffie Hellmann public key values are exchanged for recalculation of the keys. • OFF*: The values already exchanged in phase 1 are used for recalculation of the keys.

* The setting is classified as less secure. It is recommended when possible that you use a more secure setting.

Configuring fallback VPN profiles

The configuration of fallback VPN profiles is only possible in V14 SP1 projects. For information on upgrading projects, refer to section Create users (Page 733).

In addition to the properties of a VPN group, fallback VPN profiles can be assigned to a VPN group. VPN profiles contain predefined properties for the authentication method, IKE phase 1 and 2. Fallback VPN profiles can be used as alternative configurations for VPN connection establishment. These come into play when VPN properties of the initiator do not match the VPN properties of the responder. In such cases, fallback VPN profiles ensure the VPN connection establishment. The configured fallback VPN profiles just as the properties of the VPN group are transferred to all VPN group members. The priority for the use of the fallback VPN profiles is specified by their order in the configuration. The fallback VPN profiles and the properties of the VPN group must not differ in the authentication method.

The following fallback VPN profiles are available:

Table 10-40 VPN profile 1

Parameter	Setting
Authentication method	Certificate
IKE mode	Main
Phase 1 DH group	Group 14
Phase 1 encryption	AES-256
Phase 1 SA lifetime	SA lifetime proposed by the initiator: 480 minutes Range permitted by the responder for the SA lifetime: 480 … 2880 minutes
Phase 1 authentication	SHA-1
Phase 2 SA lifetime type	Time
Phase 2 encryption	AES-128
Phase 2 SA lifetime	SA lifetime proposed by the initiator: 240 minutes Range permitted by the responder for the SA lifetime: 60 … 2880 minutes

Parameter	Setting
Phase 2 authentication	SHA-1
Perfect Forward Secrecy	OFF

Table 10-41 VPN profile 2

Parameter	Setting
Authentication method	Certificate
IKE mode	Main
Phase 1 DH group	Group 2
Phase 1 encryption	AES-256
Phase 1 SA lifetime	SA lifetime proposed by the initiator: 480 minutes Range permitted by the responder for the SA lifetime: 480 … 2880 minutes
Phase 1 authentication	SHA-1
Phase 2 SA lifetime type	Time
Phase 2 encryption	3DES-168
Phase 2 SA lifetime	SA lifetime proposed by the initiator: 2880 minutes Range permitted by the responder for the SA lifetime: 60 … 2880 minutes
Phase 2 authentication	SHA-1
Perfect Forward Secrecy	OFF

Table 10-42 VPN profile 3

Parameter	Setting
Authentication method	Certificate
IKE mode	Main
Phase 1 DH group	Group 2
Phase 1 encryption	3DES-168
Phase 1 SA lifetime	SA lifetime proposed by the initiator: 480 minutes Range permitted by the responder for the SA lifetime: 480 … 2880 minutes
Phase 1 authentication	SHA-1
Phase 2 SA lifetime type	Time
Phase 2 encryption	3DES-168
Phase 2 SA lifetime	SA lifetime proposed by the initiator: 2880 minutes Range permitted by the responder for the SA lifetime: 60 … 2880 minutes
Phase 2 authentication	SHA-1
Perfect Forward Secrecy	OFF

Table 10-43 VPN profile 4

Parameter	Setting
Authentication method	Certificate
IKE mode	Main
Phase 1 DH group	Group 2
Phase 1 encryption	DES
Phase 1 SA lifetime	SA lifetime proposed by the initiator: 480 minutes Range permitted by the responder for the SA lifetime: 480 … 2880 minutes
Phase 1 authentication	MD5
Phase 2 SA lifetime type	Time
Phase 2 encryption	3DES-168
Phase 2 SA lifetime	SA lifetime proposed by the initiator: 2880 minutes Range permitted by the responder for the SA lifetime: 60 … 2880 minutes
Phase 2 authentication	SHA-1
Perfect Forward Secrecy	OFF

Table 10-44 VPN profile 5

Parameter	Setting
Authentication method	Pre-shared key
IKE mode	Main
Phase 1 DH group	Group 2
Phase 1 encryption	3DES-168
Phase 1 SA lifetime	SA lifetime proposed by the initiator: 480 minutes Range permitted by the responder for the SA lifetime: 480 … 2880 minutes
Phase 1 authentication	SHA-1
Phase 2 SA lifetime type	Time
Phase 2 encryption	3DES-168
Phase 2 SA lifetime	SA lifetime proposed by the initiator: 2880 minutes Range permitted by the responder for the SA lifetime: 60 … 2880 minutes
Phase 2 authentication	SHA-1
Perfect Forward Secrecy	OFF

After opening a STEP 7 project, that was last stored with an older version than V14 SP1, all existing VPN groups with the authentication method "Certificate" are assigned the predefined fallback VPN profile 2 - 5 as default.

Special features for the SOFTNET Security Client and SCALANCE M

Configured fallback VPN profiles are not included in the configuration files for the Road Warrior modules SOFTNET Security Client and SCALANCE M For the SOFTNET Security Client and SCALANCE M modules the following applies:

- Select the VPN profile to be used for VPN connections of SOFTNET Security Client / SCALANCE M modules to SCALANCE S / CPs in the responder settings, see section Configuring module- and connection-specific VPN settings (Page 819).
- For VPN connections from SCALANCE M modules to SCALANCE M modules, the configured VPN group settings are used.
- For VPN connections from SOFTNET Security Client to SCALANCE M modules the settings of VPN profile 3 are always used.

Unsupported VPN group properties

The following VPN group properties are not supported by the individual modules:

Module	Phase 1					Phase 2			
	IKE mode	DH group	SA lifetime	Encryption	Authentication	SA lifetime type	SA lifetime	Encryption	Authentication
CP 1543-1 as of V2.1	-	-	Minutes outside the range 1440-2500000	-	SHA-384 SHA-512	-	Limit: MB outside the range 2000-500000	-	SHA-384 SHA-512
SOFTNET Security Client (Windows XP)	Aggressive	Group 1 Group 5 Group 14 Group 15	Minutes outside the range 1440-2879	AES-128 AES-192 AES-256	SHA-256 SHA-384 SHA-512	Limit	Time: Minutes outside the range 60-2879	AES-128	MD5 SHA-256 SHA-384 SHA-512
SOFTNET Security Client (Windows 7)	Aggressive	Group 1 Group 5 Group 14 Group 15	Minutes outside the range 1440-2879	-	SHA-256 SHA-384 SHA-512	Limit	Time: Minutes outside the range 60-2879	-	MD5 SHA-256 SHA-384 SHA-512

Module	Phase 1					Phase 2			
	IKE mode	DH group	SA lifetime	Encryption	Authentication	SA lifetime type	SA lifetime	Encryption	Authentication
SCALANCE M875	Aggressive	Group 1 Group 5 Group 14 Group 15	Minutes outside the range 1-2147483647	AES-128 AES-192 AES-256 DES	MD5 SHA-256 SHA-384 SHA-512	Limit	Time: Minutes outside the range 1-2147483647	AES-128 DES	MD5 SHA-256 SHA-384 SHA-512
Security device	-	Group 1	Minutes outside the range 1-2500000	-	-	-	Limit: MB outside the range 0-500000 Time: Minutes outside the range 1-16666666	-	-

- No restriction

The following applies to the other security modules: VPN group properties with a DH group higher than group 14 and/or authentication higher than SHA-1 are not supported.

Modes of VPN groups

VPN modes

Depending on the mode of the security modules that were added to a VPN group, different modes of VPN groups are distinguished. The mode of a VPN group provides information about the security modules and their modes that can be added to the VPN group.

Rules for forming groups

Remember the following rules if you want to create VPN groups:

- For SCALANCE S612 / S613 / S623 / S627-2M / SCALANCE M / VPN device
  The first module assigned in a VPN group decides which other modules can be added to it.
  If the first SCALANCE S module to be added is in routing mode or if the first module is a SCALANCE M module or a VPN device, only additional SCALANCE S modules with routing activated or SCALANCE M modules cor VPN devices an be added because SCALANCE M modules and VPN devices always operate in routing mode. If the first SCALANCE S module added is in bridge mode, you can only add other SCALANCE S modules in bridge mode. If you want to change the mode of a VPN group, you have to remove all the modules contained in the group and add them again. A CP and an SSC can be added to a group with a SCALANCE S in bridge or routing mode.
- For CP / SSC
  If a CP / SSC is the first module in a VPN group, security modules can be added in any mode. The next module that decides the mode then also decides the mode of the VPN group. A CP / SSC can be assigned to several VPN groups at the same time and use different modes. The CP / SSC is then operated in mixed mode.
- It is not possible to add a SCALANCE M module to a VPN group that contains a module in bridge mode.

Refer to the following table to see which modules can grouped together in a VPN group:

Table 10-45 Security modules and VPN modes

Module	Can be included in a VPN group in...	
	Bridge mode	Routing mode
SCALANCE S612 / S613 / S623 / S627-2M in bridge mode	x	-*
SCALANCE S612 / S613 / S623 / S627-2M in routing mode	-	x
CP 343-1 Adv. / CP 443-1 Adv.	x	x
S7-1200/S7-1500 CPs	x	x
CP 1628	x	x
SOFTNET Security Client V4.0	x	x
SCALANCE M875 / VPN device	-	x

* SCALANCE S623/S627-2M modules in bridge mode can be inserted in a VPN group in routing mode if their DMZ interface is activated (not at the same time).

Including security module in configured VPN group

The configured VPN group properties are adopted for security modules added to in an existing VPN group.

Procedure after including a security module in a configured VPN group

Depending on whether or not you have changed the VPN group properties since the last download, you must make a distinction between the following:

- **Case a:** If you have not changed the VPN group properties and the module to be added actively establishes the connection to modules of the type SCALANCE S, CP 343-1 Adv. / CP 443-1 Adv. or CP 1628:

1. Add the new security module to the VPN group.
2. Download the configuration to the new module.

- **Case b:** If you have changed the VPN group properties or the module being added does not actively establish the connection to the already configured modules:

1. Add the new security module to the VPN group.
2. Download the configuration to all modules that belong to the VPN group.

In case a, it is not necessary to reconfigure and load the already commissioned security modules. Active communication is not influenced or interrupted.

Procedure after removing an active member from a VPN group

If you remove an active node from an existing VPN group, this can still establish a connection to the group members even if you have downloaded the project to all members of the VPN group again.

If you do not want the removed active node to be able to establish the connection any longer, renew the CA group certificate and download the project again to the members of the VPN group. The certificate can be renewed in the group properties of the VPN group or in the "CA" tab of Certificate Manager.

Configuring network nodes reachable through a tunnel

Overview

Configuring network nodes reachable through a tunnel

Each security module must know the network nodes reachable via a VPN tunnel to be able to recognize the authenticity of a frame.

The security module must know both its own nodes as well as the nodes of the security modules in the same VPN group. This information is used on a security module to decide which data packet will be transferred in which tunnel.

By adding a security module to a VPN group, the local internal nodes/subnets of the security module are automatically made known to the security module. To allow communication via the VPN tunnel with other subnets or nodes of another subnet (routed internal network; DMZ network when the VPN tunnel is on the external interface and vice versa), these subnets or nodes must be enabled for VPN tunnel communication in the configuration.

SCALANCE S modules allow network nodes to be learned automatically or configured statically. The mode of the security module also decides the options available for learning internal network nodes.

SCALANCE S in bridge mode

In bridge mode, you can configure the IP/MAC nodes and the subnets or alternatively can allow automatic learning of nodes by the SCALANCE S.

SCALANCE S in routing mode

In routing mode there is no automatic learning mode available. Instead, enter entire subnets here that need to be released for tunnel communication.

CP 343-1 Advanced / CP 443-1 Advanced and CP 1628

- CP 343-1 Advanced / CP 443-1 Advanced
  Decide whether or not tunnel communication to the CP (Gbit interface) and/or to the internal subnet (PROFINET subnet) is permitted for VPN connection partners in routing mode (SCALANCE S / M).
- CP 1628
  Enter the NDIS nodes you want to be reachable through the tunnel of VPN connection partners in routing mode (SCALANCE S / M).

CP 1543-1 as of V2.1 with activated IP routing between communications modules

Enter the subnets or IP addresses reachable via the backplane bus of the station and that need to be released for tunnel communication.

Automatic learning of internal network nodes

Module-specific function

SCALANCE S modules in bridge mode provide a learning mode with which the internal network nodes can be learned automatically during operation. For more detailed information, refer to subsection:
Using the learning mode to learn internal nodes (Page 877) in the section "SCALANCE S".

Configuring IP network nodes manually for SCALANCE S

Module-specific function

How to configure IP network nodes for SCALANCE S modules manually is explained in:
Configuring IP network nodes manually (Page 878) in the section "SCALANCE S".

Configuring MAC network nodes manually for SCALANCE S

Module-specific function

How to configure MAC network nodes for SCALANCE S modules manually is explained in Configuring MAC network nodes manually (Page 879) in the section "SCALANCE S".

Configuring internal subnets for SCALANCE S manually

Module-specific function

How to configure the internal subnets for SCALANCE S modules is explained in Configuring internal subnets manually (Page 879) in the section "SCALANCE S".

Allow access to S7-300 / S7-400 CPs for VPN connection partners

Module-specific function

How to allow access to S7-300 / S7-400 CPs for VPN connection partners is explained in Allow access to S7-300 / S7-400 CPs for VPN connection partners (Page 903) in the section "Security for S7-300 /S7-400 /PC CPs".

Configuring NDIS nodes for PC CPs that can be reached through the tunnel

Module-specific function

How you configure NDIS nodes for PC CPs that can be reached through the tunnel is explained in
Configuring NDIS nodes manually for PC CPs that can be reached through the tunnel (Page 903) in the section "Security for S7-300 / S7-400 / PC CPs".

Configuring subnets reachable via the backplane bus for CP 1543-1

Module-specific function

For information on how to configure subnets and network nodes reachable via the backplane bus of the station for CP 1543-1 as of V2.1, see section
Configuring subnets reachable via the backplane bus for CP 1543-1 (Page 914).

Configuring module- and connection-specific VPN settings

Requirement

The module is a member of a VPN group.

Module- and connection-specific settings

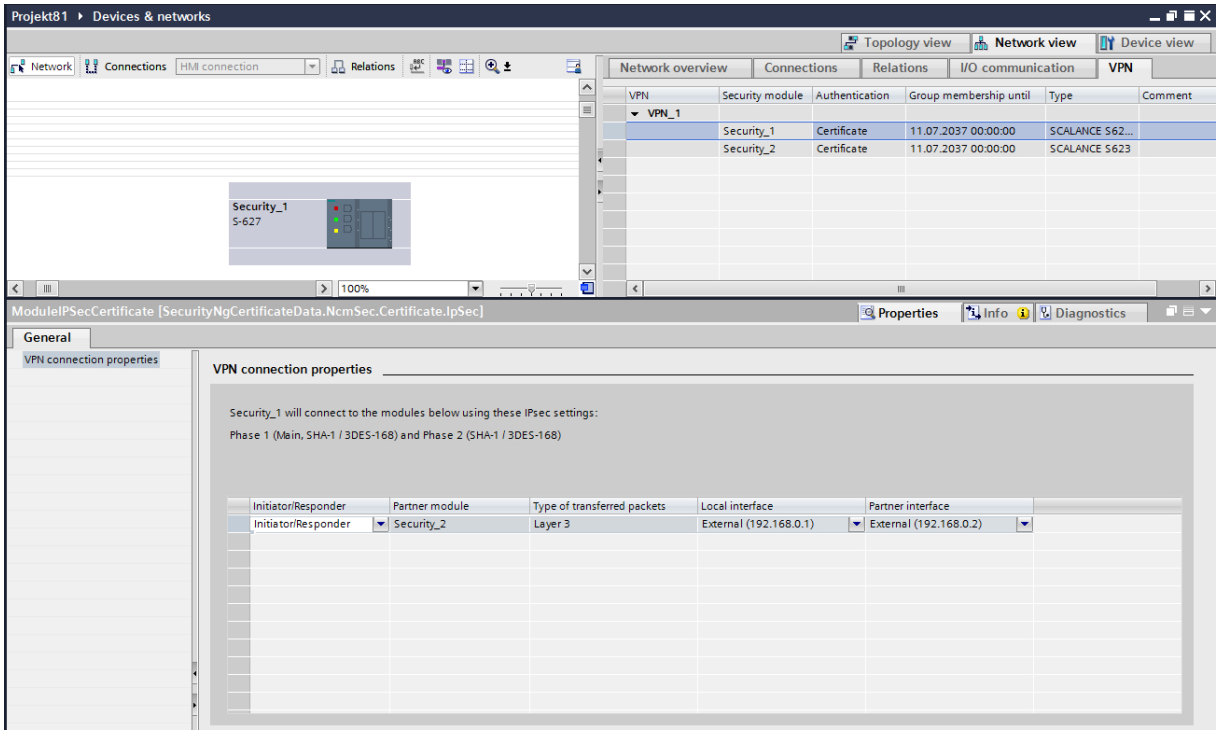
With module- and connection-specific settings, you can configure specific VPN settings. Module-specific settings are configured specifically for a security module while connection-specific settings are configured specifically for a security module in a certain VPN group.

You can configure the following **module-specific** properties in the local security settings with the "VPN" entry:

- Dead peer detection
- Permission to initiate connection establishment
- Public address (IP address / FQDN) for communication via Internet gateways
- Responder settings for Road Warrior connections
- Nodes that need to be enabled for tunnel communication

If you select a security module in the list of VPN groups in the "Network data" area, the following **connection-specific** VPN settings can be seen and can be configured:

- Permission to initiate connection establishment
- Partner modules to which tunnel connections exist
- Type of transferred packets
- Selection of the local interface of the selected security module that will serve as the tunnel endpoint
- Selection of the partner interface that will serve as the tunnel endpoint



Dead peer detection (DPD)

As default, DPD is enabled.

When DPD is enabled, the modules exchange additional messages at selectable intervals if no communication occurs at these points in time. This means that it is possible to recognize whether the IPsec connection is still valid or possibly needs to be re-established. If there is no longer a connection, the security associations (SA) of phase 2 are terminated prematurely. If DPD is disabled, the SA is ended only after the SA lifetime has expired. For information on setting the SA lifetime, see section
Configuring VPN group properties (Page 808).

Permission to initiate connection establishment

You can restrict the permission for initiating the VPN connection establishment to certain modules in the VPN.

The decisive factor for the setting of the parameter described is the assignment of the address for the gateway of the module you are configuring. If a static IP address is assigned, the module can be found by the partner. If the IP address is assigned dynamically and therefore changes constantly, the partner cannot establish a connection as things stand.

Mode	Meaning
Start connection to partner (initiator/responder) (default)	If this option is selected, the module is "active", in other words, it attempts to establish a connection to the partner with a fixed IP address. The reception of requests for VPN connection establishment is also possible. This option is recommended when you obtain a dynamic IP address from the provider for the gateway of the security module you are configuring. The partner is addressed using its configured WAN IP address, its configured external module IP address or the configured DNS name.
Wait for partner (responder)	If this option is selected, the module is "passive", in other words, it waits for the partner to initiate the connection. This option is recommended when you have been assigned a static IP address by the provider for the gateway of the security module you are configuring. It means attempts to establish a connection can only be initiated by the partner. This partner can, for example, have a dynamic WAN IP address.

Note

Make sure that you do not set all the modules in a VPN group to "Wait for connection from remote VPN gateway" otherwise no connection is established.

WAN IP address / FQDN - addresses of the modules and gateways in a VPN over Internet

When operating a VPN with IPsec tunnel over the Internet, additional IP addresses are generally required for the Internet gateways such as DSL routers. The individual security or SCALANCE M modules must know the external IP addresses of the partner modules in the VPN.

Note

To use a WAN as an external public network, enter the IP address that you received from the provider as the external IP address, through which the security module is then reachable in the WAN (Internet). To allow the security module to send packets via the WAN, you need to enter your DSL router as "Standard router".

If you use a DSL router as Internet gateway, the following ports (at least) must be opened on it as described in the relevant documentation:

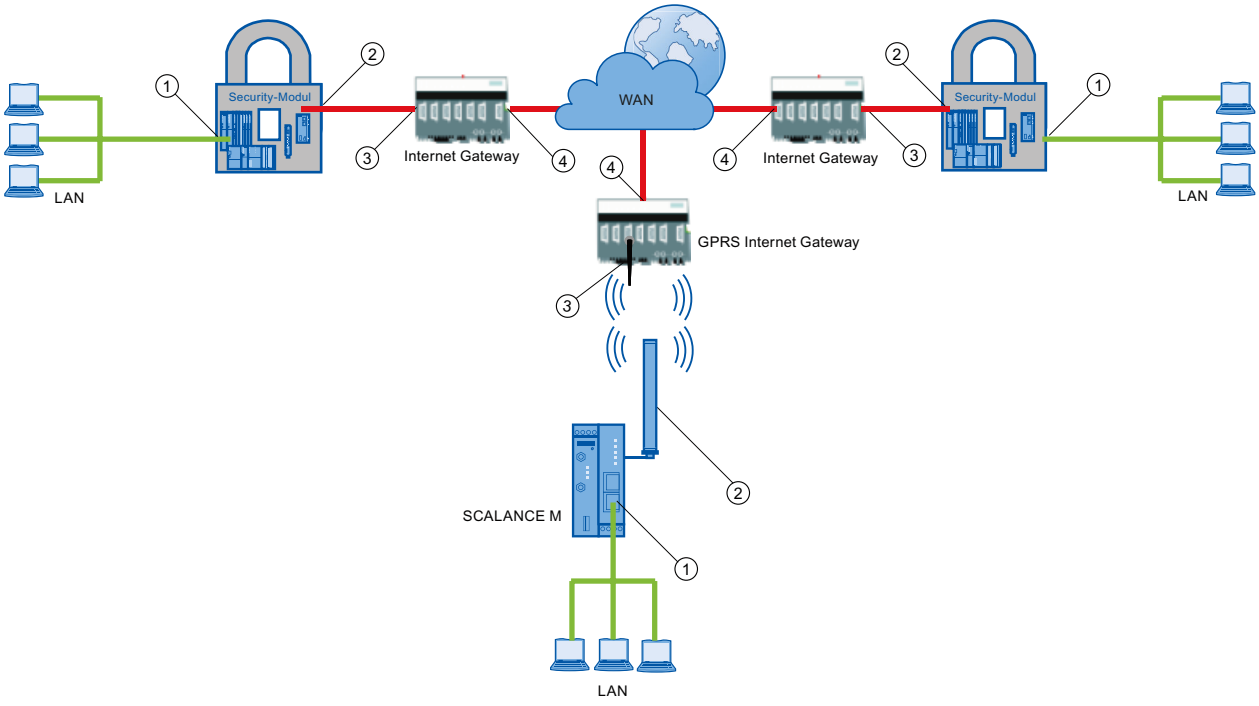
- Port 500 (ISAKMP)
- Port 4500 (NAT-T)

To allow this, in the module-specific VPN settings, you have the option of assigning an IP address as a "WAN IP address". When you download the module configuration, the group members are then informed of the WAN IP addresses of the partner modules. As an alternative to a WAN IP address, you can also enter an FQDN. Depending on the existing addresses, VPN endpoints are used as default according to the following priorities:

1. WAN address
2. FQDN of the primary dynamic DNS service
3. FQDN of the secondary dynamic DNS service
4. External IP address / DMZ IP address of the security module

Note: After removing an existing WAN address, the external IP address / DMZ IP address is always used.

In the interface selection of the connection-specific VPN settings, you yourself can decide which address the partner will be informed of. In these settings, you can also specify the interface to be used for communication by the nodes of a VPN group and the security module that is authorized to set up connections.



① Internal IP address - of a module
② External IP address - of a module
③ IP address of an Internet gateway (for example, GPRS gateway)
④ IP address (WAN IP address) of an Internet gateway (for example, DSL router)

Responder settings for Road Warrior connections

The configuration of responder setting for Road Warrior connections is only possible in V14 SP1 projects. For information on upgrading projects, refer to section Create users (Page 733).

The VPN connection establishment by Unknown Peers (Road Warriors) is allowed by the responder as default with the secure VPN properties of VPN profile 1. You can allow or block

the VPN connection establishment by Unknown Peers and the properties used by selecting a VPN group profile and by selecting VPN profiles.

Operator control element	Meaning
Allow VPN connection establishment by Road Warriors	The responder allows VPN connection establishment by Unknown Peers.
VPN profile 1 - 5 in the selection dialog for VPN profiles	The responder allows VPN connection establishment by unknown peers with the properties of the predefined VPN profiles 1 - 5. The properties of these VPN profiles correspond to the properties of the fallback VPN profiles that you can select for VPN groups, see section Configuring VPN group properties (Page 808).
VPN group profiles in the selection dialog for VPN profiles	The responder allows VPN connection establishment by Unknown Peers with the properties of the VPN group selected in the selection dialog for VPN profiles with a check mark. The VPN groups can be selected in which the responder is located. A maximum of one VPN group can be selected and used as a VPN group profile. With the "Add" button, new VPN groups can be created to which the responder is automatically assigned. These VPN groups can then be selected for the formation of a VPN group profile. If VPN connection establishment by Unknown Peers with the properties of the VPN group in which the responder and the Unknown Peers are located is to be allowed, this VPN group must be selected in the selection dialog.

After opening a STEP 7 project last saved with a version older than V14 SP1, the check box "Allow VPN connection establishment by Road Warriors" is activated s default. In addition to this, the responder is assigned the predefined VPN profiles 2 - 5.

For VPN connections from SOFTNET Security Client / SCALANCE M-800 modules to SCALANCE S / CPs the selected VPN profile with the highest priority is always used and read out. The authentication method of the selected VPN profile must match the authentication method of the corresponding VPN group. The priority of the VPN profiles is as follows:

1 Selected VPN group profile
2. VPN profile 1
3. VPN profile 2
4. VPN profile 3
5. VPN profile 4
6. VPN profile 5

For VPN connections from SCALANCE M875 modules to SCALANCE S / CPs, VPN profile 3 must be selected if the authentication method of the corresponding VPN group is "Certificate". If the authentication method of the corresponding VPN group is "Preshared key", VPN profile 5 must be selected.

Configuring internal network nodes

The configuration of internal network nodes is described in the following section:
AUTOHOTSPOT

See also

How to create an IPsec tunnel with VPN groups (Page 803)

Configuring router and firewall redundancy

Module-specific function

This function is available only for SCALANCE S623/S627-2M as of V4, see section AUTOHOTSPOT.

Online functions - diagnostics and logging

Overview of diagnostics and logging in

For test and monitoring purposes, the security module has diagnostic and logging functions.

- Diagnostic functions
  These include various system and status functions that you can use for an existing network connection to the security module.
- Logging functions
  This involves the logging of system and security events and data packets.

Recording events with logging functions

You select the events to be logged in the log settings for the relevant security module.

You can configure the following variants for logging:

- Local logging
  In this variant, you log events in the local buffer of the security module. You can then access these logs, display them and archive them on the service station in the "Online & diagnostics" dialog. The evaluation of the buffer areas of the security module is only possible if there is a network connection to the selected security module.
- Network Syslog
  With Network Syslog, you use a Syslog server that exists in the network. This logs the events according to the configuration in the log settings for the relevant security module.

Archiving log data and reading in from a file

You can save the logged events for archiving in a log file and open this later even without a network connection to the security module. For more detailed information, refer to section AUTOHOTSPOT.

Diagnostics in ghost mode (only for SCALANCE S S602 as of V3.1)

When you operate the security module in ghost mode, the external interface of the security module takes over the IP address of the internal node at runtime. Before you can run diagnostics of a security module in ghost mode, you need to establish the connection to the security module via the IP address that the security module obtained from the internal node at runtime.

To find out which IP address the security module currently has, you can search for accessible nodes in STEP 7 using the "Online" > "Accessible devices".

Protecting exported log files from unauthorized access

Log files exported from STEP 7 can contain security related information. You should therefore make sure that these files are protected from unauthorized access. This is particularly important when passing on the files.

Function overview online dialog

Functions of the "Online & diagnostics" dialog

In STEP 7, the security module provides the following functions in the "Online & diagnostics" dialog:

Entry in the online dialog	Function
Status	Display of status information about the selected security module, such as the current IP addresses of the interfaces and the current time and date.
Interface settings	Overview of the settings of the individual interfaces.
Dynamic DNS	Overview of the settings for dynamic DNS.
System log	Display of logged system events, starting and stopping logging (only if there is an online connection to SCALANCE S modules) and starting and stopping the reading out of log data from the local buffer of the security module.
Audit log	Display of logged security events, starting and stopping reading out of log data from the local buffer of the security module.
Packet filter log	Display of logged data packets, starting and stopping logging (only if there is an online connection to SCALANCE S modules) and starting and stopping the reading out of log data from the local buffer of the security module.
ARP table	Display of the ARP table of the security module.
Logged-on users	Shows the users logged in to the Internet page for user-specific IP rule sets.
Communications status	Display of status information about VPN tunnel connections and members of VPN groups to which the selected security module belongs.
Internal nodes	Display of the learned or configured internal network nodes of the security module.
Dynamically updated firewall rules	Display of the IP addresses that were released dynamically over HTTP or HTTPS, or loaded by a user.
Firewall blacklist	Display of the IP addresses that were entered in the blacklist of the firewall.
Ghost mode	Dialog for the ghost mode of the SCALANCE S602 with information on the address parameters of the internal node (identical to the external IP address of the security module) and on IP address changes of the internal node.
Log files (offline view)	
• System log	Display of logged system events as well as starting and stopping the display.
• Audit log	Display of logged security events as well as starting and stopping the display.
• Packet filter log	Display of logged data packets as well as starting and stopping the display.
Date and time of day	Date and time setting.
Firmware update	Updating the firmware.

Requirements for access

The following requirements must be met to use the online functions of a security module:

- There is a network connection to the selected module
- The project with which the module was configured is open
- a user with the required rights must be logged in to the project
- For CPs the diagnostics access must be opened in the firewall (CP 343-1 Adv. / 443-1 Adv. and CP 1628: TCP 443; S7-1200/S7-1500 CPs: TCP 8448)

Note

Requirement for online diagnostics in ghost mode (only for SCALANCE S S602 as of V3.1)

Online diagnostics is only available in ghost mode if the security module has learned the IP address of the internal node and has adopted this for its external interface. After this, the security module can be reached via the IP address of the external interface.

How to access this function

1. Right-click on the module to process.
2. Select the "Online & diagnostics" command from the shortcut menu.
3. If there is no online connection established to the security module, click the "Connect online" button in the "Diagnostics" entry.

Online settings are not saved in the configuration

Settings that you make in online mode (for example settings for the logging memory) are not stored in the configuration on the security module. This is why the configuration settings are always applied at the restart of the module.

Functions of the online diagnostics

Status information of the security module - "Status" entry

Meaning

Display of the status of the security module selected in the project.

Table 10-46 Online & diagnostics: - "Status" entry

System and status functions	Meaning
Overview	
Hardware type	The type of the security module.

System and status functions	Meaning
External IP address	The external IP address of the security module. For S7-1200/S7-1500 CPs, CP 1628: The IP address of the Industrial Ethernet interface. For CP 343-1 Adv. / 443-1 Adv.: The IP address of the Gbit interface.
Internal IP address	The internal IP address of the security module. For CP 1628: The IP address of the NDIS interface. If there is more than one NDIS address, only one is displayed. For CP 343-1 Adv. / 443-1 Adv.: The IP address of the PROFINET interface.
DMZ IP address (SCALANCE S623 / S627-2M only)	The DMZ IP address of the security module.
Tunnel IP address (only SCALANCE S612/S623/ S627-2M as of V4)	The first alias internal IP address of the security module in the VPN tunnel.
Serial number	The serial number of the security module.
Article number	The article number of the security module that is used when ordering.
Firmware version	The firmware version of the security module.
Operating mode	Mode of the security module (bridge mode / routing mode)
External MAC address	The external MAC address of the security module. For S7-1200/S7-1500 CPs, CP 1628: The MAC address of the Industrial Ethernet interface. For CP 343-1 Adv. / 443-1 Adv.: The MAC address of the gigabit interface.
Internal MAC address	The internal MAC address of the security module. For CP 1628: The MAC address of the NDIS interface. For CP 343-1 Adv. / 443-1 Adv.: The MAC address of the PROFINET interface.
DMZ MAC address (SCALANCE S623 / S627-2M only)	The DMZ MAC address of the security module
Hardware release	The hardware product version of the security module.
C-PLUG	Shows whether or not a C-PLUG is inserted.
Alias IP addresses (only for SCALANCE S as of V4)	
IP address	Alias IP address that was registered on an interface of the security module by a NAT rule.
Corresponding Interface	Interface of the security module on which the alias IP address was registered.
Local time	

System and status functions	Meaning
Current time	Date and time that is displayed on the security module. Format for "German" user interface language: dd.mm.yyyy (date) hh:mm:ss (time) Format for "English" user interface language: mm/dd/yyyy (date) hh:mm:ss AM/PM (time) Format for "French", "Italian" and "Spanish" user interface language: dd/mm/yyyy (date) hh:mm:ss (time) Format for "Chinese" user interface language: yyyy/mm/dd (date) hh:mm:ss **Note (not for CPs)** You set the local time on the SCALANCE S module in "Functions" > "Date and time".
Operating time	Time since the last restart of the security module. Format with the user interface language "German", "English", "French", "Italian", "Spanish" and "Chinese": dddd.hh:mm:ss
Time-of-day source	The source from which the date and time are obtained.
Configuration	
Created	Date and time when the project was first created. Format with the user interface language "German", "English", "French", "Italian", "Spanish" and "Chinese": dd.mm.yyyy (date) hh:mm:ss (time)
Name	File name of the project last loaded.
Author	Name of the user who created the project. Is adopted from the project properties.
Compiled	Date and time when the project was last compiled. Format with the user interface language "German", "English", "French", "Italian", "Spanish" and "Chinese": dd.mm.yyyy (date) hh:mm:ss (time)
Storage location	Specifies the location (e.g. town) that was entered in the properties of an SCT project.
File system (not for CPs)	
RAM	Indicates how much RAM and flash is occupied in the file system.
Flash	

Overview of the individual interfaces - "Interface settings" entry

Module-specific function

This function is available only for SCALANCE S V3 modules or higher, see section Overview of the individual interfaces - "Interface settings" entry (Page 885)

Overview of the Dyn. DNS settings - "Dynamic DNS" entry

Module-specific function

This function is available only for SCALANCE S V3 modules or higher, see section Overview of the Dyn. DNS settings - "Dynamic DNS" entry (Page 886)

Display of the ARP table - "ARP table" entry

Module-specific function

This function is available only for SCALANCE S V3 modules or higher, see section Display of the ARP table - "ARP table" entry (Page 888)

Users logged in to the Web page - "Logged in users" entry

Module-specific function

This function is available only for SCALANCE S V3 modules or higher, see section Users logged in to the Web page - "Logged in users" entry (Page 888)

VPN connections of the security module - "Communication status" entry

Meaning

Display of the communication status of the following network components:

- Other security modules of the VPN group to which the selected security module belongs
- Internal network nodes of these security modules

Table 10-47 Online & diagnostics: "Communication Status" entry

System and status functions	Meaning
Known security devices or modules	Display of the nodes with which the selected security module is in a VPN group. This also shows whether the tunnel status is active or passive. To obtain additional information on one of the nodes, select this in the list. **Note:** Configured, inactive tunnels are indicated for CPs only.
Endpoints	Display of information on the internal network nodes of the security module that you selected in the "Known security devices or modules" table. For each internal network node, this shows whether it was learned or configured. It also shows the subnet in which the internal network node is located. With SCALANCE S modules, the subnet of network nodes is only displayed in bridge mode.
Tunnel properties	Display of properties of the VPN tunnel established to the security module that you selected in the "Known security devices or modules" table.

Found internal network nodes - "Internal nodes" entry

Meaning

Display of all learned and configured network nodes. This also displays whether or not the learning mode of the security module is enabled.

Updated firewall rules - "Dynamically updated firewall rules" entry

Module-specific function

This function is only available for CP 343-1 Adv. / 443-1 Adv., see section: AUTOHOTSPOTin the section "Security for S7-300 / S7-400 / PC CPs".

See also

Updated firewall rules - "Dynamically updated firewall rules" entry (Page 903)

Display of the firewall blacklist - "Firewall blacklist" entry

Module-specific function

This function is available only for SCALANCE S V4 modules or higher, see section Display of the firewall blacklist - "Firewall blacklist" entry (Page 888).

Setting the date and time - "Date and Time" entry

Module-specific function

This function is available only for SCALANCE S, see section Setting the date and time - "Date and time" entry (Page 889).

Diagnostics in ghost mode - "Ghost mode" entry

Module-specific function

This function is available only for SCALANCE S602 as of V3.1, see section Diagnostics in ghost mode - "Ghost mode" entry (Page 889).

Logging functions

Logging system events - "System log" entry

Meaning

Display of logged system events and starting and stopping reading system events from the local memory of the security module.

The system log automatically logs successive system events, for example the start of a process. The logging can be scaled based on event classes.

System and status functions	Meaning
Start/stop logging (not for CPs)	Starts/stops recording of system events. The method and the event classes that are logged are configured in the local security settings.
Start/stop reading	Starts/stops reading of system events from the local memory of the security module. If you select the "Save log file" check box, the recorded log data is also saved as file. Select the storage location and enter a file name. **Note** If you select the "Save log file" check box after starting reading, the data read out up to then can no longer be saved in a log file.
Clear display	Deletes the log data shown in the table.

For more information on opening saved system events in log files, refer to section Evaluating system events in offline mode - "System log" entry (offline view) (Page 834).

Logging security events - "Audit log" entry

Meaning

Display of logged security events and starting and stopping reading of security events from the local memory of the security module.

The audit log automatically logs successive security-relevant events. This includes, for example, user actions such as enabling and disabling packet logging.

System and status functions	Meaning
Start/stop reading	Starts/stops reading of security events from the local memory of the security module. If you select the "Save log file" check box, the recorded log data is also saved as file. Select the storage location and enter a file name. **Note** If you select the "Save log file" check box after starting reading, the data read out up to then can no longer be saved in a log file.
Clear display	Deletes the log data shown in the table.

For more information on opening security events stored in log files, refer to section Evaluating security events in offline mode - "Audit log" entry (offline view) (Page 835).

Logging data packets - "Packet filter log" entry

Meaning

Display of logged data packets and starting and stopping reading of packet filter events.

The packet filter log records certain packets of the data traffic. Data packets are only logged if they match a configured packet filter rule (firewall) or to which the basic protection reacts (corrupt or invalid packets). This is only possible when logging is enabled for the packet filter rule.

For information about activation of the logging, refer to section AUTOHOTSPOT.

As well as reading the log data from the buffer and transferring it to the display, it can also be saved in a file for archiving.

System and status functions	Meaning
Start/stop logging (not for CPs)	Starts/stops logging of data packets. The method with which the data is logged is configured in the local security settings.
Start/stop reading	Starts/stops reading out of logged data packets from the local memory of the security module. If you select the "Save log file" check box, the recorded log data is also saved as file. Select the storage location and enter a file name. **Note** If you select the "Save log file" check box after starting reading, the data read out up to then can no longer be saved in a log file.
Clear display	Deletes the log data shown in the table.
Log category	Select the data packets for which the logging will be displayed. The selection depends on the settings you configured offline in the local security settings. Only the data packets for which logging was enabled are logged. If you select a category for which logging was not enabled, no data will be logged for this category.

For information on opening the stored packet filter log data, refer to section Evaluating packet filter events in offline mode - "Packet filter log" entry (offline view) (Page 835).

Evaluating log files in offline mode

Evaluating system events in offline mode - "System log" entry (offline view)

How to access this function

1. Right-click on the module to process.
2. Select the "Online & diagnostics" command from the shortcut menu.
3. Select "Diagnostics" > "Log files (offline view)" > "System log".

Meaning

Opens logged system events that you saved as a file in the online view.

For more information, refer to chapter Logging system events - "System log" entry (Page 832).

Evaluating security events in offline mode - "Audit log" entry (offline view)

How to access this function

1. Right-click on the module to process.
2. Select the "Online & diagnostics" command from the shortcut menu.
3. Select "Diagnostics" > "Log files (offline view)" > "Audit log".

Meaning

Opens logged security events that you saved as a file in the online view.

For more detailed information, refer to section Logging security events - "Audit log" entry (Page 833).

Evaluating packet filter events in offline mode - "Packet filter log" entry (offline view)

How to access this function

1. Right-click on the module to process.
2. Select the "Online & diagnostics" command from the shortcut menu.
3. Select "Diagnostics" > "Log files (offline view)" > "Packet filter log".

Meaning

Opens logged data packets that you saved as a file in the online view.

For more detailed information, refer to section Logging data packets - "Packet filter log" entry (Page 833).

Download functions

Special features when downloading security configurations

Security configurations can influence the reachability of the security module for the configuration PC. This, for example, is the case if there is a tunnel connection configured for one security module to another security module in a configuration and this configuration is downloaded from the configuration PC to the security module. After the download by the configuration PC, the security module can no longer be reached and the reachability test performed as default by STEP 7 following the download of the configuration fails. The error message output by STEP 7 relates solely to the reachability test; the actual download of the configuration is ensured if the project data is consistent and the IP address relationship between the security module and the configuration PC is correct.

Special features when downloading configurations and firmware to SCALANCE S modules are described in the following subsection of the section "SCALANCE S": AUTOHOTSPOT

Uploading configurations to the engineering station

Uploading a configuration from a SCALANCE S module or from the CP 1628 to an engineering station is not possible.

Uploading a configuration from an S7 CP that supports security to an engineering station is also possible if security features were included in the configuration. Configured security functions are not, however, transferred to the engineering station. In the transferred configuration on the engineering station, the "Activate security features" check box is also deselected.

SOFTNET Security Client

Using the SOFTNET Security Client

Area of application - access over VPN

With the SOFTNET Security Client (SSC) PC software, secure remote access is possible from PGs/PCs to automation systems protected by a security module via public networks. You require SOFTNET Security Client V4.0 HF1 for S7-300/S7-400 CPs and for the PC CP 1628. These CPs are not approved for operation with SOFTNET Security Client ≤ V4.0. The SOFTNET Security Client has not been released for other CPs.

With the SOFTNET Security Client, a PG/PC is configured automatically so that it can establish a secure IPsec tunnel communication in the VPN (Virtual Private Network) with one or more security modules.

This IPsec tunnel communication makes it possible for PG/PC applications such as NCM diagnostics to securely access devices or networks that are located in an internal network protected by the security module.

How does the SOFTNET Security Client work?

The SOFTNET Security Client reads the configuration that was created by STEP 7 and, if necessary, determines from the file which certificates are to be imported.

The CA certificate and the private key are imported when applicable and stored on the local PG/PC.

Following this, security settings are made based on the data from the configuration so that applications can access IP addresses downstream from the security modules.

If the learning mode is enabled for the internal nodes or programmable controllers, a security policy is first set for secure access to security modules. Then, the SOFTNET Security Client addresses the security modules in order to obtain the IP addresses of the relevant internal nodes.

The SOFTNET Security Client enters these IP addresses in special filter lists belonging to this security policy. Applications can then communicate with the programmable controllers via VPN.

Note

Additional security measures when using the SOFTNET Security Client

The SOFTNET Security Client provides a solution for secure communication with automation cells via VPN. For self-protection of the PC/PG and the corresponding automation cell, it is advisable to use additional measures such as a virus scanner and the Windows firewall.

In Windows 7, the firewall of the operating system must be enabled so that VPN tunnel establishment works.

Creating a configuration file in STEP 7

Configuring a SOFTNET Security Client module in the project

The SOFTNET security client is created as a module in the project. In contrast to the other security modules, you do not need to configure any further properties.

Assign the SSC module to the VPN group or groups at which an IPsec tunnel to the PG/PC is to be set up.

Note

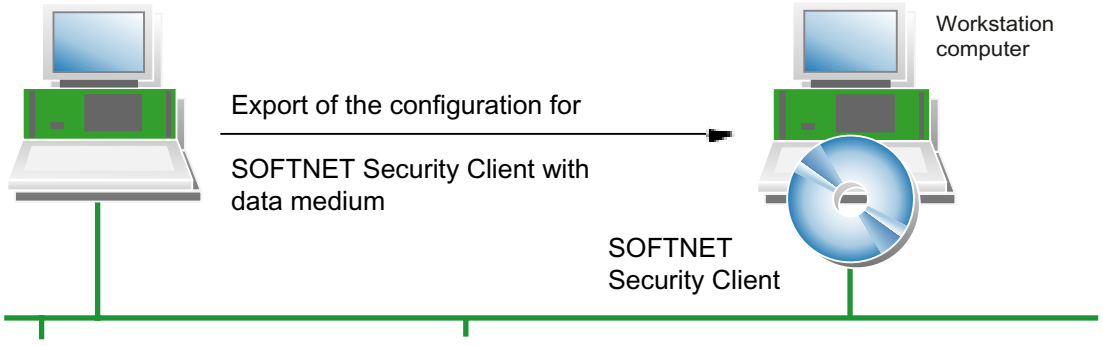
Refer to the information on parameters in section
Including security module in configured VPN group (Page 816).

Note

If you create several SOFTNET Security Clients within a group, no tunnels are set up between these clients but only from the relevant client to the security modules.

Configuration files for the SOFTNET Security Client

The interface between STEP 7 as configuration tool and the SOFTNET Security Client is controlled by configuration files.



The configuration is stored in the file types "*.dat", "*.p12" and "*.cer".

Procedure

To generate the configuration files, perform the following steps in STEP 7:

1. In the "Devices & networks" view, select the "Topology view" or the "Network view" tab.
2. Insert a PC system of the type "SOFTNET Security Client" in the selected tab from the hardware catalog.
3. Assign the SOFTNET Security Client to the VPN groups in which the PG/PC will communicate over IPsec tunnels.
4. Make sure that the "Generate SSC files" check box is selected under the "Configuration of the SOFTNET Security Client" entry in the local security settings of the SOFTNET Security Client.
5. Select the storage location for the configuration files.
6. Compile the configuration of the SOFTNET Security client to export the configuration file.
7. If you selected certificate as the authentication method, specify a password for the certificate of the VPN configuration. If you do not assign a password, the project name (not the project password) is used as the password.
   Result: Export of the configuration files is completed.
8. Adopt the files of the type *.dat, *.p12, *.cer on the PG/PC on which you want to operate the SOFTNET Security Client.

Note

Protecting exported configuration files from unauthorized access

Configuration files for SOFTNET Security Client exported from STEP 7 can contain security related information. You should therefore make sure that these files are protected from unauthorized access. This is particularly important when passing on the files.

SCALANCE S

Replacing a security module

Module-specific function

This function is available only for SCALANCE S V3 modules or higher.

Requirement

To be able to replace security modules, their module descriptions must be up to date. To update the module description of security modules, follow the steps below:

1. Select the security module to be edited.
2. In the local security settings, click on the entry "General" > "Catalog information".
3. Click the "Update module description" button.

How to access this function

1. Select the security module to be edited in the topology or network view.
2. Right-click on the security module and select the shortcut menu command "Change device...".
   Based on the following table, you can see which security modules you can replace without data loss and which could involve a possible data loss.

Initial module	Possible module replacement									
	S602 V3	S602 V3.1	S602 V4	S612 V3	S612 V4	S623 V3	S623 V4	S623 V4.0.1	S627-2 M V4	S627-2 M V4.0.1
S602 V3	-	x	x	x	x	x	x	x	x	x
S602 V3.1	!	-	x	!	!	!	!	!	!	!
S602 V4	!	!	-	!	!	!	!	!	!	!
S612 V3	!	!	!	-	x	x	x	x	x	x
S612 V4	!	!	!	!	-	!	x	x	x	x
S623 V3	!	!	!	!	!	-	x	x	x	x
S623 V4	!	!	!	!	!	!	-	x	x	x
S623 V4.0.1	!	!	!	!	!	!	!	-	!	x
S627-2M V4	!	!	!	!	!	!	!	!	-	x
S627-2M V4.0.1	!	!	!	!	!	!	!	!	!	-

x Without losses

! Possibly with losses

- The module type and the firmware version are not changed.

Configuring interfaces for SCALANCE S modules

Overview

Configuring the mode

With the mode you specify how interface routing is handled (external/internal). The DMZ interface of the security module (SCALANCE S623/S627-2M only) is always connected in routing mode. For more detailed information, refer to the section
Configuring IP address parameters (Page 842)

Configuring the interfaces

If the external interface, the DMZ interface (SCALANCE S623/S627-2M only) or the tunnel interface (only SCALANCE S612/S623/S627-2M as of V4 in VPN group(s)) of a security module needs to be configured, it must be activated using the "Activate interface" check box. Set the IP address information for each interface and settings for the individual ports. The following options are available with which you can assign an IP address for the external interface and for the DMZ interface in the "General" entry (SCALANCE S623/S627-2M only):

- Static IP address with subnet mask. For more detailed information, refer to the section Configuring IP address parameters (Page 842)
- Address assignment using PPPoE. For more detailed information, refer to the section Configuring an Internet connection (Page 844)
  The internal interface and the tunnel interface can only be configured using a static IP address.

If alias IP addresses were registered on the interfaces of the security module due to configuring NAT rules, these are displayed in the "Alias IP addresses" entry.

Note

External interface and DMZ interface as Internet access

The simultaneous operation of PPPoE on the external interface and on the DMZ interface (dual ISP) is not possible.

Point to Point Protocol over Ethernet (PPPoE)

To allow Internet/WAN access directly via a DSL modem, the IP address on the external interface or on the DMZ interface is assigned using PPPoE. PPPoE is a dial-in protocol for obtaining IP addresses from an Internet service provider (ISP). SCALANCE S is operated here in routing mode.

To use this IP address assignment mode, specify the ISP in the "PPPoE" entry. The IP address, the subnet mask, the standard router and the DNS server of the interface are specified by the ISP.

Note

A configured standard router is not taken into account when using PPPoE. This is assigned dynamically to the module by the ISP.

Note

No network components between SCALANCE S and DSL modem

If the interface of a SCALANCE S module is operated using PPPoE, there must be no other network components between this interface and the connected DSL modem, otherwise the dial-in data of the Internet Service Provider may be transferred unencrypted over this link. When using the "CHAP" authentication protocol, the data is transferred encrypted.

Configuring media modules

In addition to the functions of the SCALANCE S623, the S627-2M has two media module slots in which an electrical or optical media module with two ports can be inserted. This expands both the external and internal interface by up to two ports. In routing mode, the additional ports of the security module can be used to link the external and internal interface to ring topologies.

To integrate media modules in the SCALANCE S627-2M, select the security module and change to the device view. Then select the required media modules from the hardware catalog.

For ports with the port type "Copper", you can set the transmission speed and the duplex method manually using the port mode. For ports with the port type "Optical", the port mode is fixed by the media module used or by the SFP transceiver used and cannot be adapted.

You will find information on connecting the media module ports to MRP rings in the following section:
AUTOHOTSPOT

Setting the operating mode

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "Mode".

Operating mode - possible selections

You can change the operating mode in this dialog if the security module is not included in a VPN group. If the security module is in a VPN group, the mode cannot be changed.

The selection applies to interface routing between the external and internal interface. The DMZ interface (SCALANCE S623 and S627-2M only) is always connected in routing mode.

Bridge mode	For operation in flat networks. External and internal interface are in the same IP subnet. For S623 / S627-2M: External and internal interface are in the same IP subnet, the DMZ interface is in a different IP subnet or is deactivated.
Routing mode	All interfaces are in different IP subnets. If you have activated the routing mode, you must configure an internal IP address and subnet mask for the internal interface of the security module. **Note** If you have enabled the routing mode for the SCALANCE S module, no MAC firewall rules can be defined.
Ghost mode (only for SCALANCE S S602 as of V3.1)	In operation, the security module adopts the IP address of the node connected to the internal interface of the security module for the external interface. The IP address data specified for the external interface is only used for downloading the configuration prior to operation in ghost mode.

Configuring IP address parameters

Meaning

Specify network parameters such as the IP address and subnet mask for the interface(s) of the security module.

How to access this function

1. Select the module to be edited.
2. Select "External interface [P1] red" or "DMZ interface [P3] yellow" in the local security settings. The virtual tunnel interface can only be configured when the security module (only SCALANCE S612/S623/S627-2M as of V4) is in at least one VPN group, refer to the section "Meaning of the tunnel IP address".

   Note

   Configuration of the internal interface in routing mode

   If you have selected the "Routing" mode for the security module, you must also configure an internal IP address and subnet mask for the internal interface of the security module. You can access this function in the local security settings under "Internal interface [P2] green" > Ethernet addresses".
3. If applicable, enable the interface using the "Activate interface" check box.
4. Select the "Ethernet addresses" entry.
5. Complete the settings specified in the following table.

Parameter	Meaning
IP address	IP address for the external interface. The IP address consists of four decimal numbers from 0 to 255, with each number being separated by a period, for example, 141.80.0.16.
Subnet mask	The subnet mask consists of four decimal numbers that are separated by period, for example, 255.255.0.0
Use router (not possible for the tunnel interface)	Select this check box if you want to use a standard router and enter its IP address in the "Router address" input box.

Note

Networking the physical interfaces

Network the physical interfaces of the security module with suitable subnets to avoid IP address conflicts.

Meaning of the tunnel IP address

If you use the function "NAT/NAPT in the VPN tunnel" for a SCALANCE S612/S623/S627-2M module as of V4, you need to assign a tunnel IP address for the security module. This ensures the reachability of the security module via the VPN tunnel and provides a configuration and diagnostics option. The configured tunnel IP address can be expanded with alias tunnel IP addresses using suitable NAT / NAPT rules. The subnet mask is fixed at 32 bits for the tunnel IP address and cannot be changed by the configuration. The tunnel IP address can only be configured if the security module is in at least one VPN group.

You will find further information on address translation with NAT/NAPT in VPN tunnels in the following section:
Address translation with NAT/NAPT in VPN tunnels (Page 870)

Special features with a standard router

- If the IP assignment configured is via "PPPoE", a configured standard router is ignored because the standard route is always automatically via the PPPoE interface.
- If the address assignment configured is with "Static address" and if the security module is connected to the Internet via a DSL (NAPT) router, the DSL router must be entered as the standard router.
- For security modules in ghost mode (SCALANCE S602 as of V3.1 only), no standard routers can be configured since these are identified during runtime. Specific routes cannot be configured for security modules in ghost mode.

Configuring port mode

Meaning

The port mode specifies the transmission speed and the duplex method. The same parameters should be set for the ports involved in communication.

Configurable port modes

For SCALANCE S V3 and higher, the following port modes can be configured for the permanently installed ports:

Port mode	Meaning
Autonegotiation	The transmission speed and the duplex setting are selected automatically. **Note** A transmission speed of 1000 Mbps and the autocrossing function are supported only if autonegotiation is selected.
10 Mbps, half and full duplex	Transmission speed of 10 Mbps
100 Mbps, half and full duplex	Transmission speed of 100 Mbps

Deactivating a port is possible only for external ports and for the DMZ port of SCALANCE S623/ S627-2M. The port modes for media module ports are configured in the device view and are based on the range of functions of the media module in question.

Configuring an Internet connection

Requirement

The "PPPoE" entry is only displayed in the local security settings if the IP assignment method "PPPoE" is configured for one of the interfaces.

How to access this function

1. Select the security module to be edited
2. In the local security settings, select the entry "PPPoE".

Meaning

In this entry, you make settings related to the Internet Service Provider (ISP) if a connection using PPPoE is set for one of the interfaces of the security module.

Table 10-48 Settings for the ISP account

Function	Description
Authentication protocol	Select none or one of the following authentication protocols: • PAP (Password Authentication Protocol) • CHAP (Challenge Handshake Authentication Protocol) **Note** Both communications partners have to use the same authentication method otherwise no connection can be established.
User name	Enter the name for logging in with the ISP account.

Function	Description
Password	Enter the password for logging in with the ISP account.
Repeat password	Enter the password for logging in with the ISP account again.

Table 10-49 Rules for user names and passwords

Permitted characters	The following characters from the ANSI X 3.4-1986 character set are permitted: 0123456789 A...Z a...z !#$%&()"*'+`,-./:;<=>?@ [\]_{|}~^
Length of the user name	1 to 255 characters
Length of the password	1 to 31 characters

Table 10-50 Settings for the connection

Function	Description
Permanent connection	Permanent Internet connection. After the connection has been terminated by the provider, the connection is automatically restored even if there are currently no packets to be sent.
On-demand connection	The Internet connection is established automatically if packets need to be sent to the Internet. In this setting, delays in the sending of packets are possible.
Maximum idle time (only with the setting "on-demand connection")	If no packets are sent during a certain time, the Internet connection is automatically terminated. In the "Maximum idle time" box, enter the time in seconds after which the connection will be terminated. • Default setting: 300 • Minimum value: 10 • Maximum value: 3600
Forced disconnection (only with the "Permanent connection" setting)	Select the check box to be able to adapt the time of the forced disconnection by the security module.
Forced disconnection time (only with the "Permanent connection" setting)	The provider terminates the Internet connection automatically after a certain period. If you enter a time of day in this box, the security module terminates the Internet connection itself at this time. This allows disconnection of the Internet connection by the provider to be delayed under certain circumstances. A self-initiated forced disconnection is only possible with an existing permanent connection. • Default setting: 00.00 • Permitted entries: 00:00 ... 23:59

Configuring DNS

Module-specific function

DNS servers for resolving FQDNs specified in the entry "Dynamic DNS" can be used by SCALANCE S modules as of V3. DNS servers for all other functions can be used by SCALANCE S modules as of V4.

DNS

To address network notes using FQDNs the security module must know the IP address of a DNS server that converts the DNS queries to the corresponding IP addresses of the network nodes.

Setting up DNS - Follow the steps below:

1. Select the security module to be edited.
2. In the local security settings, select the entry "DNS".

Option	Meaning
Obtain DNS server address automatically	The address of the DNS server can be obtained automatically using PPPoE if the security module is connected to the Internet via a DSL modem. Can only be set for the external interface and the DMZ interface.
Use the following DNS server address:	Enter the address of the preferred and of the alternative DNS server manually.

Configuring dynamic DNS

Dynamic DNS

With dynamic DNS, you can access a constantly changing IP address with a permanently defined name (FQDN). This is necessary, for example, if you want to access a server that can be reached via a public IP address that changes.

The security module signals the current WAN IP address via which the security module can be reached to a provider for dynamic DNS (for example DynDNS.org, no-ip.com). The provider ensures that DNS queries sent to the FQDN of the security module are replied to with the current WAN IP address of the security module.

Dynamic DNS is permitted on the following interfaces:

- External interface
- DMZ interface

Requirement

An account has been created with a provides of dynamic DNS and an FQDN has been registered.

Setting up dynamic DNS - Follow the steps below:

1. Select the security module to be edited.
2. Select the "Dynamic DNS" entry in the local security settings.

3. Activate the "Activate service" check box in the "Primary dynamic DNS service" area and make the following settings:

Setting	Meaning
Provider	Choose the provider with which you have set up an account for dynamic DNS. With the predefined providers (DynDNS.org and No-IP.com), the provider update URL and the check IP service URL are already completed. If you select the option "User defined" you can specify the two URLs manually.
Service type	Select whether the HTTP or HTTPS protocol will be used for the provider update URL of the predefined provider.
FQDN	Enter the host name (e.g. mysecuritymodule) and the domain name (e.g. dyndns.org) that is registered with the provider separated by a period. The FQDN can function as a VPN endpoint and differ from the FQDN in the "VPN" entry. You can configure which VPN endpoint the VPN partner is informed of in the connection-specific VPN settings. The syntax of an FQDN must comply with DNS. Maximum number of characters: 99
Ignore errors when checking the server certificate	To protect the authentication data the certificate of the update server is checked. If the certificate check fails, the HTTPS connection is terminated as default and the account data is not transferred. If you activate the check box, the HTTPS connection is not terminated if the check fails (e.g. with an expired certificate) and the account data is transferred. You are advised not to ignore the check and not to select the check box. The check box can only be activated if the service type "DDNS-HTTPS" was selected.
User	Enter the user name that you specified when you created the account. The syntax of a user name must comply with DNS. Maximum number of characters: 24
Password	Enter the password that you specified when you created the account. The syntax of the password must comply with DNS. Maximum number of characters: 24
Monitor IP address change on DSL router	If the security module is connected to the Internet via a DSL router, enabling this function activates the function of the Check IP service. The security module periodically sends queries to determine the current IP address of the DSL router and to detect an IP address change on the DSL router. The IP address specified in this way is sent to the provider with each change ID.

Setting	Meaning
Period	Specify the interval at which the Check IP service is called. • Default setting: 20 minutes • Minimum value: 10 minutes • Maximum value: 1440 minutes
Provider update URL	Enter the URL with which the security module reports the public IP address to the provider. The placeholder texts <FQDN> and <CurrentWanIP> need to be placed at the correct positions in the URL. The syntax of the URL must comply with DNS. Maximum number of characters: 127
Check IP service URL	Enter the URL that the security module will use to determine its public IP address. The syntax of the URL must comply with DNS. Maximum number of characters: 127

4. In case the primary provider fails, create a second provider in the "Secondary dynamic DNS service" area (optional setting).

Configuring LLDP

Module-specific function

This function is available only for SCALANCE S V4 modules or higher.

Requirement

The security module is in routing mode.

Meaning

LLDP (Link Layer Discovery Protocol) is a protocol used to discover network topologies. A device capable of LLDP can send information about itself to neighboring devices at regular intervals and at the same time receive information from neighboring devices. The received information is stored on every device with LLDP capability in an LLDP MIB file. Network management systems can access these LLDP MIB files using SNMP and therefore recreate the existing network topology.

Configurable parameters

The degree of activity of the security module in terms of LLDP can be configured under the "LLDP mode" entry of the relevant interface.

Parameter	Description
Name	Name of the port for which the setting is configured.
LLDP mode	Configured LLDP mode: • RxTx: LLDP frames can be sent and received • Off: No LLDP frames can be sent or received

Media redundancy in ring topologies

Media redundancy with MRP

Module-specific function

This function is available only for SCALANCE S627-2M modules.

Meaning

The term "media redundancy" groups together various methods for increasing availability in Industrial Ethernet networks in which devices can be reached over different paths. This might be achieved by meshing networks, arranging parallel transmission paths or by closing a linear bus topology to form a ring.

Media redundancy method MRP

Media redundancy within a ring topology is available with SIMATIC NET products among other things with the MRP method (Media Redundancy Protocol).

With this method, one of the nodes is configured as the redundancy manager. The other nodes are redundancy clients. SCALANCE S627-2M modules can only adopt the role of an MRP client. Using test frames, the redundancy manager checks the ring to make sure it is not interrupted. The redundancy clients forward the test frames. If the test frames of the redundancy manager no longer arrive at the other ring port of the redundancy manager due to an interruption, the redundancy manager switches through its two ring ports and informs the redundancy clients of the change immediately.

The time the SCALANCE X switches need to switch through their ring ports as redundancy manager is 200 ms.

Note on the use of MRP

- MRP is supported in ring topologies with up to 100 devices. Exceeding this number of devices can lead to a loss of data traffic.
- It is recommended that you set the ring ports involved to full duplex and 100 Mbps. Otherwise there may be a loss of data traffic.

Possible uses of MRP on media module ports

MRP is supported only on the media module ports of the SCALANCE S627-2M. The following table shows the possible uses of MRP on the media module ports of a SCALANCE S627-2M:

Ring ports	Media module 1		Media module 2	
	P4	P5	P6	P7
MRP Client	-	-	-	-
	Ring 1	Ring 1	-	-
	-	-	Ring 2	Ring 2
	Ring 1	Ring 1	Ring 2	Ring 2

With two lower-layer rings per SCALANCE S module, layer 3 communication is possible between the rings.

Configuring MRP for the security module

Requirement

- The security module is in routing mode.
- Media modules are configured for the interfaces to be connected to MRP rings.
- The interfaces of the security module to be connected to rings are networked with the relevant ring managers.

How to access this function

1. Select the security module to be edited.
2. In the settings of the required interface, select the "Media Redundancy" entry.

Configurable parameters

Parameter	Meaning	Possible selections
MRP domain (only if the "MRP client" media redundancy role is selected)	The members of an MRP ring are specified with the help of MRP domains. The same MRP domain must be selected for the interfaces of all modules to be connected to the same MRP ring.	Display of the MRP domain used for the interface.
Media redundancy role	Selection of the media redundancy protocol or disabling of media redundancy for the interface.	• Not a node in the ring • MRP Client
Ring port 1 (only when the media redundancy role "MRP client" is selected)	Name of the first ring port of the selected interface if the media redundancy role "MRP client" was selected for it.	-

Parameter	Meaning	Possible selections
Ring port 2 (only when the media redundancy role "MRP client" is selected)	Name of the second ring port of the selected interface if the media redundancy role "MRP client" was selected for it.	-
Domain settings	Using the domain settings, you can add new MRP domains, edit the names of existing MRP domains and delete existing MRP domains.	-
Alternative media redundancy protocol	Select this check box to enable the interface of the security module for other media redundancy protocols.	• Enable interface for other media redundancy protocols • Disable interface for other media redundancy protocols (default setting)
Passive listening	Enable this check box if the selected interface will be connected to third-party networks in which STP/RSTP (Spanning Tree Protocol/ Rapid Spanning Tree Protocol) is used.	• Activate passive listening (default) • Deactivate passive listening

Special features of the ghost mode

Module-specific function

This function is available only for SCALANCE S602 as of V3.1.

Meaning

In ghost mode, the security module has no IP address of its own, neither on the internal nor on the external interface. Instead, the security module obtains the IP address for its external interface during runtime from a node connected to the internal interface of the security module whose IP address parameters may be unknown at the time of configuration. It is possible to change an IP address of the internal node and a corresponding IP address at the external interface. Since the internal node is identified based on its MAC address, IP address changes are made only for the learned MAC address. No IP address is configured or obtained at the internal interface of the security module.

As regards the MAC addresses, the security module replaces the MAC address of the internal node with the MAC address of the security module in all outgoing packets on the external interface (responses from the internal node).

Activating ghost mode

1. Select the module to be edited.
2. In the local security settings, select the entry "Mode".
3. Select the "Ghost mode" option.

Configurable module properties

In ghost mode, the following module properties can be configured in the local security settings:

- External interface [P1] red
- Firewall
- Time-of-day synchronization
- Log settings
- SNMP

Since no DNS servers can be configured in ghost mode, no FQDN resolution is possible.

Requirement for identifying an internal node

The security module can only obtain the IP address of the internal node if the internal node initiates data communication with a communications partner of the external network.
In addition to this, the security module does not provide any server services while obtaining the IP address. The security module can only reply to queries from external after data packets have been sent to the security module by the internal node.

Port assignment for incoming and outgoing data connections

Since the external interface of the security module and the internal node have the same IP address, the network components must be addressed explicitly via the TCP/UDP ports. For this reason, the ports are either assigned to the security module or the internal node. The assignments of the ports to the relevant devices are shown in the following tables for incoming and outgoing data connections:

Table 10-51 Port assignment for incoming connections (from external to security module)

Service	Port	Protocol	Comment
Web services, configuration and diagnostics access	443	TCP	The HTTPS port is permanently activated for configuration and diagnostics access using STEP 7 and cannot be changed.
SNMP	161	TCP UDP	Once SNMP is activated in STEP 7, incoming SNMP queries are transmitted via UDP port 161. Transfer via TCP port 161 is also possible, for example, to be able to reach the internal node. **Note** After activating SNMP, the SNMP port is permanently assigned to the security module. If SNMP is not activated, the internal node can be accessed using SNMP with the aid of a firewall rule.

Table 10-52 Port assignment for outgoing connections (from security module to external)

Service	Port	Protocol	Comment
Syslog	514	UDP	If the Syslog service is activated in STEP 7, Syslog messages are transferred via UDP port 514 by the security module. This port assignment cannot be changed.
NTP	123	UDP	If NTP servers are used for time-of-day synchronization, NTP queries are transferred via UDP port 123. This port assignment cannot be changed.

Recognizable IP addresses and subnet masks

The security module only recognizes internal nodes with IP addresses in the network class ranges A, B or C. The subnet mask is identified by the security module based on the network class (refer to the table "Network classes and corresponding subnet masks"). To allow the subnet mask to be determined correctly, a standard router must be entered for the internal node.

Nodes with IP addresses in the network classes D and E are rejected by the security module.

Network class	IP addresses		Subnet mask
	Low limit	High limit	
A	0.0.0.0	127.255.255.255	255.0.0.0
B	128.0.0.0	191.255.255.255	255.255.0.0
C	192.0.0.0	223.255.255.255	255.255.255.0
D	224.0.0.0	239.255.255.255	Rejected by the security module
E	240.0.0.0	255.255.255.255	Rejected by the security module

Configuration limits

A maximum of one internal node is recognized by the security module. If several internal nodes exist, the security module reacts as follows:

- The first device the security module recognizes in the internal network obtains access to the external network segment if the firewall is suitably configured.
- The data traffic of any additional nodes in the internal network area is blocked in the outgoing direction at level 2 (MAC layer) based on the sender address.

Loading configurations and diagnostics after commissioning

After obtaining an IP address from the internal node, the security module has an IP address on the external interface that can differ from the IP address with which the security module was initially configured. To load a configuration or to run diagnostics, in STEP 7 you need to specify the IP address for the connection to the external interface that the security module obtained from the internal node during runtime. This is possible in the local security settings or directly in the "Advanced download" or "Connect online" dialogs. For more detailed information on establishing online connections, refer to the section: Downloading a configuration (Page 890)

Routing information for hierarchical networks on the external interface

If there are hierarchical networks with subnet transitions on the external interface of the security module, the security module needs to obtain the relevant routing information from the internal node. To achieve this, the internal node must respond to ICMP queries sent to it. Responding to ICMP broadcasts is not necessary.

Authentication using a RADIUS server

Overview

Module-specific function

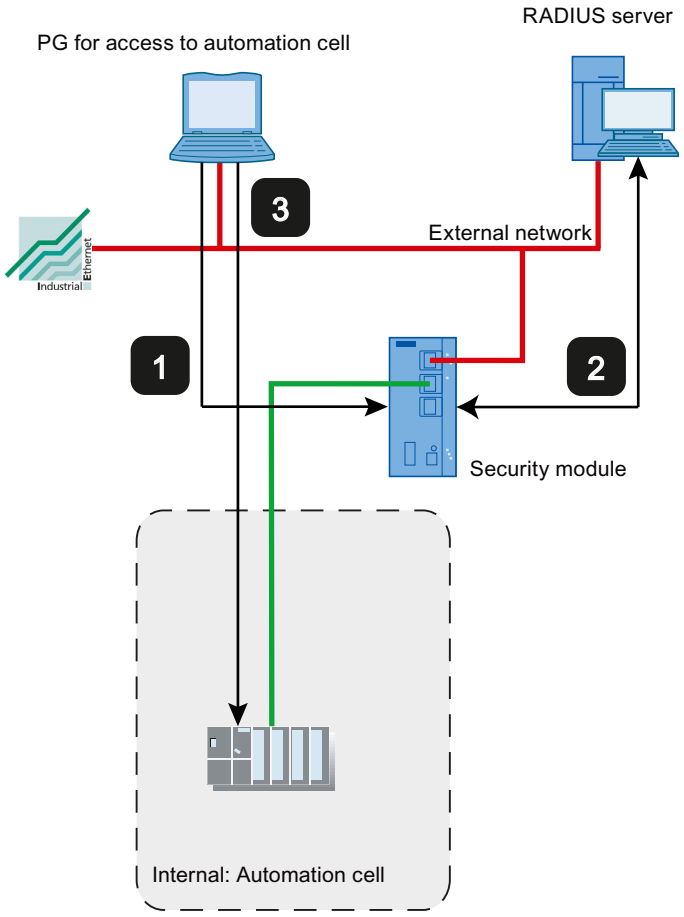
This function is available only for SCALANCE S V4 modules or higher.

Meaning

RADIUS (Remote Authentication Dial-In User Service) is a protocol for authenticating users by servers on which user data can be stored centrally. The use of RADIUS servers can increase the protection of user names, assigned roles and passwords.

Scenario for the use of RADIUS servers

Authentication by RADIUS servers can be performed when activating user-specific IP rule sets.



1 Entry of the user data on the Web page of the security module

2 Authentication by RADIUS server and activation of the user-specific IP rule set

3 Access to an automation cell

The network setup shown above is simply an example. The RADIUS server can also be located in the internal network or in the DMZ network of the security module.

For the configuration options described below, it is assumed that a RADIUS server is configured in STEP 7 and was assigned to the relevant security module. In addition to this, one user or role must be configured with the "RADIUS" authentication method. For more detailed information, refer to the following sections:

- Defining a RADIUS server (Page 859)
- Assigning a RADIUS server to a security module (Page 860)
- Create users (Page 733)
- Creating roles (Page 736)

For general information on user-specific IP rule sets, refer to the following section:

- AUTOHOTSPOT

Configuration options

To authenticate the user using a RADIUS server, there are two configuration options available:

- The user and the user's role are known on the security module, only the password management for the user is performed on the RADIUS server. The user and the password are configured on the RADIUS server.
  - A user with the "RADIUS" authentication method is configured.
  - The user is assigned to the user-specific IP rule set.

  Result:
  - When a user logs on to the Web page of the security module, the authentication query is forwarded to the RADIUS server.
  - The RADIUS server runs a password check and signals the result back to the security module.
  - If the password check is passed successfully, the user-specific IP rule set is activated.
- The role is known on the security module, user management is via the RADIUS server. The user and the password are configured on the RADIUS server.
  - A user-defined role or a system-defined role is assigned to the user-specific IP rule set.
  - Under the entry "RADIUS" > "RADIUS settings" in the local security settings of the security module, the check box "Allow RADIUS authentication of unconfigured users" and the check box "Filter ID is required for authentication" are enabled.

  Result:
  - When a user logs on to the Web page of the security module, the authentication and authorization query is forwarded to the RADIUS server.
  - The RADIUS server runs a password check and signals the result back to the security module.
  - Case a: If, in addition to this, the role name is configured on the RADIUS server:
    The RADIUS server returns the role name assigned to the user to the security module.
  - Case b: If the role name is not configured on the RADIUS server:
    The security module assigns the user the system-defined role "radius".
  - If the password check is passed successfully, the user-specific IP rule set is activated.

Conventions for RADIUS servers

- The RADIUS servers can be in any network connected to the security module.
- A maximum of two RADIUS servers can be configured per security module. During operation only one of the RADIUS servers is active.
- When defining a RADIUS server, an FQDN can also be used instead of IP an address.
- A maximum of 25 characters can be used for the name of a RADIUS server.

Defining a RADIUS server

Meaning

Before authentication by a RADIUS server is possible, this first needs to be stored in the STEP 7 project. Following this, you assign the defined RADIUS server to the security module for which the RADIUS server will handle user authentication.

Procedure

1. In the global security settings, select the entry "RADIUS".
   Result: The previously created RADIUS servers are displayed under the selected entry.
2. Double-click on the "Add new RADIUS server" entry.
   Result: The created RADIUS server is displayed with an automatically assigned number below the entry "RADIUS".
3. Double-click on the created RADIUS server.
   Result: In the working area the assignment dialog for RADIUS servers is displayed. The selected RADIUS server is selected in the "RADIUS server" drop-down list. The required security modules can be assigned to this, see section Assigning a RADIUS server to a security module (Page 860). In the local security settings, the configurable properties of the RADIUS server are displayed.
4. Enter the following data in the "General" entry in the local security settings:

Parameter	Meaning
Name	Name of the RADIUS server. The name can consist of a maximum of 25 characters. The characters 0-9, a-z, A-Z, -._ can be used. A name must start with a letter and end with a letter or number. Before the '.' character only letters and numbers are permitted, after the '.' character only letters. Before the '_' or '-* character only letters and numbers are permitted.
IP address / FQDN	IP address or FQDN of the RADIUS server.
Port	UDP port via which the RADIUS server can be reached. As default, authentication data is received at port 1812.
Shared secret	Entry of the password that will be used when transferring the logon data between the RADIUS server and security modules for encryption. The following characters from the ANSI X 3.4-1986 character set are permitted: 0123456789 A...Z a...z !#$%&()"*'+`,-./:;<=>?@ [\]_{|}~^ Length of the shared secret: 1 to 32 characters
Repeat shared secret	Confirmation of the password
Authentication method	Display of the method used to check the user data. Only the "PAP" method (Password Authentication Protocol) is supported.
Comment	Entry of freely selectable, optional comments.

Result

You have defined a RADIUS server and can now assign this to the required security modules.
Note the descriptions in the section below:
Assigning a RADIUS server to a security module (Page 860)

Assigning a RADIUS server to a security module

Requirement

You have defined a RADIUS server.

Procedure

1. Select the module to be edited.
2. Select the "RADIUS" entry in the local security settings.
3. Select the "Enable RADIUS authentication" check box.

   Note
   Changing the method of authentication with the Web server on the security module

   If RADIUS authentication is enabled on the security module, the method for authentication with the Web server is changed from "Digest Access Authentication" to "Basic Access Authentication".

4. In the "RADIUS timeout" input box, enter the maximum time in seconds that the security module will wait for a response from the RADIUS server.
5. In the "RADIUS repetitions" input box, enter the number of connection establishment attempts with the RADIUS server.
6. Select the "Allow RADIUS authentication of unconfigured users" check box if the user-specific IP rule to be activated was assigned a role instead of a user.
7. Select the "Filter ID is required for authentication" check box if the assigned role is a user-defined role.
8. Under the entry "RADIUS server", select the RADIUS server you want to assign to the security module from the "Name" drop-down list.
   As an alternative, you can assign security modules for which RADIUS authentication is enabled to RADIUS servers in the global security settings. You will find general information on authentication by the RADIUS server in the following section:
   AUTOHOTSPOT

Setting up a firewall

Local firewall rules for SCALANCE S modules

Configuring a firewall with predefined firewall rules

Configuring a firewall using predefined firewall rules

How to access this function

1. Select the module to be edited.
2. Select the "Firewall" entry in the local security settings.

Firewall enabled as default

The "Enable firewall" check box is enabled by default. The firewall is therefore activated automatically and all access from external to the security module is blocked. By clicking the relevant check box, enable the firewall rules for the specific direction.

Note

Detailed firewall settings in advanced firewall mode

In advanced firewall mode, you can restrict firewall rules to individual nodes. To change to advanced firewall mode, select the "Enable firewall in advanced mode" check box. For more detailed information about the advanced firewall mode, refer to the section
Overview of local firewall rules (Page 770)

Firewall configuration with VPN

If the security module is in a VPN group, the "Tunnel communication only" check box is enabled as default. This means that only encrypted IPsec data transfer is permitted via the external interface or via the DMZ interface. Only HTTPS access to the module (TCP port 443) remains allowed untunneled.

If you deselect the check box, tunneled communication and also the types of communication selected in the other boxes are permitted.

Table 10-53 Available firewall rules and directions

Service	From internal to external	From external to internal	From internal to DMZ	From DMZ to internal	Enabled ports	Meaning
Allow IP communication	x	x	x	x	-	IP communication for the selected communication directions is allowed.
Allow S7 protocol	x	x	x	x	TCP port 102	Communication of the nodes using the S7 protocol is allowed.

Service	From internal to external	From external to internal	From internal to DMZ	From DMZ to internal	Enabled ports	Meaning
Allow FTP/ FTPS (explicit mode)	x	x	x	x	TCP port 20 TCP port 21	For file management and file access between server and client.
Allow HTTP	x	x	x	x	TCP port 80	For communication with a Web server.
Allow HTTPS	x	x	x	x	TCP port 443	For secure communication with a Web server, for example, for Web diagnostics.
Allow DNS	x	x	x	x	TCP port 53 UDP port 53	Communication connection to a DNS server is allowed.
Allow SNMP	x	x	x	x	TCP port 161/162 UDP port 161/162	For monitoring nodes capable of SNMP.
Allow SMTP	x	x	x	x	TCP port 25	For sending e-mails via an SMTP server.
Allow NTP	x	x	x	x	UDP port 123	For synchronization of the time of day.
Allow DHCP	x	x	x	x	UDP port 67 UDP port 68	Only in bridge mode Communication connection with a DHCP server is permitted.

Table 10-54 Logging

Option	Action when activated
Log tunneled packets	Only active if the security module is a member of a VPN group. All IP packets transferred via the tunnel are logged.
Log blocked incoming packets	All incoming IP packets that are discarded are logged.
Log blocked outgoing packets	All outgoing IP packets that are discarded are logged.

You can view the logged packets at the "Packet filter log" entry in the "Online & Diagnostics" dialog. For more detailed information, refer to the section Logging data packets - "Packet filter log" entry (Page 833).

Configuring a firewall with predefined MAC rules

How to access this function

1. Select the module to be edited.
2. Select the "Firewall" entry in the local security settings.

Firewall enabled as default

The "Enable firewall" check box is enabled by default. The firewall is therefore activated automatically and all access from external to the security module is blocked. By clicking the relevant check box, enable the firewall rules for the specific direction.

Note

Detailed firewall settings in advanced firewall mode

In advanced firewall mode, you can restrict firewall rules to individual nodes. To change to advanced firewall mode, select the "Enable firewall in advanced mode" check box. For more detailed information about the advanced firewall mode, refer to the section Overview of local firewall rules (Page 770).

Firewall configuration with VPN

If the security module is in a VPN group, the "Tunnel communication only" check box is enabled as default.

If you deselect the check box, tunneled communication and also the types of communication selected in the other boxes are permitted.

Available MAC rules and directions

Service	From internal to external	From external to internal	Meaning
Allow MAC communication	x	x	MAC traffic for the selected communication directions is allowed.
Allow ISO protocol	x	x	ISO traffic for the selected communication directions is allowed.
Allow SiClock	x	x	SiClock time-of-day frames for the selected communication directions are allowed.
Allow DCP	x	x	DCP traffic for assignment of IP addresses is allowed for the selected communications directions.

Table 10-55 Logging

Option	Action when activated
Log tunneled packets	Only active if the security module is a member of a VPN group. All MAC packets transferred via the tunnel are logged.
Log blocked incoming packets	All incoming MAC packets that are discarded are logged.
Log blocked outgoing packets	All outgoing MAC packets that are discarded are logged.

User-specific IP rule sets

Overview

Meaning

Initially, individual or multiple users are assigned to user-specific IP rule sets. The user-specific IP rule sets are then assigned to individual or multiple security modules. This makes it possible, to allow user-specific access. If, for example all access to the networks downstream from a security module is blocked, certain nodes can be allowed for a user based on their IP addresses. This means that access is allowed for this user but access remains blocked for other users.

User logon via the Internet

The user can log in to the external interface or the DMZ interface via the Web page of the security module. If authentication is successful, the IP rule set defined for the user for the IP address of the device from which the login is made is enabled.

The connection to the Web page of the security module is via HTTPS using the IP address of the connected port and taking into account the valid routing rules:

Example:

External interface: 192.168.10.1

Call up of the login page with: https://192.168.10.1/

Users can log on with every role as long as the user or the role is assigned to a user-specific firewall rule set.

Options for authenticating the user

Depending on the authentication method selected when the user who will log in to the security module was created, the authentication is handled by different instances:

- Authentication method "Password": Authentication is handled by the security module.
- Authentication method "RADIUS": Authentication is handled by a RADIUS server (SCALANCE S as of V4 only).

Assignment of roles to user-specific IP rule sets

On SCALANCE S modules as of V4, it is also possible to assign user-specific IP rule sets that are assigned to roles. This makes it possible to enable a group of users for access to certain IP addresses.

If a RADIUS server is used for user authentication and a role is assigned to the user-specific IP rule set, users can also be authenticated by the RADIUS server although they are not configured on the security module. These users must be stored on the RADIUS server or on a separate database where they need to be assigned to the role assigned to the user-specific IP rule set in STEP 7. This procedure has the advantage that all user data is stored exclusively on the RADIUS server.

You will find more information on authentication by the RADIUS server in the following section:

AUTOHOTSPOT

User-specific IP rule sets are used locally - conventions

The same conventions as described in the following section apply:
Global firewall rule sets - conventions (Page 750)

Creating and assigning user-specific IP rule sets

Creating user-specific IP rule sets

1. In the global security settings, select the entry "Firewall" > "User-specific IP rule sets" > "IP rule sets".
2. Double-click on the "Add new IP rule set" entry to create a user-specific IP rule set.
   Result: The created user-specific IP rule set is shown in the entry.
3. Double-click on the created user-specific IP rule set.
   Result: In the "Properties" > "General" tab of the Inspector window, the configurable properties of the user-specific IP rules set are displayed.
4. In the Inspector window, click on the "General" entry and enter the following data:
   - Name: Project-wide, unique name of the rule set. The name appears in the local rule list of the security module after the rule set is assigned.
   - Description (optional): Enter a description of the user-specific IP rule set.
5. Click on the "IP rules" entry and enter the firewall rules one after the other in the list.
   No IP address can be entered in the “Source IP address” box. This is entered automatically when the node logs on to the security module.
   The “Stateful” check box cannot be disabled for IP rules of user-specific IP rule sets
   Note the parameter description in the following sections:
   Defining IP packet filter rules (Page 771)
   Note the special features of firewall rules generated automatically by STEP 7 for NAT/NAPT rules:
   Relationship between NAT/NAPT router and user-specific firewall (Page 872)

Assigning user-specific IP rule sets

1. Click on the "Users and roles" entry in the Inspector window.
2. In the "Available users and roles" area, select the users you want to assign to the user-specific rule set.

3. Click the "<<" button to assign the selected users or roles to the user-specific rule set. The assignment of roles to user-specific IP rule sets is possible only for SCALANCE S modules as of V4.
4. Assign the created user-specific IP rule set to the required security modules using the entry "Assign user-specific IP rule set" in the global security settings. For this, advanced firewall mode must be enabled for the security modules.

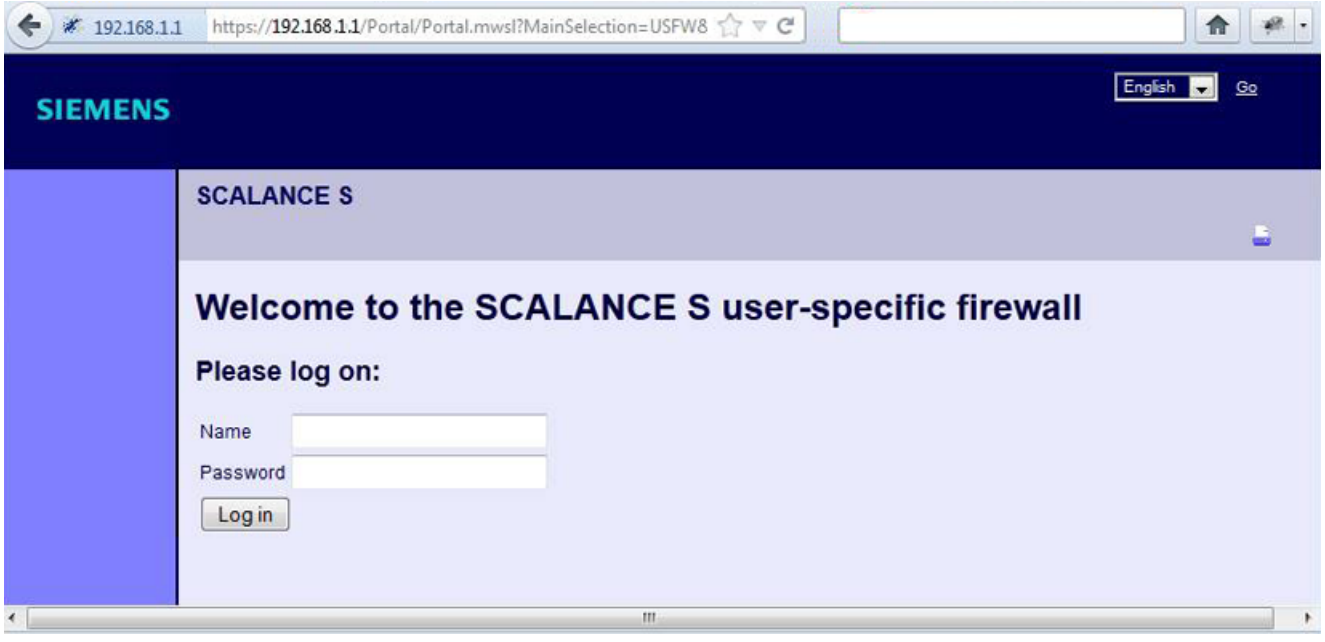
Note

Assignment of user-specific IP rule sets

- A security module can only be assigned one user-specific rule set per user.
- Due to the assignment, the right to log in to the security module is activated implicitly for all users and roles assigned to the IP rule set.

Result

- The user-specific rule set is used by the assigned security modules as a local rule set and automatically appears in the local list of firewall rules.
- The user can log on with the security module. Authentication of the user is performed depending on the selected authentication method either by the security module or a RADIUS server.



Range of values for the maximum session time

The time after which the user is automatically logged out can be specified when creating or editing a user; the default is 30 minutes. On the Web page of the security module, the session time can be extended to the value assigned to the user.

You will find more information creating users in the following section:

Create users (Page 733)

IP packet filter directions SCALANCE S

Meaning

Possible selections for the communication directions "From" and "To" in the IP rules of the advanced firewall mode.

The following directions are available

Available options / ranges of values		Security module		
From	**To**	**S602**	**S61x**	**S623 / S627-2M**
Internal	External	**x**	**x**	**x**
	Tunnel	-	**x**	**x**
	Any	-	**x**	**x**
	DMZ	-	-	**x**
	Internal	**x**	**x**	**x**
External	Internal	**x**	**x**	**x**
	Any	-	-	**x**
	Tunnel	-	-	**x**
	DMZ	-	-	**x**
Tunnel	Internal	-	**x**	**x**
	External	-	**x**	**x**
	DMZ	-	-	**x**
Any	Internal	-	**x**	**x**
	External	-	-	**x**
	DMZ	-	-	**x**
DMZ	Internal	-	-	**x**
	External	-	-	**x**
	Any	-	-	**x**
	Tunnel	-	-	**x**

x = communication direction is configurable

- = communication direction is not configurable

MAC packet filter directions SCALANCE S

Meaning

Possible selections for the communication directions "From" and "To" in the MAC rules of the advanced firewall mode.

The following directions are available

Available options / ranges of values		Security module		
From	**To**	**S602**	**S61x**	**S623 / S627-2M**
Internal	External	**x**	**x**	**x**
	Tunnel	-	**x**	**x**
	Any	-	**x**	**x**
External	Internal	**x**	**x**	**x**
	Any	-	-	**x**
	Tunnel	-	-	**x**
Tunnel	Internal	-	**x**	**x**
	External	-	**x**	**x**
Any	Internal	-	**x**	**x**
	External	-	-	**x**

x = communication direction is configurable

- = communication direction is not configurable

Adapting standard rules for IP services

Requirement

This function is available only in advanced firewall mode.

How to access this function

1. Select the security module to be edited.
2. In the local security settings, select the entry "Firewall" > "Standard rules for IP services".

Meaning of the advanced settings

Parameter	Meaning when activated
Use advanced status options	The permitted number of connections and firewall statuses per time unit is limited. If a network node exceeds this limit, its IP address is entered in the IP blacklist of the security module. The node can then no longer communicate via the security module. The IP blacklist of the security module can be viewed in online mode.
Log all activated rules	Packets that are allowed according to the default rules for IP services are logged.
Enable ICMP test for interfaces	Ping queries coming in on an interface of the security module can be forwarded to other interfaces. This means that, for example, ping queries can be made from the external network to the internal interface of the security module.

Default firewall rules for SCALANCE S

The following table lists the default firewall rules for SCALANCE S modules. In some cases, the firewall rules are only set if the relevant service is used by the security module (e.g. SNMP).

Service	Direction	X1 interface (red)	Interface X2 (green)	Interface X3 (yellow) (only for S623, S627-2M)	Tunnel interface* (not for S602)
Interface rerouting	outgoing	-	x	-	-
HTTPS		x	x*	x	x
ICMP	incoming	-	x	-	x
ICMP pathfinder (only for SCALANCE S S602 as of V3.1 in ghost mode)	outgoing	-	x	-	-
SNMP	incoming	x	x	x	x
Syslog	outgoing	x	x	x	x
NTP	outgoing	x	x	x	x
DNS	outgoing	x	x	x	x
HTTP	outgoing	x	-	x	-
VPN (IKE)		x	-	x	-
VPN (NAT Traversal)		x	-	x	-
BootP Server	incoming	-	x	x	-
BootP Client	outgoing	-	x	x	-
RADIUS	outgoing	x	x	x	x
CARP (only for SCALANCE S62x as of V4)	outgoing	x*	x*	-	-
Pfsync (only for SCALANCE S62x as of V4)	outgoing	-	-	x*	-

x enabled as default

- disabled as default

* cannot be adapted

SCALANCE S module as router

Specifying routes

Meaning

Specifies routes for addressing subnets that cannot be reached directly via the security module.

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the "Routing" entry.
3. Double-click "Add new" in the table to add a route.
4. Enter the following values:

Parameter	Function	Example of a value
Network ID	Requests to nodes of the subnet with the network ID and the subnet mask specified here are forwarded to the subnet via the specified router IP address. Based on the network ID, the router recognizes whether a destination address is inside or outside the subnet. The specified network ID must not be located in the same subnet as the IP address of the security module.	192.168.11.0
Subnet mask	The subnet mask determines the network structure. Based on the network ID, the router recognizes whether a destination address is inside or outside the subnet.	255.255.255.0
Router IP address	IP address of the router that connects to the subnet. As an alternative, an FQDN can be entered for SCALANCE S modules as of V4. The specified router IP address must be in the same subnet as the IP address of the security module.	192.168.10.2
Activate rerouting (only for SCALANCE S as of V3)	Select this check box if the frames of the entered route will enter and leave on the same interface of the security module (rerouting). Rerouting is only supported on the internal interface of the security module.	

Address translation with NAT/NAPT in VPN tunnels

Module-specific function

Address translation with NAT/NAPT in VPN tunnels is only available for SCALANCE S612/ S623/S627-2M modules as of V4.

Meaning

Address translations with NAT/NAPT can also be performed for communications relations established via a VPN tunnel.

Requirements

The following requirements apply generally to a SCALANCE S module that will perform an address translation with NAT/NAPT in a VPN tunnel:

- The SCALANCE S module is in a VPN group.
- The SCALANCE S module is in routing mode and/or the DMZ interface of the SCALANCE S module is activated.

- The tunnel interface is enabled.
- The advanced firewall mode is enabled.

Supported address translation directions

The address translation directions described in the following section are supported:
NAT/NAPT routing (Page 786)

Supported address translation actions

With tunneled communications relations, the following address translation actions are supported:

- Destination NAT ("Redirect")
- Source NAT ("Masquerading")
- Source and Destination NAT ("1:1-NAT")
- NAPT ("Port forwarding")

You will find basic information on these address translation actions in the following section:
NAT/NAPT routing (Page 786)

Supported VPN links

In conjunction with NAT/NAPT, the following VPN links are supported:

VPN link		VPN link is initiated by	Address translation is performed by
SCALANCE S (a)	SCALANCE S (b)	SCALANCE S (a) or SCALANCE S (b)	SCALANCE S (a) and/or SCALANCE S (b)
SCALANCE S	CP 343-1 Adv. / 443-1 Adv. / PC-CP	SCALANCE S or CP 343-1 Adv. / 443-1 Adv. / PC-CP	SCALANCE S
SCALANCE S	SCALANCE M	SCALANCE M	SCALANCE S and/or SCALANCE M*
SOFTNET Security Client	SCALANCE S	SOFTNET Security Client	SCALANCE S

* Only 1:1 NAT is supported.

SCALANCE S modules of the type SCALANCE S623 V4 and SCALANCE S627-2M V4 that have a VPN endpoint on the external interface and on the DMZ interface can perform address translations on both interfaces at the same time.

Address translation characteristics when involved in several VPN groups

If a SCALANCE S module is a member of several VPN groups, the address translation rules configured for the tunnel interface of the SCALANCE S module apply for all VPN connections of this SCALANCE S module.

Please note: Once you have configured a NAT address translation to or from the direction of the tunnel, only the IP addresses involved in the NAT address translation rules can be reached via the VPN tunnel.

Relationship between NAT/NAPT router and user-specific firewall

Module-specific function

The configuration of NAT/NAPT rules in the user-specific firewall is only available for SCALANCE S modules as of V3.

Meaning

After creating NAT/NAPT rules, STEP 7 automatically generates a user-specific IP rule set in the user-specific firewall that enables communication in the configured address translation direction. You can then assign this user-specific IP rule set to individual or multiple users and/or individual or multiple roles (only for SCALANCE S modules as of V4).

The generated firewall rules can, if necessary, be moved and expanded (additional IP address, services, bandwidth). Firewall parameters generated by STEP 7 cannot be adapted. If the user-specific IP rule set is assigned to a security module with NAT/NAPT deactivated, the NAT/NAPT rules from the user-specific firewall are also applied to this security module.

Note

The address translation actions "Source-NAT + Destination-NAT" and "Double-NAT" are not supported in conjunction with the user-specific firewall.

How to access this function

"NAT" or "NAPT" entry in the editor for user-specific IP rules sets, refer to the following section: Creating and assigning user-specific IP rule sets (Page 865)

Supported address translation directions for the action "Source NAT"

The action "Source NAT" can be performed in the following directions:

- External to DMZ
- DMZ to external

No IP address can be entered in the "Source IP address" box. This is entered automatically when the node logs on to the security module.

Supported address translation directions for the action "Destination NAT"

The action "Destination NAT" can be performed in the following directions:

- External to internal
- External to DMZ
- DMZ to internal
- DMZ to external
- Tunnel to internal (only SCALANCE S612/S623/S627-2M as of V4)

- Tunnel to external (only SCALANCE S612/S623/S627-2M as of V4)
- Tunnel to DMZ (only SCALANCE S612/S623/S627-2M as of V4)

Supported address translation directions for NAPT

The address translation with NAPT can be performed in the following directions:

- External to internal
- External to DMZ
- DMZ to internal
- DMZ to external
- Tunnel to internal (only SCALANCE S612/S623/S627-2M as of V4)
- Tunnel to external (only SCALANCE S612/S623/S627-2M as of V4)
- Tunnel to DMZ (only SCALANCE S612/S623/S627-2M as of V4)

NAT/NAPT address translation and corresponding user-specific IP rule sets

In the firewall rules for user-specific IP rule sets generated based on NAT/NAPT rules, no IP address can be entered in the "Source IP address" box. This is entered automatically when the node logs on to the security module. The remaining properties generated locally for individual security modules are identical. Refer to the section
Relationship between NAT/NAPT router and firewall (Page 792)

Security module as DHCP server

Overview DHCP server

Overview

You can operate the SCALANCE S module on the internal network and on the DMZ network as DHCP server (DHCP = Dynamic Host Configuration Protocol). This allows IP addresses to be assigned automatically to the devices connected to the internal network or the DMZ network.

Simultaneous DHCP server operation on both interfaces is possible (SCALANCE S623/ S627-2M only).

The IP addresses are either distributed dynamically from an address range you have specified or you can select a specific IP address and assign it to a particular device. If devices on the internal interface or on the DMZ interface should always be assigned the same IP address for firewall configuration, the address assignment must only be static based on the MAC address or based on the client ID.

Requirements

You configure the devices on the internal network or on the DMZ network so that they obtain the IP address from a DHCP server.

Depending on the mode, the security module sends an IP address of the standard router to the nodes in the relevant subnet or you need to inform the nodes in the subnet of a router IP address.

- Router IP address will be transferred
  In the following situations, the DHCP protocol of the security module will inform the nodes of the router IP address:
  - The node is connected to the DMZ interface (SCALANCE S623/S627-2M only)
    In this case, the security module sends its own IP address as the router IP address.
  - The node is connected to the internal interface and the security module is configured for router mode
    In this case, the security module sends its own IP address as the router IP address.
  - The node is connected to the internal interface and the security module is not configured for router mode, there is, however, a standard router specified in the configuration of the security module.
    In this case, the security module transfers the IP address of the standard router as the router IP address.
- Router IP address will not be transferred
  In the following situations, enter the router IP address manually on the node:
  - The node is connected to the internal interface and the security module is not configured for router mode. There is also no standard router specified in the configuration of the security module.

See also

Configuring a DHCP server (Page 874)

Configuring a DHCP server

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "DHCP server".

3. Select the interface for which you want to make the DHCP settings.
4. Make the address assignment. You have the following configuration options:
   - Static address assignment
     Devices with a specific MAC address or client ID are assigned the specified IP addresses. You specify these addresses by entering the devices in the address list in the "Static address assignment" group box.
   - Dynamic address assignment
     Devices whose MAC address or whose client ID was not specified specifically, are assigned a random IP address from the specified address range. For this purpose, activate the "Dynamic IP address pool" check box. Set the address range in the "Dynamic IP address pool" input area.

Note

Dynamic address assignment - reaction after interrupting the power supply

Please note that dynamically assigned IP addresses are not saved if the power supply is interrupted. On return of the power, you must therefore make sure that the nodes request an IP address again.

You should therefore only use dynamic address assignment for the following nodes:

- Nodes that are used temporarily in the subnet (such as service devices).
- Nodes that have been assigned an IP address and send this as the "preferred address" the next time they request an address from the DHCP server (for example PC stations).

For permanent nodes you should preferably use static address assignment by specifying a client ID or the MAC address.

Consistency check - these rules must be adhered to

Remember the following rules when making the entries.

- The IP addresses assigned in the address list in the "Static address assignments" group box must not be in the range of the dynamic IP addresses.
- IP addresses, MAC addresses and client IDs may only occur once in the "Static address assignment" table (related to the security module).
- For the statically assigned IP addresses, you must specify either the MAC address or the client ID (computer name).
- The client ID is a string with a maximum of 63 characters. Only the following characters may be used: a-z, A-Z, 0-9 and - (dash).
  Note
  In SIMATIC S7, a client ID can be assigned to the devices on the Ethernet interface to allow them to obtain an IP address using DHCP.
  With PCs, the procedure depends on the operating system being used; it is advisable to use the MAC address here for the assignment.
- For the statically assigned IP addresses, you must specify the IP address.

- The following IP addresses must not be in the range of the free IP address range (dynamic IP addresses):
  - All router IP addresses in the "Routing" entry
  - Syslog server
  - Standard router
  - Security module address(es)
- DHCP is supported by the security module on the interface to the internal subnet and on the interface to the DMZ network. The following additional requirements for IP addresses in the range of the dynamic address assignments result from this operational behavior of the security module:
  - Bridge mode
    The range must be within the network defined by the security module.
  - Routing mode
    The range must be within the internal subnet defined by the security module.
    Note
    The DMZ network always represents a separate subnet. When using DHCP on the DMZ interface, make sure that the free IP address range (dynamic IP addresses) is within the DMZ subnet.
- The free IP address range must be fully specified by entering the start address and the end address. This end address must be higher than the start address.
- The IP addresses you enter in the address list in the "Static address assignments" input area must be in the address range of the internal subnet or in the DMZ subnet of the security module.

See also

Running a consistency check (Page 727)

Configuring proxy ARP

Overview

Proxy ARP allows routers to respond to ARP queries for hosts. The hosts are in networks separated by routers but use the same IP address range.

If PC1 sends an ARP request to PC2, it receives an ARP response and the hardware address of the interface (MAC address of the port of the security module) on which the query was received from the security module located in between and not from PC2. The querying PC1 then sends its data to the security module that then forwards it to PC2.

How to access this function

This function is only available for the internal interface of a security module that is a member of a VPN group and is in bridge mode.

1. Select the security module to be edited.
2. In the local security settings, select the entry "Proxy ARP".
3. If the security module is to respond to an ARP query from its own LAN as a substitute for the specific connection partner, enter the corresponding IP address.

IPsec tunnel: Creating and assigning groups

Configuring internal network nodes

Using the learning mode to learn internal nodes

Finding nodes for tunnel communication automatically

One great advantage when configuring and operating tunnel communication is that SCALANCE S modules in bridge mode can find the nodes on the internal interface automatically. New nodes are detected by the security module during operation. The detected nodes are signaled to the security modules belonging to the same VPN group. This allows data exchange within the tunnels of a group in both directions at any time.

Detectable nodes

The following nodes are detected:

- Network nodes with IP capability
  Network nodes with IP capability are found when they transmit an ICMP response to the ICMP subnet broadcast.
  IP nodes downstream from routers can be found if the routers pass on ICMP broadcasts.
- ISO network nodes
  You can also teach-in network nodes without IP capability that can be addressed by means of ISO protocols.
  This is only possible if they reply to XID or TEST packets. TEST and XID (Exchange Identification) are auxiliary protocols for exchanging information on layer 2. By sending these packets with a broadcast address, these network nodes can be located.
- PROFINET nodes
  Using DCP (Discovery and basic Configuration Protocol), it is possible to find PROFINET nodes.

Network nodes that do not meet these conditions must be configured manually.

Subnets located on the other side of internal routers also need to be configured manually.

How to access the function

1. Select the module.
2. In the local security settings, select the entry "VPN" > "Nodes".

Enabling/disabling the learning mode

The learning function is enabled in the configuration as default for every security module.

Learning can also be disabled completely for SCALANCE S. In this case, you will need to configure all internal network nodes participating in the tunnel communication manually.

When is it useful to disable the automatic learning mode?

The default settings for the security module assume that internal networks are always secure; in other words, in a normal situation, no network node is connected to the internal network if it is not trustworthy.

Disabling the learning mode can be useful if the internal network is static; in other words, when the number of internal nodes and their addresses do not change.

If the learning mode is disabled, this reduces the load on the medium and the nodes in the internal network resulting from the learning packets. The performance of the security module is also slightly improved since it does not need to process the learning packets.

Note: In the learning mode, all nodes in the internal network are detected. The information relating to VPN configuration limits relates only to nodes that communicate over VPN in the internal network.

Note

If more than 128 internal nodes are being operated, the permitted configuration limits are exceeded and an illegal operating status results. Due to the dynamics in the network traffic, this causes internal nodes that have already been learned to be replaced by new previously unknown internal nodes.

See also

Configuring internal subnets manually (Page 879)

Configuring IP network nodes manually

Meaning

As an alternative to the learning mode that you enable using the "Enable learning of internal nodes" check box and that allows the security module to learn the internal network nodes dynamically, you can enter the network nodes to be learned manually in the "Internal IP nodes" entry and in doing so enable them for VPN tunnel communication. The MAC address of a network node can be specified as an option.

Requirement

- The security module is in bridge mode.
- The security module is a member of a VPN group.

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "VPN" > "Nodes" > "Internal IP nodes".

Configuring MAC network nodes manually

Meaning

As an alternative to the learning mode that you enable using the "Enable learning of internal nodes" check box and that allows the security module to learn the internal network nodes dynamically, you can enter the network nodes to be learned manually in the "Internal MAC nodes" entry and in doing so enable them for VPN tunnel communication.

Requirement

- The security module is in bridge mode.
- The security module is a member of a VPN group.

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "VPN" > "Nodes" > "Internal MAC nodes".

Configuring internal subnets manually

Requirement

- The security module is a member of a VPN group.

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "VPN" > "Nodes" > "Internal subnets".

Security module in bridge mode - "Internal subnets" entry

To be able to enable internal subnets for VPN tunnel communication manually, you need to enter the following address parameters:

Parameter	Function	Example of a value
Network ID	Network ID of the subnet to be enabled for VPN tunnel communication. Based on the network ID, the router recognizes whether a destination address is inside or outside the subnet. Must not be located in the same subnet as the IP address of the security module.	192.168.11.0
Subnet mask	The subnet mask determines the network structure. Based on the network ID, the router recognizes whether a destination address is inside or outside the subnet.	255.255.255.0
Router IP address	IP address of the router via which the subnet you are allowing is reached. Must be located in the same subnet as the IP address of the security module.	192.168.10.2

Security module in routing mode - "Subnets reachable through tunnel" entry

In routing mode, entire subnets are always tunneled. To be able to enable internal subnets reachable via routers, the external subnet or the DMZ subnet for VPN tunnel communication manually, you need to enter the following address parameters:

Parameter	Function	Example of a value
Network ID	Network ID of the subnet to be enabled for VPN tunnel communication. Based on the network ID, the router recognizes whether a destination address is inside or outside the subnet.	192.168.11.0
Subnet mask	The subnet mask determines the network structure. Based on the network ID, the router recognizes whether a destination address is inside or outside the subnet.	255.255.255.0
Comment	Entry of additional, optional comments.	

Router and firewall redundancy

Overview

Meaning

Failures of the security modules SCALANCE S623 as of V4 and SCALANCE S627-2M as of V4 can be automatically compensated by routers and firewall redundancy during operation. To do this, group two security modules of the type SCALANCE S623 or SCALANCE S627-2M in a redundancy relationship by activating routers and firewall redundancy for both security modules. Following this, you decide which security module of the redundancy relationship is passive in normal mode (secondary module). You make this setting for the security module of the redundancy relationship that is active in normal operation (primary module). If the primary module fails during operation, the secondary module automatically adopts the function as firewall and (NAT/NAPT) router. To ensure the identical configuration of both security modules, these are connected together via their DMZ interfaces and their configurations are synchronized during operation. In this case, the DMZ interfaces of the security modules involved cannot be used for other purposes.

Address redundancy

In addition to their module IP addresses, the two security modules share a common IP address on the external and on the internal interface so that if one of the security modules fails, the IP addresses do not need to be changed. To do this, you need to configure an IP address for the external and for the internal interface of the redundancy relationship.

Effects of redundancy relationships on security modules

When you create redundancy relationships between security modules, some properties of these security modules are automatically adapted to establish compatibility with the redundancy relationship. The following properties are affected by this adaptation:

Module property	Effect on the module property
Operating mode	Where necessary, the mode is changed to the "Routing mode" option.
Membership of VPN groups	Where necessary, the security module is removed from VPN groups.
Interface configuration	Where necessary, the external interface and the DMZ interface of the security module are enabled. Where necessary, the IP assignment method "Static address" is configured for all interfaces.

Configuration of security modules in redundancy relationships

After router and firewall redundancy have been enabled and the primary module of the redundancy relationship selected, some of the module properties are configured only using the primary module. The properties configured for the primary module then apply to the redundancy relationship and cannot be configured for the secondary module. The following properties can be configured for the redundancy relationship:

- Basic settings of the redundancy relationship (secondary module, network parameters)
- Firewall (standard rules for IP services are configured separately for the individual security modules).
- Routing
- NAT/NAPT routing (not 1:1 NAT)

The values of the properties listed above are initially adopted from the primary module for the redundancy relationship.

The settings listed below remain active for the individual security modules even after including them in a redundancy relationship. Configuring these properties for the primary module therefore has no effect on the secondary module.

- Interface configuration (disabling interfaces and changing the VIP assignment method "Static address" is not possible).
- Standard rules for IP services (firewall)
- DDNS
- Time-of-day synchronization
- Log settings
- SNMP
- RADIUS

Note

Loading a configuration on security modules of a redundancy relationship (only SCALANCE S623/S627-2M as of V4)

The properties of a redundancy relationship configured for the primary module must be loaded both on the primary module and on the secondary module. To load the configuration, the physical IP address via which your engineering station can reach the security module must be used. The virtual IP addresses of the redundancy relationship cannot be used for loading.

Note

Configuring routing when using routers and firewall redundaancy

In a redundancy relationship the only routing information that is synchronized between the primary and secondary module is that that was configured statically in the local security settings of the primary module in the “Routing” entry. Routing entries resulting dynamically due to using standard routers are not synchronized. When using routers and firewall redundancy it is therefore recommended that you configure all known routers statically.

Creating redundancy relationships between security modules

Requirement

The security modules SCALANCE S623/S627-2M as of V4 are not assigned to any other redundancy relationship.

Procedure

1. Select the security module that will be the active security module in normal operation (primary module).
2. In the local security settings, select the entry "Router and firewall redundancy".
3. Select the "Router and firewall redundancy" check box.
4. In the "Secondary module" drop-down list, select the security module that will be the passive security module in normal operation.

Result: You have created a redundancy relationship between the security modules.

Configuring redundancy relationships

How to access this function

1. Select the primary module of the redundancy relationship.
2. In the local security settings, select the entry "Router and firewall redundancy".

Configuring network parameters of the redundancy relationship

Configurable parameter	Meaning
IP address	IP address of the virtual external or internal interface of the redundancy relationship. The IP address must be located in the external or internal subnet of the primary module.
Subnet mask	Subnet mask of the virtual external or internal interface of the redundancy relationship
MAC address (adaptable only for SCALANCE S623/ S627-2M as of V4.0.1)	MAC address of the virtual external or internal interface of the redundancy relationship

For general information on configuring network parameters, refer to the following section:
Configuring IP address parameters (Page 842)

Configuring the firewall

IP packet filter rules for redundancy relationships are configured on the primary module. The communications directions "From external to internal" and "From internal to external" are available.

For general information on configuring IP packet filter rules in advanced firewall mode, refer to the following section:
Defining IP packet filter rules (Page 771)

Configuring address translation with NAT/NAPT

Address translation with NAT/NAPT for the redundancy relationship is configured on the primary module. For redundancy relationships, only Source NAT and NAPT can be configured. With Source NAT, source IP addresses in the internal subnet can only be replaced with the virtual external IP address of the redundancy relationship. No alias IP addresses can be registered on the external interface of the redundancy relationship. With NAPT, only the "External to internal" address translation direction can be configured.

For general information on configuring address translations with NAT/NAPT, refer to the following section:
Overview of NAT/NAPT (Page 784)

Configuring routing

Routes for the redundancy relationship are configured on the primary module. Standard routers must be specified in the "External interface [P1] red" or "Internal interface [P2] green" entry and must be identical per interface.

For general information on configuring routing, refer to the following section:
Specifying routes (Page 869)

Online functions - diagnostics and logging

Overview of the individual interfaces - "Interface settings" entry

Meaning

Table 10-56 Online diagnostics: "Interface settings" entry

System and status functions	Meaning
Interfaces	Table above: General overview of the interfaces of the security module. Table below: Information on the interface operated via PPPoE. • Status: Indicates whether or not a connection was established to the Internet Service Provider (ISP). • Current IP address: Current IP address of the interface • Gateway: IP address of the gateway • Primary dynamic DNS service: IP address of the primary dynamic DNS service • Secondary dynamic DNS service: IP address of the secondary dynamic DNS service • Error code (numeric): Error information if no connection could be established to the ISP.
CARP (only SCALANCE S623/S627-2M as of V4)	• CARP interface: Display of the virtual CARP interface • Physical interface: Physical interface on which the virtual CARP interface is operated (external / internal). • Status: Displays which of the modules of the redundancy relationship is active. • MAC address: MAC address of the virtual CARP interface • Preferred: Displays which of the modules of the redundancy relationship is configured as the primary module.

System and status functions	Meaning
Media redundancy (SCALANCE S627-2M only)	• Interface: Interface connected to the MRP ring. • Protocol: Protocol used (MRP) • Ring port 1: Name of the first media module port of the interface connected to the MRP ring. • Ring port 2: Name of the second media module port of the interface connected to the MRP ring. • Domain name: Name of the MRP domain. • Discrepancy: Display whether the domain of the client differs from that of the redundancy manager. • Domain UUID: UUID of the MRP domain.
Media modules (SCALANCE S627-2M only)	• Port: Port ID(s) of the media module port(s) • Name: Name of the media module • MLFB: MLFB of the media module • Revision: Version of the media module • Discrepancy: Displays whether there are differences between the configuration data in STEP 7 and the media modules.

Overview of the Dyn. DNS settings - "Dynamic DNS" entry

Meaning

Table 10-57 Online diagnostics: "Dynamic DNS"

System and status functions	Meaning
Client status	Indicates whether or not a connection was established to a dyn. DNS server.
Current IP address	WAN IP address via which the security module can currently be reached.
Defined IP address	IP address currently assigned to the FQDN.
Current time	The current time-of-day.
Force update	The security module obtains the current IP address of its Internet access and sends an update query to the configured update server(s). This publishes the current IP address on the Internet. The status is displayed in the primary or secondary dynamic DNS service fields. This means, for example that it is possible to check whether configured data such as the user name and password of the dynamic DNS account are correct.
Cancel suspension	Cancels the suspension (blocking of IP address updates from the dynamic DNS provider on the security module), for example after the dynamic DNS provider password has been corrected or an error has been eliminated.
Primary and secondary dynamic DNS service	
FQDN	Fully Qualified Domain Name registered with the provider.
Server IP Address	IP address of the update server used.
Successful update	Successful update in the dynamic DNS update service.
Last update attempt	Last update attempt for DynDNS update service.

System and status functions	Meaning
Last failed update	Last update error in the dynamic DNS update service.
Error code	Error status of the last unsuccessful update attempt during a dynamic DNS update.

What is the meaning of the messages?

The messages of the last DDNS update attempt have the following meaning:

Message	Meaning
Success	
DDNS_OK	The update query was successful.
Connection-related status messages	
DDNS_E_CON_UDP_SRV_RESOLV_ERR	DNS name of the update server unknown, FQDN cannot be resolved using the known DNS server.
DDNS_E_CON_UDP_SRV_UNREACHABLE	Update server unreachable ("Timeout").
Security-related status messages (errors)	
DDNS_E_CERT_SUBJECT_INVALID	The common name of the subject in the certificate does not match the domain name of the update server or its IP address.
DDNS_E_CERT_UNABLE_TO_GET_ISSUER_CERT	Issuer certificate not found. The certificate chain could not be followed back to the root CA because an issuer certificate was not found. The trust chain is incomplete.
DDNS_E_CERT_SIGNATURE_INVALID	The signature of a certificate could not be read or is invalid.
DDNS_E_CERT_NO_TRUST	A certificate in the trust chain is invalid, in other words: • Not yet valid or ready expired • V3 extensions invalid • Critical V3 extension is not supported
DDNS_E_CERT_DEPTH_ZERO_SELFSIGNED_CERT	The update server has supplied a self-signed certificate and the certificate is not in the certificate store for trustworthy root CA certificates.
DDNS_E_CERT_SELF_SIGNED_CERT_IN_CHAIN	The certificate chain could be established using non trustworthy certificates but no suitable root CA certificate was found in the certificate store for trustworthy certificates.
DDNS_E_CERT_CHAIN_TOO_LONG	The certificate chain exceeds the maximum supported verification depth.
DDNS_E_CERT_INVALID_CA	A CA certificate is invalid, in other words expired, not yet valid or the V3 extensions are not suitable for the intended purpose (for example CA not set to TRUE for CA certificates).
DDNS_E_CERT_KEYUSAGE_UNSUITED	The V3 extensions key usage or extended key usage set in a certificate of the trust chain are not suitable for the use of the certificate.
DDNS_E_CERT_EXTENSION_UNSUPPORTED	A certificate in the trust chain used an extension marked as critical that is not supported.

Message	Meaning
Agent-related status messages (errors)	
DDNS_E_AGT_BAD_AGENT	• The update request does not correspond to the structure required by cRSP, for example URL parameters missing. • The update request was sent to an illegal URL on the update server. • The configured update string contains errors.

Display of the ARP table - "ARP table" entry

Meaning

Display of the ARP table of the security module.

Table 10-58 Online diagnostics: "ARP table" tab

System and status functions	Meaning
ARP table	Display of the static (proxy ARP) and dynamic entries of the ARP table on the security module. The "Publication type" tab specifies whether the entry is configured statically or learnt.

Users logged in to the Web page - "Logged in users" entry

Meaning

Shows the users logged in to the Internet page for user-specific IP rule sets.

System and status functions	Meaning
User name	Name of the logged on user.
Source IP address	IP address with which the user logged on.
Time remaining	Remaining time before the user is automatically logged off.
Maximum time of the session	Configured total time of the session.
Log off	The selected user is logged off.

Display of the firewall blacklist - "Firewall blacklist" entry

Meaning

Displays the IP addresses of nodes that have exceeded the permitted number of connections and firewall statuses per time unit. These nodes are entered in the IP blacklist of the firewall.

The number of connections and firewall statuses per time unit is only limited if the "Use extended status options" check box is selected in the "Standard rules for IP services" entry in the local security settings.

If you click the "Delete all" button, the displayed IP addresses are removed from the firewall blacklist of the security module. The IP addresses are also no longer displayed.

Setting the date and time - "Date and time" entry

How to access this function

1. Select the security module whose time and date you want to check or set.
2. Select the "Online & diagnostics" command from the shortcut menu.
3. In online diagnostics, select the "Functions" > "Date and time" entry.

Setting the local time on the security module

In this area, you can read out and set the time and date of the security module. When you click the "Apply" button, the security module is assigned the time and date currently entered in the "Date" and "Time" input boxes.

Setting the local time on PC

This area shows the current time and current date of the PC on which STEP 7 is installed. If you click the "Adopt for module" button, the security module is assigned the current time and current date of the PC.

Diagnostics in ghost mode - "Ghost mode" entry

Module-specific function

This function is available only for SCALANCE S602 as of V3.1.

Meaning

Display of address information as well as information on IP address changes of the internal node.

System and status functions	Meaning
Status of SCALANCE S602	Display of the status of the security module in relation to operation in ghost mode.
IP address	IP address of the internal node (identical to the external IP address of the security module).
Subnet mask	Subnet mask of the security module.
MAC address	MAC address of the internal node.

System and status functions	Meaning
Node found on	Displays when the internal node was recognized by the security module or when an IP address change was made on the internal node.
Number of IP address changes	Number of IP address changes detected by the security module.
IP address	IP address in the external network for which the security module requires route information.
Standard router	Standard router for the IP address in the external network.

Download functions

Downloading a configuration

Downloading the configuration / establishing an online connection

1. From the "PG/PC interface" drop-down list of the "Advanced download" dialog or "Connect online", select the network adapter via which you can reach the module.
2. If the module is in the factory settings status, follow the steps below:
   - From the "Connection to interface/subnet" drop-down list, with which your engineering station is connected and for which the IP address to be assigned is configured in the local security settings.
   - Select the "Display all compatible nodes" from the drop-down list.
   - Click the "Start search" button.
   - Result: The module is displayed in the "Select target device" table with its detected MAC address.
   - Select the module entry in the table and click the "Assign IP address" button.
   - Result: The module is assigned the IP address configured in the local security settings for the selected interface.
3. If the module is not in the factory settings status, follow the steps below:
   - From the "Connection with interface/subnet", select the interface / the FQDN address / WAN address of the module via which your engineering station can reach the module. STEP 7 then uses the address configured in the local security settings for the selected component for access to the module.
   - If you select the option "Display devices with the same address" from the drop down list, the nodes will be displayed whose address matches the selected component. If you select the entry "Show all compatible devices", the devices whose module type matches that of the selected module are shown.
   - Click the "Start search" button.
   - Result: The "Address" column of the "Compatible devices in the target subnet" displays the detected IP address / FQDN address of the module.
   - Select the address entry in the table and click the "Load" or "Go online" button.

Ideally, you should configure the modules of a group via the common external network of these modules (interface X1). If the engineering station is located in an internal network, you will need to enable the IP addresses of the other modules of the group explicitly in the firewall of this SCALANCE S and configure this module first.

Note

Downloading the configuration when operating in ghost mode (only SCALANCE S602 as of V3.1)

When you operate the security module in ghost mode, the external interface of the security module takes over the IP address of the internal node at runtime. Before you can download a new configuration via the external interface to the security module, you need to specify the IP address for downloading a configuration that the security module obtained from the internal node during runtime.
To find out the current IP address of the security module, you can search for reachable nodes in STEP 7 with the menu command "Online" > "Accessible devices".

Note

Loading a configuration on security modules of a redundancy relationship (only SCALANCE S623/S627-2M as of V4)

The properties of a redundancy relationship configured for the primary module must be loaded both on the primary module and on the secondary module. To load the configuration, the physical IP address via which your engineering station can reach the security module must be used. The virtual IP addresses of the redundancy relationship cannot be used for loading.

Specifying a different address

In the "Compatible devices in target subnet" dialog area, you have the option of specifying an IP address / an FQDN address that differs from the IP address / FQDN address in the local security settings. To do this, enter the IP address / FQDN address of the module in the editable cell in the "Address" table column. After loading, the security module is reachable via the IP address / the FQDN address from the local security settings.

Firmware version

The configuration of a SCALANCE S module can also be downloaded to a SCALANCE S module whose firmware version is higher than the firmware version of the SCALANCE S module in STEP 7.

Operating mode

Configurations can be downloaded while the SCALANCE S modules are operating. Restart the SCALANCE S module to activate your configuration changes.

Note

Special characteristics

- As long as a module has not yet set IP parameters (in other words, prior to the first configuration), there must be no router between the module and the configuration computer.
- If you swap a PC from the internal to the external interface of the SCALANCE S, access from this PC to the SCALANCE S is blocked for approximately 20 minutes.

Configuration status

Prior to each download, the existing configuration on the security module is checked and compared with the configuration to be downloaded from the STEP 7 project. If the configuration of the module originates from the STEP 7 project currently to be downloaded and there are differences between these configurations, it is possible to download only files with differences between the module and project configuration to the security module. In some situations, this can speed up the download.

Transferring firmware

This needs to be taken into consideration before transferring new firmware

To transfer new firmware to a security module, the following conditions must be met:

- You have the rights required to transfer firmware; refer to section AUTOHOTSPOT.
- The security module is configured with an IP address.

The transfer is secure

The firmware is transferred over a secure connection and can therefore also be transferred from the unprotected network.

The firmware itself is signed and encrypted. This ensures that only authentic firmware can be downloaded to the SCALANCE S module.

Restart necessary following transfer

Newly downloaded firmware only becomes active after the SCALANCE S module has been restarted. If the transfer is disturbed and aborted, the module starts up again with the old firmware version.

Security for S7-300 /S7-400 / PC CPs

Setting up a firewall

Local firewall rules for S7-300 /S7-400 / PC CPs

Overview S7-300 CPs / S7-400 CPs /PC CPs

Enabling packet filter rules

If you enable the security function for the CPs in the local security settings, initially all access to and via the CP is permitted. To enable individual packet filter rules, select the "Activate firewall" check box. Then enable the required services. Firewall rules created automatically due to a connection configuration have priority over rules set manually.

Note

Detailed firewall settings in advanced firewall mode

In advanced firewall mode, you can restrict firewall rules to individual nodes. To change to advanced firewall mode, select the "Activate firewall in advanced mode" check box.

Firewall configuration with VPN

If the security module is added to a VPN group, the firewall is enabled by default. In addition, the "Tunnel communication only" check box is enabled. This means that only encrypted IPsec data transfer is permitted via the external interface. External data traffic is blocked.

If you deselect the check box, tunneled communication and also the types of communication selected in the other boxes are permitted.

Updating connection rules

Changes to the connection configuration of CPs also change the connection-related firewall rules. To display the modified firewall rules, click the "Update connection rules" button. The modified firewall rules are then displayed in advanced firewall mode.

Configuring a firewall with predefined firewall rules - CP 343-1 Adv. / 443-1 Adv.

Configuring a firewall with predefined IP rules - CP 343-1 Adv. / 443-1 Adv.

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "Firewall" > "Predefined IP rules".

Table 10-59 Available services and directions

Service	From station/ internal to external	External to internal	From external to station	Enabled ports	Meaning
Allow IP communication	x	x	x	All	IP traffic for the selected communication directions is allowed.
Allow S7 protocol	x	x	x	TCP port 102	Communication of the nodes using the S7 protocol is allowed.
Allow FTP/ FTPS (explicit mode)	x	x	x	TCP port 20 TCP port 21	For file management and file access between server and client.
Allow HTTP	x	x	x	TCP port 80	For communication with a Web server.
Allow HTTPS	x	x	x	TCP port 443	For secure communication with a Web server, for example, for Web diagnostics.
Allow DNS	x	x	-	TCP port 53 UDP port 53	Communication connection to a DNS server is allowed.
Allow SNMP	x	x	x	TCP port 161/162 UDP port 161/162	For monitoring nodes capable of SNMP.
Allow SMTP	x	x	-	TCP port 25	For sending e-mails via an SMTP server.
Allow NTP	x	x	-	UDP port 123	For synchronization of the time of day.

Table 10-60 Logging

Option	Action when activated	Relevant firewall rule		
IP log settings		**Action**	**From**	**To**
Log tunneled packets	Only active if the security module is a member of a VPN group. All IP packets transferred via the tunnel are logged.	Allow	Station	Tunnel
		Allow	Tunnel	Station
Log blocked incoming packets	All incoming IP packets that are discarded are logged.	Drop	External	Station

Note

Relationship between log settings in default mode and firewall rules

Log settings that are made in "Predefined IP rules" and "Predefined MAC rules" have no effect on firewall rules that were automatically created as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

Configuring a firewall with predefined MAC rules - CP 343-1 Adv. / 443-1 Adv.

How to access this function

1. Select the module to be edited.
2. Select the entry "Firewall" > "Predefined MAC rules".

Table 10-61 Available services and directions

Service	From station to external	From external to station	Meaning
Allow MAC communication	x	x	The MAC traffic from station to external and vice versa is allowed.
Allow ISO protocol	x	x	The ISO traffic from station to external and vice versa is allowed.

Table 10-62 Logging

Option	Action when activated	Relevant firewall rule		
MAC log settings		Action	From	To
Log tunneled packets	Only active if the security module is a member of a VPN group. All MAC packets transferred via the tunnel are logged.	Allow	Station	Tunnel
		Allow	Tunnel	Station
Log blocked incoming packets	All incoming MAC packets that are discarded are logged.	Drop	External	Station
Log blocked outgoing packets	All outgoing MAC packets that are discarded are logged.	Drop	Station	External

Note

Relationship between log settings in default mode and firewall rules

Log settings that are made in "Predefined IP rules" and "Predefined MAC rules" have no effect on firewall rules that were automatically created as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

Configuring a firewall with predefined firewall rules - CP 1628

Configuring a firewall with predefined IP rules - CP 1628

How to access this function

1. Select the module to be edited.
2. Select the "Security" > "Firewall" > "Predefined IP rules" entry.

Table 10-63 Available services and directions

Service	From external to station	Enabled ports	Meaning
Allow IP communication	x	All	IP traffic from external to station is allowed.
Allow S7 protocol	x	TCP port 102	Communication of the nodes using the S7 protocol is allowed.
Allow FTP/FTPS (explicit mode)	x	TCP port 20 TCP port 21	For file management and file access between server and client.
Allow HTTP	x	TCP port 80	For communication with a Web server.
Allow HTTPS	x	TCP port 443	For secure communication with a Web server, for example, for Web diagnostics.
Allow DNS	x	TCP port 53 UDP port 53	Communication connection to a DNS server is allowed.
Allow SNMP	x	TCP port 161/162 UDP port 161/162	For monitoring nodes capable of SNMP.
Allow SMTP	x	TCP port 25	For sending e-mails via an SMTP server.
Allow NTP	x	UDP port 123	For synchronization of the time of day.

Table 10-64 Logging

Option	Action when activated	Relevant firewall rule		
IP log settings		**Action**	**From**	**To**
Log tunneled packets	Only active if the security module is a member of a VPN group. All IP packets transferred via the tunnel are logged.	Allow	Station	Tunnel
		Allow	Tunnel	Station
Log blocked incoming packets	All incoming IP packets that are discarded are logged.	Drop	External	Station

Note

Relationship between log settings in default mode and firewall rules

Log settings that are made in "Predefined IP rules" and "Predefined MAC rules" have no effect on firewall rules that were automatically created as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

Configuring a firewall with predefined MAC rules - CP 1628

How to access this function

1. Select the module to be edited.
2. Select the entry "Security" > "Firewall" > "MAC rules".

Table 10-65 Available services and directions

Service	From station to external	From external to station	Meaning
Allow MAC level communication	x	x	The MAC traffic from external to the station and vice versa is allowed.
Allow ISO communication	x	x	ISO traffic from external to the station and vice versa is allowed.
Allow SiClock	x	x	SiClock time-of-day frames from external to the station and vice versa are allowed.

Table 10-66 Logging

Option	Action when activated	Relevant firewall rule		
MAC log settings		**Action**	**From**	**To**
Log tunneled packets	Only active if the security module is a member of a VPN group. All MAC packets transferred via the tunnel are logged.	Allow	Station	Tunnel
		Allow	Tunnel	Station
Log blocked incoming packets	All incoming MAC packets that are discarded are logged.	Drop	External	Station
Log blocked outgoing packets	All outgoing MAC packets that are discarded are logged.	Drop	Station	External

Note

Relationship between log settings in default mode and firewall rules

Log settings that are made in "Predefined IP rules" and "Predefined MAC rules" have no effect on firewall rules that were automatically created as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

IP packet filter directions S7-300-/S7-400-/PC-CPs

Meaning

Possible selections for the communication directions "From" and "To" in the IP rules of the advanced firewall mode.

The following directions are available

Available options / ranges of values		Security module		Meaning
From	To	CP 343-1 Adv. / 443-1 Adv.	CP 1628	
Internal	Station	x	-	Access from the internal network to the station.
	Any	x	-	Access from internal to the external network, VPN tunnel partner and the station.
External	Station	x	x	Access from the external network to the station.
	Any	x	-	Access from external to the internal network and the station.
Station	Internal	x	-	Access from the station to the internal network.
	External	x	x	Access from the station to the external network.
	Tunnel	x	x	Access from the station to the VPN tunnel partner.
Tunnel	Station	x	x	Access via the VPN tunnel partner to the station.
	Any	x	-	Access from VPN tunnel partners to the internal network and the station.
Any	External	x	-	Access from the internal network and the station to the external network.

MAC packet filter directions S7-300-/S7-400-/PC-CPs

Meaning

Possible selections for the communication directions "From" and "To" in the MAC rules of the advanced firewall mode.

The following directions are available

Available options / ranges of values		Security module		Meaning
From	**To**	**CP 343-1 Adv. / 443-1 Adv.**	**CP 1628**	
External	Station	**x**	**x**	Access from the external network to the station.
Station	External	**x**	**x**	Access from the station to the external network.
	Tunnel	**x**	**x**	Access from the station to the VPN tunnel partner.
Tunnel	Station	**x**	**x**	Access via the VPN tunnel partner to the station.

Configuring the access list

Module-specific function

This function is not available for the CP 1628.

Meaning

You set access protection for certain IP addresses using the IP access lists. List entries that have already been created and the corresponding rights are displayed in the local settings of the CP in the entry "Firewall" > "IP rules" (advanced firewall mode).

Note

Changed behavior after activation of security

- Once you have activated the security function for a CP, access protection will only apply to the external interface. You can apply access protection to the internal interface as well, by configuring suitable firewall rules in advanced firewall mode.
- The CP also responds to ARP requests from IP addresses that have not been released (layer 2).
- If the IP access list of a CP contains no entries and you activate security for the CP, the firewall will be activated and prevent access to the CP from external locations. Configure the corresponding firewall rules in advanced firewall mode so that the CP can be reached.

Effect of IP access list entries at activation of security

If security is enabled in the local settings of a CP, the corresponding rules are created in the advanced firewall mode. A firewall rule "Allow" > "External" > "Station" is created for an IP address you specified in the address list. The IP address from the IP access list is used accordingly as source IP address. IP addresses from a defined IP address range are also integrated into corresponding firewall rules.

Requirements for editing

Before you can edit created firewall rules, the following condition must be met:

- for editing using STEP 7: "Configure security" configuration right
- for editing using a Web server: Module right "Web: Expand IP access control list"

The requirements for editing the IP access lists outside the local security settings are described in the sections on the specific CPs.

Connection-related automatic firewall rules

Meaning

For connections that were configured using CPs, STEP 7 automatically creates firewall rules that allow communication with the partner of the CP in the specified direction (CP active/ passive). The connection establishment directions are taken into account. To display these firewall rules, if the advanced firewall mode is enabled, the "Update connection rules" button needs to be clicked. The firewall rules are then displayed in advanced firewall mode.

Note

Releasing UDP multicast connections manually

No automatic firewall rules are created for UDP multicast connections. To enable the connections, add the relevant firewall rules manually in advanced firewall mode.

Depending on how the connection establishment is configured, the following level 3 firewall rules are created. If the security module is in a VPN group, the direction "External" changes to "Tunnel". This applies only to CPs that support VPN.

The IP address of the connection partner is entered in the "Source IP address" or "Destination IP address" column of these firewall rules.

CP->external	Action	From	To
active	Allow	Station	External
	Drop	External	Station
passive	Drop	Station	External
	Allow	External	Station
active and passive	Allow	External	Station
	Allow	Station	External

CP->internal	Action	From	To
active	Allow	Station	Internal
	Drop	Internal	Station
passive	Drop	Station	Internal
	Allow	Internal	Station
active and passive	Allow	Internal	Station
	Allow	Station	Internal

For level 2 connections, "Allow" rules are created for both directions. If the security module is in a VPN group, the direction "External" changes to "Tunnel".

The MAC address of the connection partner is entered in the "Source MAC address" or "Destination MAC address" column of these firewall rules.

CP->external	Action	From	To
active, passive, active and passive	Allow	Station	External
	Allow	External	Station

Note

Changing the connection configuration

Changes to the connection configuration of CPs also change the connection-related firewall rules. To display the modified firewall rules, click the "Update connection rules" button.

Conventions for automatically created firewall rules

- Priority
  The rules have highest priority and are therefore inserted at the top in the local rule list.
- Deleting rules
  The rules cannot be deleted. Logging can be enabled and services can be assigned. Moreover, you may insert a bandwidth and a comment.
- Changing the action
  If you set the action from "Allow" to "Drop" or vice versa, this is overwritten again during renewed system synchronization. Select "Allow*" or "Drop*" to retain your changes. In this case, only the IP address is synchronized and the action and direction remain as set. Settings for logging, service, bandwidth and comment are also retained after a renewed system synchronization even without changing the action to "Allow*" or "Drop*". If the configured connection is deleted, the corresponding rules are removed from the list.

Security module in VPN group

As default, the "Tunnel communication only" check box is enabled. If you deselect the check box, in addition to tunnel communication between tunnel partners, communication is also possible with network nodes to which there is no tunnel.

- Communication is untunneled if the partner address belongs to a station known in STEP 7 for which no VPN tunnel is configured.
- Communication is through the tunnel if the partner address is a VPN endpoint.
- If it is not clear whether connection should bypass or run through the VPN tunnel, the connection is assigned to the VPN tunnel and a message to this effect is displayed. The assignment can be adapted in advanced firewall mode, for example, by changing the "From" direction "Tunnel" to "External". To avoid this adaptation being overwritten by the next system synchronization, the "Allow*" or "Drop*" action must be selected.

Note

If you want to ensure that only communication through the tunnel is possible, you will need to create suitable firewall rules in advanced firewall mode, for example, for internal nodes or NDIS addresses.

To allow only tunneled communication for a CP, add a rule with the following settings:

- "Action": "Drop"
- "From": "Any"
- "To": "External"

For the CP 1628, add a rule with the following settings:

- "Action": "Drop"
- "From": "Station"
- "To": "External"

In addition to this, you need to remove existing firewall rules that allow untunneled communication.

Activating Web server on security module

Module-specific function

This function is only available for CP 343-1 Adv. / 443-1 Adv. and CP 443-1 OPC UA:

Meaning

After activating the Web server, you have access to the Web pages of the module. In the local security settings, you can restrict access to these Web pages for the CP 343-1 Adv. / 443-1 Adv.to the HTTPS protocol. This access is controlled using the "Allow access only using HTTPS" check box. In addition, you must configure the firewall accordingly. For the CP 443-1 OPC UA access to the Web server is only permitted using HTTPS.

IPsec tunnel: Creating and assigning groups

Configuring internal network nodes - "Nodes" entry

Allow access to S7-300 / S7-400 CPs for VPN connection partners

Possible selections

Decide whether or not the VPN connection partner can have access to the CP and/or the internal subnet of the CP in routing mode (SCALANCE S / M).

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "Node".
3. Configure access for the VPN connection partner of the CP in routing mode (SCALANCE S / M):
   - Allow connection to the CP (Gbit interface)
   - Allow connection to the internal subnet (PROFINET subnet)

Configuring NDIS nodes manually for PC CPs that can be reached through the tunnel

Configuring NDIS nodes that can be reached through the tunnel

The internal nodes are learned and assigned to the routes dynamically. This concerns the NDIS IP addresses of the Windows PC.

Follow the steps below

1. Select the module to be edited.
2. In the local security settings, select the entry "Nodes" > "NDIS nodes reachable via tunnel".
3. Enter the NDIS IP addresses.

Online functions - Debug / Diagnostics and Logging

Updated firewall rules - "Dynamically updated firewall rules" entry

Module-specific function

This function is only available for CP 343-1 Adv. / 443-1 Adv.

Meaning

Display of the IP addresses or IP address ranges that were enabled dynamically using HTTP or HTTPS, or loaded by a user. The rights assigned for accessing the S7 CP are displayed for the enabled IP addresses. An update of the IP addresses in this tab can only be triggered by the following events:

- Extension/modification of the IP access control list
- Update of firewall rules
- Dynamic extensions transmitted to the CP at runtime, for example, PROFINET IO devices

Since only the dynamically updated firewall rules are displayed here, you also need to take into account the firewall rules that were configured offline and downloaded to the station for a full picture of the current firewall status of the module.

Security for S7-1200-/S7-1500-CPs

Setting up a firewall

Local firewall rules for S7-1200 / S7-1500 CPs

Overview of the local firewall rules for S7-1200 / S7-1500 CPs

Enabling packet filter rules

If you enable the security function for the CPs in the local security settings, initially all access to and via the CP is permitted. To enable individual packet filter rules, select the "Activate firewall" check box. Then enable the required services. Firewall rules created automatically due to a connection configuration have priority over rules set manually.

Note

Detailed firewall settings in advanced firewall mode

In advanced firewall mode, you can restrict firewall rules to individual nodes. To change to advanced firewall mode, select the "Activate firewall in advanced mode" check box.

Updating connection rules

Changes to the connection configuration of CPs also change the connection-related firewall rules. To display the modified firewall rules, click the "Update connection rules" button. The modified firewall rules are then displayed in advanced firewall mode.

Configuring a firewall with predefined firewall rules - CP 1543-1 / CP 1543SP-1

Configuring a firewall using predefined IP rules

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "Firewall" > "Predefined IP rules".

Table 10-67 Available services and directions

Service	From external to station	Enabled ports	Meaning
Allow IP communication	x	All	IP traffic from external to station is allowed.
Allow S7 protocol	x	TCP port 102	Communication of the nodes using the S7 protocol is allowed.
Allow FTP/FTPS* (explicit mode)	x	TCP port 20 TCP port 21	For file management and file access between server and client.
Allow HTTP	x	TCP port 80	For communication with a Web server.
Allow HTTPS	x	TCP port 443	For secure communication with a Web server, for example, for Web diagnostics.
Allow SNMP	x	TCP port 161/162 UDP port 161/162	For monitoring nodes capable of SNMP.
Allow security diagnostics	x	TCP port 8448	Allow security diagnostics.

*not available for CP 1543SP-1

Table 10-68 IP routing via backplane bus (only for the CP 1543-1 as of V2.1)

Service	Enabled ports	Meaning
Allow IP communication via backplane bus	All	IPv4 traffic is allowed via the backplane bus. If the option "IP routing between communications modules" is activated group, this check box is enabled and cannot be deselected. If the option is deactvated, this check box is deactivated and cannot be activated.

Table 10-69 Logging

Option	Action when activated	Relevant firewall rule		
IP log settings		Action	From	To
Log tunneled packets	Only active if the security module is a member of a VPN group. All IP packets transferred via the tunnel are logged.	Allow	Station	Tunnel
		Allow	Tunnel	Station
Log blocked incoming packets	All incoming IP packets that are discarded are logged.	Drop	External	Station

Note

Relationship between log settings in default mode and firewall rules

Log settings that are made in "Predefined IP rules" and "Predefined MAC rules" have no effect on firewall rules that were automatically created as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

Configuring a firewall using predefined IPv6 rules

Meaning

With the predefined IPv6 rules, you have the option of configuring the firewall for services in which IPv6 is used. By enabling a predefined IPv6 rule in the local security settings of the CP 1543-1 V1.1, the system-defined ICMPv6 services that can be seen in the global security settings in the "ICMP" tab in "Firewall" > "Services" > "Define services for IP rules"are also enabled in the firewall. The firewall of the CP 1543-1 V1.0 allows ICMPv6 packets through even without enabling a predefined IPv6 rule.

How to access this function

1. Select the module to be edited.
2. Select the "Firewall" > "Pre-defined IPv6 rules" item in the local security settings.

Table 10-70 Available services and directions

Service	From external to station	Enabled ports	Meaning
Allow IP communication	x	All	IP traffic from external to station is allowed.
Allow S7 protocol	x	TCP port 102	Communication of the nodes using the S7 protocol is allowed.
Allow FTP/FTPS* (explicit mode)	x	TCP port 20 TCP port 21	For file management and file access between server and client.
Allow SNMP	x	TCP port 161/162 UDP port 161/162	For monitoring nodes capable of SNMP.

*not available for CP 1543SP-1

Table 10-71 Logging

Option	Action when activated	Relevant firewall rule		
IP log settings		Action	From	To
Log blocked incoming packets	All incoming IP packets that are discarded are logged.	Drop	External	Station

Note

Relationship between log settings in default mode and firewall rules

Log settings made in "Predefined IPv6 rules" have no effect on firewall rules that were automatically generated as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

Configuring a firewall with predefined MAC rules

How to access this function

1. Select the module to be edited.
2. Select the entry "Firewall" > "Predefined MAC rules".

Table 10-72 Available services and directions

Service	From station to external	From external to station	Enabled ports	Meaning
Allow MAC communication	x	x	-	The MAC traffic from external to the station and vice versa is allowed.
Allow ISO protocol*	x	x	-	ISO traffic from external to the station and vice versa is allowed.
Allow DCP	x	x	-	DCP traffic from external to the station and vice versa is allowed.
Allow LLDP*	x	x	-	LLDP traffic from external to the station and vice versa is allowed.

*not available for CP 1543SP-1

Table 10-73 Logging

Option	Action when activated	Relevant firewall rule		
MAC log settings		Action	From	To
Log tunneled packets	Only active if the security module is a member of a VPN group. All MAC packets transferred via the tunnel are logged.	Allow	Station	Tunnel
		Allow	Tunnel	Station

Option	Action when activated	Relevant firewall rule		
Log blocked incoming packets	All incoming MAC packets that are discarded are logged.	Drop	External	Station
Log blocked outgoing packets	All outgoing MAC packets that are discarded are logged.	Drop	Station	External

Note

Relationship between log settings in default mode and firewall rules

Log settings that are made in "Predefined IP rules" and "Predefined MAC rules" have no effect on firewall rules that were automatically created as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

Configuring a firewall with predefined firewall rules - S7-1200 CPs

Configuring a firewall using predefined IP rules

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "Firewall" > "Predefined IP rules".

Table 10-74 Available services and directions

Service	From external to station	Enabled ports	Meaning
Allow IP communication	x	All	IP traffic from external to station is allowed.
Allow S7 protocol	x	TCP port 102	Communication of the nodes using the S7 protocol is allowed.
Allow HTTP	x	TCP port 80	For communication with a Web server.
Allow HTTPS	x	TCP port 443	For secure communication with a Web server, for example, for Web diagnostics.
Allow SNMP	x	TCP port 161/162 UDP port 161/162	For monitoring nodes capable of SNMP.
Allow security diagnostics	x	TCP port 8448	Allow security diagnostics.

Table 10-75 Logging

Option	Action when activated	Relevant firewall rule		
IP log settings		**Action**	**From**	**To**
Log tunneled packets	Only active if the security module is a member of a VPN group. All IP packets transferred via the tunnel are logged.	Allow	Station	Tunnel
		Allow	Tunnel	Station
Log blocked incoming packets	All incoming IP packets that are discarded are logged.	Drop	External	Station

Note

Relationship between log settings in default mode and firewall rules

Log settings that are made in "Predefined IP rules" and "Predefined MAC rules" have no effect on firewall rules that were automatically created as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

Configuring a firewall using predefined IPv6 rules

Meaning

With the predefined IPv6 rules, you have the option of configuring the firewall for services in which IPv6 is used. By enabling a predefined IPv6 rule in the local security settings of of an S7-1200 CP capable of IPv6, the system-defined ICMPv6 services that can be seen in the global security settings in the "ICMP" tab in "Firewall" > "Services" > "Define services for IP rules" are also enabled in the firewall.

How to access this function

1. Select the module to be edited.
2. Select the "Firewall" > "Predefined IPv6 rules" item in the local security settings.

Table 10-76 Available services and directions

Service	From external to station	Enabled ports	Meaning
Allow IP communication	x	All	IP traffic from external to station is allowed.
Allow S7 protocol	x	TCP port 102	Communication of the nodes using the S7 protocol is allowed.
Allow SNMP	x	TCP port 161/162 UDP port 161/162	For monitoring nodes capable of SNMP.

Table 10-77 Logging

Option	Action when activated	Relevant firewall rule		
IP log settings		**Action**	**From**	**To**
Log blocked incoming packets	All incoming IP packets that are discarded are logged.	Drop	External	Station

Note

Relationship between log settings in default mode and firewall rules

Log settings made in "Predefined IPv6 rules" have no effect on firewall rules that were automatically generated as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

Configuring a firewall with predefined MAC rules

How to access this function

1. Select the module to be edited.
2. Select the entry "Firewall" > "Predefined MAC rules".

Table 10-78 Available services and directions

Service	From station to external	From external to station	Enabled ports	Meaning
Allow MAC communication	x	x	-	The MAC traffic from external to the station and vice versa is allowed.
Allow DCP	x	x	-	DCP traffic from external to the station and vice versa is allowed.

Table 10-79 Logging

Option	Action when activated	Relevant firewall rule		
MAC log settings		**Action**	**From**	**To**
Log tunneled packets	Only active if the security module is a member of a VPN group. All MAC packets transferred via the tunnel are logged.	Allow	Station	Tunnel
		Allow	Tunnel	Station

Option	Action when activated	Relevant firewall rule		
Log blocked incoming packets	All incoming MAC packets that are discarded are logged.	Drop	External	Station
Log blocked outgoing packets	All outgoing MAC packets that are discarded are logged.	Drop	Station	External

Note

Relationship between log settings in default mode and firewall rules

Log settings that are made in "Predefined IP rules" and "Predefined MAC rules" have no effect on firewall rules that were automatically created as a result of configuring a connection. This means, for example, that tunneled frames belonging to a configured connection cannot be logged. In advanced firewall mode, logging can be extended to the automatically generated firewall rules of connections.

IP packet filter directions S7-1200 / S7-1500 CPs

Meaning

Possible selections for the communication directions "From" and "To" in the IP rules of the advanced firewall mode.

The following directions are available

Available options / ranges of values		Meaning
From	To	
External	Station	Access from the external network to the station.
	Backplane bus*	Access from the external network to the backplane bus of the station.
	Any**	Access from the external for all supported directions.
Station	External	Access from the station to the external network.
	Tunnel	Access from the station to the VPN tunnel partner.
	Backplane bus*	Access from the station to its backplane bus.
	Any**	Access from the station to all supported directions.
Tunnel	Station	Access by VPN tunnel partners to the station.
	Backplane bus*	Access from VPN tunnel partners to the backplane bus of the station.
	Any**	Access from NPN tunnel partners to all supported directions.

Available options / ranges of values		Meaning
Backplane bus*	External	Access from backplane bus of the station to the external network.
	Tunnel	Access from backplane bus of the station to NPN tunnel partners.
	Station	Access from the backplane bus to the station.
	Any**	Access from the backplane bus to all supported directions.
Any**	External	Access from all supported directions to the external network.
	Tunnel	Access from all supported directions to VPN tunnel partners.
	Station	Access from all supported directions to the station.
	Backplane bus*	Access from all supported directions to the backplane bus of the station.

* only for CP 1543-1 as of V2.1 with IPv4 communication. if the option "IP routing between communications modules" is activated, IP routing via the backplane bus an be allowed / blocked via this direction.

* Only for CP 1543-1 as of V2.1

MAC packet filter directions S7-1200 / S7-1500 CPs

Meaning

Possible selections for the communication directions "From" and "To" in the MAC rules of the advanced firewall mode.

The following directions are available

Available options / ranges of values		Meaning
From	To	
External	Station	Access from the external network to the station.
Station	External	Access from the station to the external network.
	Tunnel	Access from the station to the VPN tunnel partner.
Tunnel	Station	Access by VPN tunnel partners to the station.

Connection-related automatic firewall rules

Meaning

For connections that were configured using CPs, STEP 7 automatically creates firewall rules that allow communication with the partner of the CP in the specified direction (CP active/ passive). The connection establishment directions are taken into account. To display these firewall rules, if the advanced firewall mode is enabled, the "Update connection rules" button needs to be clicked. The firewall rules are then displayed in advanced firewall mode.

Note

Enabling UDP multicast and UDP broadcast connections manually

No automatic firewall rules are created for UDP multicast and UDP broadcast connections. To enable the connections, add the relevant firewall rules manually in advanced firewall mode.

Depending on how the connection establishment is configured, the following level 3 firewall rules are created. If the CP is in a VPN group, the direction "External" changes to "Tunnel".

The IP address of the connection partner is entered in the "Source IP address" or "Destination IP address" column of these firewall rules.

CP->external	Action	From	To
active	Drop	External	Station
	Allow	Station	External
passive	Drop	Station	External
	Allow	External	Station
active and passive	Allow	External	Station
	Allow	Station	External

For level 2 connections, "Allow" rules are created for both directions. If the CP is in a VPN group, the direction "External" changes to "Tunnel".

The MAC address of the connection partner is entered in the "Source MAC address" or "Destination MAC address" column of these firewall rules.

CP->external	Action	From	To
active, passive, active and passive	Allow	Station	External
	Allow	External	Station

Note

Changing the connection configuration

Changes to the connection configuration of CPs also change the connection-related firewall rules. To display the modified firewall rules, click the "Update connection rules" button.

Conventions for automatically created firewall rules

- Priority
  The rules have highest priority and are therefore inserted at the top in the local rule list.
- Deleting rules
  The rules cannot be deleted. Logging can be enabled and services can be assigned. Moreover, you may insert a bandwidth and a comment.
- Changing the action
  If you set the action from "Allow" to "Drop" or vice versa, this is overwritten again during renewed system synchronization. Select "Allow*" or "Drop*" to retain your changes. In this case, only the IP address is synchronized and the action and direction remain as set. Settings for logging, service, bandwidth and comment are also retained after a renewed system synchronization even without changing the action to "Allow*" or "Drop*". If the configured connection is deleted, the corresponding rules are removed from the list.

Security module in VPN group

As default, the "Tunnel communication only" check box is enabled. If you deselect the check box, in addition to tunnel communication between tunnel partners, communication is also possible with devices to which there is no tunnel.

- Communication is untunneled if the partner address belongs to a station known in STEP 7 for which no VPN tunnel is configured.
- Communication is through the tunnel if the partner address is a VPN endpoint.

Note

If you want to ensure that only communication through the tunnel is possible, you will need to create suitable firewall rules in advanced firewall mode.

To allow only tunneled communication for a CP, add a rule with the following settings:

- "Action": "Drop"
- "From": "Station"
- "To": "External"

In addition to this, you need to remove existing firewall rules that allow untunneled communication.

Configuring subnets reachable via the backplane bus for CP 1543-1

Requirement

- The security module CP 1543-1 as of V2.1 is a member of a VPN group and the option "IP routing between communications modules" is activated

How to access this function

1. Select the module to be edited.
2. In the local security settings, select the entry "VPN" > "Nodes" " Subnets reachable through tunnel".

Configuring subnets reachable via the backplane bus

VPN communication with network nodes and subnets reachable via the backplane bus of a station is only possible if this is released for VPN communication. In the entry "Subnets reachable through tunnel" a maximum of 16 addresses or subnets can be specified. IP addresses must be specified with a 32 bit subnet mask, subnets with a permitted subnet mask in the range 1... 31 bits. Broadcast addresses must not be specified.

10.1.3.5 Secure Communication (S7-1500)

Basics of Secure Communication (S7-1500)

Basics of Secure Communication (S7-1500)

For STEP 7 (TIA Portal) as of V14 and for S7-1500 CPUs as of firmware V2.0, the options for secure communication have been broadened considerably.

"S7-1500 CPUs" also refers to CPU versions S7-1500F, S7-1500T, S7-1500C as well as S7-1500pro CPUs and ET200SP CPUs.

Introduction

The attribute "secure" is used for the identification of communication mechanisms that are based on a Public Key Infrastructure (PKI) (for example RFC 5280 for Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List Profile). A Public Key Infrastructure (PKI) is a system that can issue, distribute and check digital certificates. The digital certificates issued are used within the PKI to secure computer-based communication. If a PKI uses asymmetric key cryptography, the messages in a network can be digitally signed and encrypted.

Components that you have configured in STEP 7 (TIA Portal) for secure communication use an asymmetric key encryption scheme with a public key (Public Key) and a private key (Private Key). TLS (Transport Layer Security) is used as the encryption protocol. TLS is the successor for the SSL (Secure Sockets Layer) protocol.

Objectives of secure communication

Secure communication is used to achieve the following objectives:

- Confidentiality
  i.e. the data is secret/cannot by read by eavesdroppers.
- Integrity
  i.e. the message that reaches the recipient is the same message, unchanged, that the sender sent. The message has not been altered on the way.
- End point authentication
  i.e. the end point communication partner is exactly who it claims to be and the party who is to be reached. The identity of the partner is verified.

In the past, these objectives were primarily relevant to IT and networked computers. Today, industrial machinery and control systems with sensitive data are at equally high risk, as they are also networked, and consequently pose high security requirements for data exchange.

Protection of the automation cell by means of the cell protection concept through firewall, or via connection through VPN, for example with the security module, was common in the past and remains so.

However, it is becoming increasingly necessary to also transfer data to external computers in encrypted form via Intranet or public networks.

Common principles of secure communication

Independent of the context, secure communication is based on the Public Key Infrastructure (PKI) concept and contains the following components:

- An asymmetric encryption scheme. This scheme allows the following:
  - Encryption or decryption of the messages using public or private keys.
  - The verification of signatures in messages and certificates.
    The messages / certificates are signed by the sender / certificate owner with their private key. The recipient / verifier checks the signature with the public key of the sender / certificate owner.
- Transport and storage of the public key using X.509 certificates.
  - X.509 certificates are digitally signed data that allow public key authentication in terms of the bound identity.
  - X.509 certificates can also contain information that describes in more detail or restricts use of the public key. For example the effective date of a public key in a certificate and when it expires.
  - X.509 certificates also contain information on the issuer of the certificate in secure form.

The following paragraphs give an overview of these basic concepts which are required for managing certificates in STEP 7 (TIA Portal), for example, and for programming the communication instructions for secure Open User Communication (sOUC).

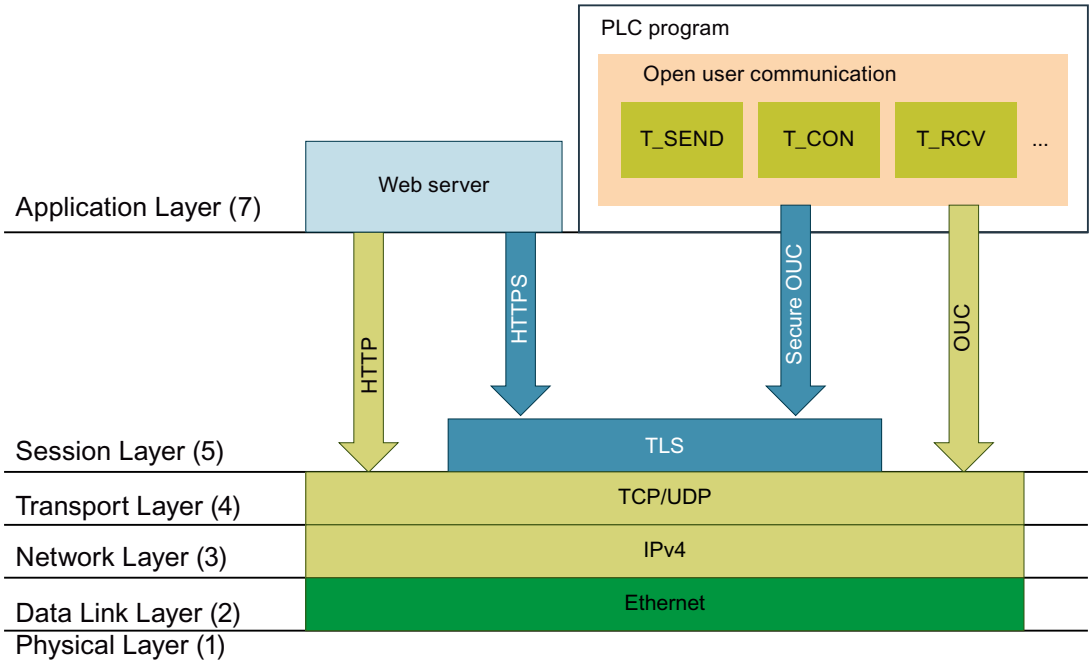
Secure communication in STEP 7

STEP 7 as of V14 provides the required PKI for the configuration and operation of secure communication.

Examples:

- The Hypertext Transfer Protokoll (HTTP) becomes Hypertext Transfer Protokoll Secure (HTTPS) by means of the TLS (Transport Layer Security) protocol. Since HTTPS is a combination of HTTP and TLS, it is called "HTTP over TLS" in the corresponding RFC. You can see in the browser that HTTPS is being used. This is indicated by the URL scheme "https://" instead of "http://" in the address bar of the browser. Most browsers highlight such secure connections.
- Open User Communication becomes secure Open User Communication. The underlying protocol is also TLS.
- E-mail providers also offer access over the "Secure SMTP over TLS" protocol to increase the security of e-mail communication.

The figure below shows the TLS protocol in the context of communication layers.



Secure communication with OPC UA

An OPC UA server is implemented in S7-1500 CPUs as of firmware V2.0. OPC UA Security also covers authentication, encryption and data integrity with digital X.509 certificates and also uses a Public Key Infrastructure (PKI). Depending on the requirements of the application you can select various security levels for the end point security. The description of the OPC UA server functionality is covered in a separate section.

Confidentiality through encryption (S7-1500)

Message encryption is an important element of data security. When encrypted messages are intercepted by third parties during communication, these potential eavesdroppers cannot access the information they contain.

There is a wide range of mathematical processes (algorithms) for encrypting messages.

All algorithms process a "key" parameter to encrypt and decrypt messages.

- Algorithm + key + message => encrypted message
- Encrypted message + key + algorithm => (decrypted) message

Symmetric encryption

The central aspect of symmetric encryption is that both communication partners use the same key for message encryption and decryption, as shown in the figure below. Bob uses the same key for encryption as Alice uses for decryption. In general, we also say that the two sides share a secret, the secret key, with which they encrypt or decrypt a message.



① Bob encrypts his message with the symmetric key
② Alice decrypts the encrypted message with the symmetric key

The process can be compared to a briefcase to which the sender and recipient have the same key, which both locks and opens the case.

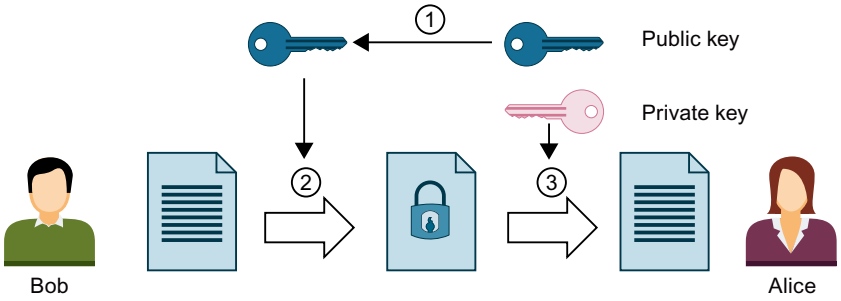
- Advantage: Symmetric encryption algorithms (for example AES, Advanced Encryption Algorithm) are fast.
- Disadvantages: How can the key be sent to a recipient without getting into the wrong hands? This is a key distribution problem. If enough messages are received, the key can also be worked out and must therefore be changed regularly.

If there is a large number of communication partners, there is also a large number of keys to distribute.

Asymmetric encryption

Asymmetric encryption works with a pair of keys consisting of one public key and one private key. In connection with a PKI it is also called a Public Key process or only PKI process: A communication partner, Alice in the figure below, has a private key and a public key. The public key is provided to the public, in other words any potential communication partner. Anyone with the public key can encrypt messages for Alice. In the figure below, this is Bob.

Alice's private key, which she must not disclose, is used by Alice to decrypt an encrypted message addressed to her.



① Alice provides Bob with her public key. No precautionary measures are required to this purpose: Anyone can use the public key for messages to Alice if they are sure that it is actually Alice's public key.

② Bob encrypts his message with Alice's public key.

③ Alice decrypts the encrypted message from Bob with her private key. As only Alice has the private key and never discloses it, only she can decrypt the message. With her private key, she can decrypt any message encrypted with her public key - not only messages from Bob.

The system can be compared to a mailbox into which anyone can put a message, but from which only the person with the key can remove messages.

- Advantages: A message encrypted with a public key can only be decrypted by the owner of the private key. As another (private) key is required for decryption, it is also much harder to work out the decryption key on the basis of large numbers of encrypted messages. This means that the public key does not have to be kept strictly confidential, unlike with symmetric keys.
  Another advantage is easier distribution of public keys. No specially secured channel is required in asymmetric cryptography to transfer the public key from the recipient to the sender encrypting the messages. Less effort is thus required in the management of the keys than with the symmetric encryption process.
- Disadvantages: Complex algorithm (e.g. RSA, named after the three mathematicians Rivest, Shamir and Adleman), and therefore poorer performance than with symmetric encryption.

Encryption processes in practice

In practice, for example with a CPU Web server and Secure Open User Communication, the TLS protocol is used below the relevant application layer. Application layers are HTTP or SMTP, for example, as detailed above.

TLS (Transport Layer Security) uses a combination of asymmetric encryption and symmetric encryption (hybrid encryption) for secure data transfer, for example, over the Internet, and uses the following subprotocols:

- TLS Handshake Protocol, responsible for authentication of communication partners and negotiation of the algorithms and keys to be used for subsequent data transfer on the basis of asymmetric encryption
- TLS Record Protocol, responsible for encryption of user data with symmetric encryption and data exchange.

Both asymmetric and symmetric encryption are considered secure encryption schemes - there is basically no difference in security between the two procedures. The degree of security depends on parameters such as the selected key length.

Improper use of encryption

You cannot tell what identity is assigned to a public key from the bit string. A fraud could provide their public key and claim to be someone else. If a third party then uses this key thinking that they are addressing their required communication partner, confidential information could end up with the fraud. The fraud then uses their private key to decrypt the message that was not intended for them, and sensitive information falls into the wrong hands.

To prevent this type of abuse, the communication partners must be confident that they are dealing with the right communication partner. This trust is established by using digital certificates in a PKI.

Authenticity and integrity through signatures (S7-1500)

Attacks from programs that intercept communication between the server and client and act as if they themselves were client or server, are called man-in-the-middle attacks. If the false identity of these programs is not detected, they can, for example, obtain important information via the S7 program or set values in the CPU and attack a machine or plants. Digital certificates are used to avoid such attacks.

Secure communication uses digital certificates that meet the standard X.509 of the International Telecommunication Union (ITU). This allows the identity of a program, a computer or an organization to be checked (authenticated).

How certificates establish trust

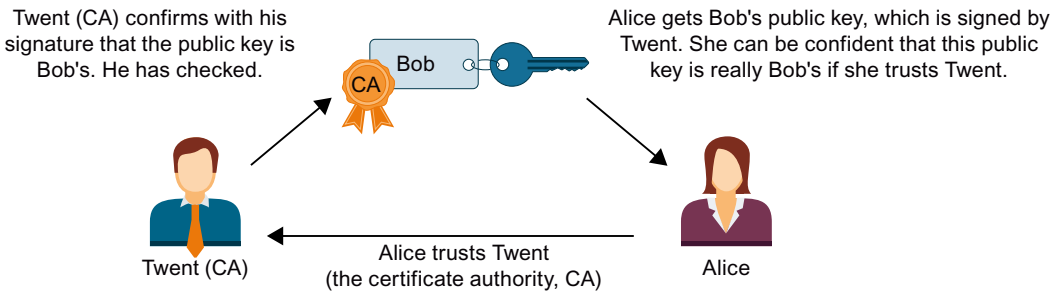
The main role of X.509 certificates is to bind an identity with the data of a certificate subject (for example e-mail address or computer name) to the public key of the identity. Identities can be people, computers or machines.

Certificates are issued by certificate authorities (Certificate Authority, CA) or by the owner of a certificate itself. PKI systems specify how users can trust the certificate authorities and the certificates that they issue.

The certificate process:

1. Anyone wishing to own a certificate submits a certificate application to a registration authority linked to the certificate authority.
2. The certificate authority assesses the application and applicant on the basis of set criteria.
3. If the identity of the applicant can be clearly established, the certificate authority confirms that identity by issuing a signed certificate. The applicant has now become the certificate holder.

The figure below is a simplified overview of the process. It does not show how Alice can check the digital signature.



Self-signed certificates

Self-signed certificates are certificates whose signature comes from the certificate owner and not from an independent certificate authority.

Examples:

- You can create and sign a certificate yourself, for example, to encrypt messages to a communication partner. In the example above, Bob (instead of Twent) could himself sign his certificate with his private key. Alice can use Bob's public key to check that the signature and public key from Bob match. This procedure is sufficient for simple internal plant communication that is to be encrypted.
- A root certificate is, for example, a self-signed certificate, signed by the certificate authority (CA), that contains the public key of the issuer.

Features of self-signed certificates

The attributes "CN" (Common Name of Subject) for the certificate holder and "Issuer" (issuer) of the self-signed certificates are identical: You have signed your certificate yourself. The field "CA" (Certificate Autority) for the certificate authority must be set to "False"; the self-signed certificate should, after all, not be used to sign other certificates.

Self-signed certificates are not embedded in a PKI hierarchy.

Certificate content

A certificate to the X.509 V3 standard, the same standard that is used by STEP 7 and the S7-1500 CPUs, consists primarily of the following elements:

- Public key
- Details of the certificate holder (meaning the holder of the key). This is, for example, the Common Name (CN) of Subject
- Attributes such as serial number and validity period
- Digital signature from the certificate authority (CA) confirming that the information is correct.

There are also extensions, for example:

- Specification of what the public key may be used for (Key Usage), for example, signing or key encryption.
  When you create a new certificate with STEP 7, for example in the context of Secure Open User Communication, select the correct entry from the list of possible usages, e.g. "TLS".
- Specification of a "Subject Alternative Name" ("SAN"), which is used in secure communication with Web servers (HTTP over TLS), for example, to ensure that the certificate in the address bar of the Web browser also belongs to the Web server specified in the URL.

How signatures are generated and verified

Asymmetric key usage ensures that certificates can be verified: The example of the certificate "MyCert" illustrates the "Sign" and "Verify signature" processes.

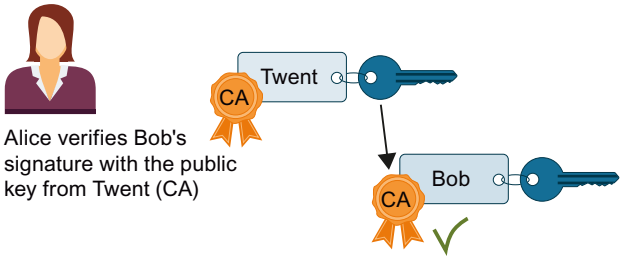
Generating a signature:

1. The issuer of the "MyCert" certificate generates a hash value from the certificate data using a specific hash function (for example SHA-1, Secure Hash Algorithm).
   The hash value is a bit string of a constant length. The advantage of the constant length of the hash value is that it always takes the same amount of time to sign.
2. Using the hash value generated in this way and the private key, the issuer of the certificate then generates a digital signature. The RSA signature scheme is often used.
3. The digital signature is saved in the certificate. The certificate is now signed.

Verifying a signature:

1. The authenticator of the "MyCert" certificate obtains the certificate of the certificate issuer and thus also the public key.
2. A new hash value is formed from the certificate data with the same hash algorithm that was used for signing (for example SHA-1).
3. This hash value is then compared with the hash value that was determined by means of the public key of the certificate issuer and the signature algorithm to verify the signature on the certificate.
4. If the signature verification supplies a positive result, both the identity of the certificate holder as well as the integrity, meaning the authenticity and genuineness of the certificate contents, is proven. Anyone who has the public key, meaning the certificate from the certificate authority, can check the signature and thus prove that the certificate was actually signed by the certificate authority.

The figure below shows how Alice uses the public key in the certificate from Twent (who represents the certificate authority, CA) to verify the signature on Bob's public key. All that is required for verification is therefore the availability of the certificate from the certificate authority at the time of verification. The validation itself is executed automatically in the TLS session.



Signing messages

The method described above for signing and verifying also uses the TLS session for signing and verifying messages:

If a hash value is generated by a message and this hash value is signed with the private key of the sender and attached to the original message, the recipient of the message is able to check the integrity of the message. The recipient decrypts the hash value with the public key of the sender, puts together the hash value from the message received and compares the two values. If the values are not the same, the message has been tampered with on the way.

Chain of certificates to root certificate

The certificates of a PKI are often organized hierarchically: The top of the hierarchy is formed by root certificates. These are certificates that are not signed by a higher-level certificate authority. The certificate holder and certificate issuer of root certificates are identical. Root certificates enjoy absolute trust. They are the "anchor" of the trust and must therefore be known at the receiver as absolute certificates. They are stored in an area provided for trusted certificates.

The function of root certificates is to sign certificates from lower-level certificate authorities, so-called intermediate certificates. This transfers the trust from the root certificate to the intermediate certificate. An intermediate certificate can sign a certificate just like a root certificate; both are therefore referred to as "CA certificates".

This hierarchy can be continued over multiple intermediate certificates until the end-entity certificate. The end-entity certificate is the certificate of the user who is to be identified.

The validation process runs through the hierarchy in the opposite direction: As described above, the certificate issuer is established and the signature checked with the issuer's public key, then the certificate of the higher-level certificate issuer is established along the entire chain of trust to the root certificate.

Conclusion: The chain of intermediate certificates to the root certificate, the certificate path, must be available in every device that is to validate an end-entity certificate of the communication partner, irrespective of the type of secure communication that you configure.

Managing certificates with STEP 7 (S7-1500)

STEP 7 as of version V14 together with the S7-1500-CPUs as of FW version 2.0 support the Internet PKI (RFC 5280) in as far as an S7-1500 CPU is able to communicate with devices that similarly support the Internet PKI.

The use of X.509 certificates, for example to verify certificates as described in the preceding sections, is a result of this.

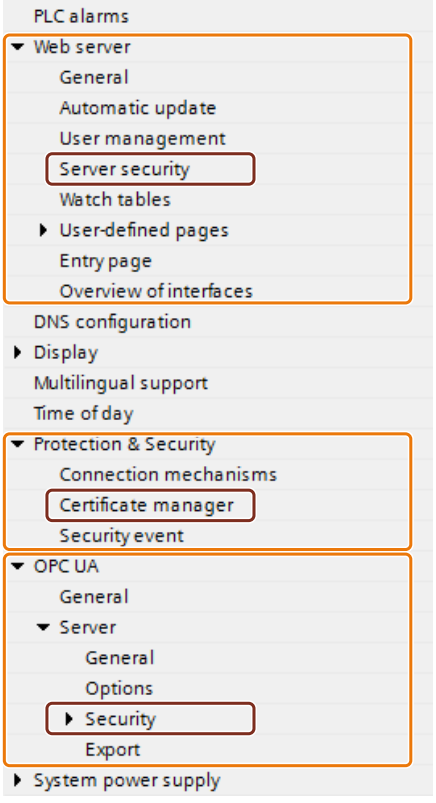
STEP 7 as of V14 uses a PKI similar to the Internet PKI. Certificate Revocation Lists (CRLs), for example, are not supported.

Create or assign certificates

For devices with security properties, such as an S7-1500 CPU as of firmware V2.0, you create certificates in STEP 7 for various applications.

The following areas in the Inspector window of the CPU allow the creation of new certificates or the selection of existing ones:

- "Protection & Security > Certificate manager" - for the generation or assignment of all types of certificates. The default setting for the creation of certificates is TLS certificates for Secure Open User Communication.
- "Web server > Security" - for the generation and assignment of Web server certificates.
- "OPC UA > Server > Security" - for the generation or assignment of OPC UA certificates

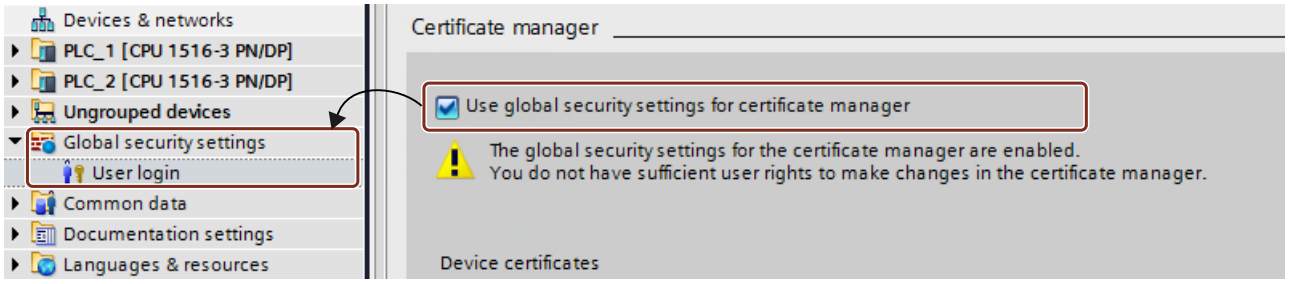


Special feature of the area "Protection & security > Certificate manager"

Only in this area of the Inspector window do you switch between the global, meaning the project-wide, certificate manager and the local, meaning the device-specific, certificate manager (option "Use global security settings for certificate manager"). The option decides whether you have access to all the certificates in the project or not.

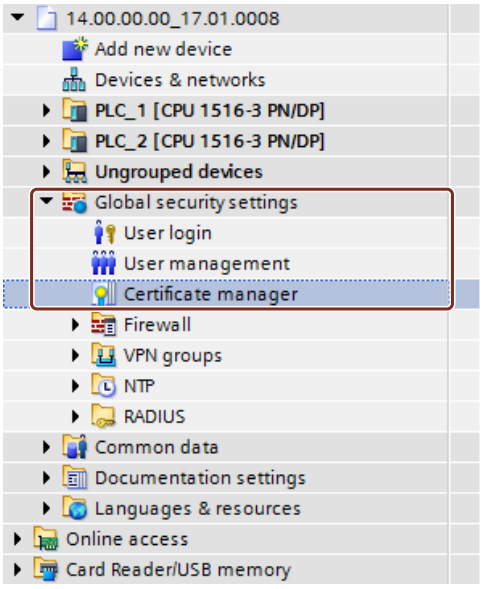
- If you do **not** use the certificate manager in the global security settings, you only have access to the local certificate memory of the CPU. You do not have any access, for example, to imported certificates or root certificates. Without these certificates only limited functionality is available. You can, for example, only create self-signed certificates.
- If you do not use the certificate manager in the global security settings and you are logged on as an administrator, you have access to the global, project-wide certificate memory. You can, for example, assign imported certificates to the CPU or create certificates that are issued and signed by the project CA (certification authority of the project).

The figure below shows how the "Global security settings" are displayed in the project tree after activation of the option "Use global security settings for certificate manager" in the Inspector window of the CPU.



When you double-click "User login" in the project tree below the global security settings and log in, a line "Certificate manager" is displayed there among other things.

With a double-click on the "Certificate manager" line you receive access to all the certificates in the project, divided into the tabs "CA" (certificate authorities), "Device certificates" and "Trusted certificates and root certification authorities".



Private keys

STEP 7 generates private keys when generating device certificates or server certificates (end-entity certificates). It depends on the use of the global security settings for the certificate manager where the private key is stored encrypted:

- If you use global security settings, the private key is stored encrypted in the global (project-wide) certificate memory.
- If you do not use global security settings, the private key is stored encrypted in the local (CPU-specific) certificate memory.

The existence of the private key that is required, for example, to decrypt data, is displayed in the "Private key" column in the "Device certificates" tab of the certificate manager in the global security settings.

When the hardware configuration is downloaded, the device certificate, the public key as well as the private key are downloaded to the CPU.

NOTICE
The option "Use global security settings for certificate manager" influences the previously used private keys: If you have already created certificates without using the certificate manager in the global security settings and then change the option to the use of the certificate manager, the private keys are lost and the certificate ID can change. A warning draws your attention to this fact. Therefore specify at the beginning of the configuration which option is required for the certificate management.

Examples for the management of certificates (S7-1500)

As explained in the preceding sections, certificates are required for every type of secure communication. The following section shows as an example how you handle the certificates with STEP 7 so that the requirements for Secure Open User Communication are met.

In the following section a distinction is made as to which devices the participating communication partners are. The respective steps for supplying the communication participants with the required certificates are described for each. An S7-1500 CPU or an S7-1500 software controller as of firmware version 2.0 is required.

The following generally applies:

During the establishment of a secure connection ("handshake") the communication partners as a rule only transfer their end-entity certificates (device certificates).

Therefore the CA certificates required to verify the transferred device certificate must be located in the certificate memory of the respective communication partner.

Note

The current date/time must be set in the CPU.

When using secure communication (for example, HTTPS, secure OUC, OPC UA), make sure that the corresponding modules have the current time of day and the current date. Otherwise, the modules evaluate the used certificates as invalid and the secure communication does not work.

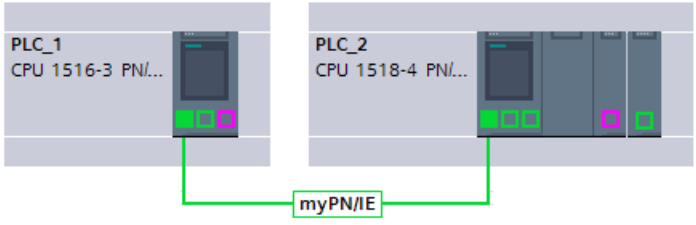
Secure Open User Communication between two S7-1500 CPUs

Two S7-1500 CPUs PLC_1 and PLC_2 should interchange data over Secure Open User Communication.

You generate the required device certificates with STEP 7 and assign them to the CPUs as described below.

STEP 7 project certification authorities (CA of the project) are used to sign the device certificates.

The certificates are to be referenced by their certificate ID in the user program (TCON communication instruction in combination with the associated system data type, for example TCON_IPV4_SEC). STEP 7 assigns the certificate ID automatically when generating or creating certificates.



Procedure

STEP 7 automatically downloads the required CA certificates together with the hardware configuration to the participating CPUs so that the requirements for certificate verification exist for both CPUs. You therefore only have to generate the device certificates for the respective CPU - everything else is done by STEP 7 for you.

1. Select PLC_1 and activate the "Use global security settings for certificate manager" option in the "Protection & Security" area.
2. Log in as a user in the project tree in the "Global security settings" area. The "Administrator" role is the default for the first login to a new project.
3. Return to the PLC 1 in the "Protection & Security" section. Click in an empty line in the "Certificate holder" column in the "Device certificates" table to add a new certificate.
4. Click the "Add" button in the drop-down list to select a certificate.
   The "Generate new certificate" dialog opens.
5. Leave the default settings in this dialog. They are tailored to the usage of Secure Open User Communication (usage: TLS).
   Tip: Extend the preset name of the certificate holder - in this case the CPU name. In order to differentiate better, leave the default CPU name in case you have to manage a large number of device certificates.
   Example: PLC_1/TLS becomes PLC_1-SecOUC-Chassis17FactoryState.
6. Compile the configuration.
   The device certificate and the CA certificate are part of the configuration.
7. Repeat the steps described for PLC_2.

In the next step you have to create the user programs for the data exchange and download the configurations together with the program.

Use self-signed certificates instead of CA certificates

When creating device certificates you can select the "Self-signed" option. You can create self-signed certificates without having to be logged in for the global security settings. This procedure is not recommended because the certificates created by this method do not exist in the global certificate memory and can therefore not be assigned directly to a partner CPU.

As described above you should select the name of the certificate holder with care so that the right certificate can be assigned to a device without any doubt.

A verification with the CA certificates of the STEP 7 project is not possible with self-signed certificates. To ensure that self-signed certificates can be verified, you have to include the self-signed certificate of the communication partner in the list of trusted partner devices for each CPU. To do so, you must have activated the option "Use global security settings for certificate manager" and be logged in as a user in the global security settings.

Proceed as follows to add the self-signed certificate of the communication partner of the CPU:

1. Select PLC_1 and navigate to the table "Certificates of partner devices" in the "Protection & Security" area.
2. Click in an empty line of the "Certificate holder" column to open the drop-down list for adding or selecting certificates.
3. From the drop-down list select the self-signed certificate of the communication partner and confirm the selection.

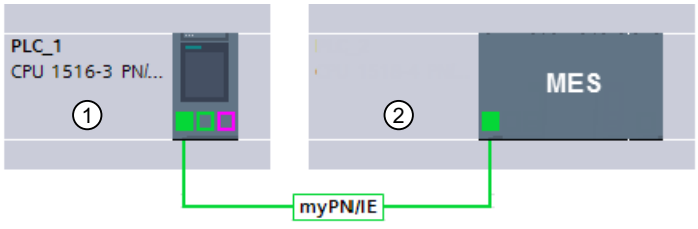
In the next step you have to create the user programs for the data exchange and download the configurations together with the program.

Secure Open User Communication between an S7-1500 CPU as a TLS client and an external device as a TLS server

Two devices are to interchange data via a TLS connection or TLS session, for example to exchange recipes, production data or quality data:

- An S7-1500 CPU (PLC_1) as the TLS client; the CPU uses Secure Open User Communication
- An external device (such as a Manufacturing Execution System (MES)) as the TLS server

As the TLS client the S7-1500 CPU establishes the TLS connection / session to the MES system.



① TLS client
② TLS server

In order to authenticate the TLS server, the S7-1500 CPU requires the CA certificates of the MES system: The root certificate and if appropriate the intermediate certificates for verifying the certificate path.

You have to import these certificates into the global certificate memory of the S7-1500 CPU.

Proceed as follows to import certificates of the communication partner:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (trusted certificates of root certification authorities) for the certificate to be imported.
3. Right-click in the table to open the shortcut menu. Click "Import" and import the required certificate or the required CA certificates.
   Through the import the certificate receives a certificate ID and can be assigned to a module in the next step.

4. Select PLC_1 and navigate to the table "Certificates of partner devices" in the "Protection & Security" area.
5. Click in an empty line of the "Certificate holder" column to add the imported certificates.
6. From the drop-down list select the required CA certificates of the communication partner and confirm the selection.

Optionally the MES system can also request a device certificate of the CPU to authenticate the CPU (meaning the TLS client). In this case the CA certificates of the CPU must be available to the MES system. Prerequisite for importing the certificates into the MES system is a preceding export of the CA certificates from the STEP 7 project of the CPU. Proceed as follows:

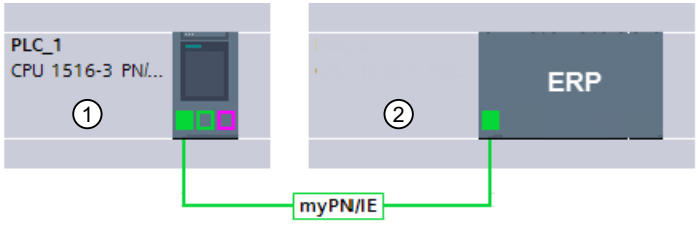
1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (CA certificates) for the certificate to be exported.
3. Right-click to open the shortcut menu for the selected certificate.
4. Click "Export".
5. Select the export format of the certificate.

In the next step you have to create the user programs for the data exchange and download the configurations together with the program.

Secure Open User Communication between an S7-1500 CPU as TLS server and an external device as TLS client

If the S7-1500 CPU acts as a TLS server and the external device, for example an ERP system (Enterprise Resource Planning System), establishes the TLS connection / session, you require the following certificates:

- For the S7-1500 CPU you generate a device certificate (server certificate) with a private key and download it with the hardware configuration. When generating the server certificate you use the option "Signed by certificate authority".
  The private key is required for the key exchange as explained in the figure for the example "HTTP over TLS".
- For the ERP system you have to export the CA certificate of the STEP 7 project and import / download it to the ERP system. With the CA certificate, the ERP system checks the server certificate of the S7-1500 that was transmitted from the CPU to the ERP system during the establishment of the TLS connection/session.



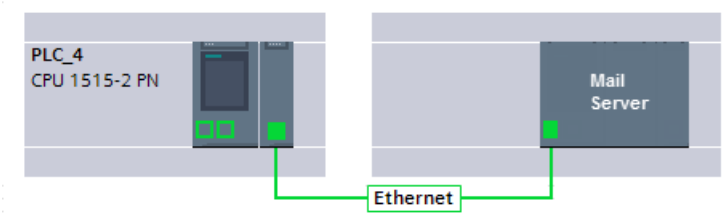
① TLS server
② TLS client

The description of the required actions is provided in the preceding sections.

Secure Open User Communication via S7-1543-1 CP as an e-mail client (SMTP over TLS)

An S7-1500 CPU can establish a secure connection to an e-mail server via the CP 1543-1 with the communication instruction TMAIL-C.

The system data types TMail_V4_SEC, TMail_V6_SEC and TMail_QDN_SEC allow you to specify the partner port of the e-mail server and thus to reach the e-mail server via "SMTP over TLS".



Prerequisite for a secure e-mail connection is an import of the root certificate and of the intermediate certificates from the e-mail server (provider) into the global certificate memory of the STEP 7 project. When these certificates have been assigned to the CP, the CP can check the server certificate of the e-mail server that is sent by the e-mail server during the establishment of the TLS connection / session by means of this certificate.

Follow these steps:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (trusted certificates of root certification authorities) for the certificate to be imported.
3. Right-click in the table to open the shortcut menu. Click "Import" and import the required certificate or the required CA certificates.
   Through the import the certificate receives a certificate ID and can be assigned to the CP in the next step.
4. Select the CP 1543-1 and navigate to the table "Certificates of partner devices" in the "Security" section.
5. Click in an empty line of the "Certificate holder" column to add the imported certificates.
6. From the drop-down list select the required CA certificates of the e-mail server and confirm the selection.

In the next step you have to create the user programs for the e-mail client function of the CPU and download the configurations together with the program.

See also

Secure OUC between two S7-1500 CPUs (Page 942)

Secure OUC from an S7-1500 CPU as a TLS client to an external PLC (TLS server) (Page 946)

Secure OUC from an S7-1500 CPU as TLS server to an external PLC (TLS client) (Page 949)

Secure OUC via e-mail (Page 955)

Secure OUC via CP interface (Page 951)

Example: HTTP over TLS (S7-1500)

The following paragraphs show how the mechanisms described are used to create secure communication between a Web browser and the Web server of an S7-1500 CPU.

First the changes for the option "Permit access only through HTTPS" in STEP 7 are described. As of STEP 7 V14 you have the option of influencing the server certificate of the Web server of an S7-1500 CPU as of firmware V2.0: The server certificate is generated with STEP 7 as of this version.

In addition the processes that are implemented when a website of the Web server of the CPU is called up with a Web browser of a PC via an encrypted HTTPS connection are displayed.

Using Web server certificates for S7-1500 CPUs as of FW V2.0

For S7-1500 CPUs with a firmware version before V2.0, you were able to set "Permit access only with HTTPS" when setting the Web server properties, without specific requirements applying.

You did not have anything to do with the handling of certificates for these CPUs; the CPU automatically generates the certificates required for the Web server.

For S7-1500 CPUs as of firmware V2.0, STEP 7 generates the server certificate (end-entity certificate) for the CPU. You assign a server certificate to the Web server in the properties of the CPU (Web server > Server security).

Since a server certificate name is always preset, nothing changes in the simple configuration of the Web server: You activate the Web server and activate the option "Permit access only with HTTPS" - STEP 7 generates a server certificate with the preset name during compiling.

Irrespective of whether you use the certificate manager in the global security settings or not: STEP 7 has all the information needed to generate the server certificate.

In addition you have the option of determining the properties of the server certificate, for example, the name or the duration of validity.

Note

The current date/time must be set in the CPU.

When using secure communication (for example, HTTPS, secure OUC, OPC UA), make sure that the corresponding modules have the current time of day and the current date. Otherwise, the modules evaluate the used certificates as invalid and the secure communication does not work.

Downloading the Web server certificate

The server certificate generated by STEP 7 is automatically also loaded when the hardware configuration is downloaded to the CPU.

- If you use the certificate manager in the global security settings, the certificate authority of the project (CA certificate) signs the server certificate of the Web server. When downloading the CA certificate of the project is automatically loaded as well.
- If you do not use the certificate manager in the global security settings, STEP 7 generates the server certificate as a self-signed certificate.

When you address the Web server of the CPU via the IP address of the CPU, a new server certificate (end-entity certificate) must be generated and loaded with each change of the IP address of an Ethernet interface of the CPU. This is necessary because the identity of the CPU changes with the IP address – and the identity requires a signature in accordance with the PKI rules.

You can avoid this problem by addressing the CPU with a domain name instead of its IP address, e.g. "myconveyer-cpu.room13.myfactory.com". To do so, you need to manage the domain names of your CPUs over a DNS server.

Supplying a Web browser with a CA certificate of the Web server

In the Web browser, the user who accesses the websites of the CPU through HTTPS should install the CA certificate of the CPU. If no certificate is installed, a warning is output recommending that you do not use the page. To view this page, you must explicitly "Add an exception".

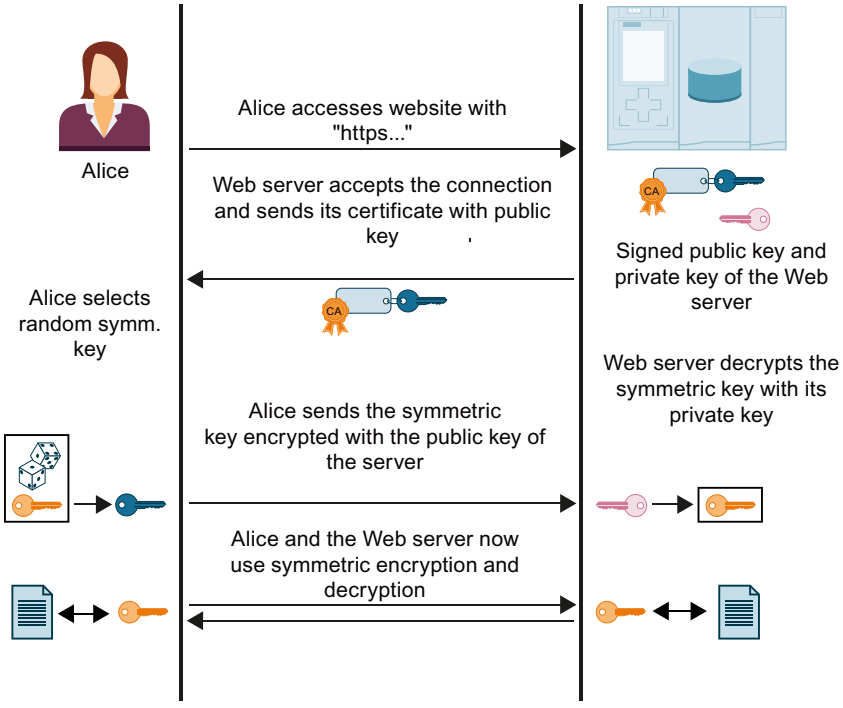
The user receives the valid root certificate (Certification Authority) for download from the "Intro" web page of the CPU Web server under "Download certificate".

STEP 7 offers a different possibility: Export the CA certificate of the project with the certificate manager into the global security settings in STEP 7. Subsequently import the CA certificate into the browser.

Course of Secure Communication

The figure below shows, in simplified terms, how communication is established ("handshake"), focusing on the negotiation of keys used for data exchange (here via HTTP over TLS).

However, the course can theoretically be applied to all communication options that are based on the use of TLS, i.e. also for Secure Open User Communication (see Basics of secure communication).



The figure does not show the measures taken at Alice's end (browser) to verify the certificate sent by the Web server. Whether Alice can trust the transferred Web server certificate and thus the identity of the Web server and allow the data exchange depends on a positive outcome of the verification.

The steps for verifying the authenticity of the Web server are:

1. Alice must know the public keys of all relevant certificate authorities, i.e. she requires the complete certificate chain to be able to verify the Web server certificate (i.e. the end-entity certificate of the Web server).
   Alice will generally have the required root certificate in her certificate memory. When a Web browser is installed, a whole range of trusted root certificates is also installed. If she does not have the root certificate, she has to download it from the certificate authority and install it in the certificate memory of the browser. The certificate authority can also be the device on which the Web server is located.
   You have the following options for obtaining the intermediate certificates:
   - The server itself sends the required intermediate certificates along with its end-entity certificate to Alice – in the form of a signed message so that Alice can check the integrity of the certificate chain.
   - The certificates often contain the URLs of the certificate issuer. Alice can load the required intermediate certificates from these URLs.

   When you handle certificates in STEP 7, it is always assumed that you have imported the required intermediate certificates and the root certificate into the project and have assigned them to the module.
2. Alice validates the signatures in the certificate chain with the public keys of the certificates.

3. The symmetric key has to have been generated and transferred to the Web server.
4. If the Web server is addressed by its domain name, Alice verifies the identity of the Web server in accordance with the PKI rules defined in RFC 2818. She is able to do this because the URL, in this case the "Fully Qualified Domain Name (FQDN) of the Web server, is saved in the end-entity certificate of the Web server. If the certificate entry in the "Subject Alternative Name" field corresponds to the entry in the address bar of the browser, everything is fine.

The process continues with the exchange of data with the symmetric key, as shown in the figure above.

Global security settings (S7-1500)

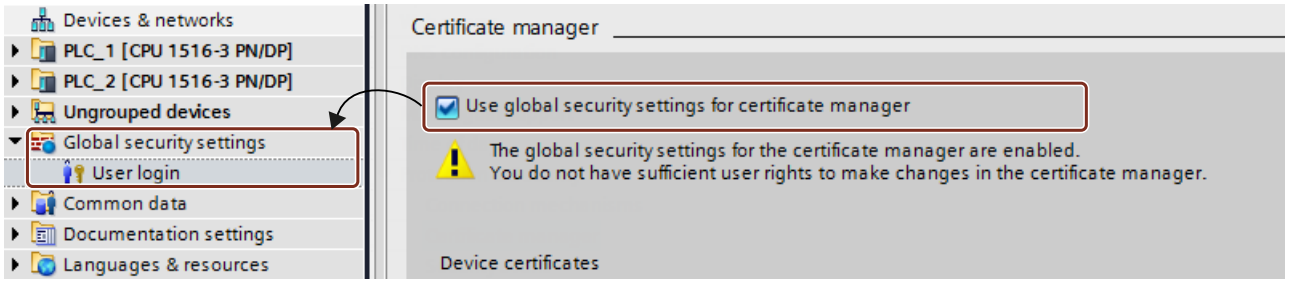
Global security certificate manager (S7-1500)

Introduction

In the global certificate manager, you have an overview of all the certificates, for example, CA certificates, used in the project with information about the certificate holder, issuer, validity, use and the existence of a private key. With the corresponding authorization you can also manage certificates for the project there.

Activation of the global security settings

The global security settings are not visible until they have been activated in the project for at last one device. The "Use global security settings for certificate manager" check box is available to this end in the local CPU-specific certificate manager:



The check box is located in the Inspector window in the general settings of a device with security functions under "Protection & Security > Certificate manager".

Principle

The CA certificate is issued by a certificate authority from which the device certificates are derived.

Certificate authorities can be:

- STEP 7: If the "certificate holder" and "issuer" are the same, this is a self-signed certificate; in other words a certificate issued by STEP 7.
- Higher-level certificate authority: These external certificates are external to the project and are imported and stored in the certificate memory of STEP 7.

Certificates created by one of the two certificate authorities always have a private key so that the device certificates can be derived from them.

Functions of the global security certificate manager

The following functions are available for selection in the global security certificate manager:

- Import of new certificates and certificate authorities.
- Import of SSL certificates (only CP x43-1 Adv.), for example, for FTP communication.
- Export of the certificates and certificate authorities used in the project.
- Renewal of expired certificates and certificate authorities.
- Replacement of existing certificate authorities.
- Adding trusted certificates and certification authorities.

Deleting manually imported certificates.

Note

Downloading the configuration

After replacing or renewing certificates the configuration has to be downloaded to the corresponding devices. After replacing or renewing CA certificates the configuration has to be downloaded to the corresponding devices.

Note

Current date and current time of day on the security modules

When using secure communication (for example, HTTPS), make sure that the corresponding devices have the current time of day and the current date. Otherwise the certificates used are not evaluated as valid and the secure communication does not work.

Managing users and roles (S7-1500)

Introduction

After you have activated the global security settings for the certificate manager, you have to log in to the global security settings or you require administrator rights. Otherwise you do not have sufficient user rights to edit the global certificates. You cannot access the global certificate manager without a login or if you do not have sufficient rights.

Roles and rights

You can configure the security settings if the predefined roles "Administrator" or "Standard" are assigned to you or if a different defined role with the associated authorization is assigned. You can view the roles in the project tree under "Global security settings > User administration > Roles" or, if you have corresponding rights, edit them.

Assigned role	Diagnosing	Configuring	Managing users and roles
Administrator	Yes	Yes	Yes
Standard	Yes	Yes	No
Diagnostics	Yes	No	No

You can create new roles and users as an administrator and assign users and roles to each other in the user administration.

Requirement

You have to be logged in as an administrator via the user login to the global security settings.

Procedure

Proceed as follows to edit users and roles:

1. In the project navigator go to user administration in the global security settings.
2. Select the role to be changed or create a new role.
3. Edit the rights of the selected role in the rights list.

Importing and exporting certificates (S7-1500)

Introduction

You can import and export certificates and the associated private keys. Certificates can only be imported with the global certificate manager. Certificates can also be exported via the local CPU-specific certificate manager.

You can import or export the certificates into the following file types:

- Certificates without private key to CER, DER, CRT or PEM
- Certificates with their private key to P12 (PKCS12).

Requirement

During import of certificates with private key you may require a password for access.

Importing certificates

To import a certificate follow these steps:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (CA certificates, device certificates, trusted certificates of root certification authorities) for the certificate to be imported.
3. Right-click in the table to open the shortcut menu.
4. Click "Import".
5. Select the import format of the certificate:
   - CER, DER, CRT or PEM for certificates without private keys.
   - P12 (PKCS12 archive) for certificates with private key.
6. Click "Open" to import the certificate.

The imported certificate has to be assigned manually to the desired device or module.

Exporting certificates

To export a certificate, follow these steps:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (CA certificates, device certificates, trusted certificates of root certification authorities) for the certificate to be exported.
3. Right-click to open the shortcut menu for the selected certificate.
4. Click "Export".
5. Select the export format of the certificate:
   - CER, DER, CRT or PEM for certificates without private keys.
   - CRL for certificate block list encoded to DER
   - P12 (PKCS12 archive) for certificates with private key.
6. Click "Save" to export the certificate.

You can also export certificates directly from the local CPU-specific certificate manager.

Selectable file formats

X.509 certificates to the certificate standard X.509 V3 from the RFC 5280 can be imported or exported in various formats with different file name extensions:

File name extension	**PEM**
File type	Privacy Enhanced Mail Security Certificate
Usage	Base64-encoded certificate
MIME type	application/x-pem-file

File name extension	PEM
Meaning	Certificate in PEM/X509 format (PEM = Privacy-Enhanced Mail) with an encoding in ASCII (Base64). Format of the certificate authorities often used for the certificate transfer via e-mail to the applicant of the certificate. Instead of the file name extension PEM, a Base64-enoded certificate can also have the extensions CER or CRT.
Usage	In TIA Portal the import and export is supported with PEM encoding. An export with PEM encoding can be effected in the file format PEM or CRT.

File name extension	DER
File type	DER Encoded X509 Certificate
Usage	Binary encoded certificate
MIME type	application/x-x509-ca-cert, application/pkix-cert
Meaning	Certificate in DER/X.509 format (DER = Distinguished Encoding Rules) with binary encoding. The binary DER format is the basis for the ASCII-encoded PEM format. The DER format is used in particular in the Microsoft Windows environment and on Java platforms. Instead of the file name extension DER a binary-encoded certificate can also have the extensions CER or CRT.
Usage	In TIA Portal the import and export is supported with DER encoding. An export with DER encoding can be effected in the file format DER or CER.

File name extension	CER
File type	Internet Security Certificate File
Usage	After PEM or DER encoded certificate
MIME type	application/x-x509-ca-cert, application/pkix-cert, application/keychain_access
Meaning	Alternative file name extension CER (Canonical Encoding Rules) for a binary encoded certificate in the DER format or for a Base64/ASCII-encoded certificate in PEM format.
Usage	In TIA Portal the import and export is supported in the file format CER. When importing a certificate in the file format CER the encoding can be to PEM or DER. The DER encoding is used for export in the file format CER.

File name extension	CRT
File type	Internet Security Certificate File
Usage	After PEM or DER encoded certificate
MIME type	application/x-x509-ca-cert, application/pkix-cert, application/keychain_access
Meaning	Alternative file name extension CRT for a binary encoded certificate in the DER format or for a Base64/ASCII-encoded certificate in PEM format.
Usage	In TIA Portal the import and export is supported in the file format CRT. When importing a certificate in the file format CRT the encoding can be to PEM or DER. The PEM encoding is used for export in the file format CRT.

File name extension	CRL
File type	Certificate Revocation List
Usage	Certificate block list encoded to DER
MIME type	application/pkix-crl, application/x-pkcs7-crl
Meaning	The CRL (Certificate Revocation List) is an X.509 V3 extension and contains the list of blocked or revoked certificates for a specific root certificate. Certificate authorities use the option to revoke the authorization of certificates already issued. This is the case when certificates are compromised or if their content turns out to be faulty. Blocked certificates can be unblocked again with the CRL, for example, when the certificate turns out not to be compromised after all or if the content was correct after all. Revoked certificates remain invalid forever and cannot be authorized again. The certificate authority updates its CRLs as soon as their validity has expired or the content has changed. The source of supply of a CRL is entered in the respective certificate.
Usage	The TIA Portal does not support a certificate revocation via CRL. Since some external clients must have certificate block lists for the validation of the certificates, an export of the CRL is however supported. But the list is empty and solely serves the purpose of making the validation of an output certificate by certain external clients possible.

File name extension	P12
File type	Personal Information Exchange File
Usage	As PKCS12-archive encoded certificates, optionally with private key
MIME type	application/keychain_access, application/x-pkcs12
Meaning	File types in the format P12 can also contain the private key in addition to the server certificate and intermediate certificates. A password can be assigned when storing.
Usage	In TIA Portal the import and export is supported in the file format P12. The option for exporting into a PKCS12 archive only exists if the certificate disposes of a private key. During export a password can be assigned for the PKCS12 archive. Analog to the password assignment during exporting, the entry of a password may be required to access the certificate during importing.

Renewing certificates (S7-1500)

Introduction

You can renew certificates, for example, to update a compromised certificate by an imported or new certificate.

Procedure

To renew a certificate, follow these steps:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (CA certificates, device certificates) for the certificate to be renewed.
3. Right-click to open the shortcut menu for the selected certificate.
4. Click "Renew".

5. You can edit the data of the selected certificate in the dialog for new certificates.
6. When you click "OK", the certificate is renewed with the changed values.

When you renew a certificate, the old certificate is replaced by the certificate with the new values. Through the renewal of certificates no signature query to a certificate authority is generated.

Replacing certificates (S7-1500)

Introduction

You can replace an existing certificate by a different certificate.

Procedure

To replace a certificate, follow these steps:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (CA certificates, device certificates, trusted certificates of root certification authorities) for the certificate to be replaced.
3. Right-click to open the shortcut menu for the selected certificate.
4. Click "Replace".
5. In the dialog for changing the certificate authority you replace the existing CA certificate of the project or the CA group certificate with a new one.

All certificates listed in the "Certificates involved" table are derived once again.

Displaying certificates (S7-1500)

Introduction

You can have an overview of all the certificate data displayed via the Windows certificate dialog

Procedure

Proceed as follows to display the certificate data:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (CA certificates, device certificates, trusted certificates of root certification authorities) for the certificate to be displayed.
3. Right-click to open the shortcut menu for the selected certificate.
4. Click "Display".

Result

Different information about the selected certificate is displayed in a window:

- General information about the certificate
- Detailed information about the certificate
- Information about the certification path

Deleting certificates (S7-1500)

Introduction

You can delete certificates that are no longer required or are compromised.

Procedure

To delete a certificate, follow these steps:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (device certificates, trusted certificates of root certification authorities) for the certificate to be deleted.
3. Right-click to open the shortcut menu for the selected certificate.
4. Click "Delete".

Note

CA certificates reserved for the project and certificates that are assigned automatically to a CP cannot be deleted.

You can also delete certificates directly in the local CPU-specific certificate manager. The certificate then remains in the global security manager for the project, but is no longer used for the device.

Secure Open User Communication (S7-1500)

Secure OUC between two S7-1500 CPUs (S7-1500)

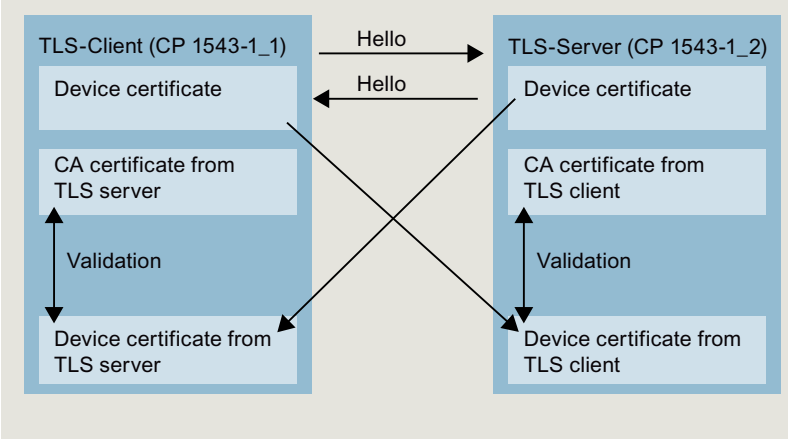
The following section describes how you can set up a Secure Open User Communication over TCP between two S7-1500 CPUs In the process one S7-1500 CPU acts as a TLS client (active connection establishment) and the other S7-1500 CPU as a TLS server (passive connection establishment).

Setting up a secure TCP connection between two S7-1500 CPUs

In order to establish secure TCP communication between two S7-1500 CPUs you have to create a data block with the system data type TCON_IPv4_SEC in each CPU, carry out the parameter assignment and call it directly at the instruction. The TCON instruction supports the system data type TCON_QDN_SEC.

Requirements:

- Both S7-1500 CPUs have a minimum firmware version of V2.0
- TLS client and TLS server have all the required certificates.



Settings at the TLS client

To set up a secure TCP connection in the TLS client, follow these steps:

1. Create a global data block in the project tree.
2. Define a tag of the data type TCON_IP_4_SEC in the global data block. To do so, enter the string "TCON_IP_V4_SEC" in the "Data type" field.
   The example below shows the global data block "Data_block_1" in which the tag "SEC connection 1 TLS Client" of the data type TCON_IP_V4_SEC is defined.

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC connection 1 TLS-Client	TCON_IP_V4_SEC		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	72	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	10	connection reference / identifier
6	ConnectionType	Byte	11	type of connetion: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	true	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of Byte		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	1	IPv4 address
13	ADDR[4]	Byte	100	IPv4 address
14	RemotePort	UInt	4711	remote UDP/TCP port number
15	LocalPort	UInt	4711	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 address against th
19	TLSServerCertRef	UDInt	1	for Server side: Reference to own X.509 V3 server certificate; fo
20	TLSClientCertRef	UDInt	5	for Client side: add id of own X.509 V3 client certificate; for Ser

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the TLS server for "RemoteAddress".
4. Set the parameters for Secure Communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerCertRef": ID of the own X.509-V3 certificate.
   - "TLSClientCertRef": Enter, for example, the value 1. This references the CA certificate of the TIA Portal project (SHA1).
5. Create a TCON instruction in the program editor.
6. Interconnect the CONNECT parameter of the TCON instruction with the tag of the data type TCON_IP_V4_SEC.

Settings at the TLS server

To set up a secure TCP connection in the TLS server, follow these steps:

1. Create a global data block in the project tree.
2. In the global data block, define a tag of the data type TCON_IP_V4_SEC.
   The example below shows the global data block "Data_block_1" in which the tag "SEC connection 1 TLS Server" of the data type TCON_IP_V4_SEC is defined.

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC connection 1 TLS-Server	TCON_IP_V4_SEC		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	120	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	10	connection reference / identifier
6	ConnectionType	Byte	11	type of connetion: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	false	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of Byte		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	1	IPv4 address
13	ADDR[4]	Byte	10	IPv4 address
14	RemotePort	UInt	4711	remote UDP/TCP port number
15	LocalPort	UInt	4711	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	true	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 address against the
19	TLSServerCertRef	UDInt	6	for Server side: Reference to own X.509 V3 server certificate; fo
20	TLSClientCertRef	UDInt	1	for Client side: add id of own X.509 V3 client certificate; for Serv

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the TLS client for "RemoteAddress".
4. Set the parameters for Secure Communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerReqClientCert ": Request for an X.509-V3 certificate from the TLS client Enter the value "true".
   - "TLSServerCertRef": ID of the own X.509-V3 certificate.
   - "TLSClientCertRef": Enter, for example, the value 1. This references the CA certificate of the TIA Portal project (SHA1).
5. Create a TCON instruction in the program editor.
6. Interconnect the CONNECT parameter of the TCON instruction with the tag of the data type TCON_IP_V4_SEC.

See also

Connection parameters according to TCON_IP_V4_SEC (Page 627)

Connection parameters in accordance with TCON_QDN_SEC (Page 629)

TCON: Establish communication connection (Page 4478)

Examples for the management of certificates (Page 926)

Secure OUC from an S7-1500 CPU as a TLS client to an external PLC (TLS server) (S7-1500)

The following section describes how you can set up a Secure Open User Communication over TCP from an S7-1500 CPU as TLS client to a TLS server.

Setting up a secure TCP connection from an S7-1500 CPU as TLS client to a TLS server

S7-1500 CPUs as of firmware version V2 support Secure Communication with addressing via a Domain Name System (DNS).

In order to establish secure TCP communication via the domain name you have to create a data block yourself with the system data type TCON_QDN_SEC, carry out the parameter assignment and call it directly at the instruction. The TCON instruction supports the system data type TCON_QDN_SEC.

Requirements:

- At least one DNS server exists in your network.
- You have configured at least one DNS server for the S7-1500 CPU.
- TLS client and TLS server have all the required certificates.

To set up a secure TCP connection to a TLS server, follow these steps:

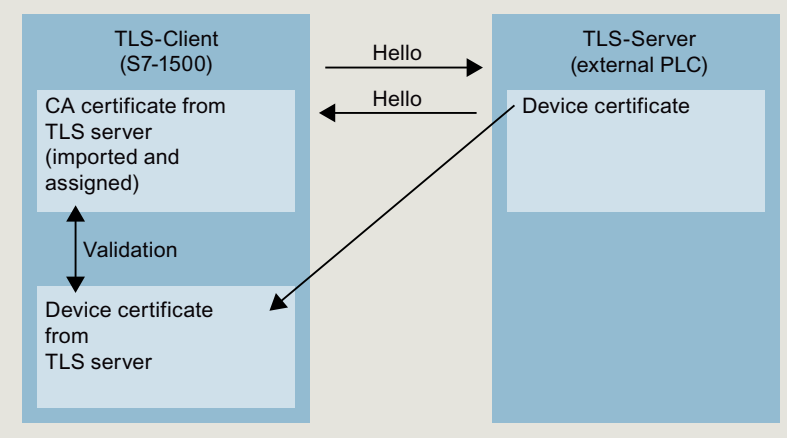
1. Create a global data block in the project tree.
2. In the global data block, define a tag of the data type TCON_QDN_SEC.
   The example below shows the global data block "Data_block_1" in which the tag "DNS ConnectionSEC" of the data type TCON_QDN_SEC is defined.

Data_block_1

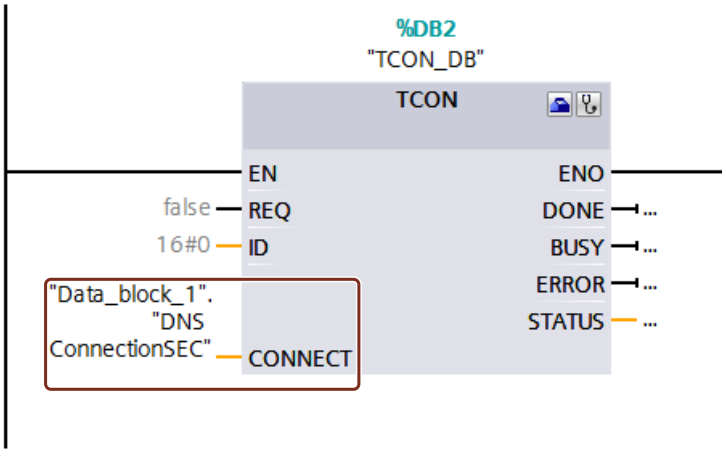
	Name	Data type	Start value	Comment
1	Static			
2	DNS Connection SEC	TCON_QDN_SEC		
3	ConnPara	TCON_QDN		parameter of the TCP connection
4	InterfaceId	HW_ANY	0	not relevant
5	ID	CONN_OUC	5	connection reference / identifier
6	ConnectionType	Byte	11	type of connection: 16#0B=11=TCP/IP, 16#13
7	ActiveEstablished	Bool	true	active/passive connection establishment
8	RemoteQDN	String[254]	'plc_1.factory127.'	fully or partially qualified domain name of rem
9	RemotePort	UInt	4000	remote UDP / TCP port number
10	LocalPort	UInt	0	local UDP / TCP port number
11	ActivateSecureConn	Bool	true	activate the security functionality of that conn
12	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a
13	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 a
14	TLSServerCertRef	UDInt	7	for Server side: Reference to own X.509 V3 se
15	TLSClientCertRef	UDInt	0	for Client side: add id of own X.509 V3 client c

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the fully qualified domain name (FQDN) of the TLS server for "RemoteQDN".

4. Set the parameters for Secure Communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "ExtTLSCapabilities": When you enter the value 1, the client validates the subjectAlternateName in the X.509-V3 certificate of the server to check the identity of the server. The certificates are checked when the connection is established.
   - "TLSServerCertRef": ID of the X.509-V3 certificate (usually a CA certificate) that is used by the TLS client to validate the TLS server authentication. If this parameter is 0, the TLS client uses all (CA) certificates currently loaded in the client certificate store to validate the server authentication. The certificate ID of the CA certificate is entered in the TLS server.



   - "TLSClientCertRef": ID of the own X.509-V3 certificate.
5. Create a TCON instruction in the program editor.
6. Interconnect the CONNECT parameter of the TCON instruction with the tag of the data type TCON_QDN_SEC.
   In the example below, the CONNECT parameter of the TCON instruction is interconnected with the tag "DNS connectionSEC" (data type TCON_QDN_SEC).



See also

Connection parameters in accordance with TCON_QDN_SEC (Page 629)

TCON: Establish communication connection (Page 4478)

Examples for the management of certificates (Page 926)

Secure OUC from an S7-1500 CPU as TLS server to an external PLC (TLS client) (S7-1500)

The following section describes how you can set up a Secure Open User Communication over TCP from one S7-1500 CPU as TLS server to a TLS client.

Setting up a secure TCP connection via the domain name of the communication partner

S7-1500 CPUs as of firmware version V2 support Secure Communication with addressing via a Domain Name System (DNS).

In order to establish secure TCP communication via the domain name you have to create a data block yourself with the system data type TCON_QDN_SEC, carry out the parameter assignment and call it directly at the instruction. The TCON instruction supports the system data type TCON_QDN_SEC.

Requirements:

- At least one DNS server exists in your network.
- You have configured at least one DNS server for the S7-1500 CPU.
- TLS client and TLS server have all the required certificates.

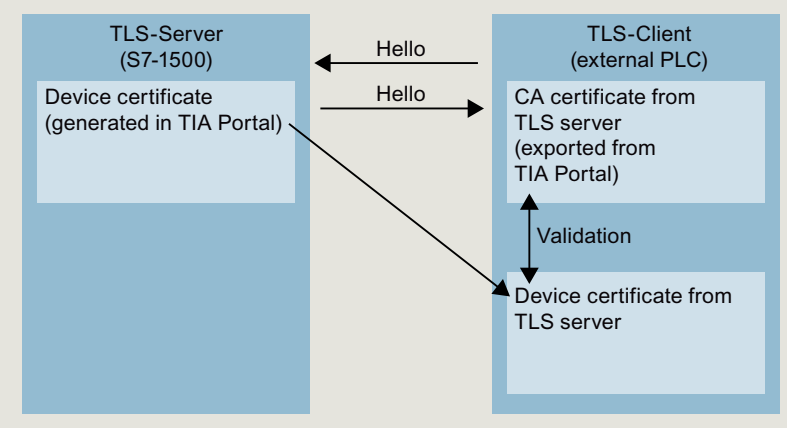
To set up a secure TCP connection to a TLS client, follow these steps:

1. Create a global data block in the project tree.
2. In the global data block, define a tag of the data type TCON_QDN_SEC.
   The example below shows the global data block "Data_block_1" in which the tag "DNS ConnectionSEC" of the data type TCON_QDN_SEC is defined.

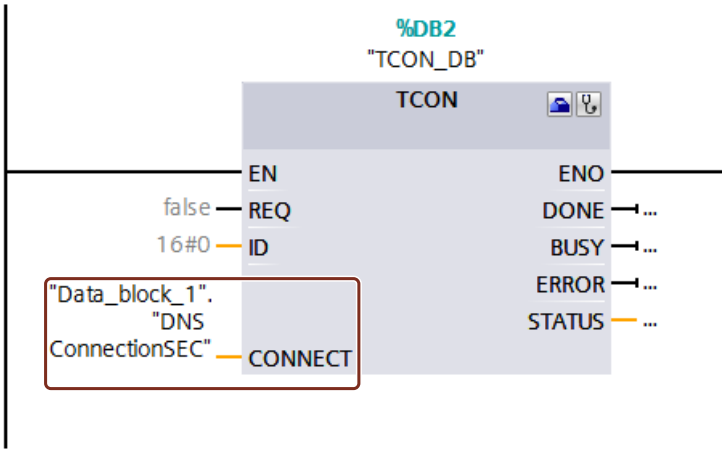
Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	DNS Connection SEC2	TCON_QDN_SEC		
3	ConnPara	TCON_QDN		parameter of the TCP connection
4	InterfaceId	HW_ANY	0	not relevant
5	ID	CONN_OUC	8	connection reference / identifier
6	ConnectionType	Byte	11	type of connection: 16#0B=11=TCP/IP, 16#13
7	ActiveEstablished	Bool	false	active/passive connection establishment
8	RemoteQDN	String[254]	''	fully or partially qualified domain name of rem
9	RemotePort	UInt	0	remote UDP / TCP port number
10	LocalPort	UInt	2010	local UDP / TCP port number
11	ActivateSecureConn	Bool	true	activate the security functionality of that conn
12	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a
13	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 a
14	TLSServerCertRef	UDInt	5	for Server side: Reference to own X.509 V3 se
15	TLSClientCertRef	UDInt	0	for Client side: add id of own X.509 V3 client c

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the local ID of the TCP connection in "ID".
4. Set the parameters for Secure Communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerReqClientCert": Request for an X.509-V3 certificate from the TLS client
   - "TLSServerCertRef": ID of the own X.509-V3 certificate.



   - "TLSClientCertRef": ID of the X.509-V3 certificate (or a group of X.509-V3 certificates) that is used by the TLS server to validate TLS client authentication. If this parameter is 0, the TLS server uses all (CA) certificates currently loaded in the server certificate store to validate the client authentication.
5. Create a TCON instruction in the program editor.
6. Interconnect the CONNECT parameter of the TCON instruction with the tag of the data type TCON_QDN_SEC.
   In the example below, the CONNECT parameter of the TCON instruction is interconnected with the tag "DNS connectionSEC" (data type TCON_QDN_SEC).



See also

Connection parameters in accordance with TCON_QDN_SEC (Page 629)

TCON: Establish communication connection (Page 4478)

Examples for the management of certificates (Page 926)

Secure OUC via CP interface (S7-1500)

The following sections describes the particular points to be taken into consideration in the case of Secure Open User Communication via a CP interface. At least one station is an S7-1500 station with the following modules:

- S7-1500 CPU as of firmware version V2.0 (with the exception of S7-1500 Software Controller)
- CP 1543-1 as of firmware version V2.0 or CP 1543SP-1 as firmware version V1.0

The CP acts in an S7-1500 station as a TLS client (active connection establishment) or a TLS server (passive connection establishment).

The fundamental procedure and the concept for using secure communication via a CP interface is similar to that of secure communication via the interfaces of the S7-1500 CPUs. Essentially, you have to assign the certificates to the CPU in the role of a TLS server or TLS client and not to the CPU. Other rules and procedures therefore apply. These are described below.

Handling certificates for CPs

The following applies in general: You have to be logged on at the certificate manager in the global security settings. The generation of self-signed certificates also requires logon for the global security settings. You have to have sufficient rights as a user (administrator or user with the "Standard" role with the right to "Configure security").

The starting point for the generation or assignment of certificates at the CP is the section "Security > Security properties". In this section, you log on for the global security settings.

Procedure:

1. In the network view of STEP 7, mark the CP and select the section "Security > Security properties" in the Inspector window.
2. Click on the "User logon" button.
3. Log on using your user name and password.
4. Enable the "Activate security functions" option.
   The security properties are initialized.
5. Click in the first line of the "Device certificates" table to generate a new certificate or select an existing device certificate.
6. If the communication partner is also an S7-1500 station, you also have to assign a device certificate to the communication partner with STEP 7 as described here or for the S7-1500 CPU.

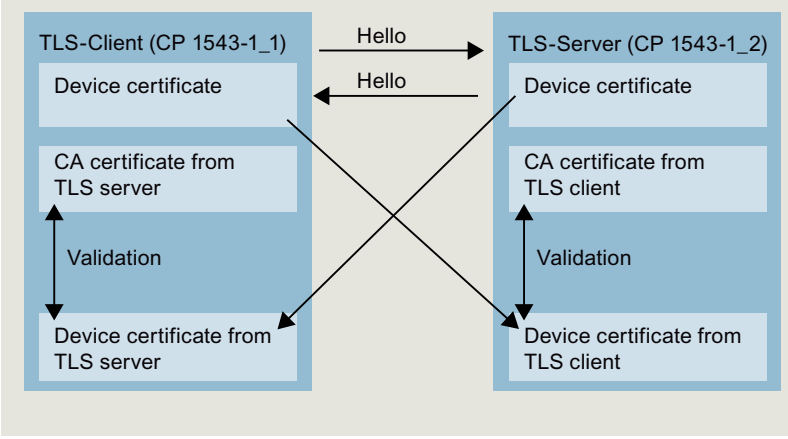
Example: Setting up a secure TCP connection between two S7-1500 CPUs via CP interfaces

In order to establish secure TCP communication between two S7-1500 CPs, you have to create a data block with the system data type TCON_IPv4_SEC in each CPU, carry out the parameter assignment and call it directly at the instruction.

Requirements:

- Both S7 1500 CPUs have at least firmware version V2.0. If you use the CP 1543SP-1: Firmware version as of V1.0.
- Both CPs (for example CP 1543-1) must have at least firmware version V2.0
- TLS client and TLS server have all the required certificates.
  - A device certificate (end-entity certificate) for the CP must be generated and be located in the certificate memory of the CP. If a communication partner is an external device (for example an MES or ERP system), a device certificate also has to exist for this device.
  - The root certificate (CA certificate) with which the device certificate of the communication partner is signed must also be located in the certificate memory of the CP or in the certificate memory of the external device. If you use intermediate certificates, you have to ensure that the complete certificate path exists in the validating device. A device uses these certificates to validate the device certificate of the communication partner.
- The communication partner must always be addressed via its IPv4 address, not via its domain name.

The following figure shows the different certificates in the devices for the case that both communication partners communicate via a CP 1543-1. In addition, the figure shows the transfer of the device certificates during establishment of the connection ("Hello").



To set up a secure TCP connection in the **TLS client**, follow these steps:

1. Create a global data block in the project tree.
2. In the global data block, define a tag of the data type TCON_IP_V4_SEC. To do so, enter the string "TCON_IP_V4_SEC" in the "Data type" field.
   The example below shows the global data block "Data_block_1" in which the tag "SEC connection 1 TLS Client" of the data type TCON_IP_V4_SEC is defined.
   The Interface ID has the value of the HW identifier of the IE interface of the local CP (TLS client).

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC connection 1 TLS-Client	TCON_IP_V4_SEC		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	258	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	10	connection reference / identifier
6	ConnectionType	Byte	11	type of connetion: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	true	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of Byte		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	1	IPv4 address
13	ADDR[4]	Byte	100	IPv4 address
14	RemotePort	UInt	4711	remote UDP/TCP port number
15	LocalPort	UInt	4711	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 address against th
19	TLSServerCertRef	UDInt	1	for Server side: Reference to own X.509 V3 server certificate; fo
20	TLSClientCertRef	UDInt	5	for Client side: add id of own X.509 V3 client certificate; for Sen

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the TLS server for "RemoteAddress".
4. Set the parameters for Secure Communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerCertRef": Enter the value 2 (reference to the CA certificate of the TIA Portal project (SHA256) or the value 1 (reference to the CA certificate of the TIA Portal project (SHA1)).
   - "TLSClientCertRef": ID of the own X.509-V3 certificate.
5. Create a TCON instruction in the program editor.
6. Interconnect the CONNECT parameter of the TCON instruction with the tag of the data type TCON_IP_V4_SEC.

Settings at the TLS server

To set up a secure TCP connection in the **TLS server**, follow these steps:

1. Create a global data block in the project tree.
2. In the global data block, define a tag of the data type TCON_IP_V4_SEC.
   The example below shows the global data block "Data_block_1" in which the tag "SEC connection 1 TLS Server" of the data type TCON_IP_V4_SEC is defined.
   The interface ID has the value of the HW identifier of the IE interface of the local CP (TLS server).

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC connection 1 TLS-Server	TCON_IP_V4_SEC		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	260	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	10	connection reference / identifier
6	ConnectionType	Byte	11	type of connetion: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	false	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of Byte		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	1	IPv4 address
13	ADDR[4]	Byte	10	IPv4 address
14	RemotePort	UInt	4711	remote UDP/TCP port number
15	LocalPort	UInt	4711	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	true	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 address against the
19	TLSServerCertRef	UDInt	6	for Server side: Reference to own X.509 V3 server certificate; fo
20	TLSClientCertRef	UDInt	1	for Client side: add id of own X.509 V3 client certificate; for Serv

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the TLS client for "RemoteAddress".
4. Set the parameters for Secure Communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerReqClientCert ": Request for an X.509-V3 certificate from the TLS client Enter the value "true".
   - "TLSServerCertRef": ID of the own X.509-V3 certificate.
   - "TLSClientCertRef": Enter the value 2 (reference to the CA certificate of the TIA Portal project (SHA256) or the value 1 (reference to the CA certificate of the TIA Portal project (SHA1)).
5. Create a TCON instruction in the program editor.
6. Interconnect the CONNECT parameter of the TCON instruction with the tag of the data type TCON_IP_V4_SEC.

Upload device as new station

When you upload a configuration with certificates and configured secure Open User Communication as a new station into your STEP 7 project, the certificates of the CP are not uploaded, in contrast to the certificates of the CPU. After the device has been loaded as a new station, no more certificates are contained in the corresponding tables of the CPs for the device certificates.

You have to perform configuration of certificates again after the upload. Otherwise, renewed loading of the configuration results in the certificates that originally exist in the CP being deleted so that secure communication does not function.

Secure OUC connections via CPU and CP interfaces - similarities

- Connection resources:
  No differences between OUC and secure OUC. A programmed secure OUC connection uses a connection resource just like an OUC connection, irrespective of which IE/ PROFINET interface communicates with the station.
- Connection diagnostics:
  No differences between OUC and secure OUC connection diagnostics.
- Loading of projects with secure OUC connections into the CPU:
  Only possible in STOP of the CPU, if certificates are loaded as well.
  Recommendation: Load to device > Hardware and software. Reason: Ensuring the consistency between the program with secure OUC, hardware configuration and certificates.
  Certificates are loaded with the hardware configuration - therefore loading requires a stop of the CPU. The reloading of blocks that utilize further secure OUC connections is only possible in RUN if the certificates required for this purpose are already located on the module.

See also

TCON: Establish communication connection (Page 4478)

Connection parameters according to TCON_IP_V4_SEC (Page 627)

Examples for the management of certificates (Page 926)

Secure OUC via e-mail (S7-1500)

The following section describes how to establish a secure connection (SNMP over TLS) to a mail server with the communication instruction TMAIL_C.

Setting up a secure connection to a mail server

In order to establish a secure communication to a mail server you have to create a data block yourself with one of the system data types TMAIL_V4_SEC, TMAIL_V6_SEC or TMAIL_QDN_SEC, carry out the parameter assignment and call it directly at the TMAIL_C instruction.

Requirements:

- S7-1500 CPU as of firmware version V2.0 with communication module CP 1543-1 as of firmware version V2.0
- ET 200SP CPU as of firmware version V2.0 with communication module CP 1542SP-1 (IRC) as of firmware version V1.0
- You have assigned all CA certificates of the mail server (TLS server) to the CP (TLS client) and have downloaded the configuration to the CPU.

To set up a secure mail server connection via the IPv4 address of the mail server, follow these steps:

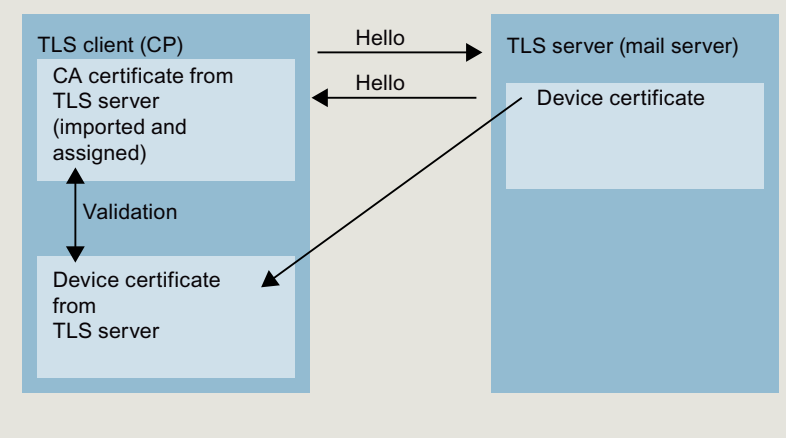
1. Create a global data block in the project tree.
2. In the global data block, define a tag of the data type TMAIL_V4_SEC.
   The example below shows the global data block "MailConnDB" in which the tag "MailConnectionSEC" of the data type TMAIL_V4_SEC is defined.

MailConnDB

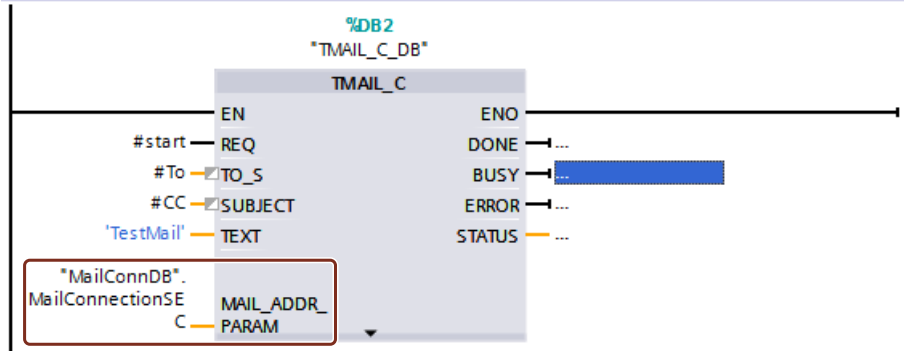
Name	Data type	Start value	Comment
Static			
MailConnectionSEC	TMail_V4_SEC		
InterfaceId	HW_ANY	"Local~CP_1543-1_1~Ethernet-Schnittstelle_1"	Use HW-identifier of the IE-interface to specify th
ID	CONN_OUC	100	connection reference / identifier
ConnectionType	Byte	16#20	type of connection 16#20=32=TMail_V4 or TMai
ActiveEstablished	Bool	true	active / passive connection establishment
WatchDogTime	Time	T#5000ms	watchdog time to monitor SMTP server associat
MailServerAddress	IP_V4		IPv4 address of mail server
ADDR	Array[1..4] of Byte		IPv4 address
ADDR[1]	Byte	144	IPv4 address
ADDR[2]	Byte	145	IPv4 address
ADDR[3]	Byte	2	IPv4 address
ADDR[4]	Byte	20	IPv4 address
UserName	String[254]	'myName'	user name which is necessary to login into the
PassWord	String[254]	'myPW'	user password which is necessary to login into
From	EMAIL_ADDR		source mail address
LocalPartPlusAt...	String[64]	'Mustermann@'	local part of e-mail address plus "@" sign
FullQualifiedD...	String[254]	'siemens.com'	full qualified domain name part of e-mail addre
RemotePort	UInt	587	remote TCP port number
ActivateSecureConn	Bool	TRUE	activate the security functionality of that conne
ExtTLSCapabilities	Byte	16#0	for further capability extensions of the TLS hand
TLSServerCertRef	UDInt	7	Reference to the X.509 V3 (CA-) certificate of the

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the mail server in "MailServerAddress".

4. Set the parameters for Secure Communication in the "Start value" column. For example, enter the certificate ID of the CA certificate of the communication partner in "TLSServerCertRef".
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerCertRef": Reference to the X.509 V3 (CA) certificate of the mail server which is used by the TLS client to validate the authentication of the mail server.



5. Create a TMAIL_C instruction in the program editor.
6. Interconnect the MAIL_ADDR_PARAM parameter of the TMAIL_C instruction with the tag of the data type TMAIL_V4_SEC.
   In the following example the MAIL_ADDR_PARAM parameter of the TMAIL_C instruction is interconnected with the "MAILConnectionSEC" tag (data type TMAIL_V4_SEC).



See also

Connection parameters in accordance with TCON_QDN_SEC (Page 629)

Description TMAIL_C as of version V4.0 (Page 4456)

Parameter MAIL_ADDR_PARAM as of Version 4.0 of TMAIL_C (Page 4462)

Examples for the management of certificates (Page 926)

10.1.4 Creating configurations

10.1.4.1 Configuring automation systems (S7-1200)

Functional description of S7-1200 CPUs (S7-1200)

Operating modes (S7-1200)

Principles of the operating modes of S7-CPUs (S7-1200)

Introduction

Operating modes describe the behavior of the CPU. The following operating modes are possible:

- STARTUP
- RUN
- STOP

In these operating modes, the CPU can communicate via the PN/IE interface, for example.

Other operating modes

If the CPU is not ready for operation, it is in one of following two operating modes:

- Deenergized, i.e. the supply voltage is switched off.
- Defective, which means an internal error has occurred.
  If the "Defective" status is caused by a firmware error, this state is indicated by the status LEDs of the CPU (refer to the description of the CPU). To find out the cause, follow these steps:
  - Turn the power supply switch off and on again.
  - Read out the diagnostics buffer when the CPU starts up and send the data for analysis to Customer Support.

  If the CPU does not start up, replace it.

See also

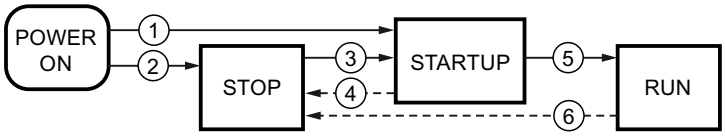
STOP mode (Page 963)

RUN mode (Page 963)

Operating mode transitions (S7-1200)

Overview

The following figure shows the operating modes and the operating mode transitions of S7-1200 CPUs:



The following table shows the conditions under which the operating modes will change:

No.	Operating mode transition	Conditions
①	POWER ON → STARTUP	After switching on, the CPU goes to "STARTUP" mode if: • "Warm restart" startup type is set, and • the hardware configuration and the program blocks are consistent. Non-retentive memory is cleared and the contents of non-retentive DBs are reset to the initial values of the load memory. Retentive memory and retentive DB contents are retained.
②	POWER ON → STOP	When startup type "No startup" is set, the CPU goes to "STOP" mode after the supply voltage is switched on. Non-retentive memory is cleared and the contents of non-retentive DBs are reset to the initial values of the load memory. Retentive memory and retentive DB contents are retained.
③	STOP → STARTUP	The CPU switches to "STARTUP" mode if: • CPU is set to "RUN" from the programming device, and • the hardware configuration and the program blocks are consistent.
④	STARTUP → STOP	The CPU returns to the "STOP" mode in the following situations: • Error detected during startup. • The CPU is set to "STOP" from the programming device. • A STOP command is processed in the STARTUP OB.
⑤	STARTUP → RUN	If the STARTUP is successful, the CPU switches to "RUN".
⑥	RUN → STOP	The CPU returns to the "STOP" mode in the following situations: • An error is detected that prevents continued processing. • The CPU is set to "STOP" from the programming device. • A STOP command is processed in the user program.

"STARTUP" operating mode (S7-1200)

Principles of the STARTUP mode (S7-1200)

Function

After turning on the CPU, it executes a startup program before starting to execute the cyclic user program.

By suitably programming startup OBs, you can specify certain initialization variables for your cyclic program in the startup program. There is no rule in terms of the number of startup OBs. That is, you can set up one or several startup OBs in your program, or none at all.

Parameter settings for startup characteristics

You can specify whether the CPU remains in STOP mode or whether a warm restart is run. Over and above this, you can set the response during startup (RUN or previous mode) in the "Startup" group of the CPU properties.

Special characteristics

Note the following points regarding the "STARTUP" mode:

- The startup OBs are executed. All startup OBs you have programmed are executed, regardless of the selected startup mode.
- No time-based program execution can be performed.
- Interrupt controlled program execution limited to:
  - OB 82 (diagnostics interrupt)
- The outputs on the modules are disabled.
- The process image is not updated; direct I/O access to inputs is possible.

See also

Editing properties and parameters (Page 575)

Principles of the operating modes of S7-CPUs (Page 958)

Startup OBs (Page 1021)

Warm restart (Page 960)

Warm restart (S7-1200)

Function

During a warm restart, all non-retentive bit memory is deleted and non-retentive DB contents are reset to the initial values from load memory. Retentive bit memory and retentive DB contents are retained.

Program execution begins at the call of the first startup OB.

Triggering a warm restart

You can trigger a "Warm restart" using a corresponding menu command on your programming device in the following situations:

- The CPU must be in "STOP" mode.
- After a memory reset
- After downloading a consistent program and a consistent hardware configuration in the "STOP" mode of the CPU.

"POWER ON" triggers a "warm restart" if you have set the following parameters for the startup response:

- Startup type "warm restart - RUN" (regardless of the CPU operating mode prior to POWER OFF).
- "Warm restart - mode prior to POWER OFF" (depending on the CPU operating mode prior to POWER OFF. The CPU must have been in RUN mode prior to this.)

See also

Retentive memory areas (Page 968)

Startup activities (S7-1200)

Overview

The following table shows which activities the CPU performs at STARTUP:

Activities in execution sequence	At warm restart
Clear non-retentive bit memories	Yes
Clear all bit memories	No
Clear the process image output	Yes
Processing startup OBs	Yes
Update the process image input	Yes
Enable outputs after changing to "RUN" mode	Yes

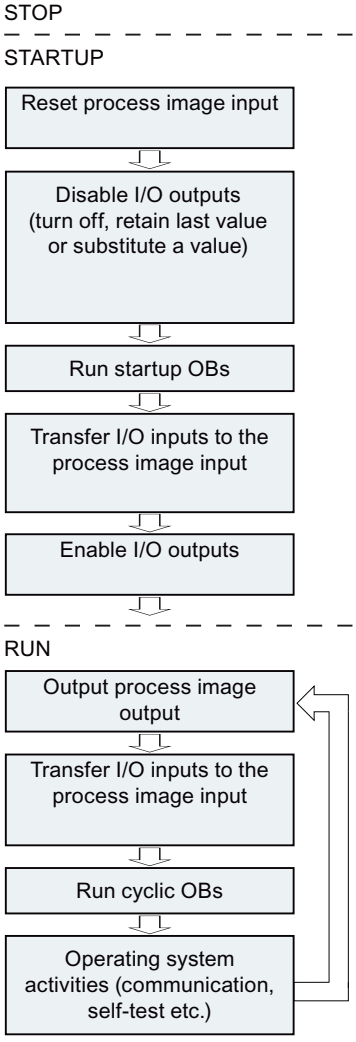
Sequence

The following figure shows the activities of the CPU in "STOP", "STARTUP", and "RUN" modes.

You can use the following measures to specify the state of the I/O outputs in the first cycle of the user program:

- Use assignable output modules to be able to output substitute values or to retain the last value.
- Set default values for outputs in startup OBs.

During the startup, all interrupt events are entered in a queue so that they can be processed later during RUN mode. In RUN mode, hardware interrupts can be processed at any time.



Special features during startup (S7-1200)

Response when expected and actual configurations do not match

The expected configuration is represented by the engineering configuration loaded on the CPU. The actual configuration is the actual configuration of the automation system.

If the expected configuration and actual configuration differ, the CPU nevertheless initially changes to RUN.

Canceling a STARTUP

If errors occur during startup, the startup is canceled and the CPU remains in "STOP" mode.

Under the following conditions, a startup will not be performed or will be canceled:

- If an invalid SD card is inserted.
- If no hardware configuration has been downloaded.

See also

Overview of the CPU properties (Page 982)

RUN mode (S7-1200)

Function

In "RUN" mode the cyclic, time-driven, and interrupt-driven program sections execute:

- The process image output is read out.
- The process image input table is read.
- The user program is executed.

Active data exchange between S7-1200 CPUs by means of Open User Communication is only possible in "RUN" mode.

Running the user program

Once the CPU has read the inputs, the cyclic program runs from the first to the last instruction.

If you have configured a minimum cycle time, the CPU will not end the cycle until this minimum cycle time is up even if the user program is completed sooner.

A maximum cycle time is set which you can adjust according to your requirements. This ensures that the cyclic program is completed within a specified time. The system will respond with a time error if the cyclic program is not completed within this time.

Other events such as hardware and diagnostic interrupts can interrupt the cyclic program flow and prolong the cycle time.

See also

Principles of the operating modes of S7-CPUs (Page 958)

Events and OBs (Page 973)

STOP mode (S7-1200)

Function

In "STOP" mode, the user program is not executed. All outputs are disabled or react according to the parameter settings: They provide a substitute value as set in the parameters or retain the last value output and bring the controlled process to a safe status.

The CPU checks the following points:

- Hardware, for example whether are all modules are available
- Whether the default settings for the CPU are applicable or parameter sets are present
- Whether the general conditions for the programmed startup behavior are correct

See also

Principles of the operating modes of S7-CPUs (Page 958)

Basics of a memory reset (S7-1200)

Function

A memory reset on the CPU is possible only in STOP mode.

When memory is reset, the CPU is changed to an "initial status". This means:

- An existing online connection between your programming device/PC and the CPU is terminated.
- The content of the work memory and the retentive and non-retentive data are deleted.
- The diagnostic buffer, the time, the IP address, the hardware configuration, and active force jobs are retained.
- The load memory (code and data blocks) is then copied to work memory. As a result, the data blocks no longer have current values but their configured initial values.

Memory areas (S7-1200)

What you need to know about SIMATIC memory cards (S7-1200)

Function of the SIMATIC memory card

The SIMATIC memory card for an S7-1200 is an SD memory card, preformatted by Siemens for the CPU user program, that is compatible with the Windows operating system.



NOTICE
Formatting the SIMATIC memory card You may only delete files and folders. If you format the SIMATIC memory card with Windows tools, for example with a commercially available card reader, you will render the memory card unusable as a storage medium for an S7 CPU.

Setting the card type

You can use the SIMATIC memory card as a transfer card, a program card or a firmware update card.

To set the card type, insert the SIMATIC memory card into the card reader of the programming device and select the "Card reader/USB memory" folder from the project tree. In the properties of the selected memory card, designate the card type:

- Program
  If the SIMATIC memory card is used as a program card, you can load the user program to the card. In this case, the internal load memory of the device is replaced by the SIMATIC memory card and the internal load memory is erased. The user program is then fully executable from the SIMATIC memory card. If the SIMATIC memory card with the user program is removed, there is then no program.
- Transfer
  If the SIMATIC memory card is used as a transfer card, you can transfer the user program from the card to the internal load memory of the CPU. You can then remove the SIMATIC memory card.
- Firmware card
  Firmware for the S7-1200 modules can be stored on a SIMATIC memory card. It is therefore possible to perform a firmware update with the help of a specifically prepared SIMATIC memory card. A backup copy of firmware for a module can also be stored on the SIMATIC memory card.

Transferring objects from the project to a SIMATIC memory card

When the SIMATIC memory card is inserted in the programming device or in an external card reader, you can transfer the following objects from the project tree to the SIMATIC memory card:

- Individual blocks (multiple selection possible)
  In this case a consistent transfer is available, as the dependencies of the blocks to each other is taken into account with block selection.
- PLC
  In this case, all objects relevant to processing, such as blocks and the hardware configuration, are transferred to the SIMATIC memory card, just as with downloading.

To perform the transfer, you can move the objects with drag-and-drop or use the command "Card reader/USB memory > Write to memory card" in the "Project" menu.

Transferring objects from the SIMATIC memory card to the project

You can transfer Individual blocks (multiple selection is possible) by dragging them to the project. A hardware configuration cannot be transferred from the SIMATIC memory card to the project.

Updating firmware with a SIMATIC memory card

You can get the latest firmware data on the Internet from the Service & Support pages:

http://support.automation.siemens.com (http://support.automation.siemens.com/WW/view/en/34143537)

Save the firmware files on the hard disk and plug the SIMATIC Memory Card into the card reader of your programming device.

To store the file on the SIMATIC memory card, select the SIMATIC memory card in the "Card Reader/USB memory" folder in the project tree. Select the shortcut menu "Card Reader/USB memory > Create firmware update memory card".

Then follow the instructions in the Service & Support portal for performing a firmware update with your CPU.

Updating the firmware changes the CPU firmware status. If you have used the CPU in the project, you will have to update the CPU already configured to the CPU with the new firmware status by changing devices offline, and adapt and then load the program or configuration.

See also

Replacing a hardware component (Page 572)

Useful information on CPU firmware versions and STEP 7 versions (Page 1043)

Displaying properties of memory cards (Page 423)

GetSMCinfo: Reading out information about the SIMATIC memory card (Page 3966)

Load memory (S7-1200)

Function

Each CPU has an internal load memory. The size of this internal load memory depends on the CPU used.

This internal load memory can be replaced by using external memory cards. If there is no memory card inserted, the CPU uses the internal load memory; if a memory card is inserted, the CPU uses the memory card as load memory.

See also

Using memory cards (Page 421)

Work memory (S7-1200)

Function

Work memory is a non-retentive memory area for storing elements of the user program that are relevant for program execution. The user program is executed exclusively in work memory and system memory.

System memory (S7-1200)

System memory areas (S7-1200)

Function

System memory contains the memory elements that each CPU makes available to the user program, such as the process image and bit memory.

By using appropriate operations in your user program, you address the data directly in the relevant operand area.

The following table shows the operand areas of the system memory:

Operand area	Description	Access via units of the following size:	S7 notation
Process image output	The CPU writes the values from the process image output table to the output modules at the start of the cycle.	Output (bit)	Q
		Output byte	QB
		Output word	QW
		Output double word	QD
Process image input	The CPU reads the inputs from the input modules and saves the values to the process image input table at the start of the cycle.	Input (bit)	I
		Input byte	IB
		Input word	IW
		Input double word	ID
Bit memory	This area provides storage for intermediate results calculated in the program.	Bit memory (bit)	M
		Memory byte	MB
		Memory word	MW
		Memory double word	MD
Data block	Data blocks store information for the program. They can either be defined so that all code blocks can access them (global DBs) or assigned to a specific FB or SFB (instance DB). Requirement: The block attribute "Optimized block access" is not enabled.	Data bit	DBX
		Data byte	DBB
		Data word	DBW
		Data double word	DBD

Operand area	Description	Access via units of the following size:	S7 notation
Local data	This area contains the temporary data of a block while the block is being processed. Requirement: The block attribute "Optimized block access" is not enabled. Recommendation: Access local data (temp) symbolically.	Local data bit	L
		Local data byte	LB
		Local data word	LW
		Local data double word	LD
I/O input area	The I/O input and output areas permit direct access to central and distributed input and output modules.	I/O input bit	<tag>:P
		I/O input byte	
		I/O input word	
		I/O input double word	
I/O output area		I/O output bit	
		I/O output byte	
		I/O output word	
		I/O output double word	

See also

Diagnostics buffer (Page 971)

Basic principles of process images (Page 969)

Basic principles for programming of data blocks (Page 4795)

Declaring local tags and constants in the block interface (Page 4772)

Overview of the block interface (Page 4764)

Access to the I/O addresses (Page 971)

Retentive memory areas (S7-1200)

Retentive memory areas

Data loss after power failure can be avoided by marking certain data as retentive. This data is stored in a retentive memory area. A retentive memory area is an area that retains its content following a warm restart, in other words, after cycling the power when the CPU changes from STOP to RUN.

The following data can be assigned retentivity:

- Bit memory: The precise width of the memory can be defined for bit memory in the PLC tag table or in the assignment list.
- Tags of a function block (FB): You can define individual tags as retentive in the interface of an FB if you have enabled optimized block access. Retentivity settings can be defined only in the assigned instance data block if optimized block access has not been activated for the FB.
- Tags of a global data block: You can define retentivity either for individual or for all tags of a global data block depending on the settings for access.
  - Block with optimized access: retentivity can be set for each individual tag.
  - Block with standard access: The retentivity setting applies to all tags of the DB; either all tags are retentive or no tag is retentive.

See also

Warm restart (Page 960)

Setting the retentivity of PLC tags (Page 4848)

process image input/output (S7-1200)

Basic principles of process images (S7-1200)

Function

When the user program addresses the input (I) and output (O) operand areas, it does not query or change the signal states on the digital signal modules. Instead, it accesses a memory area in the system memory of the CPU. This memory area is referred to as the process image.

Advantages of the process image

Compared with direct access to input and output modules, the main advantage of accessing the process image is that the CPU has a consistent image of the process signals for the duration of one program cycle. If a signal state on an input module changes during program execution, the signal state in the process image is retained until the process image is updated again in the next cycle. The process of repeatedly scanning an input signal within a user program ensures that consistent input information is always available.

Access to the process image also requires far less time than direct access to the signal modules since the process image is located in the internal memory of the CPU.

Updating the process images (S7-1200)

Sequence

The operating system updates the process images at cyclic intervals unless defined otherwise in your configuration. The process image input/output is updated in the following order:

1. The internal tasks of the operating system are performed.
2. The process image output (PIQ) table is written to the outputs of the module.
3. The status of inputs is read to the process image input (PII) table.
4. The user program is executed with all the blocks that are called in it.

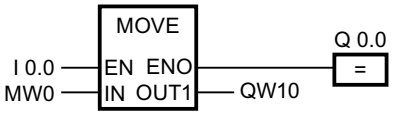
The operating system automatically controls the writing of the process image output to the outputs of the modules and the reading of the process image input.

Special characteristics

You have the option of accessing inputs or outputs directly using direct I/O access.

- If an instruction accesses an output directly and the output address is located in the process image output, the process image of the relevant output is updated.
- If an instruction accesses an output directly and the output address is **not** located in the process image output, the process image of the relevant output is **not** updated.

Example of normal I/O access by way of the process image



Update QW10 in the I/O output area with the value from MW0.

I/O access error during process image updating

If an error occurs during process image updating (I/O access error), the CPU reacts with the default system reaction "Ignore".

See also

Start address of a module (Page 971)

Access to the I/O addresses (Page 971)

Startup activities (Page 961)

Diagnostics buffer (S7-1200)

Function

The diagnostics buffer is part of the system memory of the CPU. It contains the errors detected by the CPU or modules with diagnostics capability. It includes the following events:

- Every mode change of the CPU (for example, POWER UP, change to STOP mode, change to RUN mode)
- Every diagnostics interrupt

The diagnostics buffer of the S7-1200-CPU has a capacity of 50 entries of which the last (most recent) 10 entries are retained following power cycling.

Those entries can only be cleared by restoring the CPU to factory defaults.

You can read the content of the diagnostics buffer with the help of the Online and Diagnostics view.

I/O data area (S7-1200)

Start address of a module (S7-1200)

Definition

The start address is the lowest byte address of a module. It acts as the initial address of the module user data area.

Configuring module start addresses

The addresses used in the user program and the module start addresses are coordinated when the modules are configured.

In the module properties ("I/O addresses" group), you can change the start addresses that were assigned automatically after the modules were inserted.

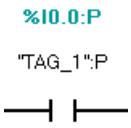
You can also make a setting that decides whether or not the addresses are located in the process image.

Access to the I/O addresses (S7-1200)

I/O addresses

If you insert a module in the device view, its user data is located in the process image of the S7-1200 CPU (default). The CPU handles the data exchange between the module and the process image area automatically during the update of the process images.

Append the suffix ":P" to the I/O address if you want the program to access the module directly instead of using the process image.



This could be necessary, for example, during execution of a time-sensitive program which also has to control the outputs within the same cycle.

Basics of program execution (S7-1200)

English designation of the OB types (S7-1200)

English designation of the OB types

When an OB is created in the "Add new block" dialog, the OB types can only be displayed in English for technical reasons – irrespective of the set user interface language. In order to facilitate handling, the following table lists the English terms and their corresponding designations in the set user interface language.

English designation of the OB type in the "Add new block" dialog.	Designation in the information system
Program cycle	Cycle OB, OB for cyclic program processing, cyclic program
Startup	Startup OB
Time delay interrupt	Time-delay interrupt OB
Cyclic interrupt	Cyclic interrupt OB
Hardware interrupt	Hardware interrupt OB
Time error interrupt	Time error OB
Diagnostic error interrupt	Diagnostic interrupt OB
Pull or plug of modules	Pull/plug interrupt OB
Rack or station failure	Rack failure OB
Time of day	Time-of-day interrupt OB
Status	Status interrupt OB
Update	Update interrupt OB
Profile	OB for vendor or profile-specific interrupt
MC-Interpolator	MC-Interpolator OB
MC-Servo	MC-Servo OB
MC-PreServo	MC-PreServo OB
MC-PostServo	MC-PostServo OB

Events and OBs (S7-1200)

Events and OBs

The operating system of S7-1200-CPUs is based on events. There are two types of events:

- Events which can start an OB
- Events which cannot start an OB

An event which can start an OB triggers the following reaction:

- It calls the OB you possibly assigned to this event. The event is entered in a queue according to its priority if it is currently not possible to call this OB.
- The default system reaction is triggered if you did not assign an OB to this event.

An event which cannot start an OB triggers the default system reaction for the associated event class.

The user program cycle is therefore based on events, the assignment of OBs to those events, and on the code which is either contained in the OB, or called in the OB.

The following table provides an overview of the events which can start an OB, including the associated event classes and OBs. The table is sorted based on the default OB priority. Priority class 1 is the lowest.

Event class	OB no.	Number of OBs	Start event	OB priority (default)
Cyclic program	1, >= 123	>= 1	Starting or end of the last program cycle OB	1
Startup	100, >= 123	>=0	STOP to RUN transition	1
Time-of-day interrupt	>= 10	Max. 2	Start time has been reached	2
Time-delay interrupt	>= 20	Max. 4	Delay time expired	3
Cyclic interrupt	>= 30		Constant bus cycle time expired	8
Hardware interrupt	>= 40	Max. 50 (more can be used with DETACH and ATTACH)	• Positive edge (max. 16) • Negative edge (max. 16)	18
			• HSC: Count value = reference value (max. 6) • HSC: Count direction changed (max. 6) • HSC: External reset (max. 6)	18
Status interrupt	55	0 or 1	CPU has received status interrupt	4
Update interrupt	56	0 or 1	CPU has received update interrupt	4
Manufacturer- or profile-specific interrupt	57	0 or 1	CPU has received manufacturer-specific or profile-specific interrupt	4
Diagnostic error interrupt	82	0 or 1	Module has detected an error	5
Pull/plug interrupt	83	0 or 1	Removal/insertion of modules of distributed I/O	6

Event class	OB no.	Number of OBs	Start event	OB priority (default)
Rack error	86	0 or 1	Error in the I/O system of the distributed I/O	6
Time error	80	0 or 1	• Maximum cycle time exceeded • Called OB is still being executed • Time-of-day interrupt missed • Time-of-day interrupt missed during STOP • Queue overflow • Interrupt loss due to high interrupt load	22

The following table describes events which do not trigger an OB start, including the corresponding reaction of the operating system. The table is sorted based on event priority.

Event class	Event	Event priority	System reaction
Insert/remove central modules	Insert/remove a module	21	STOP
I/O access error during process image update	I/O access error during process image update	22	Ignore
Programming error	Programming error in a block for which you use system reactions provided by the operating system (note: the error handling routine in the block program is executed if you activated local error handling).	23	RUN
I/O access error	I/O access error in a block for which you use system reactions provided by the operating system (note: the error handling routine in the block program is executed if you activated local error handling).	24	RUN
Maximum cycle time exceeded twice	Maximum cycle time exceeded twice	27	STOP

Assignment between OBs and events

With the exception of the cyclic program and startup program and event can only be assigned to one OB. However, in certain event classes such as hardware interrupts one and the same OB can be assigned to several events.

The assignment between OBs and events is defined in the hardware configuration. Defined assignments can be changed at runtime by means of ATTACH and DETACH instructions.

OB priority and runtime behavior

S7-1200-CPUs support the priority classes 1 (lowest) to 27 (highest). An OB is assigned the priority of its start event.

OBs are always executed on a priority basis: The OBs with the highest priority are executed first. Events of the same priority are processed in order of occurrence.

As of firmware version V4.0 of the S7-1200 CPUs you can specify in the device configuration, under properties of the CPU, if the OBs are interruptible or not. This parameter assignment has an effect on all OBs with exception of the cycle OBs which are always interruptible.

The following applies to S7-1200 CPUs with firmware version < V4.0:

- Any OB with priority >= 2 will interrupt cyclic program execution.
- An OB of priority 2 to 25 cannot be interrupted by any event of priority group 2 to 25. This rule also applies to events of a priority higher than that of the currently active OB. Such events are processed later.
- A time error (priority 26) will interrupt any other OB.

The following applies to S7-1200 CPUs as of firmware version V4.0:

If you do not configure the OBs as interruptible, an OB is always processed completely even if an event of a higher priority occurs during its runtime. Specifically, this means:

- Any OB with priority >= 2 will interrupt cyclic program execution.
- An OB of priority 2 to 25 cannot be interrupted by any event. This rule also applies to events of a priority higher than that of the currently active OB, which also includes a time error. Such events are processed later.

If you do configure the OBs as interruptible and an event of a higher priority occurs during the runtime of an OB, the running OB is interrupted and the OB associated with the occurring event is processed. Once this OB has been completed, processing of the interrupted OB continues. Specifically, this means:

- Any OB with priority >= 2 will interrupt cyclic program execution.
- An OB of priority 2 to 25 can be interrupted by any event whose priority is higher than that of the running OB. This is also true for time errors: A time error (priority 26) will interrupt any OB.

OB start information

Certain OBs have start information, while others do not. This is explained in greater detail in the description of the relevant OB.

See also

Event-based program execution (Page 975)

ATTACH: Attach an OB to an interrupt event (Page 3795)

DETACH: Detach an OB from an interrupt event (Page 3799)

Event-based program execution (S7-1200)

OB priority and runtime behavior

S7-1200-CPUs support the priority classes 1 (lowest) to 27 (highest). An OB is assigned the priority of its start event.

Interrupt OBs can only be interrupted by time error interrupts. This rule also applies to events of a priority higher than that of the currently active OB. That is, only one interrupt OB can be active, with exception of the time error interrupt OB.

Any further event of generated while an interrupt OB is being executed is added to a queue in accordance with its priority. Start events within a queue are processed later based on the chronological order of their occurrence.

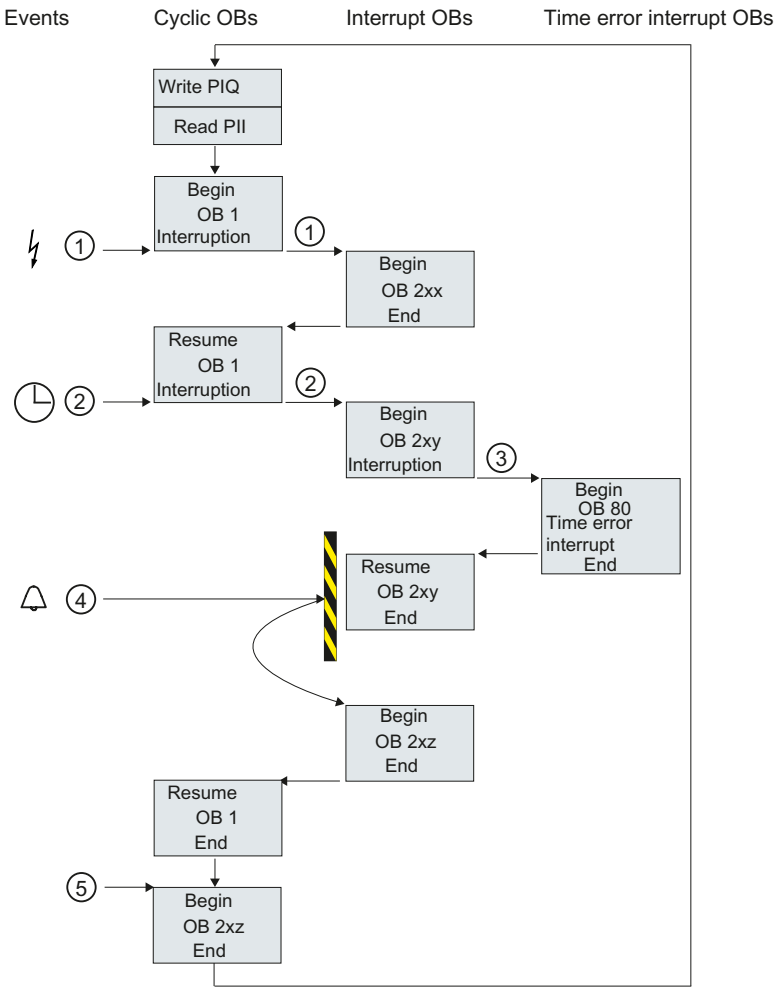
Program execution on the CPU

Cyclic OBs are interrupted by interrupt OBs.

Reactions to events which start an interrupt OB:

- For CPUs up to firmware version V3: Interrupt OBs can only be interrupted by time error interrupt OBs.
- For CPUs as of firmware version V4: Interrupt OBs can be interrupted by interrupt OBs of a higher priority.

The figure below shows the basic procedure in case interrupt OBs cannot be interrupted (behavior up to firmware version V3):



① and ② An event (e.g. a hardware interrupt) calls its associated OB.

A called OB is executed without interruption, including all of its nested blocks. Execution of the cyclic OB is resumed on completion of interrupt processing, provided the queue does not contain any events which trigger an OB start.

③ An interrupt OB can only be interrupted by a time error interrupt OB (OB 80).

④ An new alarm-triggering event occurs during interrupt processing.

Reaction for CPUs up to and including firmware version V3:

This new event is added to a queue. The queued events successively call their corresponding OBs only after execution of the current interrupt OBs was completed and according to the following rules:

- Events are processed in the order of their priority (starting at the highest priority).
- Events of the same priority are processed in chronological order.

Reaction for CPUs as of firmware version V4:

You use a CPU parameter to set the interruptibility for CPUs as of firmware version V4. Default behavior: OBs are interruptible. In this case: If the new event has a higher priority than the currently running OB, the OB started by the new event interrupts the currently running OB.

If you disable the option, the interrupt OBs cannot be interrupted.

⑤ The cyclic OBs are processed one after the other.

Notes on queues

- Every priority class (OBs of the same priority to be called) is assigned a separate queue. The size of those queues is set by default.
- Any new event leading to the overflow of a queue is discarded and therefore lost. A "time error interrupt event" is generated simultaneously. Information identifying the OB that caused the error is included in the start information of the time error interrupt OB (OB 80). A corresponding reaction such as an alarm trigger can be programmed in the time error interrupt OB.

Example of a hardware interrupt event (S7-1200)

The function principle of event-oriented program execution in the S7-1200 CPU is described based on the example of a hardware interrupt-triggering module.

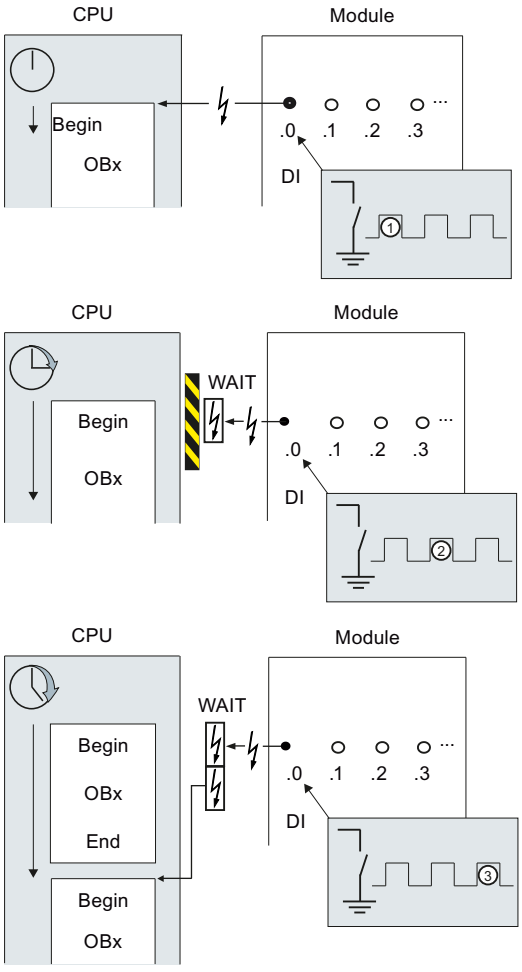
Process events and their priority

Process events are triggered by the I/O (e.g. at a digital input) and initiate a call of the assigned OB in the S7-1200 CPU. OBs assigned to a process event are called hardware interrupt OBs.

Examples of process events and their priority:

- Process events "rising edge" or "falling edge" at an interrupt-triggering module: The hardware interrupt OB started by such an event is always assigned priority 5.
- Process events from a high-speed counter
  - Count value corresponds to the reference value
  - Change count direction
  - External reset of the high-speed counter
    The hardware interrupt OB started by this event is always assigned priority 6.

The figure below shows the chronological sequence of hardware interrupt execution: In the case of two hardware interrupt events in immediate succession, the second hardware interrupt triggering event is held back in the queue until the first OBx has been processed. The next hardware interrupt triggering event can only start the associated OBx when the OBx has been processed. Additional hardware interrupt triggering events are lined up in the queue according to this principle.



Hardware interrupt execution

① A hardware interrupt-triggering event such as a rising edge at the input calls the OB to which it is assigned.

② If a new event occurs that triggers a hardware interrupt while the OB is executing, this event is entered in a queue.

③ The new event that triggers a hardware interrupt starts the hardware interrupt OB assigned to the event.

Assigning the interrupt-triggering event

The interrupt-triggering event is assigned to an OB in the input properties of the device view.

- An interrupt-triggering event can only be assigned to a single OB.
- OBs, however, can be assigned to several interrupt-triggering events.
  This means, for example, that you can assign both the rising and the falling edge to the same interrupt OB in order to trigger the same reaction to any change of the input signal.

- The started OB can interrupt a cycle OB at every instruction. Consistent data access is secured up to dword size.
- You can parameterize module-specific interrupt-triggering events such as a rising and the falling edge at the input.
- Assign the interrupt-triggering event and the OB to be started in the configuration of the interrupt-triggering module. However, within the started hardware interrupt OB you can override this assignment using the DETACH instruction, or assign the same event to a different OB using the ATTACH instruction. This functionality allows a flexible reaction to external process signals.

Setting the operating behavior (S7-1200)

Changing properties of the modules (S7-1200)

Default settings

When they leave the factory, all hardware components with parameters have default settings suitable for standard applications. These default values allow the hardware components to be used immediately without making any additional settings.

You can, however, modify the behavior and the properties of the hardware components to suit the requirements and circumstances of your application. Hardware components with settable parameters include, for example, communications modules and several analog and digital modules.

Setting and loading parameters

When you have selected a hardware component in the device or network view, you can set the properties in the Inspector window. When you save a device configuration with its parameters, data is generated that needs to be loaded on the CPU. This data is transferred to the relevant modules during startup.

Properties of the CPUs

The properties of the CPUs have special significance for system behavior. For example for a CPU you can set:

- Interfaces
- Inputs and outputs
- High-speed counters
- Pulse generators
- Startup behavior
- Time-of-day
- Protection level
- Bit memory for system and clock

- Cycle time
- Communications load

The entry possibilities specify what is adjustable and in which value ranges. Fields that cannot be edited are disabled or are not shown in the properties window.

Requirement

You have already arranged the hardware components for which you want to change properties on a rack.

Procedure

To change the properties and parameters of the hardware components, follow these steps:

1. In the device or network view, select the hardware component or interface that you want to edit.
2. Edit the settings for the selected object:
   - For example in the device view you can edit addresses and names.
   - In the Inspector window additional setting possibilities are available.

You do not need to confirm your entries, the changed values will be applied immediately.

See also

Editing properties and parameters (Page 575)

Introduction to loading a configuration (Page 1815)

CPU properties (S7-1200)

Overview of the CPU properties (S7-1200)

Overview

The following table provides you with an overview of the CPU properties:

Group	Properties	Description
General	Project information	General information to describe the inserted CPU. Except for the slot number, you can change this information.
	Catalog information	Read-only information from the hardware catalog for this CPU.
	Identification & Maintenance	For saving application-specific information such as the name of the plant and the installation location.
	Checksums	To check the identity or integrity of PLC programs. Blocks in the block folder and text lists are automatically marked with unique checksums when they are compiled. You can easily establish whether the program currently running on the CPU is the same program that you loaded a long time ago or whether the program has been changed in the meantime. The "GetChecksum" instruction is available to read out the checksum while the program is running. See: Comparison of PLC programs based on checksums (Page 5086)
PROFINET interface	General	Name and comment for this PROFINET interface. The name is limited to 110 characters.
	Ethernet addresses	Select whether the PROFINET interface is networked. If subnets have already been created in the project, they are available for selection in the drop-down list. If not, you can create a new subnet with the "Add new subnet" button. Information on the IP address, subnet mask and IP router usage in the subnet is available in the IP protocol. If an IP router is used, the information about the IP address of the IP router is necessary. See: Assigning addresses and names for PROFINET devices (Page 1767)
	Advanced options	Name, comment and additional setting options of the Ethernet interface port.
	Time synchronization	Settings for time synchronization in the NTP time format. The NTP (network time protocol) is a general mechanism for synchronizing system clocks in local and global area networks. In NTP mode, the interface of the CPU sends time queries (in client mode) at regular intervals to NTP servers on the subnet (LAN) and the addresses must be set in the parameters here. Based on the replies from the server, the most reliable and most accurate time is calculated and synchronized. The advantage of this mode is that it allows the time to be synchronized across subnets. The accuracy depends on the quality of the NTP server being used.

Group	Properties	Description
DI#/DO#	General	Name of and comment on the integrated digital inputs of the CPU.
	Digital inputs	Input delays can be set for digital inputs. The input delays can be set in groups (in each case for 4 inputs). The detection of a positive and a negative edge can be enabled for each digital input. A name and a hardware interrupt can be assigned to this event. Depending on the CPU, pulse catches can be activated at various inputs. When the pulse catch is activated, even pulse edges that are shorter than the cycle time of the program are detected.
	Digital outputs	The reaction to a mode change from RUN to STOP can be set for all digital outputs: The state can either be frozen (corresponds to retain last value) or you set a substitute value ("0" or "1")
	I/O addresses	The address space of the input and output addresses is specified as is the process image.
	Hardware identifier	The hardware identifier of the device is displayed.
AI#	General	Name of and comment on the integrated analog inputs of the CPU.
	Analog inputs	During noise reduction, the specified integration time suppresses interference frequencies at the specified frequency (in Hz). The channel address, measurement type, voltage range, smoothing and overflow diagnostics must be specified in the "Channel #" group. The measurement type and voltage range are set permanently to voltage, 0 to 10 V. Smoothing analog values provides a stable analog signal for further processing. Smoothing analog values can be useful with slow measured value changes, for example, in temperature measurement. The measured values are smoothed with digital filtering. Smoothing is achieved by the module forming mean values from a specified number of converted (digitalized) analog values. The selected level (slight, medium, strong) decides the number of analog signals used to create the mean value. If overflow diagnostics is enabled, a diagnostics event is generated if an overflow occurs.
	I/O addresses	The address space of the input addresses is specified as is the process image.
	Hardware identifier	The hardware identifier of the device is displayed.
High-speed counter (HSC)	High-speed counter (HSC)#	High-speed counters are typically used to drive counting mechanisms. See: Configuring high-speed counters (Page 995)

Group	Properties	Description
Pulse generators (PTO/PWM)	PTO#/PWM#	A pulse generator is activated and can be initialized with project information. For the configuration of an activated pulse generator, specify the usage as PWM (Pulse Width Modulation) or as PTO (Pulse Train Output). Specify the output source, time base, pulse width format, cycle time and initial pulse width for PWM. A pulse output is specified as the hardware output. The PWM output is controlled by the CTRL_PWM instruction, see CTRL_PWM (Page 3991). Specify the output source for PTO. A pulse output and a direction output are specified as the hardware outputs. A PTO is operated together with a high-speed counter in the "axis of motion" count mode and controlled by the Motion Control technology object (see keyword "Motion Control S7-1200") . The hardware ID is displayed in the I/O-diagnostics addresses and, if the PWM function is selected, the address space of the output addresses and the process image can be selected.

Group	Properties	Description
Startup	Startup after POWER ON	Setting the startup characteristics after cycling power. See: Principles of the STARTUP mode (Page 960)
	Comparison of preset configuration and actual configuration	Specifies the startup characteristics of the CPU for situations in which the actual configuration of the S7-1200 station does not correspond to the preset configuration: • Startup of the CPU only if compatible • Startup of the CPU even if there are differences With the "Startup CPU only if compatible" setting, a module in a configured slot must be compatible with the configured module. Compatible means that the module that is present matches the number of inputs and outputs and must match with respect to its electrical and functional properties. A compatible module must be able to completely replace a configured module; it may be more capable, but not less capable. Example for the "Startup CPU only if compatible" setting: A CPU can be a compatible replacement for a CPU of the same type with a lower firmware version. An input module with 32 digital inputs can be a compatible replacement for a signal module with 16 digital inputs. The CPU starts up when the configured module or a compatible module is plugged in. The CPU does not start up if an incompatible module is inserted. Example of the "Startup CPU even if mismatch" setting: Instead of a configured digital input module, an analog output module is plugged in or no module is present in this slot. Although the configured inputs cannot be accessed, the CPU starts up. Note in this case that the user program cannot function correctly and take the appropriate measures.
	Configuration time for central and distributed I/O	Specifies a maximum period (standard: 60000 ms) in which the central I/O and distributed I/O must start up. (The CMs and CPs are supplied with voltage and communication parameters during the CPU startup. This configuration time provides a time period during which I/O modules connected to the CM or CP must start up.) The CPU switches to RUN as soon as the central I/O and the distributed I/O has started and is ready for operation, regardless of the "Parameter assignment time for central and distributed I/O" parameter. If the central I/O and distributed I/O has not started up during this period, the CPU switches to RUN without the central I/O and distributed I/O.
Cycle	Maximum cycle time and minimum cycle time.	Specification of a maximum cycle time or a fixed minimum cycle time. If the cycle time exceeds the maximum cycle time, the CPU goes to STOP mode. See: Cycle time and maximum cycle time (Page 988)
Communication load	Maximum allocation of the cycle for communication (as a percentage)	Controls the duration of communication processes that always also extend the cycle time, within certain limits. Examples of communication processes include: Transferring data to another CPU or loading blocks (initiated via the PC). See: Cycle loading by communications (Page 989)

Group	Properties	Description
System and clock memory	System memory bits and clock memory bits	You use system memory bits for the following scans: • Is the current cycle the first since cycling power? • Have there been any diagnostics state changes compared with the previous cycle? • Scan for "1" (high) • Scan for "0" (low) Clock memory bits change their values at specified periodic intervals. See: Enabling system memory (Page 1006) See: Using clock memory (Page 1006)
Web server	-	Enables and configures the Web server function. See: Enabling the Web server
	Automatic update	Sends the requested web page with current CPU data periodically to the web browser. Enter the period duration under "Update interval". Automatic update can only be activated if the web server is enabled. See: AUTOHOTSPOT
	User Management	You enter the user names with their access rights here. • Everyone: pre-defined user whose access rights you assign using a drop-down list. The "Everyone" user is a user who does not log on with a password for Web access. • User#: Editable name for a user for whom you assign access rights and a password. As soon as you click in a line below "Everyone", a preset name is displayed that you can change. Each user can be assigned different access rights via a drop-down list. There are access rights that are connected to each other. Example: When you activate the access right "Perform firmware update", the access rights "Change operating mode" and "... query diagnostics" are activated automatically. These two rights are required to perform a firmware update from the Web server. See: User list
	Watch tables	Allows you to create watch tables and defines access to those tables.
	User-defined web pages	Allows access to freely-designed web pages of the CPU via a web browser. See: AUTOHOTSPOT
	Entry page	Allows you to select the entry page.
	Overview of interfaces	Table of all modules with their Ethernet interfaces providing Web server functionality for this device. Here, you can permit or deny access to the Web server over the interface for each Ethernet interface of the device (CPU, CP or CM).

Group	Properties	Description
User interface languages	-	If you want to use translations of project texts for the Web server, you need to assign the project language to the languages for the Web server (number depends on the CPU). This specifies the project languages that are displayed when the Web server is accessed. Example: You assign the project language "German (Luxembourg)" to the Web server language "German" by selecting it from the drop-down list. Enable the project language under "Languages & resources > Project languages" in the project tree to make it available for selection in the drop-down list.
Time	Local time and daylight saving time	Setting of the time zone in which the CPU is operated and setting of the daylight-saving/standard time changeover.
Protection & Security	Protection and password for read/write access	Setting the read/write protection and the password for access to the CPU. See: Setting options for the protection level (FW V1 to V3) (Page 1007) See: Setting options for the protection (FW as of V4) (Page 1008)
	Connection mechanisms	Enabling access over PUT/GET communication.
	Security event	Enabling group alarms for security events and specifying a monitoring time (interval).
	External load memory	Disables copying from the internal to the external load memory.
Configuration control	-	Enables configuration control. Allows a configuration change in the user program within certain limits. See: Configuration control (Page 1085)
Connection resources	-	Provides an overview of the assigned reserved and dynamic resources for the CPU connections. The resources currently used are also displayed in the online view. See: Use of connection resources (Page 677)
Address overview	-	Tabular representation of all addresses used by the CPU for integrated inputs/outputs as well as for the inserted modules. Addresses that are not used by any module are represented as gaps. The view can be filtered according to • Input addresses • Output addresses • Address gaps

See also

Specifying input and output addresses (Page 1046)

Assigning parameters to hardware interrupt OBs (Page 1041)

Access to the I/O addresses (Page 971)

Addressing modules (Page 1045)

Special features during startup (Page 962)

Cycle time and maximum cycle time (S7-1200)

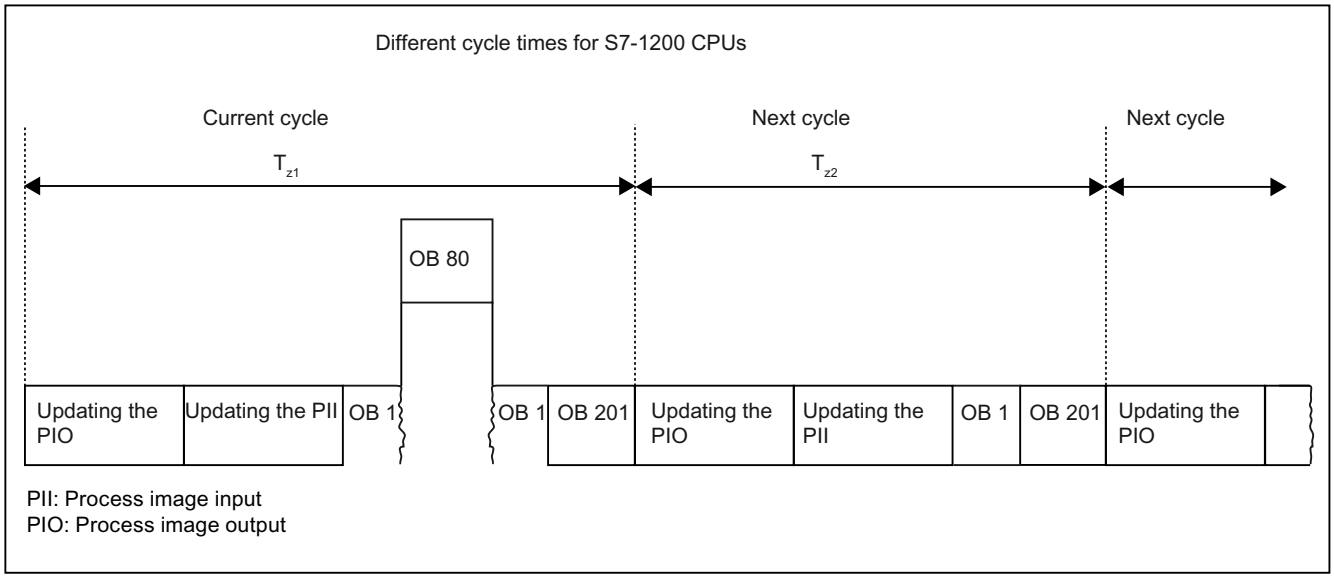
Function

The cycle time is the time that the operating system requires to execute the cyclic program and all the program sections that interrupt this cycle. The program execution can be interrupted by:

- Time errors and 2xMaxCycleTime errors
- System activities, e.g., process image updating

The cycle time (Tcyc) is therefore not the same for every cycle.

The following schematic shows an example of different cycle times (TZ1 ≠ TZ2) for S7-1200 CPUs:



In the current cycle, the cyclic OB used here (e.g. OB 1) will be interrupted by a time error (e.g. OB 80). Following the cyclic OB, the next cycle OB 201 is processed.

Maximum cycle time

The operating system monitors the execution time of the cyclic program for a configurable upper limit known as the maximum cycle time. You can restart this time monitoring at any point in your program by calling the RE_TRIGR instruction.

If the cyclic program exceeds the maximum cycle time, the operating system will attempt to start the time error OB (OB 80). If the OB is not available, the S7-1200 CPU reacts as follows:

- CPUs with firmware version V1.x to V3.x: The CPU remains in RUN mode.
- CPUs with firmware version as of V4.x: The CPU changes to STOP mode

In addition to monitoring the runtime for overshooting of the maximum cycle time, adherence to a minimum cycle time is guaranteed. To do this, the operating system delays the start of

the new cycle until the minimum cycle time has been reached. During this waiting time, new events and operating system services are processed.

If the maximum cycle time is exceeded a second time, for example while the time error OB is being processed (2xMaxCycleTime error), the CPU changes to STOP mode.

See also

Time error OB (Page 1032)

Cycle loading by communications (S7-1200)

Function

The cycle time of the CPU can be extended due to communications processes. These communications processes include for example:

- Transferring data to another CPU
- Loading of blocks initiated by a programming device

You can control the duration of these communications processes to some extent using the CPU parameter "Cycle load due to communication".

In addition to communications processes, test functions also extend the cycle time. The "Cycle load due to communication" parameter can be used to influence the duration.

How the parameter works

You use the "Cycle load due to communication" parameter to enter the percentage of the overall CPU processing capacity that can be available for communications processes. This CPU processing capacity is now available at all times for communication. This processing capacity can be used for program execution when not required for communication.

Effect on the actual cycle time

The "Cycle load due to communication" parameter can be used to extend the cycle time of the cyclic organization block (e.g., OB 1) by a factor calculated according to the following formula:

$$\frac{100}{100 - \text{"Cycle load due to communication"}}$$

The formula does not take into account the effect of asynchronous events such as hardware interrupts or cyclic interrupts on the cycle time.

If the cycle time is extended due to communication processes, more asynchronous events may occur within the cycle time of the cyclic organization block. This extends the cycle still further. The extension depends on how many events occur and how long it takes to process them.

Example 1 – no additional asynchronous events:

If the "Cycle load due to communication" parameter is set to 50%, this can cause the cycle time of the cyclic organization block to increase by up to a factor of 2.

Example 2 – additional asynchronous events:

For a pure cycle time of 500 ms, a communication load of 50% can result in an actual cycle time of up to 1000 ms, provided that the CPU always has enough communications jobs to process. If, parallel to this, a cyclic interrupt with 20 ms processing time is executed every 100 ms, this cyclic interrupt would extend the cycle by a total of 5*20 ms = 100 ms without communication load. That is, the actual cycle time would be 600 ms. Because a cyclic interrupt also interrupts communications, it affects the cycle time by adding 10 * 20 ms at 50 % communication load. That is, in this case, the actual cycle time amounts to 1200 ms instead of 1000 ms.

Note

Observe the following:

- Check the effects of changing the value of the "Cycle load due to communication" parameter while the system is running.
- You must always consider the communication load when setting the minimum cycle time as time errors will otherwise occur.

Recommendations

- Increase this value only if the CPU is used primarily for communication purposes and the user program is not time critical.
- In all other situations you should only reduce this value.

Time-of-day functions (S7-1200)

Basic principles of time of day functions (S7-1200)

All S7-1200 CPUs are equipped with an internal clock. The backup supports the display of the correct time for up to 10 hours if the power supply is interrupted.

Time-of-day format

The clock always shows the time of day with a resolution of 1 millisecond and the date including the day of the week. The time adjustment for daylight-saving time is also taken into account.

Setting and reading the time of day (S7-1200)

Setting and reading the time with instructions

You can set, start and read the time-of-day and date on the CPU clock with the following instructions in the user program:

- Set the time-of-day: "WR_SYS_T"
- Read time of day "RD_SYS_T"
- Read local time "RD_LOC_T"
- Set time zone "SET_TIMEZONE"

Manual setting

You can also read and set the time-of-day manually in the online and diagnostics view under "Functions > Set time-of-day".

See also

WR_SYS_T: Set time-of-day (Page 3467)

RD_SYS_T: Read time-of-day (Page 3469)

RD_LOC_T: Read local time (Page 3471)

SET_TIMEZONE: Set time zone (Page 3476)

Assigning the clock parameters (S7-1200)

Clock parameters

The clock parameters allow you to make the following settings:

- Enable time synchronization via NTP server
  Select this check box if you want the internal clock to be synchronized using the NTP synchronization mode.
- Network time server
  The IP addresses of up to four NTP servers need to be configured.
- Update interval
  The update interval defines the interval between time queries.

High-speed counters (S7-1200)

General information on high-speed counters (S7-1200)

Introduction

High-speed counters are typically used to drive counting mechanisms in which a shaft turning at a constant speed is equipped with an incremental step encoder. The incremental step encoder ensures a certain number of count values per rotation and a reset pulse once per rotation. The clock memory bit(s) and the reset pulse of the incremental step encoder supply the inputs for the high-speed counter.

The various S7-1200 CPUs have differing numbers of high-speed counters available:

S7-1200 CPU	Number of HSCs	HSC designation
CPU 1211C	3 (with digital signal board 4)*	HSC1…3 (and HSC5)*
CPU 1212C	4 (with digital signal board 5)*	HSC1…4 (and HSC5)*
CPU 1214C CPU 1215C CPU 1217C	6	HSC1…6

* with DI2/DO2 signal board

How it works

The first of several default values is loaded on the high-speed counter. The required outputs are enabled for the time during which the current value of the counter is lower than the default value. The counter is set up so that an interrupt occurs if the current value of the counter is equal to the default value or when the counter is reset.

If the current value is equal to the default value and an interrupt event results, a new default value is loaded and the next signal state is set for the outputs. If an interrupt event occurs because the counter is reset, the first default value and the first signal states of the outputs are set and the cycle repeated.

Since the interrupts occur much less frequently than the high-speed counter counts, a precise control of the fast operations can be implemented with only a slight influence on the overall cycle of the automation system. Since you can assign specific interrupt programs to interrupts, each new default can be loaded in a separate interrupt program allowing simple control of the state.

Note

You can also process all interrupt events in a single interrupt program.

Count algorithms of the various counters

All counters work in the same way, however some high-speed counters do not support all count algorithms. There are four basic count algorithms:

- Single-phase counter with internal direction control
- Single-phase counter with external direction control
- 2-phase counter with 2 clock inputs
- A/B counter

Each high-speed counter can be used with or without a reset input. If the reset input is activated, this resets the current value. The current value remains reset until the reset input is deactivated.

See also

Configuring high-speed counters (Page 995)

Interdependency of the counter mode and counter inputs (Page 993)

Interdependency of the counter mode and counter inputs (S7-1200)

General information on counter mode and counter inputs

You can assign not only the counter modes and counter inputs to the high-speed counters but also functions such as clock pulse generator, direction control, and reset. The following rules apply:

- An input cannot be used for two different functions.
- If an input is not required by the current counter mode of the defined high-speed counter, it can be used other purposes.

If, for example, you set HSC1 to counter mode 1, in which inputs I0.0 and I0.3 are required, you can use I0.1 for edge interrupts or for HSC2.

If, for example, you set HSC1 and HSC5, inputs I0.0 (HSC1) and I1.0 (HSC5) are always used with the counting and frequency counter modes. As a result, these two inputs are not available for any other functions when counters are operated.

You have additional inputs available if you use a digital signal board.

Overview of the interdependency of counter mode and counter inputs

Counter mode	Description	Inputs		
	HSC1	I0.0 (CPU) I4.0 (signal board)	I0.1 (CPU) I4.1 (signal board)	I0.3 (CPU) I4.3 (signal board)
	HSC2	I0.2 (CPU) I4.2 (signal board)	I0.3 (CPU) I4.3 (signal board)	I0.1 (CPU) I4.1 (signal board)
	HSC3*	I0.4 (CPU)	I0.5 (CPU)	I0.7 (CPU)
	HSC4 (CPU 1212/14/15/17C only)	I0.6 (CPU)	I0.7 (CPU)	I0.5 (CPU)
	HSC5 (CPU 1214/15/17C only)**	I1.0 (CPU) I4.0 (signal board)	I1.1 (CPU) I4.1 (signal board)	I1.2 (CPU) I4.3 (signal board)
	HSC6 (CPU 1214/15/17C only)**	I1.3 (CPU)	I1.4 (CPU)	I1.5 (CPU)
Counting / frequency	Single-phase counter with internal direction control	Clock pulse generator	-	-
Counting				Resetting
Counting / frequency	Single-phase counter with external direction control	Clock pulse generator	Direction	-
Counting				Resetting
Counting / frequency	2-phase counter with 2 clock inputs	Clock pulse generator forwards	Clock pulse generator backwards	-
Counting				Resetting
Counting / frequency	A/B counter	Clock pulse generator A	Clock pulse generator B	-
Counting				Resetting
Motion axis	Pulse generators PWM/PTO	HSC1 and HSC2 support the motion axis count mode for the PTO1 and PTO2 pulse generators: • For PTO1, HSC1 evaluates the Q0.0 outputs for the number of pulses. • For PTO2, HSC2 evaluates the Q0.2 outputs for the number of pulses. Q0.1 is used as the output for the motion direction.		

* HSC3 can only be used for CPU 1211 without a reset input

** HSC5 can be also be used for CPU 1211/12 if a DI2/DO2 signal board is used

See also

General information on high-speed counters (Page 992)

Configuring high-speed counters (Page 995)

Configuring high-speed counters (S7-1200)

Requirement

An S7-1200 CPU has been inserted in the hardware configuration.

Procedure

To configure a high-speed counter, follow these steps:

1. Select an S7-1200 CPU in the device or network view.
2. Click on the required high-speed counter under "Properties > High-speed counter (HSC)" in the Inspector window:
   - CPU 1211C: HSC1 to HSC3 (also HSC5 with a DI2/DO2 signal board)
   - CPU 1212C: HSC1 to HSC4 (also HSC5 with a DI2/DO2 signal board)
   - CPU 1214C / 1215C / 1217C: HSC1 to HSC6
3. Enable the high-speed counter in the "General" parameter group using the relevant check box.

Note

If you use a CPU 1211C or CPU 1212C with a DI2/DO2 signal board, you can also enable the high-speed counter HSC5.

Note

When you activate the pulse generators and operate them as PTO (to control stepper motors), high-speed counters are required.

With firmware versions as of V3.0, the CPU uses internal (additional) HSCs.

With firmware versions lower than V3.0, the CPU uses HSC1 for PTO1 and HSC2 for PTO2. The counters are then no longer available for other counting tasks.

In "Project information" you can enter a name and a comment for the counter.

4. Define the functionality of the counter in the "Function" parameter group:
   - Count mode: Select one of the following functions of the counter from the drop-down list:
     - Counting (decides the number of pulses)
     - Period (measurement of the period of pulses)
     - Frequency (measurement of the frequency of pulses)
     - Motion Control (use counter for motion control).
   - Operating phase: Select from the drop-down list how the counter will be operated:
     - Single phase (one clock input for counting, up and down can be set via the direction input)
     - Two-phase (one clock input for counting up, one clock input for counting down)
     - A/B counter (two clock inputs with phase-shifted pulses, single count value)
     -A/B counter fourfold (two clock inputs with phase-shifted pulses, quadruple count value)
   - Input source (only in firmware versions lower than 4.0): As the input source for the count pulses, select a CPU input or an input of a signal board.
   - Count direction is specified by: If the operating phase "Single phase" is set, you can choose whether the user program decides the count direction (instruction CTRL_HSC, parameters DIR and NEW_DIR) or a digital input.
   - Initial count direction: If the user program is set as the direction control for the count direction, you can select the count direction at the start of counting.
   - Frequency meter period: If frequency or period is set as the count mode, you can select the duration of the frequency measurement periods in the drop-down list.
5. Specify the initial values and reset condition of the counter in the "Reset to initial values" parameter group:
   - Initial counter value: Enter a start value for the counter.
   - Initial reference value: Enter a reference value.
     The current count is compared with this reference value: When the current counter value is equal to the reference value, an event is triggered (if enabled in "Event configuration").

   Here, you can also set whether the counter uses a reset input and the counter resets the current count at a high or low level at the reset input.
6. The following events can be enabled in the "Event configuration" parameter group:
   - Counter value equal to reference value
   - External reset
   - Direction change

   You can assign a hardware interrupt OB to each of these events under "Hardware interrupt". Alternatively, assign a hardware interrupt OB to every event in the user program ("ATTACH" instruction).
   Result: When an event is triggered, the CPU executes the assigned hardware interrupt OB (according to the configured priority).
   The names of the events and the hardware interrupts can be changed.

7. In the parameter group "Hardware inputs", hardware inputs are already entered. You can accept these. Alternatively, click the "..." button beside "Clock generator up", "Clock generator down" or "Reset input" and select an input.
   Note: With earlier firmware versions (lower than V4.0), the hardware inputs were preset and could not be changed.
8. In the "I/O addresses" area, the following parameters can be set:
   - Start address: The CPU writes the counter status to the address area with the set start address (data type DInt).
   - Organization block: Assignment of this address area to an organization block.
   - Process image: Assignment of the address area to a process image partition (PIP). The default setting is automatic update (PIP0).

Result

You have now adapted the parameters of the high-speed counter to the requirements of your project.

See also

General information on high-speed counters (Page 992)

Interdependency of the counter mode and counter inputs (Page 993)

Point-to-point communication (S7-1200)

Overview of point-to-point communication (S7-1200)

PtP communication is communication via a serial interface that uses standardized UART data transmission (Universal Asynchronous Receiver/Transmitter). The S7-1200 uses communications modules with an RS-232 or RS-485 interface to establish PtP communication.

Functions of point-to-point communication

Point-to-point communication (PtP) allows numerous applications:

- Direct transmission of information to an external device, for example a printer or a barcode reader
- Reception of information from external devices such as barcode readers, RFID readers, cameras and third-party optical systems as well as many other devices.
- Exchange of information with third-party devices, for example GPS devices, radio modems and many others

The Freeport protocol

The S7-1200 supports the Freeport protocol for character-based serial communication. Using Freeport communication, the data transmission protocol can be configured entirely by the user program.

Siemens provides libraries with Freeport communication functions that you can use in your user program:

- USS Drive protocol
- Modbus RTU Master protocol
- Modbus RTU Slave protocol

See also

Configuring a communications port (Page 998)

Using RS-232 and RS-485 communications modules (S7-1200)

Communications modules with RS-232 and RS-485 interfaces

In an S7-1200 CPU, you can use two different communications modules:

- RS-232 communications module
- RS-485 communications module

The communications modules can be connected to the S7-1200 CPU via the I/O channel on the left. You can plug in up to three different modules.

Properties of the communications modules

The communications modules have the following features:

- Support of the Freeport protocol
- Configuration by the user program with the aid of expanded instructions and library functions

Configuring a communications port (S7-1200)

Configuring a communications port

After you have inserted a communications module with an RS-232 or RS-485 interface, you then set the interface parameters. You set the parameters for the interface either in the properties of the interface or you control the interface parameters from the user program using the PORT_CFG instruction. The following description relates to the graphic configuration.

Note

If you use the user program to change the port setting, the settings of the graphic configuration are overwritten.

You should also keep in mind that the settings made by the user program are not retained if there is a power down.

Requirement

- A communications module is already plugged in.
- You are in the device view.

Procedure

To configure the communications port, proceed as follows:

1. Select the interface in the graphic representation in the device view.
   The properties of the interface are displayed in the Inspector window.
2. Select the "Port configuration" group in the area navigation of the Inspector window.
   The settings of the port are displayed.
3. From the "Transmission speed" drop-down list, select the speed for the data transmission.
   With user-programmed communication, remember the influence of the transmission speed on the changeover time.
4. From the "Parity" drop-down list, select the type of detection of bad information words.
5. Using the "Data bits" drop-down list, decide whether a character consists of eight or seven bits.
6. From the "Stop bit" drop-down list, select how many bits will identify the end of a transmitted word.
7. From the "Flow control" drop-down list, select the method for ensuring a trouble-free data stream between sender and receiver. This parameter can only be set for the RS-232 interface.
   - Enter a HEX value in the "XON character" box that will cause the transmission of the message to be continued when it is detected. This parameter can only be set for software-controlled data flow control.
   - Enter a HEX value in the "XOFF character" box that will cause the transmission of the message to be suspended for the set wait time. This parameter can only be set for software-controlled data flow control.
8. In the "Wait time" box, enter a wait time in ms that must be kept to after the end of the message before the next transmission can start.

Note

You can configure the interface in the network view as well. To do so, you must first select the communication module in the tabular network view and then select the interface in the Inspector window. Then you can continue as described above.

See also

Setting data flow control (Page 1000)

Setting data flow control (S7-1200)

Data flow control

Data flow control is a method that ensures a balanced send and receive behavior. In the ideal situation, the intelligent control does not allow data to be lost. It ensures that a device does not send more information than the receiving partner can process.

There are two methods of data flow control:

- Hardware-controlled data flow control
- Software-controlled data flow control

With both methods, the DSR signals of the communications partners must be active at the beginning of transmission. If the DSR signals are inactive, the transmission is not started.

The RS-232 communications module can handle both methods. The RS-485 communications module does not support data flow control.

Hardware-controlled data flow control

Hardware-controlled data flow control uses the request to send (RTS) and clear to send (CTS) signals. With the RS-232 communications module, the RTS signal is transmitted via the output by pin 7. The CTS signal is received via pin 8.

If hardware-controlled data flow control is enabled, the RTS signal is then always set to activated when data is sent. At the same time, the CTS signal is monitored to check whether the receiving device can accept data. If the CTS signal is active, the module can transfer data until the CTS signal becomes inactive. If the CTS signal is inactive, the data transfer must be suspended for the set wait time. If the CTS signal is still inactive after the set wait time, the data transfer is aborted and an error is signaled back to the user program.

Data flow control using hardware handshaking

If data flow control is controlled by hardware handshaking, the sending device sets the RTS signal to active as default. A device such as a modem can then transfer data at any time. It does not wait for the CTS signal of the receiver. The sending device itself monitors it own transmission by sending only a limited number of frames (characters), for example to prevent overflow of the receive buffer. If there is nevertheless an overflow, the transferring device must hold back the message and signal an error back to the user program.

Software-controlled data flow control

Software-controlled data flow control uses certain characters within the messages and these control the transfer. These characters are ASCII characters selected for XON and XOFF.

XOFF indicates when a transmission must be suspended. XON indicates when a transmission can be continued.

If the sending device receives the XOFF character, it must suspend sending for the selected wait time. If the XON character is sent after the selected wait time, the transfer is continued. If no XON character is received after the wait time, an error is signaled back to the user program.

Software data flow control requires full duplex communication because the receiving partner needs to send the XON character during the ongoing transfer.

See also

Configuring a communications port (Page 998)

Configuration of message transfer (S7-1200)

User-programmed communication

You can control the data traffic between a communications module and a device connected externally via the serial interface using your own mechanisms. If you want to do this, you will need to define a communications protocol yourself. In freely programmable communication, ASCII and binary protocols are supported for message transfer.

Within the communications protocol, you will need to specify the criteria by which the start and end of a transferred message can be recognized in the data stream.

User-programmed communication can only be activated in RUN mode. If there is a change to STOP mode, the user-programmed communication is stopped.

Specifying the communications protocol

You can specify the communications protocol as follows:

- With the user program
  - The behavior when sending data is controlled by the SEND_CFG instruction.
  - The behavior when receiving data is controlled by the RCV_CFG instruction.
- Using parameter settings set graphically in the Inspector window

Note

If you change the communications protocol from the user program, the settings of the graphic configuration are overwritten.

You should keep in mind that the settings made by the user program are not retained if there is a power down.

See also

User-programmed communication with RS-232 devices (Page 1002)

Making the settings for sending (Page 1003)

Specifying the start of the message (Page 1004)

Specifying the end of the message (Page 1005)

User-programmed communication with RS-232 devices (S7-1200)

RS-232/PIP multi-master cable and user-programmed communication with RS-232 devices

Using the RS-232/PIP multi-master cable and user-programmed communication, you can connect a wide variety of RS-232-compliant devices to the communications modules of the S7-1200. The cable must, however, be set to the "PIP/user-programmed communication" mode.

Settings on the cable

The switches on the cable must be set as follows:

- Switch 5 must be set to 0
- Switch 6 sets either the local mode (DCE) or the remote mode (DTE):
  - Switch set to 0 for the local mode
  - Switch set to 1 for the remote mode

Changing over between send and receive mode

The RS-232/PIP multi-master cable is in send mode when data is sent from the RS-232 interface to the RS-485 interface. The cable is in receive mode when it is idle or when data is sent from the RS-485 interface to the RS-232 interface. The cable changes from receive to send mode immediately when it detects characters on the RS-232 send line.

Supported transmission speeds

The RS-232/PIP multi-master cable supports transmission rates between 1200 baud and 115.2 kbaud. The RS-232/PIP multi-master cable can be set to the required transmission rate using the DIP switch on the PC/PIP cable.

The following table shows the switch settings for the various transmission speeds:

Transmission speed	Switchover time	Settings (1 = up)
115200 bps	0.15 ms	110
57600 bps	0.3 ms	111
38400 bps	0.5 ms	000
19200 bps	1.0 ms	001
9600 bps	2.0 ms	010
4800 bps	4.0 ms	011
2400 bps	7.0 ms	100
1200 bps	14.0 ms	101

The cable returns to receive mode when the RS-232 send line is idle for a certain time that is defined as the changeover time of the cable. The set transmission speed influences the changeover time as shown in the table.

Influence of the changeover time

When working with an RS-232/PIP multi-master cable in a system in which user-programmed communication is used, the program must take into account the changeover time for the following reasons:

- The communications module reacts to messages sent by the RS-232 device.
  Once the communications module has received a request from the RS-232 device, it must delay the reaction message for a period that is equal to or longer than the changeover time of the cable.
- The RS-232 device reacts to messages sent by the communications module.
  Once the communications module has received a reaction message from the RS-232 device, it must delay the next request message for a period that is equal to or longer than the changeover time of the cable.

In both situations, the RS-232-PIP multi-master cable has enough time to change from send to receive mode so that the data can be sent from the RS-485 interface to the RS-232 interface.

See also

Configuration of message transfer (Page 1001)

Making the settings for sending (Page 1003)

Specifying the start of the message (Page 1004)

Specifying the end of the message (Page 1005)

Making the settings for sending (S7-1200)

Sending messages

You can program pauses between individual messages.

The following table shows which pauses can be set:

Parameter	Definition
RTS ON delay	You can set the time that must elapse after the send request RTS (request to send) before the actual data transfer starts.
RTS OFF delay	You can set the time that must elapse after the complete transfer before RTS signal is deactivated.
Send pause at the start of the message	You can specify that a pause is sent at the start of every message transfer when the RTS ON delay has elapsed. The pause is specified in bit times.
Send Idle Line after a pause	You can make a setting so that following a selected pause at the start of the message, the "Idle Line" signal is output to signal that the line is not in use. To enable the parameter, "Send pause at message start" must be set. The duration of the "Idle Line" signal is specified in bit times.

See also

Specifying the start of the message (Page 1004)

Specifying the end of the message (Page 1005)

User-programmed communication with RS-232 devices (Page 1002)

Specifying the start of the message (S7-1200)

Recognizing the start of the message

To signal to the receiver when the transfer of a message is completed and when the next message transfer starts, criteria must be specified in the transmission protocol to identify the end and start of a message.

If a criterion is met that indicates the start of a message, the receiver starts searching the data stream for criteria that mean the end of the message.

There are two different methods for identifying the start of a message:

- Starting with any character:
  Any character can defined the start of a message. This is the default method.
- Starting with a specific condition:
  The start of a message is identified based on selected conditions.

Conditions for detecting the start of a message

The following table shows the various options for defining the start of a message:

Parameter	Definition
Recognize start of message by line break	The receiver recognizes a line break when the received data stream is interrupted for longer than one character. If this is the case, the start of the message is identified by the line break.
Recognize start of message by idle line	The start of a message is recognized when the send transmission line is in the idle state for a certain time (specified in bit times) followed by an event such as reception of a character.
Recognize start of message with individual characters	The start of a message is recognized when a certain character occurs. You can enter the character as a HEX value.
Recognize start of message by a character string	The start of a message is detected when one of the specified character sequences arrives in the data stream. You can specify up to four character sequences each with up to five characters.

The individual conditions can be logically linked in any way.

See also

Making the settings for sending (Page 1003)

User-programmed communication with RS-232 devices (Page 1002)

Specifying the end of the message (S7-1200)

Recognizing the end of the message

To signal to the receiver when the transfer of a message is completed and when the next message transfer starts, criteria must be specified in the transmission protocol to identify the end and start of a message.

In total, there are six different methods of recognizing the end of a message and these can all be logically linked in any way. The following table shows the various possible setting options:

Parameter	Definition
Recognize end of message by message timeout	The end of a message is recognized automatically when a selected maximum duration for a message is exceeded. Values from 0 to 65535 ms can be set.
Recognize end of message by reply timeout	The end of a message is recognized when there is no reply within a set time after transferring data. Values from 0 to 65535 ms can be set.
Recognize end of message by timeout between characters	The end of a message is detected when the time between two characters specified in bit times is exceeded. Values from 0 to 2500 bit times can be set. The S7-1200 CPU only accepts a maximum time of eight seconds even if the value that is set results in a duration of more than eight seconds.
Recognize end of message by maximum length	The end of a message is recognized when the maximum length of a message is exceeded. Values from 1 to 1023 characters can be set.
Read message length from message	The message itself contains information about the length of the message. The end of a message is reached when the value taken from the message is reached. Which characters are used for the evaluation of the message length is specified with the following parameters: • Offset of the length field in the message The value decides the position of the character in the message that will be used to indicate the message length. Values from 1 to 1022 characters can be set. • Size of the length field This value specifies how many characters starting at the first evaluation position will be used to indicate the message length. Values of 0, 1, 2 and 4 characters can be set. • The data following the length field (does not belong to the message length) The value specifies the number of bytes after the length field that must be ignored in the evaluation of the message length. Values from 0 to 255 characters can be set.
Recognize message end with a character sequence	The end of a message is detected when the specified character sequence arrives in the data stream. You can define up to five characters in the character string that are to be checked. If the specified characters appear at the correct location in the message, the message end is recognized. To recognize the message end when character 1 and character 3 have a certain value, for example, you have to activate the check box for character 1 and character 3 and enter a character value.

See also

Making the settings for sending (Page 1003)

User-programmed communication with RS-232 devices (Page 1002)

Enabling system memory (S7-1200)

System memory

A system memory is a bit memory with defined values.

You decide which memory byte of the CPU will become the system memory byte when assigning the system memory parameters.

Benefits

You can use system memory in the user program, for example to run program segments in only the first program cycle after start-up. Two system memory bits are constant 1 or constant 0.

Bits of the system memory bytes

The following table shows the meaning of the system memory:

Bit of the system memory bytes	7	6	5	4	3	2	1	0
Meaning	Reserved (=0)	Reserved (=0)	Reserved (=0)	Reserved (=0)	=0	=1	=1 with change to the diagnosis status	=1 in first program cycle after startup, otherwise 0

Note

The selected memory byte cannot be used for intermediate storage of data.

Using clock memory (S7-1200)

Clock memory

A clock memory is a bit memory that changes its binary status periodically in the pulse-no-pulse ratio of 1:1.

You decide which memory byte of the CPU will become the clock memory byte when assigning the clock memory parameters.

Benefits

You can use clock memory, for example, to activate flashing indicator lamps or to initiate periodically recurring operations such as recording of actual values.

Available frequencies

Each bit of the clock bit memory byte is assigned a frequency. The following table shows the assignment:

Bit of the clock memory byte	7	6	5	4	3	2	1	0
Period (s)	2.0	1.6	1.0	0.8	0.5	0.4	0.2	0.1
Frequency (Hz)	0.5	0.625	1	1.25	2	2.5	5	10

Note

Clock memory runs asynchronously to the CPU cycle, i.e. the status of the clock memory can change several times during a long cycle.

The selected memory byte cannot be used for intermediate storage of data.

Protection (S7-1200)

Setting options for the protection level (FW V1 to V3) (S7-1200)

Protection level

The following section describes how to use the various protection levels of the S7-1200 CPUs V1 to V3.

Effects of the protection level setting

You can choose between the following protection levels:

- No protection: This corresponds to the default behavior. You cannot enter a password. Read and write access is always permitted.
- Write protection: Only read-only access is possible. You cannot change any data on the CPU and cannot load any blocks or a configuration. HMI access and communication between CPUs are excluded from the write protection. Assignment of a password is required to select this protection level.
- Write/read protection: No write or read access is possible in the "Accessible devices" area or in the project for devices that are switched online. Only the CPU type and the identification data can be displayed in the project tree under "Accessible devices". Display of online information or blocks under "Accessible devices", or in the project for devices interconnected online, is possible.
  HMI access and communication between CPUs are excluded from the write protection. Assignment of a password is required to select this protection level.

Behavior of a password-protected CPU during operation

The CPU protection takes effect after the settings are downloaded to the CPU.

Validity is checked before the online function is executed. If password protection is in place, you are prompted to enter a password.

Example: The module was assigned write protection and you want to execute the "Modify tags" function. This requires write access; therefore, the assigned password must be entered to execute the function.

The functions protected by a password can only be executed by one programming device/PC at any one time. Another programming device/PC cannot log on with a password.

Access authorization to the protected data is in effect for the duration of the online connection or until the access authorization is manually rescinded with "Online > Delete access rights". Access authorization will also expire when the project is closed.

Note

You cannot restrict functions for process control, monitoring, and communications.

However, some functions are protected due to their use as online data. RUN/STOP in the "Online tools" task card or "Set time of day" in the online and diagnostics view are therefore write-protected.

Setting options for the protection (FW as of V4) (S7-1200)

Protection level

The following section describes how to use the various access levels of the S7-1200 CPUs as of V4.

S7-1200 CPUs provide various access levels to limit the access to specific functions.

The parameters for the access levels are assigned in a table. The green checkmarks in the columns to the right of the respective access level specify which operations are possible without knowing the password of this access level. If you want to use the functions of check boxes that are not selected, a password has to be entered.

NOTICE
Configuring an access level does not replace know-how protection Configuring access levels prevents unauthorized changes to the CPU by restricting download privileges. However, blocks on the memory card are not write- or read-protected. Use know-how protection to protect the code of blocks on the memory card.

Default characteristics

The default access level is "Full access (no protection)". Every user can read and change the hardware configuration and the blocks. A password is not set and is also not required for online access.

The access levels in detail

With an S7-1200 CPU, you can configure the following access levels:

- Full access (no protection): The hardware configuration and the blocks can be read and changed by all users.
- Read access: With this access level, only read access to the hardware configuration and the blocks is possible without entering a password - meaning that you can load the hardware configuration and blocks into the programming device. In addition, HMI access and access to diagnostics data is possible.
  You cannot load blocks or a hardware configuration into the CPU without entering the password. Moreover, writing test functions and firmware updates are **not** possible without a password.
- HMI access: With this access level, only HMI access and access to diagnostics data is possible without entering the password.
  Without entering the password, you can neither load blocks and hardware configuration into the CPU, nor load blocks and hardware configuration from the CPU into the programming device. In addition, the following is **not** possible without a password: Writing test functions, changing the operating state (RUN/STOP) and firmware updates.
- No access (complete protection): When the CPU is completely protected, no read or write access to the hardware configuration and the blocks is possible. HMI access is also not possible. The server function for PUT/GET communication is disabled in this access level (cannot be changed).
  Authorization with the password again provides you full access to the CPU.

Behavior of a password-protected module during operation

The CPU protection takes effect after the settings are downloaded to the CPU.

Validity is checked before the online function is executed. If password protection is in place, you are prompted to enter a password.

Example: The module was configured with read access and you want to execute the "Modify tags" function. This requires write access; therefore, the assigned password must be entered to execute the function.

The functions protected by a password can only be executed by one programming device/PC at any one time. Another programming device/PC cannot log on.

Access authorization to the protected data is in effect for the duration of the online connection or until the access authorization is manually rescinded with "Online > Delete access rights".

Each access level allows unrestricted access to certain functions without entering a password, for example, identification using the "Accessible devices" function.

Configuring access levels (S7-1200)

The following section describes how to configure an access level and enter passwords for an S7-1200 CPU as of V4.

For an S7-1200 CPU, you can enter multiple passwords and thereby set up different access rights for individual user groups.

The passwords are entered in a table in such a way that exactly one access level is assigned to each password.

The effect of the password is given in the "Access level" column.

Example

You select the "No access (complete protection)" access level for a standard CPU (in other words, not an F-CPU) and enter a separate password for each of the access levels that lie above it in the table.

For users who do not know any of the passwords, the CPU is completely protected. Not even HMI access is possible.

For users who know one of the set passwords, the effect depends on the table row in which the password occurs:

- The password in row 1 (Full access (no protection)) allows access as if the CPU were completely unprotected. Users who know this password have unrestricted access to the CPU.
- The password in row 2 (read access) allows access as if the CPU were write-protected. Users who know this password have read-only access to the CPU.
- The password in row 3 (HMI access) allows access as if the CPU were write-protected and read-protected so that only HMI access is possible for users who know this password.

Procedure

To configure the access levels of an S7-1200 CPU, follow these steps:

1. Open the properties of the module in the inspector window.
2. Open the "Protection" entry in the area navigation.
   A table with the possible access levels appears in the inspector window.

Protection level	Access			Access permission	
Protection	HMI	Read	Write	Password	Confirmation
Full access (no protection)	✓	✓	✓		
Read access	✓	✓			
HMI access	✓				
No access (complete protection)					

3. Activate the required access level in the first column of the table. The green checkmarks in the columns to the right of the respective protection level show you which operations are still available without entering the password.
4. In the "Password" column, specify a password for full access in the first row. In the "Confirmation" column, enter the selected password again to protect against incorrect entries.
   Ensure that the password is sufficiently secure, in other words, that is does not follow a machine-recognizable pattern!
   You must enter a password in the first row "Full access (no protection)". This enables unrestricted access to the CPU for those who know the password, regardless of the selected protection level.
5. Assign additional passwords as needed to other access levels if the selected protection level allows you to do so.
6. Download the hardware configuration so that the access level takes effect.

Result

The hardware configuration and the blocks are protected against unauthorized access according to the set access level. If an operation cannot be executed without a password due to the set access level, a dialog for entering a password is displayed.

Restriction of communication services (S7-1200)

Introduction

The CPU can be the server for a number of communication services. This means that other communication participants can access CPU data even if you have not configured and programmed connections for the CPU.

The local CPU as a server thus does not have the possibility to control communication to the clients.

The parameter "Connection mechanisms" in the "Protection" area of the CPU parameters is used to specify whether this type of communication is permitted or not for the local CPU during operation.

Permit access with PUT/GET communication from remote partners

By default, the "Permit access with PUT/GET communication from remote partners (...)" option is disabled. In this case, read and write access to CPU data is only possible for communication connections that require configuration or programming both for the local CPU and for the communication partner. Access through BSEND/BRCV instructions is possible, for example.

Connections for which the local CPU is only a server (meaning that no configuration/ programming of the communication with the communication partner exists at the local CPU), are therefore not possible during operation of the CPU, for example,

- For PUT/GET, FETCH/WRITE or FTP access via communication modules
- For PUT/GET access from other S7 CPUs
- For HMI access that is realized via PUT/GET communication

If you want to allow access to CPU data from the client side, meaning that you do not want to restrict the communication services of the CPU, activate the "Permit access with PUT/GET communication from remote partners" option.

Group alarms for security events (S7-1200)

What you should know about group alarms for security events

Security events in the diagnostics buffer

The CPU stores information on important actions and events in the diagnostics buffer.

The following security events (event types) lead to an entry in the diagnostics buffer of the S7-1200:

- Going online with the correct or incorrect password.
- Manipulated communications data detected.
- Manipulated data detected on memory card.
- Manipulated firmware update file detected.
- Changed protection level (access protection) downloaded to the CPU.
- Password legitimization restricted or enabled (by instruction or CPU display).
- Online access denied due to the possible number of simultaneous access attempts being exceeded.
- Timeout when an existing online connection is inactive.
- Logging in to the Web server with the correct or incorrect password.
- Creating a backup of the CPU.

- Restoring the CPU configuration.
- During startup:
  - Project on the SIMATIC memory card has changed (the SIMATIC memory card remains the same)
  - SIMATIC memory card was replaced

Group alarm for security events

To prevent the diagnostics buffer being "swamped" by large numbers of identical security events, as of STEP 7 V13 service pack 1, you can set parameters so that these events are entered in the diagnostics buffer as a group alarm. Per interval (monitoring time), the CPU then only generates one group alarm per event type.

Example: Brute-Force attacks

With Brute-Force attacks, attackers attempt to gain access to the CPU by systematically trying out a large number of the possible password combinations. As a result, the CPU receives a large number of login requests within seconds and the inrush of identical single entries in the diagnostics buffer leads to the loss of important diagnostics information. Group alarms help here.

Principle of operation of group alarms

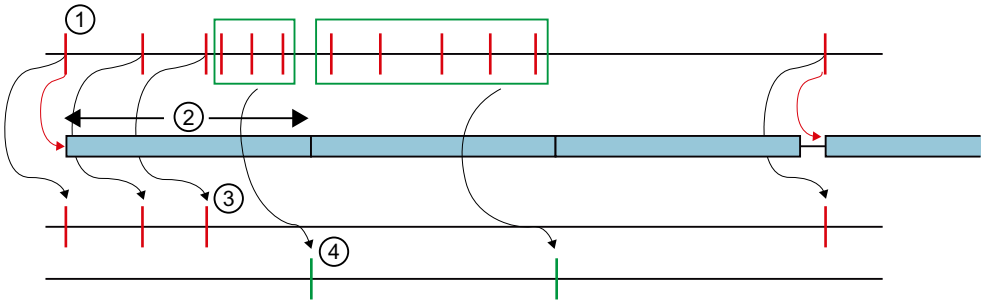
The CPU enters the first three events of an event type in the diagnostics buffer. It then ignores all subsequent security events of this type.

At the end of the monitoring time (interval), the CPU generates a group alarm in which it enters the event type and the frequency of the event during the elapsed interval.

If these security events also occur in the intervals that follow, the CPU only generates one group alarm per interval.

An attack leaves the following pattern in the diagnostics buffer: Three individual alarms followed by a series of group alarms. This series can consist of two, three or more group alarms depending on the selected monitoring time and duration of the attack.

Since the alarms in the diagnostics buffer have a time stamp, the duration of the attack can be determined.



① Security events of one type, for example login attempts with an invalid password.

② Interval (monitoring time): When a security event of a particular type occurs the first time (or reoccurs), the monitoring time is started (or restarted).

③ Single alarms: The first three security events of a type are entered in the diagnostics buffer immediately. As of the fourth security event of this type, the CPU only creates group alarms.

If a security event of this type occurs after a pause of at least one interval, the CPU enters a single alarm in the diagnostics buffer and restarts the monitoring time.

④ Group alarms: After three security events, the CPU only generates a group alarm as a summary of all other security events in this interval. If these security events also occur in the intervals that follow, the CPU only generates one group alarm per interval.

Configuring a group alarm for security events (S7-1200)

Requirements

- STEP 7 V14
- S7-1200-CPU as of firmware V4.2

Procedure

To configure group alarms for security events, follow these steps:

1. Click on the CPU symbol in the network view.
   The properties of the CPU are displayed in the Inspector window.
2. Go to the "Protection" > "Security event" area.
3. Click on "Security event".
4. Select the option "Summarize security events in case of high message volume" to enable group alarms for security events.
5. Set the duration of an interval (monitoring time); the default is 20 seconds.

See also

What you should know about group alarms for security events (Page 1012)

Deactivating SNMP (S7-1200)

The network management protocol SNMP (Simple Network Management Protocol) is a protocol that uses various services and tools for detection and diagnostics of the network topology. SNMP uses the transport protocol UDP. SNMP recognizes two network components, the SNMP manager and the SNMP client. The SNMP manager monitors the network nodes: The SNMP clients collect the various network-specific information in the individual network nodes and store it in a structured form in the MIB (Management Information Base). Various services and tools can run detailed network diagnostics with the help of this data.

Under certain conditions, it may be useful to disable SNMP. Examples:

- The security guidelines in your network do not allow the use of SNMP.
- You use your own SNMP solution, e.g. with your own communications instructions.

If you disable SNMP for a device, various options for diagnostics of the network topology (e.g. using the PRONETA tool or the Web server of the CPU) are no longer available to you.

Requirements

- CPU S7-1200 V4.1 and higher
- CPU S7-1500 V1.8 and higher

Disabling SNMP

To disable SNMP, follow these steps:

1. In STEP 7, create a data block that contains the structure of data record $F003_H$. The following table shows the structure of the data record $F003_H$.

Byte	Element	Code	Explanation
0-1	BlockID	$F003_H$	Header
2-3	BlockLength	8	The data record length is counted starting at byte 4 "Version".
4	Version	01_H	
5	Subversion	00_H	
6-7	Reserved	-	-
8-11	SNMP controller	Disable/enable SNMP	If you want to disable SNMP, enter the value 0. If you want to enable SNMP, enter the value 1.

2. Transfer the data record $F003_H$ in the startup OB (OB100) to the CPU with the WRREC instruction (write data record). Use an integrated interface of the CPU as the hardware ID.

Deactivating SNMP - complete example (S7-1200)

Introduction

The example applies to S7-1500 and S7-1200.

Follow these steps to apply the program code to your project:

1. Copy the entire program code to the clipboard.
2. Open a text editor (e.g. "Editor").
3. Paste the content of the clipboard to the text editor with Ctrl+V. Save the document as scl file, for example, SNMP_DEACT.scl.
4. Open your project in STEP 7.
5. Import the scl file as an external source.
6. Generate the startup OB and the data blocks.
7. Right-click the scl file and select "Generate block from source" in the shortcut menu.

Example program with WRREC call in OB 100

```
DATA_BLOCK "Deactivate_SNMP"
{ S7_Optimized_Access := 'TRUE' }
VERSION : 0.1
NON_RETAIN
 VAR
   snmp_deactivate : Bool;
   snmp_record : Struct
     BlockID : UInt;
     BlockLength : UInt;
     "Version" : USInt;
     Subversion : USInt;
     Reserved : UInt;
     SNMPControl : UDInt;
   END_STRUCT;
   snmp_done : Bool;
   snmp_error : Bool;
   snmp_Status : DWord;
 END_VAR

BEGIN
  snmp_deactivate := true;
  snmp_record.BlockID := 16#f003;
  snmp_record.BlockLength := 8;
  snmp_record."Version" := 1;
END_DATA_BLOCK

ORGANIZATION_BLOCK "Startup"
TITLE = "Complete Restart"
{ S7_Optimized_Access := 'TRUE' }
VERSION : 0.1
BEGIN
  REPEAT
    "WRREC_DB_1"(REQ := "Deactivate_SNMP".snmp_deactivate,
                 ID := "Local~PROFINET-Schnittstelle_1",
                 INDEX := 16#b071,
                 DONE => "Deactivate_SNMP".snmp_done,
                 ERROR => "Deactivate_SNMP".snmp_error,
```

```
                STATUS => "Deactivate_SNMP".snmp_Status,
                RECORD := "Deactivate_SNMP".snmp_record);
   UNTIL "Deactivate_SNMP".snmp_done OR "Deactivate_SNMP".snmp_error
   END_REPEAT;
END_ORGANIZATION_BLOCK

DATA_BLOCK "WRREC_DB_1"
{OriginalPartName := 'WRREC';
 VersionGUID := 'bc169451-58cd-44a3-855b-3f78cc0623c8';
 S7_Optimized_Access := 'TRUE' }
AUTHOR : SIMATIC
FAMILY : DP
NAME : WRREC
VERSION : 1.0
NON_RETAIN
WRREC
BEGIN
END_DATA_BLOCK
```

Symbolic and numerical names of instructions (S7-1200)

Description

The instructions from the task card are comprised of functions (FC), function blocks (FB), system functions (SFC) and system function blocks (SFB) that are identified internally by numbers.

The following tables show the assignment of numerical and symbolic names.

Function blocks (FBs)

Numerical name	Symbolic name
FB 105	TC_CONFIG
FB 110	Port_Config
FB 111	Send_Config
FB 112	Receive_Config
FB 113	Send_P2P
FB 114	Receive_P2P
FB 115	Receive_Reset
FB 116	Signal_Get
FB 117	Get_Features
FB 118	Set_Features
FB 163	TC_SEND
FB 164	TC_RECV
FB 165	TC_CON
FB 166	TC_DISCON
FB 804	SET_TIMEZONE

Numerical name	Symbolic name
FB 1030	TSEND_C
FB 1031	TRCV_S
FB 1071	USS_DRIVE
FB 1080	MB_COMM_LOAD
FB 1081	MB_MASTER
FB 1082	MB_SLAVE
FB 1084	MB_CLIENT
FB 1085	MB_SERVER
FB 1100	MB_Halt
FB 1101	MC_Home
FB 1102	MC_MoveAbsolute
FB 1103	MC_MoveJog
FB 1104	MC_MoveRelative
FB 1105	MC_MoveVelocity
FB 1107	MC_Power
FB 1108	MC_Reset
FB 1110	MC_MoveInterrupt
FB 1111	MC_ChangeDynamik
FB 1112	MC_CommandTable
FB 1113	MC_MoveLinearAbs_2D
FB 1114	MC_MoveLinearRel_2D
FB 1115	MC_MoveCircular_2D
FB 1130	PID_Compact
FB 1134	PID_3Step
FB 1140	HSC
FB 2040	RecipeCreate
FB 2041	RecipeOpen
FB 2042	RecipeRead
FB 2043	RecipeWrite
FB 2044	RecipeAppend
FB 2045	RecipeClose

Functions (FCs)

Numerical name	Symbolic name
FC 2 (1)	CONCAT
FC 4 (1)	DELETE
FC 11 (1)	FIND
FC 17 (1)	INSERT
FC 20 (1)	LEFT
FC 21 (1)	LEN
FC 22 (1)	LIMIT

Numerical name	Symbolic name
FC 25 (1)	MAX
FC 26 (1)	MID
FC 27 (1)	MIN
FC 31 (1)	REPLACE
FC 32 (1)	RIGHT
FC 36 (1)	ENCO
FC 36 (1)	SEL
FC 37	DECO
FC 800	LED
FC 801	IM_DATA
FC 802	DeviceStates
FC 803	ModuleStates
FC 1070	USS_PORT
FC 1072	USS_RPM
FC 1073	USS_WPM
(1) MC7+ instruction	

System data types (SDTs)

Numerical name	Symbolic name
SDT 99	WWW_CDB
SDT 513	CONDITIONS
SDT 581	Send_Conditions
SDT 582	Receive_Conditions

System function blocks (SFBs)

Numerical name	Symbolic name
SFB 0 (1)	CTU
SFB 1 (1)	CTD
SFB 2 (1)	CTUD
SFB 3 (1)	TP
SFB 4 (1)	TON
SFB 5 (1)	TOF
SFB 27	START_OB
SFB 52	RDREC
SFB 53	WRREC
SFB 54	RALRM
SFB 105	T_CONFIG
SFB 106	TDIAG
SFB 107	TRESET
SFB 110	PORT_CFG

Numerical name	Symbolic name
SFB 111	SEND_CFG
SFB 112	RCV_CFG
SFB 113	SEND_PTP
SFB 114	RCV_PTP
SFB 115	SGN_GET
SFB 116	SGN_SET
SFB 117	RCV_RST
SFB 120	CTRL_HSC
SFB 122	CTRL_PWM
SFB 124	CTRL_HSC_EXT
SFB 140	DataLogCreate
SFB 141	DataLogOpen
SFB 142	DateLogWrite
SFB 143	DataLogClear
SFB 144	DataLogClose
SFB 145	DataLogDelete
SFB 146	DataLogNewFile

System functions (SFCs)

Numerical name	Symbolic name
SFC 7	DP_PRAL
SFC 11	DPSYC_FR
SFC 13	DPNRM_DG
SFC 14	DPRD_DAT
SFC 16	RD_OBINF
SFC 23	DEL_DB
SFC 28	SET_TINT
SFC 29	CAN_TINT
SFC 30	ACT_TINT
SFC 31	QRY_TINT
SFC 32	SRT_DINT
SFC 33	CAN_DINT
SFC 34	QRY_DINT
SFC 41	DIS_AIRT
SFC 42	EN_AIRT
SFC 43	RE_TRIGR
SFC 45	D_ACT_DP
SFC 46	STP
SFC 82	CREA_DBL
SFC 83	READ_DBL
SFC 84	WRIT_DBL

Numerical name	Symbolic name
SFC 86	CREATE_DB
SFC 89	RST_EVOV
SFC 99	WWW
SFC 101	RTM
SFC 117	GET_DIAG
SFC 124	ATTR_DB
SFC 140	IO2MOD
SFC 143	RD_ADDR
SFC 154	RD_LOC_T
SFC 154	DPWR_DAT
SFC 161	WR_LOC_T
SFC 180	ID2LOG
SFC 181	LOG2ID
SFC 182	ID2GEO
SFC 190	SET_CINT
SFC 191	QRY_CINT
SFC 192	ATTACH
SFC 193	DETACH
MC7+ Anweisung	GET_ERROR
MC7+ Anweisung	GET_ERR_ID

Organization blocks (S7-1200)

Startup OBs (S7-1200)

Description

You can determine the boundary conditions for the startup characteristics of your CPU, for example, the initialization values for "RUN". To do this, write a startup program. The startup program consists of one or more startup OBs (OB numbers 100 or >= 123).

The startup program is executed once during the transition from "STOP" mode to "RUN" mode. Current values from the process image of the inputs are not available for startup program, nor can these values be set.

After the complete execution of the startup OBs, the process image of the inputs is read in and the cyclic program is started.

There is no time limit for executing the startup routine. Therefore the scan cycle monitoring time is not active. Time-driven or interrupt-driven organization blocks cannot be used.

Start information

A startup OB has the following start information:

Tag	Data type	Description
LostRetentive	BOOL	= 1, if retentive data storage areas have been lost
LostRTC	BOOL	= 1, if realtime clock has been lost

See also

Events and OBs (Page 973)

Cyclic OBs (S7-1200)

Introduction

For the program execution to start, at least one program cycle OB must be present in the project. The operating system calls this program cycle OB once in each cycle and thereby starts the execution of the user program. You can use multiple OBs (OB numbers >= 123). When multiple program cycle OBs are used, these are called in one after the other in the order of their OB numbers. The program cycle OB with the lowest OB number is called first.

The program cycle OBs have the priority class 1. This corresponds to the lowest priority of all OBs. The cyclic program can be interrupted by events of any other event class.

Programming cyclic program execution

You program cyclic program execution by writing your user program in the cycle OBs and the blocks that they call.

The first cyclic program execution begins as soon as the startup program has ended without errors. The cycle restarts after the end of each cyclic program execution.

Sequence of cyclic program execution

One cycle of the program execution encompasses the following steps:

1. The operating system starts the maximum cycle time.
2. The operating system writes the values from the process image output to the output modules.
3. The operating system reads out the state of the inputs of the input modules and updates the process image input.
4. The operating system processes the user program and executes the operations contained in the program.
5. At the end of a cycle, the operating system executes any tasks that are pending, for example, loading and deleting blocks or calling other cycle OBs.
6. Finally, the CPU returns to the start of the cycle and restarts the scan cycle monitoring time.

See also: AUTOHOTSPOT

Options for interrupting

Cyclic program execution can be interrupted by the following events:

- Interrupt
- A STOP command, triggered by
  - Operation of the programming device
  - "STP" instruction
- Supply voltage failure
- Occurrence of a device fault or program error

Start information

- None
- Optimized start information:

Name	Data type	Meaning
first_scan	BOOL	= TRUE in the first call of this OB: • Transition from STOP or HOLD to RUN • After reloading
retentivity	BOOL	= TRUE, if retentive data are available

See also

Events and OBs (Page 973)

Organization blocks for interrupt-driven program execution (S7-1200)

Time-of-day interrupt OBs (S7-1200)

Function

Organization blocks for time-of-day interrupt (OB number >=123) can be processed as follows:

- One time to a preset time (date with time of day)
- Periodically with preset start time and the following intervals:
  - Every minute
  - Hourly
  - Daily
  - Weekly
  - Monthly
  - Yearly
  - End of month

Time-of-day interrupt-OBs are therefore used to run parts of the user program on a time-controlled basis.

Status for time-of-day interrupt

The following table contains the possible status of a time-of-day interrupt as well as it's meaning.

Status	Meaning
Cancelled	The one-time processing has already taken place, or the start event of an not yet processed time-of-day interrupt has been deleted with the extended instruction CAN_TINT.
Set	You have scheduled the time or start time for processing.
Activated	You have scheduled whether processing takes place one time or periodically, and in the case of periodic processing, you have scheduled the interval.

Rules for time-of-day interrupts

The following rules apply to the use of time-of-day interrupts:

- A time-of-day interrupt can only be processed if it is set and activated, and a corresponding organization block exists in the user program.
- The start times of periodic time-of-day interrupts must correspond to a real date. For example, it is not possible to repeat an organization block monthly which first occurs on January 31st. In this case, on OB will only be started in the months that have 31 days.
- A time-of-day interrupt activated during startup by extended instruction call ACT_TINTwill not be executed until the startup is complete.
- After each CPU startup, you must reactivate previously set time-of-day interrupts

Setting and activating a time-of-day interrupt OB

Before a time-of-day interrupt can be deleted and the corresponding time-of-day interrupt OB can be processed from the operating system, you must set and activate the interrupt. You have the following options:

Set time-of-day interrupt	Activate time-of-day interrupt
Means of configuring	Means of configuring
Means of configuring	By extended instruction call ACT_TINT
By extended instruction call SET_TINTL	By extended instruction call ACT_TINT

Note

If you configure a time-of-day interrupt in such a way that the corresponding OB is to be processed once, the start time must not be in the past (relative to the real-time clock of the CPU).

If you configure a time-of-day interrupt in such a way that the corresponding OB is to be processed periodically, but the start time is in the past, then the time-of-day interrupt will be processed the next time it is due.

Time-of-day interrupt status query

In order to query the status of the time-of-day interrupt, call the extended instruction QRY_TINT.

Cancelling a time-of-day interrupt

You can cancel a time-of-day interrupt which has not yet been processed with the extended instruction CAN_TINT.

You can reset cancelled time-of-day interrupts with the extended instruction SET_TINTL and activate them with the extended instruction ACT_TINT.

Conditions that effect the time-of-day interrupt OBs

Since a time-of-day interrupt occurs only at specific intervals, certain conditions can affect the function of the associated OBs during execution of your program. The following table shows some of these conditions and describes how these affect the processing of a time-of-day interrupt OB.

Condition	Result
The extended instruction CAN_TINT is called in the user program.	The operating system deletes the start event (date and time) of the time-of-day interrupt. You must reset and reactivate the time-of-day interrupt if you are going to call the corresponding time-of-day interrupt OB again.
By synchronizing or correcting the CPU system clock, the time of day will be set ahead. With this, the start time for a time-of-day interrupt OB will be bypassed.	The operating system calls the time error interrupt OB (OB 80) and records the start event, the number, and the priority of the first bypassed time-of-day interrupt OB in its start information. After completion of the OB 80, the operating system processes the bypassed time-of-day interrupt OB only once.
By synchronizing or correcting the CPU system clock, the time of day will be set back. The corrected clock time is before the start time of an already processed time-of-day interrupt OB.	The time-of-day interrupt OB is repeated.
A time-of-day interrupt OB is still being processed when the start event for its next execution occurs.	The operating system then calls time error interrupt OB 80. The requested OB is processed only after the processing and further execution of the current time-of-day interrupt OB has completed.

Start information

A time-of-day interrupt OB has the following start information:

Tag	Data type	Description
CaughtUp	BOOL	=1, if the OB call is executed because clock time is turned ahead.
SecondTime	BOOL	=1 if the OB is called a second time because clock time is turned back (more exactly, if the planned time of the current OB processing is earlier than or the same time as the planned time for the previous OB processing). Note: SecondTime is only set once.

Status interrupt OB (S7-1200)

Description

When it receives a status interrupt, the operating system of the S7-1200 CPU calls the status interrupt OB from a DP master or IO controller. This may be the case if a module of a slave changes its operating mode, for example, from "RUN" to "STOP". For more detailed information on events that trigger a status interrupt, refer to the documentation of the slave or device manufacturer.

Structure of the start information

The status-interrupt OB has the following start information:

Name	Data type	Meaning
LADDR	HW_IO	Hardware address of the component triggering the interrupt
Slot	UINT	Slot number of the component triggering the interrupt
Specifier	WORD	Interrupt specifier from the interrupt frame

See also

Events and OBs (Page 973)

Update interrupt OB (S7-1200)

Description

When it receives a status interrupt, the operating system of the S7-1200 CPU calls the update interrupt OB from a DP master or IO controller. This may be the case if you changed a parameter on a slot of a slave or device. For more detailed information on events that trigger an update interrupt, refer to the documentation of the slave or device manufacturer.

Structure of the start information

The update interrupt OB has the following start information:

Name	Data type	Meaning
LADDR	HW_IO	Hardware address of the component triggering the interrupt
Slot	UINT	Slot number of the component triggering the interrupt
Specifier	WORD	Interrupt specifier from the interrupt frame

See also

Events and OBs (Page 973)

OB for vendor- or profile-specific interrupts (S7-1200)

Description

When it receives a manufacturer-specific or profile-specific interrupt from a DP master or IO controller, the operating system of the S7-1200 CPU calls OB 57. For more detailed information on events that trigger this type of interrupt, refer to the documentation of the slave or device manufacturer.

Structure of the start information

The OB for manufacturer-specific and profile-specific interrupts has the following start information:

Name	Data type	Meaning
LADDR	HW_IO	Hardware address of the component triggering the interrupt
Slot	UINT	Slot number of the component triggering the interrupt
Specifier	WORD	Interrupt specifier from the interrupt frame

See also

Events and OBs (Page 973)

Time-delay interrupt OBs (S7-1200)

Description

A time-delay interrupt OB is started after a configurable time delay of the operating system. The delay time starts after the SRT_DINT instruction is called.

You can use up to four time-delay interrupt OBs or cyclic OBs (OB numbers >= 123) in your program. If, for example, you are already using two cyclic interrupt OBs, you can insert a maximum of two further time-delay interrupt OBs in your program.

You can use the CAN_DINT instruction to prevent the execution of a time-delay interrupt that has not yet started.

Function of time-delay interrupt OBs

The operating system starts the corresponding OB after the delay time, which you have transferred with an OB number and an identifier to the SRT_DINT instruction.

To use a time-delay interrupt OB, you must execute the following tasks:

- You must call the instruction SRT_DINT.
- You must download the time-delay interrupt OB to the CPU as part of your program.

The delay time is measured with a precision of 1 ms. A delay time can immediately start again after it expires.

Time delay interrupt OBs are executed only when the CPU is in the "RUN" mode. A warm restart clears all start events of time-delay interrupt OBs.

The operating system calls the time-delay interrupt OB if one of the following events occurs:

- If the operating system attempts to start an OB that is not loaded and you specified its number when calling the SRT_DINT instruction.
- If the next start event for a time-delay interrupt occurs before the time delay OB has completely executed.

You can disable and re-enable time-delay interrupts using the DIS_AIRT and EN_AIRT instructions.

Note

If you disable an interrupt with DIS_AIRT after executing SRT_DINT, this interrupt executes only after it has been enabled with EN_AIRT. The delay time is extended accordingly.

Start information

- None
- Optimized start information:

Name	Data type	Meaning
sign	WORD	User ID: Input parameter SIGN from the call of the "SRT_DINT" instruction

Note

"sign" tag

The "sign" tag exists only since firmware version V4.1 of the S7-1200 CPUs. See Useful information on CPU firmware versions and STEP 7 versions (Page 1043)

See also

SRT_DINT: Start time-delay interrupt (Page 3830)

CAN_DINT: Cancel time-delay interrupt (Page 3832)

Events and OBs (Page 973)

Cyclic interrupt OBs (S7-1200)

Description

Cyclic interrupt OBs serve to start program in periodic time intervals independently of the cyclic program execution. The start times of a cyclic interrupt OB are specified using the time base and the phase offset.

The time base defines the intervals at which the cyclic interrupt OB is started and is an integer multiple of the basic clock cycle of 1 ms. The phase offset is the time by which the start time is offset compared with the basic clock cycle. If several cyclic interrupt OBs are being used, you can use this offset to prevent a simultaneous start time if the time bases of the cyclic interrupt OBs have common multiples.

You can specify a time period between 1 ms and 60000 ms as the time base.

You can use up to four cyclic interrupt OBs or time-delay OBs (OB numbers >= 123) in your program. If, for example, you are already using two time-delay interrupt OBs, you can insert a maximum of two further cyclic interrupt OBs in your program.

Note

The execution time of each cyclic interrupt OB must be noticeably smaller than its time base. If a cyclic interrupt OB has not been completely executed but execution is again pending because the cycle clock has expired, the time error OB is started. The cyclic interrupt that caused the error is executed later or discarded.

Example of the use of phase offset

You have inserted two cyclic interrupt OBs in your program:

- Cyclic interrupt OB1
- Cyclic interrupt OB2

For cyclic interrupt OB1, you have set a time base of 20 ms and for cyclic interrupt OB2 a time base of 100 ms. After expiration of the time base of 100 ms, the cyclic interrupt OB1 reaches the start time for the fifth time, cyclic interrupt OB2 for the first time. To nevertheless execute the cyclic interrupt OBs offset, enter a phase offset for one of the two cyclic interrupt OBs.

Start information

- None
- Optimized start information:

Name	Data type	Meaning
first_scan	BOOL	= TRUE in the first call of this OB • At the transition from STOP or HOLD to RUN • After reloading
event_count	INT	Number of lost start events since the last start of this OB

See also

Assigning parameters to cyclic interrupt OBs (Page 1040)

Events and OBs (Page 973)

Hardware interrupt OBs (S7-1200)

Description

You can use hardware interrupt OBs to react to specific events. You can assign an event that triggers an alarm to precisely one hardware interrupt OB. A hardware interrupt OB in contrast can be assigned to several events.

Hardware interrupts can be triggered by high-speed counters and input channels. For each high-speed counter and input channel that should trigger a hardware interrupt, the following properties need to be configured:

- The process event that should trigger the hardware interrupt (for example, the change of a count direction of a high-speed counter)
- The number of the hardware interrupt OB which is assigned to this process event

You can use up to 50 hardware interrupt OBs (OB numbers >= 123) that are independent of each other in your program.

Functionality of a hardware interrupt OB

After triggering a hardware interrupt, the operating system identifies the channel of the input or the high-speed counter and determines the assigned hardware interrupt OB.

If no other interrupt OB is active, the determined hardware interrupt OB is called. If a different interrupt OB is already being executed, the hardware interrupt is placed in the queue of its priority class. The hardware interrupt is acknowledged after the completion of the assigned hardware interrupt OB.

If another event that triggers a hardware interrupt occurs on the same module during the time between identification and acknowledgement of a hardware interrupt, the following applies:

- If the event occurs on the channel that previously triggered the hardware interrupt, then no additional hardware interrupt is triggered. Another hardware interrupt can only be triggered if the current hardware interrupt is acknowledged.
- If the event occurs on a different channel, a hardware interrupt is triggered.

Hardware interrupt OBs are called only in the CPU's "RUN" mode.

Start information

- None
- Optimized start information:

Name	Data type	Meaning
Laddr	HW_IO	Hardware identifier of the module that triggers the hardware interrupt
USI	WORD	Identifier for future extensions (not user-relevant)
IChannel	USINT	Number of the channel that triggered the hardware interrupt
EventType	BYTE	Identifier for the event type associated with the event triggering the interrupt (e.g., positive edge) This identifier can be found in the description of the respective module.

See also

Assigning parameters to hardware interrupt OBs (Page 1041)

Events and OBs (Page 973)

Time error OB (S7-1200)

Description

The operating system calls the time error OB (OB 80) if one of the following events occurs:

- The cyclic program exceeds the maximum cycle time.
- The OB called is currently still being executed (possible for time-delay interrupt OBs and cyclic interrupt OBs).
- A time-of-day interrupt was missed because the clock time was set forward by more than 20 seconds.
- A time-of-day interrupt was missed during a STOP.
- An overflow has occurred in an interrupt OB queue.
- An interrupt was lost due to the excessive interrupt load.

If you have programmed no time error OB, the S7-1200 CPU reacts as follows:

- CPUs with firmware version V1.x to V3.x: The CPU remains in RUN mode.
- CPUs with firmware version as of V4.x:
  - The CPUs goes to STOP mode when the maximum cycle time is exceeded.
  - With all other start events of the time error OB, the CPU remains in RUN mode.

Violation of the maximum cycle time twice does not lead to the calling of an OB, but to the STOP of the CPU. You can avoid the second violation by restarting the cycle monitoring of the CPU with the RE_TRIGR instruction.

You can use only one time error OB in your program.

Start information

The time error OB has the following start information:

Tag	Data type	Description
fault_id	BYTE	• 0x01: Maximum cycle time exceeded • 0x02: Called OB is still being executed • 0x05: Expired time-of-day interrupt due to time jump • 0x06: Expired time-of-day interrupt on return to RUN mode • 0x07: Queue overflow • 0x09: Interrupt loss due to high interrupt load
csg_OBnr	OB_ANY	Number of the OB being executed at the time of the error
csg_prio	UINT	Priority of the OB being executed at the time of the error

See also

Events and OBs (Page 973)

Cycle time and maximum cycle time (Page 988)

Diagnostic interrupt OB (S7-1200)

Description

You can enable the diagnostic error interrupt for diagnostics-capable modules so that the module detects changes in the I/O status. As a result, the module triggers a diagnostic error interrupt in the following cases:

- A fault is present (incoming event)
- A fault is no longer present (outgoing event)

If no other interrupt OB is active, then the diagnostic interrupt OB (OB 82) is called. If another interrupt OB is already being executed, the diagnostic error interrupt is placed in the queue of its priority group.

You can use only one diagnostic interrupt OB in your program.

Start information

The diagnostic interrupt OB has the following start information:

Tag	Data type	Description
IO_state	WORD	Contains the I/O status of the diagnostics-capable module.
laddr	HW_ANY	HW-ID
Channel	UINT	Channel number
multi_error	BOOL	= 1, if there is more than one error

IO_state tag

The following table shows the possible I/O states that the IO_state tag can contain:

IO_state	Description
Bit 0	Configuration correct: = 1, if the configuration is correct = 0, if the configuration is no longer correct
Bit 4	Error: = 1, if an error is present, e.g., a wire break = 0, if the error is no longer present
Bit 5	Configuration not correct: = 1, if the configuration is not correct = 0, if the configuration is correct again
Bit 7	I/O cannot be accessed: = 1, if an I/O access error has occurred In this case, laddr contains the hardware identifier of the I/O with the access error. = 0, if the I/O can be accessed again
Bits 8 to 15	Reserved (always 0)

See also

Events and OBs (Page 973)

Pull/plug interrupt OB (S7-1200)

Description

The S7-1200 CPU operating system calls the pull/plug interrupt OB (OB 83) if a configured and non-disabled distributed I/O module or submodule (PROFIBUS, PROFINET, AS-i) is removed or inserted.

Note

The removal or insertion of a central module leads to the STOP of the CPU.

Start information

The pull/plug interrupt OB has the following start information:

Name	Data type	Meaning
LADDR	HW_IO	Hardware identifier of the affected module or submodule
Event_Class	BYTE	• B#16#38: (Sub)module plugged • B#16#39: (Sub)module pulled or not responding
Fault_ID	BYTE	Error code (possible values: B#16#51, B#16#54, B#16#55, B#16#56, B#16#57, B#16#58)

The following table shows which event caused the start of the pull/plug interrupt OB.

ev_class (B#16# ...)	fault_id (B#16# ...)	Meaning
39	51	Module pulled
39	54	Submodule pulled
38	54	Submodule plugged, which conforms to the parameterized submodule
38	55	Submodule inserted, which does not conform to the submodule parameter assignment
38	56	Submodule inserted, but error in module parameter assignment
38	57	Submodule or module inserted, but with a fault or maintenance
38	58	Submodule access error remedied

See also

Events and OBs (Page 973)

Rack error OB (S7-1200)

Description

The S7-1200 CPU operating system calls the OB 86 in the following cases:

- The failure of a DP master system or of a PROFINET IO system is detected (in the case of either an incoming or an outgoing event).
- The failure of a DP slave or of an IO device is detected (in the case of either an incoming or an outgoing event).
- Failure of some of the submodules of a PROFINET I-device is detected.

Structure of the start information

The rack fault OB has the following start information:

Name	Data type	Meaning
LADDR	HW_IO	Hardware identifier of the defective hardware object
Event_Class	BYTE	• B#16#32: Activation of a DP slaves or an IO device • B#16#33: Deactivation of a DP slaves or an IO device • B#16#38: outgoing event • B#16#39: incoming event
Fault_ID	BYTE	Error code (possible values: B#16#C3, B#16#C4, B#16#C5, B#16#C6, B#16#C7, B#16#C8, B#16#C9, B#16#CA, B#16#CB, B#16#CC, B#16#CD, B#16#CE, B#16#CF, B#16#F8, B#16#F9)

The following table shows which event caused the start of OB 86.

Ev_class B#16# ...	Fault_id B#16# ...	Meaning
39	C3	Failure of a DP master system
39/38	C4	Failure/return of a DP slave
38	C5	Return of a DP slave; however, slave is faulty
38	C6	Expansion unit return, but error in module parameter assignment
38	C7	Return of a DP device, but there is error in module configuration
38	C8	Return of a DP device, but discrepancy between preset/actual configuration
32/33	C9	Activation/deactivation of a DP slave with the "D_ACT_DP" instruction
39	CA	Failure of a PROFINET IO system
39/38	CB	Failure/return of a PROFINET IO device
38	CC	Return of a PROFINET IO device with fault or maintenance
38	CD	Return of a PROFINET IO device, deviation between preset/actual configuration
38	CE	Return of a PROFINET IO device, error in module configuration
32/33	CF	Activation/deactivation of an IO device with the "D_ACT_DP" instruction
39/38	F8	Failure/return of some of the submodules of a PROFINET I-device
38	F9	Return of some of the submodules of a PROFINET I-device with a device configuration difference

See also

Events and OBs (Page 973)

MC-servo OB (S7-1200)

Description

When you create a technology object with drive connection using MC-Servo [OB91]/create an analog output, the organization block PROFIdrive is created automatically. The Motion Control functionality of the technology objects creates its own execution level, and is called according to the Motion Control application cycle.

The MC-Servo OB is write-protected. The content cannot be changed.

The position control algorithms of all technology objects configured for motion control on the CPU are calculated within the MC-Servo OB.

You can set the application cycle and the priority of the organization block in accordance with your requirements for control quality and system load.

Application cycle

You can set the application cycle in which the MC-Servo OB is called in the properties of the organization block.

The MC-Servo OB is called cyclically with the specified application cycle.

The selected application cycle must be long enough to allow all the technology objects to be processed in one cycle. If the processing time of the technology objects is longer than the application cycle, overflows will occur.

To avoid disruptions in the program execution on the CPU, set the application cycle depending on the number of axes used as follows:

Application cycle = number of axes x 2 ms

The following table shows the resulting application cycle as an example according to the number of axes:

Number of axes	Application cycle
1	2 ms
2	4 ms
4	8 ms
8	16 ms

The SINAMICS G120 drive updates the drive process image every 4 ms. To improve control, set the application cycle of the MC-Servo OB to 4 ms or to a multiple of 4 ms.

See also

MC-PreServo OB (Page 1037)

MC-PostServo OB (Page 1037)

MC-interpolator OB (Page 1038)

Assigning the MC-Servo OB parameters (Page 1043)

MC-PreServo OB (S7-1200)

Description

The organization block MC-PreServo [OB67] can be programmed and is called in the application cycle configured at the MC-Servo OB (Page 1036). MC-PreServo [OB67] is called directly before MC-Servo [OB91].

Via the organization block, you can read out the configured application cycle (information in µs).

Structure of the start information

Optimized start information:

Name	Data type	Meaning
Initial_Call	BOOL	=TRUE in the first call of this OB during transition from STOP to RUN
PIP_Input	BOOL	TRUE: The associated process image input is up-to-date
PIP_Output	BOOL	TRUE: The associated process image output was transferred to the outputs in good time after the last cycle
IO_System	USINT	Number of the distributed I/O system triggering the interrupt
Event_Count	INT	• = n: Number of lost cycles • = -1: An unknown number of cycles has been lost (e.g., because cycle has changed).
Synchronous	BOOL	Reserved
CycleTime	UDINT	Display of the application cycle configured for the MC-Servo OB in µs

MC-PostServo OB (S7-1200)

Description

The organization block MC-PostServo [OB95] can be programmed and is called in the application cycle configured for the MC-Servo OB (Page 1036). MC-PostServo [OB95] is called directly after MC-Servo [OB91].

Via the organization block, you can read out the configured application cycle (information in µs).

Structure of the start information

Optimized start information:

Name	Data type	Meaning
Initial_Call	BOOL	=TRUE in the first call of this OB during transition from STOP to RUN
PIP_Input	BOOL	TRUE: The associated process image input is up-to-date
PIP_Output	BOOL	TRUE: The associated process image output was transferred to the outputs in good time after the last cycle
IO_System	USINT	Number of the distributed I/O system triggering the interrupt
Event_Count	INT	• = n: Number of lost cycles • = -1: An unknown number of cycles has been lost (e.g., because cycle has changed).
Synchronous	BOOL	Reserved
CycleTime	UDINT	Display of the application cycle configured for the MC-Servo OB in µs

MC-interpolator OB (S7-1200)

Description

When you create a technology object with drive connection using MC-Interpolator [OB91]/ create an analog output, the organization block PROFIdrive is created automatically. The Motion Control functionality of the technology objects creates its own execution level, and is called according to the Motion Control application cycle of the MC-Servo OB (Page 1036).

The MC-Interpolator OB is write-protected. The content cannot be changed.

The evaluation of the motion control instructions as well as the monitoring and the setpoint generation for all technology objects configured on the CPU (motion control) is performed within the MC-Interpolator OB.

The MC-Interpolator OB is called at the end of processing of the MC-Servo OB. The clock ratio between MC-Interpolator OB and MC-Servo OB is always 1:1.

Process response

You must select the application cycle of the MC-Servo OB long enough to allow all the technology objects for motion control to be processed in one application cycle. If the application cycle is not observed, overflows occur. Note the following response to overflows:

- The CPU tolerates a maximum of three consecutive overflows of the MC-Interpolator OB.
- The processing of an MC-Interpolator OB can be interrupted by the calling of the MC Servo OB, at most.

Block parameters of organization blocks (S7-1200)

Basics of block parameters (S7-1200)

Introduction

Several organization blocks (OBs) have properties with which you can control their behavior or their assignment to specific events. You can influence these properties by assigning parameters.

Overview

You can assign parameters to the properties for the following organization blocks:

- Time-of-day interrupt OBs
- Cyclic interrupt OBs
- Hardware interrupt OBs
- MC-Servo OB

See also

Assigning parameters to hardware interrupt OBs (Page 1041)

Assigning parameters to cyclic interrupt OBs (Page 1040)

Parameter assignment for time-of-day interrupt OBs (S7-1200)

Procedure for setting the parameters

To set the parameters of a time-of-day interrupt OB, proceed as follows:

1. Open the "Properties" dialog belonging to the time-of-day interrupt OB in question.
2. Click the "Time-of-day interrupt" group in the area navigation.

Overview of the parameters that can be set

You can set the following parameters:

- Execution
- Start date and time-of-day
- Option buttons "Local time" and "System time"

"Execution" parameter

Use the drop-down list "Execution" to define the periods at which the time-of-day interrupt OB is to be executed. The time intervals relate to the settings for "Start date" and "Time-of-day".

The following values are possible for "Execution":

- Never
- Once
- Every minute
- Hourly
- Daily
- Weekly
- Monthly
- Yearly
- End of month

Note

With the value "End of month", the value specified under "Start date" is irrelevant.

"Start date" and "Time-of-day" parameter

Here, you specify the time at which the time-of-day interrupt is to be executed for the first time.

Example: Start date = 07/05/2013, time =11:16

Depending on the value of the "Execution" parameter, the CPU generates additional time-of-day interrupts periodically. Depending on the setting, the start time relates either to the local time or to the coordinated universal timeUTC.

Note

If you set the "Execution" parameter to "Monthly", the start date cannot be set to the 29th, the 30th, or the 31st. If you want the time-of-day interrupt OB to start at the end of the month, you should set the parameter "Execution" to "End of month".

"Local time" or "System time"

Here, you decide which time the start time of the time-of-day interrupt OB relates to:

- "Local time": The start time relates to the time zone set for the CPU.
- "System time": The start time relates to the coordinated universal time UTC (Universal Time Coordinated).

Assigning parameters to cyclic interrupt OBs (S7-1200)

Introduction

You can use cyclic interrupt OBs to start programs at regular time intervals. To do so you must enter a scan time and a phase shift for each cyclic interrupt OB used.

You can use up to four cyclic interrupt OBs or time-delay OBs (OB numbers >= 200) in your program. If, for example, you are already using two time-delay interrupt OBs, you can insert a maximum of two further cyclic interrupt OBs in your program.

Note

If you assign multiple cyclic OBs, make sure that you assign a different cycle time or phase offset to each cyclic interrupt OB to avoid them executing at the same time or having to queue. When you create a cyclic interrupt OB, the cycle time 100 and the phase offset 0 are entered as start value.

Procedure

To enter a scan time and a phase shift for a cyclic interrupt OB, proceed as follows:

1. Open the "Program blocks" folder in the project tree.
2. Right-click on an existing cyclic interrupt OB.
3. Select the "Properties" command in the shortcut menu.
   The "<Name of the cyclic interrupt OB>" dialog box opens.
4. Click the "Cyclic interrupt" group in the area navigation.
   The text boxes for the scan time and the phase shift are displayed.
5. Enter the scan time and the phase shift.
6. Confirm your entries with "OK".

See also

Creating organization blocks (Page 4715)

Basics of block parameters (Page 1039)

Cyclic interrupt OBs (Page 1029)

Assigning parameters to hardware interrupt OBs (S7-1200)

Introduction

You must select the corresponding event and assign the following parameters for every input channel and high-speed counter that should trigger a hardware interrupt:

- Event name
- Number of the hardware interrupt OB that is assigned to this process event

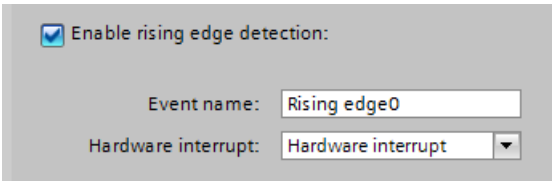
The parameters of the hardware interrupt are assigned in the properties of the corresponding device. You can assign parameters for up to 50 hardware interrupt OBs.

You can create the hardware interrupt OB to be assigned parameters either before or during activation of an event.

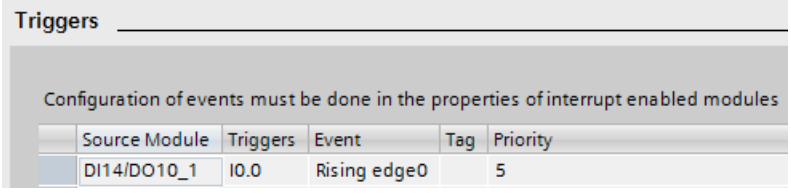
Procedure

To configure a hardware interrupt event, follow these steps:

1. Double-click the "Devices & Networks" command in the project tree.
   The hardware and network editor opens in the network view.
2. Change to the device view.
3. If the Inspector window closed in the device view, select the "Inspector window" check box in the "View" menu.
   The Inspector window opens.
4. Click the "Properties" tab.
5. In the device view, select the module for which you want to a assign a hardware interrupt.
6. Select the corresponding event that will trigger a hardware interrupt, e.g., a positive edge.



7. Enter an event name.
8. Select an existing hardware interrupt OB from the "Hardware interrupt" drop-down list or create a new hardware interrupt OB. If you have not previously created any hardware interrupt OBs, you can click the "Add new block" button in the drop-down list.
   The start information of the corresponding hardware interrupt OB, including all specifications for the interrupt-triggering event, is updated.



9. If you want to assign further hardware interrupts, repeat steps 5 to 8.

A system constant of data type Event_HwInt is created automatically for the event identified by the explicit event name. The system constants are displayed in the standard tag table.

See also

Creating organization blocks (Page 4715)

Basics of block parameters (Page 1039)

Hardware interrupt OBs (Page 1030)

Events and OBs (Page 973)

Assigning the MC-Servo OB parameters (S7-1200)

Procedure for setting the parameters

To assign the parameters of an MC-Servo OB, follow these steps:

1. Open the "Properties" dialog associated with the MC-Servo OB in question.
2. Click on the "Cycle time " group in the area navigation.

Overview of the parameters that can be set

In the "Cycle time" group, select the application cycle with which the MC-Servo OB will be called cyclically in the execution system of the SIMATIC S7-1200

You set the cycle in which the MC-Servo OB is called in the "Application cycle" field.

See also

Basics of block parameters (Page 1039)

MC-servo OB (Page 1036)

Useful information on CPU firmware versions and STEP 7 versions (S7-1200)

CPUs and engineering software for configuring the CPUs is constantly developed further to improve performance and security. New versions that have some special features with respect to the interaction of the components are created in this way. The sections below describe the special features of the S7-1200 CPUs with firmware version V4 as compared to firmware versions V1 to V3.

For a detailed comparison of the range of functions, read the S7-1200 System Manual (description of new instructions, new organization blocks and advanced configuration options).

Required engineering software

S7-1200 CPUs V4 can be configured with STEP 7 as of V13.

Compatibility between memory card content and firmware version of the CPU

Memory cards (transfer cards or program cards) with configuration and program for an S7-1200 CPU V1, V2 or V3 do not work with an S7-1200 CPU V4.
Memory cards with configuration and program for an S7-1200 CPU V4 do not work with an S7-1200 CPU V1, V2 or V3.

You must change an S7-1200 CPU V1 to V3 configuration to an S7-1200 CPU V4 configuration (device replacement) and then download it to the CPU. A gradual device replacement is required for an S7-1200 CPU V1-V2 (see below).

When you insert the memory card into a CPU with an incompatible firmware version, the CPU does not start up. When you insert a memory card for a CPU V1, V2 or V3 into an S7-1200 CPU V4, this CPU outputs a version error.

Going online and loading

When you have configured an S7-1200 CPU with firmware version V1, V2 or V3 with STEP 7, the CPU to which you want to go online or which you want to download must have one of these firmware versions. You cannot go online to an S7-1200 CPU V4 with a configured S7-1200 CPU V1, V2 or V3.

On the other hand, you cannot go online to an S7-1200 CPU V1, V2 or V3 with a configured S7-1200 CPU V4 or download this CPU.

Replacing an existing CPU

You can replace a configured S7-1200-CPU V1, V2 or V3 with a new CPU with a firmware version greater than or equal to 4. In case of an existing S7-1200 CPU V1 or V2, you must start by replacing the device with an S7-1200 CPU V3: It is not possible to directly replace it with an S7-1200 CPU V4.

1. S7-1200 CPU V1 (V2) > S7-1200 CPU V3
2. S7-1200 CPU V3 > S7-1200 CPU V4

As long as you do not download the configuration, you can undo the device replacement ("Undo" command in the "Edit" menu). When you download the configuration for the new CPU firmware version ("V4 configuration") to the CPU, you can no longer return to version V3.

This means you should save the existing project, for example, with a V3 configuration as project archive so that you can access it later.

Special notes for device replacement (V3 > V4):

- The interrupt behavior of interrupt OBs remains the same; they are not configured as interruptible. This is also the default behavior of S7-1200 CPUs V1-V3. The interrupt behavior of the interrupt OB can be configured for S7-1200 V4 CPUs. If you drag an S7-1200 CPU V4 directly from the hardware catalog into the network view, this option is activated (interrupt OBs are interruptible).
- The behavior for PUT/GET access of remote partners remains the same; access is permitted. This is also the default behavior of S7-1200 CPUs V1-V3. The access via PUT/GET communication by means of remote partners can be configured for S7-1200 V4 CPUs ("Protection" area of CPU parameters). If you drag an S7-1200 CPU V4 directly from the hardware catalog into the network view, access is not permitted and must be explicitly enabled.
- The wording for the protection levels changes but the effect of the settings remains the same. You can also select the access level "No access (complete protection)".
- The web server settings for activation of the web server and HTTP/HTTPS settings are applied. You also have the option to create users and assign them specific rights (Web server area > User management of CPU parameters). Web server users only have access to standard websites when you do not configure users. An S7-1200 CPU V4 no longer supports the user "admin" and his/her password.
- For reasons of compatibility, the interfaces of used OBs are not changed by the device replacement. You can therefore continue to use used OBs. If you want to use OB properties that have been added or changed in the new firmware version of the S7-1200 CPU, you will need to recreate the corresponding OB.

Communication with HMI devices

When you connect an HMI device to an S7-1200 CPU V4, make sure that you use the appropriate Runtime software version of the HMI device.

You may have to transfer the latest HMI Runtime version by means of the WinCC engineering software.

You must compile the HMI configuration once again and download it to the HMI device for the CPU-HMI communication to work.

See also

What you need to know about SIMATIC memory cards (Page 964)

Addressing modules (S7-1200)

Addressing modules (S7-1200)

Introduction

In the device overview, you see the addresses or address ranges of the modules in the I address and Q address columns. There are other addresses as well, which are explained below.

I/O address

I/O addresses (input/output addresses) are required to read inputs and set outputs in the user program.

Input and output addresses are assigned automatically when modules are inserted in the rack. The address of the first channel is the start address of a module. The addresses of the other channels are derived from this start address. The address end is obtained from the module-specific address length.

Device address (e.g., Ethernet address)

Device addresses are addresses of programmable modules (Industrial Ethernet addresses). They are required to address the different stations of a subnet, for example, to download a user program to a CPU.

Hardware identifier for identifying modules and submodules

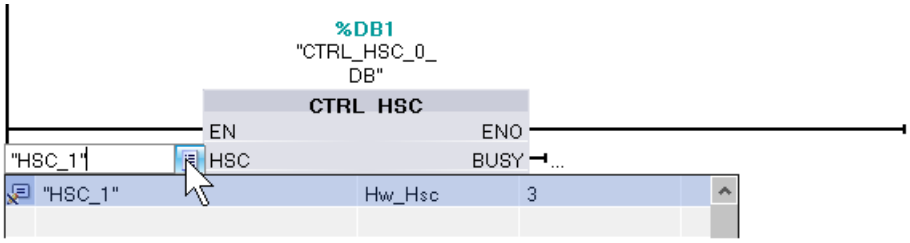
In addition to the I addresses and Q addresses, a hardware identifier (HW ID) is assigned automatically and is used to address and identify the module. Submodules (units of a module), such as an integrated counter, also receive such a hardware identifier.

The hardware identifier consists of an integer and is output by the system with diagnostics alarms to allow the faulty module or the faulty submodule to be localized.

In addition, the hardware identifier is used for a number of instructions to address the corresponding module.

The hardware identifier cannot be changed.

Example: Identifying high-speed counters of the S7-1200 CPU



The hardware identifier is assigned automatically when components are inserted in the device or network view and in the PLC tags. A name is also assigned automatically for the hardware identifier. The system constants of the PLC tags cannot be changed either.

See also

Specifying input and output addresses (Page 1046)

Assigning addresses to a location in the program (Page 1047)

Introduction to loading a configuration (Page 1815)

Inspector window (Page 539)

Specifying input and output addresses (S7-1200)

Default input and output addresses are set automatically. You can, however, change the address assignment later.

All addresses of modules are located in the process image area. The process image is automatically updated cyclically.

Requirement

You are in the device view.

Procedure

To change the preset address range proceed as follows:

1. In the device view, click on the module for which you want to set the start address.
2. Go to "I/O addresses" in "Properties" in the inspector window.
3. Under "Start address" enter the required start address.
4. Press <Return> or click on any object to accept a modified value.

If you have entered an invalid address, a message indicating the next available address is displayed.

Note

You can also change the addresses directly in the device overview.

See also

Editing properties and parameters (Page 575)

Input and output addresses in the address overview (Page 577)

Assigning addresses to a location in the program (S7-1200)

You can assign addresses of the I/O channels of modules directly to the points of use in the program or to a tag table.

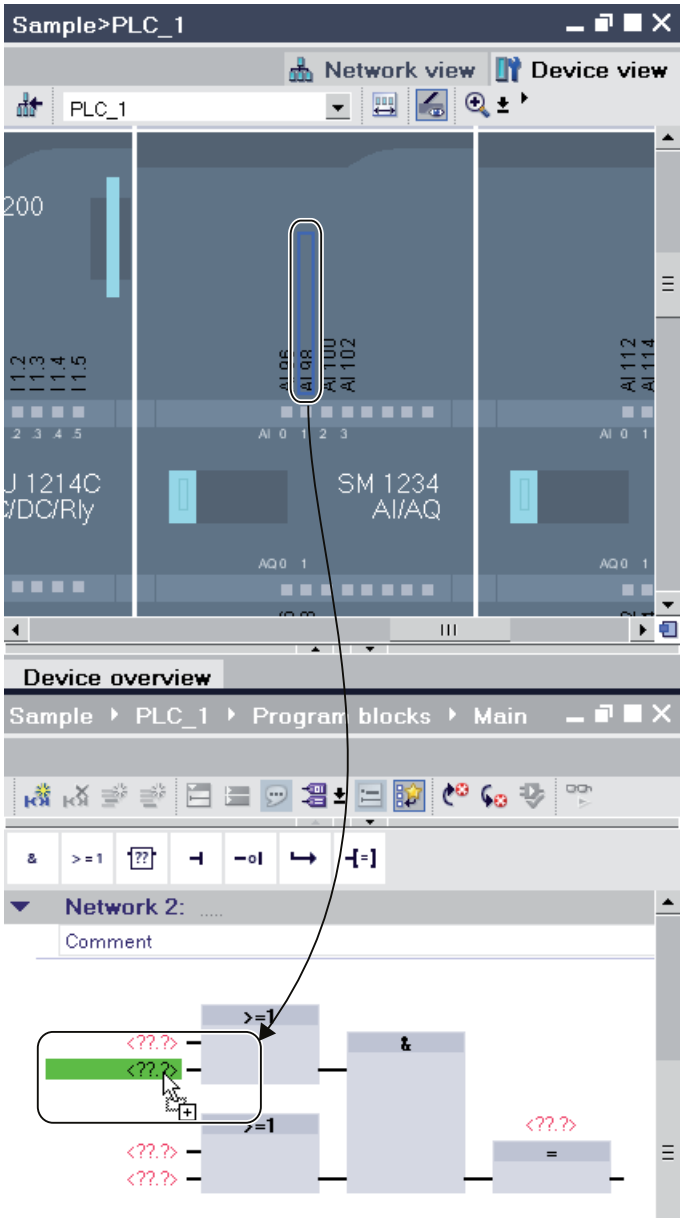
Requirement

- The device view of the hardware and network editor is open.
- The zoom level in the device view must be set to at least 200% to allow you to see the individual I/O channels.
- The instruction window of the programming editor or a tag table is open.

Procedure

To assign I/O channels of modules to the points of use in the program or to a tag table, follow these steps:

1. In the device view, navigate to the module with the desired I/O channel.
2. Click and hold down the mouse button to drag the desired I/O address to the corresponding point of use of the block or to the tag table.



The address of the module is assigned to the point of use in the program or entered as a tag in the tag table.

Note

The tag for an input or output of a block can also be dragged to the input or output of a module in order to link the tag to the I/O channel of the module.

See also

Overview of PLC tag tables (Page 4837)

Signal board (S7-1200)

Inserting a signal board in a CPU (S7-1200)

Introduction

Signal boards allows you to increase the number of the S7-1200 CPU's own inputs and outputs. Just like all other hardware components, you will find signal boards in the hardware catalog. However, you do not insert signal boards in the rack like other modules but directly in a slot of the CPU itself.

Note the following points when using a signal board:

- Each CPU can have only one signal board inserted in it.
- A signal board can only be inserted when the slot in the CPU is free.

There are various ways of inserting a signal board in a CPU:

- Double click on a signal board in the hardware catalog when there is a free slot in the CPU
- Drag from the hardware catalog to a free slot in the CPU
- Shortcut menu of a signal board in the hardware catalog for copying and pasting

Requirement

- The hardware catalog is open.
- The S7-1200 CPU has a free slot for the signal board.

Inserting a signal board in a CPU

To insert a signal board in a CPU, proceed as follows:

1. Go to the required signal board in the hardware catalog.
2. Select the signal board you want to use.
3. Drag the signal board to the free slot in the CPU.



You have now inserted a signal board in the slot of the CPU.

If you are in the network view, you can also drag a signal board to a device. If the CPU has a an empty slot for a signal board, the signal board is inserted automatically into this slot.

Configurations for Web server (S7-1200)

Information about the web server (S7-1200)

Introduction

The Web server allows you to monitor the CPU via the Internet or the intranet of your company. This permits evaluation and diagnostics over long distances.

Alarms and status information are visualized on HTML pages.

Web browser

You need a web browser to access the HTML pages of the CPU.

The information with which Web browsers and Web browser versions the Web server was tested is available in the corresponding manual (http://support.automation.siemens.com/WW/view/en/36932465).

Reading information via the Web server

The following information can be read from the CPU. The availability of the respective web pages depends on the CPU and its firmware version.

As of firmware version 4 the web pages are available in several languages.

Page/information	Description
Introduction	Entry page for the standard web pages
Start page with general CPU information	The start page provides an overview of general information on the CPU, the name of the CPU, the type of CPU and basic information on the current operating state.
Diagnostics	Display of the detailed information about the CPU including the serial number, article number and version numbers. Displays the content of the diagnostics buffer with the most recent entries first.
Module information	Display of the identification information whether the centrally inserted components of a station are OK, whether there are maintenance requirements or components cannot be reached, for example. As of firmware version 4 a firmware update is possible via this Web page.
Communication	Displays the communication connections during open communication (OUC); displays resources and address parameters.
Diagnostic buffer	Display of the diagnostic buffer content
Tags	Displays the status of operands of the user program to monitor and change the values.
File browser (as of firmware version 4)	Data logs in CSV format to transfer to the hard disk of the programming device. The data logs are created with data log instructions in the user program and filled with data. As of firmware version 4 you have access to files of the internal load memory and of the external load memory (Memory Card) via this web page, for example, to the content of the directories "DataLogs" and "Recipes".
Login	Login as a user or logoff.
User pages (if user-defined web pages have been configured and loaded)	The user web pages deliver a list of web pages with customer-specific web applications.

Web access to the CPU via PG/PC

Proceed as follows to access the Web server:

1. Connect the client (PG/PC) to the CPU via the PROFINET interface.
2. Open the web browser.
   Enter the IP address of the CPU in the "Address" field of the Web browser in the format http://ww.xx.yy.zz (example: http://192.168.3.141).
   The start page of the CPU is opened. From the start page, you can navigate to further information.

Standard web pages (S7-1200)

Requirements for web access (S7-1200)

The requirements for access to standard CPU web pages are explained in the following, as well as the effects of missing or existing configuration information.

Requirement

The web server must be started.

The web server only starts when it has been activated in the properties of the CPU in the "Web server" section.

Note the following:

The web pages are normally transmitted via an non-secure connection and are not secured against hacker attacks. If you want to transfer the web pages in encrypted form to the browser, use the URL https://, followed by the IP address of the CPU.

Logon

No logon is required to access the standard web pages read-only. A user must be logged on to execute certain actions like changing the operating mode of the CPU or for write access.

For S7-1200 CPUs up to FW version V3:

You must be logged on as "admin" for the actions listed above. The logon input boxes are on the top left of each standard web page.



If you log on as "admin", you must enter the user name and password there.

Name: admin.

Password: configured CPU password (for password-protected CPU).

For S7-1200 CPUs as of FW version 4:

You can freely select the names of users and the passwords (CPU parameter "Web server", area "User management").

You assign rights to the users, for example the right to query diagnostics or to update the firmware.

JavaScript and cookies

The standard web pages use JavaScript and cookies. You must enable both in your web browser.

If JavaScript is not enabled, the following limitations apply:

- Data from standard web pages is not automatically updated.
- You cannot log on as user.
- Fields cannot be sorted (module information)

If cookies are not enabled, you cannot log on.

See also

Access for HTTPS (Page 1054)

Settings for operation (S7-1200)

Settings for operation

To be able to use the web server of an S7-1200-CPU, you must select the CPU in the network view or the device view and make the following settings in the inspector window under "Properties > General > Web server":

- Enable the web server
- Restricting access to the CPU to HTTPS transmission protocol (encrypted transmission) Access via port 80 is then blocked. Communication is only possible via port 443.

- Enabling automatic update of web pages
  The update interval is set by default and cannot be changed. The CPU updates web pages with changing content ((for example, status information or diagnostics information) at regular intervals.
- Creating and managing users
  Users are exclusively entitled to options that are assigned to the access rights. Depending on the CPU and firmware used, you can assign different user rights. Rights that your CPU does not support cannot be activated. A user called "Everybody" with minimal access rights, that you can, however, extend, is set by default in the user list. A user who uses the Web server without entering a password has "Everybody" user rights.
  You have the possibility of configuring further users with different access rights. These users have to log on with the configured user name and password.

WARNING

Unauthorized access to the CPU by means of the Web server

Unauthorized access to the CPU or the changing of PLC variables to invalid values can result in interruptions of the processes controlled by the CPU and cause death, serious bodily harm or material damage!

Since the activation of the Web server allows authorized persons for example to change operating modes, to write-access CPU data or to update the firmware, we recommend that the following security measures be taken:

- If possible limit access to the HTTPS protocol.
- Create users with secure passwords. An example of a secure password is one which is only used for a single application, is more than 8 characters long, and consists of lower- and upper-case letters as well as symbols and numbers (?!+%$1234...). In addition, passwords based on common keyboard sequences or words from the dictionary should be avoided.
  Change the password at regular intervals.
- Do not extend the rights of the "Everybody" user.
- Check the PLC variables in the user program and limit the range of values to permissible ranges since users can set invalid values via the Web server.

Access for HTTPS (S7-1200)

Access via HTTPS

HTTPS is used for encrypting and authentication of communication between the browser and web server.

To transfer data between the browser and the CPU using the HTTPS protocol, enter the URL as https://ww.xx.yy.zz in the address line of your browser, whereby ww.xx.yy.zz stands for the IP address of the CPU.

You require a valid, installed certificate for error-free HTTPS access to the CPU.

If no certificate is installed a warning is displayed with a recommendation not to use this page. To view the page you must explicitly "Add exception".

You can receive a valid certificate (Certification Authority) "SIMATIC CONTROLLER" as a download from the "Intro" web page under "Download certificate". The help function for your respective web browser provides information on how to install a certificate.

Accessing data of the CPU memory (S7-1200)

You can access data in the internal or external CPU load memory by means of a standard web page.

- Use the web page "Data logs" for S7-1200 CPUs up to and including FW version 3. From this web page, you transfer the data logs from the CPU to a drive on your PC.
- Use the web page "File Browser" for S7-1200 CPUs as of FW version 4. From this web page, you transfer data from the folders "Data logs" or "Recipes", for example, to a drive on your PC.

Depending on the file type and the access permissions you have configured for the web server user, you can download, delete, rename or upload the files. The actual directories can only be created, deleted or renamed.

Example: Data logs

To open a data log, click on the link of the desired data log. You can then open the file (.csv), for example, in Microsoft Excel or in another program you choose or you can save the file.

Special note: Data logs are saved in U.S. American CSV format. You can only open the file directly using the U.S. version of Microsoft Excel. If you are using another national version of Microsoft Excel, you must import the file, selecting "comma" in the import assistant as the delimiter.

Downloading a data log

To download a data log, click on the download icon of the desired data log. You can then open the file (.csv), for example, in Microsoft Excel or in another program you choose or you can save the file.

Downloading and clearing or deleting a data log

For a CPU with FW version up to V3.0:

To download and delete the current entries of the data log, you must be logged on. To do this, click on the "Download and delete" icon of the required data log. You can then open the file (.csv), for example, in Microsoft Excel or in another program you choose or you can save the file.

For a CPU as of FW version V4.0:

To reset the data log, follow these steps:

1. Open the CSV file, for example with Excel.
2. Delete the rows between the headline and the row with the entry "//END", if this row exists.

3. Save the file to a drive on your PC.
4. On the web page "File Browser", delete the data log (which means the CSV file) and load the prepared CSV file to the CPU with the "Upload file" button of the "File Browser" web page.

Additional information is available in the S7-1500 CPU system manual.

See also

Data logging - Overview (Page 4011)

Creating and loading user pages (S7-1200)

What you need to know about user pages (S7-1200)

Concept

The concept of user pages or of user-defined web pages allows you to access freely-designed web pages of the CPU with a web browser. The Web server of the CPU provides this function.

You are not dependent on special tools for the design and functionality of the user-defined web pages. You can adapt the pages in the layout with CSS, provide dynamic content with JavaScript or use any framework to produce web pages.

The totality of files processed by the Web server is also referred to as the "web application".

Web application and user program

Using HTML code in user-defined web pages (AWP commands), you can also transmit data via a web browser to the user program of the CPU for further processing and can display data from the operand area of the CPU in the web browser.

You can use script instructions (such as Javascript) to optimize your web pages, for example to dynamically change contents or validate user entries.

To synchronize between the user program and the Web server, but also to initialize, you must call the WWW (SFC 99) instruction in the user program.

- If no interaction is required between the web application and the user program, for example, if a web page only provides static information, only initialization in the user program is required.
- If a simple data exchange is necessary between PLC tags and tags in web applications, to display the contents of PLC tags or write a value in a PLC tag for example, the syntax for reading and writing tags has to be observed. In this case only an initialization is required in the user program, for example in the startup OB.
- If a further interaction is required between the web application and the user program, you must handle status and control information from the Web Control DB in addition to the synchronization between Web server and user program. This is the case, for example, when user entries are transmitted via the web browser to the Web server for evaluation by the CPU. Unlike simple data exchange, the user program directly influences the time at which the requested web page is relayed back to the web browser. In this case, you must be acquainted with the concept of manual fragments and the structure of the Web Control DB.

Initialization

User-defined web pages are "packaged" in data blocks for processing by the CPU. You must generate appropriate data blocks from the source data (HTML files, images, JavaScript files, etc.) during configuration to be able to download the web application into the CPU. The Web Control DB takes on a special role (default: DB 333). It contains status and control information as well as links to additional data blocks with coded web pages. Data blocks that contain coded web pages are termed "Fragment DBs".

When the data block is downloaded into the CPU, the CPU does not "know" that user-defined web pages are coded inside it. The "WWW" (SFC 99) instruction, for example, in the Startup OB informs the CPU which DB is the Web Control DB. The user-defined web pages can be accessed via a web browser after this initialization.

Synchronization

If the user program is to exchange data with the user-defined web pages, the WWW (SFC 99) instruction must be used in the cyclic program section.

Examples of interaction between user program and web page:

- Check received data
- Assemble and send back data to the web browser making the request

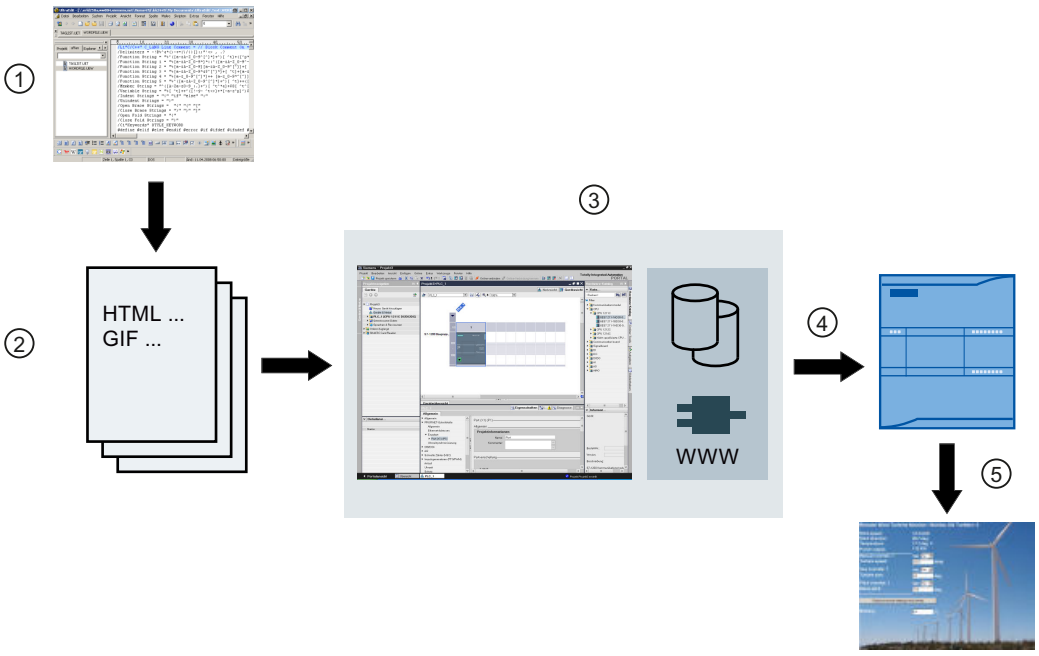
In this case, the status information must be able to be evaluated and control information must be transmitted to the Web server, for example, to release a requested web page.

Procedural overview (S7-1200)

Basic information

This section provides a step-by-step explanation of the basic procedure used to create and download custom web pages and to use them in the operating phase.

The following graphic provides a simplified representation of the process used in creating and displaying custom web pages:



① Programming a web application (using suitable tools when required and AWP commands for dynamic pages when applicable).

② The web application is comprised of single source files, for example, *.html, *.gif, *.js, etc.

③ Using STEP 7:

- Generate the data blocks (Web Control DB and fragment DBs) from source files. The DBs contain meta information and the complete web application, including the images and the dynamic and static parts of the web application. The DBs are stored under "System blocks" in the project tree.
- Call the "WWW" instruction in the user program. This instruction initializes the web server of the CPU for a web application.
- If required, complete final programming for interaction between the web server and user program

④ Downloading the blocks to the CPU.

⑤ Call the web page in the browser. The web pages of the CPU are called by entering the IP address of the CPU.

Additional information

You can find additional information and examples relating to the S7-1200 web server on the Internet (http://support.automation.siemens.com/WW/view/en/36932465).

Creating web pages (S7-1200)

Web design tools from various companies can be used to create user-defined web pages. As a rule, the web pages should be programmed and designed compliant to the conventions of the W3C (World Wide Web Consortium). No check is made for compliance to W3C criteria in the web server of the CPU.

Rules

- The tool must be able to directly edit the HTML code so that the AWP command can be inserted into the HTML page.
  Only the AWP commands are parsed in the CPU and, for example, replaced by values from the user program/process image of the CPU.
- Files containing AWP commands must be coded in UTF-8. In the metadata of the HTML page, therefore, set the attribute charset to UTF-8 and save the file UTF-8 coded.
- Files containing AWP commands must not contain the following sequence:]]
- Files containing AWP commands must not contain the following sequence outside of the "Tag read ranges" (:=<Tag name>:): :=
  Tip: Replace the first character of a prohibited sequence with its character coding; for the colon, for example, :.

A small example for a custom web page should make clear the basic design.

Requirement

- The CPU must have a web server and the web server of the CPU must be activated.
- To be able to access PLC tags with write access as a user, you must be logged on as "admin".
- For the example below, PLC tags must be defined for those PLC tags that are to be shown on the web page. This is shown here for the first tab used, "Tank_below_max".

	Name	Data type	Address
1	Tank_below_max	Bool	%I0.0

Creating user-defined web pages

The following code for an example web page reads values from the process image and provides them in a table.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/
TR/html4/strict.dtd">
<html>
  <head>
   <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
```

```
   <title>Mix</title>
  </head>
   <body>
    <h1>Mix</h1>
    <h2> Actual State </h2>
    <table border="1">
     <tr>
      <th>Variable</th>
      <th>State</th>
     </tr>
     <tr>
      <td>Tank below max</td>
      <td>:="Tank_below_max":</td>
     </tr>
     <tr>
      <td>Tank above min</td>
      <td>:="Tank_above_min":</td>
     </tr>
    </table>
   </body>
</html>
```

AWP commands for controlling Web pages (S7-1200)

Using AWP commands in Web pages (S7-1200)

With AWP (Automation Web Programming) commands, you declare the interface between your user pages (Web application, e.g. a simple HTML page) and the CPU data.

To develop user pages or Web applications, you need only observe the restrictions of the Web browser. In one of the programming languages of STEP 7, control with the user program in the CPU which CPU data is displayed at what time in the Web browser of the viewer.

Use AWP commands, which you comment within the HTML page, to declare data to be used for intentional interaction between the Web application and the user program.

You insert AWP commands as HTML comments with a special syntax into HTML pages. Using AWP commands, the following features can be implemented:

- Read PLC tags
- Write PLC tag
- Read special tags
- Write special tags
- Define enum types
- Assign tags to enum types
- Defining fragments
- Import fragments

Syntax of AWP commands

An AWP command begins with "<!-- AWP_" and ends with " -->". In JavaScript files, the commands should also be enclosed by JavaScript comments ("/*...*/").

Notation rules for PLC tag names within an AWP command

The AWP commands "AWP_In_Variable" and "AWP_Out_Variable" contain a name attribute and optionally a use attribute. A PLC tag name is assigned to these attributes, by means of which the PLC tags in the browser are written or read. The following rules apply to handling PLC tag names in HTML code:

- PLC tags must be enclosed in quotation marks (" ... ").
- PLC tags used in AWP commands must also be enclosed by single quotation marks ('" ... "') or with quotation marks masked by a backslash ("\" ... \"").
- If the PLC tag name contains the character \ (backslash) or * (star), this character must be designated with the escape sequence \\ as standard character of the PLC tag name. See below for examples.
- If the PLC tag name in the AWP command is also enclosed by single quotation marks and the single quotation mark (') occurs within the name, it must also be designed as normal character by the escape sequence \'.
- If an absolute address (input, output, bit memory) is used in AWP command, it is enclosed by single quotation marks.

PLC tag	PLC tag in HTML code
"Velocity"	<!-- AWP_In_Variable Name='"Velocity"' -->
	<!-- AWP_In_Variable Name="\"Velocity\"" -->
"abc\de"	<!-- AWP_In_Variable Name='"abc\\de"' -->
"abc'de"	<!-- AWP_In_Variable Name='"abc\'de"' -->
"abc'de"	<!-- AWP_In_Variable Name="abcde" Use'"abc\'de"' -->
"DB name".tag	<!-- AWP_In_Variable Name='"DB name".tag' -->
"DB name"."ta.g"	<!-- AWP_In_Variable Name='"DB name"."ta.g"' -->
-	<!-- AWP_Out_Variable Name=’flag1’ Use='M0.0' -->

See also

Reading tags (Page 1061)

Writing tags (Page 1064)

Special tags (Page 1065)

Reading tags (S7-1200)

User-defined web pages can read PLC tags.

The PLC tag must be specified by a PLC tag name.

These OUT variables (direction of output as viewed from the controller) are inserted at any location within the HTML text with the syntax described in the following.

Syntax

```
:=<varname>:
```

These references are replaced when the Web server is in operation by the current values of the PLC tag in each case.

<varname> can be a simple, global PLC tag but also a complete tag path to a structural element.

Notation rules for PLC tag names

- PLC tags in HTML code are enclosed by quotation marks ("), if they are defined in the tag table. In the case of data block tags, the name of the data block is enclosed by quotation marks. If special characters are used in the structure elements of the data block, for example the dot (.) or blank, this part must also be enclosed by quotation marks.
- Quotation marks are not used for absolute addresses of inputs, outputs or bit memories.

PLC tag	PLC tag in HTML code
"DB_name".var_name	:="DB_name".var_name:
"DB_name"."var.name"	:="DB_name"."var.name":
"memory"	:="memory":
-	:=I0.0:
	:=Q0.0:
	:=MW100:
	:=%MW100:
"My_Data_Block".flag1	<!-- AWP_Out_Variable Name='flag1' Use='"My_Data_Block".flag1' -->
	...
	:=flag1:

- If the PLC tag name contains the character : (colon) or \ (backslash), this character must be designated with the escape sequence \: or \\ as standard character of the PLC tag name.

PLC tag	PLC tag in HTML code
"abc:de"	:="abc\:de":
"abc\de"	:="abc\\de":

- Special characters "<, &, >"
  Display problems can occur if these characters are contained in the tag name (for example, "a<b").
  Avoid expressions such as :="a<b": in the HTML page.
  To prevent display problems from occurring, use e.g. an AWP command with a use expression according to the pattern depicted below. The use attribute defines the PLC tag with the problematic character, the name attribute defines the name without problematic character, as it is used in the HTML page.

PLC tag	PLC tag in HTML code
"a<b"	<!-- AWP_Out_Variable Name='simplename' Use='"a<b"' --> ... :=simplename:

Reading tags of the type String and Character

Below, these types of quotation marks are used in the explanation: single quotes ('), double quotes (").

As of firmware V1.6, with the "Read PLC tags" function, an S7-1500 CPU outputs tags of the type String or Character enclosed in single quotes to the browser.
Example:

- String tag "Varname".MyString with the content ABC
- You read the tag in HTML using the function :="Varname".MyString:
- The Web server outputs the character string 'ABC' to the browser

Using String or Character tags in expressions

On your HTML page, you use an expression in which the character string for reading a tag is enclosed in quotes, for example in forms.

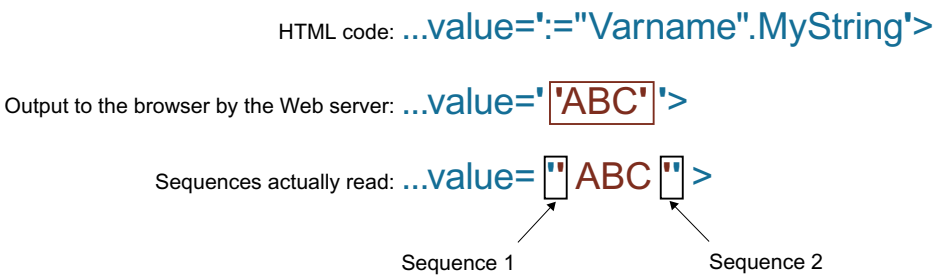
Possible HTML code used:
```
<input type="text" name="appfield" value="myvalue">
```

If you read the displayed value for the "value" attribute from a PLC tag in this expression, the HTML code appears as follows:
```
<input type="text" name="appfield" value=":="Varname".MyString:">
```

By reading the PLC tag, the Web server outputs the value 'ABC'. In HTML, the code is then represented as follows:
```
<input type="text" name="appfield" value=" 'ABC' ">
```

If you have used single quotes instead of double quotes in your HTML code to enclose the attributes, the Web server passes on the content of the tags enclosed in two single quotes to the browser. As a result of this, the browser does not output the content of the String or Character tag, since two consecutive single quotes each form a closed sequence. The values to be read are located between these sequences and are not output by the browser.

In this context, note in particular that the character string with double quotes is not identical to two single quotes even if they appear to be identical, as shown in the following figure.



See also

Using AWP commands in Web pages (Page 1060)

Writing tags (S7-1200)

Custom web pages can write data into the CPU.

This requires an AWP command that identifies the PLC tag to be written.

The PLC tag must also be specified by a PLC tag name.

The IN tags (direction of input as viewed from the controller) are placed on the browser page. This can be done, for example, in a form.

The tags are either set in the HTTP header (by cookie or POST method) or in the URL (GET method) by the browser and are then written by the Web server into the respective PLC tag.

Syntax

To allow the IN tags to be written to the CPU, the tags must first be defined by an explicit AWP instruction:

```
<!-- AWP_In_Variable Name='<PLC_Varname1>' Name='<PLC_Varname2>'
Name='<PLC_Varname3>' -->
```

Several tags can be defined in an instruction - such as that shown above.

The specific PLC tag name is hereby written in double quotation marks; for example <PLC_Varname1> = "myVar".

In cases where the name of the tag that you use for the web application is not identical to the name of the PLC tag, the "Use" parameter can be used to assign to a PLC tag:

```
<!-- AWP_In_Variable Name='<Webapp_Varname>' Use='<PLC_Varname>'
```

Example

The "AWP_In_Variable" AWP command is indispensable when handling forms.

```
<form method='post' action='/awp/appl/x.html'>
 <p>
  <input name='"var1"' type='text'>
  <input value='set' name='Button1' type='submit'>
 </p>
</form>
```

In the form defined above, the HTTP request method "post" is used to transfer the tag "var1" to the Web server. The user places the "var1" tag in the form field. The tag 'Button1' has the value 'set', but is not required for the CPU. To allow the "var1" tag to be written to the CPU, the following instruction must be included in the same fragment:

```
<!-- AWP_In_Variable Name='"var1"' -->
```

Since PLC tags are enclosed in double quotation marks ("), the name in the AWP command must be enclosed in single quotation marks (') or in masked quotation marks (\"). To avoid the numerous escape sequences, we recommend the use of single quotation marks.

```
<!-- AWP_In_Variable Name='"Info".par1' -->
<!-- AWP_In_Variable Name="\"Info".par1\"" -->
```

Conditions for write access during operation

The following requirement has to be met in order for a user to be able to write to PLC tags from a user-defined web page:

The user must have rights to change tags. The Web server ignores the commands if the user has no change rights.

This rules applies to all writing access to web pages on a CPU.

See also

Requirements for web access (Page 1052)

Using AWP commands in Web pages (Page 1060)

Special tags (S7-1200)

Special tags are mainly HTTP tags set in the definition of the World Wide Web Consortium (W3C) . Special tags are also used for cookies and server tags.

The AWP command to read and write special tags differ only in that they have additional parameters than the AWP command used to read and write normal tags.

Reading a special tag

The Web server can read PLC tags and transfer these to special tags in the HTTP Response Header. You can, for example, read a URL for a diversion to another web page and transfer to the special tag HEADER:Location using the special tag HEADER:Location.

The following special tags can be read:

Name	Description
COOKIE_VALUE:name	Value of cookie with name: "name"
COOKIE_EXPIRES:name	Execution time of cookie with name: "name" in seconds (must be set beforehand).
HEADER:Status	HTTP status code (if no other value has been set, status code 302 is returned).
HEADER:Location	Path for forwarding to another page. Status code 302 must be set.

Name	Description
HEADER:Retry-After	Anticipated time in which the service is not available. Status code 503 must be set.
HEADER: …	All other header tags can also be forwarded in this way.

Use the AWP command "AWP_Out_Variable" to specify which PLC tags are to be transferred in the HTTP header to the web browser.

Basic structure:

```
<!-- AWP_Out_Variable Name="<Typ>:<Name>" [Use="<Varname>"] -->
```

Parameter description

- Name: Type and name of special tag
- Use (optional parameter): In cases where the name of the special tag is not identical to the name of the PLC tag, parameter "Use" can be used to assign to a PLC tag.

Example:

```
<!-- AWP_Out_Variable Name="COOKIE_VALUE:siemens" Use='"info".language' -->
```

Writing a special tag

In principle, all HTTP tags written in the HTTP header by the web browser can be evaluated by the user program of the CPU. Examples of tag types:

Name	Description
HEADER:Accept-Language	Accepted or preferred language
HEADER:Authorization	Proof of authorization for a requested resource
HEADER:Host	Host and port of the requested resource
HEADER:User-Agent	Information on the browser
HEADER: …	All other header tags can also be forwarded in this way
SERVER:current_user_id	Indicates whether a user is logged on (current_user_id=0: no user logged on)
SERVER:current_user_name	User name of the user logged on
SERVER:GET	Request method is GET
SERVER:POST	Request method is POST
COOKIE_VALUE:name	Value of cookie with name: "name"

The AWP command "AWP_In_Variable" is used to define which special tags are to be evaluated in the user program of the CPU.

Basic structure:

```
<!-- AWP_In_Variable Name="<Typ>:<Name>" [Use="<Varname>"] -->
```

Parameter description:

Name: Type and name of special tag

Use (optional parameter): In cases where the name of the special tag is not identical to the name of the PLC tag, the parameter Use can be used to assign to a PLC tag.

Examples:

```
<!-- AWP_In_Variable Name="COOKIE_VALUE:siemens" Use='"info".language' -->
```

The tag name in the HTTP header is replaced by the PLC tag name specified by Use . The cookie is written to the PLC tag "info".language .

```
<!-- AWP_In_Variable Name='COOKIE_VALUE:siemens' Use='"info".language' -->
```

The tag name in the HTTP header is replaced by the PLC tag name specified by Use. The cookie is written to the PLC tag "info".language .

```
<!-- AWP_In_Variable Name='"COOKIE_VALUE:siemens"' -->
```

The HTTP-header variable is written in the same-name PLC variable.

See also

Using AWP commands in Web pages (Page 1060)

Enumeration types (S7-1200)

Enumeration types (enums)

Numerical values from the PLC program can be converted into text and vice versa using enums. The numerical values can also be assigned for several languages.

Creating enums

Enter an AWP command using the following syntax at the start of the HTML file:

```
<!-- AWP_Enum_Def Name="<Name of the enum type>"
Values='0:"<Text_1>", 1:"<Text_2>", ... , x:"<Text_x>"' -->
```

For example, for German values to be saved as an HTML file in the "de" folder of the HTML directory:

```
<!-- AWP_Enum_Def Name="Enum1" Values='0:"an", 1:"aus", 2:"Störung"' -->
```

For example, for English values, to be saved as an HTML file in the "en" folder of the HTML directory:

```
<!-- AWP_Enum_Def Name="Enum1" Values='0:"on", 1:"off", 2:"error"' -->
```

Assigning enums

Tags are assigned from the user program to the individual enum texts using a special AWP command:

```
<!-- AWP_Enum_Ref Name="<VarName>" Enum="<EnumTypeName>" -->
```

<VarName> is thereby the symbolic name from the user program and <EnumTypeName> is the previously set name of the enum type.

Note

In each fragment in which enum texts are referenced by a PLC tag, this PLC tag must be assigned by the appropriate AWP command of the enum type name.

Ensure that no AWP command for importing fragments is positioned between an enum assignment and enum usage because this import can result in the enum assignment lying in a different fragment than the enum usage.

Example

Enum type "state" is defined with values "0" and "1". "0" means "off", "1" means "on":

```
<!-- AWP_Enum_Def Name="state" Values='0:"off", 1:"on"' -->
```

The following code is contained in the HTML code of the web page to be output:

```
<!-- AWP_Enum_Ref Name="operating state" Enum="state" -->
:=operating state:
```

Depending on the value of the "operating state" tag, the ´result displayed is no longer "0" or "1", but "off" or "on".

Simplified use of enumeration types (S7-1200)

At S7-1200 CPUs as of firmware version 4 it is possible to use enumerations directly in AWP commands to read and write PLC variables.

You create enums as described in the previous section, and you can then utilize the values with user program read and write commands.

Creating enums

```
<!-- AWP_Enum_Def Name="<Name des Enum Typs>" Values='0:"<Text_1>",
1:"<Text_2>", ... , x:"<Text_x>"' -->
```

Utilizing enums in the user program read and write commands

```
<!-- AWP_In_Variable Name='<Varname>' Enum="<EnumType>" -->
<!-- AWP_Out_Variable Name='<Varname>' Enum="<EnumType>" -->
```

Example of reading PLC tags

```
<!-- AWP_Enum_Def Name='AlarmEnum' Values='0:"No alarms", 1:"Tank is
full", 2:"Tank is empty"' --><!-- AWP_Out_Variable Name='"Alarm"'
Enum="AlarmEnum" -->...<p>The current value of "Alarm"
is :="Alarm":</p>
```

If the value of "Alarm" in the CPU is "2", the following text will be displayed on the HTML page:

'The current value of "Alarm" is Tank is empty' because the enum definition assigns the string "Tank is empty" to the numerical value 2.

Example of writing PLC tags

```
<!-- AWP_Enum_Def Name='AlarmEnum' Values='0:"No alarms", 1:"Tank is
full", 2:"Tank is empty"' --><!-- AWP_In_Variable Name='"Alarm"'
Enum='AlarmEnum' -->...
<form method="POST">
<p><input type="hidden" name='"Alarm"' value="Tank is full" /></p>
<p><input type="submit" value='Set Tank is full' /></p>
</form>
```

Because the enum definition assigns the string "Tank is full" to the numerical value "1", the value "1" is written to the PLC tag "Alarm".

Definition of fragments (S7-1200)

Fragments

Fragments are "logical sections" of a web page to be processed by the CPU individually.

Fragments are usually complete pages but can also be individual elements such as files (for example, images) or complete documents.

Defining fragments

```
<!-- AWP_Start_Fragment Name="<Name>" [Type="<Type>"] [ID="<Id>"]
[Mode=<Mode>]-->
```

The start of a fragment is specified by this command. A fragment runs to the start of the next fragment or to the end of the file.

- <Name> Indicates the name of the fragment.
  The name must start with a letter [a-zA-Z] or an underscore (_). Letters, underscores or numbers [0-9] can follow after this first character.
- <Type> Indicates the type of the fragment.
  - "manual" The user program is informed of the request for a fragment; the web page to be returned can be changed by the user program.
  - "automatic" The page is automatically processed (default).
- <id> A numeric ID can be stipulated for the fragment. If no ID is assigned, the fragment is automatically assigned an ID. For manual pages (<Type>=manual) , the fragment can be addressed in the user program of the CPU by this ID.

  Note

  Keep the ID low because the highest ID influences the size of the Web Control DB.

- <Mode> Fragments support the visible and hidden modes.
  - "visible" The fragment is a part of the web page. This mode is preset and can also be omitted.
  - "hidden" The fragment is not part of the web page. However, the fragment will be saved in the Web DB and is available to the user program for inserting in a requested web page. You use an exchange of the fragment ID (Web-Control-DB.fragment_index tag) to insert a "hidden" fragment in the requested web page.

The input document is completely divided into fragments by the "AWP_Start_Fragment" command. "AWP_End_Fragment" is therefore unnecessary.

Without a start fragment command, a file is mapped as a fragment; the fragment name is derived from the file name. If a file is divided into several fragments (by "AWP_Start_Fragment"), the file must begin with the "AWP_Start_Fragment" command.

Importing fragments (S7-1200)

You can declare a fragment in an HTML page and import this fragment into other web pages.

Example

A company logo is to be displayed on all web pages of a web application.

There is only one instance of the HTML code for the fragment that displays the company logo. You can import the fragment as often and into as many HTML files as required.

Syntax

```
<!-- AWP_Import_Fragment Name = "<name>"-->
```

- <name> is the name of the fragment to be imported.

Example

HTML code within a web page that declares a fragment:

```
<!-- AWP_Start_Fragment Name = "My_Company_Logo" -->
<p><img src = "compay_logo.jpg"></p>
```

Example

HTML code within another web page that imports the declared fragment:

```
<!-- AWP_Import_Fragment Name = "My_Company_Logo" -->
```

Creating and loading a data block (S7-1200)

Requirement

- All source files required for the web application (*.html, *.js, *.png, ...) have been created.
- The source files are located in one folder, but only those source files that are required for the web application. No other files may be located in this folder.

Note

Length of file names and tag names

If you have a comprehensive web application with many files and directories, the generation of the web data blocks may possibly fail. If this happens, the generation is aborted with the message "Text list overflow...". The cause is system-internal size limitations for management information saved in the web data block.

Remedy: Use short file names and short tag names.

Procedure

To create data blocks from the source files for user-defined web pages in STEP 7, proceed as follows:

1. Select the CPU, for example, in the device configuration.
2. Select the properties for user-defined web pages in the inspector window under "Properties > General > Web server".
3. As "HTML source", select the folder that contains the source files for the web application.
4. Enter the HTMP page to be opened on starting the web application as the start HTML page.
5. Enter a name for the application if required.
6. You can supplement a range of file name extensions as "Files with dynamic content" if necessary. Only enter those file name extensions that also contain AWP commands.
7. The number for the Web Control DB and for the fragment DB start number can be kept as long as they are not already being used by your user program.
8. Click on the "Generate" button to create DBs from the source files.
   The generated data blocks are saved in the project navigation in the "System block" folder (in the "Web server" subfolder).
9. In the CPU, select the network view to be loaded and then select the "Download to device" command in the "Online" menu to download the blocks. Compilation of the blocks is implicitly initiated before downloading.
   If errors are reported during this process, you must correct these errors before you can download the configuration.

Creating and loading a data block (S7-1200)

Requirement

- All source files required for the web application (*.html, *.js, *.png, ...) have been created.
- The source files are located in one folder, but only those source files that are required for the web application. No other files may be located in this folder.

Note

Length of file names and tag names

If you have a comprehensive web application with many files and directories, the generation of the web data blocks may possibly fail. If this happens, the generation is aborted with the message "Text list overflow...". The cause is system-internal size limitations for management information saved in the web data block.

Remedy: Use short file names and short tag names.

Procedure

To create data blocks from the source files for user-defined web pages in STEP 7, proceed as follows:

1. Select the CPU, for example, in the device configuration.
2. Select the properties for user-defined web pages in the inspector window under "Properties > General > Web server".
3. As "HTML source", select the folder that contains the source files for the web application.
4. Enter the HTMP page to be opened on starting the web application as the start HTML page.
5. Enter a name for the application if required.
6. You can supplement a range of file name extensions as "Files with dynamic content" if necessary. Only enter those file name extensions that also contain AWP commands.
7. The number for the Web Control DB and for the fragment DB start number can be kept as long as they are not already being used by your user program.
8. Click on the "Generate" button to create DBs from the source files.
   The generated data blocks are saved in the project navigation in the "System block" folder (in the "Web server" subfolder).
9. In the CPU, select the network view to be loaded and then select the "Download to device" command in the "Online" menu to download the blocks. Compilation of the blocks is implicitly initiated before downloading.
   If errors are reported during this process, you must correct these errors before you can download the configuration.

Structure of the PLC program (S7-1200)

Your user program must call the " WWW" instruction to even allow the web application, for example, the user-defined web pages, to be available to the CPU on the standard web pages and to allow them to be called up there.

The Web Control DB you have created from the source files is the input parameter (CTRL_DB) for the "WWW" instruction. The Web Control DB references the content of the user-defined web pages coded in the fragment DB and then receives status and control information.

Calling the "WWW" instruction in the startup program

If you do not want the user program to influence requested web pages, it is sufficient to only call the "WWW" instruction once in a startup OB. This instruction initializes communication between the web server and the CPU.

Calling the "WWW" instruction in the cyclic program

The "WWW" instruction can also be called in an OB processed in cycles (for example, OB 1). This has the advantage of being able to respond to web server requests from within the user program. Manual fragments must be used for this.

In this case, you must evaluate information from the Web Control DB in order to identify the requested web page or the requested fragment. On the other hand, you must set a bit in the user program in order to explicitly release the web page to be returned by the web server after processing the web page request.

The structure of the Web Control DB is described in the following section.

Web Control DB (S7-1200)

The Web Control DB (DB 333 by default) is created by STEP 7 and contains information on the structure of user pages, the status of communication and any errors that occur.

Additional fragment DBs are also created as well as the Web Control DB. These fragment DBs (there may also only be one fragment DB) are referenced in the Web Control DB. The fragment DBs contain the web pages and media data coded in fragments, for example, images. The content of the fragment DB cannot be changed by the user program. It is created automatically and is only for data management.

The status and control tags of the Web Control DB are accessed via symbols.

The following lists the tags of the Web Control DB required for status evaluation and to control interaction.

The Web Control DB provides two types of information:

- Global status information: Not bound to a concrete web page request.
- Request status and control information: Information about queued requests.

Global status information

"WEB-Control_DB".commandstate.init	Activates and initializes the web application.
"WEB-Control_DB".commandstate.deactivate	Deactivates the web application.
"WEB-Control_DB".commandstate.inititializing	The web application is initialized (read Web Control DB, etc.).
"WEB-Control_DB".commandstate.error	Web application could not be initialized. The reason is coded in "WEB-Control_DB".commandstate.last_error .
"WEB-Control_DB".commandstate.deactivating	The web application is closed.

"WEB-Control_DB".commandstate.initialized	The web application has been initialized and is ready.
"WEB-Control_DB".commandstate.last_error	Refer to the next table for a value table of possible errors.

Last_error	Description
1	Fragment DB is inconsistent (does not match the Web Control DB).
2	A web application already exists with this name.
3	Memory problem initializing in the web server.
4	Inconsistent data in the Web Control DB.
5	A fragment DB is not available (not loaded).
6	No AWP ID for a fragment DB.
7	The enum fragment is not available (contains the texts and information on the enum types).
8	An action requested via the command flag in the Web Control DB is prohibited in the current state.
9	Web application is not initialized (if there is no reinitializing after disabling).
10	Web server is disabled.
...	Last_error is reset once the web application has been successfully initialized.

Request status information

Request status information is bound to one of four possible requests, x = [1 … 4].

"WEB-Control_DB".requesttab[x].idle	Nothing need be done.
"WEB-Control_DB".requesttab[x].waiting	The user program must react to a request from a manual fragment and explicitly initiate further processing in the web browser.
"WEB-Control_DB".requesttab[x].sending	The web server is occupied with processing the request/fragment.
"WEB-Control_DB".requesttab[x].aborting	The TCP connection is closed by the web server.

Request control information

Request control information is bound to one of four possible requests, x = [1 … 4].

"WEB-Control_DB".requesttab[x].continue	Releases the fragment being processed for transmission. Processing of the next fragment is initiated.
"WEB-Control_DB".requesttab[x].repeat	Releases the fragment being processed for transmission. The fragment is then processed again.

"WEB-Control_DB".requesttab[x].abort	Closes the TCP connection.
"WEB-Control_DB".requesttab[x].finish	Releases the fragment being processed for transmission. Stops further processing of requests (terminates the request).

Example:

The tag for the DB is: "WEB-Control_DB". Whether errors have occurred during initialization of the web application can be determined by requesting bit "WEB-Control_DB".commandstate.error in the user program.

If an error has occurred you can analyze it using the "WEB-Control_DB".commandstate.last_error value.

Interaction with the user program (S7-1200)

With the help of manual fragments, you can make sure that the user program reacts synchronously to browser entries so that the returned website can be prepared by the user program.

Fragment type

To react to the received data in the user program the "manual" fragment type must be used for the fragment writing the data (for "manual pages"):

```
<!-- AWP_Start_Fragment Name="testfrag" ID="1" Type="manual" -->
```

The values are always transferred to the Web server of the CPU for automatic and manual pages in the same way:

Example:

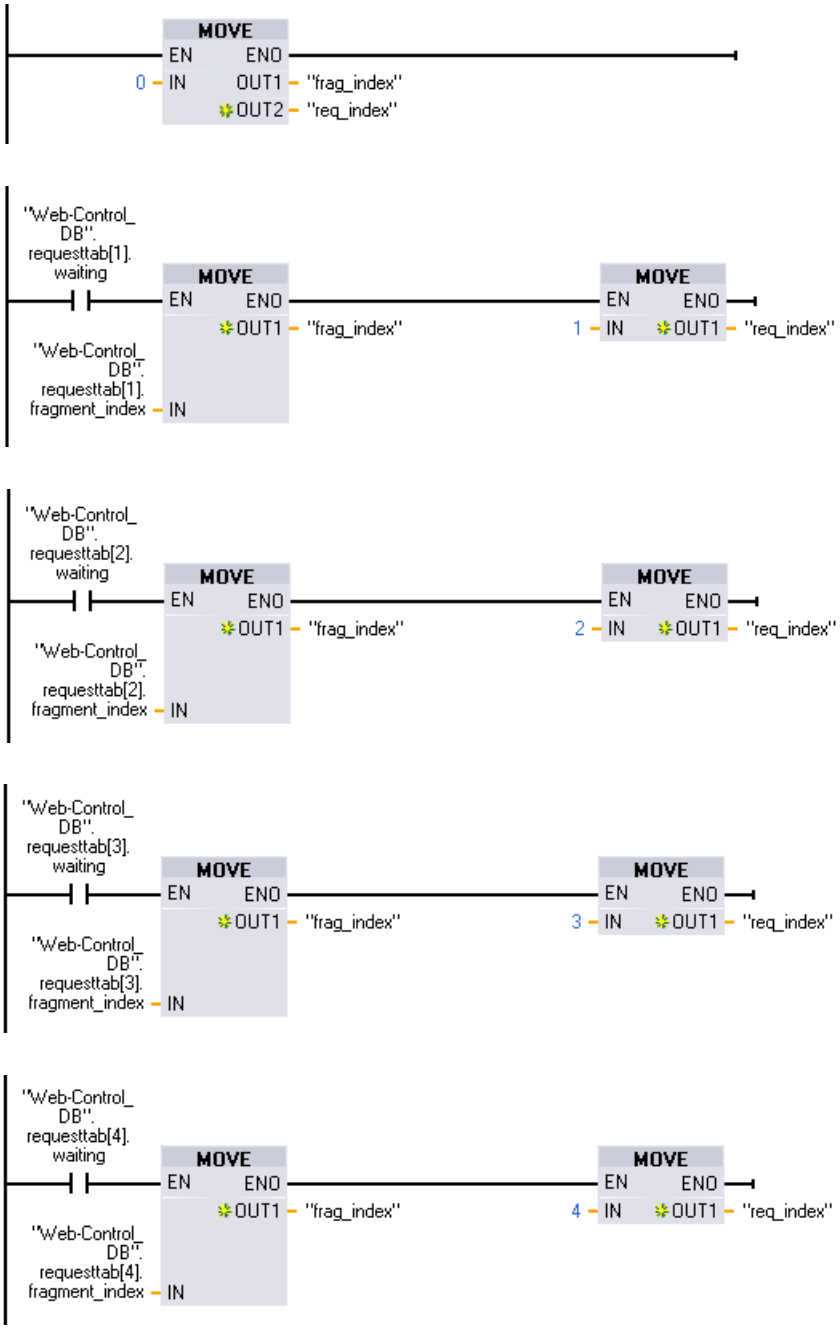
```
<form method="POST" action="">
 <p>
  <input type="submit" value="Set new value">
  <input type="text" name='"Velocity"' size="20">
 </p>
</form>
```

User program for manual fragments

When using manual pages, the "WWW" instruction must be called in cycles in the user program of the CPU.

To react to values entered in the browser, the request - which is made by the manual page to the Web server - must first be evaluated in the user program. The web-control-DB (for example, DB 333) must investigate pending requests to do this. The array that manages four requests is contained in the "requesttab" section of the Web Control DB. Each element of the array contains information about the respective request in a structure.

A simple programming example shows how queued requests are checked based on the tags of the Web Control DB.

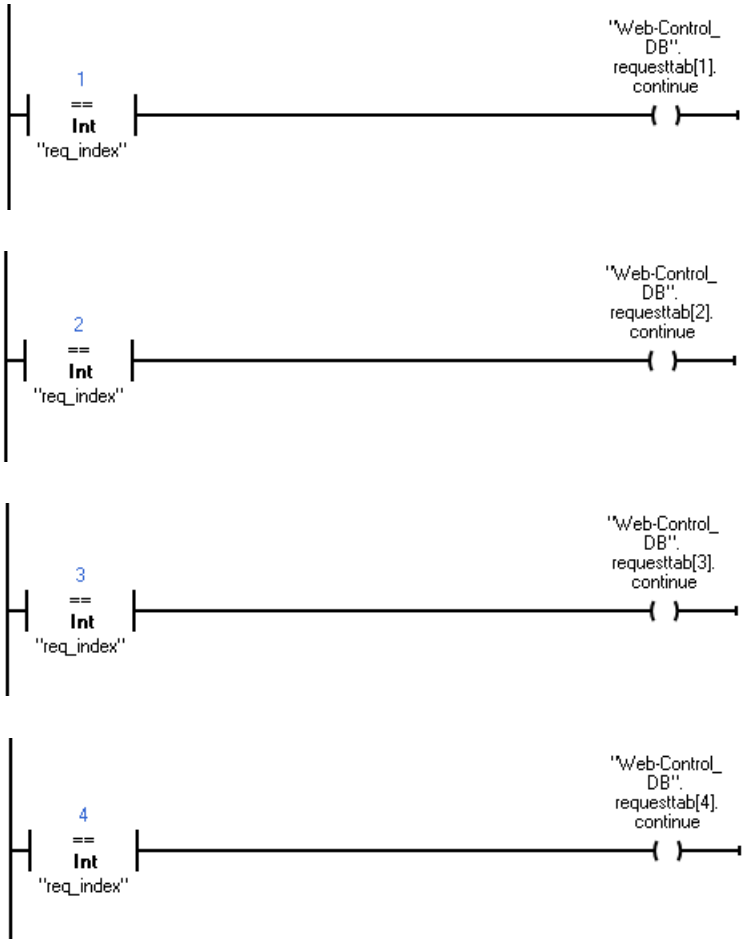


In cases where a request has been made, this program section writes the fragment ID in the #frag_index tag and the request no. (value range 1-4) in the #req_index tag.

Using the information from this, the information transferred in the request can now be processed separately for each fragment ID in the program (for example, plausibility check).

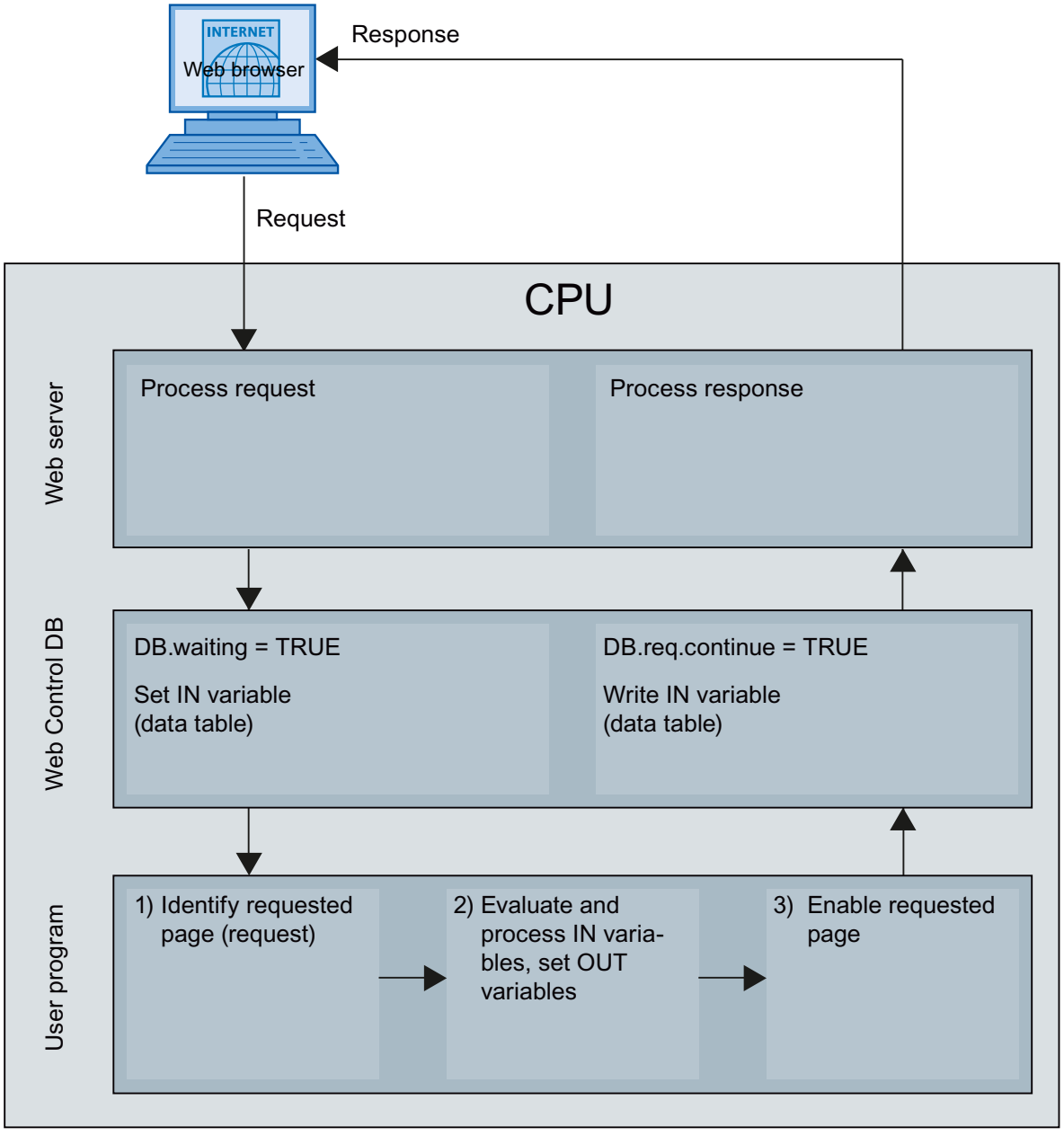
Once processing of the request has been completed by the program, the request must be answered and the appropriate entry is once more reset under"requesttab" of the Web Control DB (for example, DB 333).

A simple programming example for replying to requests:



Principle sequence of a browser request with interaction from the user program

The following figure shows the simplified, principle sequence of the web browser request on the effects of Web Control DB content and the actions required from the user program until the processed web page is returned (response).



Displaying custom web pages in the browser (S7-1200)

Display web pages in browser

Web pages are called from the standard web pages of the web browser.

In addition to the other links in the navigation bar, the standard web pages also have a link to "user pages".

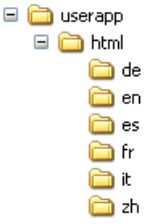
Click on the "user pages" link to open the web browser you have configured as the default HTML page.

Creating custom web pages in several languages (S7-1200)

You can make each of your custom web pages available in various languages.

Requirements

The language-dependent HTML; pages must be stored in a folder structure containing folders with the respective language abbreviations:



Specified language abbreviations

Language abbreviations "de", "en", "fr", "es", "it" and "zh" are fixed. Additional language folders or other designated language folders are not supported.

Additional folders within the same folder hierarchy for other files can be created as required; for example, an "img" folder for images and a "script" folder for JavaScript files.

Selecting an entry page (S7-1200)

Defining the user page as the start page

In addition to the default intro page, you can also define the start page of your user pages as the start page of the Web server.

Follow these steps to define the user pages in STEP 7 as the start page of the Web server:

1. Select the CPU in the device configuration.
2. Open the settings in the Inspector window of the CPU under "Properties > General > Web server".
3. Select the entry "AWP1" in the area "Entry page" under "Select entry page".

If you now enter the IP address of the CPU in the browser, a connection to your user pages is automatically established.

If you want to access the Web pages of your CPU again, link the Web pages from your user pages, e.g. via the URL "http://a.b.c.d./Portal/Portal.mwsl?PriNav=Start" or "https://a.b.c.d/

Portal/Portal.mwsl?PriNav=Start". In this case, the information "a.b.c.d" represents, as an example, the IP address of the configured CPU.

Example of the link in HTML:

```
<a href="/Portal/Portal.mwsl?PriNav=Start">SIMATIC Web pages</a>
```

Note

If you define your user page as start page of the Web server, all direct access to the Web pages of the CPU is blocked. This applies also to the bookmarks you saved for the Web pages of the CPU as well as the page for reading out the service data.

Reading out service data

If you define your user page as the start page of the Web server, all direct access to the Web pages for reading out the service data is also blocked.
If you want to continue to read out service data via the Web server in a service situation, here is how you can link the service data page directly to your user page.
Just as for the web pages of the CPU, link the service data page e.g. via the URL "http://a.b.c.d/save_service_data" or "https://a.b.c.d/save_service_data"; the "a.b.c.d" here is an example of the IP address of the configured CPU.

Example of the link in HTML:

```
<a href="/save_service_data">Service data</a>
```

Language switching for custom web pages (S7-1200)

Requirements

The HTML pages are contained in the predefined language folders, for example, HTML pages with German text are in the "de" folder, HTML pages with English text are in the "en" folder.

Language switching concept

Language switching is based on a predefined cookie named "siemens_automation_language". If the cookie is set to value "de", at the next web page request or web page update, the web server switches to the web page from the "de" folder.

Similarly, the web server switches to the web page from the "en" folder when the cookie is set to "en".

Example of language switching (S7-1200)

The example is structured as follows:

- The language-dependent HTML files with the same name, for example, "langswitch.html" are located in both language folders "de" and "en". The text to be displayed within the two files are German or English, corresponding to the name of the folder.
- There is an additional "script" folder in the folder structure containing the JavaScript file "lang.js". Functions required for language switching are stored in this file .

Structure of the "langswitch.html" file ("de" folder)

Meta data "content language", charset and path to JavaScript file are set in the file header.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Language" content="de">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Switch language to German page</title>
<script type="text/javascript" src="script/lang.js" ></script>
<head>
```

Language selection is implemented in the body of the file by the "select" HTML element. The select element initiates a list box and contains the "de" option, labeled as "German" and "en", labeled as "English"; "de" is the default.

The "DoLocalLanguageChange(this)" function is called using the "onchange" event handler. The "this" parameter transmits the select object with the selected option to this function. "onchange" calls the function each time the option is changed.

```
<!-- Language Selection -->
<table>
    <tr>
        <td align="right" valign="top" nowrap>
         <!-- change language immediately on change of the selection
-->
          <select name="Language"
onchange="DoLocalLanguageChange(this)" size="1">
                <option value="de" selected >Deutsch</option>
                <option value="en" >English</option>
            </select>
        </td>
     </tr>
</table>
<!-- Language Selection End-->
```

Structure of the "langswitch.html" file ("en" folder)

The header of the HTML file with English text is structured similarly to the HTML file with German text.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Language" content="en">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Language switching english page</title>
<script type="text/javascript" src="script/lang.js" ></script>
```

Language selection is also implemented in the body of the file by the "select" HTML element. In contrast to the German HTML file, the English option is already selected as a default and the test or the labels are in English.

```
<!-- Language Selection -->
<table>
    <tr>
        <td align="right" valign="top" nowrap>
```

```
         <!-- change language immediately on change of the selection
-->
          <select name="Language"
onchange="DoLocalLanguageChange(this)" size="1">
                <option value="de" >German</option>
                <option value="en" selected >English</option>
            </select>
        </td>
     </tr>
</table>
<!-- Language Selection End-->
```

Structure of "lang.js" file (in the "script" folder)

The " DoLocalLanguageChange" function is defined in the Java script file and calls the "SetLangCookie" function with the language selection value. SetLangCookie combines the cookie name and cookie value and then sets the cookie by means of the corresponding document.cookie property. The web page must then be reloaded (top.window.location.reload) to allow the web server to react to the setting of the cookie by displaying the required language.

```
function DoLocalLanguageChange(oSelect) {
        SetLangCookie(oSelect.value);
        top.window.location.reload();
    }

 function SetLangCookie(value) {
        var strval = "siemens_automation_language=";
            // this is the cookie by which the web server
            // detects the desired language
            // this name is required by the web server
        strval = strval + value;
        strval = strval + "; path=/ ;";
            // set path to the application, since otherwise
            // path would be set to the requesting page
            // would not get the cookie.
            // The path for user defined applications follows this
sample:
            // path=/awp/<application name>/<pagename>
            // example: path=/awp/myapp/myappstartpage.htm
            //(where myapp is the name of the web application
            // entered in the web server properties of the cpu)
            /*
            use expiration if this cookie should live longer
            than the current browser session
            var now      = new Date();
            var endttime = new Date(now.getTime() + expiration);
            strval = strval + "; expires=" + endttime.toGMTString()
+ ";";
            */
        document.cookie = strval;
    }
```

Additional examples (S7-1200)

Filling bottles with different drinks

This example describes a bottling plant (http://support.automation.siemens.com/WW/view/en/94681612) that can be monitored and controlled via a user-defined Web page.

It is possible to decide the type of drinks to be bottled using a recipe.

For the control and visualization of the plant, no panel (HMI) is necessary.

A Web browser serves to display the Web page.

The Web page is regularly supplied with current process values.

This is done with an invisible inline frame (iframe) that is re-sent regularly by the Web server. Current process values reach the browser with the inline frame. Using JavaScript, the process values are read from the inline frame and inserted in the main page.

In addition to this, users can call functions on the CPU via the Web page, for example starting or stopping bottling or importing or exporting recipes (refer to the link Bottling plant application (http://support.automation.siemens.com/WW/view/en/94681612)).

Filling and emptying a tank

This example describes a tank for liquids (http://support.automation.siemens.com/WW/view/en/58862931) that can be monitored and controlled via a user-defined Web page. Using buttons, the user controls the inflow into the tank and the outflow from the tank.

The current fill level is displayed along with the last ten fill levels with time information

The pages "Overview" and "Data" read in the current process values at fixed intervals whereby the Web server sends the pages completely new to the browser.

The "DataOpti" page, on the other hand, uses an inline frame simply to receive the current fill levels of the tank from the Web server (see Tank simulation application (http://support.automation.siemens.com/WW/view/en/58862931)).

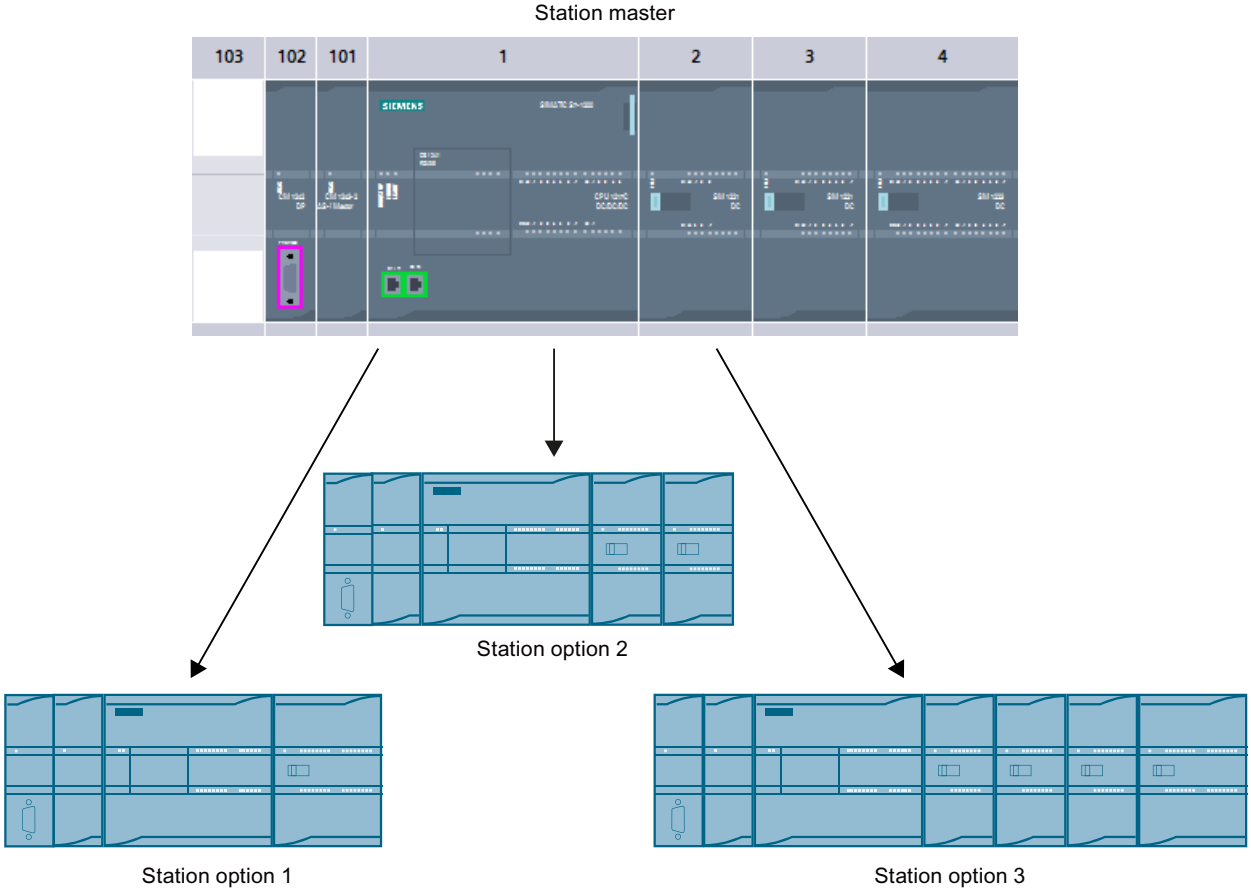
Configuration control (option handling) for devices (S7-1200)

Important information regarding configuration control (option handling) (S7-1200)

Operating principle

With S7-1200 Firmware Version 4.1 or higher, the configuration control enables you to configure the structure of a control system and to work with variants (options) that differ from this configuration.

- All modules that are needed in a set of similar plant units or machines are configured in a master project with a maximum configuration (station master).
- Provision is made in the user program of the master project for various station options for various plant units or machines as well as the selection of a station option. A station option uses, for example, only some of the configured modules and these modules are inserted in varying order.
- An operator selects a station option for a specific plant on-site. The operator does not have to modify the project and thus also does not have to download a modified configuration for this.



A control data record you have programmed in the startup program notifies the CPU as to which modules are missing or located in different slots as compared to the preset configuration. The configuration control has no effect on the parameter assignment of the modules.

Configuration control gives you the flexibility to vary the central installation as long as the real configuration can be derived from a preset maximum configuration.

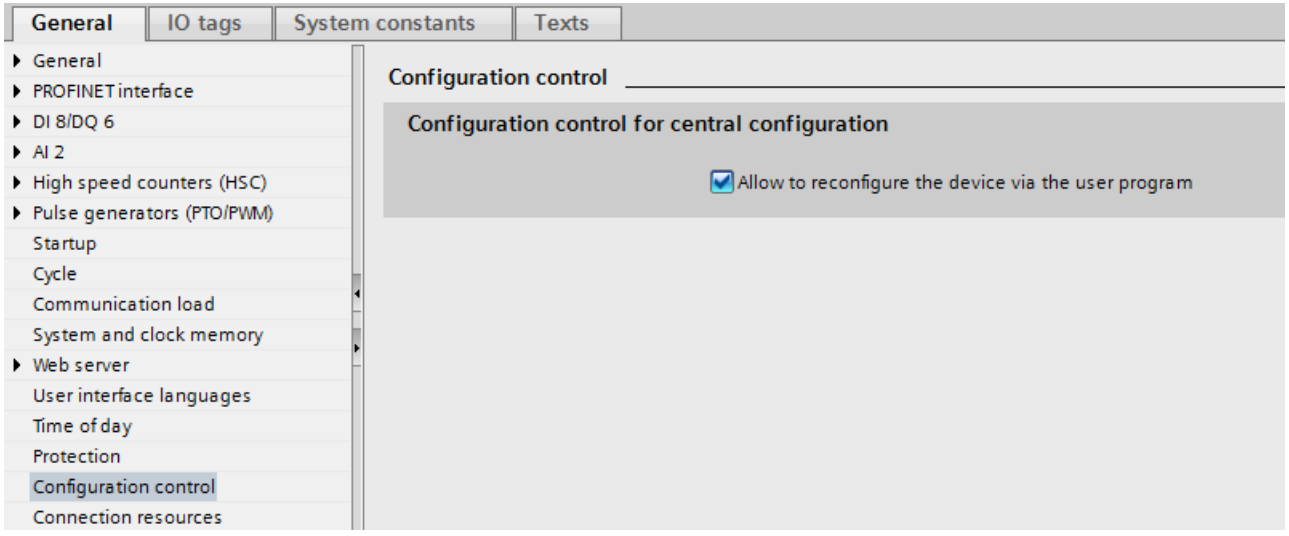
Below you will find a description of how to activate configuration control (CPU parameter assignment) and how to structure the required data record.

Requirements

- STEP 7 Version V13 SP1 or higher
- CPU S7-12XX Firmware Version V4.1 or higher Modules that support the "Configuration control" function also have the "Configuration control" entry in the description (info text) of the hardware catalog.
- Recommendation: Before you load a new program with a modified control data record, perform a memory reset. This action will prevent inconsistent states that may result from the presence of incompatible control data records.
- The startup parameter "Compare preset to actual configuration" is set to "Startup CPU even if mismatch" (default setting).

Required steps

1. Enable the "Allow to reconfigure the device via the user program" parameter when configuring the CPU ("Configuration control" area).



2. Create a control data record (e.g., in a data block) according to the current configuration based on the sample described below for the control data record. The control data record has the number 196. If you want to transfer the control data record as a whole block to the WRREC instruction (input parameter RECORD), note that you must first create a PLC data type containing the structure of the control data record and base the data block on this PLC data type.

ConfDB

	Name	Data type	Start value	...	...	...	...	Comment
1	Static							
2	ConfigControl	Struct			☑	☑		
3	Block_length	USInt	9		☑	☑		Length of control data record, including header
4	Block_ID	USInt	196		☑	☑		Data record number
5	Version	USInt	5		☑	☑		
6	Subversion	USInt	0		☑	☑		
7	Slot_1	USInt	16#FF		☑	☑		Assignment for CPU annex card/Actual annex card
8	Slot_2	USInt	16#FF		☑	☑		Configured slot 2/ Assigned "real" slot
9	Slot_3	USInt	16#FF		☑	☑		Configured slot 3 / Assigned "real" slot
10	Slot_4	USInt	16#FF		☑	☑		Configured slot 4 / Assigned "real" slot
1*	Slot_5	USInt	16#FF		☑	☑		Configured slot 5 / Assigned "real" slot
8	Slot_6	USInt	16#FF		☑	☑		Configured slot 6 / Assigned "real" slot
9	Slot_7	USInt	16#FF		☑	☑		Configured slot 7 / Assigned "real" slot
10	Slot_8	USInt	16#FF		☑	☑		Configured slot 8 / Assigned "real" slot
1*	Slot_9	USInt	16#FF		☑	☑		Configured slot 9 / Assigned "real" slot
8	Slot_101	USInt	16#FF		☑	☑		Configured slot 101 / Assigned "real" slot
9	Slot_102	USInt	16#FF		☑	☑		Configured slot 102 / Assigned "real" slot
10	Slot_103	USInt	16#FF		☑	☑		Configured slot 103 / Assigned "real" slot

3. Transfer the control data record to the CPU in the startup program.
The configuration control for the centrally inserted modules takes effect only after an operating mode change of the CPU from STOP to RUN. For this reason, call the extended WRREC (Write data record) instruction in the startup OB, and transfer the created control data record to the CPU; see next section.

If a valid control data record is not transferred in the startup OB, the control is not ready for operation. The CPU returns from startup to STOP in this case.

Transferring a control data record in the startup program

The CPU processes the WRREC instruction for asynchronous transfer of the control data record. For this reason, you must call WRREC in the startup OB repeatedly in a loop until the output parameter "BUSY" or "DONE" indicate that the data record has been transferred.

Tip: Use the SCL programming language (Page 4991) with the REPEAT ... UNTIL instruction for programming the loop.

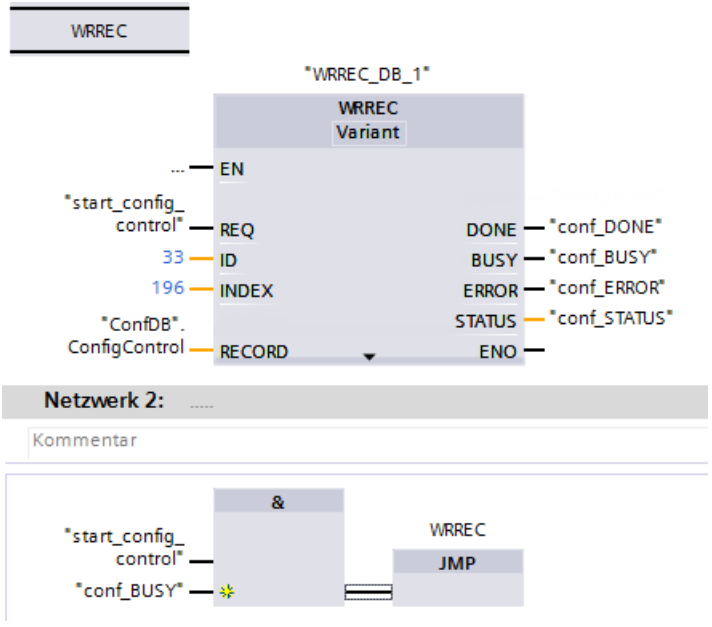
```
REPEAT
    "WRREC_DB"(REQ := "start_config_control",
               ID := 33,
               INDEX := 196,
               LEN := "conf_LEN",
               DONE => "conf_DONE",
               BUSY => "conf_BUSY",
               ERROR => "conf_ERROR",
               RECORD := "ConfDB".ConfigControl,
               STATUS => "conf_STATUS");
UNTIL NOT "conf_BUSY"
END_REPEAT;
```

Below, you will find explanations for individual block parameters that you must supply with specific values in the configuration control context. For the remaining block parameters, see also WRREC (Page 3596):

Parameter	Explanation
ID	HW identifier; is always 33 (decimal) for configuration control for centrally arranged modules.
INDEX	Data record number; is always 196 (decimal) for configuration control for centrally arranged modules.
RECORD	Control data record to be transferred. See the section "Control data record" for the structure of the control data record. Tip: The "RECORD" block parameter of the WRREC instruction (V1.1 and higher) is of the "VARIANT" data type and therefore requires a tag with data type. If you store the control data record in a data block, this data block must thus have a data type. The data block created must not be of the "Global-DB" type. Rather, it must be derived from a user data type. Proceed as follows: 1. First, create a new PLC data type (user data type) with the structure of the control data record and name it, for example, "ConfDB". 2. Create a new data block. Select the newly created user data type, for example, "ConfDB", as type for this data block.

In graphical programming languages, you realize the loop using instructions for program control.

Example in FBD: Use the LABEL (jump label) and JMP (jump at RLO=1) instructions to program a loop.



Arrangement of the modules

The following table shows the slot number assignment:

Slot	Modules	Note
1	Signal board, communication board, battery board	Slot on front of the CPU
2 - 9	Signal modules	Slots to the right of the CPU
101 - 103	Communications modules	Slots to the left of the CPU

Control data record

A control data record 196 containing a slot assignment is defined for configuration control.

The following codes apply:

0	Module is included in the hardware configuration but is not used in the current configuration.
1 to 9, 101 to 103	Current slot of the module
16#FF (255)	A module is not located in this slot in the hardware configuration.

Byte	Element	Code	Explanation
0	Block length	4 + number of slots	Header
1	Block ID	196	
2	Version	5 (for central I/O)	
3	Subversion	0	
4	Assignment of the CPU expansion board	Expansion board, 0 or 16#FF	Control element Describes in each element which real slot in the device is assigned to the configured slot. The structure of a control element is described in the following section.
5	Assignment of configured slot 2	Real slot, 0 or 16#FF	
...	...	...	
12	Assignment of configured slot 9	Real slot, 0 or 16#FF	
13	Assignment of configured slot 101	Real slot or 16#FF	In contrast to signal modules, the real slot of the communications modules must correspond to the configured slot.
14	Assignment of configured slot 102	Real slot or 16#FF	
15	Assignment of configured slot 103	Real slot or 16#FF	

Structure of a control element

A control element contains the information on which module is inserted in which slot.

The byte numbers represent the configured slots in ascending order (see above):

- Byte 4 stands for the configured slot of the expansion board
- Byte 5 to byte 9 stand for the configured slots 2 to 9
- Byte 13 to byte 15 stand for the configured slots 101 to 103

The value that you need to enter in the corresponding byte depends on the following rule:

- If the module is present in the real configuration, enter the real slot number of the module.
  - Example 1: The module configured in slot 2 is located in slot 2.
    Enter value 2 (= actual slot) in byte 5 (= configured slot 2).
  - Example 2: The module configured in slot 3 is located in slot 2.
    Enter value 2 (= actual slot) in byte 6 (= configured slot 3).
- If the module is configured but is missing in the real structure, enter 0 in the byte for the configured slot.
- If a module is not located in this slot in the hardware configuration, enter 16#FF (255) in the byte for the configured slot.

Rules

Observe the following rules:

- The configuration control does not support the repositioning of communications modules. The slot entries in the control data record for slots 101 to 103 must correspond to the real positions of the modules or be defined as not present in the hardware configuration by entering 16#FF (255).
- Slot gaps in the configuration are not allowed. For example, if a signal module is inserted in slot 4 in the real configuration, slots 2 and 3 of the real configuration must also be occupied. The same applies to the slots 101 to 103. If a communications module is inserted in slot 102 in the real configuration, slot 101 must also have a communications module inserted in the real configuration.
- If you have enabled configuration control, the CPU is not ready for operation without a control data record. The CPU returns from startup to STOP if a valid control data record is not transferred in the startup OB. The central I/O is not initialized in this case. The cause for the STOP mode is entered in the diagnostics buffer.
- For addressing the WRREC instruction, use the HW identifier 33 (decimal, for the ID block parameter) to write the control data record.
- The control data record is saved retentively in the CPU, which means that it is not necessary to write the control data record 196 again at a restart if the configuration is unchanged. Prior to commissioning, we recommend that a memory reset be performed for the CPU in order to delete any control data record that is present.
- Slot entries in the control data record outside the configured preset configuration are ignored by the CPU.
- Each real slot may only be present once in the control data record.
- A real slot may only be assigned to one configured slot.

Note

Modified configuration

The writing of a control data record with a modified configuration triggers the following automatic reaction by the CPU:

Memory reset with subsequent startup with this modified configuration.

As a result of this reaction, the retentively saved original data record 196 is deleted and the new data record 196 is saved retentively.

Behavior during operation

Effect of the discrepancy between the configured configuration and real configuration:

- For the online display and for the display in the diagnostics buffer (module OK or module faulty), the hardware configuration is always used and not the differing real configuration. Example: A module supplies diagnostic information. This module is configured in slot 4, but is really inserted in slot 3 (missing module; see example in the next section). In the online view, a configured slot 4 is indicated as faulty. In the real configuration, the module in slot 3 indicates an error via an LED display.

If modules are entered as missing in the control data record, the automation system behaves as follows:

- Modules designated as not present in the control data record do not supply diagnostics and their status is always OK. The value status is OK
- Direct write access to the outputs or write access to the process image of outputs that are not present: Remains without effect; no access error is signaled.
- Direct read access to the inputs or read access to the process image of inputs that are not present: Value "0" is supplied; no access error is signaled.
- Write data record to module that is not present: Remains without effect; no error is signaled.
- Read data record from module that is not present: An error is signaled because a valid data record cannot be returned.

Error messages

The following error messages are returned if an error occurs during writing of the control data record:

Table 10-80 Error messages

Error code	Meaning
16#80B1	Invalid length; the length information in data record 196 is not correct.
16#80B5	Configuration control parameters not assigned.
16#80E2	Data record was transferred in the wrong OB context. The data record must be transferred in the startup program.
16#80B8	Parameter error; module signals invalid parameters.

See also

Basic information on VARIANT (Page 2308)

S7-1200 System Manual (http://support.automation.siemens.com/WW/view/en/89851659)

Example of a configuration control (S7-1200)

A configuration consisting of a CPU and 3 signal modules is configured in the following.

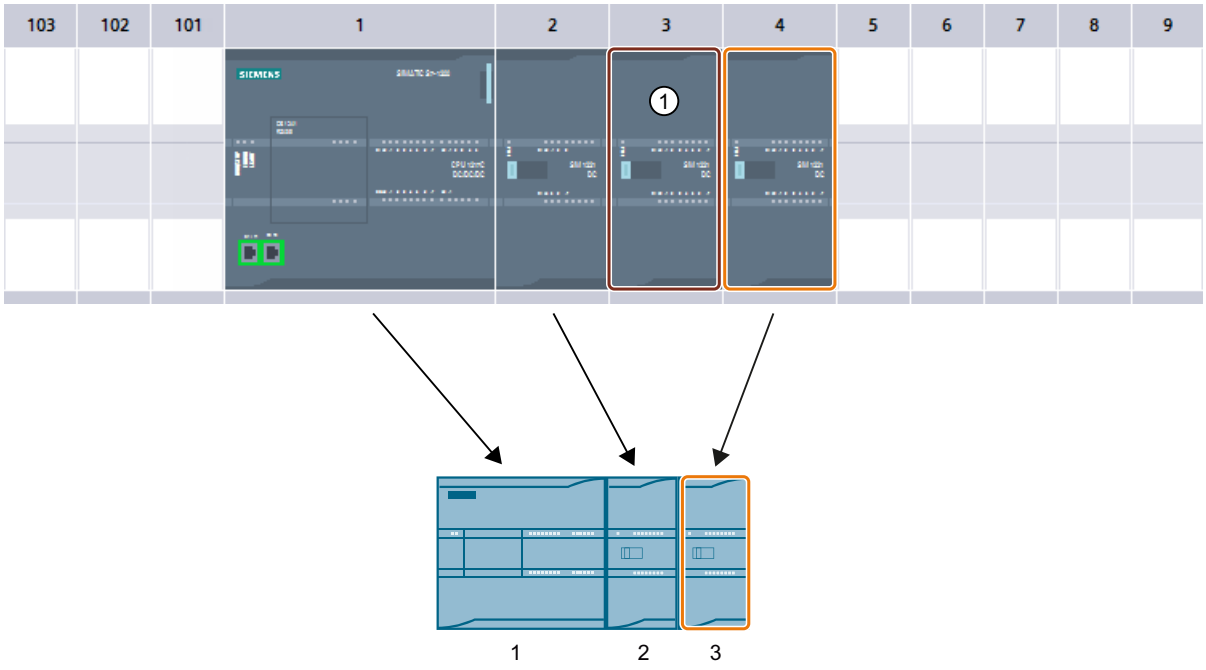
The module at slot 3 is not present in the first configuration expansion and is "hidden" by the configuration control.

In the second configuration expansion, the module that was initially hidden is located in the last slot. The added slot is made known to the CPU by a modified control data record.

Actual configuration with missing module

The specified configuration contains all modules that can be present in a final expansion stage.

The module that is inserted in slot 3 in the specified configuration is missing in the real expanded configuration. For this reason, slot 3 must be coded in the control data record accordingly with "FF_H" (= not present).



ControlDataRecord

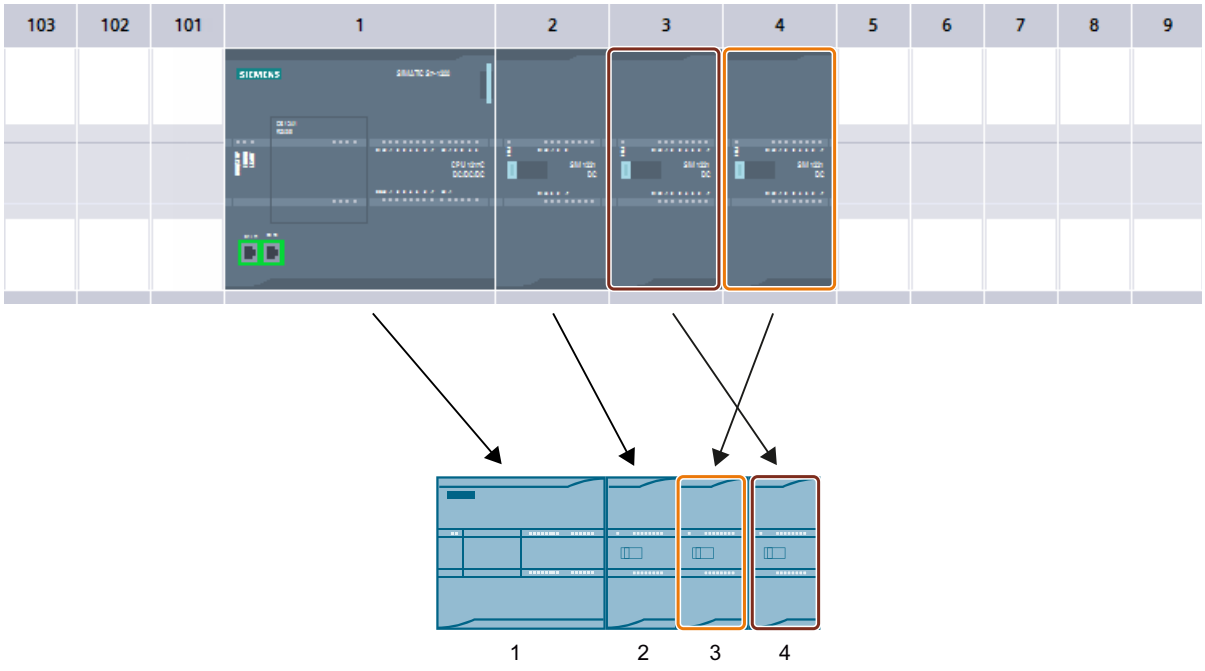
		Name	Data type	Start value	Comment
1		Static			
2		ConfigControl	Struct		
3		Block_length	USInt	16	Length of control data record, including header
4		Block_ID	USInt	196	Data record number
5		Version	USInt	5	
6		Subversion	USInt	0	
7		Slot_1	USInt	255	Assignment for CPU annex card/Actual annex ca...
8		Slot_2	USInt	2	Configured slot 2 / Assigned ①al" slot
9		Slot_3	USInt	0	Configured slot 3 / Assigned "real" slot
10		Slot_4	USInt	3	Configured slot 4 / Assigned "real" slot
11		Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
12		Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
13		Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
14		Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
15		Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
16		Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
17		Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
18		Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

① Module is missing in the real configuration

Actual configuration with subsequently added module

The module present in slot 3 in the specified configuration is added to the back of the real configuration by inserting it as the last module in slot 4.

The control data record is adapted accordingly.



ControlDataRecord

	Name	Data type	Start value	Comment
1	Static			
2	ConfigControl	Struct		
3	Block_length	USInt	16	Length of control data record, including header
4	Block_ID	USInt	196	Data record number
5	Version	USInt	5	
6	Subversion	USInt	0	
7	Slot_1	USInt	255	Assignment for CPU annex card/Actual annex card
8	Slot_2	USInt	2	Configured slot 2/ Assigned "real" slot
9	Slot_3	USInt	4	Configured slot 3 / Assigned "real" slot
10	Slot_4	USInt	3	Configured slot 4 / Assigned "real" slot
11	Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
12	Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
13	Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
14	Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
15	Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
16	Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
17	Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
18	Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

Additional configurations (S7-1200)

Configuring additional functions

The S7-1200 automation system has numerous additional functions that are useful as integrated CPU functions or available via plug-in modules (for example, communication modules). You can find the description via the following links.

See also

Overview of point-to-point communication (Page 997)

General information on high-speed counters (Page 992)

Motion functionality of the CPU S7-1200 (Page 6140)

Configuring PID_Compact V1 (Page 6044)

Configuring PID_3Step V1 (Page 6079)

10.1.4.2 Configuring PC systems

Overview

Important information about PC systems and PC stations

Definition of PC systems

PC systems are available in a wide range of designs, which can flexibly be expanded with software and hardware components. In this case, we are describing PC systems with a software controller that makes a PC (e.g. IPC) into a controller - similar to a modular controller such as a CPU S7-300/400 or a CPU S7-1500.

PC Station and possibly other components are installed in a PC system.

Definition of PC Station

PC Station is a software component that manages SIMATIC software products and the use of interfaces on a PC. In addition, it controls communication among the SIMATIC products on a PC.

PC Station is required by the SIMATIC software products SIMATIC NET PC software, WinAC RT or the SIMATIC S7-1500 software controllers and installed on the target PC.

SIMATIC HMI products such as WinCC Advanced RTX can work with different versions of PC Station.

As of version V2.x, PC Station can also be installed without the software controller.

Selecting the version of PC Station

During configuration, the version of your PC Station is automatically adapted by the software components configured in the PC system.

The following scenario is an example of a version change for PC Station:

- You have selected an IPC from the hardware catalog that conforms to PC Station V2.x:
  - In the initial state of the IPC, PC Station V1.0 is the default.
  - PC Station V1.0 remains installed as long as components that are compatible with PC Station V1.0 are inserted in the PC system.
  - The version of the PC Station only adjusts automatically to V2.x when the S7-1500 version of the software controller is inserted in an IPC.
  - If you **only** insert WinCC Runtime, the display changes from "PC Station" to "not installed". No installation of PC Station is required on the PC.
  - If you insert WinCC Runtime and a software controller is already inserted, the version of PC Station remains unchanged.

Note

When you include incompatible software products in the PC system, you receive an error message when trying to compile the station.

The properties of PC Station indicate which version of PC Station is installed: For the selected PC Station, you can find the settings in the Inspector window in the "PC Station" area.

Reference

You can find more information on operating a software controller with a PC Station in the SIMATIC ET 200SP Open Controller CPU 1515SP PC (https://support.industry.siemens.com/cs/de/en/view/109248384) manual, in the SIMATIC S7-1500 CPU 150xS (https://support.industry.siemens.com/cs/de/en/view/109249299) operating manual and in the SIMATIC NET PC software Commissioning PC Stations (https://support.industry.siemens.com/cs/de/en/view/109488960) configuration manual.

Software components of a PC station

A PC station contains a range of configurable software components, which are applied during loading to the PC station. There are different applications which have different effects on the generation and handling of the configuration data.

- PC station V2.x:
  - WinCC RT Advanced, as of V13 SP1
  - S7-1500 Software Controller
- PC station V1.0:
  - Standard application
  - OPC server as a central communication service provider
  - WinCC RT Advanced
  - WinCC RT Professional
  - WinCC V7
  - WinAC RTX
  - Communications processors
- Not installed:
  - WinCC RT Advanced
  - WinCC RT Professional
  - WinCC Client

You configure the specific properties of these components according to the utilized services.

WinCC RT Advanced

WinCC RT Advanced is a PC-based HMI solution for single-user systems directly at the machine level. The WinCC RT software is integrated in the SIMATIC HMI devices (e.g. Basic/ Comfort/Mobile Panel) and permits data exchange over HMI connections.

If you want to set up several HMI connections to a CPU, use for example:

- The PROFINET and PROFIBUS DP interfaces of the PC station
- CPs and CMs with the relevant interfaces

Other communication options:

- OPC UA client or as server (all HMI versions)
- Communication between HMI systems over the Intranet/Internet
  - Read and write access to tags

S7-1500 Software Controller

The SIMATIC S7-1500 Software Controller is a PC-based controller.

- It offers the same functionality as CPUs of the SIMATIC S7-1500 automation system.
- Software Controllers offer the functionality of a programmable logic controller (PLC) in a PC-based real-time environment.
- In order to develop your user program, use the same programming languages, program structure, and programming interface (STEP 7) with the PC-based controller as those used for hardware controllers.
- For the SIMATIC S7-1500 Software Controller, you can use the same user program as for a hardware controller. The Software Controller CPU has a display application that runs on the PC. The CPU's operating mode can be shown with the display application. With the display application - similar to the display of a hardware CPU - you can execute diagnostics and commissioning tasks.
- The Software Controller makes use of the interfaces of the PC for PROFINET and PROFIBUS.
  Windows-independent use of PROFINET or PROFIBUS for operating distributed I/O.
  Depending on the interface hardware used, the following functions are possible:
  - PROFINET IO RT
  - PROFIenergy
  - I-device
  - PROFIBUS DP master
- Communication (SIMATIC communication, Open User Communication) with Windows applications or external devices via the Windows interfaces of the PC.
  The communications services are based on the module in use.

The Software Controller based on S7-1500 is available in the following versions:

- As SIMATIC ET 200SP Open Controller CPU 1515SP PC and as CPU 1515SP PC including HMI in ET 200SP design, with pre-installed and pre-configured SIMATIC S7-1500 Software Controller. Can be centrally expanded using ET 200SP I/O modules.
- As Software Controller CPU 1507S for PC-based automation. The SIMATIC S7-1500 Software Controller provides the benefits of the standard SIMATIC S7-1500 controller on high-performance industrial PCs.

Standard application

With standard applications, you communicate with other applications and devices via your PC module.

You can configure the following communication services depending on the PC module in use:

- S7 connections
- DP services

OPC server as central communication service provider

OPC clients use the interface to an OPC server to communicate with an automation device (such as a SIMATIC S7-1500).

Configure its properties to adapt the OPC server. The set or changed parameters take effect after the configuration data is downloaded to the PC station and the OPC server is started.

You can use the following communication services depending on the PC module in use:

- All connection types
- DP services (DPV0)

WinAC RTX

The SIMATIC WinAC (Windows Automation Center) is a PC-based controller and provides the same functionality as SIMATIC S7-300/400 CPUs (hardware controllers). WinAC RTX is used as a solution for typical PC applications (for example, MS Office and user programs) and automation tasks in a system.

Distributed I/O is controlled via:

- PROFIBUS
  - The connection of I/Os via PROFIBUS DP takes place over the integrated DP interface of the SIMATIC PCs or over a communications processor.
- PROFINET
  - The connection of I/O devices via PROFINET takes place over an integrated Ethernet interface of the SIMATIC IPC or over a communications processor.

Optimum access to the data, e.g. for image processing, measured value acquisition, etc., takes place via the WinAC option Open Development Kit (ODK).

You can use the following communication services depending on the PC module in use:

- PG/OP communication
- S7 communication
- Open User Communication (OUC)
- Process data access via OPC

Communications processors

Connecting the SIMATIC controllers via communications processors enables direct integration of controllers in industrial networks.

You can use the following communication services depending on the PC module in use:

- OPC UA client or as server
- Open communication (TCP/IP and UDP): Multicast with UDP
- PG/OP communication: cross-network via S7 routing
- S7 communication (client, server)
- IT communication (HTTP(S), e-mail)

- PROFINET IO controller with real-time properties (RT and IRT)
- IP address assignment via DCP
- Shared device
- PROFIBUS DP
- PG/OP communication

WinCC RT Professional

WinCC RT Professional is a PC-based control and monitoring system for visualizing and operating processes, production flows, machines and plants – from simple single-user systems to multi-user systems and multi-site solutions with Web clients.

The WinCC RT software is integrated in the SIMATIC HMI devices (e.g. Basic/Comfort/Mobile Panel) and permits data exchange over HMI connections.

If you want to set up several HMI connections to a CPU, use for example:

- The PROFINET and PROFIBUS DP interfaces of the PC station
- CPs and CMs with the relevant interfaces

Other communication options:

- OPC server
- OLE DB server
- WinCC WebNavigator
- WinCC DataMonitor

WinCC Client

Depending on requirements, a WinCC single-user system can be expanded into a high-performance client/server system. In this way, several coordinated HMI stations can be operated in a single group with networked automation systems.

Reference

You can find more information on WinCC RT in the WinCC online help.

You can find additional information on the S7-1500 Software Controller and WinAC RTX in the section What you need to know about PC systems (Page 1096).

You can find additional information about the meaning and handling of PC applications and the OPC server in the SIMATIC NET - Industrial Communication (https://support.industry.siemens.com/cs/de/en/view/https://support.industry.siemens.com/cs/de/de/view/61630799) configuration manual and the SIMATIC NET - Commissioning PC Stations (https://support.industry.siemens.com/cs/de/en/view/109488960) configuration manual.

You can find more information on communication processors on theService&Support (http://w3.siemens.com/mcms/industrial-communication/en/ie/system-interfacing/advanced-controller/Pages/CPs-fuer-advanced-controller.aspx) Web pages.

You can find application examples for changing to the new Software Controller generation in the Guide for Migrating PC-based Controllers: from SIMATIC WinAC RTX to the SIMATIC S7-1500 Software Controller and TIA Portal (https://support.industry.siemens.com/cs/document/109478804/guide-for-migrating-pc-based-controllers%3A-from-simatic-winac-rtx-to-the-simatic-s7-1500-software-controller-and-tia-portal?dti=0&lc=en-WW).

Improvements of PC Station V2.x compared to PC Station V1.0

The common features and improvements of the two PC Stations are listed briefly below.

Common features of PC Station V2.x and V1.0

PC Station V2.x and PC Station V1.0 have the following common features:

- Simple installation of all required software components via a setup (for example, S7-1500 Software Controller, WinCC Advanced RT, etc.)
- PC systems with pre-installed software controller or pre-installed Runtime Advanced can also be ordered.

Improvements of PC Station V2.2 compared to PC Station V2.1

PC Station V2.2 offers the following improvements compared to PC Station V2.1:

- Loading project data in a configuration file (Page 1113)

Improvements of PC Station V2.1 compared to PC Station V2.0

PC Station V2.1 offers the following improvements compared to PC Station V2.0:

- Access possible via a Web server (Page 1118)

Improvements of PC Station V2.0 compared to PC Station V1.0

Compared to PC Station V1.0, PC Station V2.x provides the following improvements:

- Loading the PC Station configuration is easy, similar to a modular S7-1500 controller; no prior configuration of the PC system setup is needed on the IPC.
- Integrated PC hardware diagnostics is integrated into the system diagnostics, for example, to diagnose the state of PC components such as fan or hard disk.

Rules for the configuration

Rules and special features for configuring a PC system

Due to the numerous possible combinations of software and interface cards with various IPC versions, there are complex dependencies.

We recommend simply configuring PC Station according to your requirements and then compiling. STEP 7 performs most of the required adjustments, such as setting the station type.

If a combination cannot be compiled, an error message indicates that a rule has been violated and you receive information about how to rectify the error.

The following sections provide information on the basic rules:

- If an IPC is suitable for an S7-1500 Software Controller, it can also be inserted in a WinAC RTX. Applies to most IPCs.
- The following components are suitable or not suitable for the S7-1500 controller software:
  - The open controller (CPU 1515SP PC) is not suitable for the S7-1500 Software Controller.
  - The IPC2x7D and Embedded Controller (S7-mEC 31) are not suitable for the S7-1500 controller software.
- Device replacement between S7-1500 Software Controller and WinAC RTX is not possible.
- You can only insert one software controller in a PC system in each case.
- Neither an OPC server nor an application (user application) can be used in combination with an S7-1500 Software Controller. An installed PC Station V2.x prevents installation of the OPC server or the application on an IPC.
  Both components are used for a PC Station V1.0.
- Assigning interfaces: Assigning "None, or a different Windows setting" to an interface is only practical for onboard interfaces. The interfaces of the configured CPs/CMs are initially assigned to the PC station, because they are normally used by the configured components of the PC station.
- The graphic view of a PC Station in STEP 7 is identical in the network view and the device view.
  The "slots" on which the components of a PC system are located are not numbered. When you insert components by dragging them from the hardware catalog, STEP 7 places them automatically: The hardware components (e.g. CPs) are on the left side, the software components (e.g. a software controller) are on the right.
  Detailed information, for example, about the order number (i.e. the localization information similar to a slot in the PC system) is available only in the device overview table of the device view.
  - In the device overview table, index 125 is permanently assigned for the PC station (it is also referred as the "Station Manager" for PC Station V1.0). This index is displayed in the first row of the device overview table.
  - For the open controller CPU 1515SP PC, there are the columns "Rack", "Slot" and "I address" and "Q address" in addition to the "Index" column. These columns contain the corresponding information for inserted ET 200SP modules.
  - Inserted software components and CPs obtain their index in the order in which they are inserted in the device view. You can force a specific index only by dragging it from the hardware catalog to the corresponding row in the device overview table. This has no impact on the representation in the graphic device view.
  - You can find onboard interfaces to which no software components have been assigned starting at index 100. As soon as you have assigned an onboard interface, it is assigned the next free index, for example 4. In this way, assigned interfaces always appear at the start of the device overview table.

IPCs, CPs and software components for PC Station V2.x

The following IPCs are suitable for PC Station V2.x:

- IPC227E, IPC277E
- IPC427D, IPC477D
- IPC427E, IPC477E
- as of IPC547G (without CPU 1505SP software controller)
- IPC627D, IPC647D, IPC677D
- IPC827D, IPC847D

The following CPs are suitable for PC Station V2.x (can only be used as an interface of the S7-1500 Software Controller, CPU 1507S):

- IE general
- CP5622
- CP5623

The following software components are suitable for PC Station V2.x:

- CPU S7-1507S Software Controller
- WinCC RT as of V13 SP1

Procedure for configuring a PC system

Introduction

The following section describes how to select a PC system from the hardware catalog in the network view. Drag the required modules from the hardware catalog to this PC system; they will be arranged automatically.

Selecting the components in the hardware catalog

Each component is displayed as a folder in the hardware catalog. When you open this folder, you see the different versions of the selected component with their respective article numbers.

The following provides an example of how to create a PC system in the network view.

Requirement

- The hardware catalog is open.
- You are in the network view.

Procedure

To configure a PC system with a required software controller, follow these steps:

1. Navigate to the "PC Systems" folder in the hardware catalog.
2. Open the folder with the required industrial PC type. You will see all types of the selected IPC type.
3. Click the chosen IPC type to select it.
4. Drag the IPC from the hardware catalog and drop it into the network view.
5. Navigate to the "SIMATIC Controller Application" folder in the hardware catalog.
6. Select the desired software controller:
   - CPU 1507S
   - SIMATIC WinAC RTX
7. Drag the desired software controller and drop it onto this PC system.
8. Configure the properties of the PC system and the Software Controller.

Note

Requirements for loading

In order to load the hardware configuration, the software controller or PC station must be assigned to an interface.

Refer the following sections to learn how to assign interfaces: Important information about assigning interfaces (Page 1139) and Assigning interfaces for the communication (Page 1141).

Installation requirements depending on configured components

The configured components of a PC system in STEP 7 determine which installation requirements apply to the target system, for example, to an IPC. These installation requirements must be met in order to successfully load the configuration and enable you to access the PC system with STEP 7 online. STEP 7 shows the applicable (minimum) installation requirements for a selected PC system in the "PC System" area of the Inspector window.

Example: Only WinCC RT configured

If only WinCC RT is configured, "not installed" must be shown in the PC system in the sense of "no PC Station installed".

Therefore, STEP 7 displays "not installed".

Example: WinAC RTX configured

If the WinAC RTX Software Controller is configured, SIMATIC WinAC RTX 2010 must be installed (referred to as "PC Station V1.0" below) on the PC system. SIMATIC WinAC RTX 2010 includes PC Station or S7-RTM, referred to as "PC Station V1.0" in the following, as an installation component.

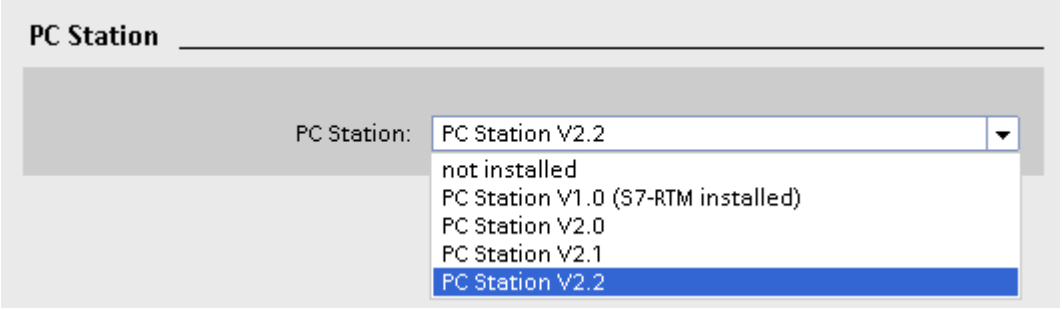
STEP 7 automatically sets this version of PC Station once a WinAC RTX has been configured.

Example: CPU 1507S Software Controller configured

If the CPU 1507S Software Controller is configured, the SIMATIC S7-1500 Software Controller must be installed on the PC system. This includes PC Station installation component, referred to as "PC Station V2.x" below.

STEP 7 automatically sets this version of PC Station once a SIMATIC S7-1500 software controller has been configured.

- PC Station V2.2 (for Software Controller V2.x as of TIA Portal V14 SP1)
- PC Station V2.1 (for Software Controller V2.0 as of TIA Portal V14)
- PC Station V2.0 (for Software Controller V1.x as of TIA Portal V13 SP1)



Reference

You can find additional information about the installation requirements of the relevant PC Station version in the section Loading with STEP 7 (Page 1108).

Special features of the diagnostics in the PC System V2.x

Special features of the diagnostics in the PC System V2.x

The diagnostics concept for a PC System V2.x has the same hierarchical structure as that of a modular CPU. You can see at a glance whether the station is free from faults or if, for example, an underlying component has a fault.

There is only one difference: They have one hierarchical level more than modular CPUs; the PC system is on a level higher than the Software Controller. The PC system has its own hardware status (separate diagnostics).

The following figure shows the diagnostics of the complete PC system in the project tree. The PC station itself is OK (green check mark), but has an underlying component with a fault (exclamation mark in a green check mark). The symbol provides preliminary information about disrupted or missing modules in the underlying structure.

To obtain a more detailed description of the error, double-click on the "Online & Diagnostics" entry in the project tree.

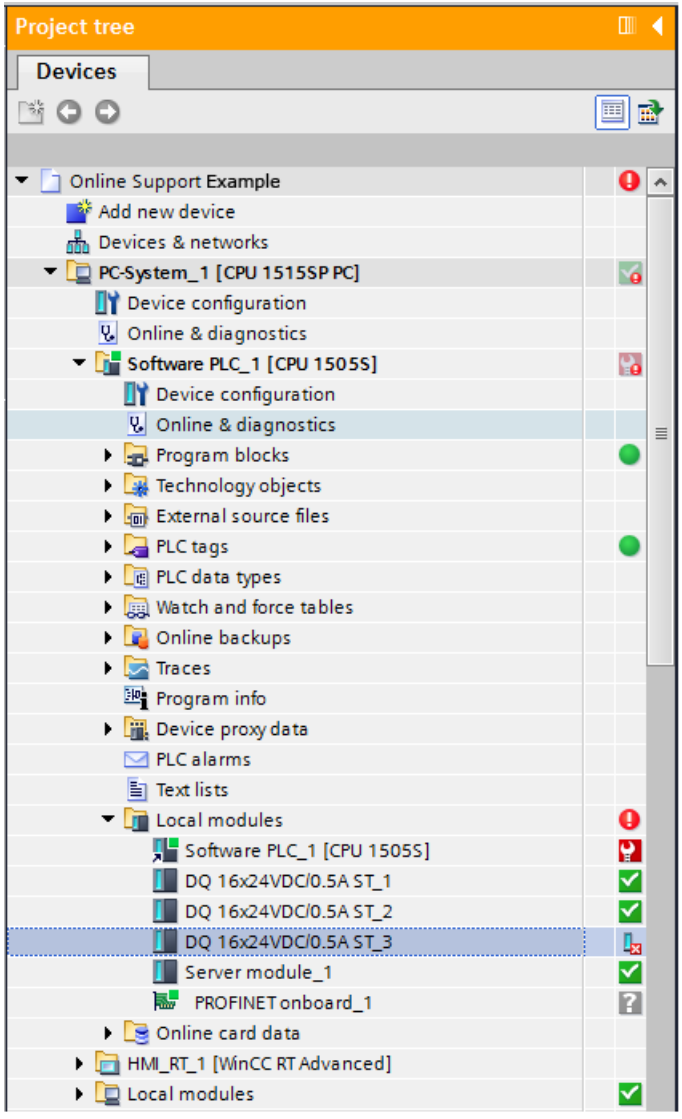
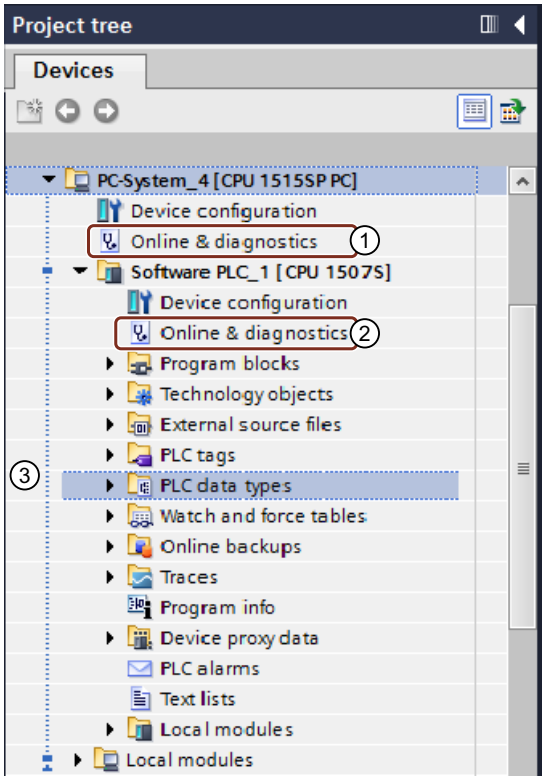


Figure 10-5 Displaying faults in the project tree

The following figure shows the two entries in the project tree for starting the online and diagnostics view of the PC system. For details, refer to the figure's legend.



① Hardware diagnostics of the PC system (for example, fans, temperatures, voltages for IPCs)
② Standard diagnostics (as for S7-1500 CPU)
② Visual brackets for the components of the PC station

Figure 10-6 "Online & Diagnostics" project navigation

Additional information

You can find additional information on "Diagnostics" and the legend with the meaning of each symbol in the Diagnostics (http://support.automation.siemens.com/WW/view/en/59192926) function manual.

Loading with STEP 7

Loading PC station V2.x with STEP 7

The special features for loading a PC System V2.x with STEP 7 are listed below.

Requirement

- The corresponding software components are installed on the PC system (e.g. SIMATIC S7-1500 CPU 1507S).
- If needed, also install the corresponding version of the Setup DVD for SIMATIC WinCC Runtime Advanced on the PC system.

- The SIMATIC IPC hardware component is physically connected via Ethernet to the PC on which STEP 7 is installed.
- The Software Controller is started.

Other settings on the PC station are not required, because PC System V2.x is fully configured by loading the project.

Note

Recommended interfaces for download

- Use a "PN/IE" interface to load a SIMATIC IPC, which is not to be assigned to the Software Controller as a PROFINET interface.
  The "X1" interface is recommended.
- Use the "X2" interface to load a CPU 1515SP PC.

Procedure

To download the STEP 7 project, follow these steps:

1. Select the PC system in the device view.
2. Select the "Download to device" shortcut menu command.
   The "Extended download to device" dialog opens.
3. Configure the settings for the interface.
4. Click the "Download" button to start the download.
   If the PC system is not executed in the "Extended download to device" dialog, this may be due to the lack of an interface assignment.

Result

The project is downloaded step-by-step. A dialog shows the download progress.

- If you have made changes to the PC system configuration (for example, interface assignment, LED, NVRAM or index), then the PC system configuration is downloaded first. If a restart is required after this, STEP 7 displays a corresponding message. You can stop the download at this point, for example, to first configure the Software Controller and then perform the download later. Configurations already loaded are only reloaded if they have changed.
- If you wait instead of stopping the download after the download of the PC system configuration, the PC system reboots and automatically loads the remaining components, such as the configuration of the Software Controller and WinCC Advanced.

Checking the result of the download

After successful completion of the download, the CPU link appears in the Windows Start menu with the name you have assigned in the CPU settings in STEP 7.

The name assigned in STEP 7 is also visible in the CPU display.

Loading PC station V1.0 with STEP 7

The special features for loading a PC System V1.0 with STEP 7 are listed below.

Requirement

- The SIMATIC WinAC RTX DVD is installed on the PC system.
- Each device to which you want to load is accessible via online access.
- The PC station (runtime PC) is linked to the configuration station over a network. The PC station can be reached as communication station, for example, by setting the parameters in the initial configuration.
- An existing connection is checked, for example, with the "Accessible devices" function.
- The target PC station must be configured with the Station Configuration Editor.

Transfer of the configuration

You have the following options for transferring a created configuration to the PC station:

- Online operation
  - Networked (engineering station networked with Runtime PC); access point: "S7ONLINE (STEP 7)"
  - Local (configuration station same as PC station); access point: "PC internal (local)"
- Offline operation (separate engineering station and runtime PC) - Create and import XDB file and WinAC program file (*.wld)

Online operation

Online operation enables direct loading of the configuration data:

- Loading over a PC station connected to a network (MPI, PROFIBUS or Ethernet)
- Loading to the local PC station that is also used as configuration station

Procedure: Online operation - networked

1. Select the station to be loaded in STEP 7.
2. Load the configuration data with "Online" > "Download to device".

Note

For programming device operation

Make sure that you set the correct interface with "Set PG/PC Interface" (access point "S7ONLINE (STEP 7)").

Procedure: Online operation - local

The configuration data is transferred over an internal PC connection in this case.

1. Select the station to be loaded in STEP 7.
2. Load the configuration data with "Online" > "Download to device".

Offline mode

A file of the type XDB is required for data transfer in this operating mode. The configuration system generates an XDB file for each configured PC station.

The storage location of the configuration file is available in the properties of the "PC Station" object in the configuration system. This file can be imported with the "Station Configuration Editor" on the PC station.

You can find more information on offline information in section "Loading PC station V1.0 via storage medium (Page 1111)".

Reference

Additional information can be found in the "NET PC software Commissioning PC Stations - Manual and Quick Start (https://support.industry.siemens.com/cs/de/en/view/109488960)".

Load via file

Load PC Station V1.0 via file

The particularities of loading a PC Station V1.0 via storage medium are listed below.

To load the complete configuration, you have two options:

- Option 1: Create and import a configuration using an XDB file (Page 1111)
- Option 2: Create the program with the help of the memory card and load it to the Software Controller (Page 1112)

Create and import a configuration using an XDB file

Note

WinAC RTX cannot be installed.

Procedure

Follow these steps to configure a PC station offline using an XDB file and then importing it:

1. Drag a device for a PC station into the network view from the hardware catalog under "PC systems".
2. Select the PC station.
3. Change the name of the PC station to suit your needs under "Properties > General" in the Inspector window.

4. Drag-and-drop the additional components required into the PC station from the hardware catalog. An added component is inserted on a new index.

   Note

   You can change the index of a selected component in the Inspector window under "Properties > General > Position number". There may be gaps in index numbering for this reason. These gaps are not displayed in the graphic representation of the device view. However, gaps are displayed in the tabular area of the device view.

5. Select the PC station.
6. Under "Properties > XDB configuration" in the Inspector window, select the "Generate XDB file" check box.
   The connection data and addresses for CPs and applications are saved in the XDB file.
7. Specify the storage path under "XDB file path".
8. Select the PC station.
9. Select the shortcut menu command "Compile > ...".
   When the configuration of a PC station is saved and compiled, system data and the XDB configuration file are generated. This can then be loaded into the target system or be installed.

   Note

   The location of the configuration file is set on the target PC station via the program "Set PG-PC interface" (in the "STEP 7 Configuration" tab).

10. Copy the XDB file to a storage medium that can be accessed by the target station.
11. Import the XDB file to your target PC station with the Station Configuration Editor.

Create the program with the help of the memory card and load it to the Software Controller

Procedure

The configuration data, the current values of the DBs and the STEP 7 user program can be archived, reused and passed on in wld files. The log file (*.wld) does not save the configuration of the PC station in the Station Configuration Editor.

The controller must be in STOP mode for this to take place.

To create a log file, follow these steps:

1. Select the "New..." command in the "Project > Memory card file" menu.
   A dialog opens.
2. Select the directory in which you want to create the file.
3. Enter the file name and the path.
4. Confirm with "Create".

   Note

   You can also create a log file in WinAC with the menu command "File > Archive from CPU".

5. Select the "Open..." command in the "Project > Memory card file" menu to display the created log file in the project tree.
6. Select the directory in which the log file with the file extension (*.wld) is located.
7. The log file is displayed in the project tree under "Card Reader/USB memory".
8. Copy the wld file to a storage medium that can be accessed by the target station.
9. To restore the log file, select the menu command "File > Download to CPU**".**
10. Select the log file with the extension *.wld that you want to restore.
11. Click "OK" to confirm. The STEP 7 user program and the configuration for the Software Controller are loaded again.

Load PC Station V2.x via file

The possibility to save and transport the system configuration of the PC system in a configuration file, offers the following advantages:

- Update of large plants without the TIA Portal
- Simple provision of program and configuration updates
- Plant-level update no longer necessary
- No special software required

Create configuration file

Create configuration file

The entire configuration of your PC station is saved in a configuration file from the TIA Portal. The data may be re-used and distributed. The configuration file has the extension *.psc.

To create a configuration file, follow these steps:

1. Select the menu command "Project > PC Station configuration file > New > PC Station configuration file".
2. Enter the file name in the "Create memory card file" dialog that opens. Make sure your entries here are correct to avoid error messages:
   - Use short, unique name
   - Name can have a maximum of 255 characters
   - Name may not contain spaces
   - Use only permitted characters; permitted characters are letters and numbers, and the special characters '-' and '_'.
3. Select the desired directory in which you want to create the file. In this case too, make sure your entries under point 2 are correct to avoid error messages.
4. Confirm with "Create".

NOTICE
Secure data against access by third parties
The customer is fully responsible for the secure transport of data.

Result

The "Memory card file" folder is created in the project tree under "Card Reader/USB memory" with the following structure:

- PC system configuration file
  This file contains the PC system configuration file. The information indicates the file name and path information, for example: Drive:\PC-SystemConfiguration01.psc
  - Icon "Display PC system info"
    With a double-click on the icon, you can display all information relating to the project, device and module in an editor via the loaded configuration. If additional data is loaded, the user can view the latest metadata using the "Update" button.
  - Folder with station name already assigned in the project navigation, for example, PC-System_1.
    This folder contains the configuration of the PC system.

Loading project data to configuration file

To load data into the PC system configuration file, you have the following options:

- Load project data to a memory card using Drag&Drop or Copy&Paste
- Write project data to a memory card

Requirement

A PC system with PC Station V2.2 or later is configured in the TIA project.

A *.psc file is created and opened in the project tree.

If you use a CPU1515SP PC, Microbox, Nanobox "E", or IPC IPC547G or IPC xxxD in your configuration, then this is only possible in conjunction with a PC Station version ≥ V2.2.

Note

Note the following versions:

- Loading is not possible if you use a CPU1515SP PC F and a CPU 1507S F in your configuration.
- It is not possible to copy a single software CPU in the *.psc file!

Loading the project data on a memory card

To load project data to a memory card file, follow these steps:

1. Drag the project data that you want to load from the project tree to the memory card.
   If necessary, the project data is compiled.
2. Then, the "Load preview" dialog opens. This dialog displays messages and recommends actions necessary for loading.
3. Check the messages and enable the actions in the "Action" column if necessary. As soon as loading is possible, the "Load" button is enabled.
4. Click the "Load" button.
   The loading is performed.

or:

1. Select the "PC system" folder in the project tree.
2. Right-click on the selection and select the "Copy" command from the shortcut menu. Alternatively, you can also use the shortcut <Ctrl+V>.
3. Right-click on the "*.psc" file level or "PC system" folder in the memory card file and select the "Paste" shortcut menu command. Alternatively, you can also use the shortcut <Ctrl+V>. All other levels are locked. If necessary, the project data is compiled.
4. Then, the "Load preview" dialog opens. This dialog displays messages and recommends actions necessary for loading.
5. Check the messages and enable the actions in the "Action" column if necessary. As soon as loading is possible, the "Load" button is enabled.
6. Click the "Load" button.
   The loading is performed.

or:

1. Select the "PC system" folder in the project tree.
2. In the "Project" menu, select the command "Card Reader / USB memory > Write to memory card".
   The "Select memory card Select" dialog opens.
3. Select a memory card.
   A button with a green check mark is enabled in the lower part of the dialog.
4. Click on the button with the green check mark.
   If necessary, the project data is compiled.
5. Then, the "Load preview" dialog opens. This dialog displays messages and recommends actions necessary for loading.
6. Check the messages and enable the actions in the "Action" column if necessary. As soon as loading is possible, the "Load" button is enabled.
7. Click the "Load" button.
   The loading is performed.

Result

The psc file contains the configuration for all components in corresponding subfolders. The name of the folder is changed to the current PC system name.

Note

Check that file is complete

Ensure that the psc file in your TIA Portal is complete, because outside the TIA Portal it cannot be edited.

Open existing configuration files

Open configuration file

To view a configuration file in the project tree, follow these steps:

1. Select the menu command "Project > PC Station configuration file > Open > PC Station configuration file".
2. Select the directory containing the psc file.

The memory card file appears with the mentioned content under "Card Reader / USB memory" in the project tree.

Import configuration file on the target system

The following lists the tools with which you can commission the target system without TIA Portal:

- Command line tool
- PC Station display in the information area (Page 1127)

Requirement

You have administrator rights.

PC Station ≥ V2.2 is installed.

The open controller CPU 1515 Open SP PC (F) and/or an IPC can be accessed.

The psc file is located in a local directory or on a medium such as a USB flash drive.

Overview of the supported command line commands

The following table provides an overview of the supported command line commands:

Command	Explanation
PCSystem_Control.exe /Help	Displays the help text in the command line editor.
PCSystem_Control.exe /HelpExitCode	Displays the help text for the error codes in the command line editor.

Command	Explanation
PCSystem_Control.exe /PrintConfig <file.psc*>	Displays the information about the components; output format: Standard
PCSystem_Control.exe /PrintConfig <file.psc*> /xml	Displays the information about the components; output format: XML format
PCSystem_Control.exe /ImportConfig <file.psc*>	The call starts the full import of the psc file.
PCSystem_Control.exe /GetStatus /ImportConfig	Displays the current status of all the components, the import process; output format: XML format
PCSystem_Control.exe /GetStatus /SimaticComponents	Displays information on all installed Simatic components; output format: XML format

* The <file.psc > parameter stands for the full path and file name, for example, C:\Data\Test.psc

Result

The data is imported in the background.

If a restart is configured, this is performed automatically once the import is completed. If software CPU is contained in the imported configuration, it remains in STOP after the restart.

Procedure for error messages

If an error occurs with a command, you can determine the bit number of the error and to obtain an accurate description of it.

Proceed as follows:

1. To obtain the error code, enter the command "echo% errorlevel%" in the command line tool.
2. This command gives you the following result:
   - If the error code = 0, there is no error
   - If the error code > 0, there is an error and it is output as a decimal number
3. The decimal number must be converted into a binary number.
4. To get an overview of the bit number describing the error, enter the command line command "HelpExitCode".

Example

Here is an example for a conversion and a bit number associated with the corresponding error description:

- Decimal number: 288
- Binary number:100100000

Result: You can read the error at bit number 5 "Err_Net_45_Full=1" and bit number 8 "Err_IIS_Running=1".

Web server

Settings of the Web server on the target system

The Web server functionality is available as of PC Station V2.1. Below you will find the settings and special features for using the Web server.

Requirement

The following software requirements are necessary to use the full range of functions offered by the Web server:

- Windows 7 Embedded is installed
- Windows Ultimate 7 including the Internet Information Services (IIS) ≥ V7.0 is installed (see section Installing the Internet Information Service (Page 1119))
- .NET version ≥ 4.5 is installed

Note

The software must be installed BEFORE installing the PC station.

Principle

The Web server is a Web application and part of the Microsoft Internet Information Server installation. During installation, the Web server integrates itself into the standard website under the pre-installed IIS with access via port 80.
The Web server is used when the URL address contains /simatic or /simatic/.

Example

Examples of how to reach the Web server:

- http://IP address/simatic/ or http://IP address/simatic
- http://computer name/simatic/ or http://computer name/simatic

Reference

You can find additional information on the Web server in the Web server function manual (https://support.industry.siemens.com/cs/de/en/view/59193560).

Enabling the Internet Information Services (IIS)

Below we will show you how to enable the Internet Information Services (IIS) and the required settings.

Note

The additional installation of the IIS is only necessary when Windows 7 Ultimate is installed on your computer.

Requirement

Windows 7 Ultimate is installed.

Procedure

Follow these steps to enable the Internet Information Services (IIS):

1. Click on the button "Start > System Control > Programs and Features > Turn Windows features on or off".
2. Confirm the message for user account control with "Yes".
3. Navigate to the menu command: "Internet Information Services" and make the following settings:

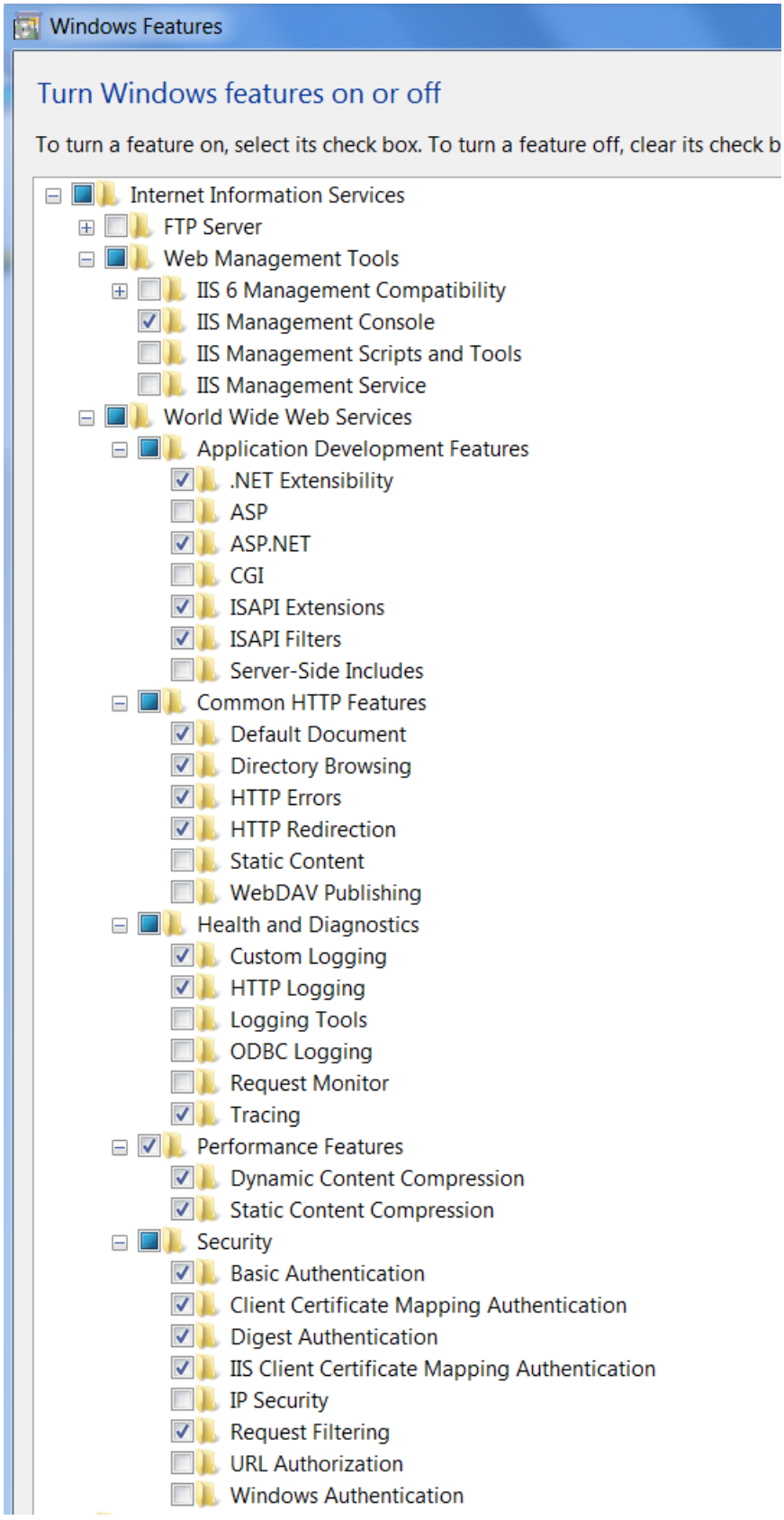

Windows Features
Turn Windows features on or off
To turn a feature on, select its check box. To turn a feature off, clear its check b
Internet Information Services
FTP Server
Web Management Tools
IIS 6 Management Compatibility
IIS Management Console
IIS Management Scripts and Tools
IIS Management Service
World Wide Web Services
Application Development Features
.NET Extensibility
ASP
ASP.NET
CGI
ISAPI Extensions
ISAPI Filters
Server-Side Includes
Common HTTP Features
Default Document
Directory Browsing
HTTP Errors
HTTP Redirection
Static Content
WebDAV Publishing
Health and Diagnostics
Custom Logging
HTTP Logging
Logging Tools
ODBC Logging
Request Monitor
Tracing
Performance Features
Dynamic Content Compression
Static Content Compression
Security
Basic Authentication
Client Certificate Mapping Authentication
Digest Authentication
IIS Client Certificate Mapping Authentication
IP Security
Request Filtering
URL Authorization
Windows Authentication

Figure 10-7 Settings for Internet Information Services

Problems when accessing the Web server

You can find possible causes and their remedies here:

- Internet Information Services were enabled after installation of .NET
- A 64-bit version of Windows is run on the computer

To remedy the problem, you must register a .NET version afterwards in the Internet Information Services (IIS). To do so, execute the following command in the Windows CMD window: C:\Windows\Microsoft.NET\Framework64\v4.0.30319>aspnet_regiis.exe –ir

Configuration of the Web server in the TIA Portal

The procedure for configuring the Web server in the TIA Portal is described below.

Procedure

To use the Web server, follow these steps:

1. Select the desired PC system.
2. Select the following settings in the Inspector window under "Properties > General > Web server":
   - Activate web server on this module
   - Enable automatic update
   - Create and manage users

Reference

You can find additional information on the Web server in the Web server function manual (https://support.industry.siemens.com/cs/de/en/view/59193560).

Setting up the Web server using command line commands

The following command line tool is installed when installing the PC system: PCSystem_Control.exe.

The tool enables you to query information over the Web server and perform the following tasks:

- Query and check the Web server registration
- Register a Web server
- Revoke Web server registration
- Subsequently change the Web server settings

Requirement

The user has administrator rights.

Overview of the supported command line commands

The following table provides an overview of the supported command line commands:

Command	Explanation
PCSystem_Control.exe/Help	Displays the help text in the command line editor.
PCSystem_Control.exe /HelpExitCode	Displays the help text for the error codes in the command line editor.
PCSystem_Control.exe/Help/RegWebserver	Displays the help text for this call.
PCSystem_Control.exe/Help/UnregWebserver	
PCSystem_Control.exe/Help/VerifyWebserver	
PCSystem_Control.exe/RegWebserver/	The Dispatcher Web server registers itself in the Internet Information Services (IIS).
PCSystem_Control.exe/RegWebserver/HTTP	The call expects an existing connection for the HTTP protocol to the standard port "80".
PCSystem_Control.exe/RegWebserver/HTTPS	The call expects an existing connection for the HTTPS protocol to the standard port "443".
PCSystem_Control.exe/RegWebserver/HTTP/HTTPS	The call expects an existing connection for the HTTP protocol to the standard port "80" and for the HTTPS protocol to the standard port "443".
PCSystem_Control.exe/RegWebserver/Sitename"..."	The call expects an existing website in the IIS with the transferred name.
PCSystem_Control.exe/UnregWebserver	The registration of the Web server in the IIS is undone.
PCSystem_Control.exe/VerifyWebserver	Displays whether the user can register the Web server in the IIS or if prerequisites have to be met.

Procedure for error messages

If an error occurs with a command, you can determine the bit number of the error and obtain an accurate description of it.

Proceed as follows:

1. To obtain the error code, enter the command "echo% errorlevel%" in the command line tool.
2. This command gives you the following result:
   - If the error code = 0, there is no error
   - If the error code > 0, there is an error and it is output as a decimal number
3. The decimal number must be converted into a binary number.
4. To get an overview of the bit number describing the error, enter the command line command "HelpExitCode".

Example

Here is an example for a conversion and a bit number associated with the corresponding error description:

- Decimal number: 288
- Binary number:100100000

Result: You can read the error at bit number 5 "Err_Net_45_Full=1" and bit number 8 "Err_IIS_Running=1".

Access via HTTPS

Requirements

The requirements for error-free HTTPS access are as follows:

- The current time-of-day is set.
- The firewall is configured appropriately.
- A valid certificate is installed.
  - If no certificate is installed, a recommendation is displayed that you should not use the page. You also have the option of creating your own certificate.

Input and example

To access the Web pages with the HTTPS protocol, enter the URL in the following format:

- https://ww.xx.yy.zz, where ww.xx.yy.zz
  Example input: https://192.168.3.141

You can recognize an encrypted connection by a lock symbol in the status bar of the Web page.

Reference

You can find additional information on how to create your own certificate in the section Creating your own certificate (Page 1123).

Creating your own certificate

If your company does not have a Certification Authority (CA), you can use the procedure described in this section. The key files are created with the "OpenSSL" program. If you have not yet installed OpenSSL on your PC, you can download and install the program for free from the following website: http://openssl.org/ (http://openssl.org/)

Procedure

To create your own certificate, follow these steps:

1. Open the "Internet Information Services manager" application.
2. Click on the computer name.
3. Double-click "Server Certificates" in the "Features" view.
   The "Server Certificates" directory opens.
4. Click "Create Self-Signed Certificate" in the "Actions" area.
5. On the "Create Self-Signed Certificate" page, enter a display name for the certificate in the "Specify a friendly name for the certificate" field.
6. Click "OK".
   The new certificate is displayed in the "Certificate Server" overview.
7. Switch back to the "Internet Information Services Manager".
8. In the "Pages" directory, select the website in which the new HTTPS connection shall be created.
9. On the right side, click on "Edit Page > Binding".
   The dialog box with the available Web pages opens.
10. Select the connection type "https" and click "Add new".
    The dialog with the connection properties opens.
11. Select the protocol type "HTTPS" and the port number 443; insert the certificate you created in the "SSL Certificates" selection box.
12. Click "OK" to create a new connection.

Integrating IPCs into the TIA Portal

In addition to control and visualization tasks, PC systems can also be used for the following applications:

- Data gateway to higher-level MES
- Plant-level data server
- Hardware platform for SCADA systems
- Industrial workstations (for example, at test stations)
- Process & data analysis on site

Integrating SIMATIC IPCs in TIA Portal

Integrating the SIMATIC PC station (IPC in TIA Portal) integrates a PC system in a plant configuration (TIA project), with the following advantages:

- Fast configuration
  - Configuration via drag-and-drop from the hardware catalog
  - Detailed display of all interfaces – suitable for configuration of SIMATIC IPC
  - Network-spanning view including display of IP addresses
  - Fine-grained status diagnostic network topology
- Simplified commissioning
  - Central overview of all IP addresses
    Excludes conflicts due to duplicate assignments
    Easy to find SIMATIC IPC (including host name and IP address) via "Accessible devices"
  - Status diagnostics within the TIA Portal
    Diagnostics of the Ethernet/PROFINET connection to the SIMATIC IPC from the configuration
    Simple and fast error localization, for example, topology errors in the network, inaccessible devices
    Port-granular diagnostics (including the ports assigned in the topology on the SCALANCE switch)
  - Overview of the entire network
    Comprehensive information on topology, structures and hierarchies
- Integration of hardware diagnostics for IPCs in the TIA Portal
  - Fine-grained status diagnostics of the network topology, for example, topology errors in the network, inaccessible devices
  - Preventive maintenance, for example, of hard disks or fans
  - Early reaction to violation of high/low limit of the permitted operating temperature
  - Central overview of all SIMATIC IPCs in a plant
  - Display of hardware diagnostics in the WinCC Diagnostic Viewer
    Display of PC diagnostics in the running plant
    Preventive maintenance

System requirements

To install a PC station in an IPC, the following requirements must be met:

Requirements

- You have administrator rights on your computer.
- A minimum of 400 MB space is available.

Hardware requirements

The following IPCs are supported:

- SIMATIC IPC4x7D
- SIMATIC IPC2x7E
- SIMATIC IPC547G
- SIMATIC IPC6x7D
- SIMATIC IPC8x7D

Note

Make sure that the IPCs you use in your configuration are compatible.

If you ignore the error message about your configuration, diagnostics errors in TIA Portal can result.

Always select the appropriate IPC for your configuration.

Software requirements

- The SIMATIC IPC software can be used with the following operating systems:
  - Windows Embedded Standard 7 E/P SP1 32-bit
  - Windows Embedded Standard 7 E/P SP1 64-bit
  - Windows 7 SP1 32-bit
  - Windows 7 SP1 64-bit
- The following diagnostic software is installed:
  - DiagBase >= V1.5.0 or
  - DiagMonitor >= V4.5.0

Restrictions

- The following filters are disabled:
  - EWF
  - FBWF
- The following components may not be installed:
  - PC Station classic (S7-RTM)
  - CPU 150xS
  - SIMATIC NET
  - WinAC RTX

Installation

Storage media for the installation

Setup can be installed from the following media:

- USB stick
- DVD
- Hard disk
- Network drive

Installation

Setup is started by running "start.exe" in the Setup folder.

Proceed as follows for this:

1. Double-click on "start.exe".
2. Select the desired product properties for the installation.
3. Follow the instructions on the screen during the installation.

Result

After a successful installation, the following components have been installed:

- PC Station V2.x
- S7 device drivers
- SIMATIC NET CP Manager Plus
- PC Station Web server

In addition, an icon for the PC Station service is also displayed in the notification area of the Windows task bar during operation of the CPU.

Reference

You can find more information on the PC Station display in the section PC Station display in the notification area (Page 1127).

PC station display in the notification area

An icon for the PC Station service is also displayed in the notification area of the Windows task bar during operation of the CPU. Among other things the icon indicates the current state of the PC Station service, and provides you with the opportunity to do configurations.
Right clicking the icon in the notification area opens the PC Station shortcut menu.

States of the notification area icon

The state of the icon for the PC Station service in the notification area of the task bar changes as soon as the mode of the PC station changes.
The notification area icon can display the following states:

RUN	STOP
	

Configuration options using the "Station Manager" service's icon

The icon for the PC Station service in the task bar's notification area gives you the following configuration options via the shortcut menu:

- Import > Import Configuration
  This shortcut menu command opens and displays the metadata of a selected PC system configuration file. The user also starts the import of the psc file with this. The feature is not supported for F-CPUs.
- Configuration
  - Delete existing configuration
    Administrative rights are required for this configuration option.
    This shortcut menu command causes the configuration to be deleted.
    This function is disabled by default in an F-CPU. The "Failsafe Operators" user group must be added explicitly authorized in order to provide them with permission on an individual basis. Furthermore, the configured passwords for an F-CPU are retained after the configuration is deleted. These passwords are required later again for downloading. To reset the password, load a project without passwords.
  - Change the storage path of configuration data
    Administrative rights are required for this configuration option.
    If you are protecting a partition with an enhanced write filter (EWF), for example, the configuration and diagnostic data is also thereby write-protected.
    Save the diagnostic data in an area of the hard disk that is not write-protected. You can also save the configuration data containing the configuration in a section of the hard disk that is not write-protected.
- Restart all of the PC Station services
  Administrative rights are required for this configuration option.
  This shortcut menu command causes all of the PC Station services to be restarted.
- Exit
  This shortcut menu command causes the PC Station panel to be closed. The icon for the PC Station service in the notification area of the task bar is hidden. Restart the PC Station panel using the following entry in the Windows Start menu: **Siemens Automation > SIMATIC > PC Station > "PC Station"**

Repairing/uninstalling

You have the option to repair or uninstall the installed program.

Repairing

To repair the program, proceed as follows:

1. Navigate to the path where your Setup is installed.
2. Double-click on "start.exe".
3. Select "TIA integration for SIMATIC IPC" by placing a check mark and click "Next".
4. Select the "Repair" button.
5. Click "Next" and follow the instructions on the screen.

Uninstalling

To uninstall the program, you have two options.

Via the Control Panel

1. In the Windows taskbar, select **Start > Control Panel > Run Advertised Programs**
2. Select the program "TIA integration for SIMATIC IPC".
3. Click "Uninstall".
4. Follow the on-screen instructions during the uninstall process.

Via "start.exe":

1. Navigate to the path where your Setup is installed.
2. Double-click on "start.exe".
3. Select "TIA integration for SIMATIC IPC" by placing a check mark and click "Next".
4. Select the "Uninstall" button.
5. Click "Next" and follow the instructions on the screen.

Restrictions for cancellation

When the setup is aborted, all installation steps must be carried out again.

Basic configuration of the Open Controller

Creating the configuration

You have created a new project in the TIA Portal.

To create the configuration in the TIA Portal, follow these steps:

1. Double-click "Add new device" in the project tree.
2. Select "CPU 1515SP PC (+HMI)" under "PC Systems > Open Controller".
3. Select the appropriate version.
   The configured Open Controller is displayed in the device view.

4. The following components can be seen in the Open Controller:
   – Onboard interface X2 (GB Ethernet Windows interface) that is assigned directly to the PC station (1 port)
   – Exchangeable BusAdapter X1 that is assigned directly to the Software Controller (2 ports)
   – On the right side (behind the plugged modules in the rack): the configured CPU 1505S Software Controller and WinCC Runtime Advanced.
5. Insert the server module.
   The server module at the right end of the configuration forms the termination of the CPU with the I/O modules.

Setting the IP addresses:

In the Inspector window under "Properties":

- Onboard interface X2: The configured, preset IP address is identical to the Windows IP address that is set on the Open Controller during initial commissioning.
- BusAdapter X1: The configured, preset IP address is identical to the IP address that was set in the panel of the Software Controller (display application) during initial commissioning.

Important properties of the Software Controller

If necessary, you can change the properties in the Inspector window under "Properties".

- Selecting startup type
- System diagnostics
- Setting the storage location for retentive data
- Setting up copy protection
- Using the LEDs of the hardware
- Configuring the Web server
- Assigning interfaces for the communication

Establish HMI connection

1. Right-click the WinCC RT Advanced in the device view.
2. Start the HMI device wizard.
3. Apply all default settings.
   The wizard creates system images for you with the corresponding navigation.
4. Switch to the connection view in the network view.
5. Click on WinCC RT Advanced.
6. Draw a line from the WinCC RT Advanced software to the S7-1500 Software Controller (e.g. CPU 1505SP) while keeping the mouse button pressed.
   A network connection is established between the two devices.

Downloading the project to the target system

The complete PC station is downloaded the first time you download the configuration.

1. Select the complete PC system in the device view.
2. Click the "Download" button.
3. Make the following settings:
   - Type of connection
   - Specify the interface of the programming device
   - Specify the X2 interface on the Open Controller

Note

The first TIA Portal download **must** take place over the "X2" interface.

Only ASCII characters are permitted in the name of the interface X2 PN/IE (LAN) in the TIA Portal, e.g. PROFINET_2.

Download and compile the project. The hardware configuration and the first download are now completed.

WinCC RT Advanced

WinCC Runtime Advanced contains all the essential functions for operator control and monitoring of machinery or plants.

Reference

You can find additional information on configuration and downloading in the ET 200SP Open Controller CPU 1515SP PC (https://support.industry.siemens.com/cs/de/en/view/109248384) manual.

You can find more information on WinCC RT Advanced in the WinCC Advanced V13.0 SP1 (https://support.industry.siemens.com/cs/de/en/view/109091876) system manual.

Parameter assignment

Booting of PC (startup parameters)

Configuration of the startup type of the CPU

With a CPU 150xS, you have the option of assigning parameters for the startup characteristics in addition to the standard startup parameters, for example "Startup after POWER ON". The CPU can be started (POWER ON) in two different ways:

- Manual start via the "Power" button on the CPU display
- Automatic start during PC start (recommended)

Using the "Automatic start after booting the PC" option in the properties of the software controller, you determine how the CPU starts.

We recommend you enable the option "Automatic start after booting the PC", otherwise the software controller does not automatically boot after the PC system starts. A software controller can be only loaded after it has started.

If this option is disabled, you must start the CPU manually before a download procedure via the CPU display.

Note

BIOS memory test on SIMATIC IPCs

PCs provide the option of a memory test. Some hardware tests, such as the memory test, are disabled by default in the BIOS setup program and are skipped during startup of the PC. Booting is accelerated as a result.

If you are using the CPU on a SIMATIC IPC or CPU 1515SP PC, the BIOS memory test must not be enabled.

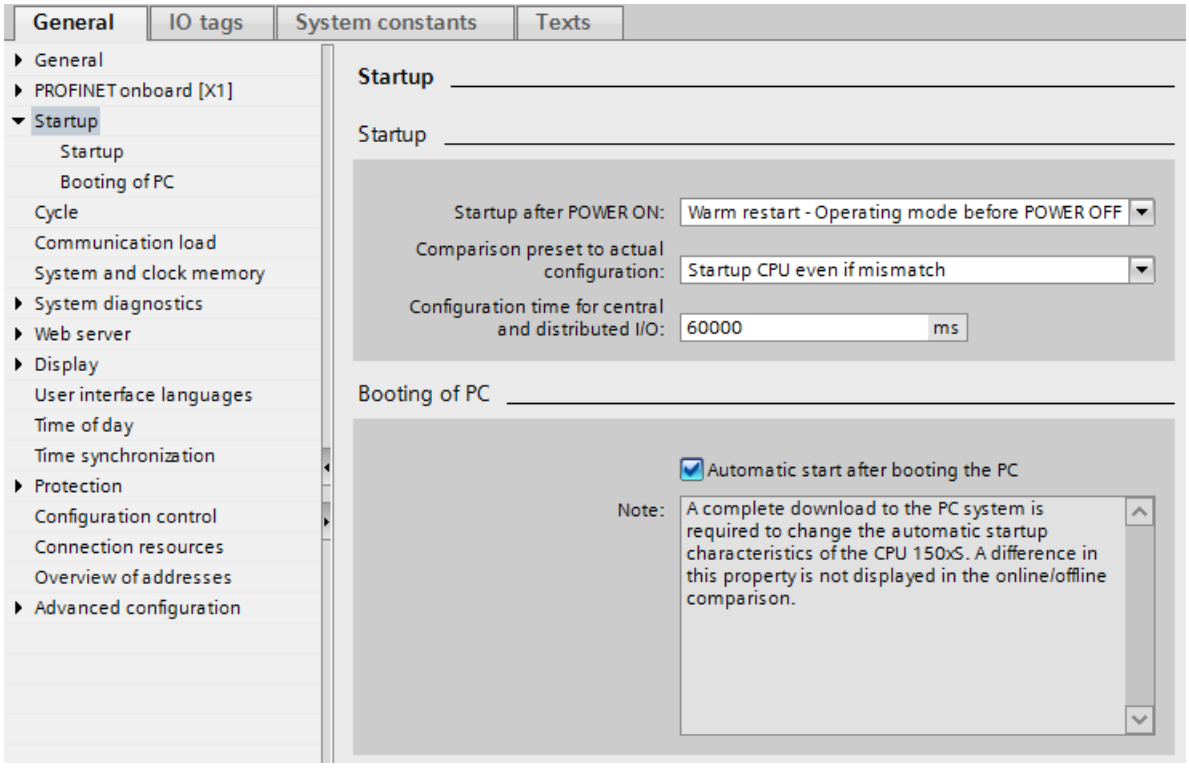
The default startup parameters are no different from the startup parameters of S7-1500 CPUs.

Setting the startup type

To set the startup type, follow these steps:

1. Start STEP 7.
2. Open your project.
3. Change to the project view.
4. Open the device view.
5. Select the CPU.
6. On the "Properties" > "General" tab of the Inspector window, select the "Startup" area.

7. Configure the startup characteristics of your CPU.



8. Download the project to the CPU, by selecting the PC system before you start the download.

Result

If, in addition to the settings for the startup type, you also change the "Automatic start after booting the PC" option, the CPU is stopped automatically before the download. At the beginning of the download, the CPU starts again in STOP mode.

The project is downloaded. The new settings for the startup type are active.

Special features in the parameter assignment of the access protection

Assigning access protection parameters in STEP 7

The access protection parameters are assigned using the properties of the PC system in which the software controller is inserted.

The access levels are read-only in the properties of the CPU. To access the properties of the PC system, click the arrow. Here you can define the access levels of the PC system and overwrite them for the entire application.

Note

Assigning access protection parameters for the entire PC system

Unlike for a hardware CPU, parameter assignment for access protection is not done directly in the CPU's properties. This ensures that consistent protection level passwords are configured for all of a PC system's components.

Unlike a hardware CPU of the S7-1500 automation system, a separate display password cannot protect the CPU against unauthorized access . Because the CPU can also be controlled by remote access, it uses the access protection passwords from STEP 7 to ensure access protection for the display.

Reference

For additional information on the protection concept in conjunction with display access, refer to theSIMATIC S7-1500 CPU 150xS (https://support.industry.siemens.com/cs/de/en/view/109249299) operating manual.

Advanced configuration for PC systems

If you select an S7-150xS software controller in the device view or network view, click on the "Advanced configuration" area in the Inspector window. The parameters under "Advanced configuration" are explained below or you can find references to explanations for these parameters.

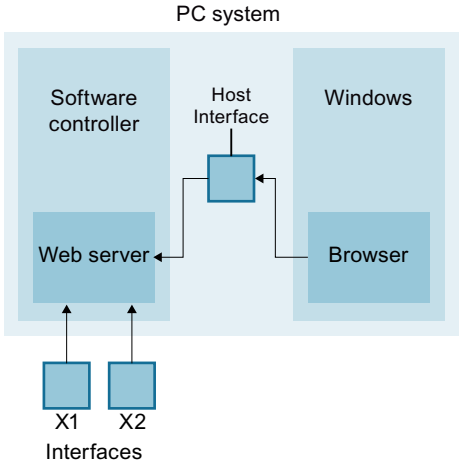
Host interface

The host interface offers access, for example, to an office network via a Windows-controlled interface independent of a PROFIBUS or PROFINET subnet. Your user program uses this interface to communicate with other devices or fetch information via the web server of the software controller:

Accessing the web server

Similar to the hardware interface, you also need to enable access via the host interface to be able to access the web server of the controller software through this interface.

The following figure shows the web server access from the web browser of the PC system via the host interface.



Communication via Open User Communication (OUC)

You can also use Ethernet interfaces on the PC via the host interface.
Basically, only the name of the host interface and the hardware ID is displayed for the configuration.
In the properties of a communication instruction (e.g. TSEND_C), you can access the properties of the host interface and use the "Connection data" area to access the address parameters for the programmed OUC communication. The hardware ID of the host interface is entered in the connection data.

Index

The index of a component in the PC system corresponds to the position of the component (for example, a hardware resource as an interface). It is a sort of "slot address on the PC-internal subnet" (Softbus).

The index of a software or hardware component is not displayed in the graphical view.

When a software controller is selected in the network view or device view, you can read or change the index in the properties of the software controller under "Advanced configuration". You can also change the index in the device overview table in the device view.

Reference

You can find information on using the hardware LEDs here (Page 1137).

You can find information about setting the storage location for retentive data here (Page 1136).

Setting the storage location for retentive data

The S7-1500 Software Controller (CPU) provides the option of storing data retentively in the PC mass storage or in the integrated NVRAM when the CPU is stopped or a power failure occurs. You set the type of data storage in the CPU properties.

Note

Data loss when changing the storage type

When you change the storage type, the current retentive data and the contents of the diagnostic buffer are deleted when the changed configuration is loaded.

Requirement

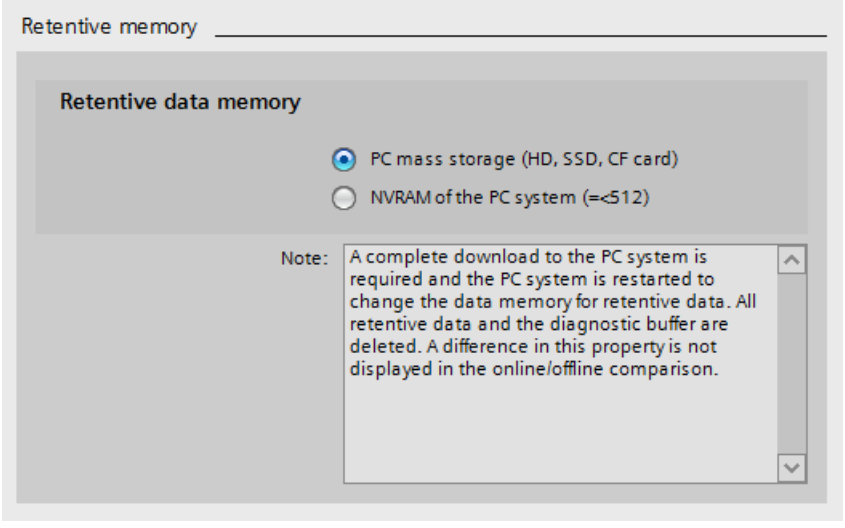
The project view is open.

The device view is open.

Procedure

To configure the type of storage, follow these steps:

1. Select the CPU.
2. Select the "Advanced configuration" area on the "Properties" tab of the Inspector window.
   - Select the "PC mass storage" option button to store the retentive data in the mass storage of your PC.
   - Select the "NVRAM of PC platform" option button to save the retentive data in the integrated NVRAM of your PC.



3. To change the type of data storage in STEP 7 at a later time, download the project to the PC system again.

Note

Downloading the change of configuration

When you change the storage type for storage of retentive data in STEP 7, you must download the entire PC system for this configuration change. When you change the storage type, the CPU reboots during the download process. Without a reboot the storage type is not changed despite the download process.

To ensure the required restart of the CPU, you have to load the PC and then restart it.

See also

Advanced configuration for PC systems (Page 1134)

Using the LEDs of the hardware

The S7-1500 Software Controller (CPU) can display its status on the LEDs of the hardware platform on which it is installed. You set this functionality in the CPU properties in STEP 7.

Note

Simultaneous access by multiple components

Take care that multiple competing components (for example DiagBase and CPU) do not make simultaneous access to the hardware LED.

Requirement

Your project is open.

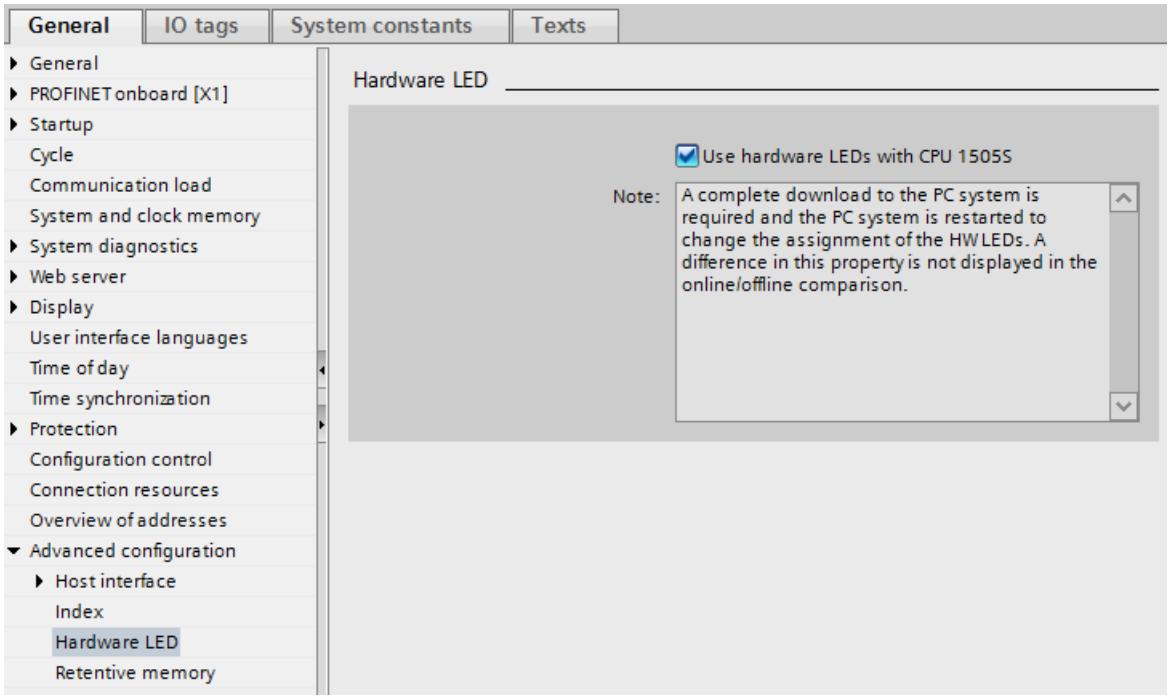
You are in the project view.

Procedure

To use the LEDs of the hardware platform for the CPU, follow these steps:

1. Open the network view or device view.
2. Select the CPU.

3. Select the "Advanced configuration > Hardware LED" area on the "Properties" tab of the Inspector window.



4. Select the "Use hardware LEDs with CPU 150xS" option.

   As described in the note below the option, you then have to fully download the PC system and reboot it.

Result

The PC is shut down and the CPU is stopped. The project is downloaded.

The PC is restarted automatically. The CPU starts in STOP mode. The download is continued. After completion of the download, the CPU is put into RUN mode.

Special feature

If there is an online/offline difference with regard to the "Use hardware LEDs" property, this difference is not shown in the online/offline comparison.

See also

Advanced configuration for PC systems (Page 1134)

Important information about assigning interfaces

As of STEP 7 V13 SP1, relationships between a software component, such as a Software Controller, and its interfaces to the outside world, e.g. a PROFINET interface, are represented by reference lines.

Representation of relationships

If nothing was selected, all interfaces and the software components are connected by a gray line. The lines have a square connector at the end. These lines show the relationship between software components (e.g. S7-1500 Software Controller, WinCC RTX or OPC server) and interfaces (PROFINET, PROFIBUS, ...).

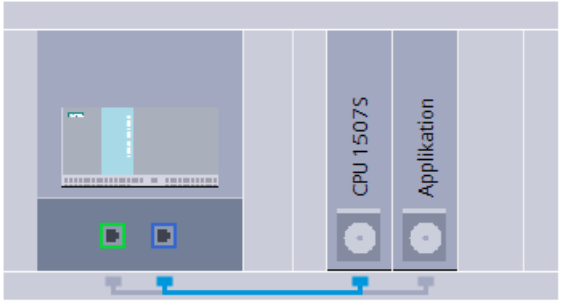
When you select an interface, the assigned software components can be recognized by the blue line that is clearly distinguishable from the gray lines.

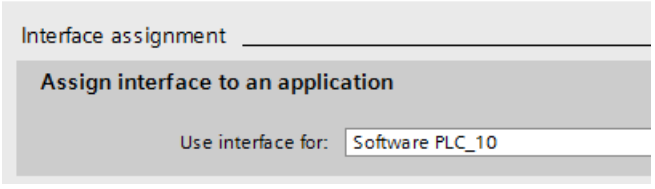
Interfaces without relation lines or with gray reference lines are not assigned.

Example 1: Assigning an interface to the Software Controller (CPU)

A blue reference line between a selected interface and an S7-1500 Software Controller means that the interface is assigned to the S7-1500 Software Controller and can only be used by it. In this case, the Software Controller, as IO controller, can operate a PROFINET IO system on this interface.

Only with this type of interface assignment is it possible to configure the IP address and the device name of the PROFINET interface and load it in the PC system.

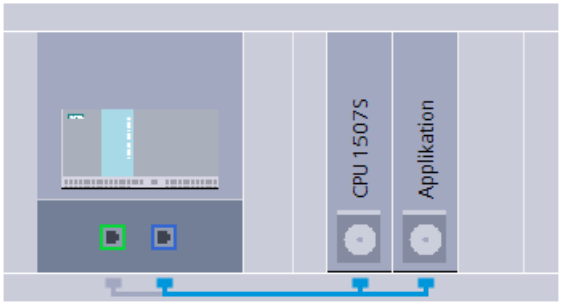


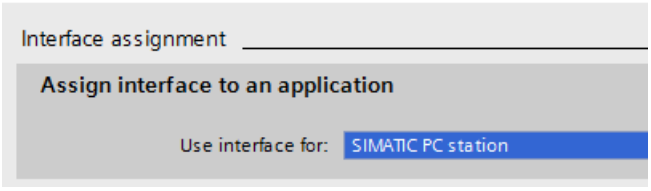


Example 2: Assigning the interface to the SIMATIC PC station

A blue reference line from a selected onboard interface to several software components means that this interface is assigned to the PC station and can therefore be used by all software components. Programming device communication, for example, can be performed through this interface.

Although you can edit the IP address of the interface with this type of interface assignment, this setting is not loaded in the PC system. The IP address must be set in Windows on the PC system. It still makes sense to synchronize IP addresses, for example, to be offered exactly this PC station in the "Extended download" dialog (filter function).

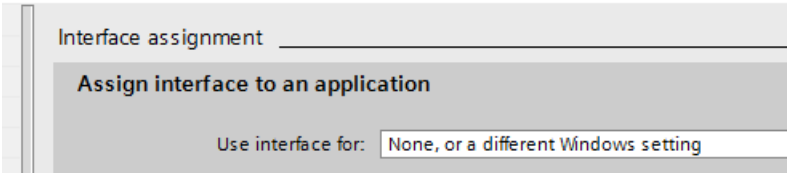




Example 3: None, or a different Windows setting

If you assign the "None, or a different Windows setting" option to an onboard interface, no relation to the inserted software components is possible and therefore also not visible. Only Windows applications can use this interface; however, they are not visible in the configuration. The IP address of the interface is not visible in the configuration as well.





Displaying the position of the interface

The position of the interface is displayed below the "Interface assignment" area in the Inspector window. Depending on the assignment to the PC station or to the Software Controller, the index or the hardware resource/interface type is displayed.

Assigning interfaces for communication of the PC Station V2.x

Interface types

The PC station uses the Ethernet interfaces on the PC for communication. The following communication services are available through these Ethernet interfaces:

- Engineering access to the PC station and the installed components, for example, the S7-1500 Software Controller or WinCC RT Advanced
- HMI connections to the PC station or to the Software Controller
- Data communication (S7 communication of the Software Controller)
- HMI connections, for example, between WinCC RT Advanced and external controllers
- S7 connections routed through the PC

Communication between devices

To enable configured communication between components on the PC and other SIMATIC devices, you need to assign the interface to the PC station. The following Ethernet interfaces can be assigned to PC Station V2.x:

- On-board Ethernet interfaces, for example, X1 and X2, of the SIMATIC IPC (not: on-board CP 1616)
- On-board X2 interface of CPU 1515SP PC

A maximum of 2 on-board interfaces can be assigned.

Engineering functions are possible with all Ethernet interfaces of the PC as long as they are operated under Windows, even if they are not assigned to the PC station.
However, for simplified commissioning, at least one interface must be assigned:

- SIMATIC PC: X1 interface
- CPU 1515SP PC: X2 interface
- No use of external CPs

Note

Interfaces for exclusive use can also be assigned to the S7-1500 Software Controller. These are then not available for the PC station.

Procedure

To assign the interfaces for the communication to the PC station, follow these steps:

1. Select the PC system.
2. Select the desired integrated interface in the device view.
3. In the Inspector window, select the area "General > Interface assignment".
4. Assign the interface to the "SIMATIC PC station".

5. Compile the project with "Edit > Compile".
6. Download the hardware configuration of the PC system to the target device.

Result

After downloading, the interfaces can be used for configured communication services.

Reference

You can find more information on the special aspects of configuring interfaces with the S7-1500 Software Controller in the SIMATIC S7-1500 CPU 150xS operating manual (https://support.industry.siemens.com/cs/de/en/view/109249299).

SIMATIC NET version

Information on the SIMATIC Net version is available at Service&Support under the Entry ID: 63098071 (https://support.industry.siemens.com/cs/document/63098071?lc=en-WW).

SIMATIC NET version for onboard interfaces

The SIMATIC NET version of the onboard interfaces can be found as follows:

1. Select the onboard interface.
2. In the Inspector window, select the area "General > Catalog information".
   - The displayed version is 2.2 for PC Station V2.2
   - The displayed version is 2.1 for a PC System V2.1.
   - The displayed version is 2.0 for a PC System V2.0.
   - For a PC System V1.0, the latest version of the SIMATIC NET CD from the hardware catalog is displayed, for example V8.2, see the description text. STEP 7 does not know the SIMATIC NET version actually used because you are installing and configuring PC System V1.0 with S7-RTM (Station Manager) from the SIMATIC NET CD.

SIMATIC NET version for PC Station

If you select a PC system in the network view and select the "PC Station" area in the Inspector window, the various PC system versions can be selected in the drop-down list.

- Select "PC System V1.0" option: **No** SIMATIC NET version displayed because STEP 7 does not know the SIMATIC NET version actually used, see above.
- Select the "PC-System V2.x" option: The SIMATIC NET version that is loaded and installed along with the PC system configuration is displayed.

Special features of the S7 Modular Embedded Controller

Special features of the S7 Modular Embedded Controller

Recommendation

For new projects we recommend switching to the new product generation with S7-1500 Software Controllers.

Reference

You can find additional information on the S7-modular Embedded Controller in the SIMATIC Embedded Automation S7 Modular Embedded Controller (https://support.industry.siemens.com/cs/de/en/view/37971572) operating instructions.

Introduction and basic procedure

Introduction

S7-modular Embedded Controllers (S7-mEC) are PCs in S7-300 design with preinstalled and preconfigured software:

- Windows Embedded Standard operating system
- WinAC RTX (F)
- SIMATIC NET
- WinCC Runtime (only with EC31-HMI/RTX)

Ethernet interfaces of the S7-modular Embedded Controller

An S7-modular Embedded Controller has two integrated Industrial Ethernet interfaces:

- The X1 PN LAN P1 / X1 PN LAN P2 interface can be configured as a PROFINET interface.
- The X2 P1 PN/IE LAN interface is assigned to the PC station in index 3 at IE General and is preconfigured for Industrial Ethernet communication.

Note

If accessing S7-mEC via Remote Desktop Protocol, ensure that S7-mEC and the Engineering PC/programming device are connected to the same Ethernet subnet.

To access S7-mEC via Remote Desktop protocol, you have to use the IE General network connection (usually X2 P1 PN/IE LAN interface).

Basic configuration and programming of the S7-mEC

Since an S7-mEC is a preconfigured system, follow these steps:

1. Create a project.
2. Insert a PC station.
3. Configure the hardware.
4. Create an S7 program.
5. Configure visualization in WinCC for the EC31-HMI/RTX.
6. Download the configuration data to the WinLC RTX EC.

EC31-HMI/RTX: Configuring visualization

For EC31-HMI/RTX only

Create the visualization project for the EC31-HMI/RTX on the engineering PC, using SIMATIC WinCC.

Requirement

- WinCC is installed on the engineering PC.
- A WinCC RT object had been inserted in the PC station.

Procedure

In order to create a visualization project for EC31-HMI/RTX, follow these steps:

1. Open the device view of the PC station of the EC31-HMI/RTX.
2. Select WinCC RT object.
3. Select the "Open object" command.
   WinCC is started.
4. Create and configure the necessary elements: Pictures, alarms, tags, etc., using the WinCC editors.
5. Save and test the visualization.
6. Transfer the project to the EC31-HMI/RTX target system.

Downloading the configuration data to the target system

Download options

You can download the following configuration data from STEP 7 to the embedded controller:

- The entire PC station (including, for example, IE General)
- Only the configuration data of the WinLC RTX / WinLC RTX F
- Only the S7 program of the WinLC RTX / WinLC RTX F

For an EC31-HMI/RTX, you will need to transfer the configuration data from WinCC.

Requirement

- The enhanced write filter (EWF) must be deactivated on the Embedded Controller if the entire PC station is to be downloaded.
- The embedded controller is connected to the programming device / PC via Industrial Ethernet. The PG/PC interface must be set to TCP/IP.
- **EC31-HMI/RTX only:** The visualization project has been created in WinCC.
- Before the following functions become available, the embedded controller must be connected to the engineering system via a port of the X1 PN LAN interface:
  - Assign PROFINET IO device names
  - PROFINET CBA: Interconnection download and online functions

Procedure

1. Deactivate EWF on the EC31-RTX / EC31-RTX F / EC31-HMI/RTX (if not already deactivated). After EWF has been deactivated, turn off and restart the embedded controller.
2. Start WinLC.
3. Activate the PC station in the Station Configuration Editor ("Enable station" button).
4. Change the embedded controller to STOP mode.
5. Download the hardware configuration, the S7 program and the configuration data to the target system.

Making ready for operation

To make the EC31-RTX / EC31-HMI/RTX ready for operation, follow these steps:

1. Set the enhanced write filter for drive C:\ to the "activated" state.

NOTICE
Flash memory failure If you leave the enhanced write filter in "deactivated" state, you risk an early failure of your flash memory due to the persistent write access of the operating system. When the enhanced write filter is activated, the number of write accesses to the operating system partition is minimized, thereby increasing the life of the flash memory.

2. Record all special settings (such as changes to the IP addresses) made in the course of application development. Store them along with your project.
3. Create a backup copy of your settings.

10.1.4.3 Configuring AS-i devices

Digital addresses

Bit assignment

The bit assignment is determined from the IO code (I/O configuration) of the configured AS-i slave. The following table shows the bit assignment.

IO code	Offset of the input bits				Offset of the output bits			
	0.0	0.1	0.2	0.3	0.0	0.1	0.2	0.3
0 (EEEE)	IN1	IN2	IN3	IN4	-	-	-	-
1 (EEE-)	IN1	IN2	IN3	-	-	-	-	-
2 (EEEE)	IN1	IN2	IN3	IN4	-	-	-	-
3 (EEA-)	IN1	IN2	-	-	-	-	OUT1	-
4 (EEBE)	IN1	IN2	IN3	IN4	-	-	OUT1	-
5 (EAAA)	IN1	-	-	-	-	OUT1	OUT2	OUT3
6 (BBBB)	IN1	IN2	IN3	IN4	OUT1	OUT2	OUT3	OUT4
7 (BBBB)	IN1	IN2	IN3	IN4	OUT1	OUT2	OUT3	OUT4
8 (AAAA)	-	-	-	-	OUT1	OUT2	OUT3	OUT4
9 (AAAE)	-	-	-	IN1	OUT1	OUT2	OUT3	-
A (AAAB)	-	-	-	IN1	OUT1	OUT2	OUT3	OUT4
B (AAEE)	-	-	IN1	IN2	OUT1	OUT2	-	-
C (AABB)	-	-	IN1	IN2	OUT1	OUT2	OUT3	OUT4
D (AEEE)	-	IN1	IN2	IN3	OUT1	-	-	-
E (ABBB)	-	IN1	IN2	IN3	OUT1	OUT2	OUT3	OUT4

IN1 = first input of the AS-i slave, IN2 = second input of the AS-i slave, etc.

OUT1 = first output of the AS-i slave, OUT2 = second output of the AS-i slave, etc.

Note

The labeling of the inputs and outputs on the AS-i slaves may differ from this illustration.

For A/B slaves (with ID code = A), output bit offset 0.3 is not occupied. Exception: An A/B slave with 4 inputs and 4 outputs (IO code = 7, ID code = A) occupies all output bits.

The actual address of the bit is calculated from the input and/or output address plus that of the offset given in this table.

Example

Input address 100.0 and output address 200.4 have been set. The IO code of the slave has been set to "3 (EEA-)". The AS-i slave data then lies in the process image on input and/or output bits E100.0, E100.1 and A200.6.

Network configuration of AS interface

AS-Interface consists of an AS-i master and AS-i slaves connected to each other over an AS-i subnet.

Rules for AS interface network configuration

All the nodes in an AS-i subnet must have a different AS-i node address.

You should only load the settings over the network when all the modules in a subnet have different addresses and when the actual structure matches that of the network configuration you have created.

One AS-i master and up to 62 AS-i slaves can be operated on an AS-i subnet.

For more information on configuring an AS-Interface with an AS-i master and AS-i slaves, refer to the section on AS-Interface and the documentation of the AS-i master modules.

I/O range

Input and output ranges

The CM AS-i Master ST module has an I/O range (process image for inputs and outputs) for transferring the AS-i slave input and output data to the CPU of the PLC.

The permissible length of the I/O range, as well as the assignment of I/O addresses to AS-i slaves, is dependent on the configuration used. The input and output ranges have the same length.

The length of data configured is always transferred, although data not used by slaves is filled with substitute values.

If a configured slave is not present (e.g. in the event of slave failure), the allocated input range is populated with substitute values.

In the event of a slave failure, no I/O area access error is generated because the entire I/O range is transferred, filled with either process values or substitute values.

In order to achieve the best possible performance, select the length of the I/O range based on the volume of data used by the slaves.

If the length is set to > 32 bytes, the module in the ET 200SP slot uses time division multiplexing to internally transfer data with a block size of 32 bytes.

Properties of the I/O range for different configurations

CM AS-i Master ST, FW version V1.0

Length of I/O range: 32 bytes

The digital inputs and outputs of the AS-i slaves are assigned to the addresses in the I/O range according to a defined format. See the table below for more details.

The analog inputs and outputs of the AS-i slaves can be transferred via data record communication.

The following table shows the assignment of the digital I/O address to the AS-i slave address:

Byte	Bit 7 … 4	Bit 3 … 0
n+0	Reserved	Slave 1 or 1A
n+1	Slave 2 or 2A	Slave 3 or 3A
n+2	Slave 4 or 4A	Slave 5 or 5A
n+3	Slave 6 or 6A	Slave 7 or 7A
n+4	Slave 8 or 8A	Slave 9 or 9A
n+5	Slave 10 or 10A	Slave 11 or 11A
n+6	Slave 12 or 12A	Slave 13 or 13A
n+7	Slave 14 or 14A	Slave 15 or 15A
n+8	Slave 16 or 16A	Slave 17 or 17A
n+9	Slave 18 or 18A	Slave 19 or 19A
n+10	Slave 20 or 20A	Slave 21 or 21A
n+11	Slave 22 or 22A	Slave 23 or 23A
n+12	Slave 24 or 24A	Slave 25 or 25A
n+13	Slave 26 or 26A	Slave 27 or 27A
n+14	Slave 28 or 28A	Slave 29 or 29A
n+15	Slave 30 or 30A	Slave 31 or 31A
n+16	Status data	Slave 1B
n+17	Slave 2B	Slave 3B
n+18	Slave 4B	Slave 5B
n+19	Slave 6B	Slave 7B
n+20	Slave 8B	Slave 9B
n+21	Slave 10B	Slave 11B
n+22	Slave 12B	Slave 13B
n+23	Slave 14B	Slave 15B
n+24	Slave 16B	Slave 17B
n+25	Slave 18B	Slave 19B

n+26	Slave 20B	Slave 21B
n+27	Slave 22B	Slave 23B
n+28	Slave 24B	Slave 25B
n+29	Slave 26B	Slave 27B
n+30	Slave 28B	Slave 29B
n+31	Slave 30B	Slave 31B

n = Start address

Status data only possible with CM AS-i Master ST from FW version V1.1 onwards, otherwise these 4 bits are not used (reserved).

CM AS-i Master ST, from FW version V1.1 with configuration via acceptance of ACTUAL configuration, e.g. via SET button

Note

No configuration of AS-i slaves in STEP 7 and/or configuration via GSD

Switchable I/O range length: 16 bytes or 32 bytes

The digital inputs and outputs of the AS-i slaves are assigned to the addresses in the I/O range according to a defined format. See above table.

The analog inputs and outputs of the AS-i slaves can be transferred via data record communication.

When the length is set to 16 bytes, the digital I/Os of the AS-i slaves with B addresses are not transferred.

When the length is set to 32 bytes, 4 bits of status data can be additionally transferred.

CM AS-i Master ST, from FW version V1.1 with configuration of AS-i slaves in STEP 7 (i.e. selection of AS-i slaves from hardware catalog)

Settable I/O range length: 4 ... 288 bytes

Possible settings: 4, 8, 12, 16, 20, 24, 28, 32, 64, 96, 128, 160, 192, 224, 256, 288 bytes

The digital and analog inputs and outputs of the AS-i slaves can be freely assigned within the I/O range.

The digital and analog inputs and outputs of the AS-i slaves can also be transferred via data record communication.

Further factors affecting the length of the I/O range:

Length of I/O range: 4 ... 256 bytes for configurations with IM 155-6 PN ST from FW V3.1 onwards
Length of I/O range: 4 ... 32 bytes for configurations with IM 155-6 PN ST < FW V3.1
Restriction when operating under CPU S7-300 FW version < V3 - length of I/O range: max. 224 bytes

Length of I/O range: 4 ... 32 bytes for configurations with IM 155-6 DP HF

I/O configuration for AS-i slaves

I/O configuration

The I/O configuration is assigned as follows on the basis of the I/O code standard:

Table 10-81 AS-i standard slave

I/O code	I/O configuration			
	D0	D1	D2	D3
0	I	I	I	I
1	I	I	I	O
2	I	I	I	B
3	I	I	O	O
4	I	I	B	B
5	I	O	O	O
6	B	B	B	B
7	B	B	B	B
8	O	O	O	O
9	O	O	O	I
A	O	O	O	B
B	O	O	I	I
C	O	O	B	B
D	O	I	I	I
E	O	B	B	B
F	F	F	F	F

Key: I=Input; O=Output; B=Bidirectional (input/output); F=TriState

Table 10-82 AS-i A/B slave

I/O code	I/O configuration			
	D0	D1	D2	D3
0	I	I	I	I
1	I	I	I	-
2	I	I	I	I
3	I	I	O	-
4	I	I	B	I
5	I	O	O	-
6	B	B	B	I
7	B	B	B	I
8	O	O	O	-
9	O	O	O	I
A	O	O	O	I
B	O	O	I	I

I/O code	I/O configuration			
	D0	D1	D2	D3
C	O	O	B	I
D	O	I	I	I
E	O	B	B	I
F	F	F	F	F

Key: I=Input; O=Output; B=Bidirectional (input/output); F=TriState

Operating principle of the safe AS-i outputs

Safety-related communication for safe AS-i outputs also uses code sequences. However, they have a different data structure compared to the code sequences of safe AS-i input slaves. The code sequences for safe AS-i outputs do not have to be taught and transferred.

For the function of a safe AS-i output, you need the following components:

- A control unit
- An evaluation unit

Control unit

The control unit outputs safety-related IN or OUT control commands to the AS-i bus. The control unit is integrated in the F-CM AS-i Safety ST module. The control unit assumes an active position in the communication. It acts like an AS-i slave and has an AS-i address that must be called by the AS-i master. The values in the output data of the AS-i master are not relevant in this case. The F-CM AS-i Safety ST module inserts the safety-related control data into the communication data.

Besides controlling the OFF state according to specification, a control unit can control up to four different output switching states: F-OUT 1, F-OUT 2, F-OUT 3, and F-OUT 4. The F-CM AS-i Safety ST module, however, makes available only the output switching state F-OUT 1, i.e., The OFF and ON states are controlled.

In addition to the switching states, a control unit can send out two different auxiliary signals: AUX1 and AUX2. You can use these auxiliary signals for error acknowledgement in the evaluation unit. For technical reasons, the AUX1 and AUX2 signals are mapped in the safe process image output. The auxiliary signals are not safety-related signals, however. During control by the safety program, the F-CM AS-i Safety ST module sends a sequence from each auxiliary signal one after the other on the AS-i bus. Up to 16 different control units can be activated in the F-CM AS-i Safety ST module. The control units are managed in the F-CM AS-i Safety ST module as switching group 0 to switching group 15.

If you want to use more than 16 different control units, you can plug two F-CM AS-i Safety ST modules into the ET 200SP station.

Observe the information on the bus configuration in Section AUTOHOTSPOT.

NOTICE
Uniqueness of the AS-i address on the AS-i bus Carefully check the AS-i addresses for the control of the safe AS-i outputs. Ensure that the assigned AS-i address exists only once on the AS-i bus. This is particularly important when you are using multiple devices that can control safe AS-i outputs on the same AS-i bus. If this is not observed, reliable shutdown of a safe output module is not guaranteed.

Evaluation unit

The evaluation unit reads the safety-related control commands ON or OFF on the AS-i bus and passes corresponding switching commands, e.g., to switching contacts. In case of error, e.g., communication error on the AS-i bus, the evaluation unit switches off. The evaluation unit is integrated in the safe AS-i output module to which, for example, drive contactor coils are connected. The evaluation unit behaves as a passive unit in the AS-i communication. The master does not detect the evaluation unit and does not call it. The evaluation unit monitors the communication of the AS-i address of the control unit and reacts to its safety-related control signals.

You set the AS-i address (see Section "Control unit") the evaluation unit is to monitor.

You can set multiple evaluation units to the same address so that a control unit can control multiple safe output modules simultaneously.

NOTICE
Bus cycle time for safe AS-i outputs An evaluation unit requires a minimum AS-i bus cycle time. Ensure that at least 5 AS-i addresses are present on the AS-i bus. Otherwise, the control unit goes to error state. When A/B slaves are used, at least 5 different numerical AS-i addresses must be present. That is, a combination of slave addresses 1A and 1B count as 1 numerical address. The F-CM AS-i Safety ST module does not have a monitoring function for this configuration rule.

Teaching ASIsafe code sequences

Operating principle of code sequences of safe AS-i slaves

Function of the code sequences

To monitor safety-related AS-i input slaves, they must be declared to the system. To do this, the F-CM AS-i Safety module requires information about the code sequences sent by the slaves. To prevent mix-ups of the AS-i input slaves on an AS-i bus, each slave must use a different code sequence. The code sequences are permanently stored in the input slave.

The recording, checking, and storing of these code sequences in the F-CM is referred to as "teaching code sequences”.

Once the F-CM detects a new code sequence, the teaching of the code sequences starts automatically. The user must confirm the transfer of the code sequences into the memory to prevent unauthorized changes to the AS-i components.

The F-CM module checks each code sequence for correctness of data content and then sets the safe input image depending on the evaluation.

Initial commissioning

During initial commissioning, the code sequences are taught at the F-CM from the safe AS-i input slaves that exist on the AS-i bus and then transferred into the non-volatile memory. Both the slaves for which input monitoring is activated as well as the other safe input slaves are detected.

Replacement of a safe input slave

After replacement of a safe input slave, the new code sequence must be taught and saved.

Transferring code sequences

In the non-volatile memory, you can transfer the code sequences in one of the following ways:

- Click the "Transfer taught code sequences" button
- Press the "Transfer taught code sequences" button
- Via the user program, see F-CM AS-i Safety ST Manual

Replacement of the F-CM module

The non-volatile memory is located in the BaseUnit, which means that code tables that are transferred after the F-CM module is replaced are retained.

Unsaved code sequences

For safe input slaves having code sequences that are not saved or do not correspond to the saved code sequence, the input channel is passivated and assigned the substitute value 0.

Safety-related communication for safe AS-i outputs

Safety-related communication for safe AS-i outputs also uses code sequences. However, they have a different data structure compared to the code sequences of safe input slaves. The code sequences for safe AS-i outputs do not have to be taught and transferred.

"Learn ASIsafe code sequences" dialog

Description of the “Learn ASIsafe code sequences” dialog

The dialog displays an overview of the existing code sequences of the safety-oriented AS-i slaves. For this purpose, code sequences read in at the AS-i bus can be transferred to the F-CM AS-i Safety ST module memory.

Code sequences can be read in by activating the dialog under "Functions" in "Online & Diagnostics" of the F-CM.

See also: Operating principle of code sequences of safe AS-i slaves (Page 1152)

The dialog can only be opened if there is an online connection to the controller or F-CM.

Before the teaching of the code sequences, load the hardware configuration into the controller modules so that the online configuration of the F-CM agrees with the offline configuration.

The "ASIsafe code sequences" dialog does not make any changes to the module configuration and can also be opened for information during operation.

The displayed information is obtained online from the F-CM module. There is no comparison with the offline configuration in this case.

The dialog displays the following information for each AS-i address within the "AS-i addresses 0…15" and "AS-i addresses 16…31":

Slave status (Page 1154)

Code sequence (Page 1155)

F-IN 1…2 (Page 1155)

F-OUT 1…4 (Page 1156)

Information on the slave status is displayed in the TIA Portal for the AS-i address 0.

Under this information, you will find the Transfer taught code sequences (Page 1157) button and the progress display for the teaching process (Page 1156).

You can exit the dialog at any time without having to make changes to the data. Code sequences can also be only partially transferred.

Description of the “Slave Status” column

Slave status

The "Slave status" column displays information about the AS-i address via a color code:

Gray: The AS-i slave is neither configured nor detected on the AS-i network.

Green (without checkmark): The address is present on the AS-i bus but is not activated in the configuration of the F-CM (input monitoring/output control deactivated). The slave on the AS-i bus is a safe slave or a standard slave. It is available for use by the AS-i master or another module.

Green with checkmark: The address is present on the AS-i bus and corresponds to the configuration of the F-CM (input monitoring or output control is activated).

Yellow: Error detected: The address is present on the AS-i bus but does not correspond to the configuration of the F-CM (input monitoring or output control is activated). Check the slave, replace the slave with the required slave or change the configuration.

Red: Error detected: The address is not present on the AS-i bus but is required in the configuration of the F-CM (input monitoring or output control is activated). Check whether the slave exists on the bus and was included in the communication by the AS-i master. Check whether the slave is contained in the configuration of the AS-i master and correct the bus configuration as required.

Description of the “Code sequence" column

The "Code sequence" column shows information about the AS-i address via a color code:

Gray: No code sequence is needed for the address

Green (without checkmark): A new code sequence was taught but not transferred to the memory. The transfer is possible via the "Transfer taught code sequences" button.

Green with checkmark: The code sequence was transferred to the non-volatile memory.

Note: The status is also displayed when the output control is activated for this address. The code sequences for safe outputs are automatically pre-defined in the F-CM and are not transferred into the non-volatile memory of the code sequences.

Yellow: Error detected: A code sequence was taught that is already being used by another slave. Safe operation is not possible. Replace the slave with a different slave.

Red: Warning or error: The address is sending an incomplete or invalid code sequence. Check whether the ON signal is assigned to both inputs of the safe input slave. If required, actuate the associated safety contacts (contacts must be closed). Check whether there is a cross circuit at the inputs and correct the wiring of the inputs as needed.

Description of the "F-IN 1...2" column

The states of the safety-related AS-i input channels are displayed in the F-IN1 and F-IN2 columns via a color code.

Meanings of the color codes for F-IN 1...2

Red: Error, e.g. code sequence error. The status of the input channel cannot be read in. The substitute value 0 is used.

Yellow: The input channel shows the value 0 (OFF state) or is not present.

Note

Condition monitoring

The status indication is unambiguous once the code sequence has been taught for a safe input slave (ON signal assigned to both inputs), or when an ON signal has been read in for externally controlled safe output slaves.

Green: Signal=1: Contact F-IN closed

White: Signal=0: Contact F-IN open

Gray: The input channel is not present. A safe input slave was not detected at the address.

Description of the "F-OUT 1...4" column

The states of the safety-related AS-i output channels are displayed in the columns F-OUT1 to F-OUT4 via a color code.

Meanings of the color codes for F-OUT 1...4

Red: Error, e.g. code sequence error. The status of the output channel cannot be read in.

Yellow: The output channel shows the value 0 (OFF state) or is not present.

Note

Condition monitoring

The status indication is unambiguous once the code sequence has been taught for a safe input slave (ON signal assigned to both inputs), or when an ON signal has been read in for externally controlled safe output slaves.

Green: Signal=1: Channel On (ON status)

White: Signal=0: Channel Off (OFF status)

Gray: The output channel is not present. A safe output slave was not detected at the address.

Display of teaching progress

Progress bar "Teaching in progress"

If at least one safe slave whose code sequence has not yet been taught is detected on the AS-i bus, an active progress bar is displayed. Check the displays in the "Code sequence" column and ensure that the safe slaves send their code sequences.

Trained slaves

The "Trained slaves" output field shows the number of code sequences that has already been applied in relation to the number of code sequences that can still be applied. The number ratio of the output field "Taught slaves" is also displayed as a percentage. If the display reads 100%, all of the code sequences of the existing safe AS-i slaves are transferred into the non-volatile memory.

"Transfer taught code sequences" dialog

Description of the "Transfer taught code sequences" dialog

After the "Transfer taught code sequences" button is actuated, a dialog opens with a prompt asking if the code sequences should be transferred. You must confirm that you are addressing the correct module. As a check, the green LED "ADDR" on the front panel of the F-CM module flashes.

- You can cancel the prompt using the "No" button without transferring the code sequences.
- If you click "Yes", the taught code sequences are transferred into the non-volatile memory. Next, the input data of the safe AS-i input slaves is evaluated, allowing reintegration in the safety program.

WARNING

Plant components can be switched on

Depending on the safety program, the transfer of code sequences may cause plant components to be switched on. Before the transfer, ensure that no personnel injuries or property damage can occur. After transferring, check the safety equipment for proper functioning.

The "Transfer taught code sequences" button is deactivated when all taught code sequences have already been transferred to the non-volatile memory.

10.1.4.4 S7-1200 CM/CP

Telecontrol modules

Telecontrol CPs and TIM

CP identification

Reference

CPs using the "TeleControl Basic" protocol.

"CP identification" parameter group

One project number for the entire STEP 7 project

If you change the project number for a CP in the "CP identification" parameter group, this parameter is changed for all CPs in the entire STEP 7 project using the "TeleControl Basic" protocol.

TCSB evaluates project numbers from 1 ... 2000.

Configuration of the station number

You need to configure a unique station number for each station in the STEP 7 project that uses a CP using the "TeleControl Basic" protocol.

TCSB evaluates station numbers from 1 ... 8000.

Access ID

The access ID displayed in the "CP identification" parameter group is made up of the hexadecimal values of the project number, station number and slot. The parameter of the type DWORD is allocated as follows:

- Bits 0 - 7: Slot
- Bits 8 to 20: Station number
- Bits 21 to 31: Project number

Parameter groups

Communication types

In this parameter group, you enable the communication types you want to use for the particular module.

To minimize the risk of unauthorized access to the station via Ethernet, you need to enable the communications services that the module will execute individually. You can leave all options disabled if you do not need them. However, at least one option should be enabled in a telecontrol CP.

"Communication types" parameter group

- **Activate online functions**
  In the module enables access to the CPU for the online functions (diagnostics, loading project data etc.). If the function is enabled, the engineering station can access the CPU via the module.
  If the option is disabled, you have no access to the CPU via the module with the online functions. Online diagnostics of the CPU with a direct connection to the interface of the CPU however remains possible.
- **Enabling S7 communication**
  Enables the functions of S7 communication with a SIMATIC S7 and S7 routing.
  If you configure S7 connections to the relevant station, and these run via the module, you will need to enable this option.
- **Enable telecontrol communication**
  In the module, enables the telecontrol communication. Depending on the module type, the following protocols can be used:
  - ST7
    Enables communication with a control system that supports the ST7 protocol.
  - TeleControl Basic
    Enables communication with the telecontrol server.
  - DNP3
    Enables communication with up to four DNP3 masters.
  - IEC 60870-5-104
    Enables communication with up to four IEC masters.

Note

Full telecontrol functionality only enabled Security functions

For the following functions, you need to enable the Security functions:

- Send messages (e-mails / SMS) via the telecontrol functionality
- Use of the "TeleControl Basic" (general) protocol
- Use of the DNP3 Security functions
- Use of certificates

Note

Loss of configuration data when changing the telecontrol protocol

If you change the protocol on a configured module, protocol-specific configuration data is lost, for example, the data point configuration, the partner configuration and the messages (e-mails).

Ethernet interface

Time-of-day synchronization

Time-of-day synchronization with S7-1200

Note

Recommendation: Time-of-day synchronization only by 1 module

Only have the time of day of the station from an external time source synchronized by a single module so that a consistent time of day is maintained within the station.

When the CPU takes the time from the CP, disable time-of-day synchronization of the CPU.

Module and synchronization method

When using an external time source, the S7-1200 station can obtain the current time of day both via the CPU as well as via a CP.

With the S7-1200 there is no forwarding of the time of day from the station to the subnet.

Time-of-day synchronization of the CPU

As the synchronization method, with the CPU only NTP can be selected.

For the options for setting the CPU time via a telecontrol CP see below.

Time-of-day synchronization of telecontrol CPs

The following synchronization methods are available:

- **NTP**
  NTP can only be used when the telecontrol communication is disabled. In this case, the telecontrol CP is used as an extended PROFINET interface.
- **Time from partner**
  When telecontrol communication is enabled, the CP automatically takes the time-of-day from the communications partner.
- **Time from the CPU**
  All CMs/CPs of the station are synchronized by the time of the CPU.

Note

Forwarding the time to the CPU

Depending on the firmware version of the modules involved, the time-of-day of the CP is forwarded to the CPU in different ways:

- Optional forwarding of the CP time to the CPU using a PLC tag
- Obligatory forwarding of the CP time to the CPU via the backplane bus

Forwarding the time from the CP to the CPU

The forwarding of the CP time to the CPU depends on the firmware version of the CP and the CPU. Note the following behaviour.

- **CP firmware ≤ V2.1.5**
  With this firmware version the CP can make the time-of-day available to the CPU as an option via a PLC tag. When this PLC tag is read cyclically by the CPU, the CPU adopts the CP time.
  For TLC tags, see parameter group "Communication with the CPU" of the CP.
- **CP firmware ≥ V2.1.7 and CPU firmware ≥ V4.2**
  If both modules in the station have one of the named firmware versions, the time of day of the CP is automatically forwarded to the CPU. Since the CPU automatically adopts the CP time, you no longer require the forwarding option using the PLC tag.
  If for the CPU the option "CPU synchronizes the modules of the device" is enabled in "PROFINET interface > Time synchronization", all smart modules of the station are synchronized with the CPU time.

Time-of-day synchronization with NTP

NTP mode

In NTP mode, the CP sends time-of-day queries at regular intervals to one or more NTP servers. From the responses of the servers, the module selects the most accurate time of day.

The advantage of this mode is that it allows the time to be synchronized across subnets.

NTP generally uses UTC (Universal Time Coordinated). This corresponds to GMT (Greenwich Mean Time).

NTP server and synchronization interval

The IP address of at least one NTP server must be configured.

The synchronization interval specifies the cycle of the time-of-day queries to the NTP server. Depending on the selected module, the cycle can be selected from a drop down list or configured within a range of values between 10 seconds and 1 day (86400 seconds).

"Accept time of day from non-synchronized NTP servers" option

If the option is supported by the selected module the following behavior applies:

- Enabled
  The module also accepts the time-of-day from non-synchronized NTP servers with stratum ≥ 15.
- Disabled
  If the module receives a time-of-day frame from a non-synchronized NTP server with stratum ≥ 15, the time of day is not set according to the frame.

Reconnection delay

Note

If the partner cannot be reached, connection establishment via the mobile wireless network can take several minutes. This may depend on the particular network and current network load.

Depending on your contract, costs may result from each connection establishment attempt.

"Reconnection delay" parameter

The reconnection delay for connections in telecontrol communication is the waiting time between repeated attempts to establish the connection by the CP when the telecontrol server is not reachable or the connection has aborted. This waiting time avoids continuous connection establishment attempts at short intervals if there are connection problems.

A basic value is configured for the waiting time before the next connection establishment attempt. Starting at the basic value, the current waiting time is doubled after every 3 unsuccessful retries up to a maximum value of 900 s. Range of values for the basic value: 10 .. 300 s

Example: The basic value 20 results in the following intervals (waiting times) between the attempts to re-establish a connection.

- three times 20 s
- three times 40 s
- three times 80 s
- etc. up to max. 900 s

Note

Special characteristics with CP 1242-7 (6GK7242-7KX30-0XE0) together with TELECONTROL SERVER BASIC (V2)

If a second telecontrol server or a second router of the telecontrol server is configured, the CP attempts to connect to the second partner at the 4th attempt. If the second partner is also unreachable, the 7th time the CP attempts to connect to the first partner again.

Acknowledgment with IEC protocol

Acknowledgment mechanisms with telecontrol protocol IEC 60870-5-104

Reference: Configuration of the Ethernet interface of the CP > Parameter group "Advanced options" > Transfer settings"

With each sent data frame, the CP sends a continuous send sequence number. The data frame remains initially stored in the send buffer of the CP.

When it receives the data frame, the master sends the send sequence number from this or (if several frames are received) the last frame as an acknowledgment to the CP. The CP saves

the send sequence number returned by the master as a receive sequence number and uses it as an acknowledgment.

Frames whose send sequence number is equal to or lower than the current receive sequence number are evaluated as having been successfully transferred and are deleted from the send buffer of the CP.

Recommendations of the specification:

- w should not be higher than 2/3 of k.
- Recommended value for k: 12
- Recommended value for w: 8

Parameter:

- **k: Difference between send sequence number N(S) and receive sequence number N(R)**
  Maximum number of unacknowledged data frames (I-APDUs) as maximum difference between send sequence number N(S) and receive sequence number N(R).
  When k is reached and t_1 has not yet elapsed, the CP does not send any frames until all the frames sent have been acknowledged by the master.
  When k is reached and t_1 has elapsed, the TCP connection is terminated.
- **w: Max. number of unacknowledged data frames**
  Maximum number of received data frames (I-APDUs), after which the oldest frame received from the master must be acknowledged.

Parameters of the TIM for classic WAN

Interface operating mode

The operating mode of the interface specifies the mode of the data transfer. A maximum of one of two serial interfaces of a module may be operated in DMA mode. Range of values:

- Interrupt (block)
  This operating mode applies to the transmit and receive mode. The default mode Interrupt (block) is suitable for all connections. 4 characters are transferred per block. Following this, there is an interrupt. The received characters are checked only after a complete message has arrived.
- DMA
  This operating mode applies to the transmit and receive mode. The DMA mode should be used for connections with a high baud rate or a lot of message traffic, however not for GSM networks. A maximum of one of the two interfaces of a TIM module may be operated in DMA mode.
- Interrupt (individual characters)
  This operating mode is used only in the receive direction. In the transmit direction, the block mode continues to the used. This interrupt mode is suitable for extremely bad lines. An interrupt is triggered per transmitted character and each character is analyzed immediately after it is received allowing extremely good diagnostics of transmission errors. This mode is more reliable than the block mode but is slower.

Transmission criterion

The transmission criterion controls connection establishment for the transmission of conditional unsolicited messages from stations and node stations.

This reduces the number of connection retries. Range of values:

- Standard conditions
  No connection will be established due to the existence of conditionally spontaneous messages . The conditional spontaneous messages are sent only when a connection is established for sending unconditional spontaneous messages or a buffer overflow is threatening or the connection is established by the other end.
- Degree of filling
  The TIM only sends the conditionally spontaneous messages when the configured fill level of the send buffer for conditionally spontaneous messages is exceeded.
  In the input box, enter the fill level of the send buffer as a percentage at which when exceeded the TIM establishes a connection.
- Time
  The TIM sends the conditionally spontaneous messages at a configured time of day.
- Time scheme
  The TIM sends the conditionally spontaneous messages cyclically at a configurable interval.

Polling monitoring time

The TIM monitors the message traffic via WAN and sends a message to its local CPU indicating that the master station is disrupted if it is not called after a certain time has elapsed. The sequence is as follows:

- Polling monitoring time configured
  After the configured time elapses, the TIM sends a message to its local CPU.
- Polling monitoring time not configured
  If no monitoring time is configured, the TIM outputs a disruption message after the following internally calculated times.
  - At a transmission speed of 9,600 bps: After 4 seconds without message traffic
  - At a transmission speed of 1,200 bps: After 32 seconds without message traffic

Partner

Addressing a double / redundant TCSB system

Addressing of the duplicate or redundant telecontrol server

- **Addressing the main and substitute telecontrol server in TCSB V2**
  Only for CP 1242-7 (6GK7 242-7KX30-0XE0)
  With TCSB V2, two independent server PCs can be installed. The second IP address only needs to be configured when these are connected to the Internet via two routers.
- **Addressing the TCSB redundancy group in TCSB V3**
  In the LAN in the master to which the TCSB server PCs and the DSL router (e.g. SCALANCE M) are connected, the Network Load Balancing (NLB) of the computer operating system will assign a common virtual IP address to the two server PCs.
  This IP address is configured depending on the network setup:
  - If only CP 1243-1 modules without a DSL router are connected, the virtual address assigned by the NLB must be configured in the CPs as the IP address of the telecontrol server.
  - If a DSL router is used, only one IP address will be configured to address the redundant telecontrol server in the stations, the public address of the DSL router.
    Set the port forwarding on the DSL router so that the public IP address (external network) is led to the virtual IP address of the TCSB server PCs (internal network). Only the public IP address is reachable from the Internet. The station does not therefore receive any information telling it which of the two computers of the redundancy group it is connected to.

  If you configure a second IP address, you need to make sure that TCSB is reachable via the IP address of a second router.

Communication with the CPU

CPU scan cycle

Input data points are assigned to the read cycle of the CPU in the data point configuration in the "General > Read cycle" tab.

Structure of the CPU scan cycle

The cycle with which the transferring module (CP / TIM) scans the memory area of the CPU is made up of the following phases:

- **High-priority read jobs**
  (Fast cycle)
  For all data points with the assignment "Fast cycle" the PLC tags are read in every scan cycle.
  Normally it is adequate to assign only data to be acquired quickly such as alarms and contact changeover messages and command, setpoint and parameter objects for the 1oon check to the fast cycle.
- **Write jobs**
  In every cycle, the values of a certain number of unsolicited write jobs are written to the CPU.
  - Telecontrol CP / TIM 1500
    The number of tags written per cycle is specified for the transferring module in the "Communication with the CPU" parameter group with the "Max. number of write jobs" parameter. The tags whose number exceeds this value are then written in the next or one of the following cycles.
- **Low-priority read jobs - proportion**
  (Normal cycle)
  For data points with the assignment "Normal cycle", a proportion of the values of their PLC tags is read in every scan cycle.
  - Telecontrol CP / TIM 1500
    The number of tags read per cycle is specified for the transferring module in the "Communication with the CPU" parameter group with the "Max. number of read jobs" parameter. The tags that exceed this value and can therefore not be read in one cycle are then read in the next or one of the following cycles.
- **Cycle idle time**
  This waiting time between two scan cycles is used to reserve adequate time for other processes that access the CPU.

Duration of the CPU scan cycle

Since no fixed time can be configured for the cycle and since the individual phases cannot be assigned a fixed number of objects, the duration of the scan cycle is variable and can change dynamically.

Partner status and Path status of the TIM

You will find the parameters described below in the following parameter group of the TIM:

"Communication with the CPU" > "Partner status"

Partner

Here, you select the communications partner whose status and connection to the currently selected TIM will be monitored. The partner can be a CPU or an ST7cc.

In the table, the partner name can be selected from the possible partners.

Partner status

Here you select a PLC tag of the data type bool (DB, memory bit, output) to which the partner status will be written. Range of values:

- 0: Partner not reachable
- 1: Partner reachable

Path status

Here you select a PLC tag of the data type byte (DB, memory bit, output) to which the path status to a partner status will be written.

The PLC tag shows the status of the path to the partner from the perspective of the local TIM.

A maximum of 2 paths (main and substitute path) to a partner can be configured. Both paths must begin or end on a local TIM.

The PLC tag shows the following:

- The paths via which the partner can be reached.
- The path currently being used
- The TIM interface via which the main path was configured.
- The TIM interface via which the substitute path was configured.

The path of a connection is specified as a combination of the used interfaces of the TIM and the status of the path.

The path of a connection is specified as a combination of the used interfaces of the TIM and the status of the path.

- **Assignment of the byte of the PLC tag**
  The byte of the PLC tag is assigned as follows:
  - Two bits for the interface of the main path
  - Two bits for the interface of the substitute path
  - Two bits for the status of the main path
  - Two bits for the status of the substitute path

Bit 6 + 7	Bit 4 + 5	Bit 2 + 3	Bit 0 + 1
Configured interface		Path status	
No. for substitute path	No. for main path	Substitute path (2nd path)	Main path (1st path)

- **Configured interface**
  The TIM interfaces "Ethernet 1" (IE1), "Ethernet 2" (IE2), WAN1 and WAN2 are numbered through from 0 .. 3 (decimal):
  - 0 = "Ethernet 1" (IE1)
  - 1 = "Ethernet 2" (IE2)
  - 2 = WAN1
  - 3 = WAN2

Status of bit 5 (7)	Status of bit 4 (6)	Meaning
0	0	Coding for interface IE1 (decimal): No. 0)
0	1	Coding for interface IE2 (decimal: no. 1)
1	0	Coding for interface WAN1 (decimal: no. 2)
1	1	Coding for interface WAN2 (decimal: no. 3)

- **Path status**
  - Main path = 1. Path (bits 0 + 1)
  - Substitute path = 2nd path (bits 2 + 3)

Status of bit 1 (3)	Status of bit 0 (2)	Meaning bit 1	Meaning bit 0
0	0	Bit 1: Path not current	Bit 0: Subscriber not reachable
0	1	Bit 1: Path not current	Bit 0: Subscriber reachable
1	0	Bit 1: Path current	Bit 0: Subscriber not reachable
1	1	Bit 1: Path current	Bit 0: Subscriber reachable

Example of coding options

Same coding of the configured interface for the main and the substitute path means that there is no path redundancy (only 1 interface configured). The path status is output via the bits of the main path (1st path).

Table 10-83 Example of the coding for the PLC tag "Path status"

Configured interface		Path status	
No. for substitute path	No. for main path	Substitute path (2nd path)	Main path (1st path)
0 0	0 0 = coding for IE1	Irrelevant (not redundant)	Status IE1
0 0	0 1 = coding for IE2	Status IE1	Status IE2
0 0	1 0 = coding for WAN1	Status IE1	Status WAN1
0 0	1 1 = coding for WAN2	Status IE1	Status WAN2
0 1	0 0	Status IE2	Status IE1
0 1	0 1	Irrelevant (not redundant)	Status IE2
0 1	1 0	Status IE2	Status WAN1
0 1	1 1	Status IE2	Status WAN2

1 0	0 0	Status WAN1	Status IE1
1 0	0 1	Status WAN1	Status IE2
1 0	1 0	Irrelevant (not redundant)	Status WAN1
1 0	1 1	Status WAN1	Status WAN2
1 1	0 0	Status WAN2	Status IE1
1 1	0 1	Status WAN2	Status IE2
1 1	1 0	Status WAN2	Status WAN1
1 1	1 1	Irrelevant (not redundant)	Status WAN2

Operator input monitoring

Operator input monitoring

The Operator input monitoring is used to monitor operator input (buttons etc.) to protect against incorrect input.

For monitored operator input a minimum and maximum duration is specified. The monitoring status is saved in the "PLC tag for operator input monitoring" (byte).

The PLC tag can be used for entries via the following data point types:

- Command output (Cmd01B_S)
- Setpoint output (Set01W_S)
- Parameter entries (Par12D_S)

The Operator input monitoring is recommended particularly when commands are entered via digital inputs, for example using buttons connected to them. This also applies to the situation when setpoint and parameter entries are transmitted as the result of a trigger and this triggering is via a digital input, in other words for example once again via a button.

Monitoring incorrect input (input time)

Using Operator input monitoring reduces the risk of incorrect input when the entries are made via digital inputs. The following times and the code bits are relevant only for operator input over digital inputs.

- Min. input time
  For these entries, a minimum input time can be specified, in other words, the button must be pressed for the minimum time. The accidental activation of a button does not then lead to an unwanted transfer. When the minimum input time has elapsed and the button can be released, the "PLC tag for operator input monitoring" shows "entry OK" in bit 0 (see below).
  A time of at least 1 second is recommended.
  If you do not require the parameter, configure 0 (zero).
- Max. input time
  Apart from the minimum duration, a maximum input time can also be set for digital inputs. This allows a button that is sticking or defective digital inputs that supply a permanent 1 signal to be detected in good time. Such errors are written as "input error" to bit 1 of the PLC tag.
  Further hardware entries are not processed as long as the "input error" bit is set.
  If you do not require the parameter, configure 0 (zero).

Monitoring for 1 out of n errors.

Generally for all operator input via digital inputs. bit memory or data blocks the PLC tag returns the error status "1 out of n error" in bit 2. The error status is set when more than 1 bit is set at the same time.

The error is output in the following situations:

- 1-out-of-n error with "Cmd01B_S"
  More than 1 bit was set in the input byte of the command data point type "Cmd01B_S". To increase reliability of command input, only 1 bit may ever be set with this object. If two or more bits are set at the same time, the command input is rejected.
- 1-out-of-n errors with all command, setpoint and parameter entries in the fast read cycle
  If increased reliability is required for the input of commands, setpoints and parameters, all objects with which this data is sent should be assigned to the fast read cycle. All command, setpoint and parameter objects in the fast cycle are then subjected to a 1-out-of-n check. At the end of the fast cycle there is a check to make sure that there is a command, setpoint or parameter entry for only one of the acquired objects. Only then is the corresponding entry processed and transferred. If there is more than one entry, the entries are rejected. A new command, setpoint or parameter is processed only when previously no entry was acquired in at least one fast cycle.

Note

Resetting the command bit

If commands are entered over a memory or data byte, or a setpoint or parameter entry is enabled by a memory or data bit (trigger signal), the set command bit or trigger signal is automatically reset to zero.

If, however, a 1-out-of-n error is detected, the command bits are not automatically reset. They must then be reset by the user program.

Assignment of the status byte of "PLC tag for operator input monitoring"

Table 10-84 Bit assignment of the status byte

Bit:	.7 *	.6 *	.5 *	.4 *	.3 *	.2	.1	.0
Status:	-	-	-	-	-	1-out-of-n error	Input error	Input OK
For value:	0	0	0	0	0	1	1	1
* Unused bits are set to 0								

Data point editor

Configuring data points and messages

Data point-related communication with the CPU

No program blocks need to be programmed for the telecontrol modules with data point configuration to transfer user data between the station and communications partner. The data areas in the memory of the CPU intended for communication with the communications partner are configured data point-related on the module. Each data point is linked to a PLC tag or a or the tags of a data block.

Requirement: Created PLC tags

PLC tags or DBs must first be created in the CPU program to allow configuration of the data points.

The PLC tags for data point configuration can be created in the standard tag table or in a user-defined tag table. All PLC tags intended to be used for data point configuration must have the attribute "Visible in HMI".

Address areas of the PLC tags are input, output or bit memory areas on the CPU.

The formats and S7 data types of the PLC tags that are compatible with the protocol-specific data point types of the CP can be found in the section Datapoint types (Page 1177).

Access to the memory areas of the CPU

The values of the PLC tags or DBs referenced by the data points are read and transferred to the communications partner by the CP.

Data received from the communications partner is written by the CP to the CPU via the PLC tags or DBs.

Configuring the data points and messages in STEP 7

You configure the data points in STEP 7 in the data point and message editor. You can find this using the project tree:

Project > directory of the relevant station > Local modules > CP 1200 / TIM 3vxx/4Rxx

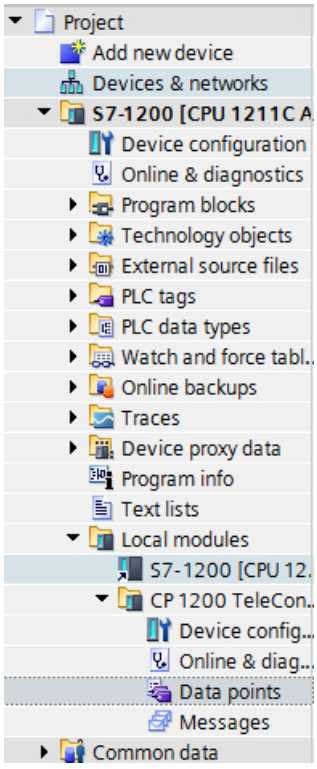


Figure 10-8 Configuring data points and messages

By double-clicking on the entry, the data point or message editor.

Using the two entries to the right above the table, you can switch over between the data point and message editor.

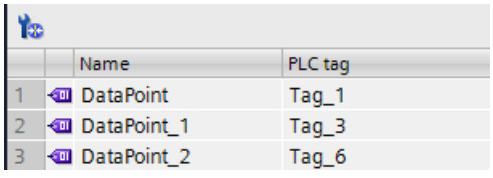


Figure 10-9 Switching over between the two editors

Creating obects

With the data point or message editor open, create a new object (data point / message) by double clicking "<Add object>" in the first table row with the grayed out entry.

A preset name is written in the cell. You can change the name to suit your purposes but it must be unique within the CP.



	Name	PLC tag
1	DataPoint	Tag_1
2	DataPoint_1	Tag_3
3	DataPoint_2	Tag_6

Figure 10-10 Data point table

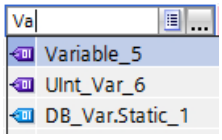
You configure the remaining properties of every object using the drop-down lists of the other table columns and using the parameter boxes shown at the bottom of the screen.

Assigning data points to their data source

After creating it, you assign a new data point to its data source. Depending on the data type of the data point a PLC tag can serve as the data source.

For the assignment you have the following options:

- Click on the table symbol in the cell of the "PLC tag" column.
  All configured PLC tags and the tags of the created data blocks are displayed. Select the required data source with the mouse or keyboard.
- Click the symbol ....
  A selection list of the configured PLC Tags and the blocks is displayed. From the relevant table, select the required data source.
- In the name box of the PLC tag, enter part of the name of the required data source.
  All configured PLC tags and tags of the data blocks whose names contain the letters you have entered are displayed.



Select the required data source.

Note

Assignment of parameter values to PLC tags

The mechanisms described here also apply when you need to assign the value of a parameter to a PLC tag. The input boxes fro the PLC tag (e.g.: PLC tag for partner status support the functions described here for selecting the PLC tag.

Arranging and copying objects

As with many other programs in the data point or message editor you can also arrange the columns, sort the table according to your requirements and copy and insert objects.

- Arrange columns
  If you click on a column header with the left mouse button pressed, you can move the column.
- Sorting objects
  If you click briefly with the left mouse button on a column header, you can sort the objects of the table in ascending or descending order according to the entries in this column. The sorting is indicated by an arrow in the column header.
  After sorting in descending order of a column the sorting can be turned off by clicking on the column header again.

- Adapting the column width
  You call this function using the shortcut menu that opens when you click on a column header with the right mouse key.
- Showing / hiding columns
  You call this function using the shortcut menu that opens when you click on a column header with the right mouse key.
- Copying, pasting, cutting and deleting objects
  If you click in a parameter box of an object in the table with the right mouse key, you can use the functions named with the shortcut menu (copy, paste, cut, delete).
  You can paste cut or copied objects within the table or in the first free row below the table.

Exporting and importing data points

To simplify the engineering of larger plants, you can export the data points of a configured module and import them into other modules in the project. This is an advantage particularly in projects with many identical or similar stations or modules.

The export / import function is available when you select the module for example in the network or device view and select the relevant shortcut menu.

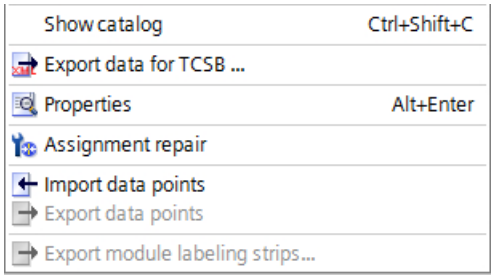


Figure 10-11 Shortcut menu of the CP

When it is exported the data point information of a CP is stored in one CSV file.

Export

When you call the export function, the export dialog opens. Here, you select the CPs or modules of the project whose data point information needs to be exported. When necessary, you can export the data points of all modules of the project at one time.

In the export dialog, you can select the storage location in the file directory. When you export the data of a module you can also change the preset file name.

When you export from several modules, the files are formed with preset names made up of the station name and module name.

The file itself contains the following information in addition to the data point information:

- Module name
- Module type
- CPU name
- CPU type

Editing the data point information

You can edit the data point information in an exported CSV file. This allows you to use this file as a configuration template for many other stations.

If you have a project with many stations of the same type, you can copy the CSV file with the data points of a fully configured module for other as yet unconfigured stations and adapt individual parameters to the particular station. This saves you having to configure the data points for every module in STEP 7. Instead, you simply import the copied and adapted CSV file to the other modules of the same type. When you import this file into another module, the changed parameter values of the CSV file are adopted in the data point configuration of this module.

The lines of the CSV file have the following content:

- Line 1: ,Name,Type,
  This line must not be changed.
- Line 2: PLC,<CPU name>, <CPU type>,
  Meaning: PLC (designation of the station class), CPU name, CPU type
  Only the elements <CPU name> and <CPU type> may be changed.
  The CPU type must correspond exactly to the name of the CPU in the catalog.
- Line 3: Module,<module name>,<module type>,
  Meaning: Module (Designation of the module class), module type, module name
  Only the elements <module name> and <module type> may be changed.
  Be careful when changing the module names if you want to import data points into several modules (see below).
  The module type must correspond exactly to the name of the module in the catalog.
- Line 4: Parameter names (English) of the data points
  This line must not be changed.
- Lines 5..n: Values of the parameters according to line 4 of the individual data points
  You can change the parameter values for the particular station.

Importing into a module

Before importing the data points make sure that the PLC tags required for the data points have been created.

Note that when you import a CSV file all the data points existing on the module will be deleted and replaced by the imported data points.

Select a CP and select the import function from the shortcut menu of the module. The import dialog opens in which you select the required CSV file in the file directory.

If the information on the assignment of the individual data points to the relevant PLC tags matches the assignment in the original module, the data points will be assigned to the corresponding PLC tags.

When you import data points into a CP, but some required PLC tags have not yet been created in the CPU, the corresponding data point information cannot be assigned. In this case, you can subsequently create missing PLC tags and them assign them the imported data point information. The "Assignment repair" function is available for this (see below).

If the names of the PLC tags in the module into which the import is made have different names than in the module that exported, the corresponding data points cannot be assigned to your PLC tags.

Importing into several modules

You can import the data points from several modules into the modules of a different project. To do this in the import dialog select all the required CSV files with the control key.

Before importing the data points, make sure that the respective stations have been created with CPUs of the same name, modules of the same name and PLC tags of the same name.

When you import the corresponding stations of the project are searched for based on the module names in the CSV files. If a target station does not exist in the project or the module has a different name, the import of the particular CSV file will be ignored.

Restrictions for the import of data points

In the following situations the import of data points will be aborted:

- The telecontrol protocol used by the modules differs from the original module.
  The following CPs with the "TeleControl Basic" protocol are compatible with each other:
  - CP 1242-7 GPRS V2
  - CP 1243-1
  - CP 1243-7 LTE

  This means, for example, that data points can be imported from a CP 1243-1 into a CP 1243-7 LTE or from a CP 1242-7 GPRS V2 to a CP 1243-1.
- An attribute required by the module is missing in the CSV file to be imported.
  Example: If a data point to be imported uses a time trigger, the import will be aborted if no time-of-day synchronization was configured for the module.

Only when importing into several modules:

- The import is aborted when a module or CPU name is different from the data in the CSV file.

Assignment repair

If you have named the PLC tags in a station into which you want to import differently from the station from which the CSV file was exported, the assignment between data point and PLC tag is lost when you import.

You then have the option to either rename the existing PLC tags appropriately or add missing PLC tags. You can then repair the assignment between unassigned data points and PLC tags. This function is available either via the shortcut menu of the CP (see above) or with the following icon to the upper left in the data point editor: 

If a PLC tag with a matching name is found for a data point by the repair function, the assignment is restored. However the data type of the tag is not checked.

After the assignment repair make sure that you check whether the newly assigned PLC tags are correct.

Datapoint types

During the configuration of the user data to be transferred by the module, each data point is assigned a protocol-specific data point type. The data point types supported by the module along with the compatible S7 data types are listed below. They are grouped according to format (memory requirements).

The direction relates to the direction of transfer (monitoring direction = "in", control direction = "out").

Note

Effect of the change of arrays for data points

If an array is modified later, the data point must be recreated.

CP 1243-1, CP 1242-7 GPRS V2, CP 1243-7 LTE: Supported data point types of the protocol "TeleControl Basic"

Table 10-85 Supported data point types and compatible S7 data types

Format (memory requirements)	Data point type	S7 data types	Operand area
Bit	Digital input	Bool	I, Q, M, DB
	Digital output	Bool	Q, M, DB
Byte	Digital input	Byte, Char, USInt	I, Q, M, DB
	Digital output	Byte, Char, USInt	Q, M, DB
Integer with sign (16 bits)	Analog input	Int	I, Q, M, DB
	Analog output	Int	Q, M, DB
Counter (16 bits)	Counter input	Word, UInt	I, Q, M, DB
Integer with sign (32 bits)	Analog input	DInt	Q, M, DB
	Analog output	DInt	Q, M, DB
Counter (32 bits)	Counter input	UDInt, DWord	I, Q, M, DB
Floating-point number with sign (32 bits)	Analog input	Real	Q, M, DB
	Analog output	Real	Q, M, DB
Floating-point number with sign (64 bits)	Analog input	LReal	Q, M, DB
	Analog output	LReal	Q, M, DB
Data block (1 .. 64 bytes)	Data	ARRAY [1)]	DB
	Data	ARRAY [1)]	DB

[1)] For the possible formats of the ARRAY data type, refer to the following section.

For the data types of the DNP3 and IEC protocols of the CP 1243-1 see below.

Data block (ARRAY)

With the ARRAY data type, contiguous memory areas up to a size of 64 bytes can be transferred. The following S7 data types are compatible components of ARRAY:

- Byte, USInt (total of up to 64 per data block)
- Char (total of up to 64 per data block) - CP as of firmware version 2.1.77

- Int, UInt, Word (total of up to 32 per data block)
- DInt, UDInt, DWord (total of up to 16 per data block)

If the array is modified later, the data point must be recreated.

Time stamp in UTC format

Time stamps are transferred in UTC format (48 bits) and contain the time difference in milliseconds since 01.01.1970.

ST7-TIM, CP 1243-8 IRC (ST7): Supported data point types

Table 10-86 Supported data point types and compatible S7 data types

Format (memory requirements)	Data point type	S7 data types STEP 7 Basic	Operand area	Number of data points per SINAUT object	SINAUT object
Bit	Digital input (Status Input)	Bool	I, Q, M, DB	8	Bin08X_S
	Digital output (Status Output)	Bool	Q, M, DB	8	Bin08X_R
Byte	Digital input (Binary Input)	Byte, USInt	I, Q, M, DB	4	Bin04B_S
	Digital output (Binary Output)	Byte, USInt	Q, M, DB	4	Bin04B_R
	Command output (Command Output)	Byte, USInt	Q, M, DB	1	Cmd01B_R
Integer with sign (16 bits)	Analog input (Analog Input)	Int	I, Q, M, DB	4	Ana04W_S
	Mean value (Mean Value Input)	Int	I, Q, M, DB	4	Mean04W_S
	Analog output (Analog Output)	Int	Q, M, DB	4	Ana04W_R
	Mean value (Mean Value Output)	Int	Q, M, DB	4	Mean04W_R
	Setpoint (Setpoint Output) [1]	Int	Q, M, DB	1	Set01W_R
Counter (16 bits)	Counter input (Counter Input)	UInt, Word	I, Q, M, DB	1	Cnt01D_S
	Counter input (Counter Input)	UInt, Word	I, Q, M, DB	4	Cnt04D_S
Floating-point number (32 bits)	Analog input (Analog Input)	Real	I, Q, M, DB	4	Ana04R_S
	Analog output (Analog Output)	Real	Q, M, DB	4	Ana04R_R
Data block (4 .. 48 bytes)	Data (Data Input)	ARRAY [1...12] of DInt / UDInt / DWord / Real [2]	DB	12	Dat12D_S
	Data (Data Output)		DB	12	Dat12D_R
	Parameters (Parameter Output) [1]		DB	12	Par12D_R

[1] See below, section "1: Mirroring back"

[2] See below, section "2: Data block"

1: Mirroring

The mirroring back function using the "Value monitoring" parameter can be configured for the following data point types:

- Setpoint Output
- Parameter Output

The local values of the data points of this type can be monitored for change and the changes transferred to the master with the Value monitoring function.

Changing a local value can, for example, be caused by manual operator input on site.

To allow the value resulting from local events or interventions to be transferred to the master, the Value monitoring function generates a mirroring back channel for the relevant data point via which the locally changed value is mirrored back.

Remember that to use the mirror back function, you need to interconnect the local values in the controller with the relevant PLC tag of the data point.

2: Data block

With the ARRAY data type, blocks of data from contiguous memory areas up to a size of 4 .. 48 bytes can be transferred.

Compatible components of ARRAY are DInt, UDint, DWord or Real. The components within an array must be of the same type.

CP 1243-1 (DNP3), CP 1243-8 IRC (DNP3), DNP3 TIM: Supported data point types

Table 10-87 Supported data point types, DNP3 object groups, variants and compatible S7 data types

Format (memory requirements)	Data point type CP [Data point type TIM]	DNP3 object group [variations]	Direction	S7 data types		Operand area
				CP	TIM	
Bit	Binary Input [Bin08X_S]	**1** [1, 2]	in	Bool	Bool	I, Q, M, DB
	Binary Input Event [Bin08X_S]	**2** [1, 2]	in	Bool	Bool	I, Q, M, DB
	Binary Output [1] [Bin08X_R]	**10** [2]	out	Bool	Bool	Q, M, DB
	Binary Output Event [1] [Bin08X_R]	**11** [1, 2]	out	Bool	Bool	Q, M, DB
	Binary Command [Cmd01B_S], [Cmd01B_R]	**12** [1]	out in / out	Bool	Byte	Q, M, DB

Format (memory requirements)	Data point type CP [Data point type TIM]	DNP3 object group [variations]	Direction	S7 data types		Operand area
				CP	TIM	
Integer (16 bits)	Counter Static [Cnt01D_S], [Cnt04D_S], [Set01W_S] [Set01W_R]	**20** [2]	in in out	UInt, Word	Word	I, Q, M, DB I, Q, M, DB Q, M, DB
	Frozen Counter 2) [Cnt01D_S], [Cnt04D_S]	**21** [2, 6]	in	UInt, Word	Word	I, Q, M, DB
	Counter Event [Cnt01D_S], [Cnt04D_S]	**22** [2, 6]	in	UInt, Word	Word	I, Q, M, DB
	Frozen Counter Event 3) [Cnt01D_S], [Cnt04D_S]	**23** [2, 6]	in	UInt, Word	Word	I, Q, M, DB
	Analog Input [Ana04W_S], [Set01W_S], [Mean04W_S]	**30** [2]	in	Int	Int	I, Q, M, DB
	Analog Input Event [Ana04W_S], [Set01W_S], [Mean04W_S]	**32** [2]	in	Int	Int	I, Q, M, DB
	Analog Output Status 4) [Ana04W_R], [Set01W_R], [Mean04W_R]	**40** [2]	out	Int	Int	Q, M, DB
	Analog Output [Ana04W_R], [Set01W_R], [Mean04W_R]	**41** [2]	out	Int	Int	Q, M, DB
	Analog Output Event 4) [Ana04W_R], [Set01W_R], [Mean04W_R]	**42** [2, 4]	out	Int	Int	Q, M, DB

Format (memory requirements)	Data point type CP [Data point type TIM]	DNP3 object group [variations]	Direction	S7 data types		Operand area
				CP	TIM	
Integer (32 bits)	Counter Static [Cnt01D_R], [Cnt04D_R], [Par12D_R]	**20** [1]	in out	DWord	 DWord	I, Q, M, DB Q, M, DB
	Frozen Counter [2] [Cnt01D_R], [Cnt04D_R], [Par12D_R]	**21** [1, 5]	in out	DWord	 DWord	I, Q, M, DB Q, M, DB
	Counter Event [Cnt01D_R], [Cnt04D_R], [Par12D_R]	**22** [1, 5]	in out	DWord	 DWord	I, Q, M, DB Q, M, DB
	Frozen Counter Event [3] [Cnt01D_R], [Cnt04D_R], [Par12D_R]	**23** [1, 5]	in out	DWord	 DWord	I, Q, M, DB Q, M, DB
	Analog Input	**30** [1]	in	DInt	-	Q, M, DB
	Analog Input Event	**32** [1]	in	DInt	-	Q, M, DB
	Analog Output Status [4] [Par12xD_R]	**40** [1, 3]	out	DInt	 DWord	Q, M, DB
	Analog Output [Par12xD_R]	**41** [1]	out	DInt	 DWord	Q, M, DB
	Analog Output Event [4] [Par12xD_R]	**42** [1]	out	DInt	 DWord	Q, M, DB
Floating-point number (32 bits)	Analog Input	**30** [5]	in	Real	-	Q, M, DB
	Analog Input Event	**32** [5, 7]	in	Real	-	Q, M, DB
	Analog Output Status [4]	**40** [3]	out	Real	-	Q, M, DB
	Analog Output	**41** [3]	out	Real	-	Q, M, DB
	Analog Output Event [4]	**42** [5, 7]	out	Real	-	Q, M, DB
Floating-point number (64 bits)	Analog Input	**30** [6]	in	LReal	-	Q, M, DB
	Analog Input Event	**32** [6, 8]	in	LReal	-	Q, M, DB
	Analog Output	**41** [4]	out	LReal	-	Q, M, DB
	Analog Output Event [4]	**42** [6, 8]	out	LReal	-	Q, M, DB
Data block (1...64 bytes) [5]	Octet String / Octet String Output	**110** [-]	in, out	[5]	[5]	DB
	Octet String Event [5]	**111** [-]	in, out	[5]	[5]	DB

[1] This object group can be configured in the Data point editor of STEP 7 using the substitute object group 12.

[2] This object group can be configured in the Data point editor of STEP 7 using the substitute object group 20.

[3] This object group can be configured in the Data point editor of STEP 7 using the substitute object group 22.

[4] This object group can be configured in the Data point editor of STEP 7 using the substitute object group 41.

[5] With these data point types, contiguous memory areas up to a size of 64 bytes can be transferred. All S7 data types with a size between 1 and 64 bytes are compatible.

Explanation of the table footnotes [1], [2], [3], [4]: Configuring data points using substitute object groups

The initial data point types of the following object groups can be configured using the substitute object groups listed above:

- 10 [2]
- 11 [1, 2]
- 21 [1, 2, 5, 6]
- 23 [1, 2, 5, 6]
- 40 [1, 2, 3]
- 42 [1, 2, 4, 5, 6, 7, 8]

To configure the DNP3 CP, use the specified substitute object group.

Assign each data point on the master using the configurable data point index in STEP 7. The data point of the DNP3 CP is then assigned to the corresponding data point on the master.

Example of configuring the data point Binary Output (10 [2])
The data point is configured as follows:
On the DNP3 CP as Binary Command (12 [1])
On the master as Binary Output (10 [2])

With the data point types Binary Output Event (11) and Analog Output Event (42), you also need to enable mirroring; refer to the next section.

Configuration of the mirroring for output events (object groups 11 and 42)

You first create the data point types Binary Output Event (object group 11) and Analog Output Event (object group 42) as described above as data points of the object groups 12 or 41.

The local values of these two object groups can be monitored for change and the changes transferred to the master (). Changing a local value can, for example, be caused by manual operator input on site.

To allow the value resulting from local events or interventions to be transferred to the master, the data point in question requires a channel for mirroring back. You configure this mirroring back function is configured using the "Value monitoring" option in data point configuration, General tab.

Remember that to use the mirror back function, you need to interconnect the local values in the controller with the relevant PLC tag of the data point.

Time stamp for DNP3-CP in UTC format

Time stamps are transferred in UTC format (48 bits) and contain milliseconds since 01.01.1970.

CP 1243-1 (IEC), CP 1243-8 IRC (IEC): Supported data point types

Table 10-88 Supported data point types, IEC types and compatible S7 data types

Format (memory requirements)	Data point type	IEC type	Direction	S7 data types	Operand area
Bit	Single point information	<1>	in	Bool	I, Q, M, DB
	Single point information with time tag CP56Time2a [1)]	<30>	in	Bool	I, Q, M, DB
	Single command	<45>	out	Bool	Q, M, DB
	Single command with time tag CP56Time2a [1)]	<58>	out	Bool	Q, M, DB
	Double command with time tag CP56Time2a [1)]	<59>	out	Bool	Q, M, DB
Byte	Step position information	<5>	in	Byte, USInt	I, Q, M, DB
	Step position information with time tag CP56Time2a [1)]	<32>	in	Byte, USInt	I, Q, M, DB
	Regulating step command with time tag CP56Time2a [1)]	<60>	out	Byte, USInt	Q, M, DB
Integer (16 bits)	Measured value, normalized value	<9>	in	Int	I, Q, M, DB
	Measured value, normalized value with time tag CP56Time2a [1)]	<34>	in	Int	I, Q, M, DB
	Measured value, scaled value	<11>	in	Int	I, Q, M, DB
	Measured value, scaled value with time tag CP56Time2a [1)]	<35>	in	Int	I, Q, M, DB
	Set point command, normalised value	<48>	out	Int	Q, M, DB
	Set point command, scaled value	<49>	out	Int	Q, M, DB
	Set point command, normalised value with time tag CP56Time2a [1)]	<61>	out	Int	Q, M, DB
	Set point command, scaled value with time tag CP56Time2a [1)]	<62>	out	Int	Q, M, DB
Integer (32 bits)	Bitstring of 32 bits	<7>	in	UDInt, DWord	I, Q, M, DB
	Bitstring of 32 bits with time tag CP56Time2a [1)]	<33>	in	UDInt, DWord	I, Q, M, DB
	Integrated totals	<15>	in	UDInt, DWord	I, Q, M, DB
	Integrated totals with time tag CP56Time2a [1)]	<37>	in	UDInt, DWord	I, Q, M, DB
	Bitstring of 32 bits	<51>	out	UDInt, DWord	Q, M, DB
	Bitstring of 32 bits with time tag CP56Time2a [1)]	<64>	out	UDInt, DWord	Q, M, DB
Floating-point number (32 bits)	Measured value, short floating point number	<13>	in	Real	Q, M, DB
	Measured value, short floating point number with time tag CP56Time2a [1)]	<36>	in	Real	Q, M, DB
	Set point command, short floating point number	<50>	out	Real	Q, M, DB
	Set point command, short floating point with time tag CP56Time2a [1)]	<63>	out	Real	Q, M, DB

Format (memory requirements)	Data point type	IEC type	Direction	S7 data types	Operand area
Data block (1...2 Bit) [2]	Double-point information	<3>	in	[2]	DB
	Double-point information with time tag CP56Time2a [1]	<31>	in	[2]	DB
	Double command	<46>	out	[2]	DB
	Regulating step command	<47>	out	[2]	DB
	Double command with time tag CP56Time2a [1]	<59>	out	[2]	DB
	Regulating step command with time tag CP56Time2a [1]	<60>	out	[2]	DB
Data block (1...32 Bit) [3]	Bitstring of 32 bits [3]	<7>	in	[3]	DB
	Bitstring of 32 bits with time tag CP56Time2a [1] [3]	<33>	in	[3]	DB
	Bitstring of 32 bits [3]	<51>	out	[3]	DB
	Bitstring of 32 bits with time tag CP56Time2a [1] [3]	<64>	out	[3]	DB

[1] For the format of the time stamp, see the following section.

[2] For these data point types, create a data block with an array of precisely 2 bool.

[3] With these data point types, contiguous memory areas up to a size of 32 bits can be transferred. Only the S7 Bool data type is compatible.

Time stamp with the IEC CP

IEC CP time stamps are transferred according to the IEC specification in the "CP56Time2a" format. Note that in the frames only the first 3 bytes for milliseconds and minutes are transferred.

CP 1243-1 PCC: Supported data point types

Table 10-89 Supported data point types and compatible S7 data types

Format (memory requirements)	Data point type	S7 data types	Operand area
Bit	Digital input	Bool	I, Q, M, DB
Byte	Digital input	Byte, USInt	I, Q, M, DB
Integer with sign (16 bits)	Analog input	Int	I, Q, M, DB
Counter (16 bits)	Counter input	Word, UInt	I, Q, M, DB
Integer with sign (32 bits)	Analog value	DInt	Q, M, DB
Counter (32 bits)	Counter	UDInt, DWord	I, Q, M, DB
Floating-point number with sign (32 bits)	Analog value	Real	Q, M, DB
Floating-point number with sign (64 bits)	Analog value	LReal	Q, M, DB
Data block (1 .. 64 bytes)	Data	ARRAY [1]	DB
	Data	ARRAY [1]	DB

[1] For the possible formats of the ARRAY data type, refer to the following section.

Data block (ARRAY)

With the ARRAY data type, contiguous memory areas up to a size of 64 bytes can be transferred.

The following equivalent S7 data types are compatible components of ARRAY:

- Bool (total of up to 512 per data block)
- Byte, USInt (total of up to 64 per data block)
- Int, UInt, Word (total of up to 32 per data block)
- DInt, UDInt, Real, DWord (total of up to 16 per data block)
- LReal (total of up to 8 per data block)

Note

No processing of arrays in the operation center

The array data type cannot be processed in the operation center.

Time stamp in UTC format

Time stamps are transferred in UTC format (48 bits) and contain the time difference in milliseconds since 01.01.1970.

Process image, type of transmission, event classes, triggers

Introduction

Some of the functions described below differ depending on the protocol used by the module in question:

- TeleControl Basic CPs:
  - CP 1242-7 GPRS V2
  - CP 1243-1
  - CP 1243-7 LTE
- CP 1243-8 IRC / TIM 3V-IE / TIM 4R-IE
- CP 1243-1 DNP3 / TIM 3V-IE DNP3 / TIM 4R-IE DNP3
- CP 1243-1 IEC

Storage of values

As a rulethe values of all data points are stored in the image memory of the module. Values in the image memory are transferred only after being called by master station (ST7 TIM, TeleControl Basic CPs) or the master (DNP 3 TIM, CP 1243-1 DNP3 / IEC).

Events are also stored in the send buffer and can be transferred unsolicited.

The image memory, the process image of the module

The image memory is the process image of the CP. All the current values of the configured data points are stored in the image memory. New values of a data point overwrite the last stored value in the image memory.

The values are sent only after a query by the communications partner - see below "Transfer after call" in the "Types of transmission" section - or along with a message from the send buffer that needs to be transferred immediately.

The send buffer

The send buffer of the CP is the memory for the individual values of data points that are configured as an event. You will find the size of the send buffer in the manual of the relevant module.

The capacity of the send buffer is divided up equally for all enabled partners.

If the connection to a communications partner is interrupted, the individual values of the events are retained in the buffer. When the connection returns, the buffered values are sent. The frame memory operates chronologically; in other words, the oldest frames are sent first (FIFO principle).

If a frame was transferred to the communications partner, the transferred value is deleted from the send buffer.

If frames cannot be transferred for a longer period of time and the send buffer is threatening to overflow, the response is as follows:

- Telecontrol Basic-CPs, ST7 TIM, ST7 CP (CP 1243-8 IRC)
  The forced image mode
  If the send buffer reaches a fill level of 80%, the CP changes to the forced image mode. New values from data points configured as an event are no longer added to the send buffer but rather they overwrite older existing values in the image memory.
  When the connection to the communications partner returns, the CP changes back to the send buffer mode if the fill level of the send buffer has fallen below 50%.
- CP 1243-1 DNP3 / CP 1243-1 IEC
  If a send buffer fill level of 100% is reached, the oldest values are overwritten.

Saving the data point values

As a rule, the values of data points are stored in the image memory of the moodule and transferred only when queried by the communications partner.

Events are also stored in the send buffer and can be transferred unsolicited.

Data points are configured as a static value or as an event using the "Type of transmission" parameter (see below):

- **Static value (no event)**
  Static values are entered in the image memory (process image of the CP).
  Static values correspond to the following classes:
  - DNP3: Class 0
  - IEC: Class 2
- **Event**
  The values of data points configured as an event are also entered in the image memory of the CP. The value of the event is sent unsolicited to the communications partner if this function is enabled by the master.
  The values of events are also entered in the send buffer of the CP.
  Events correspond to the following classes:
  - DNP3: Class 1 / 2 / 3
  - IEC: Class 1

Types of transmission

The following types of transmission are possible:

- **Transfer after call (class 0)**
  The current value of the data point is entered in the image memory of the CP. New values of a data point overwrite the last stored value in the image memory.
  After being called by the communications partner, the current value at this time is transferred.
- **Triggered (events)**
  The values of data points configured as an event are entered in the image memory and also in the send buffer of the CP.
  The values of events are saved in the following situations:
  - The configured trigger conditions are fulfilled (data point configuration > "Trigger" tab, see below)
  - The value of a status bit of the status identifiers of the data point changes; see sectionStatus IDs of the data points (Page 1189).
    Example:
    When the value of a data point configured as an event is updated during startup of the station by reading the CPU data for the first time, the status "RESTART" of this data point changes (bit status change 1 → 0). This leads to generation of an event.

  The configurable event classes of the various CP types are described in the following sections.

Event classes with the "Triggered" type of transmission

Depending on the protocol used, the following event classes are available:

- **TeleControl Basic / ST7**
  - **Every value triggered**
    Each value change is entered in the send buffer in chronological order.
  - **Current value triggered**
    Only the last, current value is entered in the send buffer. It overwrites the value stored there previously.
- **DNP3**
  The evaluation of the following classification must be handled by the master.
  - **Event class 1**
    Class according to DNP3 protocol: Class 1
    Each value change is entered in the send buffer in chronological order.
  - **Event class 2**
    Class according to DNP3 protocol: Class 2
    Each value change is entered in the send buffer in chronological order.
  - **Event class 3**
    Class according to DNP3 protocol: Class 3
    Only the current value at the time the trigger condition was met is entered in the send buffer and overwrites the last value stored there.
- **IEC**
  Both of the following event classes correspond to the user data class 1 of the IEC protocol
  - **Every value triggered**
    Each value change is entered in the send buffer in chronological order.
  - **Current value triggered**
    Only the current value at the time the trigger condition was met is entered in the send buffer and overwrites the last value stored there.

Trigger

Various trigger types are available for starting event-driven transfer:

- **Threshold value trigger**
  The value of the data point is transferred when this reaches a certain threshold. The threshold is calculated as the difference compared with the last stored value, refer to the section Threshold value trigger (Page 1196).
- **Time trigger**
  The value of the data point is transferred at configurable intervals or at a specific time of day.
- **Event trigger**
  The value of the data point is transferred when a configurable trigger signal is fired. For the trigger signal, the edge change (0 → 1) of a trigger tag is evaluated that is set by the user program. When necessary, a separate trigger tag can be configured for each data point.
  Resetting the trigger tag in the bit memory area / DB:
  If the memory area of the trigger tag is in the bit memory or in a data block, the trigger tag is reset to zero when the data point value is transferred.

Whether the value of a data point is transferred to the communications partner immediately after the trigger or after a delay depends on the protocol used and the settings.

- DNP3 / IEC
  The spontaneous transmission in these protocols depends on whether spontaneous sending or asymmetric communication is possible in the network.
- TeleControl Basic
  Here you set the time of transmission with the "Transmission mode" parameter; see below.

Transmission mode

Only when using the "TeleControl Basic" or "ST7" protocol.

The transmission mode of a frame is set in the "Trigger" tab of the data point. With the two options, you specify whether frames of events are sent immediately or following a delay:

- Spontaneous
  The value is transferred immediately.
- Conditional spontaneous
  The value is transferred only when one of the following conditions is fulfilled:
  - The communications partner queries the station.
  - The value of another event with the transmission mode "Unsolicited" is transferred.
  - The fill level of the transmission buffer has reached 80% of its maximum capacity.

Status IDs of the data points

Status identifiers

The status identifiers of the data points listed in the following tables are transferred along with the value in each frame to the communications partner. They can be evaluated by the communications partner.

The entries in the table row "Significance" relate to the entry in the table row "Bit status".

CP 1243-1, CP 1242-7 GPRS V2, CP 1243-7 LTE, CP 1542SP-1 IRC (TeleControl Basic)

The status bits are converted to the OPC quality code as follows by TCSB.

- Quality = BAD, if:
  NON_EXISTENT or OVER_RANGE = 1
- Quality = UNCERTAIN, if:
  RESTART or CARRY or SB = 1
- Quality = GOOD, if:
  Bits 1, 2, 3, 5 and 6 = 0

Table 10-90 Bit assignment of status byte 0

Bit	7	6	5	4	3	2	1	0
Flag name	-	NON_ EXISTENT	Substituted	LOCAL_ FORCED	CARRY	OVER_ RANGE	RESTART	ONLINE
Meaning	-	Data point does not exist or S7 address unreachable	Substitute value	Local operator control	Counted value overflow before reading the value	Limit value of the analog preprocessing overshot / undershot	Value not yet updated after start	Value is valid
Bit status	*(always 0)*	1	1	1	1	1	1	1

CP 1243-1 DNP3, CP 1542SP-1 IRC (DNP3)

The status IDs correspond to the following elements of the specification:

OBJECT FLAGS - DNP3 Specification, Volume 6, Data Object Library - Part 1

Table 10-91 Bit assignment of the status byte

Bit	7	6	5	4	3	2	1	0
Flag name	-	-	-	LOCAL_ FORCED	DISCONTI NUITY	OVER_ RANGE	RESTART	ONLINE
Meaning	-	-	-	Local operator control	Counted value overflow before reading the value	Limit value of the analog preprocessing overshot / undershot	Value not yet updated after start	Value is valid
Bit status	*(always 0)*	*(always 0)*	*(always 0)*	1	1	1	1	1

CP 1243-1 IEC, CP 1542SP-1 IRC (IEC)

The status IDs correspond to the following elements of the specification:

Quality descriptor - IEC 60870 Part 5-101

Table 10-92 Bit assignment of the status byte

Bit	7	6	5	4	3	2	1	0
Flag name	-	-	SB substituted	-	CY carry	OV overflow	NT not topical	IV invalid
Meaning	-	-	Substitute value	-	Counted value overflow before reading the value	Value range exceeded, analog value	Value not updated	Value is valid
Bit status	*(always 0)*	*(always 0)*	1	*(always 0)*	1	1	1	0

CP 1243-1 PCC

Table 10-93 Bit assignment of status byte 0

Bit	7	6	5	4	3	2	1	0
Flag name	-	NON_ EXISTENT	Substituted	LOCAL_ FORCED	CARRY	OVER_ RANGE	RESTART	ONLINE
Meaning	-	Data point does not exist or S7 address unreachable	Substitute value	Local operator control	Counted value overflow before reading the value	Limit value of the analog preprocessing overshot / undershot	Value not yet updated after start	Value is valid
Bit status	*(always 0)*	1	1	1	1	1	1	1

CP 1243-8 IRC

The CP 1243-8 IRC does not supply any status identifiers.

Rules for configuring the data point index:

Configuration of the data point index

Below you will find the rules for configuring the data point index for the individual CP types.

CPs using the TeleControl Basic protocol

Within a CP, the indexes of the data point classes must comply with the following rules:

- Input
  The index of a data point of the type input must be unique throughout all data point types (digital inputs, analog inputs etc.).
- Output
  - A data point of the type output can have the same index as a data point of the type input.
  - Several data points of the type output can have the same index.

Note

Data points for the inter-station communication with a CP in another S7 station

Note that for inter-station communication, the indexes of the two corresponding data points (data point pair) must be identical for the sending and receiving CP.

CPs using the DNP3 protocol

On a CP, data point indexes must be unique within each of the following object groups:

- Binary Input / Binary Input Event
- Binary Output / Binary Command
- Counter / Counter Event
- Analog Input / Analog Input Event
- Analog Output
- Octet String / Octet String Event

Indexes of two data points in different object groups can be identical.

CPs using the IEC protocol

The data point indexes must be unique in a CP.

Data point indexes assigned twice are indicated as errors in the consistency check and prevent the project being saved.

CP 1243-8 IRC

The data point indexes must be unique in a CP.

Data point indexes assigned twice are indicated as errors in the consistency check and prevent the project being saved.

CP 1243-1 PCC

The data point indexes must be unique in a CP.

Data point indexes assigned twice are indicated as errors in the consistency check and prevent the project being saved.

"Local setpoint input" (TIM)

Functions of the data point types for setpoint transfer

For transferring setpoints, for local setpoint entries and for feedback of the local values, the two data point types described below are available. For the functions the relevant input and output data points must be created for the communications partners.

The parameters for local setpoint entries and for the feedback are optional. If you do not require this, leave it disabled.

- **Setpoint input (Set01W_S)**
  Functions:
  - Sending a setpoint to the communications partner
    The setpoint is stored in the PLC tag (Word) in the "General" (Data source) tab.
  - Reception of the remote setpoint input mode "local" from the partner object
    The value is transferred to the PLC tag "Setpoint input mode "local"" (Bool).
  - Reception of the return value from the partner
    The value is transferred to the PLC tag "Return value" (Word, Int).
- **Setpoint output (Set01W_R)**
  Functions:
  - Reception of a setpoint from the communications partner
    The setpoint is transferred to the PLC tag (Word) in the "General" (Data source) tab.
  - Sending the local setpoint input status to the partner object
    The information as to whether the locally assigned setpoint or the remote setpoint is valid is read from the PLC tag "Setpoint status" (Bool).
  - Sending a locally entered setpoint to the partner
    The setpoint is read from the PLC tag "Local setpoint" (Word, Int).

Local setpoint input - Reception

The **Setpoint input (Set01W_S)**is used to send a setpoint to the communications partner.

In addition, the following parameters for local setpoints are available in the "Local setpoint input" tab of data point configuration:

- **Setpoint input mode of the partner**
  If the option is enabled, the setpoint input mode "local" received from the remote partner data point is saved in a PLC tag (Bool).
  - 0 = remote data point of the partner in the 'setpoint receipt' mode
  - 1 = remote data point of the partner in the setpoint input mode 'local'

  Via the parameter the feedback that the remote data point is set to local operator input (Setpoint status = 1) is received.
  The parameter is used only for signaling. With this parameter and with the corresponding parameter of the partner data point, there is no interlocking of the remote with the local setpoint assignment. An interlock of the remote and local setpoint assignment must be implemented in the user program.
  After startup of the CPU or partner CPU, or after the return of the connection, a general request ensures that the current, valid status of the partner is displayed at the parameter.
- **Return value**
  The partner data point that receives the setpoint feeds back the currently valid setpoint if the parameter "Setpoint status" is enabled there. This mirrored back setpoint is displayed here.
  If the partner object is set to 'local' and if new input is made there, the setpoint changed there is displayed here if the Local setpoint parameter is enabled for the partner object.
  After startup of the CPU or partner CPU, or after the return of the connection, a general request ensures that the current, valid setpoint of the partner is displayed at the parameter.

Local setpoint input - Sending

The **Setpoint output (Set01W_R)** is used to receive a setpoint from the communications partner.

In addition, the following parameters for local setpoints are available in the "Local setpoint input" tab of data point configuration:

- **Setpoint status**
  If the option is activated, the information as to whether the locally entered setpoint or the remote setpoint is valid is read from the PLC tag "Setpoint status" (Bool) and sent to the remote partner object.
  - 0 = remote setpoint of the partner
  - 1 = locally entered setpoint

  The information as to whether the locally set setpoint or the remote setpoint is valid can be signaled via this input. The current status of the Local input and a copy of the current local setpoint is returned (mirrored) to the partner. A setpoint sent by the remote partner (for example, master station) can also be accepted by the object if "Setpoint status" is set.
  The parameter is used only for information. An interlock with the remote setpoint assignment must be implemented in the user program.
- **Local setpoint**
  Via the parameter, the locally active setpoint is read from the PLC tag (Word, Int) and mirrored back to the send data point of the communications partner.

"Trigger“ tab

Trigger

Data points are configured as a static value or as an event using the "Type of transmission" parameter:

Saving the value of a data point configured as an event

Saving the value of a data point configured as an event in the send buffer (message memory) can be triggered by various trigger types:

- **Threshold value trigger**
  The value of the data point is saved when this reaches a certain threshold. The threshold is calculated as the difference compared with the last stored value, refer to the section Threshold value trigger (Page 1196).
- **Time trigger**
  The value of the data point is saved at configurable intervals or at a specific time of day.
- **Event trigger (Trigger tag)**
  The value of the data point is saved when a configurable trigger signal is fired. For the trigger signal, the edge change (0 → 1) of a trigger tag is evaluated that is set by the user program. When necessary, a separate trigger tag can be configured for each data point.
  Resetting the trigger tag in the bit memory area / DB:
  If the memory area of a trigger tag is in the bit memory or in a data block, the CP resets the trigger variable itself to 0 (zero) as soon as the value of the data point has been transferred. This can take up to 500 milliseconds.

Note

Fast setting of triggers

Triggers must not be set faster than a minimum interval of 500 milliseconds. This also applies to hardware triggers (input area).

Note

Hardware trigger

You need to reset hardware triggers via the user program

Transferring the value of a data point configured as an event

You specify whether the value of a data point is transferred to the communications partner immediately after the trigger fires or after a delay in the "Transmission mode" parameter.

Transmission mode

The transmission mode of a frame is set in the "Trigger" tab of the data point. With the option, you specify whether messages of events are sent immediately or following a delay:

- Immediate transfer - Spontaneous
  The value is transferred immediately.
- Buffered transfer - Conditionally spontaneous
  The value is transferred only when one of the following conditions is fulfilled:
  - The communications partner queries the station.
  - The value of another event with the transmission mode “Spontaneous" is transferred.

Threshold value trigger

Note

Threshold value trigger: Calculation only after "Analog value preprocessing"

Note that the analog value preprocessing is performed before the check for a configured threshold value and before calculating the threshold value.

This affects the value that is configured for the threshold value trigger.

Note

No Threshold value trigger if Mean value generation is configured

If mean value generation is configured, no threshold value trigger can be configured for the analog value event involved.

For the time sequence of the analog value preprocessing refer to the section Analog value preprocessing (Page 1198).

Threshold value trigger

Function

If the process value deviates by the amount of the threshold value, the process value is saved.

Two methods are used to calculate the threshold value deviation:

- **Absolute method**
  With binary and counter values as well as with analog values with configured mean value generation, the absolute method is used to calculate the threshold value deviation.
- **Integrative method**
  With analog values without configured mean value generation, the integrating method is used to calculate the threshold value deviation.
  In the integration threshold value calculation, it is not the absolute value of the deviation of the process value from the last stored value that is evaluated but rather the integrated deviation.

Absolute method

For each binary value a check is made to determine whether the current (possibly smoothed) value is outside the threshold value band. The current threshold value band results from the last saved value and the amount of the configured threshold value:

- Upper limit of the threshold value band: Last saved value + threshold value
- Lower limit of the threshold value band: Last saved value - threshold value

As soon as the process value reaches the upper or lower limit of the threshold value band, the value is saved. The newly saved value serves as the basis for calculating the new threshold value band.

Integrative method

The integration threshold value calculation works with a cyclic comparison of the integrated current value with the last stored value. The calculation cycle in which the two values are compared is 500 milliseconds.
(Note: The calculation cycle must not be confused with the scan cycle of the CPU memory areas).

The deviations of the current process value are totaled in each calculation cycle. The trigger is set only when the totaled value reaches the configured value of the threshold value trigger and a new process value is entered in the send buffer.

The method is explained based on the following example in which a threshold value of 2.0 is configured.

Table 10-94 Example of the integration calculation of a threshold value configured with 2.0

Time [s] (calculation cycle)	Process value stored in the send buffer	Current process value	Absolute deviation from the stored value	Integrated deviation
0	**20.0**	**20.0**	0	0
0.5		20.3	+0.3	0.3
1.0		19.8	-0.2	0.1
1.5		20.2	+0.2	0.3
2.0		20.5	+0.5	0.8
2.5		20.3	+0.3	1.1
3.0		20.4	+0.4	1.5
3.5	**20.5**	**20.5**	+0.5	**2.0**
4.0		20.4	-0.1	-0.1
4.5		20.1	-0.4	-0.5
5.0		19.9	-0.6	-1.1
5.5		20.1	-0.4	-1.5
6.0	**19.9**	**19.9**	-0.6	**-2.1**

With the changes in the process value shown in the example, the threshold value trigger configured with 2.0 fires twice:

- At the time 3.5 s: The value of the integrated deviation is at 2.0. The new process value stored in the send buffer is 20.5.
- At the time 6.0 s: The value of the integrated deviation is at 2.1. The new process value stored in the send buffer is 19.9.

In this example, if a deviation of the process value of approximately 0.5 should fire the trigger, then with the behavior of the process value shown here a threshold value of approximately 1.5 ... 2.5 would need to be configured.

Analog value preprocessing

CPs with data point configuration support analog value preprocessing. For analog value data points, some or all of the functions described below can be configured.

Requirements and restrictions

You will find the requirements for the configuration of the preprocessing options and restrictions in the section relating to the particular function.

Note

Restrictions due to configured triggers

The analog value preprocessing options "Fault suppression time", "Limit value calculation" and "smoothing" are not performed if no threshold value trigger is configured for the relevant data point.. In these cases, the read process value of the data point is entered in the image memory of the CP before the preprocessing cycle of the threshold value calculation (500 ms) elapses.

Sequence of the analog value preprocessing options

The values of analog inputs configured as an event are processed on the CPU according to the following scheme:

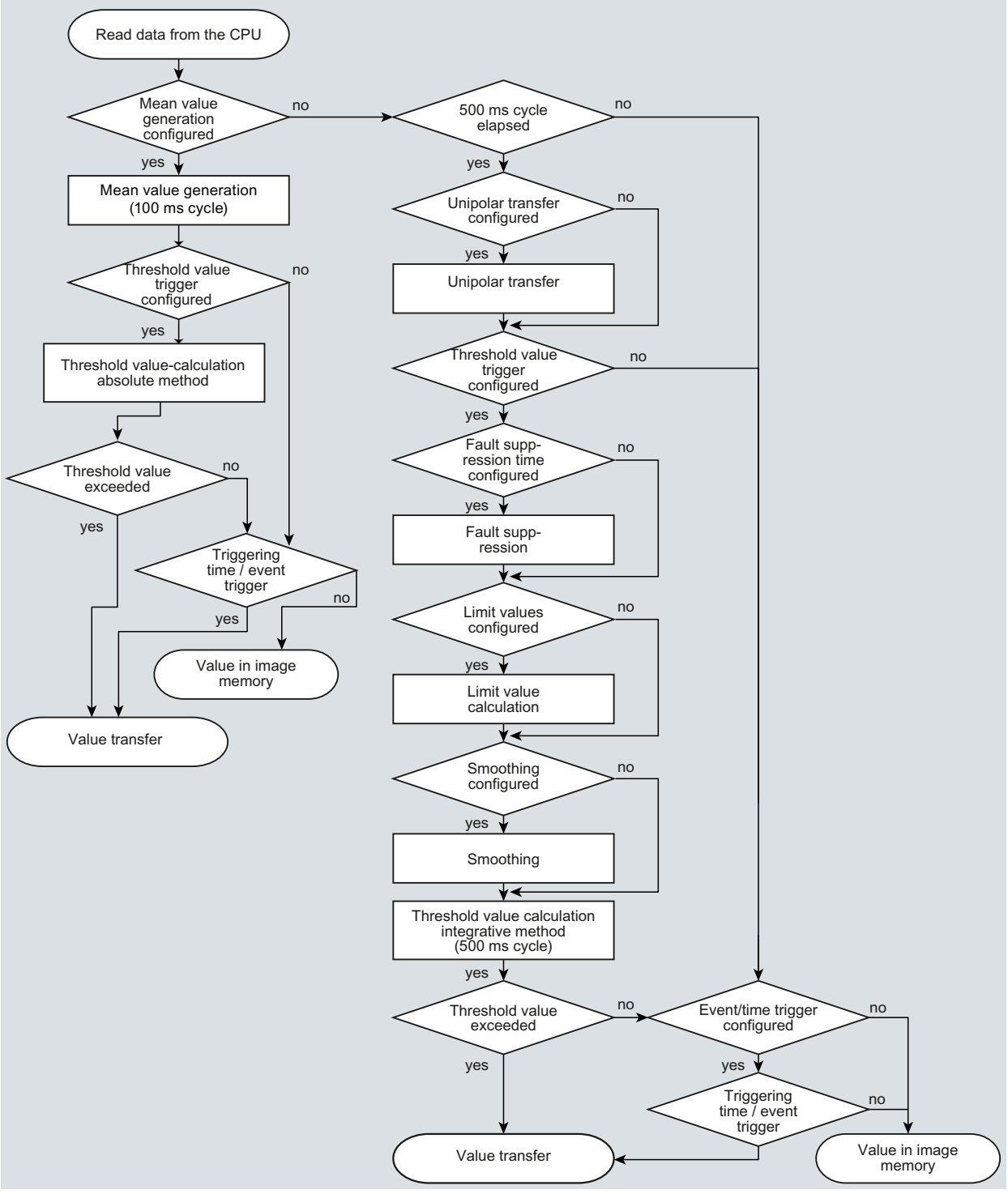


Figure 10-12 Sequence of the analog value preprocessing

The 500 millisecond cycle is started by the integrative threshold value calculation. In this cycle, the values are saved even when the following preprocessing options are enabled:

- Unipolar transfer
- Fault suppression time
- Limit value calculation
- Smoothing

Mean value generation

Note

Restricted preprocessing options if mean value generation is configured

If you configure mean value generation for an analog value event, the following preprocessing options are not available:

- Unipolar transfer
- Fault suppression time
- Smoothing

Function

With this parameter, acquired analog values are transferred as mean values.

If mean value generation is active, it makes sense to configure a time trigger..

The current values of an analog data point are read in a 100 millisecond cycle and totaled. The number of read values per time unit depends on the read cycle of the CPU and the CPU scan cycle of the CP.

The mean value is calculated from the accumulated values as soon as the transfer is triggered by a trigger. Following this, the accumulation starts again so that the next mean value can be calculated.

The mean value can also be calculated if the transmission of the analog value message is triggered by a request from the communications partner. The duration of the mean value calculation period is then the time from the last transmission (for example triggered by the trigger) to the time of the request. Once again, the accumulation restarts so that the next mean value can be calculated.

Input modules: Overflow range / underflow range

As soon as a value is acquired in the overflow or underflow range, mean value generation is stopped. The value 32767 / $7FFF_h$ or -32768 / 8000_h is saved as an invalid mean value for the current mean value calculation period and sent with the next message.

The calculation of a new mean value is then started. If the analog value remains in the overflow or underflow range, one of the two values named is again saved as an invalid mean value and sent when the next message is triggered.

Note

Fault suppression time > 0 configured

If you have configured an error suppression time and then enable mean value generation, the value of the error suppression time is grayed out but no longer used. If mean value generation is enabled, the error suppression time is set to 0 (zero) internally.

Unipolar transfer

Restrictions

Unipolar transfer cannot be configured at the same time as mean value generation. Enabling unipolar transfer has no effect when mean value generation is activated.

Function

With unipolar transfer, negative values are corrected to zero. This can be desirable if values from the underrange should not be transferred as real measured values.

Exception: With process data from input modules, the value -32768 / 8000_h for wire break of a live zero input is transferred.

With a software input, on the other hand, all values lower than zero are corrected to zero.

Fault suppression time

Requirements for the function

Configuration of the threshold trigger for this data point

Restrictions

The fault suppression time cannot be configured at the same time as mean value generation. A configured value has no effect when mean value generation is activated.

Function

A typical use case for this parameter is the suppression of peak current values when starting up powerful motors that would otherwise be signaled to the control center as a disruption.

The transmission of an analog value in the overflow ($7FFF_h$) or underflow range (8000_h) is suppressed for the specified time. The value $7FFF_H$ or 8000_H is only sent after the fault suppression time has elapsed, if it is still pending.

If the value returns to the measuring range before the fault suppression time elapses, the current value is transferred.

Input modules

The suppression is adjusted to analog values that are acquired directly by the S7 analog input modules as raw values. These modules return the specified values for the overflow or underflow range for all input ranges (also for live zero inputs).

An analog value in the overflow range (32767 / $7FFF_h$) or underflow range (-32768 / 8000_h) is not transferred for the duration of the fault suppression time. This also applies to live zero inputs. The value in the overflow/underflow range is only sent after the fault suppression time has elapsed, if it is still pending.

Recommendation for finished values that were preprocessed by the CPU:

If the CPU makes preprocessed finished values available in bit memory or in a data block, suppression is only possible or useful if these finished values also adopt the values listed above 32767 / $7FFF_h$ or -32768 / 8000_h in the overflow or underflow range. If this is not the case, the parameter should not be configured for preprocessed values.

For finished values preprocess in the CPU, the limits for the overflow and underflow can be freely assigned.

Smoothing factor

Requirements for the function

Configuration of the threshold trigger for this data point

Restrictions

The smoothing factor cannot be configured at the same time as mean value generation. A configured value has no effect when mean value generation is activated.

Function

Analog values that fluctuate quickly can be evened out using the smoothing function.

The smoothing factors are calculated according to the following formula as with S7 analog input modules.

$$y_n = \frac{x_n + (k-1)y_{n-1}}{k}$$

where
y_n = smoothed value in the current cycle
x_n = value acquired in the current cycle n
k = smoothing factor

The following values can be configured for the module as the smoothing factor.

- 1 = No smoothing
- 4 = Weak smoothing
- 32 = Medium smoothing
- 64 = Strong smoothing

Set limit value 'low' / Set limit value 'high'

Requirements for the function

- Configuration of the threshold trigger for this data point
- PLC tag in the bit memory operand area or data area
  The analog value data point must be linked to a PLC tag in the bit memory or data area (data block). For PLC tags of hardware modules (input operand area) limit value configuration is not possible.

The configuration of limit values is pointless for measured values that have already been preprocessed on the CPU.

Function

In these two input boxes, you can set a limit value in the direction of the start of the measuring range or in the direction of the end of the measuring range. You can also evaluate the limit values, for example as the start or end of the measuring range.

Status identifier "OVER_RANGE"

With CPs that support status identifiers (not the CP 1243-8 IRC), if the limit value is overshot or undershot, the status identifier "OVER_RANGE" of the data point is set. This status identifiers are described in the section Status IDs of the data points (Page 1189).

The "OVER_RANGE" bit of the status identifier of the data point is set as follows when the relevant analog value is transferred:

- Limit value 'high':
  - If the limit value is exceeded: OVER_RANGE = 1
  - If the value falls below the limit value: OVER_RANGE = 0
- Limit value 'low':
  - If the value falls below the limit value : OVER_RANGE = 1
  - If the value then exceeds the limit value: OVER_RANGE = 0

Configuration of the limit value

The limit value is configured as a whole decimal number. The range of values is based on the range of values of the raw value of analog input modules.

Range	Raw value (16 bits) of the PLC tag		Module output [mA]			Measuring range [%]
	Decimal	Hexadecimal	0 .. 20 (unipolar)	-20 .. +20 (bipolar)	4 .. 20 (life zero)	
Overflow	32767	7FFF	> 23.515	> 23.515	> 22.810	> 117.593
Overrange	32511 ... 27649	7EFF ... 6C01	23.515 ... 20.001	23.515 ... 20.001	22.810 ... 20.001	117.593 ... 100.004
Nominal range (unipolar / life zero)	27648 ... 0	6C00 ... 0000	20 ... 0		20 ... 4	100 ... 0

Range	Raw value (16 bits) of the PLC tag		Module output [mA]			Measuring range [%]
	Decimal	Hexadecimal	0 .. 20 (unipolar)	-20 .. +20 (bipolar)	4 .. 20 (life zero)	
Nominal range (bipolar)	27648 ... 0 ... -27648	6C00 ... 0000 9400		20 ... 0 ... -20		100 ... 0 ... -100
Underrange (unipolar / life zero)	-1 ... -4864	FFFF ... ED00	-0.001 ... -3.518		3.999 ... 1.185	-0.004 ... -17.59
Underrange (bipolar)	-27649 ... -32512	93FF ... 8100		-20.001 ... -23.516		-100.004 ... -117.593
Undershoot / wire break	-32768	8000	< -3.518		< 1.185	< -17.593

Note

Evaluation of the value even when the option is disabled

If you enable one or both options and configure a value and then disable the option later, the grayed out value is nevertheless evaluated.

To disable the two options, delete the previously configured values limit values from the input boxes and then disable the relevant option.

Recommendation for quickly fluctuating analog values:

If the analog value fluctuates quickly, it may be useful to smooth the analog value first if limit values are configured.

Masks

Masking of individual bits in the byte of binary inputs (Bin04B_S)

The "Masks" parameter group provides three options for transmitting binary value messages. The corresponding bits are masked in hexadecimal format.

By entering hexadecimal values you can specify bit-by-bit whether certain bits do not trigger message transmission or which bits are transferred in a different transmission mode.

- **Alarm mask**
  Changes in masked bits cause an unconditional spontaneous message transmission. Changes from 0 to 1 and from 1 to 0 are evaluated.
  The "alarm mask" is only practical when the data point is transmitted over a dialup network and the option "Conditional spontaneous" was activated.
- **Send buffer principle mask**
  Changes in masked bits cause a message transmission according to the send buffer principle. Changes from 0 to 1 and from 1 to 0 are evaluated.
  The send buffer mask is only practical with data points that have the "Image memory" option enabled.
- **Disable mask**
  Masked bits are ignored when changes are checked. This means that changes to the masked bits for this data point do not trigger message transmission. A masked bit always has the value 0 in the message.

Example of byte assignment

The following table shows the masking of the bits according to the hexadecimal value "A3" entered in the input box od the parameters.

Table 10-95 Byte assignment

Bit:	.7	.6	.5	.4	.3	.2	.1	.0
Masked	1	0	1	0	0	0	1	1
Hex mask	A				3			

The bits with the value 1 are masked; in other words bits no. 0, 1, 5 and 7 cause the relevant function in the described masks.

DNP3 parameters

The following parameters apply only to the CP when using the DNP3 protocol

Parameters for the data point type Binary Command Output (12)

The data point type allows receipt of a DNP3 command with the following control information (control code):

- **LATCH_ON / LATCH_OFF**
  or
- **PULSE_ON**
  When the byte "Control Code" is received with the function "PULSE_ON" information sent with it by the master "Count", "On-time" and "Off-time" is evaluated and compared to the object parameters "Max. pulse duration", "Pulse duration replacement time" and "Max. number of pulses" (see below).

The following control codes sent by the master are evaluated.

Table 10-96 Functions of the data object

Receipt of:			Reaction of the object depends on the configuration	
Control Code	**TCC ***	**Op Type ****	**Output mode = Pulse on**	**Output mode = Latch on/off**
0x01	NUL	PULSE_ON	The output is set to 1 for the duration of “On-time“.	The command is rejected.
0x03	NUL	LATCH_ON	The command is rejected.	The output is set permanently to 1.
0x04	NUL	LATCH_OFF	The command is rejected.	The output is set permanently to 0.
0x41	CLOSE	PULSE_ON	The output is set permanently to 1 (as with LATCH_ON).	The command is rejected.
0x81	TRIP	PULSE_ON	The output is set permanently to 0 (as with LATCH_OFF).	The command is rejected.

***** Trip-Close Code field

****** Operation Type field

Parameter

Name: **Max. number of pulses**

Range of values: 0 ... 255

Default: 0

Explanation: Monitors the number of pulses sent by the master (Count). If the number of pulses received from the master exceeds the value configured here, the command is rejected.

If you enter 0 (zero), the monitoring is disabled.

Name: **Max. pulse duration**

Range of values: 0 ... 2147483647

Default: 0

Explanation: Monitors the pulse duration sent by the master (On-time). If the pulse duration received from the master exceeds the value configured here, the command is rejected.

If you enter 0 (zero), the monitoring is disabled.

Name: **Pulse duration equivalent time**

Range of values: 0 ... 2147483647

Default: 0

Explanation: Replacement value for the pulse duration If the pulse duration received from the master exceeds the value configured in “Max pulse duration“. The parameter is only used when “Max. pulse duration“ is configured.

If the value is 0 (zero), no replacement value is used.

Message editor

Processing status of messages (e-mails / SMS messages)

Enable status identifier / External status

If the option is enabled, the CP writes a processing status about the sent e-mail / SMS to a PLC tag.

The meaning of the statuses is as follows:

Table 10-97 SMS: Meaning of the status ID output in hexadecimal format

Status	Meaning
0000	Transfer completed free of errors
0001	Error in the transfer, possible causes: • SIM card invalid. • No network • Wrong destination phone number (number not reachable)

Table 10-98 E-mails: Meaning of the processing status output in hexadecimal format

Status	Meaning
0000	Transfer completed free of errors
82xx	Other error message from the e-mail server Apart from the leading "8", the status corresponds to the three-digit error number of the SMTP protocol.
8401	No channel available Possible cause: There is already an e-mail connection via the CP. A second connection cannot be set up at the same time.
8403	No TCP/IP connection could be established to the SMTP server.
8405	The SMTP server has denied the login request.
8406	An internal SSL error or a problem with the structure of the certificate was detected by the SMTP client.
8407	Request to use SSL was denied.
8408	The client could not obtain a socket for creating a TCP/IP connection to the mail server.
8409	It is not possible to write via the connection. Possible cause: The communications partner reset the connection or the connection aborted.

Status	Meaning
8410	It is not possible to read via the connection. Possible cause: The communications partner terminated the connection or the connection was aborted.
8411	Sending the e-mail failed. Cause: There was not enough memory space for sending.
8412	The configured DNS server could not resolve specified domain name.
8413	Due to an internal error in the DNS subsystem, the domain name could not be resolved.
8414	An empty character string was specified as the domain name.
8415	An internal error occurred in the cURL module. Execution was aborted.
8416	An internal error occurred in the SMTP module. Execution was aborted.
8417	Requests to SMTP on a channel already being used or invalid channel ID. Execution was aborted.
8418	Sending the e-mail was aborted. Possible cause: Execution time exceeded.
8419	The channel was interrupted and cannot be used before the connection is terminated.
8420	Certificate chain from the server could not be verified with the root certificate of the CP.
8421	Internal error occurred. Execution was stopped.
8450	Action not executed: Mailbox not available / unreachable. Try again later.
84xx	Other error message from the e-mail server Apart from the leading "8", the status corresponds to the three-digit error number of the SMTP protocol.
8500	Syntax error: Command unknown. This also includes the error of having a command chain that is too long. The cause may be that the e-mail server does not support the LOGIN authentication method. Try sending e-mails without authentication (no user name).
8501	Syntax error. Check the following configuration data: Message configuration > Message parameters: • Recipient address ("To" or "Cc").
8502	Syntax error. Check the following configuration data: Message configuration > Message parameters: • Email address (sender)
8535	SMTP authentication incomplete. Check the "User name" and "Password" parameters in the CP configuration.
8550	SMTP server cannot be reached. You have no access rights. Check the following configuration data: • CP configuration > E-mail configuration: – User name – Password – Email address (sender) • Message configuration > Message parameters: – Recipient address ("To" or "Cc").
8554	Transfer failed
85xx	Other error message from the e-mail server Apart from the leading "8", the status corresponds to the three-digit error number of the SMTP protocol.

Activation of the serial interface interface of the CP 1243-8 IRC (V3) for the ST7 protocol

Serial interface of the CP 1243-8 IRC

When you insert the CP from the catalog in the project, the serial interface of the CP is not visible in the network view of STEP 7.

When using the types of communication "DNP3" and "IEC" the serial interface cannot be used.

After activating thde type of communication "ST7", the serial interface is visible but cannot be configured. To use the serial interface of the CP, a TS module is required.

Follow the steps below to insert the TS module and to activate the serial interface for ST7 communication.

1. From the network view, open the device view of the CP (double-click on the module).
   The device view opens.
   At the same time the catalog view changes. Among other things the Communications modules" directory appears.
2. Expand the "Communications modules" directory:
   > Industrial Remote Communication > TS Modules
3. Expand the subdirectory of the required TS module.
4. Select the TS module.
   A vertical blue line appears on the symbol of the CP.
5. Now drag the TS module onto the blue line of the CP in the device view.
6. The TS moduell with the serial interface is inserted in the station.
   At the same time a new WAN interface is inserted in the network view.

You can configure the serial interface directly in the device view with the TS adapter or in the WAN interface of the CP in the network view.

Configuring telecontrol connections for the TIM

To be able to use ST7 communication of the TIM via classic WAN networks (dedicated line, dialup network) you need to create telecontrol connections in the "Network data" editor.

Opening the "Network data" editor with the "Telecontrol" tab

To open the editor, follow the steps below:

1. Open the network view of the project.
   On the right you will find the initially hidden "Network data" editor.
2. Open the "Network data" editor using the arrow symbol.
   The "Network data" editor is displayed with several tabs. The first tab on the left is the network overview.
3. Expand the "Network data" editor until the "Telecontrol" tab appears on the right.
   In this tab you configure the telecontrol connections.

Creating telecontrol connections for the the ST7 communication of the TIM

For ST7 communication between two stations with a TIM module via classic WAN networks (dedicated line, dialup network) you require at least one telecontrol connection. Follow the steps outlined below:

1. In the "Starting point" box, click on a free line in the "Telecontrol" tab of the "Network data" editor.
   The drop-down list that opens displays the configured stations (CPUs or applications) that use the ST7 protocol.
2. Select a station by double-clicking.
   The connection will then be included in the list of telecontrol connections.
   The name of the connection is preset, but you can change it.
3. Now in the "Endpoint" box select the connection partner from the displayed drop-down list.

Boxes with a missing or bad configuration are shown on a red background.

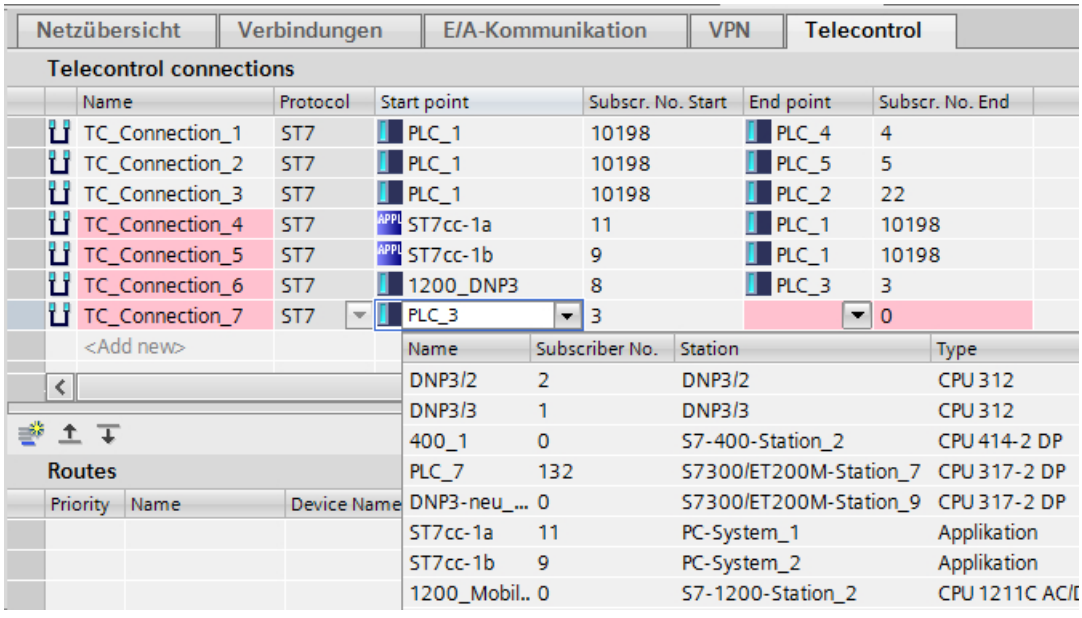


Figure 10-13 "Network data" editor, "Telecontrol" tab

Specification of the connection sequence

Then specify the connection sequence of the created telecontrol connection.

1. Above the second table "Configured connections" click the paths symbol  .
   The dialog "Add connection paths" opens and the possible connections re searched for.
2. if one or more connections are displayed in the upper table, select the required connection in the upper table.
   The connection details with the connection sequence are shown in the second table.
3. Close the dialog with the "Close" button if no connection is found.
   In this case, there is no suitable network between the two stations and you will need to complete the configuration.
4. Click on "Add" if you want to choose the selected connection as the configured connection.

5. The reaction whether the connection was added or whether there is an error in the configuration is displayed at the bottom of the dialog in "Information".
6. Close the dialog with the "Close" button if the configured connection corresponds to the demands of the project.

The connection details of every possible and selected connection are displayed in the lower table "Connection details".

- The "Position" column shows a station symbol with symbol for the connection point (starting point, node, endpoint)
  Based on the color of the symbols for the connection endpoint you can recognize connections that are not permitted.
  - Blue symbols for the connection point: Valid connections
  - Red symbols for the connection point: Invalid connections

Delete invalid or redundant connections

If you have configured several connections between two stations and the connections are shown on a red background in the table, the connection configuration is invalid. When checking the possible connections. the table "Connection details" supports you.

If you have non-permitted redundant connections you need to delete a connection. Follow the steps outlined below:

1. To delete a connection in the table "Configured connections", select the connection you do not need.
2. Delete the connection using the shortcut menu "Delete".

Rules for connection configuration

Note the following rules for connection configuration:

- For a redundant ST7cc control center, i STEP 7 create two PC stations each with an ST7cc application.
  Only in this way can you configure a connection from a subscriber to every ST7cc.
- An MSC connection can only run via the first Ethernet interface [X1] of a TIM.
- A connection via an inconsistent network is invalid. Examples:
  - Subscriber with incompatible modems
  - Incompatible settings of two modems in a connection
  - Incompatible settings between modem and network parameters
  - Connections via nodes that are not configured s node stations
- Two connections to one partner (endpoint) via the same interface of a TIM (start point) are not permitted.
- A connection via a network in which two MD3 modems communicate with each other and that are configured with 1200 bps / half duplex / AT mode is invalid.

Configuration of further connection properties

If you select a connection in the table "Telecontrol connections" at the top in the "network data" editor, three parameter groups are displayed in the "Properties" tab of the Inspector window in which you can check, complete or correct the configuration of the connections.

- **General**
  Here, you will find the basic properties of a telecontrol connection.
- **Reduntant ST7cc**
  If you use a redundant ST7cc control center in your project, you must first create the two PC stations and the two connections as described above.
  Afterwards you create a redundancy group for each individual connection to the ST7cc here in the parameter group "Redundant ST7cc".
  - To do this, activate the option "Add redundant ST7cc/ST7cc partner" for every connection of a station to the ST7cc.
  - Select the required application from the "Redundant application" drop-down list.
- **Destination subscriber properties**
  - General request supervision time
    This is the maximum time required by a destination subscriber to respond fully to a general request (GR).
    If the GR response has not arrived completely at the requesting TIM when the supervision time has expired, a message is entered in the diagnostics buffer of this TIM.
    The time should be selected generously. In particular with dial-up connections, it should be remembered that the time for connection establishment is included in the supervision time. In addition to this, with dial-up stations the messages stored in the send buffer further delay processing of the GR because the queried messages are entered in the send buffer behind all other messages.
  - Frames to destination subscriber with time stamp
    The parameter relates to the destination subscriber that was configured as the endpoint of the connection.
    If the option is enabled all messages are sent to this destination subscriber with a time stamp.
    If the option is disabled all messages are sent to this destination subscriber without a time stamp.

Mobile wireless CPs

CP 1242-7 (6GK7 242-7KX30-0XE0)

Operating modes of CP 1242-7

Modes of the CP

CP 1242-7 (6GK7 242-7KX30-0XE0) allows an S7-1200 to communicate as a GPRS station with a master or other remote networks via the GSM network. For communication using GPRS, the CP is set to one of the following operating modes:

- **Telecontrol**
  This operating mode of the CP allows the GPRS station to exchange data with the following partners:
  - Communication with the Telecontrol server
    This CP mode allows the GPRS station to exchange data with a telecontrol server. The Telecontrol server is a PC connected to the Internet with the "TCSB" application. It is generally located in the master and serves to monitor and control the remote GPRS stations. Data can be exchanged with the OPC client of a central control system via the integrated OPC interface.
    The Telecontrol server PC is not configured in STEP 7. The "TCSB" application has its own configuration user interface.
  - Communication with another remote GPRS station
    The message frames are transmitted via the Telecontrol server.
  - Communication with an engineering station (for TeleService)

  Communication with the Telecontrol server is performed via the GSM network and the Internet.
  This operating mode requires a SIM card with GPRS service enabled and a Telecontrol server that can be reached by the CP.
- **GPRS direct**
  This operating mode of the CP is used for direct communication between remote stations via the GSM network. No Telecontrol server is required.
  To allow network nodes in public wireless networks to be directly accessible, these need to be addressed using a fixed address. Here, SIM cards with a fixed IP address are used that allow the stations to address each other directly.
  The possible communications services and security functions (for example VPN) depend on what is offered by the network provider.
  Possible communications partners of the station with a CP 1242-7 in "GPRS direct" mode are:
  - A node that can be reached by the CP via an IP address (S7 station with CP 1242-7)
  - An engineering station (for TeleService)

See also

Connection establishment with a CP 1242-7 (Page 1214)

Connection establishment with a CP 1242-7

Connection modes of CP 1242-7 (6GK7 242-7KX30-0XE0)

- "GPRS direct" mode
  There are no different connection modes in the "GPRS direct" mode.
- "Telecontrol" mode
  The CP can be configured for the following connection modes.
  - "Permanent" connection mode
    There is a permanent TCP connection to the telecontrol server. Following connection establishment, there is a permanent TCP connection to the telecontrol server even if data is not transferred permanently.
  - "Temporary" connection mode
    A connection is only established to the telecontrol server when required.

If a TCP connection is established, process data is sent as soon as the telecontrol instructions are called on the CPU.

A connection is always established by the CP. If a connection established by the CP is interrupted, the CP automatically attempts to re-establish the connection.

Triggering connection establishment for permanent stations ("Telecontrol" mode)

In the "Telecontrol" mode, the permanent connection to the telecontrol server is established when the station starts up. If there is an interruption on the connection, the establishment of the connection can be triggered by a wake-up SMS (see below).

Triggering connection establishment for temporary stations ("Telecontrol" mode)

With "temporary" stations, connection establishment can be triggered by the following events:

- Event on the local CPU that needs to be evaluated by the program.
  In terms of the program, two situations need to be distinguished:
  - Events that lead to a single connection establishment (for example alarms or commands from the operator).
  - Expiry of an interval that leads to cyclic connection establishment (for example once daily for data transmission)
- Request by a communications partner (OPC client or S7 station)
  The request from the communications partner leads to the connection being established.
- Request for TeleService by an engineering station
  The request switched by the telecontrol server or TeleService gateway does not need to be evaluated in the program.
- Wake-up SMS of the telecontrol server
  The wake-up SMS can be triggered spontaneously on the telecontrol server. It is also possible to configure cyclic sending on the telecontrol server.

- Telephone wake-up call
  The wake-up call can be sent from a telephone that has a phone number authorized in the STEP 7 project. The telephone must support the CLIP function (transfer of its own call number).
  The connection establishment with the (main) telecontrol server is triggered.
- Telephone wake-up SMS
  The wake-up SMS can be sent from a telephone that has a phone number authorized in the STEP 7 project. The telephone must support the CLIP function (transmission of its own number) and sending of SMS.
  The connection establishment with the telecontrol server specified in the SMS is triggered.

When a temporary station is woken up, all the data is transferred if this has changed since the last data transfer.

Wake-up call / wake-up SMS in the "Telecontrol" operating mode

For wake-up call or wake-up SMS, see section Wake-up of cellular CPs (Page 1215).

Triggering connection establishment in "GPRS direct" mode

In "GPRS direct" mode, a connection establishment is triggered by the following events:

- Event on the local CPU that is evaluated by the program.
- Request by a communications partner (not an engineering station)
  The request in the frame received from the communications partner is evaluated in the program by calling the telecontrol instructions.
- Request for TeleService by an engineering station
  The request switched by the telecontrol server or TeleService gateway does not need to be evaluated in the program.

Wake-up of cellular CPs

Right to wake-up by "Authorized phone numbers"

The CP only accepts a wake-up SMS or a wake-up call if the sending communication partner is authorized based on its phone number. These numbers are in configured for the CP in STEP 7 in the "authorized phone numbers" list.

Note

"Authorized phone numbers" in the STEP 7 project

- A phone number entered here gives the sender who transfers this phone number the right to trigger connection establishment.
- If an asterisk (*) is entered in the list, the CP accepts SMS messages from all senders.
- If the list is empty, the CP cannot be woken up for connection establishment.

Wake-up SMS

Depending on the connection type and the triggering server or intermediary TeleService server, the following text must be transferred in the wake-up SMS:

- For telecontrol connections:
  - Text for the wake-up SMS message for establishing a connection to the telecontrol server:
    `TELECONTROL`
  - Text for the wake-up SMS message for establishing a connection to the main telecontrol server:
    `TELECONTROL MAIN`
  - Text for the wake-up SMS message for establishing a connection to the substitute telecontrol server:
    `TELECONTROL BACKUP`

  The configuration of the telecontrol server for the GPRS CP is set in STEP 7 in "Telecontrol interface > Operating mode > main or substitute telecontrol server".

  Note

  Wake-up with a mobile phone

  - One of the texts listed above can be used in a wake-up SMS message.
  - With a wake-up call, the station always connects to the main telecontrol server.

- For TeleService connections:
  - Text for the wake-up SMS message for establishing a connection to the first configured TeleService server:
    `TELESERVICE`
    or
    `TELESERVICE 1`
  - Text for the wake-up SMS message for establishing a connection to the second configured TeleService server:
    `TELESERVICE 2`

  The configuration of the TeleService server for the GPRS CP is set in STEP 7 in "Telecontrol interface > TeleService authorization > 1st or 2nd TeleService server".

Preferred mobile wireless networks

Selection of the preferred mobile wireless networks

The following options are available to select the networks that the mobile wireless CP should preferably dial up:

- Automatic
  The CP dials with highest priority into the mobile wireless network of the contracted network provider set on the SIM card. If a dialing up the contract network fails, the CP dials up other mobile wireless networks with which the network provider has roaming contracts and whose access data is stored on the SIM card.
- Contract network only
  The CP only dials up the mobile wireless network of the contract network provider whose SIM card is plugged into the CP. No roaming.
- Contract network and preferred networks
  The CP dials up the contracted network with the highest priority. If dialing up the contract network is unsuccessful, the CP dials up alternative mobile wireless networks, in decreasing priority, as entered in the list of preferred network providers.
  The alternative networks are entered in the list as "Public Land Mobile Network" (PLMN). PLMN is a construct of Mobile Country Code (MCC) and Mobile Network Code (MNC).
  Example: 26276
  This is the PLMN for the test network of Siemens AG with MCC = 262 and MNC = 76.

TeleService via mobile wireless

Download using TeleService

The following behavior applies to all mobile wireless CPs:

- CP 1242-7 (6GK7 242-7KX30-0XE0)
- CP 1242-7 GPRS V2 (6GK7 242-7KX31-0XE0)
- CP 1243-7 LTE-EU (6GK7 243-7KX30-0XE0)
- CP 1243-7 LTE-US (6GK7 243-7SX30-0XE0)

Connection resources for TeleService

The TeleService function occupies a connection resource on the engineering station.

The function download to device or upload from device during a TeleService session occupies a second connection resource on the engineering station.

Download to device

Only use the "Download to device" function with the mobile wireless CP via a TeleService connection as follows:

1. Select the CP in STEP 7.
2. Select the "Online" > "Download to device" menu.
3. In the "Extended download" dialog that appears, select the TeleService interface.
4. Download the project data from the "Extended download" dialog.

Download from device

The "Download from device" function via a TeleService connection is supported by the mobile wireless CPs with the following TeleService server applications:

- TeleControl Server Basic as of version V3
- TeleService Gateway as of version V3

TeleService via telecontrol

Communication path for TeleService via telecontrol

With TeleService via telecontrol, the connection between the engineering station and a remote S7 station always runs through a TeleService server as an intermediary.

The request for connection establishment is triggered by the engineering station and forwarded by the TeleService server. Depending on the available communication paths to the station, the TeleService server forwards the request to the next station via LAN or by a wake-up SMS. The telecontrol CP in the S7-1200 station establishes a connection to the engineering station via the TeleService server.

An alternative intermediary (TeleService server) is:

- A telecontrol server
  The telecontrol server can be a separate PC or installed on the engineering station in the form of the "TCSB" application.
- A TeleService gateway
  A TeleService gateway is used when no telecontrol server is available.

The TeleService server can be connected to the engineering station via LAN or Internet and the TeleService function can be called from there.

Requirements for TeleService via telecontrol

- A TeleService server
- The STEP 7 project with the required stations

Note on project engineering

The TeleService server is not configured in STEP 7.

Establish remote connection via telecontrol

Requirements for TeleService via telecontrol

For the requirements and mechanisms, see section TeleService via telecontrol (Page 1218).

Start a TeleService session via telecontrol

Start TeleService via telecontrol as follows:

1. In the project on the authorized engineering station, select the remote S7 station with which you want to establish a TeleService connection via WAN.
2. Alternatively, open the "Connect online" dialog box as follows:
   - "Connect online" button
   - "Connect online" shortcut menu (right mouse button)
   - "Online" > "Connect online" menu

   The "Connect online" dialog box opens.
3. Select the interface type "TeleService via Telecontrol" in the "Type of PG/PC interface" drop-down list.
4. In the "PG/PC interface" drop-down list, select the "TeleService board" option if it is not automatically displayed.
5. Click on the "Connect" icon next to the "PG/PC interface" drop-down list.
   The "Establish remote connection via telecontrol" dialog box opens.
6. Make the necessary settings in this dialog.
   Details of this are contained in the tooltip cascade of STEP 7.

The following details are required to successfully establish a connection:

Details required to establish a connection with the S7 station

The following information is necessary in the Establish remote connection" dialog:

- IP address or DNS name of the telecontrol server
- TCP port number of the telecontrol server or the DSL router via which the connection between the engineering station and the remote S7 station is running.
- Server password of the ES to authenticate the engineering station at the telecontrol server
  Only required if a group-specific password has been configured in the "TCSB" application.
- TeleService user name
  See CP configuration in STEP 7.
- TeleService password
  See CP configuration in STEP 7.
- Access ID of the CP
  Only required when there are several telecontrol CPs in the station. See CP configuration in STEP 7.

Status

Connection states with TeleService via telecontrol

The connection states described below can be displayed in the "Establish remote connection via telecontrol" dialog box.

When opening the dialog

- Not connected
  There is no connection to the remote S7 station. Connection establishment has not yet started.

When establishing a connection

If connection establishment was started by clicking the "Connect" button, the following states are displayed one after the other after the connection has been established:

- Connect to telecontrol server
  The engineering station connects to the telecontrol server.
- Wait for S7 station
  The wake-up SMS has been sent to the remote station. Wait for reply from station.
- Authentication at S7 station
  The S7 station has established an IP connection with the engineering station via GPRS and Internet and is checking the received logon and authentication data.
- Connected
  The station has successfully established the connection to the engineering station.

If connection establishment is unsuccessful

The following states may be displayed if the connection cannot be successfully established:

- Telecontrol server not accessible
  Possible causes:
  - The connection between the engineering station and the telecontrol server has been interrupted.
  - The telecontrol server is switched off.
- Wrong server password
  Cause: The wrong server password for logging on and authentication was entered at the telecontrol server.
- S7 station not responding
  Possible causes:
  - The communication between the telecontrol server and the station is disrupted.
  - There is a disrupted connection between the mobile wireless network and the Internet.
  - Faulty Internet connection.
  - The telecontrol server could not send a wake-up SMS.
  - The CP has not received a wake-up SMS.
  - The sender of the SMS was not configured in the list of authorized wake-up call numbers.

- Wrong TeleService user name or TeleService password
  Possible causes:
  - An incorrect TeleService user name or incorrect TeleService password was entered in the authentication dialog for the CP.
  - The TeleService user name or TeleService password was not configured in STEP 7.
- All TeleService access points are in use.
- The CP is not known to the telecontrol server.
  Cause: The CP originates from a STEP 7 project that does not match the project of the telecontrol server.
- No resources available for TeleService on the CP: Please contact the hotline.
- Protocol error
  Cause: Wrong frame or frame from wrong subscriber. Please contact the hotline.

CP 1243-8 IRC (V2.1)

Ethernet interface

"Station address" (CP 1243-8 IRC V2.1) parameter group

"Station address": Adoption of the subscriber numbers from STEP 7 V5

The subscriber numbers are only displayed after you have imported the configuration data of the station from the STEP 7 V5 project.

The subscriber numbers cannot be edited.

As a default, all CPUs and CP 1243-8 have the subscriber number 0.

The displayed parameters are shown alongside the corresponding parameters in STEP 7 V5.

In the text file of the configuration data you will find the parameters in the SDB "Subscriber data".

Table 10-99 "Station address" parameter group

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
Node number of the CPU	Node number of the CPU	SINAUT configuration tool > Subscriber administration	
Node number of the TIM	Node number of the TIM	SINAUT configuration tool > Subscriber administration	

Time-of-day synchronization

Time synchronization: Adoption of parameters from STEP 7 V5

The parameter group "Time synchronization" is only displayed after you have imported the configuration data from the STEP 7-V5 project.

The displayed parameters are shown alongside the corresponding parameters in STEP 7 V5.

In the text file of the configuration data you will find the parameters in the SDB "WAN data".

Table 10-100 "Time synchronization" parameter group

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
Synchronization cycle	Synchronization cycle	Properties dialog TIM > "Time Service" tab	
• Seconds intervals • Minute intervals • Hourly intervals • Time • No synchronization	• Seconds intervals • Minute intervals • Hourly intervals • Time • No synchronization	Properties dialog TIM > "Time Service" tab	
Hourly intervals / Minute intervals / Seconds intervals	Hourly intervals / Minute intervals / Seconds intervals	Properties dialog TIM > "Time Service" tab	Displays the configured value
Time master	Synchronization master	Properties dialog TIM > "Time Service" tab	

Advanced options > TCP connection monitoring

Ethernet interface (X1) > Advanced options > TCP connection monitoring

The setting applies to all TCP connections of the CP.

Exception: The settings made here do not apply to connections programmed for Open User Communication with the program blocks.)

- **TCP connection monitoring time**
  If there is no data traffic within the connection monitoring time, the CP sends a keepalive to the communications partner.
- **TCP keepalive monitoring time**
  After sending a keepalive frame, the CP expects a reply from the communications partner within the keepalive monitoring time. If the CP does not receive a reply within the configured time, the keepalive frame is repeated twice. After the monitoring time elapses three times without a reply, the CP terminates and re-establishes the connection.

"MSC station parameters" (CP 1243-8 IRC V2.1) parameter group

"Advanced options" > "MSC station parameters": Adoption of the data from STEP 7 V5

The following parameters are only displayed after you have imported the configuration data of the station from the STEP 7 V5 project.

The displayed parameters are shown alongside the corresponding parameters in STEP 7 V5.

Table 10-101 "MSC master" parameter group

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
Port number MSC master	MSC port of the TIM	SINAUT configuration tool > "MSC Master Properties"	MSC port of the master
MSCsec	Security protocol active	SINAUT configuration tool > "MSC Station Properties"	MSCsec protocol is used when option is enabled
Key exchange interval	Rekeying interval [hours]	SINAUT configuration tool > "MSC Station Properties"	
Own WAN address	WAN address	Properties WAN network nodes > "Network Attachment" tab	Own WAN address (station address)

Table 10-102 Parameter group "MSC partner parameters"

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
Partner WAN address	WAN address	Properties WAN network nodes > "Network Attachment" tab	WAN address (station address) of the partner (master)
Connection mode	Connection mode	Properties TIM > "Interfaces" tab	Interface mode of the partner:
Address type		SINAUT configuration tool > "MSC Master Properties"	Internet Access (MSC master): • Name (via DNS) • IP address
MSC protocol version	Enable security protocol	SINAUT configuration tool > "MSC Station Properties"	Protocol variant of the CP (MSC / MSCsec)
User name	User name	SINAUT configuration tool > "MSC Station Properties"	
Password	Password	SINAUT configuration tool > "MSC Station Properties"	
Partner IP address		SINAUT configuration tool > "MSC Master Properties"	IP address of the router (Internet Access)

Serial interface

WAN parameters(CP 1243-8 IRC V2.1)

"WAN parameters": Adoption of parameters from STEP 7 V5

The following parameters are only displayed after you have imported the configuration data of the station from the STEP 7 V5 project.

The displayed parameters are shown alongside the corresponding parameters in STEP 7 V5. Depending on the configuration of the CP (protocol, network type, station type) not every parameter is displayed.

In the text file of the configuration data you will find the parameters in the SDBs "WAN data" and "Ethernet data".

Table 10-103 Parameter group "WAN parameters"

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
Interface number	Number of the interface	Properties TIM > "Interfaces" tab	
TIM type	Node type	Properties WAN network nodes > "Network Attachment" tab	
Network type	Network type	HW Config	With Ethernet nodes derived from "Connection mode" (Properties TIM > "Interfaces" tab)
Operating mode	Operating mode	Properties Network > "Network Settings"	• Spontaneous (with Ethernet networks) • Polling • Polling with time slots • Multi-master-polling / time slots • Network operating mode not used
Frame format	Frame format	Properties Network > "Network Settings"	
Acknowledgment type	Acknowledgment type	Properties Network > "Network Settings"	
Repetition factor	Repetition factor	Properties Network > "Network Settings"	Preassigned with "3" for Ethernet
Connection type	Connection type	Properties Network > "Network Settings"	Dependence on direction
Telecontrol protocol	WAN protocol	Properties Network > "Network Settings"	
Own WAN address	WAN address	Properties WAN network nodes > "Network Attachment" tab	Own WAN address (station address)

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
Max. frame length	Max. message length	Properties Network > "Network Settings"	
Number of local CPUs			Number of local CPUs
Number of spontaneous frames	Number of spontaneous frames	Properties WAN network nodes > "Basic Param." tab	
Max. permitted disturbance time	Max. permitted disturbance time	Properties Network > "Dedicated line" tab	
Partner WAN address	WAN address	Properties WAN network nodes > "Network Attachment" tab	WAN-address (Station address) of the partner
Baud rate	Baud rate [bps]	Properties Network > "Network Settings"	[Bit/s]
Monitoring time	Depending on the network type: • Polling Monitoring time • Call answer delay • Monitoring time		Network type: • Dedicated line • Dial-up network • MSC network
RTS/CTS delay time	RTS/CTS delay time [ms]	Properties WAN network nodes > "Dedicated line" tab	
Send delay time	Send delay time	Properties WAN network nodes > "Dedicated line" tab	
Dial mode	Dial mode	Properties WAN network nodes > "Dialing Param." tab	
Dialing format	Dialing format	Properties WAN network nodes > "Dialing Param." tab	
Dial test interval	Dial test interval	Properties Network nodes > "Dialup network" tab	
Additional frame run time	Additional frame run time	Properties WAN network nodes > "Basic Param." tab	
ID	Customer identification	Properties Network > "Network Settings"	
Transmission criterion	Transmission criterion	Properties WAN network nodes > "Dialup network" tab	
Hour / minute (transmission criterion)		"Transmission criterion" parameter	Display of "Hour" / "Minute" when transmission criterion = "Time" or "Interval".
Interface operating mode "Interrupt"	Operating mode	Properties Network nodes > "Basic Param." tab	
AT initialization	Initialization string	Properties Network nodes > "AT initialization" tab	
AT initialization when using MSC			AT initialization when using MSC
PIN	PIN number	Properties WAN network nodes > "Dialing Param." tab	

Table 10-104 "Call parameters" parameter group

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
Own phone number	Own phone number	Properties WAN network nodes > "Dialing Param." tab	
Redial	Redial	Properties Network > "Network Settings"	
Cancel parameter	Cancel parameter	Properties Network > "Network Settings"	
Station type			WAN driver type (master / node / station)
Special services	Special service	Properties WAN network nodes > "Dialing Param." tab	SMS special services
Subscriber release status		SINAUT configuration tool > tab "Telephone Directory" >Properties - Telephone Number	Enable status of the subscriber (enabled / disabled)
Station address	WAN address	Properties WAN network nodes > "Network Attachment" tab	
Phone number		SINAUT configuration tool > Network configuration	Phone number of the subscriber, components: Initialization command (max. 28 characters) or "AT", Dial command (max.. 4 characters.), telephone number (max. 32 characters)

Partner stations

"Partner stations (CP 1243-8 IRC V2.1)" parameter group

Import partner configuration

With the "Import partner configuration" button, you import the text data with the STEP 7 V5 configuration data of the proxy from the file system of the engineering station.

You will find a description of the parameters in the following sections.

Reset partner configuration

With the "Reset partner configuration" button, you delete all the configuration data of the CP that you previously imported from a text file with the "Import partner configuration" button. With this function you reset the partner configuration data to the status of an unconfigured module.

Imported configuration data file

This output box shows the name of the text file whose configuration data you last loaded on this CP.

Last import

This output box shows the time at which you imported the configuration data file.

Partner configuration data

Adoption of parameters from STEP 7 V5

After importing the configuration data of the station from the STEP 7 V5 project, the partner configuration data is displayed.

For the individual partner stations, only the two following parameters can be configured:

- Report partner status (connection to partner)
- PLC tag for partner status

The remaining parameters are only displayed. In the text file of the configuration data you will find the parameters in the SDBs "Subscriber data" and "Ethernet data".

Table 10-105 "Partner stations" parameter group

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
Subscriber x			
Subscriber ID			Identifier for the subscriber type: 0 = CPU 1 = TIM
Subscriber number	Subscriber number	SINAUT configuration tool > Subscriber administration	
Subscriber additional identifier			Automatically assigned identifier
Subscriber type			Module type of the STEP 7 V5 configuration
Report partner status (connection to partner)	- -		If the option is enabled, the CP signals the status of the connection to the remote communications partner to the CPU.
PLC tag for partner status	- -		PLC tag for the partner status
Connection to partner			
Interface identifier			Automatically assigned identifier
Subnet of the partner			Automatically assigned identifier

Parameter in STEP 7 Basic / Professional	Parameter in STEP 7 V5	Configuration in STEP 7 V5	Remarks
CFB reference			Automatically assigned identifier
Connection type			Automatically assigned identifier
WAN address	WAN address	Properties WAN network nodes > "Network Attachment" tab	WAN-address (Station address) of the partner
Partner			
Partner subscriber number	Subscriber number of the partner:	SINAUT configuration tool > Subscriber administration	

Download from device

Restrictions when downloading from a CP 1243-8 IRC

When you download the configuration data from a CP 1243-8 IRC using the online functions, the following data will not be fully downloaded:

- Partner station data
- Configuration and partner data of the data points
- References to PLC tags related to communication partners

In order to use the STEP 7 project after a complete download from the devices, you need to download the original export files from STEP 7 V5 with the configuration data of proxies into each CP 1243-8 IRC (Import partner configuration).

10.1.4.5 SCALANCE X, W and M

Configuring SCALANCE X / W / M

Legal notice

Qualified personnel

The product/system belonging to this documentation may only be handled by qualified personnel for the intended purpose taking into account the documentation relating to the intended purpose and, in particular, the safety and warning notices it contains. Due to training and experience, **qualified personnel** is capable of recognizing risks and avoiding possible dangers when handling these products/systems.

Configuring a SCALANCE device as an IO device

With a SCALANCE device configured as a PROFINET IO device and assigned to an IO controller, the functions "Compile" and "Download to device" only download the data to the SCALNCE device that can also be configured in Web Based Management (WBM) (System, Layer 2, Layer 3, Security).

If you want to use the "Compile" or "Download to device" functions for the PROFINET IO device data of the device, first select the assigned IO controller.

Configuring SCALANCE X

System functions

Availability of the system functions

The following table shows the availability of the system functions on the IE switches. Note that all functions are described in the online help. Depending on your IE switch, some functions are not available.

We reserve the right to make technical changes.

		SCALANCE XB-200	SCALANCE XC-200	SCALANCE XP-200	SCALANCE XM-400/ SCALANCE XR-500
Information	ARP table	✓	✓	✓	✓
	Log table	✓	✓	✓	✓
	Ethernet Statistics	✓	✓	✓	✓
	Diagnostics	-	✓ (Temperature)	✓ (Temperature)	✓ (Temperature & power)

		SCALANCE XB-200	SCALANCE XC-200	SCALANCE XP-200	SCALANCE XM-400/ SCALANCE XR-500
System functions	DNS	-	-	-	✓ 1)
	SMTP client	✓	✓	✓	✓
	DHCP client	✓	✓	✓	✓
	DHCP server	✓ (restricted)	✓	✓	✓
	SNMP	✓	✓	✓	✓
	Manual time setting	✓	✓	✓	✓
	DST	-	✓	✓	✓ 1)
	SNTP	✓	✓	✓	✓
	NTP	✓	✓	✓	✓
	SIMATIC Time Client	✓	✓	✓	✓
	PTP	-	-	-	✓ 3)
	Auto logout	✓	✓	✓	✓
	Syslog Client	✓	✓	✓	✓
	Fault monitoring	✓	✓	✓	✓
	PNIO	✓	✓	✓	✓
	EtherNet/IP	✓	✓	✓	-
	PLUG	-	-	✓	✓
	Power over Ethernet	-	-	✓ ("PoE" identifier in the device name)	✓
	Port diagnostics	✓	✓	✓	✓

		SCALANCE XB-200	SCALANCE XC-200	SCALANCE XP-200	SCALANCE XM-400/ SCALANCE XR-500
Layer 2 functions	NFC	-	✓	-	✓ 4)
	Sending priorities	-	✓	✓	
	CoS assignment	✓	✓	✓	✓
	DSCP assignment	✓	✓	✓	✓
	QoS prioritization	✓	✓	✓	✓
	CoS port reassignment	-	✓	✓	
	Load control	✓	✓	✓	✓
	GVRP	-	✓	✓	✓
	Port-based VLAN	✓	✓	✓	✓
	Protocol-based VLAN group	-	-	-	✓
	Protocol-based VLAN port	-	-	-	✓
	IPv4 subnet-based VLAN	-	-	-	✓
	Port-based mirroring	✓	✓	✓	✓
	RSPAN	-	-	-	✓
	VLAN-based mirroring	-	-	-	✓
	MAC flow mirroring	-	-	-	✓
	IP flow mirroring	-	-	-	✓
	Dynamic MAC aging	✓	✓	✓	✓
	Ring redundancy	✓	✓	✓	✓
	Standby	✓	✓	✓	✓
	Observer	-	✓	✓	✓ 1)
	Spanning Tree	✓	✓	✓	✓
	RSTP	✓	✓	✓	✓
	MSTP	-	✓	✓	✓
	Enhanced Passive Listening Compatibility	✓	✓	✓	✓
	Ring with RSTP	-	-	-	✓ 2)
	Loop detection	✓	✓	✓	✓
	Link aggregation	-	✓	✓	✓
	DCP forwarding	✓	✓	✓	✓
	LLDP	✓	✓	✓	✓
	Unicast filter	✓	✓	✓	✓
	Locked ports	✓	✓	✓	✓
	Unicast learning	✓	✓	✓	✓
	Unicast blocking	✓	✓	✓	✓
	Multicast groups	✓	✓	✓	✓
	IGMP	✓	✓	✓	✓
	GMRP	-	✓	✓	✓
	Multicast blocking	✓	✓	✓	✓
	Broadcast blocking	✓	✓	✓	✓
	RMON	✓	✓	✓	✓

		SCALANCE XB-200	SCALANCE XC-200	SCALANCE XP-200	SCALANCE XM-400/ SCALANCE XR-500
	RMON history	-	✓	✓	✓
Layer 3 functions		-	-	-	✓ 5)
Security functions	Passwords	✓	✓	✓	✓
	RADIUS authentication	✓	✓	✓	✓
	MAC authentication	-	✓	✓	✓ 1)
	Guest VLAN	-	✓	✓	✓ 1)
	802.1X reauthentication	✓	✓	✓	✓
	MAC ACL	-	-	-	✓
	IP ACL	-	-	-	✓
	Management ACL	✓	✓	✓	✓

1) as of firmware version V4.0

2) as of firmware version V4.1

3) only SCALANCE XR528-6M and SCALANCE XR552-12M as of firmware version 5.0

4) only SCALANCE XM-400

5) integrated or using a KEY-PLUG

Availability of hardware

The following table shows the hardware of the IE switches.

We reserve the right to make technical changes.

	SCALANCE XB-200	SCALANCE XC-200	SCALANCE XP-200	SCALANCE XM-400/ SCALANCE XR-500
Modularity	-	-	-	✓ 1)
C-PLUG	-	✓	✓	✓
KEY-PLUG	-	-	-	✓
Combo ports	-	-	-	✓
Pluggable transceiver slots	-	✓	-	✓
SELECT/SET button	-	✓	✓	✓
RESET button	✓	-	✓	-
Signaling contact	-	✓	✓	✓
Serial interface	✓	✓	✓	✓
Out-of-band interface	-	-	-	✓
Display modes	-	✓	✓	✓

1) SCALANCE XM-400 via extender, SCALANCE XR-500 via media modules

Configuration limits

Configuration limits of the device

The following table lists the configuration limits for Web Based Management and the Command Line Interface of the devices.

The usability of various functions depends on the device type and whether or not a KEY-PLUG is inserted.

	Configurable function	Maximum number			
		SCALANCE XB-200	SCALANCE XC-200	SCALANCE XP-200	SCALANCE XM-400/ SCALANCE XR-500
System	Maximum frame size (ingress)	1632 bytes	1632 bytes	1632 bytes	9194 Byte (configurable)
	Syslog server	3	3	3	3
	E-mail server	3	3	3	3
	DHCP pools	16 [1)]	24	24	24
	IPv4 addresses per DHCP pool	1	24	24	24
	IPv4 addresses managed by the DHCP server (dynamic + static)	16 [1)]	575	575	575
	DHCP static assignments per DHCP pool	-	24	24	24
	SNMP trap recipient	10	10	10	10
	SNTP server	1	1	1	3
	NTP server	1	1	1	3
	Agent/TIA interfaces [2)]	1	1	1	1

	Configurable function	Maximum number			
		SCALANCE XB-200	SCALANCE XC-200	SCALANCE XP-200	SCALANCE XM-400/ SCALANCE XR-500
Layer 2	Virtual LANs (port-based; including VLAN 1)	17	257	257	257
	Protocol-based VLAN groups per port	-	-	-	12
	IPv4 subnet-based VLANs	-	-	-	150
	Private VLAN	-	-	-	1
	Primary PVLANs	-	-	-	1
	Secondary isolated PVLANs	-	-	-	24
	Secondary community PVLANs	-	-	-	256
	Mirroring sessions	1	1	1	7
	Multiple Spanning Tree instances	-	4	4	16
	Link aggregations or Etherchannels each with a maximum of 8 ports per aggregation	-	8	8	8
	Ports in a link aggregation	-	8	8	8
	Static MAC addresses in the FDB (Forward Database)	128	128	128	256
	Multicast addresses without active GMRP	256	512	512	512
	Multicast addresses with active GMRP	-	50	50	50
	VLANs whose data traffic can be mirrored to a monitor port	-	-	-	255
	RSPAN sessions	-	-	-	1

	Configurable function	Maximum number			
		SCALANCE XB-200	SCALANCE XC-200	SCALANCE XP-200	SCALANCE XM-400/ SCALANCE XR-500
Layer 3	IP interfaces	1	1	1	127
	Entries in the hardware routing table	-	-	-	4096
	Static routes	-	-	-	100
	Possible routes to the same destination	-	-	-	8
	DHCP relay agent interfaces	1	1	1	127
	DHCP Relay Agent servers	4	4	4	4
	NAT interfaces	-	-	-	5
	VRRP router interfaces (only VLAN interfaces)	-	-	-	52
	OSPF areas per device	-	-	-	5
	OSPF area range entries per OSPF area (intra-area summary)	-	-	-	3
	OSPF interfaces	-	-	-	40
	OSPF interfaces per OSPF area	-	-	-	40
	OSPF virtual links (within an autonomous system)	-	-	-	8
	OSPF interface authentication keys	-	-	-	200 (40 interfaces each with 5 keys)
	OSPF virtual link authentication key	-	-	-	40 (8 virtual links each with 5 keys)
	PIM components	-	-	-	1
	Rendezvous points	-	-	-	3
	Candidates for rendezvous points	-	-	-	3
	Static rendezvous points	-	-	-	3
Security	Roles	1	1	1	29
	Users	1	1	1	28
	Groups	-	-	-	32
	IP addresses from RADIUS servers	4	4	4	4
	Management ACLs (access rules for management)	10	10	10	10
	Rules for port ACL MAC	-	-	-	128
	Ingress and egress rules for port ACL MAC (total)	-	-	-	312
	Rules for port ACL IP	-	-	-	128
	Ingress and egress rules for port ACL IP (total)	-	-	-	312
	Rules for VLAN ACL IP	-	-	-	128

[1] With the SCALANCE XB-200, the number of DHCP pools and manageable IPv4 addresses depends on the number of ports. The number of ports corresponds to the maximum number of DHCP pools and manageable IPv4 addresses.

[2] This is an IP interface.

Useful information

IP addresses

IPv4 / IPv6

What are the essential differences?

	IPv4	IPv6
IP configuration	• DHCP server • Manual	• Stateless Address Autoconfiguration (SLAAC): Stateless autoconfiguration using NDP (Neighbor Discovery Protocol) – Creates a link local address for every interface that does not require a router on the link. – Checks the uniqueness of the address on the link that requires no router on the link. – Specifies whether the global addresses are obtained via a status-free mechanism, a mechanism with status or via both mechanisms. (Requires a router on the link.) • Manual • DHCPv6 (status dependent)
Available IP addresses	32-bit: 4, 29 * 10^9 addresses	128-bit: 3, 4 * 10^{38} addresses
Address format	Decimal: 192.168.1.1 with port: 192.168.1.1:20	Hexadecimal: 2a00:ad80::0123 with port: [2a00:ad80::0123]:20
Loopback	127.0.0.1	::1
IP addresses of the interface	4 IP addresses	Multiple IP addresses • LLA: A link local address (formed automatically) fe80::/128 per interface • ULA: Several unique local unicast addresses per interface • GUA: Several global unicast addresses per interface
Header	• Checksum • Variable length • Fragmentation in the header • No security	• Checking at a higher layer • Fixed size • Fragmentation in the extension header
Fragmentation	Host and router	Only endpoint of the communication

	IPv4	IPv6
Quality of service	Type of Service (ToS) for prioritization	The prioritization is specified in the header field "Traffic Class".
Types of frame	Broadcast, multicast, unicast	Multicast, unicast, anycast
Identification of DHCP clients/ server	Client ID: MAC address	DUID + IAID(s) = exactly one interface of the host DUID = DHCP unique identifier Identifies server and clients uniquely and should not change, not even when replacing network components! IAID = Identity Association Identifier At least one per interface is generated by the client and remains unchanged when the DHCP client restarts Three methods of obtaining the DUID • DUID-LLT • DUID-EN • DUID-LL
DHCP	via UDP with broadcast	via UDP with unicast RFC 3315, RFC 3363 **Stateful DHCPv6** Status-dependent configuration in which the IPv6 address and the configuration settings are transferred. Four DHVPv6 messages are exchanged between client and server: 1. SOLICIT: Sent by the DHCPv6 client to localize DHCPv6 servers. 2. ADVERTISE The available DHCPv6 servers reply to this. 3. REQUEST The DHCPv6 client requests an IPv6 address and the configuration settings from the DHCPv6 server. 4. REPLY The DHCPv6 server sends the IPv6 address and the configuration settings. If the client and server support the function "Rapid commit" the procedure is shortened to two DHCPv6 messages SOLICIT and REPLY . **Stateless autoconfiguration** In stateless DHCPv6, only the configuration settings are transferred. **Prefix delegation** The DHCPv6 server delegates the distribution of IPv6 prefixes to the DHCPv6 client. The DHCPv6 client is also known as PD router.
Resolution of IP addresses in hardware addresses	ARP (Address Resolution Protocol)	NDP (Neighbor Discovery Protocol)

IPv4 address

Structure of an IPv4 address

Address classes

IP address range	Max. number of networks	Max. number of hosts/network	Class	CIDR
1.x.x.x through 126.x.x.x	126	16777214	A	/8
128.0.x.x through 191.255.x.x	16383	65534	B	/16
192.0.0.x through 223.255.255.x	2097151	254	C	/24
224.0.0.0 - 239.255.255.255	Multicast applications		D	
240.0.0.0 - 255.255.255.255	Reserved for future applications		E	

An IP address consists of 4 bytes. Each byte is represented in decimal, with a dot separating it from the previous one. This results in the following structure, where XXX stands for a number between 0 and 255:

XXX.XXX.XXX.XXX

The IP address is made up of two parts, the network ID and the host ID. This allows different subnets to be created. Depending on the bytes of the IP address used as the network ID and those used for the host ID, the IP address can be assigned to a specific address class.

Subnet mask

The bits of the host ID can be used to create subnets. The leading bits represent the address of the subnet and the remaining bits the address of the host in the subnet.

A subnet is defined by the subnet mask. The structure of the subnet mask corresponds to that of an IP address. If a "1" is used at a bit position in the subnet mask, the bit belongs to the corresponding position in the IP address of the subnet address, otherwise to the address of the computer.

Example of a class B network:

The standard subnet address for class B networks is 255.255.0.0; in other words, the last two bytes are available for defining a subnet. If 16 subnets must be defined, the third byte of the subnet address must be set to 11110000 (binary notation). In this case, this results in the subnet mask 255.255.240.0.

To find out whether two IP addresses belong to the same subnet, the two IP addresses and the subnet mask are ANDed bit by bit. If both logic operations have the save result, both IP addresses belong to the same subnet, for example, 141.120.246.210 and 141.120.252.108.

Outside the local area network, the distinction between network ID and host ID is of no significance, in this case packets are delivered based on the entire IP address.

Note

In the bit representation of the subnet mask, the "ones" must be set left-justified; in other words, there must be no "zeros" between the "ones".

IPv6 addresses

IPv6 terms

Network node

A network node is a device that is connected to one or more networks via one or more interfaces.

Router

A network node that forwards IPv6 packets.

Host

A network node that represents an end point for IPv6 communication relations.

Link

A link is, according to IPv6 terminology, a direct layer 3 connection within an IPv6 network.

Neighbor

A network node located on the same link as the network node.

IPv6 interface

Physical or logical interface on which IPv6 is activated.

Path MTU

Maximum permitted packet size on a path from a sender to a recipient.

Path MTU discovery

Mechanism for determining the maximum permitted packet size along the entire path from a sender to a recipient.

LLA

Link local address FE80::/10

As soon as IPv6 is activated on the interface, a link local address is formed automatically. Can only be reached by accounts located on the same link.

ULA

Unique Local Address

Defined in RFC 4193. Via this address, the IPv6 interface can be reached in the LAN.

GUA

Global Unicast Address Via this address, the IPv6 interface can be reached, e.g. via the Internet.

Interface ID

The interface ID is formed with the EUI-64 method or manually.

EUI-64

Extended Unique Identifier (RFC 4291); method for forming the interface ID. In Ethernet, the interface ID is formed from the MAC address of the interface. Divides the MAC address into

the manufacturer-specific part (OUI) and the network-specific part (NIC) and inserts FFFE between the two parts.

Example:

MAC address = AA:BB:CC:DD:EE:FF

OUI = AA:BB:CC

NIC = DD:EE:FF

EUI-64 = OUI + **FFFE** + NIC = AA:BB:CC:**FF:FE**:DD:EE:FF

Scope

Defines the range of the IPv6 address.

Structure of an IPv6 address

IPv6 address format - notation

IPv6 addresses consist of 8 fields each with four-character hexadecimal numbers (128 bits in total). The fields are separated by a colon.

Example:

fd00:0000:0000:ffff:02d1:7d01:0000:8f21

Rules / simplifications:

- If one or more fields have the value 0, a shortened notation is possible.
  The address fd00:**0000:0000**:ffff:02d1:7d01:0000:8f21 can also be shortened and written as follows:
  fd00**::**ffff:02d1:7d01:0000:8f21
  To ensure uniqueness, this shortened form can only be used once within the entire address.
- Leading zeros within a field can be omitted.
  The address fd00:0000:0000:ffff:**02d1**:7d01:0000:8f21 can also be shortened and written as follows:
  fd00**::**ffff:**2d1**:7d01:0000:8f21
- Decimal notation with periods
  The last 2 fields or 4 bytes can be written in the normal decimal notation with periods.
  Example: The IPv6 address fd00::ffff.125.1.0.1 is equivalent to fd00::ffff:7d01:1

Structure of the IPv6 address

The IPv6 protocol distinguishes three types of address: Unicast , anycast and multicast. The following section describes the structure of the global unicast addresses.

IPv6 prefix		Suffix
Global prefix: n bits	Subnet ID m bits	Interface ID 128 - n - m bits
Assigned address range	Description of the location, also subnet prefix or subnet	Unique assignment of the host in the network. The ID is generated from the MAC address.

The prefix for the link local address is always fe80:0000:0000:0000. The prefix is shortened and noted as follows: fe80::

IPv6 prefix

Specified in: RFC 4291

The IPv6 prefix represents the subnet identifier.

Prefixes and IPv6 addresses are specified in the same way as with the CIDR notation (Classless Inter-Domain Routing) for IPv4.

Design

IPv6 address / prefix length

Example

IPv6 address: 2001:0db8:1234::1111/48

Prefix: 2001:0db8:1234::/48

Interface ID: ::1111

Entry and appearance

The entry of IPv6 addresses is possible in the notations described above. IPv6 addresses are always shown in the hexadecimal notation.

PROFINET

PROFINET

PROFINET is an open standard (IEC 61158/61784) for industrial automation based on Industrial Ethernet. PROFINET uses existing IT standards and allows end-to-end communication from the field level to the management level as well as plant-wide engineering. PROFINET also has the following features:

- Use of TCP/IP
- Automation of applications with real-time requirements
  - Real-Time (RT) communication
  - Isochronous Real-Time (IRT) communication
- Seamless integration of fieldbus systems

You configure PROFINET in "System > PROFINET (Page 1374)".

PROFINET IO

Within the framework of PROFINET, PROFINET IO is a communications concept for implementing modular, distributed applications. PROFINET IO is implemented by the PROFINET standard for programmable controllers (IEC 61158-x-10).

EtherNet/IP

EtherNet/IP

EtherNet/IP (Ethernet/Industrial Protocol) is an open industry standard for industrial real-time Ethernet based on TCP/IP and UDP/IP. With EtherNet/IP, Ethernet is expanded by the Common Industrial Protocol (CIP) at the application layer. In EtherNet/IP, the lower layers of the OSI reference model are adopted by Ethernet with the physical, network and transport functions.

You configure EtherNet/IP in "System > EtherNet/IP (Page 1375)".

Common Industrial Protocol

The Common Industrial Protocol (CIP) is an application protocol for automation that supports transition of the field buses in Industrial Ethernet and in IP networks. This industry protocol is used by field buses/industrial networks such as DeviceNet, ControlNet and EtherNet/IP at the application layer as an interface between the deterministic fieldbus world and the automation application (controller, I/O, HMI, OPC, ...). The CIP is located above the transport layer and expands the pure transport services with communications services for automation engineering. These include services for cyclic, time-critical and event-controlled data traffic. CIP distinguishes between time-critical I/O messages (implicit messages) and individual query/response frames for configuration and data acquisition (explicit messages). CIP is object-oriented; all data "visible" from the outside is accessible in the form of objects. CIP has a common configuration basis: EDS (Electronic Data Sheet).

Electronic Data Sheet

Electronic Data Sheet (EDS) is an electronic datasheet for describing devices.

The EDS required for EtherNet/IP operation can be found in "System > Load&Save".

VLAN

VLAN

Network definition regardless of the spatial location of the nodes

VLAN (Virtual Local Area Network) divides a physical network into several logical networks that are shielded from each other. Here, devices are grouped together to form logical groups. Only nodes of the same VLAN can address each other. Since multicast and broadcast frames are only forwarded within the particular VLAN, they are also known as broadcast domains.

The particular advantage of VLANs is the reduced network load for the nodes and network segments of other VLANs.

To identify which packet belongs to which VLAN, the frame is expanded by 4 bytes. This expansion includes not only the VLAN ID but also priority information.

Options for the VLAN assignment

There are various options for the assignment to VLANs:

- Port-based VLAN
  Each port of a device is assigned a VLAN ID. You configure port-based VLAN in "Layer 2 > VLAN > Port-based VLAN".
- Protocol-based VLAN
  Each port of a device is assigned a protocol group. You configure protocol-based VLAN in "Layer 2 > VLAN > Protocol-based VLAN port".
- IPv4 Subnet-based VLAN
  The IPv4 address of the device is assigned a VLAN ID. You configure subnet-based VLAN in "Layer 2 > VLAN > IPv4 subnet-based VLAN".

processing the VLAN assignment

If more than one VLAN assignment is created on the device, the assignments are processed in the following order:

1. IPv4 subnet-based VLAN
2. Protocol-based VLAN
3. Port-based VLAN

The frame is first examined for the IPv4 address. If a rule on the "IPv4 subnet-based VLAN" tab applies, the frame is sent to the corresponding VLAN. If no rule applies, the protocol type of the frame is examined. If a rule on the "Protocol-based VLAN port" tab applies, the frame

is sent to the corresponding VLAN. If no rule applies, the frame is sent via the port-based VLAN. The rules for the port-based VLAN are specified on the "Port-based VLAN" tab.

See also

VLAN tagging (Page 1244)

General (Page 1394)

GVRP (Page 1397)

Port-based VLAN (Page 1398)

Protocol-based VLAN group (Page 1399)

Protocol-based VLAN port (Page 1400)

IPv4 subnet-based VLAN (Page 1400)

VLAN tagging

Expansion of the Ethernet frames by four bytes

For CoS (Class of Service, frame priority) and VLAN (virtual network), the IEEE 802.1Q standard defined the expansion of Ethernet frames by adding the VLAN tag.

Note

The VLAN tag increases the permitted total length of the frame from 1518 to 1522 bytes. The end nodes on the networks must be checked to find out whether they can process this length / this frame type. If this is not the case, only frames of the standard length may be sent to these nodes.

The additional 4 bytes are located in the header of the Ethernet frame between the source address and the Ethernet type / length field:

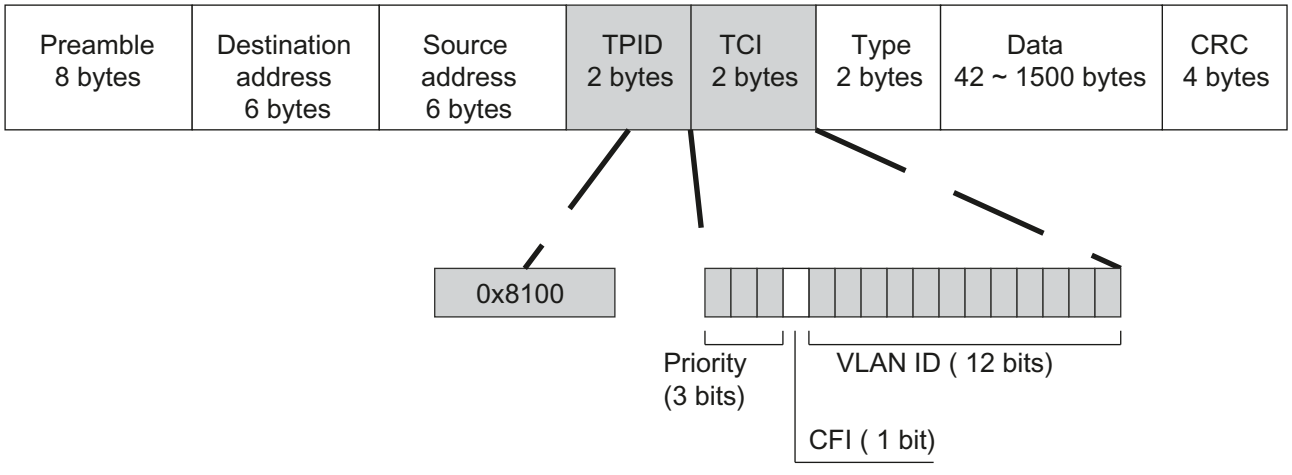


Figure 10-14 Structure of the expanded Ethernet frame

The additional bytes contain the tag protocol identifier (TPID) and the tag control information (TCI).

Tag protocol identifier (TPID)

The first 2 bytes form the Tag Protocol Identifier (TPID) and always have the value 0x8100. This value specifies that the data packet contains VLAN information or priority information.

Tag Control Information (TCI)

The 2 bytes of the Tag Control Information (TCI) contain the following information:

QoS Trust

The tagged frame has 3 bits for the priority that is also known as Class of Service (CoS), see also IEEE 802.1Q.

CoS bits	Priority	Type of the data traffic
000	0 (lowest)	Background
001	1	Best Effort
010	2	Excellent Effort
011	3	Critical Applications
100	4	Video, < 100 ms delay (latency and jitter)
101	5	Voice (language), < 10 ms delay (latency and jitter)
110	6	Internetwork Control
111	7 (highest)	Network Control

The prioritization of the data packets is possible only if there is a queue in the components in which they can buffer data packets with lower priority.

The device has multiple parallel queues in which the frames with different priorities can be processed. As default, first, the frames with the highest priority are processed. This method ensures that the frames with the highest priority are sent even if there is heavy data traffic.

Canonical Format Identifier (CFI)

The CFI is required for compatibility between Ethernet and the token Ring.
The values have the following meaning:

Value	Meaning
0	The format of the MAC address is canonical. In the canonical representation of the MAC address, the least significant bit is transferred first. Standard-setting for Ethernet switches.
1	The format of the MAC address is not canonical.

VLAN ID

In the 12-bit data field, up to 4096 VLAN IDs can be formed. The following conventions apply:

VLAN ID	Meaning
0	The frame contains only priority information (priority tagged frames) and no valid VLAN identifier.
1- 4094	Valid VLAN identifier, the frame is assigned to a VLAN and can also include priority information.
4095	Reserved

Private VLAN

With a private VLAN (PVLAN) you can divide up the layer 2 broadcast domains of a VLAN.

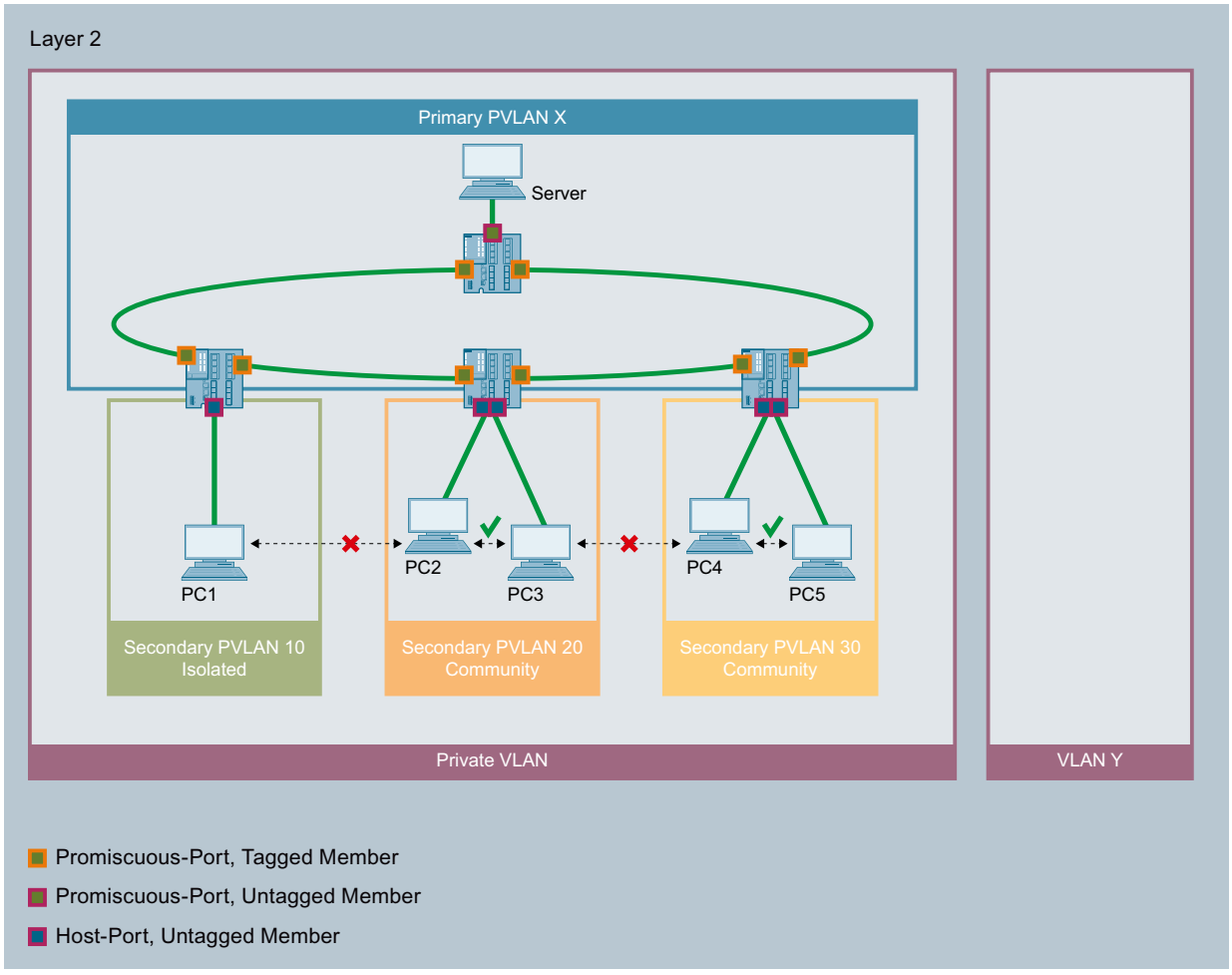
A private VLAN consists of the following units:

- A primary private VLAN (primary PVLAN)
  The VLAN that is divided up is called primary private VLAN.
- secondary private VLANs (secondary PVLAN)
  Secondary PVLANs exist only within a primary PVLAN. Every secondary PVLAN has a specific VLAN ID and is connected to the primary PVLAN.
  Secondary PVLANs are divided into the following types:
  - Isolated Secondary PVLAN
    Devices within an isolated secondary PVLAN cannot communicate with each other via layer 2.
  - Community Secondary PVLAN
    Devices within a community secondary PVLAN can communicate with each other directly via layer 2. The devices cannot communicate with devices in other communities of the PVLAN via layer 2.

Note

VALAN ID with secondary PVLANs

If you use the same VLAN ID for secondary PVLANs on different IE switches, the end devices in these secondary PVLANs can communicate with other via layer 2 across the different switches.



In this example, the ports of the IE switches that connect them to other IE switches are promiscuous ports. These network ports are tagged members in all PVLANs: Primary PVLAN and all secondary PVLANs.

The ports to which the PCs are connected are host ports. The host ports are all untagged members in the primary PVLAN and in their secondary PVLAN.

The port to which the server is connected is a promiscuous port. This promiscuous port ports is an untagged member in all PVLANs: Primary PVLAN and all secondary PVLANs.

In this example all PCs can communicate with the server. The server can communicate with all PCs. PC1 cannot communicate with any other PC. The PCs within a community secondary PVLAN can communicate with each other but not with the PCs in another secondary PVLAN.

SNMP

Introduction

With the aid of the Simple Network Management Protocol (SNMP), you monitor and control network components from a central station, for example routers or switches. SNMP controls the communication between the monitored devices and the monitoring station.

Tasks of SNMP:

- Monitoring of network components
- Remote control and remote parameter assignment of network components
- Error detection and error notification

In versions v1 and v2c, SNMP has no security mechanisms. Each user in the network can access data and also change parameter assignments using suitable software.

For the simple control of access rights without security aspects, community strings are used.

The community string is transferred along with the query. If the community string is correct, the SNMP agent responds and sends the requested data. If the community string is not correct, the SNMP agent discards the query. Define different community strings for read and write permissions. The community strings are transferred in plain text.

Standard values of the community strings:

- public
  has only read permissions
- private
  has read and write permissions

Note

Because the SNMP community strings are used for access protection, do not use the standard values "public" or "private". Change these values following the initial commissioning.

Further simple protection mechanisms at the device level:

- Allowed Host
  The IP addresses of the monitoring systems are known to the monitored system.
- Read Only
  If you assign "Read Only" to a monitored device, monitoring stations can only read out data but cannot modify it.

SNMP data packets are not encrypted and can easily be read by others.

The central station is also known as the management station. An SNMP agent is installed on the devices to be monitored with which the management station exchanges data.

The management station sends data packets of the following type:

- GET
  Request for a data record from the SNMP agent
- GETNEXT
  Calls up the next data record.
- GETBULK (available as of SNMPv2c)
  Requests multiple data records at one time, for example several rows of a table.
- SET
  Contains parameter assignment data for the relevant device.

The SNMP agent sends data packets of the following type:

- RESPONSE
  The SNMP agent returns the data requested by the manager.
- TRAP
  If a certain event occurs, the SNMP agent itself sends traps.

SNMPv1/v2c/v3 use UDP (User Datagram Protocol) and use the UDP ports 161 and 162. The data is described in a Management Information Base (MIB).

SNMPv3

Compared with the previous versions SNMPv1 and SNMPv2c, SNMPv3 introduces an extensive security concept.

SNMPv3 supports:

- Fully encrypted user authentication
- Encryption of the entire data traffic
- Access control of the MIB objects at the user/group level

With the introduction of SNMPv3 you can no longer transfer user configurations to other devices without taking special action, e.g. by loading a configuration file or replacing the C-PLUG.

According to the standard, the SNMPv3 protocol uses a unique SNMP engine ID as an internal identifier for an SNMP agent. This ID must be unique in the network. It is used to authenticate access data of SNMPv3 users and to encrypt it.

Depending on whether you have enabled or disabled the “SNMPv3 User Migration” function, the SNMP engine ID is generated differently.

Restriction when using the function

Use the "SNMPv3 User Migration" function only to transfer configured SNMPv3 users to a substitute device when replacing a device.
Do not use the function to transfer configured SNMPv3 users to multiple devices. If you load a configuration with created SNMPv3 users on several devices, these devices use the same SNMP engine ID. If you use these devices in the same network, your configuration contradicts the SNMP standard.

Compatibility with predecessor products

You can only transfer SNMPv3 users to a different device if you have created the users as migratable users. To create a migratable user the "SNMPv3 User Migration" function must be activated when you create the user.

Redundancy mode

Spanning Tree

Avoiding loops on redundant connections

The spanning tree algorithm allows network structures to be created in which there are several connections between two IE switches / bridges. Spanning tree prevents loops being formed in the network by allowing only one path and disabling the other (redundant) ports for data traffic. If there is an interruption, the data can be sent over an alternative path. The functionality of the spanning tree algorithm is based on the exchange of configuration and topology change frames.

Definition of the network topology using the configuration frames

The devices exchange configuration frames known as BPDUs (Bridge Protocol Data Units) with each other to calculate the topology. The root bridge is selected and the network topology created using these frames. BPDUs also bring about the status change of the root ports.

The root bridge is the bridge that controls the spanning tree algorithm for all involved components.

Once the root bridge has been specified, each device sets a root port. The root port is the port with the lowest path costs to the root bridge.

Response to changes in the network topology

If nodes are added to a network or drop out of the network, this can affect the optimum path selection for data packets. To be able to respond to such changes, the root bridge sends configuration messages at regular intervals. The interval between two configuration messages can be set with the "Hello Time" parameter.

Keeping configuration information up to date

With the "Max Age" parameter, you set the maximum age of configuration information. If a bridge has information that is older than the time set in "Max Age", it discards the message and initiates recalculation of the paths.

New configuration data is not used immediately by a bridge but only after the period specified in the "Forward Delay" parameter. This ensures that operation is only started with the new topology after all the bridges have the required information.

RSTP, MSTP, CIST

Rapid Spanning Tree Protocol (RSTP)

One disadvantage of STP is that if there is a disruption or a device fails, the network needs to reconfigure itself: The devices start to negotiate new paths only when the interruption occurs. This can take up to 30 seconds. Fur this reason, STP was expanded to create the "Rapid Spanning Tree Protocol" (RSTP, IEEE 802.1w). This differs from STP essentially in that the devices are already collecting information about alternative routes during normal operation and do not need to gather this information after a disruption has occurred. This means that the reconfiguration time for an RSTP controlled network can be reduced to a few seconds.
This is achieved by using the following functions:

- Edge ports (end node port)
  Edge ports are ports connected to an end device.
  A port that is defined as an edge port is activated immediately after connection establishment. If a spanning tree BPDU is received at an edge port, the port loses its role as edge port and it takes part in (R)STP again. If no further BPDU is received after a certain time has elapsed (3 x hello time), the port returns to the edge port status.
- Point-to-point (direct communication between two neighboring devices)

  By directly linking the devices, a status change (reconfiguration of the ports) can be made without any delays.
- Alternate port (substitute for the root port)

  A substitute for the root port is configured. If the connection to the root bridge is lost, the device can establish a connection over the alternate port without any delay due to reconfiguration.
- Reaction to events

  Rapid spanning tree reacts to events, for example an aborted connection, without delay. There is no waiting for timers as in spanning tree.
- Counter for the maximum bridge hops
  The number of bridge hops a package is allowed to make before it automatically becomes invalid.

In principle, therefore with rapid spanning tree, alternatives for many parameters are preconfigured and certain properties of the network structure taken into account to reduce the reconfiguration time.

Multiple Spanning Tree Protocol (MSTP)

The Multiple Spanning Tree Protocol (MSTP) is a further development of the Rapid Spanning Tree Protocol. Among other things, it provides the option of operating several RSTP instances within different VLANs or VLAN groups and, for example, making paths available within the individual VLANs that the single Rapid Spanning Tree Protocol would globally block.

Common and Internal Spanning Tree (CIST)

CIST identifies the internal instance used by the switch that is comparable in principle with an internal RSTP instance.

HRP

HRP - High Speed Redundancy Protocol

HRP is the name of a redundancy method for networks with a ring topology. The switches are interconnected via ring ports. One of the switches is configured as the redundancy manager (RM). The other switches are redundancy clients. Using test frames, the redundancy manager checks the ring to make sure it is not interrupted. The redundancy manager sends test frames via the ring ports and checks that they are received at the other ring port. The redundancy clients forward the test frames.

If the test frames of the RM no longer arrive at the other ring port due to an interruption, the RM switches through its two ring ports and informs the redundancy clients of the change immediately. The reconfiguration time after an interruption of the ring is a maximum of 0.3 seconds.

Standby redundancy

Standby redundancy is a method with which rings each of which is protected by high-speed redundancy can be linked together redundantly. In the ring, a master/slave device pair is configured and these monitor each other via their ring ports. If a fault occurs, the data traffic is redirected from one Ethernet connection (standby port of the master or standby server) to another Ethernet connection (standby port of the slave).

Requirements

- HRP is supported in ring topologies with up to 50 devices. Exceeding this number of devices can lead to a loss of data traffic.
- For HRP, only devices that support this function can be used in the ring.
- All devices must be interconnected via their ring ports.
- Devices that do not support HRP must be linked to the ring using special devices with HRP capability. Up to the ring, this connection is not redundant.

Standby

General

SCALANCE X switches support not only ring redundancy within a ring but also redundant linking of rings or open network segments (linear bus). In the redundant link, rings are connected together over Ethernet connections. This is achieved by configuring a master/slave device pair in one ring so that the devices monitor each other and, in the event of a fault, redirect the data traffic from the normally used master Ethernet connection to the substitute (slave) Ethernet connection.

Standby redundancy

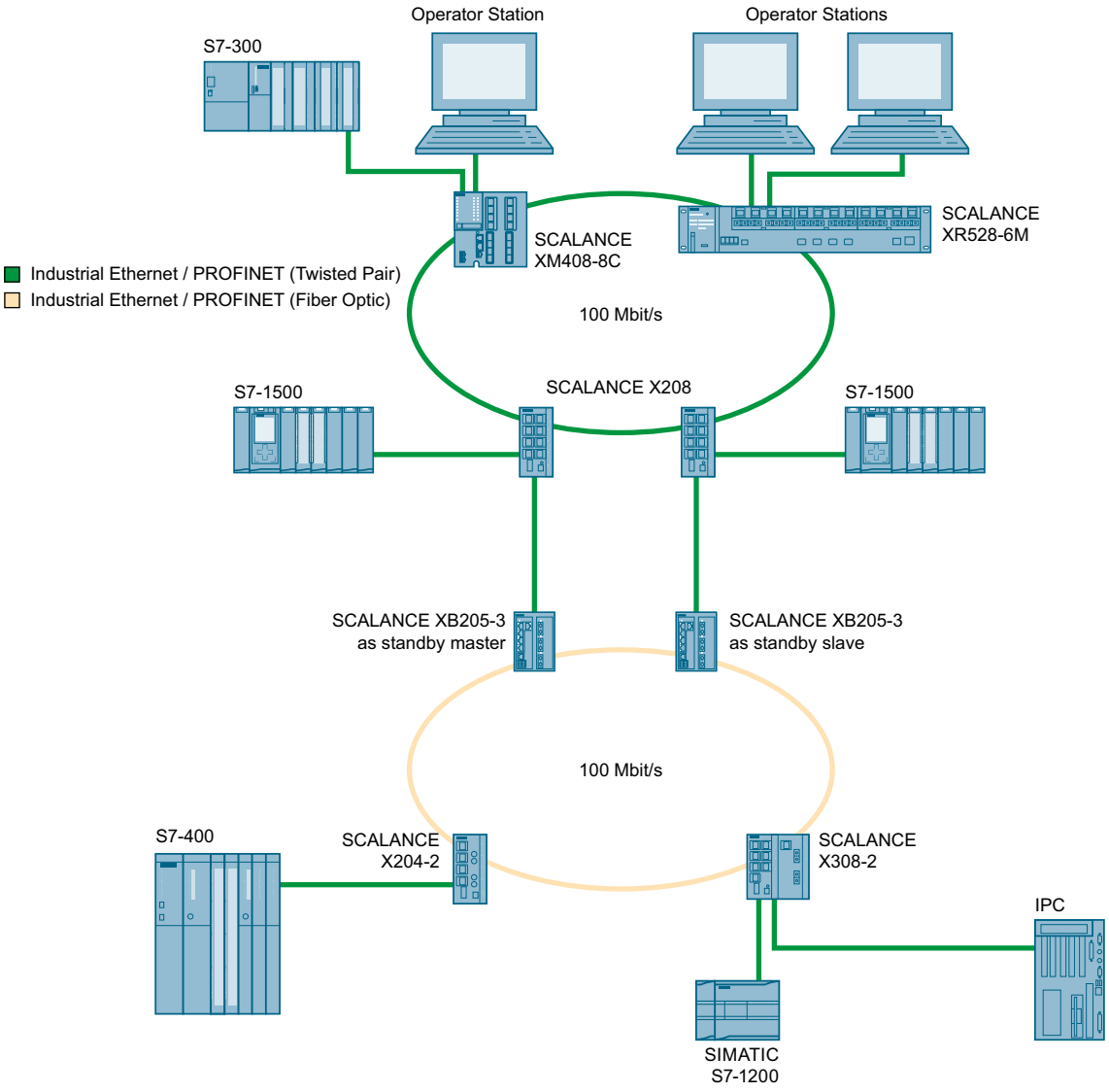


Figure 10-15 Example of a redundant link between rings

For a redundant link as shown in the figure, two devices must be configured as standby redundancy switches within a network segment. In this case, network segments are rings with a redundancy manager. Instead of rings, network segments might also be linear.

The two standby redundancy switches connected in the configuration exchange data frames with each other to synchronize their operating statuses (one device is master and the other slave). If there are no problems, only the link from the master to the other network segment is active. If this link fails (for example due to a link-down or a device failure), the slave activates its link as long as the problem persists.

MRP

MRP - Media Redundancy Protocol

The "MRP" method conforms to the Media Redundancy Protocol (MRP) specified in the following standard:

IEC 62439-2 Release 1.0 (2010-02) Industrial communication networks - High availability automation networks Part 2: Media Redundancy Protocol (MRP)

The reconfiguration time after an interruption of the ring is a maximum of 200 ms.

Topology

The following figure shows a possible topology for devices in a ring with MRP.

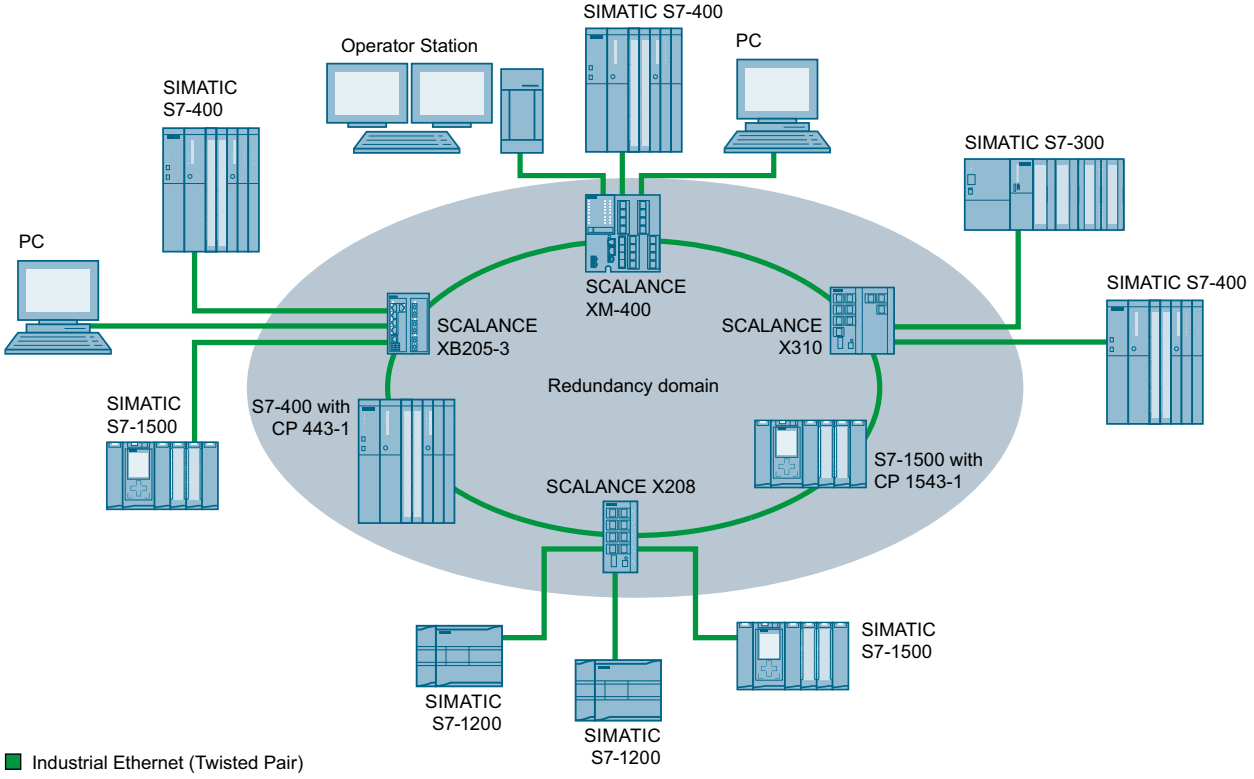


Figure 10-16 Example of a ring topology with the MRP media redundancy protocol

The following rules apply to a ring topology with media redundancy using MRP:

- All the devices connected within the ring topology are members of the same redundancy domain.
- One device in the ring is acting as redundancy manager.
- All other devices in the ring are redundancy clients.

Non MRP-compliant devices can be connected to the ring via a SCALANCE X switch or via a PC with a CP capable of MRP.

Requirements

Requirements for problem-free operation with the MRP media redundancy protocol are as follows:

- MRP is supported in ring topologies with up to 50 devices.
  Exceeding this number of devices can lead to a loss of data traffic.
- The ring in which you want to use MRP may only consist of devices that support this function. These include, for example, some of the Industrial Ethernet SCALANCE X switches, some of the communications processors (CPs) for SIMATIC S7 and PG/PC or non-Siemens devices that support this function.
- All devices must be interconnected via their ring ports.
  Multimode connections up to 3 km and single mode connections up to 26 km between two SCALANCE X IE switches are possible. At greater distances, the specified reconfiguration time may be longer.
- "MRP" must be enabled for all devices in the ring.
- The connection settings (transmission medium / duplex) must be set to full duplex and at least 100 Mbps for all ring ports. Otherwise there may be a loss of data traffic.
  - STEP 7: Set all the ports involved in the ring to "Automatic settings" in the "Options" tab of the properties dialog.
  - WBM: If you configure with Web Based Management, the ring ports are set automatically to autonegotiation.

See also

Configuration in STEP 7 (Page 1256)

Note

Number of devices

Except in PROFINET IO systems, topologies with up to 100 SCALANCE X-200 and SCALANCE X-300 IE switches were tested successfully.

Configuration in WBM

Role

The choice of role depends on the following use cases:

- You want to use MRP in a ring topology only with Siemens devices:
  - For at least one device in the ring select "Automatic Redundancy Detection" or "MRP Auto Manager".
  - For all other devices in the ring select "MRP Client" or "Automatic Redundancy Detection".
- You want to use MRP in a ring topology that also includes non-Siemens devices:
  - For exactly one device in the ring select the role "MRP Auto Manager".
  - For all other devices in the ring topology, select the role of "MRP client".

Note

The use of "Automatic Redundancy Detection" is not possible when using non-Siemens devices.

- You configure the devices in an MRP ring topology partly with WBM and partly with STEP 7:
  - With the devices you configure using WBM, select "MRP Client" for all devices.
  - With the devices that you configure using STEP 7, select precisely one device as "Manager" or "Manager (Auto)" and "MRP Client" for all other devices.

Note

If a device is assigned the role of "Manager" with STEP 7, all other devices in the ring must be assigned the "MRP Client" role. If there is a device with the "Manager" role and a device with the "Manager (Auto)"/"MRP Auto-Manager" in a ring, this can lead to circulating frames and therefore to failure of the network.

Configuration

In WBM, you configure MRP on the following pages:

- AUTOHOTSPOT
- AUTOHOTSPOT

Configuration in STEP 7

Configuration in STEP 7

To create the configuration in STEP 7, select the parameter group "Media redundancy" on the PROFINET interface.

Set the following parameters for the MRP configuration of the device:

- Domain
- Role
- Ring port
- Diagnostic interrupts

These settings are described below.

Note

Valid MRP configuration

In the MRP configuration in STEP 7, make sure that all devices in the ring have a valid MRP configuration before you close the ring. Otherwise, there may be circulating frames that will cause a failure in the network.

One device in the ring needs to be configured as "redundancy manager" and all other devices in the ring as "clients".

Note

Note factory settings

MRP is disabled and spanning tree enabled for the following brand new IE switches and those set to the factory settings:

- SCALANCE XB-200 (Ethernet/IP variants)
- SCALANCE XP-200 (Ethernet/IP variants)
- SCALANCE XM-400
- SCALANCE XR-500

To load a PROFINET configuration into one of the specified devices, first disable spanning tree on the device.

Note

Reconfiguration only when the ring is open

First open the ring before you

- change the MRP role or
- reconfigure ring ports

Note

Starting up and restarting

The MRP settings are still effective after a restart of the device or a power failure and hot restart as long as the power failure does not occur within 90 seconds after the configuration change.

Note

Prioritized startup

If you configure MRP in a ring, you cannot use the "prioritized startup" function in PROFINET applications on the devices involved.

If you want to use the "prioritized startup" function, then disable MRP in the configuration.

In the STEP 7 configuration, set the role of the relevant device to "Not a node in the ring".

Domain

Single MRP rings

If you want to configure a single MRP ring, leave the factory setting "mrpdomain 1" in the "Domain" drop-down list.

All devices configured in a ring with MRP must belong to the same redundancy domain. A device cannot belong to more than one redundancy domain in a single ring.

Multiple MRP rings

If you configure multiple single MRP rings, the nodes of the ring will be assigned to the individual rings with the "Domain" parameter. Set the same domain for all devices within a ring. Set different domains for different rings. Devices that do not belong to the same ring must have different domains.

If you want to configure MRP multiple rings, as the central redundancy manager for up to four rings select a device that is capable of multiple rings. Specify different domains for all ring instances and assign these to the corresponding ring ports of the redundancy manager. Configure the other devices as clients. The same domain must be set for all devices within a ring.

Role

Note

Reconfiguration only when the ring is open!

The choice of role depends on the following use cases.

- You want to use MRP in a topology with **one ring** only with Siemens devices and without monitoring diagnostic interrupts:
  Assign all devices to the "mrpdomain-1" domain and the role "Manager (Auto)".
  The device that actually takes over the role of redundancy manager, is negotiated by Siemens devices automatically.
- You want to use MRP in a topology with **multiple rings** only with Siemens devices and without monitoring diagnostic interrupts (MRP multiple rings):
  - Assign all instances of the device that connects the rings the role of "Manager".
  - For all other devices in the ring topology, select the role of "Client".

- You want to use MRP in a ring topology that also includes non-Siemens devices or you want to receive diagnostic interrupts relating to the MRP status from a device (see "Diagnostic interrupts"):
  - Assign precisely one device in the ring the role of "Manager (Auto)".
  - For all other devices in the ring topology, select the role of "Client".
- You want to disable MRP:
  Select the option "Not node in the ring" if you do not want to operate the device within a ring topology with MRP.

Note

Role after resetting to factory settings

With brand new Siemens devices and those reset to the factory settings the following MRP role is set:

- "Manager (Auto)"
  CPs
- "Automatic Redundancy Detection"
  - SCALANCE X-200
  - SCALANCE XC-200
  - SCALANCE XB-200 (PROFINET variants)
  - SCALANCE XP-200 (PROFINET variants)
  - SCALANCE X-300
  - SCALANCE X-400

If you are operating a non-Siemens device as the redundancy manager in the ring, this may cause loss of the data traffic.

MRP is disabled and spanning tree enabled for the following brand new IE switches and those set to the factory settings:

- SCALANCE XB-200 (Ethernet/IP variants)
- SCALANCE XP-200 (Ethernet/IP variants)
- SCALANCE XM-400
- SCALANCE XR-500

Ring port 1 / ring port 2

Here, select the port you want to configure as ring port 1 and ring port 2.

With devices with more than 8 ports, not all ports can be selected as ring port.

The drop-down list shows the selection of possible ports for each device type. If the ports are specified in the factory, the boxes are grayed out.

NOTICE
Ring ports after resetting to factory settings If you reset to the factory settings, the ring port settings are also reset. If other ports were used previously as ring ports before resetting, with the appropriate attachment, a previously correctly configured device can cause circulating frames and therefore the failure of the data traffic.

Note

Reconfiguration only when the ring is open

First open the ring before you reconfigure the ring ports of a multiple ring manager.

Diagnostic interrupts

Enable the "Diagnostic interrupts" option, if you want diagnostic interrupts relating to the MRP status on the local CPU to be output.

The following diagnostic interrupts can be generated:

- Wiring or port error
  Diagnostic interrupts are generated if the following errors occur at the ring ports:
  - Connection abort on a ring port
  - A neighbor of the ring port does not support MRP.
  - A ring port is connected to a non-ring port.
  - A ring port is connected to the ring port of another MRP domain.
- Status change active/passive (redundancy manager only)
  If the status changes (active/passive) in a ring, a diagnostics interrupt is generated.

Parameter assignment of the redundancy is not set by STEP 7 (redundancy alternatives)

This option only affects SCALANCE X switches. Select this option if you want to set the properties for media redundancy using alternative mechanisms such as WBM, CLI or SNMP.

If you enable this option, existing redundancy settings are retained and are not overwritten. The parameters in the "MRP configuration" box are then reset and grayed out. The entries then have no meaning.

Parallel Redundancy Protocol

Parallel Redundancy Protocol

The "Parallel Redundancy Protocol" (PRP) is a redundancy protocol for Ethernet networks. It is defined in Part 3 of the IEC 62439 standard. This redundancy method allows data communication to be maintained without interruption/reconfiguration time if there are interruptions in the network.

The PRP method is supported, for example, by the devices of the SCALANCE X-200RNA product line.

Overlong frames

When sending PRP frames, the IE switch expands the frame with a PRP trailer. With frames with the maximum length, appending the PRP trailer results in an overlong frame that exceeds the maximum permitted frame length (according to the IEEE 802.3 standard).

To prevent data loss with overlong frames, all network components located in a PRP network must support a frame length of at least 1528 bytes.

The devices described in this manual can be used in PRP networks, see also section "Configuration limits (Page 1233)".

Routing function

Introduction

The term routing describes the specification of routes for communication between different networks; in other words, how does a data packet from subnet A get to subnet B.

SCALANCE X supports the following routing functions:

- Static routing
  With static routing, the routes are entered manually in the routing table.
- Router redundancy
  With standardized VRRP (Virtual Router Redundancy Protocol), the availability of important gateways is increased by redundant routers.
  - VRRPv2 (IPv4)
  - VRRPv3 (IPv4 / IPv6)
- Dynamic routing
  The entries in the routing table are dynamic and are updated continuously. The entries are created with one of the following dynamic routing protocols:
  - OSPFv2 (IPv4)
  - OSPFv3 (IPv6)
  - RIPv2 (IPv4)
  - RIPng (IPv6)

VRRP

VRRPv2

Router redundancy with VRRP

With the Virtual Router Redundancy Protocol (VRRP), the failure of a router in a network can be countered.

VRRP can only be used with virtual IP interfaces (VLAN interfaces) and not with router ports.

Several VRRP routers in a network segment are put together as a logical group representing a virtual router (VR). The group is defined using the virtual ID (VRID). Within the group, the VRID must be the same. The VRID can no longer be used for other groups.

The virtual router is assigned a virtual IP address and a virtual MAC address. One of the VRRP routers within the group is specified as the master router. The master router has priority 255. The other VRRP routers are backup routers. The master router assigns the virtual IP address and the virtual MAC address to its network interface. The master router sends VRRP packets (advertisements) to the backup routers at specific intervals. With the VRRP packets, the master router signals that it is still functioning. The master router also replies to the ARP queries.

If the virtual master router fails, a backup router takes over the role of the master router. The backup router with the highest priority becomes the master router. If the priority of the backup routers is the same, the higher MAC address decides. The backup router becomes the new virtual master router.

The new virtual master router adopts the virtual MAC and IP address. This means that no routing tables or ARP tables need to be updated. The consequences of a device failure are therefore minimized.

You configure VRRP in "Layer 3 (IPv4) > VRRP".

VRRP3

Version 3 of VRRP is based on version 2.

Note

- Enable routing to be able to use VRRPv3.
- You can only use VRRPv3 in conjunction with VLAN interfaces. Router ports are not supported.
- Simultaneous operation of VRRP and VRRPv3 is not possible.
- VRRPv3 supports IPv4 and IPv6. Both can be configured and operated at the same time with VRRP3.

You configure VRRPv3 in:

IPv4: Layer 3 (IPv4)> VRRPv3

IPv6: Layer 3 (IPv6)> VRRPv3

OSPF

OSPFv2

Dynamic routing with OSPFv2

OSPF (Open Shortest Path First) is a cost-based routing protocol. To calculate the shortest and most cost-effective route, the Short Path First algorithm by Dijkstra is used. OSPF was developed by the IETF (Internet Engineering Task Force).

You configure OSPFv2 in "Layer 3 (IPv4) > OSPFv2".

OSPFv2 divides an autonomous system (AS) into different areas.

Areas in OSPF

The following areas exist:

- Backbone
  The backbone area is area 0.0.0.0. All other areas are connected to this area. The backbone area is connected either directly or via virtual connections with other areas.
  All routing information is available in the backbone area. As a result, the backbone area is responsible for forwarding information between different areas.
- Stub Area
  This area contains the routes within its area within the autonomous system and the standard route out of the autonomous system. The destinations outside this autonomous system are assigned to the standard route.
- Totally Stubby Area
  This area knows only the routes within its area and the standard route out of the area.
- Not So Stubby Area (NSSA)
  This area can forward (redistribute) packets from other autonomous systems into the areas of its own autonomous system. The packets are further distributed by the NSSA router.

Routers of OSPF

OSPF distinguishes the following router types:

- Internal router (IR)
  All OSPF interfaces of the router are assigned to the same area.
- Area Border Router (ABR)
  The OSPF interfaces of the router are assigned to different areas. One OSPF interface is assigned to the backbone area. Where possible, routes are grouped together.
- Backbone Router (BR)
  At least one of the OSPF interfaces is assigned to the backbone area.
- Autonomous System Border Router (ASBR)
  One interface of the router is connected to a different AS, for example an AS that uses the routing protocol RIP.

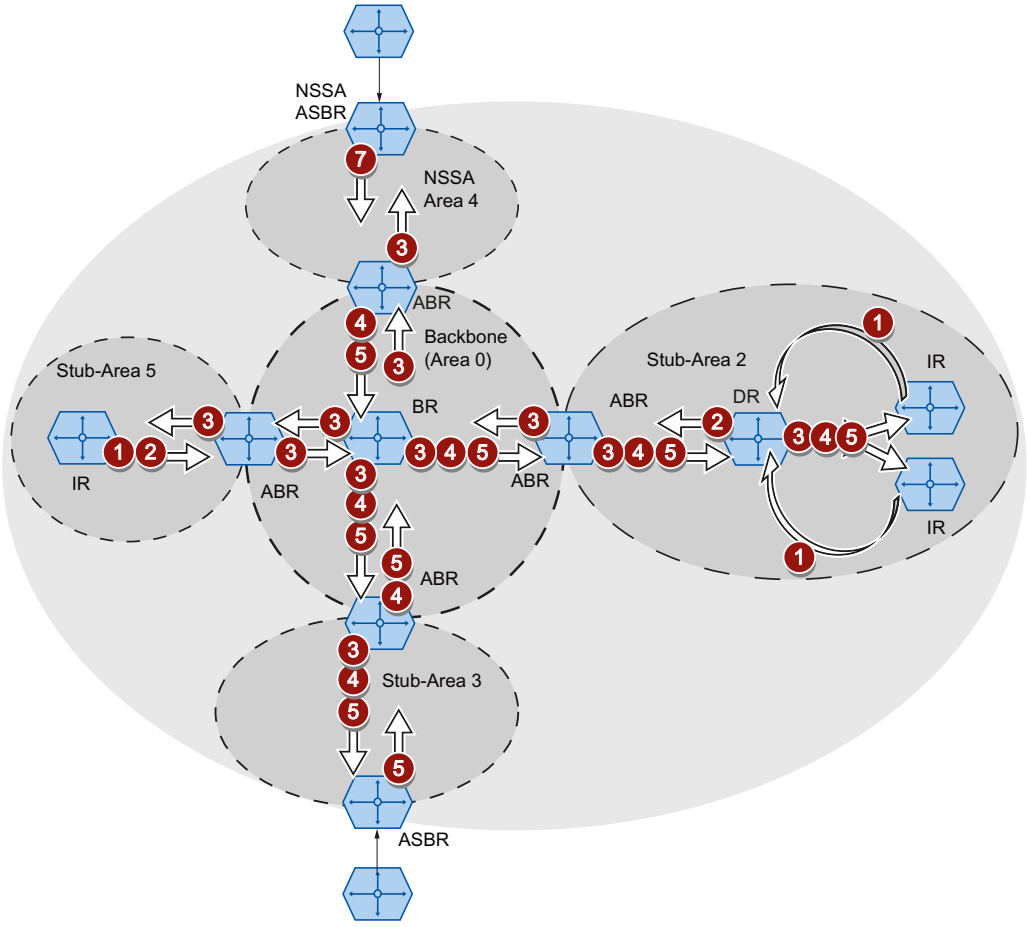
Virtual connection

Each area must be connected to the backbone area. In some situations a direct physical connection is not possible. In this case, a router of the relevant area must be connected to a backbone router via a virtual connection.

LSA types

Within the autonomous system, packets are exchanged that contain information about the connections of a router and the connection status message. The packets are also known as LSAs (Link State Advertisements). The LSAs are always sent from the router to the neighbor router.

If there are changes in the network, LSAs are sent to all routers in the network. The information depends on the LSA type.



1. **Router LSA (LSA Type 1)**
   The LSA Type 1 is only sent within an area. For each active connection of the router that belongs to the area in consideration, an LSA Type 1 is generated. The LSA Type 1 contains information about the status and the costs of the connection, for example IP address, network mask, network type
2. **Network LSA (LSA Type 2)**
   The LSA Type 2 is sent only within an area. For each network that belongs to the relevant area, the router generates an LSA Type 2. If several routers are interconnected in a network, the LSA Type 2 is sent by the designated router (DR). The LSA Type 2 includes the network address, the network mask and a list of routers that are connected to the network
3. **Summary LSA (LSA Type 3 / LSA Type 4)**
   The Summary LSA is generated by the area border router and sent into the area. The
4. Summary LSA contains information about routes outside the area but inside the AS. Where possible, the routes are grouped together.
   - Summary LSA (LSA Type 3)
     The LSA Type 3 describes the routes to the networks and advertises the standard route to the areas.
   - AS Summary LSA (LSA Type 4)
     The LSA Type 4 describes the routes to the ASBR.
5. **External LSA (LSA Type 5 / LSA Type 7)**
   The External LSA is generated by the ASBR. The LSA type depends on the area.
7. - AS External LSA (LSA Type 5)
     The LSA Type 5 is sent by the AS border router into the areas of the autonomous system except the Stub and NSSA areas. The LSA contains information about routes to a network in another AS. The routes are either created manually or learned externally. The ASBR uses LSA Type 5 to distribute standard routes to the backbone area.
   - NSSA External LSA (LSA Type 7)
     The LSA Type 7 is generated by the AS border router of an NSSA. The router is also known as the NSSA ASBR. The LSA Type 7 is sent only within the NSSA. If the P bit in LSA Type 7 = 1, these LSAs are converted to LSA Type 5 by the ABR and sent to the backbone area.

Establishing the neighborhood

The router runs through the following statuses to establish a connection to the neighbor router.

1. Attempt state / Init state
   The router activates OSPF and begins to send and receive Hello packets. Based on the received Hello packets, the router learns which OSPF routers are in its vicinity. The router checks the content of the Hello packet. The Hello packet also contains the list of the neighbor routers (neighbor table) of the "sender".
2. Two way state
   If, for example, the ID of the area, the area type and the settings for the times match, a connection (adjacency) can be established to the neighbor. In a point-to-point network, the connection is established directly. If several neighbor routers can be reached in a network, the designated router (DR) and the designated backup router (DBR) are identified based on Hello packets. The router with the highest router priority becomes the designated router. If two routers have the same router priority, the router with the highest router ID becomes the designated router. The router establishes a connection to the designated router.
3. Exchangestart state
   The neighbor routers decide which router starts communication. The router with the higher router ID becomes the designated router.
4. Exchange state
   The neighbor routers send packets that describe the content of their neighborhood database. The neighborhood database (link state database - LSDB) contains information on the topology of the network.
5. Loading state
   The router completes the received information. If the router still has questions relating to the status of a specific connection, it sends a link state request. The neighbor router sends a response (link state update). The response contains a suitable LSA. The router confirms receipt of the response (link state acknowledge).
6. Full State
   The information exchange with the neighbor router is completed. The neighborhood database of the neighbor router is the same. Based on the Short Path First algorithm, the router calculates a route to every destination. The route is entered in the routing table.

Check the neighborhood

The Hello packets are only used to establish the neighborhood relations. Hello packets are used to check the connection to the neighbor router by sending them cyclically. If no Hello packet is received within a certain interval (dead interval), the connection to the neighbor is marked as "down". The relevant entries are deleted.

Updating the neighborhood database

Once the neighborhood database is established, LSAs are sent to all routers in the network if there are changes in the topology.

OSPFv3

Version 3 of OSPF is based on version 2 and is only used with IPv6. A large part of the routing mechanisms was adopted. OSPFv3 is defined in the RFCs 2740 and 5340.

You configure OSPFv3 under "Layer 3 (IPv6) > OSPFv3".

The following has not changed:

- The statuses that a router runs through to establish a connection to the neighbor router.
- The areas : Backbone, Stub Area, Totally Stubby Area, Not So Stubby Area (NSSA)
- The router types: Internal Router (IR), Area Border Router (ABR), Backbone Router (BR), Autonomous System Area Border Router (ASBR), Designated Router (DR)
- The router ID, the area ID and the ID of the LSA are entered in the IPv4 address format: x.x.x.x

What has changed?

Terms

The terms network or subnet are replaced by link.

Authentication

The authentication was removed. Instead OSPFv3 uses IPsec, that is implemented in IPv6.

Neighbor routers

The neighbor routers are identified via the router ID.

Neighbor database

The neighbor database (link state database - LSDB) is divided into different areas of application:

- Link scope LSDB
  Contains the link LSA
- Area scope LSDB
  contains the following LSAs
  - Router LSA
  - Network LSA
  - Inter-area prefix LSA
  - Inter area router LSA
  - Intra area prefix LSA
- AS scope LSDB
  Contains the AS external LSA

LSA types

Two new LSA types were defined for OSPFv3.

OSPFv2		OSPFv3		Who	Within	Description
1	Router LSA	0x2001	Router LSA	every router	Area	No longer contains address information. This is contained in the new LSA type 2009.
2	Network LSA	0x2002	Network LSA	DR	Area	No longer contains address information. This is contained in the new LSA type 2009.
3	Summary LSA	0x2003	Inter-area prefix LSA	ABR	Area	Same function as in OSPFv2, simply renamed
4	AS Summary LSA	0x2004	Inter-Area Router LSA	ABR	Area	Same function as in OSPFv2, simply renamed
5	AS External LSA	0x4005	AS External LSA	ASBR	AS	Same function as in OSPFv2, simply renamed
7	NSSA External LSA	0x2007	Type 7 LSA	NSSA ASBR	NSSA	Same function as in OSPFv2, simply renamed
		0x2008	Link LSA	every router	Link	The LSA is sent by the router to every router linked to it. The LSA contains the link local address of the router and a list with IPv6 prefixes configured on the link.
		0x2009	Intra area prefix LSA	every router	Area	The LSA is sent only within an area. The LSA contains the IPv6 prefixes connected to the router or network.

In contrast to OSPFv2, OSPFv3 can forward unknown LSA types. Previously these were deleted and not distributed further.

RIP

RIPv2

Dynamic routing with RIPv2

The Routing Information Protocol (RIPv2) is used to create routing tables automatically. RIPv2 is used in autonomous systems (AS) with a maximum of 15 routers. It is based on the Distance-Vector algorithm.

RIPv2 was developed by the IETF (Internet Engineering Task Force) and is described in RFC 2453.

You configure RIPv2 in "Layer 3 (IPv4) > RIPv2".

Setting up a routing table

Since a router initially only knows its directly connected networks, it sends a request to its direct neighbor routers. As the reply, it receives the routing tables of the neighbor routers. Based on the information it receives, the router set up its own routing table.

The routing table contains entries for all possible destinations. Each entry includes the distance to the destination and the first router on the route.

The distance is also known as the metric. This indicates the number of routers to be passed through on the route to the destination (hop count). The maximum distance is 15 routers (hops).

Updating the routing table

Once the routing table is set up, the router sends its routing table to each direct neighbor router via the UDP port 520 at intervals of 30 seconds.

The router compares new routing information with its existing routing table. If the new information includes shorter routes, the existing routes are overwritten. The router only keeps the shortest route to a destination.

Checking neighbor routers

If a router does not receive messages from a neighbor router for longer than 180 seconds, it marks the router as being invalid. The router assigns the metric 16 for the neighbor router.

RIPng

RIPng (RIP next generation) is only used with IPv6 and is defined in RFC 2080. As with RIP (IPv4), RIPng is based on the distance vector algorithm of Bellman-Ford.

In contrast to RIPv2, RIPng is activated directly on the layer 3 interface (VLAN interface / router port) and not globally on the device.

RIPng uses the UDP port 521 and RIP the UDP port 520.

You configure RIPng in "Layer 3 (IPv6) > RIPng".

NAT/NAPT

Note

NAT/NAPT is possible only on layer 3 of the ISO/OSI reference model. To use the NAT function, the networks must use the IP protocol.

When using the ISO protocol that operates at layer 2, it is not possible to use NAT.

In Network Address Translation (NAT) IP subnets are divided into "Inside" and "Outside". The division is from the perspective of a NAT interface. All networks reachable via the NAT interface itself count as "Outside" for this interface. All networks reachable via other IP interfaces of the same device count as "Inside" for the NAT interface.

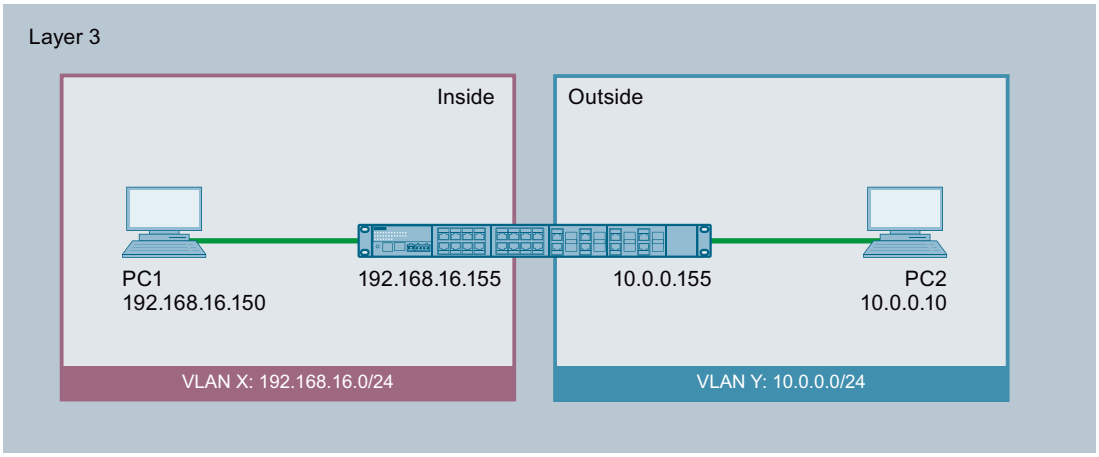
If there s routing via the NAT interface, the source or destination IP addresses of the transferred data packets are changed at the transition between "Inside" and "Outside". Whether or not the source or destination IP address is changed depends on the communications direction. It is always the IP address of the communications node that is located "Inside" that is adapted.

Depending on the perspective the IP address of a communications node is always designated as "Local" or "Global".

		Perspective	
		Local	**Global**
Position	**Inside**	An actual IP address that is assigned to a device in the internal network. This address cannot be reached from the external network.	An IP address at which an internal device can be reached from the external network.
	Outside	An actual IP address that is assigned to a device in the external network. Since only "Inside" addresses are converted, there is no distinction made between outside local and outside global.	

Example

In the example two IP subnets are connected together via an IE switch. The division is from the perspective of the NAT interface 10.0.0.155. The communication of PC2 with PC1 is implemented via NAT/NAPT.



The actual IP address of PC1 (inside local) is implemented statically with NAT. For PC2, PC1 can be reached at the inside global address.

		Perspective	
		Local	**Global**
Position	**Inside**	192.168.16.150	10.0.0.7
	Outside	10.0.0.10	

The actual IP address of PC1 (inside local) is implemented statically with NAPT. For PC2, PC1 can be reached at the inside global address.

		Perspective	
		Local	**Global**
Position	**Inside**	192.168.16.150:80	10.0.0.7:80
	Outside	10.0.0.10:1660	

Computing capacity

Due to the load limitation of the CPU packet receipt of the device is limited to 300 packets a second. This corresponds to a maximum data through of 1.7 Mbps. This load limitation does not apply per interface but generally for all packets going the CPU.

The entire NAT communication runs via the CPU and therefore represents competition for IP communication going to the CPU, e.g. WBM and Telnet.

Note that a large part of the computing capacity is occupied if you use NAT.

NAT

With Network Address Translation (NAT), the IP address in a data packet is replaced by another. NAT is normally used on a gateway between an internal network and an external network.

With source NAT, the inside local source address of an IP packet from a device in the internal network is rewritten by a NAT device to an inside global address at the gateway.

With destination NAT, the inside global source address of an IP packet from a device in the external network is rewritten by a NAT device to an inside local address at the gateway.

To translate the internal into the external IP address and back, the NAT device maintains a translation list. The address assignment can be dynamic or static. You configure NAT in "Layer 3 (IPv4) > NAT (Page 1470)".

NAPT

In "Network Address Port Translation" (NAPT), several internal source IP addresses are translated into the same external IP address. To identify the individual nodes, the port of the internal device is also stored in the translation list of the NAT device and translated for the external address.

If several internal devices send a query to the same external destination IP address via the NAT device, the NAT device enters its own external source IP address in the header of these forwarded frames. Since the forwarded frames have the same external source IP address, the NAT device assigns the frames to the devices using a different port number.

If a device from the external network wants to use a service in the internal network, the translation list for the static address assignment needs to be configured. You configure NAPT in "Layer 3 (IPv4) > NAT > NAPT (Page 1473)".

NAT/NAPT and IP routing

You can enable NAT/NAPT and IP routing at the same time. In this case, you need to regulate the reachability of internal addresses from external networks with ACL rules.

Link aggregation

Link aggregation

With link aggregation, several parallel physical connections with the same transmission speed are grouped together to form a logical connection with a higher transmission speed. This method based on IEEE 802.3ad is also known as port trunking or channel bundling.

Link aggregation works only with full duplex connections with the same transmission speed in point-to-point mode. This achieves multiplication of the bandwidth or transmission speed. If part of the connection fails, the data traffic is handled via the remaining parts of the connection.

To control and monitor, the Link Aggregation Control Layer (LACL) and the Link Aggregation Control Protocol (LACP) are used.

Quality of service

Quality of Service (QoS) is a method to allow efficient use of the existing bandwidth in a network.

QoS is implemented by prioritization of the data traffic. Incoming frames are sorted into a Queue according to a certain prioritization and further processed. This gives certain frames priority.

The different QoS methods influence each other and are therefore taken into account in the following order:

1. The switch first checks whether the incoming frame is a broadcast or agent frame.
   → When the first condition is met, the switch takes into account the priority set on the "General (Page 1389)" page.
   The switch sorts the frame into a queue according to the assignment on page "CoS Map (Page 1390)".
2. If the first condition is not met the switch checks whether the frame contains a VLAN tag.
   → If the second condition is met the switch checks the settings for the priority on the "General (Page 1389)" page. The switch checks whether a value other than "Do not force" is set for the priority.
   If the priority is set the switch sorts the frame into a queue according to the assignment on page "CoS Map (Page 1390)".
3. If the second condition is also not met the frames are further processed according to the Trust mode. You configure the Trust mode on the page "QoS Trust (Page 1391)".

See also

General (Page 1394)

Authentication

Overview of user management

Access to the device is managed by configurable user settings. Set up users with a password for authentication. Assign a role with suitable rights to the users.

The authentication of users can either be performed locally by the device or by an external RADIUS server. You configure how the authentication is handled on the "Security > AAA > General" page.

When you transfer the configuration of a device to TIA, the configured users, roles and groups are not transferred.

Compatibility with predecessor versions

With firmware version 5.1, user management was expanded by the RADIUS authorization mode "Vendor Specific". To ensure compatibility with firmware versions ≤ 5.0, the default setting was selected so that following a firmware update the earlier authentication mode "conventional" continues to be used.

Local login

The local logging in of users by the device runs as follows:

1. The user logs in with user name and password on the device.
2. The device checks whether an entry exists for the user.
   → If an entry exists, the user is logged in with the rights of the associated role.
   → If no corresponding entry exists, the user is denied access.

Login via an external RADIUS server

RADIUS(Remote Authentication Dial-In User Service) is a protocol for authenticating and authorizing users by servers on which user data can be stored centrally.

Depending on the RADIUS authorization mode you have selected on the "Security > AAA > RADIUS Client" page, the device evaluates different information of the RADIUS server.

RADIUS authorization mode "conventional"

If you have set the authorization mode "conventional", the authentication of users via a RADIUS server runs as follows:

1. The user logs in with user name and password on the device.
2. The device sends an authentication request with the login data to the RADIUS server.
3. The RADIUS server runs a check and signals the result back to the device.
   - The RADIUS server reports a successful authentication and for the "Service Type" attribute returns the value "Administrative User" to the device
     → The user is logged in with administrator rights.
   - The RADIUS server reports a successful authentication and returns a different or even no value to the device for the attribute "Service Type".
     → The user is logged in with read rights.
   - The RADIUS server reports a failed authentication to the device:
     → The user is denied access.

RADIUS authorization mode "Vendor Specific"

Requirement

For the RADIUS authorization mode "Vendor Specific" the following needs to be set on the RADIUS server:

- Manufacturer code: 4196
- Attribute number: 1
- Attribute format: Character string (group name)

Procedure

If you have set the authorization mode "Vendor Specific", the authentication of users via a RADIUS server runs as follows:

1. The user logs in with user name and password on the device.
2. The device sends an authentication request with the login data to the RADIUS server.
3. The RADIUS server runs a check and signals the result back to the device.
   Case A: The RADIUS server reports a successful authentication and returns the group assigned to the user to the device.
   - The group is known on the device and the user is not entered in the table "External User Accounts"
     → The user is logged in with the rights of the assigned group.
   - The group is known on the device and the user is entered in the table "External User Accounts"
     → The user is assigned the role with the higher rights and logged in with these rights.
   - The group is not known on the device and the user is entered in the table "External User Accounts"
     → The user is logged in with the rights of the role linked to the user account.
   - The group is not known on the device and the user is not entered in the table "External User Accounts"
     → The user is logged in with the rights of the role "Default".

   Case B: The RADIUS server reports a successful authentication but does not return a group to the device.
   - The user is entered in the table "External User Accounts":
     → The user is logged in with the rights of the linked role.
   - The user is not entered in the table "External User Accounts":
     → The user is logged in with the rights of the role "Default".

   Case C: The RADIUS server reports a failed authentication to the device:
   - The user is denied access.

Authentication method

You can configure the authentication methods "802,1X" and "MAC Authentication" as well as the "Guest VLANGuest VLAN" option separately for each port.

The functions have a hierarchical order. If all three functions are enabled, an attempt is made initially to authenticate the end device using "802.1x". If this authentication is unsuccessful, "MAC Authentication" is started. If this authentication is also unsuccessful, the end device is enabled for communication in the "Guest VLAN". "Guest VLAN" can only be used if at least one authentication method is active.

The two authentication methods depend on the end device. If the end device supports EAP (Extensible Authentication Protocol), it can be authenticated using the "802.1X" method. If the end device does not support EAP, it can be authenticated using "MAC Authentication". In this case, the IE switch adopts the role of the end device and uses the MAC address of the device as the authentication parameter.

802.1X

The "802.1X" authentication method works as follows:
An end device that supports EAP sends authentication information to the IE switch. The IE switch forwards the information to the authentication server. The authentication server checks the information and allows or denies the end device access to the network.

MAC Authentication

The "MAC Authentication" authentication method works as follows:
As soon as the IE switch receives a frame from the end device, the IE switch sends a query to the RADIUS server to allow or deny the end device access to the network.

Assignment of a VLAN via RADIUS or guest VLAN

Authentication with a change to the VLAN configuration

If during authentication a port is assigned to a VLAN dynamically using the function "RADIUS VLAN Assignment Allowed" or "Guest VLAN" the following happens:

- If the VLAN that is to be assigned has not been created on the device, the authentication is rejected.
- If the VLAN that is to be assigned has been created on the device:
  - The port becomes an untagged member in the assigned VLAN if it was not already.
  - The PVID of the port is changed to the ID of the assigned VLAN.

Note

If the port is only to be assigned to one VLAN, you need to adapt the VLAN configuration manually. As default, all ports are untagged members in "vlan 1".

If the authentication is canceled, e.g. by link down, the dynamic changes are canceled.

- The port is no longer a member in the assigned VLAN.
- The PVID of the port is reset to the value it had prior to authentication.

Authentication without a change to the VLAN configuration

If during authentication no VLAN is assigned either by the function "RADIUS VLAN Assignment Allowed" or by "Guest VLAN", the existing VLAN configuration of the port remains unchanged.

Editing properties and parameters

Editing options

You have the following options for editing properties and parameters:

- Hardware and network editor
  Once you have inserted the network component, you can edit the properties and parameters, for example the device name. You will find more detailed information in "Hardware and network editor".
- Web Based Management (WBM)
  The parameters and properties are accessible using the supplied HTML pages (WBM pages). Each WBM page has its own help page that describes the properties and parameters. For further information, refer to the configuration manuals:
  - "SCALANCE XB-200/XC-200/XP-200 Web Based Management"
    You will find the configuration manual on the Internet pages of Siemens Industry Online Support inSCALANCE XB-200 (https://support.industry.siemens.com/cs/ww/en/ps/15291/man), SCALANCE XC-200 (https://support.industry.siemens.com/cs/ww/en/ps/15291/man) and SCALANCE XP-200 (https://support.industry.siemens.com/cs/ww/en/ps/21869/man).
  - "SCALANCE XM-400/XR-500 Web Based Management"
    You will find this configuration manual on the Internet pages of Siemens Industry Online Support in SCALANCE XM-400 (https://support.industry.siemens.com/cs/ww/en/ps/15315/man) and SCALANCE XR-500 (https://support.industry.siemens.com/cs/ww/en/ps/15317/man).
- Command Line Interface
  All the configuration settings for the device can be made using the CLI. The CLI provides the same options as Web Based Management (WBM). For further information, refer to the configuration manuals:
  - "SCALANCE XB-200/XC-200/XP-200 Command Line Interface"
    You will find the configuration manual on the Internet pages of Siemens Industry Online Support inSCALANCE XB-200 (https://support.industry.siemens.com/cs/ww/en/ps/15291/man), SCALANCE XC-200 (https://support.industry.siemens.com/cs/ww/en/ps/15291/man) and SCALANCE XP-200 (https://support.industry.siemens.com/cs/ww/en/ps/21869/man).
  - "SCALANCE XM-400/XR-500 Command Line Interface"
    You will find this configuration manual on the Internet pages of Siemens Industry Online Support in SCALANCE XM-400 (https://support.industry.siemens.com/cs/ww/en/ps/15315/man) and SCALANCE XR-500 (https://support.industry.siemens.com/cs/ww/en/ps/15317/man).

Availability

The availability of the settings depends

- on the port type
- on the configuration mode

SCALANCE X distinguishes the following port types:

- Switch port
- Router port

The following configuration modes are available:

- Offline configuration
  You can configure a PC station the first time offline. In this mode only the settings that require no connection to the device are available.
- Online configuration mode
  If there is a connection to the device, there are additional pages available in the Inspector window. These pages include the note "This page is only available if there is an online connection to the device". With some settings, the additional information "only available online" is included.

Creating and deleting an entry

As an example, an entry will be created and deleted on the Syslog client. The procedure is valid on all pages.

Creating a new entry

1. Select the device in the network or device view.
2. Open the properties of the device in the Inspector window.
3. In the Inspector window go to System > Syslog client.
4. Enter the IP address for the Syslog server.
5. Click in the table.
6. Select the entry "New entry" in the shortcut menu.
   A new entry is created in the table.

Deleting an entry

1. Select the required entry in the table.
2. Select the "Delete" entry in the shortcut menu.

Buttons you require often

- **Refresh the display with "Refresh"**
  Pages that display current parameters have a "Refresh" button at the lower edge of the page. Click this button to request up-to-date information from the device for the current page.
- **Save settings for all ports with "Transfer to table"**
  Pages on which you can configure several ports have 2 tables. In the first table, you can make the settings for all ports and transfer these to the second table. In the last column of the first table, there is the "Transfer to table" button. Click this button to save the settings you have made for all ports.

Assigning the IP address

Configuration options

When shipped and after restoring to the factory settings, the device does not have an IP address.

The following options are available to assign an IP address to the device:

- DHCP (default setting)
- Primary Setup Tool
  For further information, refer to the configuration manual "Primary Setup Tool". You will find the MIB at Siemens Industry Automation and Drives Service & Support on the Internet under the entry ID 19440762 (http://support.automation.siemens.com/WW/view/en/19440762).
- STEP 7
- CLI via the serial interface
  For further information, refer to the configuration manual "SCALANCE XM-400/XR-500 Command Line Interface". You will find the configuration manual on the Internet pages of Siemens Industry Automation and Drives Service & Support (http://support.automation.siemens.com/WW/view/en/48805144/133300).

Requirement

- "Set IP address on device" is enabled in the properties of the device. You will find more detailed information in "Addressing PROFINET devices".

Uploading the configuration data with a PROFINET IO device

With a SCALANCE device that is assigned to a PROFINET IO controller, you must distinguish between the data of the PROFINET IO device configuration and the SCALANCE configuration:

- The PROFINET IO device configuration can only be supplied by the PROFINET IO controller.
- The SCALANCE configuration relates only to the data that is configurable in the Web Based Management (WBM) (System, Layer 2, Layer 3, Security).

Follow the steps below to load the entire configuration of a SCALANCE device:

1. Download the PROFINET IO device configuration via the PROFINET IO controller:
   - Select the PROFINET IO controller in the project tree.
   - Select the command "Online" > "Load device as new station (hardware and software)" in the toolbar.
   - See also the section "Loading project data from a device"
2. Load the SCALANCE configuration as a local upload via the device:
   - Select the SCALANCE device in the network view.
   - In the shortcut menu of the selected device select the command "SCALANCE configuration > Upload to PG/PC".
   - See also section "Configuring deevices".

Show information

Versions

Note

The page is only available if there is an online connection to the device.

This page shows the versions of the hardware and software of the device.

Description

Table 1 has the following columns:

- **Hardware**
  - Basic device
    Shows the basic device.
  - PX.X
    X.X = port in which the SFP module is inserted.
  - Slot X
    X = slot number: Module plugged into this slot.
- **Name**
  Shows the name of the device or module.
- **Revision**
  Shows the hardware version of the device.
- **Order number**
  Shows the order number of the device or described module.

Table 2 has the following columns:

- **Software**
  - Firmware
    Shows the current firmware version. If a new firmware file was downloaded and the device has not yet restarted, the firmware version of the downloaded firmware file is displayed here. After the next restart, the downloaded firmware is activated and used.
  - Bootloader
    Shows the version of the boot software stored on the device.
- **Description**
  Shows the short description of the software.
- **Version**
  Shows the version number of the software version.
- **Date**
  Shows the date on which the software version was created.

I&M

Note

The page is only available if there is an online connection to the device.

This page contains information about device-specific vendor and maintenance data such as the article number, serial number, version number etc. You cannot configure anything on this page.

Description of the displayed values

The table has the following rows:

- **Manufacturer ID**
  Shows the manufacturer ID.
- Article number
  Shows the article number.
- **Serial number**
  Shows the serial number.
- **Hardware revision**
  Shows the hardware version.
- **Software version**
  Shows the software version.
- **Revision counter**
  Shows the revision counter: Counter for revisions since the initial commissioning
- **Revision Date**
  Date and time of the last revision

- **Function tag**
  Shows the function tag (plant designation) of the device. The plant designation (HID) is created during configuration of the device with HW Config of STEP 7.
- **Location tag**
  Shows the location tag of the device. The location identifier (LID) is created during configuration of the device with HW Config of STEP 7.
- **Date**
  Shows the date created during configuration of the device with HW Config of STEP 7.
- **Description**
  Shows the description created during configuration of the device with HW Config of STEP 7.

ARP / neighbor table

ARP table

Note

The page is only available if there is an online connection to the device.

Assignment of MAC address and IP address

With the Address Resolution Protocol (ARP), there is a unique assignment of MAC address to IPv4 addresses. This assignment is kept by each network node in its own separate ARP table. The page shows the ARP table of the device.

Displayed values

The table has the following columns:

- **Interface**
  Shows the interface via which the row entry was learnt.
- **MAC Address**
  Shows the MAC address of the destination or source device.
- **IP address**
  Shows the IP address of the target device.
- **Media Type**
  Shows the type of connection.
  - Dynamic
    The device recognized the address data automatically.
  - Static
    The addresses were entered as static addresses

IPv6 Neighbor Table

Assignment of MAC address and IPv6 address

Via the IPv6 neighbor table, there is a unique assignment of MAC address to IPv6 address. This assignment is kept by each network node in its own separate neighbor table.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Displays the interface via which the row entry was learnt.
- **MAC Address**
  Shows the MAC address of the destination or source device.
- **IP Address**
  Shows the IPv6 address of the destination device.
- **Media Type**
  Shows the type of connection.
  - Dynamic
    The device recognized the address data automatically.
  - Static
    The addresses were entered as static addresses.

Log table

Logging events

Note

The page is only available if there is an online connection to the device.

The device allows you to log events that occur, some of which you can specify in " System > Events". This, for example, allows you to record when an authentication attempt failed or when the connection status of a port has changed. The content of the events log table is retained even when the device is turned off.

Settings

- **Severity Filter**
  You can filter the entries in the table according to severity. Select the required entries in the check boxes above the table.
  - Info
    Information
  - When this parameter is enabled, all entries of the category "Info" are displayed.
  - Warning
    Warning
    When this parameter is enabled, all entries of the category "Warning" are displayed.
  - Critical
    Critical
    When this parameter is enabled, all entries of the category "Critical" are displayed.

  The table has the following columns:

  - **Restart**
    Counts the number of restarts since you last reset to factory settings and shows the device restart after which the corresponding event occurred.
  - **System Up Time**
    Shows the time the device has been running since the last restart when the described event occurred.
  - **System Time**
    Shows the date and time when the described event occurred.
    If the system time is set, the time is also displayed at which the event occurred.
  - **Severity**
    Shows the severity of the message.
  - **Log Message**
    Displays a brief description of the event that has occurred. You will find the list of possible messages in Appendix D of the configuration manual.

Buttons

- **Delete**
  Click this button to delete the content of the event log file. All entries are deleted regardless of what you have selected under "Severity Filters".
  The display is also cleared. The restart counter is only reset after you have restored the device to the factory settings and restarted the device.

  Note

  The table can contain 400 entries for each severity. The number of entries in this table is restricted to 1200. When this number is reached, the oldest entries of the relevant severity are discarded. The table remains permanently in memory.

Fault

Error status

Note

The page is only available if there is an online connection to the device.

if an error occurs, it is shown on this page. On the device, errors are indicated by red fault LED lighting up.

Internal errors of the device and errors that you configure on the following pages are indicated:

- System > Events"
- "System" > Fault Monitoring"

Errors of the "Cold/Warm Start" event can be deleted by a confirmation.

The calculation of the time of an error always begins after the last system start.

If there are no errors present, the fault LED switches off.

Description of the displayed values

- **No. of Signaled Faults**
  Shows the number of reported faults.

The table contains the following columns:

- **Fault Time**
  Shows the time the device has been running since the last restart when the described error/ fault occurred.
- **Fault Description**
  Displays a brief description of the fault/error that has occurred.
- **Clear Fault State**
  If the "Clear Fault State" button is enabled, you can delete the fault.
- "Reset Counters" button
  Click "Reset Counters" to reset all counters. The counters are reset by a restart.

Redundancy

Spanning Tree

Note

The page is only available if there is an online connection to the device.

The page shows the current information about the Spanning Tree and the settings of the root bridge.

Displayed values

- **Spanning Tree Mode**
  Shows the set mode. You specify the mode in "Layer 2 > Spanning Tree > General".
  The following values are possible:
  - '-'
  - STP
  - RSTP
  - MSTP
- **Instance ID**
  Shows the number of the instance. The parameter depends on the configured mode.
- **Bridge Priority / Root Priority**
  Which device becomes the root bridge is decided by the bridge priority. The bridge with the highest priority (in other words, with the lowest value for this parameter) becomes the root bridge. If several devices in a network have the same priority, the device whose MAC address has the lowest numeric value will become the root bridge. Both parameters, bridge priority and MAC address together form the bridge identifier. Since the root bridge manages all path changes, it should be located as centrally as possible due to the delay of the frames. The value for the bridge priority is a whole multiple of 4096.
- **Bridge address / root address**
  The bridge address shows the MAC address of the device and the root address shows the MAC address of the root bridge.
- **Root Cost**
  Shows the path costs from the device to the root bridge.
- **Bridge Status**
  Shows the status of the bridge, e.g. whether or not the device is the root bridge.
- **Regional root priority** (available only with MSTP)
  For a description, see Bridge priority / Root priority.
- **Regional root address** (available only with MSTP)
  Shows the MAC address of the device.
- **Regional Root Cost** (only available with MSTP)
  The path costs from this device to the regional root bridge.

The table has the following columns:

- **Port**
  Shows the port via which the device communicates.
- **Role**
  Shows the status of the port. The following values are possible:
  - Disabled
    The port was removed manually from the spanning tree and will no longer be taken into account by the spanning tree.
  - Designated
    The ports leading away from the root bridge.
  - Alternate
    The port with an alternative route to a network segment
  - Backup
    If a switch has several ports to the same network segment, the "poorer" port becomes the backup port.
  - Root
    The port that provides the best route to the root bridge.
  - Master
    This port points to a root bridge located outside the MST region.
- **Status**
  Displays the current status of the port. The values are only displayed. The parameter depends on the configured protocol. The following statuses are possible:
  - Discarding
    The port receives BPDU frames. Other incoming or outgoing frames are discarded.
  - Listening
    The port receives and sends BPDU frames. The port is involved in the spanning tree algorithm. Other outgoing and incoming frames are discarded.
  - Learning
    The port actively learns the topology; in other words, the node addresses. Other outgoing and incoming frames are discarded.
  - Forwarding
    Following the reconfiguration time, the port is active in the network. The port receives and sends data frames.
- **Oper. Version**
  Shows which Spanning Tree protocol is used by the port.
- **Priority**
  If the path calculated by the spanning tree is possible over several ports of a device, the port with the highest priority (in other words the lowest value for this parameter) is selected. A value between 0 and 240 can be entered for the priority in steps of 16. If you enter a value that cannot be divided by 16, the value is automatically adapted.

- **Path Cost**
  This parameter is used to calculate the path that will be selected. The path with the lowest value is selected. If several ports of a device have the same value, the port with the lowest port number will be selected.
  If the value "Cost Calc." is "0", the automatically calculated value is shown. Otherwise, the value of "Cost Calc." is displayed.
  The calculation of the path costs is largely based on the transmission speed. The higher the achievable transmission speed is, the lower the value of the path costs.
  Typical values for path costs with rapid spanning tree:
  - 10,000 Mbps = 2,000
  - 1000 Mbps = 20,000
  - 100 Mbps = 200,000
  - 10 Mbps = 2,000,000.

  You configure the "Cost. Calc." on the pages "Layer 2 > Spanning Tree > CIST Port" and "Layer 2 > Spanning Tree > MST Port".
- **Edge type**
  Shows the type of the connection. The following values are possible:
  - Edge Port

    An edge port is connected to this port.
  - No Edge Port
    There is a spanning tree device at this port.
- **P.t.P. Type**
  Shows the type of point-to-point link. The following values are possible:
  - P.t.P.
    With half duplex, a point-to-point link is assumed.
  - Shared Media

    With a full duplex connection, a point-to-point link is not assumed.

VRRP statistics

Introduction

Note

The page is only available if there is an online connection to the device.

This page shows the statistics of the VRRP protocol and all configured virtual routers.

Description of the displayed values

The following fields are displayed:

- VRID errors
  Shows how many VRRP frames containing an unsupported VRID were received.
- Version errors
  Shows how many VRRP frames containing an invalid version number were received.
- Checksum errors
  Shows how many VRRP frames containing an invalid checksum were received.

The table has the following columns:

- Interface
  Interface to which the settings relate.
- VRID
  Shows the ID of the virtual router.
- Change to master status
  Shows how often this virtual router changed to the "Master" role.
- Advertisements recd.
  Shows how often a VRRP frame was received that contained a bad address list.
- Advertisement Interval Errors
  Shows how many bad VRRP packets were received whose interval does not match the value set locally.
- IP TTL errors
  Shows how many bad VRRP packets were received whose TTL (Time to live) value in the IP header is incorrect.
- Prio 0 received
  Shows how many VRRP packets with priority 0 were received. VRRP packets with priority 0 are sent when a master router is shut down. These packets allow a fast handover to the relevant backup router.
- Prio 0 sent
  Shows how many VRRP packets with priority 0 were sent. Packets with priority 0 are sent when a master router is shut down. These packets allow a fast handover to the relevant backup router.
- Invalid Type
  Shows how many bad VRRP packets were received whose authentication type was not 0. Type 0 means "no authentication".
- Address List Errors
  Shows how many bad VRRP packets were received whose address list does not match the locally configured list.
- Invalid Auth. Type
  Shows how many bad VRRP packets were received whose authentication type was not type 0. Type 0 means "no authentication".

- **Auth. Type Mismatch**
  Shows how many bad VRRP frames were received whose authentication type does not match.
- **Frame length error**
  Shows how many bad VRRP frames were received whose length is not correct.

VRRP Statistics

Introduction

This page shows the statistics of the VRRPv3 protocol and all configured virtual routers.

Note

The page is only available if there is an online connection to the device.

Displayed values

The following fields are displayed:

- **VRID Errors**
  Shows how many VRRPv3 packets containing an unsupported VRID were received.
- **Version Errors**
  Shows how many VRRPv3 packets containing an invalid version number were received.
- **Checksum Errors**
  Shows how many VRRPv3 packets containing an invalid checksum were received.

The table has the following columns:

- **Interfaces**
  Interface to which the settings relate.
- **VRID**
  Shows the ID of the virtual router. Valid values are 1 ... 255.
- **Type**
  Shows the version of the IP protocol.
- **Become Master**
  Shows how often this virtual router changed to the "Master" status.
- **Advertisements Received**
  Shows how many VRRPv3 packets were received.
- **Advertisement Interval Errors**
  Shows how many bad VRRPv3 packets were received whose interval does not match the value set locally.
- **IP TTL Errors**
  Shows how many bad VRRPv3 packets were received whose TTL (Time to live) value in the IP header is incorrect.

- **Prio 0 received**
  Shows how many VRRPv3 packets with priority 0 were received. VRRPv3 packets with priority 0 are sent when a master router is shut down. These packets allow a fast handover to the relevant backup router.
- **Prio 0 sent**
  Shows how many VRRPv3 packets with priority 0 were sent. Packets with priority 0 are sent when a master router is shut down. These packets allow a fast handover to the relevant backup router.
- **Invalid Type**
  Shows how many bad VRRPv3 packets were received whose value in the "Type" field of the IP header is invalid.
- **Address List Errors**
  Shows how many bad VRRPv3 packets were received whose address list does not match the locally configured list.
- **Packet Length Errors**
  Shows how many bad VRRPv3 packets were received whose length is not correct.

Ring Redundancy

Note

The page is only available if there is an online connection to the device.

Automatic Redundancy Detection (ARD) is the default when the device ships. If you want to use the previous High Speed Redundancy Protocol (HRP), HRP must be configured.

- Reconfiguration time of the frame traffic following a failover in MRP: 200 ms
- Reconfiguration time of the frame traffic following a failover in HRP: 300 ms

This page informs you about the values currently set for the device.

Description

The following fields are displayed:

- **Redundancy Function**
  The "Redundancy Function" column shows the role of the device within the ring:
  - no Ring Redundancy
    The IE switch is operating without redundancy function.
  - HRP Client
    The IE switch is operating as an HRP client.
  - HRP manager
    The IE switch is operating as an HRP manager.
  - MRP Client
    The IE switch is operating as an MRP client.
  - MRP manager
    The IE switch is operating as an MRP manager. Using STEP 7, the role "Manager" was set for the device.
  - MRP Auto-Manager
    The IE switch is operating as an MRP manager. Using WBM or CLI the role "MRP Auto-Manager" or using STEP 7 the role "Manager (Auto)" was set.
- **RM Status**
  The "RM Status" column shows whether or not the IE switch is operating as redundancy manager and whether it has opened or closed the ring in this role.
  - Passive
    The IE switch is operating as redundancy manager and has opened the ring; in other words, the line of switches connected to the ring ports is operating problem free. The "Passive" status is also displayed if the IE switch is not operating as the redundancy manager (Redundancy manager disabled).
  - Active
    The IE switch is operating as redundancy manager and has closed the ring; in other words, the line of switches connected to the ring ports is interrupted (problem). The redundancy manager connects its ring ports through and restores an uninterrupted linear topology.
  - "-"
    The redundancy function is disabled.
- **Observer Status**
  Shows the current status of the observer.
- **Ring Port 1 and Ring Port 2**
  The "Ring Port 1" and "Ring Port 2" columns show the ports being used as ring ports.

- **No. of Changes to RM Active State**
  Shows how often the device as redundancy manager switched to the active status, i.e. closed the ring.
  If the redundancy function is disabled or the device is not the HRP/MRP manager, the text "Redundancy manager disabled" appears.
- **Max. delay of the RM test frames [ms]**
  Shows the maximum delay for test frames of the redundancy manager.
  If the redundancy function is disabled or the device is not the HRP/MRP manager, the text "Redundancy manager disabled" appears.

Standby

Note

The page is only available if there is an online connection to the device.

On this page, you obtain information about the status of the device in terms of standby redundancy. The text boxes on this page are read-only.

Note

Device with the higher MAC address becomes master

When linking HRP rings redundantly, two devices are always configured as a master/slave pair. This also applies to interrupted HRP rings = linear buses. When operating normally, the device with the higher MAC address adopts the role of master. This type of assignment is important in particular when a device is replaced. Depending on the MAC addresses, the previous device with the slave function can take over the role of the standby master.

The Standby tab shows the status of the standby function:

Description

The following fields are displayed:

- **Standby ports**
  Shows the standby port.
- **Standby name**
  Standby connection name

- **Standby Function**
  Shows whether the standby connection is activated or deactivated.
  - Master
    The device has a connection to the partner device and is operating as master. In normal operation, the standby port of this device is active.
  - Slave
    The device has a connection to the partner device and is operating as slave. In normal operation, the standby port of this device is inactive.
  - Disabled
    The standby link is disabled. The device is operating neither as master nor slave. The port configured as a standby port works as a normal port without standby function.
  - Connection establishment
    No connection has yet been established to the partner device. The standby port is inactive. In this case, either the configuration on the partner device is inconsistent (for example incorrect connection name, standby link disabled) or there is a physical fault (for example device failure, link down).
  - Connection lost
    The existing connection to the partner device has been lost. In this case, either the configuration on the partner device was modified (for example a different connection name, standby link disabled) or there is a physical fault (for example device failure, link down).
- **Standby status**
  Shows the status of the standby port:
  - Active
    The standby port of this device is active; in other words is enabled for frame traffic.
  - Passive
    The standby port of this device is inactive; in other words is blocked for frame traffic.
  - "-"
    The standby function is disabled.
- **No. of Changes to Standby Active State**
  Shows how often the IE switch has changed the standby status from "Passive" to "Active". If the connection of a standby port fails on the standby master, the IE switch changes to the "Active" status. If the standby function is disabled, the text "Standby Disabled" appears in this box.
  Following each restart on the device, the counters are automatically reset.

Ethernet Statistics

Interface statistics

Interface statistics

The page shows the statistics from the interface table of the Management Information Base (MIB).

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **In Octet**
  Shows the number of received bytes.
- **Out Octet**
  Shows the number of sent bytes.
- **In Unicast**
  Shows the number of received unicast frames.
- **In Non Unicast**
  Shows the number of received frames that are not of the type unicast.
- **Out Unicast**
  Shows the number of sent unicast frames.
- **Out Non Unicast**
  Shows the number of sent frames that are not of the type unicast.
- **In Errors**
  Shows the number of all possible RX errors, refer to the tab "Packet Error".

Packet Size

Frames sorted by length

This page displays how many frames of which size were sent and received at each port.

Note

The page is only available if there is an online connection to the device.

The displayed values are transferred by RMON.

On the page "Layer 2 > RMON > Statistics", you can set the ports for which values will be displayed.

Displayed values

The table has the following columns:

- **Port**
  Shows the available ports and link aggregations. The port is made up of the module number and the port number, for example port 0.1 is module 0, port 1.

  Note
  Display of frame statistics

  In the statistics relating to frame lengths, note that both incoming and outgoing frames are counted.

- **Frame lengths**
  The other columns after the port number contain the absolute numbers of frames according to their frame length.
  The following frame lengths are distinguished:
  - 64 bytes
  - 65 - 127 bytes
  - 128 - 255 bytes
  - 256 - 511 bytes
  - 512 - 1023 bytes
  - 1024 - max.

Note
Data traffic on blocked ports

For technical reasons, data packets can be indicated on blocked ports.

Button

Reset Counter

Click "Reset Counter" to reset all counters. The counters are reset by a restart.

Frame type

Received frames sorted by type

This page displays how many frames of the type "unicast", "multicast", and "broadcast" were received at each port.

You cannot configure anything on this page.

The displayed values are transferred by RMON. On the page "Layer 2 > RMON > Statistics", you can set the ports for which values will be displayed.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Port**
  Shows the available ports and link aggregations. The port is made up of the module number and the port number, for example port 0.1 is module 0, port 1.
- **Unicast / Multicast / Broadcast**
  The other columns after the port number contain the absolute numbers of the incoming frames according to their frame type "unicast", "multicast" and "broadcast".

Button

Reset Counter

Click "Reset Counter" to reset all counters. The counters are reset by a restart.

Packet Errors

Received bad frames

This page shows how many bad frames were received per port.

You cannot configure anything on this page.

The displayed values are transferred by RMON. On the page "Layer 2 > RMON > Statistics", you can set the ports for which values will be displayed.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Port**
  Shows the available ports and link aggregations.
- **Error types**
  The other columns after the port number contain the absolute numbers of the incoming frames according to their error type.
  In the columns of the table, a distinction is made according to the following error types:
  - CRC

    Packets whose content does not match the CRC checksum.
  - Undersize
    Packets with a length less than 64 bytes.
  - Oversize
    Frames discarded because they were too long.
  - Fragments

    Packets with a length less than 64 bytes and a bad CRC checksum.
  - Jabbers
    VLAN-tagged packets with an incorrect CRC checksum that were discarded because they were too long.
  - Collisions
    Collisions that were detected.

Button

Reset Counter

Click "Reset Counter" to reset all counters. The counters are reset by a restart.

History

Samples of the statistics

The page shows samples from each port with information from the statistics.

On the page "Layer 2 > RMON > History", you can set the ports for which samples will be taken.

Note

The page is only available if there is an online connection to the device.

Settings

- **Port**
  Select the port for which the history will be displayed.

Displayed values

- **Entries**
  Maximum number of samples that can be saved at the same time.
- **Interval [s]**
  Interval after which the current status of the statistics will be saved as a sample.

The table has the following columns:

- **Sample**
  Number of the sample
- **Time for the Sample**
  System up time at which the sample was taken.
- **Unicast**
  Number of received unicast frames.
- **Multicast**
  Number of received multicast frames.
- **Broadcast**
  Number of received broadcast frames.
- **CRC**
  Number of frames with a bad CRC checksum.
- **Undersize**
  Number of frames that are shorter than 64 bytes.
- **Oversize**
  Number of frames discarded because they were too long.
- **Fragments**
  Number of frames that are shorter than 64 bytes and have a bad CRC checksum.
- **Jabbers**
  Number of frames with a VLAN tag that have a bad CRC checksum and will be discarded because they are too long.
- **Collisions**
  Number of collisions of received frames.
- **Utilization**
  Utilization of the port during a sample.

Unicast

This page shows the current content of the unicast filter table. This table lists the source addresses of unicast address frames. Entries can be made either dynamically when a node sends a frame to a port or statically by the user setting parameters.

Description

- **VLAN ID**
  Shows the VLAN-ID assigned to this MAC address.
- **MAC Address**
  Shows the MAC address of the node that the device has learned or the user has configured.
- **Status**
  Shows the status of each address entry:
  - Learnt
    The specified address was learned by receiving a frame from this node and will be deleted when the aging time expires if no further packets are received from this node.

    Note

    If there is a link down, learned MAC entries are deleted.

  - Static
    Configured by the user. Static addresses are stored permanently; in other words, they are not deleted when the aging time expires or when the switch is restarted.
- **Port**
  Shows the port via which the node with the specified address can be reached. Frames received by the device whose destination address matches this address will be forwarded to this port.

Multicast

This table shows the multicast frames currently entered in the multicast filter table and their destination ports. The entries can be dynamic (the device has learned them) or static (the user has set them).

Description

- **VLAN ID**
  Shows VLAN ID of the VLAN to which the MAC multicast address is assigned.
- **MAC Address**
  Shows the MAC multicast address that the device has learned or the user has configured.

- **Status**
  Shows the status of each address entry. The following information is possible:
  - Static
    The address was entered statically by the user. Static addresses are stored permanently; in other words, they are not deleted when the aging time expires or when the device is restarted. These must be deleted by the user.
  - IGMP
    The destination port for this address was obtained by IGMP configuration.
  - MLD
    The destination port for this address was determined by MLD configuration.
  - GMRP
    The destination port for this address was registered by a received GMRP frame.
- **List of ports**
  There is a column for each slot. Within a column, the multicast group to which the port belongs is shown:
  - M (Member)
    Multicast frames are sent via this port.
  - R (Registered)
    Member of the multicast group, registration was by an GMRP frame.
  - I (IGMP/MLD)
    Member of the multicast group, registration was by an IGMP/MLD frame.
  - –
    Not a member of the multicast group. No multicast frames with the defined multicast MAC address are sent via this port.
  - F (Forbidden)
    Not a member of the multicast group. Moreover, this address must not be an address learned dynamically with GMRP or IGMP/MLD.

LLDP

Note

The page is only available if there is an online connection to the device.

Status of the neighborhood table

This page shows the current content of the neighborhood table. This table stores the information that the LLDP agent has received from connected devices. You set the interfaces via which the LLDP agent receives or sends information in the following section: "Layer 2 > LLDP".

Description of the displayed values

The table contains the following columns:

- **System name**
  System name of the connected device.
- **Device ID**
  Device ID of the connected device.
- **Local interface**
  Port at which the IE switch received the information.
- **Hold Time**
  An entry remains stored in the MIB for the time specified here. If the IE switch does not receive any new information from the connected device during this time, the entry is deleted.
- **Property**
  Shows the properties of the connected device:
  - Router
  - Bridge
  - Telephone
  - DOCSIS Cable Device
  - WLAN Access Point
  - Repeater
  - Station
  - Other
- **Port ID**
  Port of the device with which the IE switch is connected.

DHCP Server

This page shows whether IPv4 addresses were assigned to the devices by the DHCP server.

Note

The page is only available if there is an online connection to the device.

Description

- **IP Address**
  Shows the IPv4 address assigned to the DHCP client.
- **Pool ID**
  Shows the number of the IPv4 address band.
- **Identification method**
  Shows the method according to which the DHCP client is identified.

- **Identification value**
  Shows the MAC address or he client ID of the DHCP client.
- **Remote ID**
  Shows the remote ID of the DHCP client.
- **Circuit ID**
  Shows the circuit ID of the DHCP client.
- **Assignment method**
  Shows whether the IPv4 address was assigned statically or dynamically. You configure the static entries in "System > DHCP > Static Leases".
- **Assignment status**
  Shows the status of the assignment.
  - Assigned
    The assignment is used.
  - Not used
    The assignment is not used.
  - Probing
    The assignment is being checked.
  - Unknown
    The status of the assignment is unknown.
- **Expire Time**
  Shows how long the assigned IPv4 address is still valid. Once this period has elapsed, the DHCP client must either request a new IPv4 address or extend the lease time of the existing IPv4 address.

Diagnostics

This page shows the diagnostics values of internal and external modules of the device. The modules are only shown if they make diagnostics information available. If you add or remove a module, the display is automatically adapted.

If the diagnostic value falls below or exceeds the displayed threshold values, the status changes accordingly.

On the "System > Events > Configuration" page, you can specify how the device signals the status change.

Description

The Usage Table contains the following columns:

- **Name**
  Shows the name of the module.
- **Status**
  Depending on the relationship between the threshold values and the current temperature the following statuses are displayed in ascending priority.
  - OK
    The temperature value is within the preset threshold values.
  - WARNING
    The lower or upper threshold value of the severity level "Warning" was exceeded.
  - CRITICAL
    The lower or upper threshold value of the severity level "Critical" was exceeded.
  - INVALID
    The value could not be read out or is invalid. The "Power [%] box shows "-".
  - INITIAL
    No data has been read out yet. "-" is displayed in all boxes.
- **Power [%]**
  Shows the current value of the power. The display is updated at regular intervals.
- **High Warning Threshold [%]**
  If the value exceeds this value, the status changes to "WARNING". You can configure that you are informed by a message.
- **High Critical Threshold [%]**
  If the value exceeds this value, the status changes to "CRITICAL". You can configure that you are informed by a message.

The Temperature Table contains the following columns:

- **Name**
  Shows the name of the module.
  The information in the row "Chassis" relates to the inner temperature of the housing.
  With modular devices, the port is also specified.
- **State**
  Depending on the relationship between the threshold values and the current temperature the following statuses are displayed in ascending priority.
  - OK
    The temperature value is within the preset threshold values.
  - WARNING
    The lower or upper threshold value of the severity level "Warning" was exceeded.
  - CRITICAL
    The lower or upper threshold value of the severity level "Critical" was exceeded.
  - INVALID
    The value could not be read out or is invalid. In the "Temperature [°C]" box "-" is displayed.
  - INITIAL
    No data has been read out yet. "-" is displayed in all boxes.
- **Temperature [°C]**
  Shows the current value of the temperature. The display is updated at regular intervals. The value can have a tolerance of +/- 3 °C
- **Lower Threshold [°C] (Critical)**
  If the value falls below this value, the status changes to "CRITICAL". You can configure that you are informed by a message.
- **Lower Threshold [°C] (Warning)**
  If the value falls below this value, the status changes to "WARNING". You can configure that you are informed by a message.
- **Upper Threshold [°C] (Warning)**
  If the value exceeds this value, the status changes to "WARNING". You can configure that you are informed by a message.
- **Upper Threshold [°C] (Critical)**
  If the value exceeds this value, the status changes to "CRITICAL". You can configure that you are informed by a message.

FMP

Monitoring optical links

With Fiber Monitoring, you can monitor optical links. The table shows the current status of the ports.

You set the values to be monitored on the following page: "Layer 2 > FMP".

Note

The page is only available if there is an online connection to the device.

Displayed values

- **Port**
  Shows the available optical ports.
- **Rx Power State**
  - disabled
    Fiber monitoring is disabled.
  - ok
    The value for the received power of the optical link is within the set limits
  - maint. req.
    Check the link.
    A warning is signaled.
  - maint. dem.
    The link needs to be checked.
    An alarm is signaled and the fault LED is lit.
  - link down
  - The connection is interrupted.

- **RX Power**
  Shows the current value of the received power. The value can have a tolerance of +/- 3 dB. If there is no connection (link down) or fiber monitoring is disabled, "-" is displayed. If fiber monitoring is not enabled on the partner port, the value 0.0 is displayed.
- **Power Loss State**
  To be able to monitor the power loss of the connection, the port requires a link to another port with FM enabled.
  - disabled
    FM is disabled.
  - ok
    The value for the power loss of the optical link is OK.
  - maint. req.
    Check the link.
    A warning is signaled.
  - maint. dem.
    The link needs to be checked.
    An alarm is signaled and the fault LED is lit.
  - idle
    The port has no connection to another port with fiber monitoring enabled.
    If no diagnostics information is received from the optical port of the connection partner for 5 cycles, the fiber monitoring connection is assumed to be interrupted. A cycle lasts 5 seconds.
- **Power Loss [dB]**
  Shows the current value of the power loss. The value can have a tolerance of +/- 3 dB. If there is no connection (link down), fiber monitoring is disabled or the partner port does not support fiber monitoring, "-" is displayed.

IPv4 routing

Routing table

This page shows the routes currently being used.

Description

The table has the following columns:

- **Destination network**
  Shows the destination address of this route.
- **Subnet mask**
  Shows the subnet mask of this route.
- **Gateway**
  Shows the gateway for this route.
- **Interface**
  Shows the interface for this route.

- **Metric**
  Shows the metric of the route. The higher value, the longer the packets require to their destination.
- **Routing protocol**
  Shows the routing protocol from which the entry in the routing table originates. The following entries are possible:
  - Connected: Connected routes
  - Static: Static routes
  - RIP: Routes via RIP
  - OSPF: Routes via OSPF
  - other: Other routes

OSPFv2 interfaces

Overview

Note

The page is only available if there is an online connection to the device.

This page shows the configuration of the OSPF interface.

Description of the displayed values

The table has the following columns:

- **IP Address**
  Shows the IPv4 address of the OSPF interface
- **Area ID**
  Shows the Area ID to which the OSPF interface belongs.

- **Interface Status**
  Shows the status of the WLAN interface:
  - Down
    The interface is not available.
  - Loopback
    Loopback interface
  - Waiting
    Startup and negotiation of the interface.
  - Point to Point
    Point-to-point connection
  - Designated Router
    The router is a designated router and generates a network LSA.
  - Backup D. Router
    The router is backup for the designated router.
  - Other D. Router
    The interface is started up. The router is neither a designated router nor a backup designated router.
- **OSPF status**
  Shows the status of OSPF.
  - Enabled: OSPF is enabled on the interface.
  - Disabled: OSPF is disabled on the interface.
- **Designated Router**
  Shows the IPv4 address of the designated router for this OSPF interface.
- **Backup Designated Router**
  Shows the IPv4 address of the designated backup router for this OSPF interface.
- **Events**
  Shows the number of status changes of OSPF.

OSPFv2 neighbors

Note

The page is only available if there is an online connection to the device.

This page shows the dynamically detected neighbor routers in the relevant networks.

Description of the displayed values

The table has the following columns:

- **IP Address**
  Shows the IPv4 address of the neighbor router in this network.
- **Router ID**
  Shows the ID of the neighbor router. The two addresses can match.

- **Status**
  Shows the status of the neighbor router. The status can adopt the following values:
  - unknown
    Status of the neighbor router is unknown.
  - down
    The neighbor router cannot be reached.
  - attemptand init
    Short-lived statuses during initialization
  - two-way
    Two-way receipt of Hello packets. Specification of the designated router and the designated backup router.
  - exchangestart, exchange and loading
    Status during exchange of the LSAs
  - full
    The database is complete and synchronized within the area. The routes can now be detected.

    Note

    Normal status

    If the partner router is a designated router or a designated backup router, the status is "full". Otherwise the status is "two-way".

- **Assoc. Area Type**
  Shows the area type via which the neighbor-neighbor relation is maintained. The following area types exist:
  - Normal
  - Stub
  - NSSA
  - Backbone
- **Priority**
  Shows the priority of the neighbor router. This is only significant when selecting the designated router on a network. For virtual neighbor routers, this information is irrelevant.
- **Hello suppr**
  Shows the suppressed Hello packets to the neighbor router. This field normally displays "no".
- **Hello Queue**
  Shows the length of the queue with Hello packets still to be transmitted.
- **Events**
  Shows the number of status changes.

OSPFv2 virtual neighbors

Note

The page is only available if there is an online connection to the device.

This page shows the configured virtual neighbor routers.

Overview

The table has the following columns:

- **IP Address**
  Shows the IPv4 address of the virtual neighbor router in this network.
- **Router ID**
  Shows the router ID of the virtual neighbor router.
- **Status**
  Shows the status of the neighbor router. The status can adopt the following values:
  - unknown
    Status of the neighbor router is unknown.
  - down
    The neighbor router cannot be reached.
  - attemptand init
    Short-lived statuses during initialization
  - two-way
    Two-way receipt of Hello packets. Specification of the designated router and the designated backup router.
  - exchangestart, exchange and loading
    Status during exchange of the LSAs
  - full
    The database is complete and synchronized within the area. The routes can now be detected.

    Note

    Normal status

    If the partner router is a designated router or a designated backup router, the status is "full". Otherwise the status is "two-way".
- **Transit Area ID**
  Shows the ID of the area via which the virtual neighborhood relation exists.
- **Hello Suppr.**
  Shows whether there are suppressed Hello packets to the virtual neighbor router.
  - no: There are no suppressed Hello packets (default)
  - yes: There are suppressed Hello packets.

- **Hello Queue**
  Shows the length of the queue with Hello packets still to be transmitted.
- **Events**
  Shows the number of status changes.

OSPFv2 LSDB

Overview

Note

The page is only available if there is an online connection to the device.

The link state database is the central database for managing all links within an area. It consists of the link state advertisements (LSAs). The most important data of these LSAs is shown on the this WBM page.

Description of the displayed boxes

The table has the following columns:

- **Area ID**
  Shows the ID of the area to which the LSA belongs. If the LSA is an external connection, '-' is displayed.
- **LSA type**
  Shows the LSA type. The following values are possible:
  - unknown
    LSA type is unknown.
  - Router
    The router LSA (Type 1) is sent by the OSPF router within an area. The LSA contains information about the status of all router interfaces.
  - Network
    The network LSA (Type 2) is sent by the designated router within an area. The LSA contains a list of routers connected to the network.
  - NSSA External
    The NSSA external LSA (Type 7) is sent by the NSSA-ASBR within an NSSA. The NSSA-ASBR receives LSAs of Type 5 and converts the information to LSAs of Type 7. The NSSA router can forward these LSAs within an NSSA.
  - Summary
    The summary LSA (Type 3) is sent by the ABR within an area. The LSA contains information about routes to other networks.
  - AS Summary
    The AS summary LSA (Type 4) is sent by the area border router within an area. The LSA contains information about routes to other autonomous systems.
  - AS External
    The AS external LSA (Type 5) is sent by the AS border router within an autonomous system. The LSA contains information about routes from one network to another.
- **Link State ID**
  Shows the ID of the LSA.
- **Router ID**
  Shows the ID of the router that sent this LSA.
- **Sequence number**
  Shows the sequence number of the LSA. Each time an LSA is renewed, this sequence number is incremented by one.

RIPv2 statistics

Overview

Note

The page is only available if there is an online connection to the device.

This page shows the statistics of the RIP interface.

Description of the displayed values

The table has the following columns:

- **IP address**
  Shows the IP address of the RIPv2 interface
- **Bad packets**
  Number of received RIP packets that were deleted and therefore ignored.
- **Bad Routes**
  Number of routes of valid RIP packets that could not be taken into consideration.
- **Updates Sent**
  Shows how often the router has sent its routing table to its neighbor routers.

NAT Translations

Overview

This page displays the active NAT connections.

Description of the displayed values

The table has the following columns:

- **Interface**
  Shows the IP interface.
- **Inside Local Address**
  Shows the actual address of the device that should be reachable from external.
- **Inside Local Port**
  Shows the port that is assigned to the Inside Local Address.
- **Inside Global Address**
  Shows the address at which the device can be reached from external.
- **Inside Global Port**
  Shows the port that is assigned to the Inside Global Address.
- **Outside Local/Global Address**
  Shows the address of the communications partner.
- **Outside Local/Global Port**
  Displays the port of the external communications partner.
- **Last Use Time**
  Shows the time at which the last packet was transferred.

IPv6 routing

Routing table

Introduction

This page shows the IPv6 routes currently being used.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Destination Network**
  Shows the destination address of this route.
- **Prefix Length**
  Shows the prefix length of this route.
- **Gateway**
  Shows the gateway for this route.
- **Interface**
  Shows the interface for this route.
- **Metric**
  Shows the metric of the route. The higher value, the longer packets require to their destination.
- **Routing Protocol**
  Shows the routing protocol from which the entry in the routing table originates. The following entries are possible:
  - connected: Connected routes
  - Static: Static routes
  - RIPng: Routes via RIPng
  - OSPFv3: Routes via OSPFv3
  - other: Other routes

OSPFv3 Interfaces

Overview

This page shows the configuration of the OSPFv3 interface.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Shows the interface to which the settings relate.
- **Area ID**
  Shows the Area ID to which the OSPF interface belongs.
- **Interface Status**
  Shows the status of the interface.
  - Down
    The interface is not available.
  - Loop back
    Loop back interface
  - Waiting
    Startup and negotiation of the interface.
  - Point to Point
    Point-to-point connection
  - Designated Router
    The router is a designated router and generates network LSAs.
  - Backup D. Router
    The router is the backup router for the designated router.
  - S D. Router
    The Interface has started up. The router is neither a designated nor a designated backup router.
- **OSPF Status**
  Shows the status of OSPF.
  - Enabled: OSPF is enabled on the interface.
  - Disabled: OSPF is disabled on the interface.
- **Designated Router**
  Shows the IPv6 address of the designated router for this OSPFv3 interface.

- **Backup Designated Router**
  Shows the IPv6 address of the designated backup router for this OSPFv3 interface.
- **Events**
  Shows the number of status changes of OSPFv3.

OSPFv3 neighbors

Overview

This page shows the dynamically detected neighbor routers in the relevant networks.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Displays the OSPFv3 interface via which the neighbor router can be reached.
- **Router ID**
  Shows the ID of the neighbor router.
- **Status**
  Shows the status of the neighbor router. The status can adopt the following values:
  - unknown
    Status of the neighbor router is unknown.
  - down
    The neighbor router cannot be reached.
  - attempt und init
    Short-lived status during initialization
  - two-way
    Two-way receipt of Hello packets. Specification of the designated router and the designated backup router.
  - exchangestart, exchange und loading
    Status during exchange of the LSAs
  - full
    The database is complete and synchronized within the area. The routes can now be detected.

    Note

    Normal status

    If the partner router is a designated router or a designated backup router, the status is "full". Otherwise the status is "two-way".

- **Assoc. Area Type**
  Shows the area type via which the neighbor-neighbor relation is maintained. The following area types exist:
  - Standard
  - Stub
  - NSSA
  - Backbone
- **Priority**
  Shows the priority of the neighbor router. This is only significant when selecting the designated router on a network. For virtual neighbor routers, this information is irrelevant.
- **Hello suppr**
  Shows the suppressed Hello packets to the neighbor router. This field normally displays "no".
- **Retrans Queue**
  Shows the length of the queue with Hello packets still to be transmitted.
- **Events**
  Shows the number of status changes.

OSPFv3 Virtual Neighbors

Overview

This page shows the configured virtual neighbors.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Displays the OSPFv3 interface via which the neighbor router can be reached.
- **Router ID**
  Shows the ID of the virtual neighbor router.

- **Status**
  Shows the status of the virtual neighbor router. The status can adopt the following values:
  - unknown
    Status of the neighbor router is unknown.
  - down
    The neighbor router cannot be reached.
  - attempt und init
    Short-lived status during initialization
  - two-way
    Two-way receipt of Hello packets. Specification of the designated router and the designated backup router.
  - exchangestart, exchange und loading
    Status during exchange of the LSAs
  - full
    The database is complete and synchronized within the area. The routes can now be detected.

  Note

  Normal status

  If the partner router is a designated router or a designated backup router, the status is "full". Otherwise the status is "two-way".

- **Trans. Area-ID**
  Shows the ID of the area via which the virtual neighborhood relation exists.
- **Hello suppr**
  Shows the suppressed Hello packets to the neighbor router. This field normally displays "no".
- **Retrans Queue**
  Shows the length of the queue with Hello packets still to be transmitted.
- **Events**
  Shows the number of status changes.

OSPFv3 AS-Scope LSDB

Overview

The AS-Scope LSDB consists of AS external LSAs. The most important data of these LSAs is shown on this page.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Link State Type**
  - AS External
    The AS external LSA (Type 0x4005) is sent by the AS border router within an autonomous system. The LSA contains information about routes from one network to another.
- **Link State ID**
  Shows the ID of the LSA.
- **Router ID**
  Shows the ID of the router that sent this LSA.
- **Sequence Number**
  Shows the sequence number of the LSA. Each time an LSA is renewed, this sequence number is incremented by one.

OSPFv3 Area-Scope LSDB

Overview

The Area-Scope LSDB is the central database for managing all links within an area. The most important data of the LSAs is shown on this page.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Area ID**
  Shows the ID of the area to which the LSA belongs. If the LSA is an external connection, '-' is displayed.
- **Link State Type**
  Shows the LSA type. The following values are possible:
  - Router
    The router LSA (Type 0x2001) is sent by the OSPF router within an area. The LSA contains information about the status of all router interfaces. However, no longer contains address information. This is contained in the new LSA type 0x2009.
  - Network
    The network LSA (Type 0x2002) is sent by the designated router within an area. The LSA contains a list of routers connected to the network. However, no longer contains address information. This is contained in the new LSA type 2009.
  - Inter-Area Prefix
    the inter-area prefix LSA (Type 0x2003) is sent by the ABR within an area. The LSA contains information about routes to other networks.
  - Inter-Area Router
    The inter-area router LSA (Type 0x2004) is sent by the area border router within an area. The LSA contains information about routes to other autonomous systems.
  - Type 7
    The Type 7 LSA (Type 0x2007) is sent by the NSSA-ASBR within an NSSA. The NSSA-ASBR receives LSAs of Type 0x4005 and converts the information to LSAs of Type 0x2007. The NSSA router can forward these LSAs within an NSSA.
  - Intra-Area Prefix
    The intra-area prefix LSA (Type 0x2009) is only sent within an area. It contains the IPv6 prefixes connected to the router or network.
- **Link State ID**
  Shows the ID of the LSA.
- **Router ID**
  Shows the ID of the router that sent this LSA.
- **Sequence Number**
  Shows the sequence number of the LSA. Each time an LSA is renewed, this sequence number is incremented by one.

OSPFv3 Link Scope LSDB

Overview

The link scope LSDB consists of the link LSAs. The most important data of these LSAs is shown on this page.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Shows the OSPFv3 interface to which the link LSA belongs.
- **Link State Type**
  - Link
    The link LSA (Type 0x2009) is sent by the router to every router linked to it. It contains the link local address of the router and a list with IPv6 prefixes configured on the link.
- **Link State ID**
  Shows the ID of the LSA.
- **Router ID**
  Shows the ID of the router that sent this LSA.
- **Sequence Number**
  Shows the sequence number of the LSA. Each time an LSA is renewed, this sequence number is incremented by one.

RIPng Statistics

Overview

This page shows the statistics of the RIPng interfaces.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Shows the RIPng interface.
- **Status**
  Shows the status of the RIPng interface.
  - up
  - down
- **Messages Received**
  Shows how often the router has received messages.
- **Requests Received**
  Shows how often the router has received requests.
- **Responses Received**
  Shows how often the router has received responses.
- **Unknown Commands**
  Shows how many RIPng packets were received whose value in the "Commands" field of the RIPng header is unknown. Known values are 1 for requests and 2 for responses.
- **Other version**
  Shows how many RIPng packets were received whose value in the "Version" field is not 1.
- **Discards**
  Shows how often the router has discarded RIPng packets.
- **Messages Sent**
  Shows how often the router has sent messages.
- **Requests Sent**
  Shows how often the router has sent requests.
- **Responses Sent**
  Shows how often the router has sent responses.
- **Updates Sent**
  Shows how often the router has sent its routing table to its neighbor routers.

Security

Overview

This page shows the security settings and the local and external user accounts.

Note

The page is only available if there is an online connection to the device.

The values displayed depend on the rights of the logged-on user.

Displayed values

Services

The "Services" list shows the security settings.

- **Telnet Server**
  You configure the setting in "System > Configuration".
  - Enabled: Unencrypted access to the CLI.
  - Disabled: No unencrypted access to the CLI.
- **SSH Server**
  You configure the setting in "System > Configuration".
  - Enabled: Encrypted access to the CLI.
  - Disabled: No encrypted access to the CLI.
- **Web Server**
  You configure the setting in "System > Configuration".
  - HTTP/HTTPS: Access to the WBM is possible with HTTP and HTTPS.
  - HTTPS: Access to the WBM is now only possible with HTTPS.
- **SNMP**
  You can configure setting in "System > SNMP > General".
  - "-" (SNMP disabled)
    Access to device parameters via SNMP is not possible.
  - SNMPv1/v2c/v3
    Access to device parameters is possible with SNMP versions 1, 2c or 3.
  - SNMPv3
    Access to device parameters is possible only with SNMP version 3.
- **Management ACL**
  You configure the setting in "Security > Management ACL".
  - Disabled: No access restriction
    The access control is disabled.
  - Enabled: No access restriction
    The access control is enabled but no access rules have been defined.
  - Enabled: Restricted access only
    The access control is enabled and access rules have been defined.

- **Login Authentication**
  You configure the setting in "Security > AAA > General".
  - Local
    The authentication must be made locally on the device.
  - RADIUS
    The authentication must be handled via a RADIUS server.
  - Local and RADIUS
    The authentication is possible both with the users that exist on the device (user name and password) and via a RADIUS server.
    The user is first searched for in the local database. If the user does not exist there, a RADIUS query is sent.
  - RADIUS and fallback local
    The authentication must be handled via a RADIUS server.
    A local authentication is performed only when the RADIUS server cannot be reached in the network.
- **Password Policy**
  Shows which password policy is currently being used.

Local and external user accounts

You configure local user accounts and roles in "Security > User Accounts"

When you create a local user account an external user account is generated automatically. Local user accounts involve users each with a password for logging in on the device.

In the table "External User Accounts" a user is linked to a role, for example the user "Observer" with the role "user" In this case, the user is defined on a RADIUS server. The roll is defined locally on the device. When a RADIUS server authenticates a user, the corresponding group however is unknown or does not exist, the device checks whether or not there is an entry for the user in the table "External User Accounts". If an entry exists, the user is logged in with the rights of the associated role. If the corresponding group is known on the device, both tables are evaluated. The user is assigned the role with the higher rights.

Note

The table "External User Accounts" is only evaluated if you have set "Vendor Specific" in the RADIUS Authorization Mode".

With CLI you can access external user accounts.

The table "Local User Accounts" has the following columns:

- **User Account**
  Shows the name of the local user.
- **Role**
  Shows the role of the user. You can obtain more information on the function rights of the role in "Information > Security > Roles".

The table "External User Accounts" has the following columns:

- **User Account**
  Shows the name of the user on the RADIUS server.
- **Role**
  Shows the role assigned to the user on the device. You can obtain more information on this in "Information > Security > Roles".

Supported Function Rights

The page shows the function rights available locally on the device.

Note

The page is only available if there is an online connection to the device.

The values displayed depend on the role of the logged-on user.

Displayed values

- **Function Right**
  Shows the number of the function right. Different rights relating to the device parameters are assigned to the numbers.
- **Description**
  Shows the description of the function right.

Groups

This page shows which group is linked to which role. The group is defined on a RADIUS server. The roll is defined locally on the device.

Note

The page is only available if there is an online connection to the device.

The values displayed depend on the role of the logged-on user.

Displayed values

The table has the following columns:

- **Group**
  Shows the name of the group. The name matches the group on the RADIUS server.
- **Role**
  Shows the role. Users who are authenticated with the linked group on the RADIUS server receive the rights of this role locally on the device.
- **Description**
  Shows a a description for the link.

Roles

The page shows the roles valid locally on the device.

Note

The page is only available if there is an online connection to the device.

The values displayed depend on the role of the logged-on user.

Displayed values

This table contains the following columns:

- **Role**
  Shows the role.
- **Function Right**
  Shows the function rights of the role:
  - 1
    Users with this role can read device parameters but cannot change them.
  - 15
    Users with this role can both read and change device parameters.
  - 0
    This is a role that the device assigns internally when the user could not be authenticated. The user is denied access to the device.
- **Description**
  Shows a description of the role.

Configuring system functions

Configuration

The page contains the configuration overview of the access options of the device. Specify the services that access the device. With some services, there are further configuration pages on which more detailed settings can be made.

Settings

- **Telnet Server**
  Enable or disable the "Telnet Server" service for unencrypted access to the CLI.
- **SSH server**
  Enable or disable the "SSH Server" service for encrypted access to the CLI.
- **HTTPS server only**
  If this function is enabled, you can only access the device via HTTPS.

- **DNS client**
  Enable or disable depending on whether the IE switch should operate as a DNS client. You can configure other settings in "System > DNS Client".
- **SMTP Client**
  Enable or disable the SMTP client. You can configure other settings in "System > SMTP Client".
- **Syslog Client**
  Enable or disable the Syslog client. You can configure other settings in "System > Syslog client".
- **DCP Server**
  Specify whether or not the device can be accessed with DCP (Discovery and Configuration Protocol):
  - "-" (disabled)
    DCP is disabled. Device parameters can neither be read nor modified.
  - Read/write access
    With DCP, device parameters can be both read and modified.
  - Read-Only
    With DCP, device parameters can be read but cannot be modified.
- **Time**
  Select the required setting. The following settings are possible:
  - Manual
    The system time is set manually. You can configure other settings in "System > System Time > Manual Setting".
  - SNTP Client
    The system time is set via an SNTP server. You can configure other settings in "System > System time > SNTP client".
  - NTP Client
    The system time is set via an NTP server. You can configure other settings in "System > System time > NTP client".
  - SIMATIC Time
    The system time is set using a SIMATIC time transmitter. You can configure other settings in "System > System Time SIMATIC Time Client".
  - PTP Client(SCALANCE XR528-6M and XR552-12M only)
    The system time is set via a PTP server. You can configure other settings in "System > System Time > PTP Client".

- **SNMP**
  Select the required protocol. The following settings are possible:
  - "-" (SNMP disabled)
    Access to device parameters using SNMP is not possible.
  - SNMPv1/v2c/v3
    Access to device parameters is possible with SNMP versions 1, 2c or 3. You can configure other settings in "System > SNMP > General".
  - SNMPv3
    Access to device parameters is possible with SNMP version 3. You can configure other settings in "System > SNMP > General".
- **SNMPv1/v2 Read-Only**
  Enable or disable write access to SNMP variables with SNMPv1/v2c.
- **SNMPv1 traps**
  Enable or disable the sending of SNMP traps (alarm frames). You can configure other settings in "System > SNMP > Traps".
- **SINEMA configuration interface**
  If the SINEMA configuration interface is enabled, you can download configurations to the IE switch via the TIA Portal.
- **NFC** (only with SCALANCE XM-400)
  Enable or disable the "NFC" function.
  You will find further information on NFC in the "SCALANCE XM400" Operating Instructions.
- **Configuration mode**
  Select the required operating mode. The following modes are possible:
  - Automatic save
    Automatic backup mode. Approximately 1 minute after the last parameter change or when you restart the device, the configuration is automatically saved.
  - Trial
    In the "Trial" configuration mode, although changes are adopted, they are not saved in the configuration file (startup configuration).
    To save changes in the configuration file, use the "Write startup config" button. The "Write startup config" button is displayed when you set trial mode. The display area also shows the message "Trial Mode Active – Click the "Write Startup Config" button to make your settings persistent" as soon as there are unsaved modifications. This message can be seen on every page until the changes made have either been saved or the device has been restarted.

General

Device

This page contains the general device information.

Settings

- **Current System Time**(Only available online)
  Shows the current system time. The system time is either set by the user or by a time-of-day frame: either SINEC H1 time-of-day frame, NTP or SNTP. (readonly)
- **System Up Time**(only available online)
  Shows the operating time of the device since the last restart. (readonly)
- **Device Type**(only available online)
  Shows the type designation of the device. (readonly)
- **System Name**
  You can enter the name of the device. The entered name is displayed in the selection area. A maximum of 255 characters are possible. The system name is also displayed in the CLI input prompt. The number of characters in the CLI input prompt is limited. The system name is truncated after 16 characters.
- **System Contact**
  You can enter the name of a contact person responsible for managing the device.
- **Location**
  You can enter the location of the device. The entered installation location is displayed in the selection area.

Note

The ASCII characters 0x20 to 0x7e are used in the input boxes.

At the start and end of the input boxes"System name", "System Contact" and "System Location", the characters "<", ">" and "space" are not permitted.

Coordinates

On this page, you configure the geographic coordinates (latitude, longitude and the height above the ellipsoid according to WGS84). These boxes are purely for information with a maximum length of 32 characters.

Getting the coordinates

Use suitable maps for obtaining the geographic coordinates of the device.

The geographic coordinates can also be obtained using a GPS receiver. The geographic coordinates of these devices are normally displayed directly and only need to be entered in the input boxes of this page.

Settings

- **Latitude**
  Enter the north or south latitude for the location of the device.
  For example, +49° 1´ 31.67" means that the device is located at 49 degrees, 1 arc minute and 31.67 arc seconds north latitude.
  A south latitude is shown by a preceding minus character.
  You can also append the letters N (north latitude) or S (south latitude) to the numeric information (49° 1´ 31.67" N).
- **Longitude**
  Enter the value of the eastern or western longitude of the location of the device.
  For example, +8° 20´ 58.73" means that the device is located at 8 degrees, 20 minutes and 58.73 seconds east.
  A western longitude is indicated by a preceding minus sign.
  You can also add the letter E (eastern longitude) or W (western longitude) to the numeric information (8° 20´ 58.73" E).
- **Height**
  Enter the value of the height above sea level in meters.
  For example, 158 m means that the device is located at a height of 158 m above sea level.
  Heights below sea level (for example the Dead Sea) are indicated by a preceding minus sign.

Agent IP

Configuration of the IP addresses

Note

The page is only available if there is an online connection to the device.

On this page, you specify the IP configuration for the device.

With devices with more than one IP interface, this call references the "Subnets > Configuration" menu item in the "Layer 3 (IPv4)" menu and the configuration of the TIA interface there.

Settings

The page contains the following boxes:

- **IP Assignment Method**
  Shows how the IP address is assigned.
  - Static
    The IP address is static. You enter the IP settings in the input boxes "IP Address" and "Subnet Mask".
  - Dynamic (DHCP)
    The device obtains a dynamic IP address from a DHCP server.
- **IP Address**
  Enter the IP address of the device.
  After clicking the "Set Values" button, this IP address is also displayed in the address bar of the Internet browser. If this does not take place automatically, you will need to enter the IP address in the address bar of the Internet browser manually.
- **Subnet Mask**
  Enter the subnet mask of the device.
- **Default gateway**
  Enter the IP address of the default gateway to be able to communicate with devices in another subnet, for example diagnostics stations, e-mail server.
- **Agent VLAN ID**
  Select the VLAN ID from the drop-down list. You can only select VLANs that have already been configured.
  In the mode "802.1D Transparent Bridge", this drop-down list is grayed out, see also "Layer 2 > VLAN > General".

  Note

  Changing the Agent VLAN ID

  If the configuration PC is connected directly to the device via Ethernet and you change the Agent VLAN ID , the device is no longer reachable via Ethernet following the change.
- **MAC Address**
  Shows the MAC address of the device. The MAC address is linked to the hardware and cannot be modified.

DNS client

On this page, you can configure up to 3 DNS servers. If there is more than one server, the order in the table specifies the order in which the servers are queried. The top server is queried first.

The DNS () server (Domain Name System) assigns a domain name to an IP address so that a device can be uniquely identified.

If this function is enabled, the IE switch can communicate with a DNS server as a DNS client. You have the option of entering names in IP address boxes.

Note

The DNS client function can only be used if there is a DNS server in the network.

Settings

The page contains the following boxes:

- **DNS Client**
  Enable or disable depending on whether the IE switch should operate as a DNS client.
- **Used DNS Servers**
  Specify which DNS server the device uses:
  - learned only
    The device uses only the DNS servers assigned by DHCP.
  - manual only
    The device uses only the manually configured DNS servers. The DNS servers must be connected to the Internet. A maximum of three DNS servers can be configured.
  - all
    The device uses all available DNS servers.
- **DNS Server IP Address**
  Enter the IP address of the DNS server.
- **Origin**
  Shows whether the DNS server was configured manually or was assigned by DHCP.

The table contains the following columns:

- **DNS Server Address**
  Shows the IP address of the DNS server.
- **Origin**
  Shows whether the DNS server was configured manually or was assigned by DHCP.

Restart

On this page, there is a button with which you can restart the device and the option of resetting to the factory settings.

Note

The page is only available if there is an online connection to the device.

Restart

Note the following points about restarting a device:

- You can only restart the device with administrator privileges.
- A device should only be restarted with the buttons of this menu or with the appropriate CLI commands and not by a power cycle on the device.
- Any modifications you have made only become active on the device after clicking the "Set values" button on the relevant WBM page. If the device is in "Trial" mode, configuration modifications must be saved manually before a restart. In "Automatic Save" mode, the last changes are saved automatically before a restart.

Reset to Factory Defaults

By resetting all the settings to the factory settings, the IP address and the passwords are also lost. Following this, the device can only be accessed via the serial interface, using the Primary Setup Tool or using DHCP.

NOTICE
With the appropriate attachment, a previously correctly configured device can cause circulating frames and therefore the failure of the data traffic.

Resetting to defaults (profiles)

The profiles provide a preconfiguration for various use cases of the devices.

When you start a device with the default settings of a profile, the settings are reset to the factory settings and some parameters are set so that they are designed for a use case. In contrast to resetting to the factory settings. the users and passwords are retained after the restart. The configured IP address is lost so that device can then only be accessed via the serial interface, using the Primary Setup Tool or using DHCP.

NOTICE
With the appropriate attachment, a previously correctly configured device can cause circulating frames and therefore the failure of the data traffic.

Which settings were set specially for the profile are displayed before the restart.

The profiles can be used independently of the factory setting of the device.

Settings

Note

Note the effects of the individual functions described in the sections above.

To restart the device, the buttons on this page provide you with the following options:

- **Restart**
  Click this button to restart the system. You must confirm the restart in a dialog box. During a restart, the device is reinitialized, the internal firmware is reloaded, and the device runs a self-test. The settings of the start configuration are retained, e.g. the IP address of the device. The learned entries in the address table are deleted. You can leave the browser window open while the device restarts. After the restart you will need to log in again.
- **Restore Factory Defaults and Restart**
  Click this button to restore the factory defaults of the device and to restart the device. You must confirm the restart in a dialog box.
  The factory settings depend on the device.

To restart the device with a predefined profile, the buttons on this page provide you with the following options:

- **PROFINET Defaults**
  Click this button to restore the default settings of the PROFINET profile and to restart the device. You must confirm the restart in a dialog box. The dialog box displays the settings specially made for operation with the PROFINET protocol.
- **EtherNet/IP Defaults**
  Click this button to restore the default settings of the EtherNet/IP profile and to restart the device. You must confirm the restart in a dialog box. The dialog box displays the settings specially made for operation with the EtherNet/IP protocol.
- **Industrial Ethernet Defaults**
  Click this button to restore the default settings of the Industrial Ethernet profile and to restart the device. You must confirm the restart in a dialog box. The dialog box displays the settings specially made for operation in the Industrial Ethernet environment.

Load & save

File types

Overview of the file types

File type	Description
Config	This file contains the start configuration. Among other things, this device contains the definitions of the users, roles, groups and function rights. The passwords are stored the file "Users".
ConfigPack	Detailed configuration information. for example, start configuration, users, certificates ZIP file consisting of the Config, Users and LSYS fle.
Debug	This file contains information for Siemens Support. It is encrypted and can be sent by e-mail to Siemens Support without any security risk.
EDS	Electronic Data Sheet (EDS) Electronic data sheets for describing devices in the EtherNet/IP mode

File type	Description
Firmware	The firmware is signed and encrypted. This ensures that only firmware created by Siemens can be downloaded to the device.
GSDML	PROFINET information on the device properties
HTTPSCert	Default HTTPS certificates including key The preset and automatically created HTTPS certificates are self-signed. We strongly recommend that you create your own HTTPS certificates and make them available. We recommend that you use HTTPS certificates signed either by a reliable external or by an internal certification authority. The HTTPS certificate checks the identity of the device and controls the encrypted data exchange. Certificates with a different format cannot be copied in.
LogFile	File with entries from the event log table
MIB	Private MSPS MIB file
RunningCLI	Text file with CLI commands This file contains an overview of the current configuration in the form of CLI commands. You can download the text file. The file is not intended to be uploaded again unchanged.
Script	Text file with CLI commands You can upload a script file in a device. The CLI commands it contains are executed appropriately.
StartupInfo	Startup log file This file contains the messages that were entered in the log during the last startup.
Users	This file contains the assignment of the user names to the corresponding passwords.
WBMFav	WBM favorite pages This file contains the favorites that you created in the WBM.. You can download this file and upload it in other devices.

Uploading and saving using HTTP

Loading and saving data via HTTP

This page allows you to store device data in an external file on your client PC or to load such data from an external file from the client PC to the devices.
This means, for example, that you can also load new firmware from a file located on your client PC.

Note

The page is only available if there is an online connection to the device.
This page is available both for connections using HTTP and for connections using HTTPS.

Firmware

The firmware is signed and encrypted. This ensures that only firmware created by Siemens can be downloaded to the device.

Note

Incompatibility with predecessor versions

During the installation of a previous version, the configuration data and log files can be lost. In this case, the device starts up with the factory settings after the firmware has been installed.

Incompatibility with previous firmware versions with/without PLUG inserted

During the installation of a previous version, the configuration data can be lost. In this case, the device starts up with the factory settings after the firmware has been installed.

In this situation, if a PLUG is inserted in the device, following the restart, this has the status "Not Accepted" since the PLUG still has the configuration data of the previous more up-to-date firmware. This allows you to return to the previous, more up-to-date firmware without any loss of configuration data. If the original configuration on the PLUG is no longer required, the PLUG can be deleted or rewritten manually using the WBM page "System > PLUG".

Configuration files

Note

Configuration files and trial mode/automatic save mode

In "Automatic Save" mode, the data is saved automatically before the configuration files (ConfigPack and Config) are transferred.
In trial mode, although the changes are adopted, they are not saved in the configuration files (ConfigPack and Config). Use the "Write Startup Config" button on the "System > Configuration" WBM page to save changes in the configuration file.

CLI script file

You can download existing CLI configurations (RunningCLI) and upload your own CLI scripts (Script).

Note

The downloadable CLI script is not intended to be uploaded again unchanged.

Settings

The table has the following columns:

- **Type**
  Shows the designation of the file.
- **Description**
  Shows the short description of the file type.

- **Load**
  With this button, you can upload files to the device. The button can be enabled, if this function is supported by the file type.
- **Save**
  With this button, you can save files from the device. The button can only be enabled if this function is supported by the file type and the file exists on the device.
- **Delete**
  With this button, you can delete files from the device. The button can only be enabled if this function is supported by the file type and the file exists on the device.

Note

Following a firmware update, delete the cache of your Internet browser.

Uploading and saving using TFTP

Loading and saving data via a TFTP server

On this page, you can configure the TFTP server and the file names. You can also store device data in an external file on a TFTP server or load such data from an external file from the TFTP server to the devices. This means, for example, that you can also load new firmware from a file located on a TFTP server.

Firmware

The firmware is signed and encrypted. This ensures that only firmware created by Siemens can be downloaded to the device.

Note

Incompatibility with predecessor versions

During the installation of a previous version, the configuration data and log files can be lost. In this case, the device starts up with the factory settings after the firmware has been installed.

Incompatibility with previous firmware versions with/without PLUG inserted

During the installation of a previous version, the configuration data can be lost. In this case, the device starts up with the factory settings after the firmware has been installed.

In this situation, if a PLUG is inserted in the device, following the restart, this has the status "Not Accepted" since the PLUG still has the configuration data of the previous more up-to-date firmware. This allows you to return to the previous, more up-to-date firmware without any loss of configuration data. If the original configuration on the PLUG is no longer required, the PLUG can be deleted or rewritten manually using the WBM page "System > PLUG".

Configuration files

Note

Configuration files and Trial mode/Automatic Save

In "Automatic Save" configuration mode, the data is saved automatically before the configuration files (ConfigPack and Config) are transferred.
In trial mode , although the changes are adopted, they are not saved in the configuration files (ConfigPack and Config). Use the "Write Startup Config" button on the "System > Configuration" WBM page to save changes in the configuration files.

CLI script file

You can download existing CLI configurations (RunningCLI) and upload your own CLI scripts (Script).

Note

The downloadable CLI script is not intended to be uploaded again unchanged.

Settings

- **TFTP Server IP Address**
  Enter the IP address or the FQDN of the TFTP server with which you exchange data.
- **TFTP Server Port**
  Here, enter the port of the TFTP server via which data exchange will be handled. If necessary, you can change the default value 69 to your own requirements.

The table has the following columns:

- **File type**
  Shows the designation of the file.
- **Description**
  Shows the short description of the file type.

- **File name**
  A file name is preset here for every file type.

  Note

  Changing the file name

  You can change the file name preset in this column. After clicking the "Set Values" button, the changed name is stored on the device and can also be used with the Command Line Interface.

- **Select action** (only available online)
  Select the required action. The selection depends on the selected file type, for example you can only save the log file.
  The following actions are possible:
  - Save file
    With this selection, you save a file on the TFTP server.
  - Load file
    With this selection, you load a file from the TFTP server.

Passwords

There are files to which access is password protected. To load the file on the device, enter the password specified for the file on the WBM page.

Settings

The table has the following columns:

- **Type**
  Shows the file type.
- **Description**
  Shows the short description of the file type.
- **Enabled**
  When selected, the password is used. Can only be enabled if the password is configured.
- **Password**
  Enter the password for the file.

- **Password Confirmation**
  Confirm the new password.
- **Status**
  Shows whether the current settings for the file match the device.
  - Valid
    the "Enabled" setting is selected and the password matches the certificate.
  - Invalid
    the "Enabled" setting is selected but the password does not match the certificate or no certificate has been loaded yet.
  - '-'
    The password cannot be evaluated or is not yet being used. The "Enabled" check box is not selected.

Events

Configuration

On this page, you specify how the device reacts to system events. By enabling the appropriate options, you specify how the device reacts to events. To enable or disable the options, click the relevant settings of the columns.

Settings

- **Signaling Contact Response**
  Select the behavior of the signaling contact. The following reactions are possible:
  - conventional
    Default setting for the signaling contact. An error/fault is displayed by the fault LED and the signaling contact is opened. When the error/fault state no longer exists, the fault LED goes off and the signaling contact is closed.
  - User-defined
    The way the signaling contact works does not depend on the error/fault that has occurred. The signaling contact can be opened or closed as required by user actions.
- **Signaling Contact Status**
  Select the status of the signaling contact from the drop-down list. The following statuses are possible:
  - close
    Signaling contact is closed.
  - open
    Signaling contact is opened.

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports of table 2.
- **E-mail/Trap/Log table/Syslog/Errors**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Event**
  The "Event" column contains the following values:
  - Cold/Warm Start
    The device was turned on or restarted by the user.
  - Link Change
    This event occurs only when the port status is monitored and has changed, see "System > Fault Monitoring > Link Change".
  - Authentication Failure
    This event occurs when attempting access with a bad password.
  - RMON Alarm
    An alarm or event has occurred relating to the remote monitoring of the system.
  - Power Supply Switchover
    This event occurs only when power supply lines 1 and 2 are monitored. It indicates that there was a change to line 1 or line 2, see System > Fault Monitoring > Power Supply".
  - RM status change
    The redundancy manager has recognized an interruption or restoration of the ring and has switched the line over or back.
  - Spanning Tree Change
    The STP or RSTP or MSTP topology has changed.
  - Fault State Change
    The fault state has changed. The fault state can relate to the activated port monitoring, the response of the signaling contact or the power supply monitoring.
  - Standby State Change
    A device with an established standby connection (master or slave) has activated or deactivated the link to the other ring (standby port). The data traffic was redirected from one Ethernet connection (standby port of the master) to the other Ethernet connection (standby port of the slave).
  - VRRP State Change (only with routing using VRRP)
    The status of the virtual router has changed
  - Loop Detection
    A loop was detected in the network segment.
  - OSPF State Change
    The status of OSPF has changed.
  - Pnac Port Authentication State Change
    Port authentication
  - PoE State Change
    The status of PoE has changed.
  - Diagnostics Alarms
    The diagnostics value has fallen below or exceeded a certain limit.
  - FMP Status Change
    The value of the received power or the power loss has fallen below a certain limit.
- **E-mail**

The device sends an e-mail. This is only possible if the SMTP server is set up and the "SMTP client" function is enabled.

- **Trap**
  The device sends an SNMP trap. This is only possible if "SNMPv1 traps" is enabled in "System > Configuration".
- **Log table**
  The device writes an entry in the event log table, see "Information > Log Table"
- **Syslog**
  The device writes an entry to the system log server. This is only possible if the system log server is set up and the "Syslog client" function is enabled.
- **Fault**
  The device triggers a fault. The error LED lights up

Severity filter

On this page, set the threshold levels for sending system event notifications.

Settings

The table has the following columns:

- **Client Type**
  Select the client type for which you want to make settings:
  - E-mail
    Sending system event messages by e-mail
  - Log table
    Entry of system events in the log table.
  - Syslog
    Entering system events in the Syslog file
  - WLAN Authentication Log
    Forwarding of the entries from the WLAN Authentication Log to the system protocol server.
    Can only be configured with SCALANCE W770 and SCALANCE W780
- **Severity**
  Select the required level. The following settings are possible:
  - Info
    System events are processed as of the severity level "Info".
  - Warning
    System events are processed as of the severity level "Warning".
  - Critical
    System events are processed as of the severity level "Critical".

SMTP client

The device provides the option of automatically sending an e-mail if an alarm event occurs (for example to the network administrator). The e-mail contains the identification of the sending device, a description of the cause of the alarm in plain language, and a time stamp. This allows centralized network monitoring to be set up for networks with few nodes based on an E-mail system. When an e-mail error message is received, the WBM can be started by the Internet browser using the identification of the sender to read out further diagnostics information.

On this page, you can configure up to three SMTP servers and the corresponding e-mail addresses.

Settings

- **SMTP Client**
  Enable or disable the SMTP client.
- **Sender Email Address**
  Enter the name of the sender to be included in the e-mail, for example the device name. This setting applies to all configured SMTP servers.
- **Send Test Mail**
  Send a test e-mail to check your configuration.
- **SMTP Port**
  Enter the port via which your SMTP server can be reached.
  Factory settings: 25
  This setting applies to all configured SMTP servers.
- **SMTP Server Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the SMTP server.

This table contains the following columns:

- **SMTP Server Address**
  Shows the IP address or the FQDN (Fully Qualified Domain Name) of the SMTP server.
- **Email address of the recipient**
  Enter the e-mail address to which the device sends an e-mail if a fault occurs.

Syntax of email addresses

The following conditions apply to e-mail addresses:

- Alphanumeric characters are permitted.
- The following special characters are permitted:
  - @
  - _ (underscore)
  - - (hyphen)
  - . (period)
- A @ character must be included.

- Only one @ may be used.
- The characters "@" and the "." must not be the first or last character.

DHCP

DHCP client

If the device is configured as a DHCP client, it starts a DHCP query. As the reply to the query the device receives an IPv4 address from the DHCP server. The server manages an address range from which it assigns IPv4 addresses. It is also possible to configure the server so that the client always receives the same IPv4 address in response to its request.

Settings

- **DHCP client configuration file request (opt 66, 67)**
  Select this option if you want the DHCP client to use options 66 and 67 to download and then enable a configuration file.
- **DHCP Mode**
  Select the DHCP mode. The following modes are possible:
  - via MAC Address
    Identification is based on the MAC address.
  - via DHCP Client ID
    Identification is based on a freely defined DHCP client ID.
  - Via System Name
    Identification is based on the system name. If the device name is 255 characters long, the last character is not used for identification.
  - via PROFINET Name of Station
    Identification is based on the PROFINET name of the station.

This table contains the following columns:

- **Interface**
  Interface to which the setting relates.
- **DHCP**
  Enable or disable the DHCP client for the relevant interface.

DHCP Server

You can operate the device as a DHCP server. This allows IP addresses to be assigned automatically to the connected devices. The IP addresses are either distributed dynamically from an address band (pool) you have specified or a specific IP address is assigned to a particular device.

On this page, specify the address band from which the connected device receives any IP address. You configure the static assignment of the IP addresses in "Static Leases".

Requirement

- The connected devices are configured so that they obtain the IP address from a DHCP server.

Description

The page contains the following boxes:

- **DHCP Server**
  Enable or disable the DHCP server on the device.

  Note

  To avoid conflicts with IPv4 addresses, only one device may be configured as a DHCP server in the network.

- **Probe address with ICMP echo before offer**
  When selected, the DHCP server checks whether or not the IP address has already been assigned. To do this the DHCP server sends ICMP echo messages (ping) to the IPv4 address. If no reply is received, the IPv4 address is assigned.

  Note

  If there are devices in your network on which the echo service is disabled as default, there may be conflicts with the IPv4 addresses. To avoid this, assign these devices an IPv4 address outside the IPv4 address band.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Pool ID**
  Shows the number of the IPv4 address band. If you click the "Create" button, a new row with a unique number is created (pool ID).
- **Interface**
  Select a VLAN IP interface. The IPv4 addresses are assigned dynamically via this interface. The requirement for the assignment is that the IPv4 address of the interface is located in the subnet of the IPv4 address band. If this is not the case, the interface does not assign any IPv4 addresses.
- **Enable**
  Specify whether or not this IPv4 address band will be used.

  Note

  If you enable the IPv4 address band, its settings in this and the other DHCP tabs are grayed out and can no longer be edited.

- **Subnet**
  Enter the network address range that will be assigned to the devices. Use the CIDR notation.
- **Lower IP Address**
  Enter the IPv4 address that specifies the start of the dynamic IPv4 address band. The IPv4 address must be within the network address range you configured for "Subnet".

- **Upper IP address**
  Enter the IPv4 address that specifies the end of the dynamic IPv4 address band. The IPv4 address must be within the network address range you configured for "Subnet".
- **Lease Time (sec)**
  Specify for how many seconds the assigned IPv4 address remains valid. When half the period of validity has elapsed. the DHCP client can extend the period of the assigned IPv4 address. When the entire time has elapsed, the DHCP client needs to request a new IPv4 address.

Port Range

On this page, you define the ports via which the IPv4 addresses of an address band are assigned.

After you have created an IPv4 address band in the "DHCP Server" tab, a new line is created in this tab and all ports selected that are currently located in the corresponding VLAN. If you add ports to the VLAN later, the ports are not automatically enabled in this tab.

Description

This table contains the following columns:

- **Pool ID**
  Shows the number of the IPv4 address band. A line is created for every address band.
- **Interface**
  Shows the assigned VLAN IP interface.
- **All ports**
  Select the setting from the drop-down list. You have the following setting options:
  - Enabled
    The check box is enabled for all ports.
  - Disabled
    The check box is disabled for all ports.
  - No Change
    The table remains unchanged.
- **Px.y**
  Specify the ports via which IPv4 addresses of the address band will be assigned.
  You can only select ports located in the corresponding VLAN.

DHCP Options

On this page you specify which DHCP options the DHCP server supports. The various DHCP options are defined in RFC 2132.

The DHCP options 1, 3, 6, 66 and 67 are created automatically when the IPv4 address band is created. With the exception of DHCP option 1, the options can be deleted. With DHCP option 1 the subnet mask is set automatically that you entered for the address band in "DHCP Server". With the DHCP option 3, you can set the internal IPv4 address of the device as a DHCP parameter using a check box.

Description

The page contains the following boxes:

- **Pool ID**
  Select the required IPv4 address band.
- **Option Code**
  Enter the number of the required DHCP option. The various DHCP options are defined in RFC 2132. The supported DHCP options are listed in the following paragraph.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Pool ID**
  Shows the number of the IPv4 address band.
- **Option Code**
  Shows the number of the DHCP option.
- **Use Interface IP**
  If you enable the check box, the IPv4 address is used as the default gateway that is assigned to the VLAN IP address. If the check box is disabled, you can enter an IPv4 address.
- **Value**
  Enter the DHCP parameter that is transferred to the DHCP client. The content depends on the DHCP option.
  - DHCP option 3 (default gateway):
    Enter the DHCP parameter as an IPv4 address, e.g. 192.168.100.2.
  - DHCP option 6 (DNS):
    Enter the DHCP parameter as an IPv4 address, e.g. 192.168.100.2. You can specify up to three IPv4 addresses separated by commas.
  - DHCP option 66 (TFTP Server):
    Enter the DHCP parameter as an IPv4 address, e.g. 192.168.100.2.
  - DHCP option 67 (boot file name)
    Enter the name of the boot file in the string format.
  - DHCP option 12 (host name)
    Enter the host name in the string format.

DHCP options supported

The following DHCP options are supported:

- Option 1
- Option 3
- Option 6
- Option 12
- Option 66
- Option 67

Relay Agent Information

On this page you define that devices with a certain remote ID and circuit ID are assigned the IPv4 addresses from a specific address band.

If you create such an entry for an address band, the ports of the address band only react to DHCP queries via a DHCP relay agent (option 82). You can create further address bands for the same VLAN IP interfaces so that ports react to different requests.

Note

Extension or release of an IPv4 address assigned via a relay agent.

With address assignments via a relay agent "Renew" and "Release" messages going directly from the DHCP client to the DHCP server are ignored by the server.

- The extension of the period for an IPv4 address assigned via a relay agent is achieved using a "Rebinding" message that the client sends automatically as a broadcast.
- To speed up the release of an IPv4 address assigned via a relay agent, configure a shorter period of validity.

Description

The page contains the following boxes:

- **Pool ID**
  Select the required IPv4 address band.
- **Remote ID**
  Enter the remote ID.
- **Circuit ID**
  Enter the circuit ID.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Pool ID**
  Shows the number of the IPv4 address band.
- **Remote ID**
  Shows the remote ID.
- **Circuit ID**
  Shows the circuit ID.

Static Leases

On this page you define that DHCP clients are assigned a preset IPv4 address depending on their client ID or MAC address.

Description

The page contains the following boxes:

- **Pool ID**
  Select the required IPv4 address band.
- **Client identification method**
  Select the method according to which a client is identified.
  - Ethernet MAC
    The client is identified by its MAC address.
  - Client ID
    The client is identified by a freely defined DHCP client ID.
- **Value**
  Enter the MAC address (Ethernet MAC) or the client ID of the client.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Pool ID**
  Shows the number of the IPv4 address band.
- **Identification method**
  Shows whether the client is identified by its MAC address or the client ID.
- **Value**
  Shows the MAC address or client ID of the client.
- **IP Address**
  Specify the IPv4 address that will be assigned to the client. The IPv4 address must be within the IPv4 address band.

SNMP

General

On this page, you make the basic settings for SNMP. Enable the options according to the function you want to use.

Settings

- **SNMP**
  Select the SNMP protocol. The following settings are possible:
  - "-" (disabled)
    SNMP is disabled.
  - SNMPv1/v2c/v3
    SNMPv1/v2c/v3 is supported.
  - SNMPv3
    Only SNMPv3 is supported.
- **SNMPv1/v2 Read-Only**
  If you enable this option, SNMPv1/v2c can only read the SNMP variables.

  Note

  Community String

  For security reasons, do not use the standard values "public" or "private". Change the community strings following the initial installation.

- **SNMPv1/v2c Read Community String**
  Enter the community string for access of the SNMP protocol.
- **SNMPv1/v2c Read/Write Community String**
  Enter the community string for read and write access of the SNMP protocol.
- **SNMPv1 Traps**
  Enable or disable the sending of SNMP traps (alarm frames). On the "Trap" tab, specify the IP addresses of the devices to which SNMP traps will be sent.
- **SNMPv1/v2c Trap Community String**
  Enter the community string for sending SNMPv1/v2 messages.
- **SNMPv3 User Migration**
  - **Enabled**
    If the function is enabled, an SNMP engine ID is generated that can be migrated. You can transfer configured SNMPv3 users to a different device.
    If you enable this function and load the configuration of the device on another device, configured SNMPv3 users are retained.
  - **Disabled**
    If the function is disabled, a device-specific SNMP engine ID is generated. To generate the ID, the agent MAC address of the device is used. You cannot transfer this SNMP user configuration to other devices.
    If you load the configuration of the device on another device, all configured SNMPv3 users are deleted.
- **SNMP Engine ID**
  Shows the SNMP engine ID.

Traps

If an alarm event occurs, a device can send SNMP traps (alarm frames) to up to ten different management stations at the same time. Traps are only sent if the events specified in "Events" occur.

Note

SNMP traps are sent only when the "SNMPv1 Traps" setting was selected in "SNMP > General".

Settings

- **Trap Receiver Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the station to which the device sends SNMP traps. You can specify up to ten different recipients.

The table has the following columns:

- **Trap Receiver Address**
  If necessary, change the IP addresses or the FQDN of the stations.
- **Trap**
  Enable or disable the sending of SNMP traps. Stations that are entered but not selected do not receive SNMP traps.

v3 Groups

Security settings and assigning permissions

SNMP version 3 allows permissions to be assigned, authentication, and encryption at protocol level. The security levels and read/write permissions are assigned according to groups. The settings automatically apply to every member of a group.

Settings

- **Group Name**
  Enter the name of the group. The maximum length is 32 characters.
- **Security Level**
  Select the security level (authentication, encryption) valid for the selected group. You have the following options for the security levels:
  - no Auth/no Priv
    No authentication enabled / no encryption enabled.
  - Auth/no Priv
    Authentication enabled / no encryption enabled.
  - Auth/Priv
    Authentication enabled / encryption enabled.

The table has the following columns:

- **Group Name**
  Shows the defined group names.

  Note

  Once a group name and the security level have been specified, they can no longer be modified after the group is created. If you want to change the group name or the security level , you will need to delete the group and recreate it and reconfigure it with the new name.

- **Security Level**
  Shows the configured security level.
- **Read**
  Enable or disable read access for the required group.
- **Write**
  Enable or disable write access for the required group.

  Note

  For write access to work, you also need to enable read access.

- **Persistence**
  Shows whether or not the group is assigned to an SNMPv3 user. If the group is not assigned to an SNMPv3 user, no automatic saving is triggered and the configured group is deleted after restarting the device.
  - Yes
    The group is assigned to an SNMPV3 user.
  - No
    The group is not assigned to an SNMPV3 user.

v3 users

User-specific security settings

On the WBM page, you can create new SNMPv3 users and modify or delete existing users. The user-based security model works with the concept of the user name; in other words, a user ID is added to every frame. This user name and the applicable security settings are checked by both the sender and recipient.

Settings

- **User name**
  Enter a freely selectable user name. After you have entered the data, you can no longer modify the name.

The table has the following columns:

- **User name**
  Shows the created users.
- **Group Name**
  Select the group to which the user will be assigned.
- **Authentication Protocol**
  Specify the authentication protocol for which a password will be stored.
  The following settings are available:
  - None
  - MD5
  - SHA
- **Encryption Protocol**
  Specify whether or not a password should be stored for encryption with the DES algorithm. Can only be enabled when an authentication protocol has been selected.
- **Authentication Password**
  Enter the authentication password. This password must have at least 6 characters, the maximum length is 32 characters.
- **Confirm authentication password**
  Confirm the password by repeating the entry.
- **Encryption Password**
  Enter your encryption password. This password must have at least 6 characters, the maximum length is 32 characters.
- **Confirm Encryption Password**
  Confirm the encryption password by repeating the entry.
- **Persistence**
  Shows whether or not the user is assigned to an SNMPv3 group. If the user is not assigned to an SNMPv3 group, no automatic saving is triggered and the configured user is deleted after restarting the device.
  - Yes
    The user is assigned to an SNMPv3 group.
  - No
    The user is not assigned to an SNMPv3 group.

System time

Manual time setting

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

On this page, you set the date and time of the system. For this setting to be used, enable "Manual time setting".

Note

The page is only available if there is an online connection to the device.

Settings

- **Time Manually**
  Enable or disable the manual time setting. If you enable the option, the "System Time" input box can be edited.
- **System Time**
  Enter the date and time in the format "MM/DD/YYYY HH:MM:SS".
- **Use PC Time**
  Click the button to use the time setting of the PC.
- **Last Synchronization Time**
  Shows when the last time-of-day synchronization took place.
  If no time-of-day synchronization was possible, the box displays "Date/time not set".
- **Last Synchronization Mechanism**
  This box displays how the last time-of-day synchronization was performed.
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame.
  - PTP
    Automatic time-of-day synchronization with PTP

DST overview

On this page, you can create new entries for the daylight saving time changeover.

The table provides an overview of the existing entries.

Settings

- **DST No.**
  Shows the number of the entry.
  If you create a new entry, a new line with a unique number is created.
- **Name**
  Shows the name of the entry.
- **Year**
  Shows the year for which the entry was created.
- **Start Date**
  Shows the month, day and time for the start of daylight saving time.
- **End Date**
  Shows the month, day and time for the end of daylight saving time.
- **Recurring Date**
  With an entry of the type "Rule", the period in which daylight saving time is active is displayed consisting of week, day, month and time of day. With an entry of the type "Date" a "-" is displayed.
- **State**
  Shows the status of the entry:
  - enabled
    The entry was created correctly.
  - invalid
    The entry was created new and the start and end date are identical.
- **Type**
  Shows how the daylight saving time changeover is made:
  - Date
    A fixed date is entered for the daylight saving time changeover.
  - Recurring
    A rule was defined for the daylight saving time changeover.

DST configuration

On this page, you can configure the entries for the daylight saving time changeover. As result of the changeover to daylight saving or standard time, the system time for the local time zone is correctly set.

You can define a rule for the daylight saving time changeover or specify a fixed date.

Settings

Note

The content of this page depends on the selection in the "Type" box.

The boxes "DST No.", "Type" and "Name" are always shown.

- **DST No.**
  Select the type of the entry.
- **Type**
  Select how the daylight saving time changeover is made:
  - Date
    You can set a fixed date for the daylight saving time changeover.
    This setting is suitable for regions in which the daylight saving time changeover is not governed by rules.
  - Recurring
    You can define a rule for the daylight saving time changeover.
    This setting is suitable for regions in which the daylight saving time always begins or ends on a certain weekday.
- **Name**
  Enter a name for the entry.

Settings with "Date" selected

You can set a fixed date for the start and end of daylight saving time.

- **Year**
  Enter the year for the daylight saving time changeover.
- **Start Date**
  Enter the following values for the start of daylight saving time:
  - Day
    Specify the day.
  - Hour
    Specify the hour.
  - Month
    Specify the month.
- **End Date**
  Enter the following values for the end of daylight saving time:
  - Day
    Specify the day.
  - Hour
    Specify the hour.
  - Month
    Specify the month.

Settings with "Recurring" selected

You can create a rule for the daylight saving time changeover.

- **Start Date**
  Enter the following values for the start of daylight saving time:
  - Hour
    Specify the hour.
  - Month
    Specify the month.
  - Week
    Specify the week.
    You can select the 1st to 4th or the last week of the month.
  - Weekday
    Specify the weekday.
- **End Date**
  Enter the following values for the end of daylight saving time:
  - Hour
    Specify the hour.
  - Month
    Specify the month.
  - Week
    Specify the week.
    You can select the 1st to 4th or the last week of the month.
  - Weekday
    Specify the weekday.

SNTP client

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

SNTP (Simple Network Time Protocol) is used for synchronizing the time in the network. The appropriate frames are sent by an SNTP server in the network.

Note

To avoid time jumps, make sure that there is only one time server in the network.

Settings

- **SNTP Client**
  Enable or disable automatic time-of-day synchronization using SNTP.
- **Current System Time** (only available online)
  Shows the current date and current normal time received by the IE switch. If you specify a time zone, the time information is adapted accordingly.

- **Last Synchronization Time**(only available online)
  Shows when the last time-of-day synchronization took place.
- **Last Synchronization Mechanism** (only available online)
  Shows how the last time-of-day synchronization was performed. The following methods are possible:
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame
  - PTP
    Automatic time-of-day synchronization with PTP
- **Time Zone**
  Enter the time zone you are using in the format "+/- HH:MM". The time zone relates to UTC standard world time.
  The time in the "Current System Time" box is adapted accordingly.
- **Daylight Saving Time**
  Shows whether the daylight saving time changeover is active.
  - active (offset +1 h)
    The system time was changed to daylight saving time; in other words an hour was added. The normal time including the time zone continues to be displayed in the "Current System Time" box.
    The normal time including the time zone continues to be displayed in the "Current System Time" box.
  - inactive (offset +0 h)
    The current system time is not changed.
- **SNTP Mode**
  Select the synchronization mode. The following types of synchronization are possible:
  - Poll
    If you select this protocol type, the input boxes "SNTP Server Address", "SNTP Server Port" and "Poll Interval[s]" are displayed to allow further configuration. With this type of synchronization, the device is active and sends a time query to the SNTP server.
  - Listen
    With this type of synchronization, the device is passive and "listens" for SNTP frames that deliver the time of day.
- **Poll Interval[s]**
  Here, enter the interval between two-time queries. In this box, you enter the query interval in seconds. Possible values are 16 to 16284 seconds.

- **SNTP Server Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the SNTP server.
- **SNTP Server Port**
  Enter the port of the SNTP server.
  The following ports are possible:
  - 123 (standard port)
  - 1025 to 36564
- **Primary**
  The check mark is set for the SNTP server that you create first. If several SNTP servers have been created, the primary server is queried first.

NTP client

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

If you require time-of-day synchronization using NTP, you can make the relevant settings here.

Note

To avoid time jumps, make sure that there is only one time server in the network.

Settings

- **NTP Client**
  Enable or disable automatic time-of-day synchronization using NTP.
- **Current System Time** (only available online)
  Shows the current date and current normal time received by the IE switch. If you specify a time zone, the time information is adapted accordingly.
- **Last Synchronization Time**(only available online)
  Shows when the last time-of-day synchronization took place.

- **Last Synchronization Mechanism** (only available online)
  This box displays how the last time-of-day synchronization was performed. The following methods are possible:
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame
  - PTP
    Automatic time-of-day synchronization with PTP
- **Time Zone**
  Enter the time zone you are using in the format "+/- HH:MM". The time zone relates to UTC standard world time.
  The time in the "Current System Time" box is adapted accordingly.
- **Daylight Saving Time**
  Shows whether the daylight saving time changeover is active.
  - active (offset +1 h)
    The system time was changed to daylight saving time; in other words an hour was added. The normal time including the time zone continues to be displayed in the "Current System Time" box.
  - inactive (offset +0 h)
    The current system time is not changed.
- **NTP Server Index**
  Select the index of the NTP server. The server with the lowest index is queried first.
- **NTP Server Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the NTP server.
- **NTP Server Port**
  Enter the port of the NTP server.
  The following ports are possible:
  - 123 (standard port)
  - 1025 to 36564
- **Poll Interval[s]**
  Here, enter the interval between two time queries. In this box, you enter the poll interval in seconds. Possible values are 64 to 1024 seconds.

SIMATIC Time Client

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

On this page, you configure time synchronization with the SIMATIC Time Client.

Note

To avoid time jumps, make sure that there is only one time server in the network.

Settings

- **SIMATIC Time Client**
  Enable or disable the SIMATIC time client.
- **Current System Time** (only available online)
  Shows the current system time.
- **Last Synchronization Time**(only available online)
  Shows when the last time-of-day synchronization took place.
- **Last Synchronization Mechanism** (only available online)
  This box displays how the last time-of-day synchronization was performed.
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame
  - PTP
    Automatic time-of-day synchronization with PTP

PTP Client

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

If you require time-of-day synchronization using PTP, you can make the relevant settings here.

The following devices support time-of-day synchronization with PTP:

- SCALANCE XR528-6M
- SCALANCE XR552-12M

Settings

- **PTP Client**
  Enable or disable time synchronization using PTP. You can configure other settings in "Layer 2 > PTP".
- **Current System Time** (only available online)
  Shows the current date and current normal time obtained due to time synchronization in the network. If you specify a time zone, the time information is adapted accordingly.
- **Last Synchronization Time**(only available online)
  Shows when the last time-of-day synchronization took place. If no time-of-day synchronization was possible, the box displays "Date/time not set".
- **Last Synchronization Mechanism** (only available online)
  This box displays how the last time-of-day synchronization was performed.
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame
  - PTP
    Automatic time-of-day synchronization with PTP
- **Time Zone**
  Enter the time zone you are using in the format "+/- HH:MM". The time zone relates to UTC standard world time.
  The time in the "Current System Time" box is adapted accordingly.
- **Daylight Saving Time**
  Shows whether the daylight saving time changeover is active.
  - active (offset +1 h)
    The system time was changed to daylight saving time; in other words an hour was added. The normal time including the time zone continues to be displayed in the "Current System Time" box.
  - inactive (offset +0 h)
    The current system time is not changed.

Auto logout

On this page, set the times after which there is an automatic logout from WBM or the CLI following user in activity. If you have been logged out automatically, you will need to log in again.

Settings

- **Web Based Management [s]**
  Enter the time in seconds for the automatic logout from WBM. If you enter the value 0, the automatic logout is disabled.
- **CLI (TELNET, SSH, Serial) [s]**
  Enter the time in seconds for the automatic logout from the CLI. If you enter the value 0, the automatic logout is disabled.

Button

The "Select/Set" button has the following functions:

- Change display mode
- Enable redundancy manager
- Restore to factory defaults
- Fault mask and LED display

You will find a detailed description of the individual functions available with the buttons in the device operating instructions.

On this page, the functionality of the SELECT/SET button can be restricted or fully disabled.

Settings

- **Restore Factory Defaults**
  Enable or disable the "Restore Factory Defaults" function with the Select/Set button.

  CAUTION

  Button function "Restore Factory Defaults" active during startup

  If you have disabled this function in your configuration, disabling is only valid during operation. When restarting, for example after power down, the function is active until the configuration is loaded so that the device can inadvertently be reset to the factory settings. This may cause unwanted disruption in network operation since the device needs to be reconfigured if this occurs. An inserted PLUG is also deleted and returned to the status as shipped

- **Redundancy Manager**
  Enable or disable the redundancy manager function.
- **Set Fault Mask**
  Enable or disable the function "Define fault mask via the LED display" with the Select/set button. This function only works in display mode D.

Syslog client

Syslog according to RFC 3164 is used for transferring short, unencrypted text messages over UDP in the IP network. This requires a system log server.

Requirements for sending log entries:

- The Syslog function is enabled on the device.
- The Syslog function is enabled for the relevant event.
- There is a Syslog server in your network that receives the log entries. Since this is a UDP connection, there is no acknowledgment to the sender.
- The IP address or the FQDN (Fully Qualified Domain Name) of the Syslog server is entered in the device.

Settings

- **Syslog Client**
  Enable or disable the Syslog function.
- **Syslog Server Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the Syslog server.

The table contains the following columns:

- **Syslog Server Address**
  Shows the IP address or the FQDN (Fully Qualified Domain Name) of the Syslog server.
- **Server Port**
  Enter the port of the Syslog server being used.

Ports

Port overview

The page shows the configuration for the data transfer for all ports of the device.

You cannot configure anything on this page.

Settings

- **Port**
  Shows the configurable ports.
- **Port name**
  Shows the name of the port.
- **Port Type** (only with routing)
  Shows the type of the port. The following types are possible:
  - Router Port
  - Switch Port VLAN Hybrid
  - Switch-Port VLAN Trunk
  - Switch Port VLAN Host
  - Switch-Port PVLAN Promiscuous

- **Combo Port Media Type**
  The values are only available with combo ports. Shows the mode of the combo port:
  - auto
  - rj45
  - sfp
- **Status**
  Shows whether the port is on or off. Data traffic is possible only over an enabled port.
- **OperState**
  Displays the current operational status. The operational status depends on the configured "Status" and the "Link". The available options are as follows:
  - up
    You have configured the status "enabled" for the port and the port has a valid connection to the network.
  - down
    You have configured the status "disabled" or "Link down" for the port or the port has no connection.
  - not present
    With modular devices, this status is displayed when, for example, no media module is inserted.
- **Link** (only available online)
  Shows the connection status to the network. With the connection status, the following is possible:
  - Up
    The port has a valid link to the network, a link integrity signal is being received.
  - Down
    The link is down, for example because the connected device is turned off.
- **Mode** (only available online)
  Shows the transfer parameters of the port
- **MTU** (Maximum Transmission Unit)
  Shows the maximum packet size.
- **Negotiation**
  Shows whether the automatic configuration is enabled or disabled.
- **Flow Ctrl. Type**
  Shows whether flow control is enabled or disabled for the port.
- **Flow Ctrl.**
  Shows whether or not flow control is working on this port.

- **MAC Address** (only available online)
  Shows the MAC address of the port.
- **Blocked by**
  Shows why the port is in the "blocked" status:
  - -
    The analysis of the port status Is not supported.
  - forwarding
    The port is not blocked.
  - ring redundancy
    The port belongs to a redundancy manager. When the redundancy manager is in the "Passive status, one of the ring ports is in the "blocking" status.
  - spanning tree
    The port has the status "Discarding" in the spanning tree. The port is part of a spanning tree, however it is located on a redundant path and is deactivated for data traffic.
  - loop detection
    A loop was detected and as the reaction to a loop, the status "disable" was configured for the port.
  - down-in-bundle
    The port is part of a link aggregation and was deactivated by LACP.
  - la-loop-detection
    The port is part of a link aggregation. A loop was detected and as the reaction to a loop, the status "disable" was configured for the link aggregation.
  - la-spanning-tree
    The port is part of a link aggregation. The link aggregation was switched to the status "Discarding" by the spanning tree.
  - admin-down
    The status "disabled" is configured for the port, see "System > Ports > Configuration".
  - link down
    The status "enabled" is configured for the port but there is no connection, see "System > Ports > Configuration".
  - power-down
    The status “Link down" is configured for the port, see "System > Ports > Configuration".
  - standby
    Standby redundancy is enabled on the device. The port is a standby port with the status "Passive".

Differing display of the transmission parameters with combo ports. In the connection status "down", the displayed transmission parameters do not match the actual values of the combo port. In the connection status "up", the correct values are displayed.

Initial situation

A pluggable transceiver is plugged into the combo port with the following settings:

- Combo Port Media Type: auto
- Status: enabled
- Link: down display of the transmission parameters with 100 Mbps pluggable transceivers

- Actual response: Mode: 1G HD
- Expected response: Mode: 100M FD with 1 Gbps pluggable transceivers
- Actual response: Mode: 1G HD
- Expected response: Mode: 1G FD

Configuration

With this page, you can configure all the ports of the device.

Settings

- **Port**
  Select the port to be configured. The port is made up of the module number and the port number, for example port 0.1 is module 0, port 1.
- **Status**
  Specify whether the port is enabled or disabled.
  - enabled
    The port is enabled. Data traffic is possible only over an enabled port.
  - disabled
    The port is disabled but the connection remains.
  - Link down
    The port is disabled and the connection to the partner device is terminated.
- **Port name**
  Enter a name for the port.
- **MAC Address** (only available online)
  Shows the MAC address of the port.
- **Mode Type**(only available online)
  Select the transmission speed and the transmission method of the port. If you set the mode to "Autonegotiation", these parameters are automatically negotiated with the connected end device. This must also be in the "Autonegotiation" mode.

  Note

  Before the port and partner port can communicate with each other, the settings must match at both ends.

  Note

  "Mode Type" with combo ports

  To be able to set the "Mode Type" of a combo port, change the "Combo Port Media Type" to "rj45". If "auto" is set for the "Combo Port Media Type" and the RJ-45 port is used, you cannot set the "Mode Type".

- **Mode** (only available online)
  Shows the transmission speed and mode of the port.
  The transmission speed can be 10 Mbps, 100 Mbps, 1000 Mbps or 10 Gbps.
  As the transmission mode, full duplex (FD) or half duplex (HD) can be configured.

- **Negotiation**
  Shows whether the automatic configuration of the connection to the partner port is enabled or disabled.
- **Flow Ctrl. Type**
  Enable or disable flow control for the port.

  Note

  Turning flow control on/off with autonegotiation

  Flow control can only be enabled or disabled if the "autonegotiation" function is turned off. The function cannot enabled again afterwards.

- **Flow Ctrl.**
  Shows whether flow control is working on this port.
- **MTU**
  Enter the packet size.
- **Port Type** (only with routing)
  Select the type of the port:
  - Router Port
    The port is an IP interface. It does not support layer 2 functions.
  - Switch Port VLAN Hybrid
    The port sends tagged and untagged frames. It is not automatically a member of a VLAN.
  - Switch-Port VLAN Trunk
    The port only sends tagged frames and is automatically a member of all VLANs.
  - Switch Port VLAN Host
    Host ports belong to a secondary PVLAN. Connect devices to host ports that are only intended to communicate with certain devices of the PVLAN.
  - Switch-Port PVLAN Promiscuous
    Promiscuous ports belong to a primary PVLAN. Connect devices to promiscuous ports that are intended to communicate all devices of the PVLAN.
- **Combo Port Media Type**
  Specify the mode of the combo port:
  - auto
    If you select this mode, the SFP transceiver port has priority. As soon as an SFP transceiver is plugged in, an existing connection at the fixed RJ-45 port is terminated. If no SFC transceiver is plugged in, a connection can be established via the fixed RJ-45 port.
  - rj45
    If you select this mode, the fixed RJ-45 port is used regardless of the SFP transceiver port. If a pluggable transceiver is plugged in, it is disabled and the power turned off.
  - sfp
    If you select this mode, the SFP transceiver port is used regardless of the built-in RJ-45 port. If an RJ-45 connection is established, it is terminated because the power of the RJ-45 port is turned off.

  The factory setting for the combo ports is the auto mode.

- **OperState**
  Displays the current operational status. The operational status depends on the configured "Status" and the "Link". The available options are as follows:
  - up
    You have configured the status "enabled" for the port and the port has a valid connection to the network.
  - down
    You have configured the status "disabled" or "Link down" for the port or the port has no connection.
  - not present
    With modular devices, this status is displayed when, for example, no media module is inserted.

- **Link** (only available online)
  Shows the connection status to the network. The available options are as follows:
  - Up
    The port has a valid link to the network, a link integrity signal is being received.
  - Down
    The link is down, for example because the connected device is turned off.
- **Blocked by**
  Shows why the port is in the "blocked" status:
  - -
    The analysis of the port status Is not supported.
  - forwarding
    The port is not blocked.
  - ring-redundancy
    The port belongs to a redundancy manager. When the redundancy manager is in the "Passive status, one of the ring ports is in the "blocking" status.
  - spanning-tree
    The port has the status "Discarding" in the spanning tree. The port is part of a spanning tree, however it is located on a redundant path and is deactivated for data traffic.
  - loop-detection
    A loop was detected and as the reaction to a loop, the status "disable" was configured for the port.
  - down-in-bundle
    The port is part of a link aggregation and was deactivated by LACP.
  - la-loop-detection
    The port is part of a link aggregation. A loop was detected and as the reaction to a loop, the status "disable" was configured for the link aggregation.
  - la-spanning-tree
    The port is part of a link aggregation. The link aggregation was switched to the status "Discarding" by the spanning tree. - admin-down The status "disabled" is configured for the port, see "System > Ports > Configuration".
  - link-down
    The status "enabled" is configured for the port but there is no connection, see "System > Ports > Configuration".
  - power-down
    The status “Link down" is configured for the port, see "System > Ports > Configuration".
  - standby
    Standby redundancy is enabled on the device. The port is a standby port with the status "Passive".

Changing the port configuration

Note

Optical ports only work with the full duplex mode and at maximum transmission rate. As a result, the following settings cannot be made for optical ports:

- Automatic configuration
- Transmission speed
- Transmission technique

Note

With various automatic functions, the device prevents or reduces the effect on other ports and priority classes (Class of Service) if a port is overloaded. This can mean that frames are discarded even when flow control is enabled.

Port overload occurs when the device receives more frames than it can send, for example as the result of different transmission speeds.

Fault Monitoring

Power supply

Configure whether or not the power supply should be monitored by the messaging system. Depending on the hardware variant, there are one or two power connectors (Supply 1 / Supply 2). With a redundant power supply, configure the monitoring separately for each individual feed-in line.

If there is no power on one of the monitored lines (Supply 1 or Supply 2) or when the voltage is too low, a fault is signaled by the signaling system.

Note

You will find the permitted operating voltage limits in the compact operating instructions of the device.

A fault causes the signaling contact to trigger and the fault LED on the device to light up and, depending on the configuration, can trigger a trap, an e-mail, or an entry in the event log table.

Settings

- **Line 1**
  Enable or disable the monitoring of power connector 1.
- **Line 2**
  Enable or disable the monitoring of power connector 2.

Link Change

On this page, you configure whether or not an error message is triggered if there is a status change on a network connection.

If connection monitoring is enabled, an error is signaled

- when there should be a link on a port and this is missing.
- or when there should not be a link on a port and a link is detected.

An error causes the signaling contact to trigger and the fault LED to light up on the device. Depending on the configuration, the error may trigger a trap, an e-mail or an entry in the event log table.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns

- **Port**
  Shows the available ports.
- **Setting**
  Select the setting. You have the following options:
  - Up
    Error handling is triggered when the port changes to the active status.
    (From "Link down" to "Link up")
  - Down
    Error handling is triggered when the port changes to the inactive status.
    (From "Link up" to "Link down")
  - "-" (disabled)
    The error handling is not triggered.

Redundancy

On this page, you configure whether or not an error message is triggered if there is a status change on a network connection.

Setting

- **Redundancy loss (HRP only)**
  Enable or disable connection monitoring. If the redundancy of the connection is lost, an error is signaled.

PROFINET

This page shows the PROFINET IO AR status and the device name.

Settings

- **PNIO Device Diagnostics**
  Shows whether PNIO is enabled ("On") or disabled ("Off").
- **PNIO Device Diagnostics for next boot**
  Set whether PNIO will be enabled ("On") or disabled ("Off") after the next device restart.

  Note

  PNIO and EtherNet/IP

  When PNIO is turned on, EtherNet/IP is turned off. The switchover from PNIO and EtherNet/IP has no effect on DCP.

  Note

  PNIO AR Status

  If a PROFINET connection is established; in other words the PNIO AR status is "Online", you cannot disable PNIO.

- **PNIO AR Status**
  This box shows the status of the PROFINET IO connection; in other words whether the device is connected to a PROFINET IO controller "Online " or "Offline".
  Here, online means that a connection to a PROFINET IO controller exists, that this has downloaded its configuration data to the device and that the device can send status data to the PROFINET IO controller. In this status known as "in data exchange", the parameters set via the PROFINET IO controller cannot be configured.
- **PNIO Name of Station**
  This box displays the PROFINET IO device name according to the configuration in HW Config of STEP 7.
- **Restore PROFINET IO Defaults and Restart**
  Click this button to restore the default settings of the PROFINET IO profile and to restart the device. You must confirm the restart in a dialog box. The dialog box displays the settings specially made for operation with the PROFINET protocol.

NOTICE
By resetting the settings to the default settings of a profile, the IP address is also lost. Following this, the device can only be accessed via the serial interface, using the Primary Setup Tool or using DHCP. With the appropriate attachment, a previously correctly configured device can cause circulating frames and therefore the failure of the data traffic.

- **Allow PNIO Data Exchange**(only with SCALANCE XB-200)
  Enable or disable PNIO data exchange
- **Simulate PNIO Data Exchange**(only with SCALANCE XB-200)
  Enable or disable the simulation of PNIO data exchange.

EtherNet/IP

EtherNet/IP

On this page, you configure the mode of EtherNet/IP.

Description of the displayed boxes

The page contains the following boxes:

- **EtherNet/IP Device Diagnostics**
  Shows whether EtherNet/IP is enabled ("On") or disabled ("Off").
- **EtherNet/IP Device Diagnostics for next boot**
  Set whether EtherNet/IP will be enabled ("On") or disabled ("Off") after the next device restart.

  Note

  EtherNet/IP and PROFINET

  When EtherNet/IP is turned on, PROFINET is turned off. The switchover from EtherNet/IP and PROFINET has no effect on DCP.

  Note

  PROFINET AR Status

  If a PROFINET connection is established; in other words the PROFINET AR status is "Online", you cannot enable EtherNet/IP.

- **Restore EtherNet/IP Defaults and Restart**
  Click this button to restore the default settings of the EtherNet/IP profile and to restart the device. You must confirm the restart in a dialog box. The dialog box displays the settings specially made for operation with the EtherNet/IP protocol.

  NOTICE

  Loss of the IP address

  By resetting the settings to the default settings of a profile, the IP address is also lost. Following this, the device can only be accessed via the serial interface, using the Primary Setup Tool or using DHCP.

  With the appropriate attachment, a previously correctly configured device can cause circulating frames and therefore the failure of the data traffic.

PLUG

PLUG configuration

Note

The page is only available if there is an online connection to the device.

NOTICE
Do not remove or insert a C-PLUG / KEY-PLUG during operation! A PLUG may only be removed or inserted when the device is turned off. The device checks whether or not a PLUG is present at one second intervals. If it is detected that the PLUG was removed, there is a restart. If a valid KEY-PLUG was inserted in the device, the device changes to a defined error state following the restart.

Information about the configuration of the C-PLUG / KEY-PLUG

This page provides detailed information about the configuration stored on the C-PLUG or KEY-PLUG. It is also possible to reset the PLUG to "factory defaults" or to load it with new contents.

Note

The action is only executed after you click the "Set Values" button.

The action cannot be undone.

If you decide against executing the function after making your selection, click the "Refresh" button. As a result the data of this page is read from the device again and the selection is canceled.

Note

Incompatibility with previous versions with PLUG inserted

During the installation of a previous version, the configuration data can be lost. In this case, the device starts up with the factory settings after the firmware has been installed. In this situation, if a PLUG is inserted in the device, following the restart, this has the status "NOT ACCEPTED" since the PLUG still has the configuration data of the previous more up-to-date firmware. This allows you to return to the previous, more up-to-date firmware without any loss of configuration data.

If the original configuration on the PLUG is no longer required, the PLUG can be deleted or rewritten manually using "System > PLUG".

Settings

The table has the following rows:

- **Status**
  Shows the status of the PLUG. The following are possible:
  - ACCEPTED
    There is a PLUG with a valid and suitable configuration in the device.
  - NOT ACCEPTED
    Invalid or incompatible configuration on the inserted PLUG.
  - NOT PRESENT
    There is no C-PLUG or KEY-PLUG inserted in the device.
  - FACTORY
    PLUG is inserted and does not contain a configuration. This status is also displayed when the PLUG was formatted during operation.
  - MISSING
    There is no PLUG inserted. Functions are configured on the device for which a license is required.
- **Device Group**
  Shows the SIMATIC NET product line that used the C-PLUG or KEY-PLUG previously.
- **Device Type**
  Shows the device type within the product line that used the C-PLUG or KEY-PLUG previously.
- **Configuration Revision**
  The version of the configuration structure. This information relates to the configuration options supported by the device and has nothing to do with the concrete hardware configuration. This revision information does not therefore change if you add or remove additional components (modules or extenders), it can, however, change if you update the firmware.
- **File System**
  Displays the type of file system on the PLUG.

NOTICE
New file system UBI As of firmware version 3.0, UBI is the standard file system for the C-PLUG or KEY-PLUG. If a C-PLUG with the previous file system IECP is detected in such a device, this C-PLUG will be formatted for the UBI file system and the data will be rewritten to the C-PLUG. This change in the file system also occurs following a firmware update to V3.0. A downgrade to the previous version of the firmware then presents a problem. The firmware can neither read nor write the C-PLUG or KEY-PLUG and it is not even possible to "Erase PLUG to factory default".

- **File System Size [bytes]**
  Shows the maximum storage capacity of the file system on the C-PLUG.
- **File System Usage**
  Displays the storage space in use in the file system of the C-PLUG.

- **Info String**
  Shows additional information about the device that used the PLUG previously, for example, article number, type designation, and the versions of the hardware and software. The displayed software version corresponds to the version in which the configuration was last changed. With the "NOT ACCEPTED" status, further information on the cause of the problem is displayed.
- **Modify PLUG**
  Select the required setting. You have the following options for changing the configuration on the C-PLUG or KEY-PLUG:
  - Write current configuration to PLUG
    This option is available only if the status of the PLUG is "NOT ACCEPTED" or FACTORY. The configuration in the internal flash memory of the device is copied to the PLUG.
  - Erase PLUG to factory default
    Deletes all data from the C-PLUG and triggers low-level formatting.

PLUG license

Note

The page is only available if there is an online connection to the device.

NOTICE
Do not remove or insert a C-PLUG / KEY-PLUG during operation! A PLUG may only be removed or inserted when the device is turned off. The device checks whether or not a PLUG is present at one second intervals. If it is detected that the PLUG was removed, there is a restart. If a valid KEY-PLUG was inserted in the device, the device changes to a defined error state following the restart. If the device was configured at some time with a PLUG, the device can no longer be used without this PLUG. To be able to use the device again, reset the device to the factory settings.

Note

Incompatibility with previous versions with PLUG inserted

During the installation of a previous version, the configuration data can be lost. In this case, the device starts up with the factory settings after the firmware has been installed. In this situation, if a PLUG is inserted in the device, following the restart, this has the status "NOT ACCEPTED" since the PLUG still has the configuration data of the previous more up-to-date firmware. This allows you to return to the previous, more up-to-date firmware without any loss of configuration data.

If the original configuration on the PLUG is no longer required, the PLUG can be deleted or rewritten manually using "System > PLUG".

Information about the license of the KEY-PLUG

A C-PLUG can only store the configuration of a device. In addition to the configuration, a KEY-PLUG also contains a license that enables certain functions of your SIMATIC NET device.

This page provides detailed information about the license on the KEY-PLUG.

Displayed values

- **Status**
  Shows the status of the KEY-PLUG. The following are possible:
  - ACCEPTED
    The KEY-PLUG in the device contains a suitable and valid license.
  - NOT ACCEPTED
    The license of the inserted KEY-PLUG is not valid.
  - NOT PRESENT
    No KEY-PLUG is inserted in the device.
  - MISSING
    There is no KEY-PLUG or a C-PLUG with the status "FACTORY" inserted in the device. Functions are configured on the device for which a license is required.
  - WRONG
    The inserted KEY-PLUG is not suitable for the device.
  - UNKNOWN
    Unknown content of the KEY-PLUG.
  - DEFECTIVE
    The content of the KEY-PLUG contains errors.
- **Order ID**
  Shows the order number of the KEY-PLUG. The KEY-PLUG is available for various functional enhancements and for various target systems.
- **Serial Number**
  Shows the serial number of the KEY-PLUG.
- **Info String**
  Shows additional information about the device that used the KEY-PLUG previously, for example, order number, type designation, and the versions of the hardware and software. The displayed software version corresponds to the version in which the configuration was last changed. With the "NOT ACCEPTED" status, further information on the cause of the problem is displayed.

Note

When you save the configuration, the information about whether or not a KEY-PLUG was inserted in the device at the time is also saved. This configuration can then only work if a KEY-PLUG with the same article number / license is inserted.

Ping

Reachability of an address in an IP network

Note

The page is only available if there is an online connection to the device.

With the ping function, you can check whether a certain IPv4 address is reachable in the network.

Settings

- **IP Address**
  Enter the IPv4 address of the device.
- **Repeat**
  Enter the number of ping requests.
- **Ping**
  Click this button to start the ping function.
- **Ping Output**
  This box shows the output of the ping function.
- **Delete**
  Click this button to delete the ping output.

PoE

General

Note

The menu item “PoE” is only displayed for devices that support PoE

On this page, you see information about the power that the IE switch supplies with PoE.

The SCALANCE X-500 represents a PSE (Power Sourcing Equipment).

With the SCALANCE XM400, each group of four ports with PoE capability is known as a PSE.

The displayed values apply only to the corresponding PSE.

Setting

- **PSE**
  Shows the number of the PoE power supply.
- **Maximum Power**
  Maximum power that a PSE provides to supply PoE devices.

- **Allocated Power**
  Sum of the power reserved by the PoE devices according to the "Classification".
- **Used power**
  Sum of the power used by the end devices.
- **Usage Threshold [%]**
  When the power being used by the end devices exceeds the percentage shown here, an event is triggered.

Power over Ethernet with SCALANCE XM-400

With a SCALANCE XM-400, you can use the "Power over Ethernet" function via the port extender PE408PoE. PoE power supply The connection of the PoE power supply is external. You can connect 2 PoE power supplies with each PE408PoE port extender. Each PE408PoE therefore has 2 PSE units (Power Sourcing Equipment) each with 4 ports.

Port

Settings for the ports

For each individual PoE port, you can specify whether or not the power will be supplied via Ethernet. You can also set a priority for each connected powered device (PD). Devices for which a high priority was set, take preference over other devices for the power supply.

Settings

Table 1 has the following columns:

- **Port**
  Shows that the settings are valid for all ports.
- **Setting**
  Select the required setting. Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Priority**
  Select the required priority of the port. Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Type**
  Enter a string that describes the connected device in greater detail. The maximum length is 255 characters.
- **Copy to Table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the configurable PoE ports.
  The port is made up of the port number and the slot number, for example port 0.1 is slot 0, port 1.
- **Setting**
  Enable the PoE power supply for this port or interrupt it.
- **Priority**
  Select which priority this port will have for the power supply.
  The following settings are possible, in ascending order of relevance:
  - low
    low priority
  - high
    medium priority
  - critical
    high priority

  If the same priority is set for two ports, the port with the lower port number will be preferred when necessary.
- **Type**
  Here, you can enter a string to describe the connected device in greater detail.
- **Classification**
  The classification specifies the class of the device. From this, it is possible to recognize the maximum power of the device.
- **Status**
  Shows the current status of the port.
  The following states are possible:
  - disabled
    The PoE power supply is deactivated for this port.
  - delivering
    The PoE power supply is activated for this port and a device is connected.
  - searching
    The PoE power supply is activated for this port but there is no device connected.

  Note

  If a device is connected to a port with PoE capability, a check is made to determine whether the power of the port is adequate for the connected device. If the power of the port is inadequate, although PoE is enabled in "Setting", the port nevertheless has the status "disabled". This means that the port was disabled by the PoE power management.

- **Power [mW]**
  Shows the power that the SCALANCE provides at this port.

- **Voltage [V]**
  Shows the voltage applied to this port.
- **Current [mA]**
  Shows the current with which a device connected to this port is supplied.

Port diagnostics

Cable tester

Note

The page is only available if there is an online connection to the device.

With this page, each individual Ethernet port can run independent fault diagnostics on the cable. This test is performed without needing to remove the cable, connect a cable tester and install a loopback module at the other end. Short-circuits and cable breaks can be localized to within a few meters.

Note

Please note that this test is permitted only when no data connection is established on the port to be tested.

Settings

- **Port**
  Select the required port from the drop-down list.
- **Run Test**
  Activates error diagnostics. The result is shown in the table.

This table contains the following columns:

- **Pair**
  Shows the wire pair in the cable.

  Note
  Wire pairs

  Wire pairs 4-5 and 7-8 of 10/100 Mbps network cables are not used.

  In 1000 Mbps or gigabit Ethernet, all 4 wire pairs are used.

  The wire pair assignment - pin assignment is as follows (DIN 50173):

  Pair 1 = pin 4-5

  Pair 2 = pin 1-2

  Pair 3 = pin 3-6

  Pair 4 = pin 7-8

- **Status**
  Displays the status of the cable.
- **Distance [m]**
  Displays the distance to the cable end, cable break, or short-circuit.

SFP diagnostics

On this page, you run independent error diagnostics for each individual SFP port. This test is performed without needing to remove the cable, connect a cable tester or install a loopback module at the other end.

Note

Please note that this test is permitted only when no data connection is established on the port to be tested.

If, however, there is a data connection to the port to be tested, this is briefly interrupted.

Automatic re-establishment of the connection can fail and then needs to be done manually.

Description

The page contains the following boxes:

- **Port**
  Select the required port from the drop-down list.

The values are shown in the following boxes:

- **Name**
  Shows the name of the interface.
- **Model**
  Shows the type of interface.

- **Revision**
  Shows the hardware version of the SFP.
- **Serial**
  Shows the serial number of the SFP.
- **Nominal Bit Rate [Mbps]**
  Shows the nominal bit rate of the interface.
- **Max. Link (50.0/125um) [m]**
  Shows the maximum distance in meters that is possible with this medium.
- **Max. Link (62.5/125um) [m]**
  Shows the maximum distance in meters that is possible with this medium.

The following table shows the values of the SFP transceiver used in this port:

- **Temperature [°C]**
  Shows the temperature of the interface.
- **Voltage [V]**
  Shows the voltage applied to the interface [V].
- **Current [mA]**
  Shows the current consumption of the interface [mA].
- **RX Power [mW]**
  Shows the receive power of the interface [mW].
- **Tx Power [mW]**
  Shows the transmit power of the interface [mW].
- **Current**
  Shows the current value.
- **Low**
  Shows the lowest value.
- **High**
  Shows the highest value.

Configuring layer 2 functions

Configuration

On this page, you create a basic configuration for the functions of layer 2. On the configuration pages of these functions, you can make more detailed settings. You can also check the settings on the configuration pages.

Settings

- **Protocol Based VLAN**
  Enable or disable protocol-based VLAN. You can configure other settings in "Layer 2 > VLAN".
- **Subnet Based VLAN**
  Enable or disable subnet-based VLAN. You can configure other settings in "Layer 2 > VLAN".
- **Dynamic MAC Aging**
  Enable or disable the "Aging" mechanism. You can configure other settings in "Layer 2 > Dynamic MAC Aging".
- **Redundancy Type**
  The following settings are available:
  - **"-" (disabled)**
    The redundancy function is disabled.
  - **Spanning Tree**
    If you select this option, you specify the required redundancy mode in "Redundancy Mode".
  - **Ring**
    Enables ring redundancy. You can configure other settings in "Layer 2 > Ring Redundancy > Ring".
  - **Ring with RSTP**
    If you select this option, the compatibility mode for spanning tree is set permanently to RSTP. In the "Redundancy Type" drop-down list, you specify the redundancy mode of the ring redundancy. You can change the current setting in the "Ring Redundancy" and "Spanning Tree" menus.

    Note

    Restriction relating to ports with the "Ring with RSTP" option

    If you have enabled the "Ring with RSTP" option, the following ports must not be included in the spanning tree:
    - Ring ports
    - Standby ports
    - Standby coupling ports

- **Redundancy Mode**
  If you select "Spanning Tree" for the "Redundancy Type", the following options are then available:
  - **STP**
    Enables the Spanning Tree Protocol (STP). Typical reconfiguration times with spanning tree are between 20 and 30 seconds. You can configure other settings in "Layer 2 > Spanning Tree".
  - **RSTP**
    Enables the Rapid Spanning Tree Protocol (RSTP). If a spanning tree frame is detected at a port, this port reverts from RSTP to spanning tree. You can configure other settings in "Layer 2 > Spanning Tree".

    Note

    When using RSTP (Rapid Spanning Tree Protocol), loops involving duplication of frames or frames being overtaken may occur briefly. If this is not acceptable in your particular application, use the slower standard spanning tree mechanism.

  - **MSTP**
    Enables the Multiple Spanning Tree Protocol (MSTP). You can configure other settings in "Layer 2 > Spanning Tree".

  If you select "Ring" or "Ring with RSTP" for the "Redundancy Type", the following options are then available:

  - Automatic Redundancy Detection
    Select this setting to create an automatic configuration of the redundancy mode.
    In the Automatic Redundancy Detection mode, the device automatically detects whether or not there is a device with the "HRP Manager" role in the ring. If there is, the device adopts the role "HRP Client" client.
    If no HRP manager is found, all devices with the "Automatic Redundancy Detection" or "MRP Auto Manager" setting negotiate among themselves to establish which device adopts the role of "MRP Manager". The device with the lowest MAC address will always become "MRP Manager". The other devices automatically set themselves to "MRP Client" mode.
  - MRP Auto-Manager
    In the "MRP Auto Manager" mode, the devices negotiate among themselves to establish which device will adopt the role of "MRP Manager". The device with the lowest MAC address will always become "MRP Manager". The other devices automatically set themselves to "MRP Client" mode.
    In contrast to the setting "Automatic Redundancy Detection", the devices are not capable of detecting whether or not an HRP manager is in the ring.

    Note

    MRP configuration in STEP 7

    If you set the role "Manager (Auto)" or "Manager" for the device in STEP 7, in both cases, "MRP Auto Manager" is displayed on this WBM page. In the display in the CLI, a distinction is made between the two roles.

  - MRP Client
    The device adopts the role of MRP client.

  - HRP Client
    The device adopts the role of HRP client.
  - HRP Manager
    The device adopts the role of HRP manager.
    When you configure an HRP ring, one device must be set as HRP manager. For all other devices, "HRP Client" or "Automatic Redundancy Detection" must be set.

- **Standby**
  Enable or disable the "Standby redundancy" function. You can configure other settings in "Layer 2 > Ring Redundancy > Standby".
- **Passive Listening**
  Enable or disable the "passive listening" function.
- **RMON**
  If you select this check box, Remote Monitoring (RMON) allows diagnostics data to be collected on the device, prepared and read out using SNMP by a network management station that also supports RMON. This diagnostic data, for example port-related load trends, allow problems in the network to be detected early and eliminated.
- **Dynamic Multicast**
  The following settings are possible:
  - **"-" (disabled)**
  - **IGMP Snooping**
    Enables IGMP (Internet Group Management Protocol). You can configure other settings in "Layer 2 > Multicast > IGMP".
  - **MLD Snooping**
    Enables MLD. You can configure other settings in "Layer 2 > Multicast > MLD".
  - **IGMP and MLD Snooping**
    Enables IGMP and MDL.
  - **GMRP**
    Enables GMRP (GARP Multicast Registration Protocol). You can configure other settings in "Layer 2 > Multicast > GMRP".

    Note

    GMRP and IGMP cannot operate at the same time.

- **GVRP**
  Enable or disable "GVRP" (GARP VLAN Registration Protocol). You can configure other settings in "Layer 2 > VLAN > GVRP".
- **Mirroring**
  Enable or disable port mirroring. You can configure other settings in "Layer 2 > Mirroring > Port".

- **Loop Detection**
  Enable or disable loop detection. You can configure other settings in "Layer 2 > Loop Detection".
- **PTP**
  Specify how the device will process PTP messages. You can configure other settings in "Layer 2 > PTP".
  - off
    The device does not process any PTP messages. PTP messages are, however, forwarded according to the rules of the switch.
  - transparent
    The device adopts the function of a transparent clock and forwards PTP messages to other nodes while at the same time making entries in the correction field of the PTP message.

QoS

General

Transmission priorities

On this page, you can specify the priorities of different frames. In addition to this, depending on the priority you can set the method according to which the processing order of the frames is specified. You should also refer to the section "Basics (Page 1272)".

Description of the displayed values

The page contains the following boxes:

- **Broadcast Priority**
  Specify the priority of broadcast frames. The switch sorts the frame into a queue according to this prioritization . You configure the assignment of the priority to a queue on the page ""Layer 2 > QoS > CoS Map".
- **Agent Priority**
  Specify the priority of agent frames. The switch sorts the frame into a queue according to this prioritization . You configure the assignment of the priority to a queue on the page ""Layer 2 > QoS > CoS Map".
- **Scheduling Mode**
  Select the order in which the frames are processed in the queues.
  The higher the queue number, the higher the processing priority.
  - Strict Queueing
    As long as there are frames with high priority in the queue, only these high-priority frames are processed.
  - Weighted Fair Queueing
    Even if there are frames with high priority in the queue, frames with a lower priority will be processed occasionally.

CoS Map

On this page, you can assign CoS priorities to different Queues.

Settings

- **CoS**
  Shows the CoS priority of the incoming packets.
- **Queue**
  From the drop-down list, select the queue that is assigned to the CoS priority.
  The higher the number of the queue, the higher the processing priority.
  With queues 1 - 6 frames with a lower priority are occasionally processed even if there are frames with high priority in the queue.
  With queues 7 - 8 only frames with a high priority are processed as long as there are frames with high priority in the queue.

The service classes (CoS) are assigned to the queues as follows:

SCALANCE XB-200/XP-200	SCALANCE XM-400/XR-500
CoS 0 → Queue 2	CoS 0 → Queue 2
CoS 1 → Queue 1	CoS 1 → Queue 1
CoS 2 → Queue 1	CoS 2 → Queue 3
CoS 3 → Queue 2	CoS 3 → Queue 4
CoS 4 → Queue 3	CoS 4 → Queue 5
COo 5 → Queue 3	CoS 5 → Queue 6
CoS 6 → Queue 4	CoS 6 → Queue 7
CoS 7 → Queue 4	CoS 7 → Queue 8

DSCP Mapping

On this page, you can assign DSCP priorities to different Queues.

Settings

- **DSCP**
  Shows the DSCP priority of the incoming packets.
- **Queue**
  From the drop-down list, select the queue that is assigned to the DSCP priority. The higher the queue number, the higher the send priority.
  With queues 1 - 6 frames with a lower priority are occasionally processed even if there are frames with high priority in the queue.
  With queues 7 - 8 only frames with a high priority are processed as long as there are frames with high priority in the queue.

The DSCP codes are assigned to the queues as follows:

SCALANCE XB-200/XP-200	SCALANCE XM-400/XR-500
DSCP codes 0 - 15 → Queue 1	DSCP codes 0 - 7 → Queue 2
DSCP codes 16 - 31 → Queue 2	DSCP codes 8 - 15 → Queue 1
DSCP codes 32 - 47 → Queue 3	DSCP codes 16 - 23 → Queue 3
DSCP codes 48 - 63 → Queue 4	DSCP codes 24 - 31 → Queue 4
	DSCP codes 32 - 39 → Queue 5
	DSCP codes 40 - 47 → Queue 6
	DSCP codes 48 - 55 → Queue 7
	DSCP codes 56 - 63 → Queue 8

QoS Trust

Specifying the subnet priority

On this page you can set the method according to which frames to be forwarded are prioritized port by port.

Description of the displayed values

Table 1 has the following columns:

- 1st column
  Shows that the setting is valid for all ports of table 2.
- Trust Mode
  Select the required setting. If "No setting" is selected, the entry in table 2 remains unchanged.
- Transfer to table
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the configurable PoE ports.
  The port is made up of the module number and the port number, for example port 0.1 is module 0, port 1.
- **Trust Mode**
  Select the required mode from the drop-down list:
  - No Trust
    The switch sorts the incoming frames into a queue according to the prioritization of the receiving port.
    If there is a DSCP value in the IP header, this is ignored. If a VLAN tag exists, it is replaced by the priority value of the receiving port.
  - Trust COS
    If an incoming frame contains a VLAN tag, the switch sorts it into a queue according to this prioritization.
    If the frame does not contain a VLAN tag, the switch sorts the frame into a queue according to the prioritization of the receiving port.
    If there is a DSCP value in the IP header, this is ignored.
  - Trust DSCP
    If an incoming frame contains a DSCP prioritization, the switch sorts it into a queue according to this prioritization.
    If the frame does not contain a DSCP prioritization, the switch sorts the frame into a queue according to the prioritization of the receiving port.
    If the frame contains a VLAN tag, this is ignored.
  - Trust COS-DSCP
    With an incoming frame, there is a sequential check to determine which prioritization it contains.
    If the frame contains a DSCP prioritization, it is handled as in the "Trust DSCP" mode.
    If the frame contains no DSCP prioritization, the switch checks whether the frame contains a VLAN tag. If it contains a VLAN tag, the switch sorts it into a queue according to this prioritization.
    If the frame contains neither a DSCP prioritization nor a VLAN tag, the switch sorts the frame into a queue according to the prioritization of the receiving port.

CoS Port Remap

Changing priority when sending

On this page depending on the priority when receiving, you can change the priority of a frame with which it is sent.

Description of the displayed boxes

The page contains the following boxes:

- **CoS Remap**
  Enable or disable frames being sent with changed priorities according to Table 2.

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports of table 2.
- **Priority 0 - 7**
  The priority in the column stands for the priority with which a frame is received.
  - 0 - 7
    Select the priority with which a frame will be sent.
  - No Change
    No change in table 2.
- **Copy to Table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows all available ports. The port is made up of the module number and the port number, for example port 0.1 is module 0, port 1.
- **Priority 0 - 7**
  The priority in the column stands for the priority with which a frame is received.
  In the drop-down list select the priority with which a frame will be sent.

Rate Control

Limiting the transfer rate of incoming and outgoing data

On this page, you configure the load limitation (maximum number of data packets per second) for the individual ports. You can specify the category of frame for which these limit values will apply.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports.
- **Limit Ingress Unicast (DLF) / Limit Ingress Broadcast / Limit Ingress Multicast**
  Select the required setting. Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Total Ingress Rate [pkts/s]**
  Specify the maximum number of incoming packets processed by the device. If "No Change" is entered, the entry in table 2 remains unchanged
- **Egress Rate [Kb/s]**
  Specify the data rate for all outgoing frames. If "No Change" is entered, the entry in table 2 remains unchanged
- **Transfer to table**
  If you click the button, the settings are adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the port to which the settings relate.
- **Limit Ingress Unicast (DLF)**
  Enable or disable the data rate for limiting incoming unicast frames with an unresolvable address (Destination lookup failure).
- **Limit Ingress Broadcast**
  Enable or disable the data rate for limiting incoming broadcast frames.
- **Limit Ingress Multicast**
  Enable or disable the data rate for limiting incoming multicast frames.
- **Total Ingress Rate [pkts/s]**
  Specify the maximum number of incoming packets processed by the device.
- **Egress Rate [Kb/s]**
- Specify the data rate for all outgoing frames.

Note

Rounding of the values, deviation from desired value

When you enter the values for transmission rates, note that the WBM rounds to correct values.

If values are configured for total ingress transmission rate and egress transmission rate, the actual values in operation can exceed or fall below the set values by 10%.

VLAN

General

On this page, you define the VLAN and specify the use of the ports.

Note

Changing the "Agent VLAN ID"

If the configuration PC is connected directly to the device via Ethernet and you change the "agent VLAN ID", the device is no longer reachable via Ethernet following the change.

Important rules for VLANs

With SCALANCE XB-200 the rules of the Base Bridge mode apply "802.1Q VLAN Bridge".

Make sure you keep to the following rules when configuring and operating your VLANs:

- Frames with the VLAN ID "0" are handled as untagged frames but retain their priority value.
- As default, all ports on the device send frames without a VLAN tag to ensure that the end node can receive these frames.
- With SCALANCE X devices, the VLAN ID "1" is the default on all ports.

- If an end node is connected to a port, outgoing frames should be sent without a tag (static access port). If, however, there is a further switch at this port, the frame should have a tag added (trunk port).
- With a trunk port, the VLAN assignment is dynamic. Static configurations can only be created if, in addition to the trunk port property, the port is also entered statically as a member in the VLANs involved. An example of a static configuration is the assignment of the multicast groups in certain VLANs.

Settings

- **VLAN ID**
  Enter the VLAN ID.

The table has the following columns:

- **VLAN ID**
  Shows the VLAN ID. The VLAN ID is assigned once when you create a new data record and can then no longer be changed. To make a change, the entire data record must be deleted and created again.
- **Base Bridge mode (only with SCALANCE XB-200)**
  Specify the base bridge mode.

  Note

  Changing base bridge mode

  Note the section "Changing base bridge mode". This section describes how a change affects the existing configuration.

  - 802.1Q VLAN Bridge
    Sets the mode "VLAN-aware" for the device. In this mode, VLAN information is taken into account.
    Default setting with EtherNet/IP variants
  - 802.1D transparent bridge
    Sets the mode "VLAN-unaware" for the device. In this mode, VLAN tags are not taken into account or changed but are forwarded transparently. In this mode, you cannot create any VLANs. Only a management VLAN is available: VLAN 1:
    Default setting with PROFINET variants
- **Name**
  Enter a name for the VLAN. The name only provides information and has no effect on the configuration.
  The VLAN name can be a maximum of 32 characters long.
- **Status**
  Shows the status of the entry. Here, static means that the address was entered as a static address by the user. The entry GVRP means that the configuration was registered by a GVRP frame. This is, however, only possible if GVRP was enabled for the device.
- **Private VLAN Type**
  Shows the type of the PVLAN.
- **Primary VLAN ID**
  , With secondary PVLANs shows the ID of the corresponding primary PVLAN.

- Transparent(only with SCALANCE XM-400/XR-500)
  If you enable these settings, you switch a VLAN to the transparent mode. Ports that were assigned to this VLAN as members or untagged members now become transparent ports. This means the following:
  - The port VLAN ID of the transparent port is set to the ID of this VLAN.
  - Untagged frames that are received at these ports are forwarded to all other transparent ports once again without tag as long as they are not forwarded to a standard VLAN by a protocol or subnet rule.
  - Frames tagged with VLAN ID "0" and that are received at these ports are forwarded to all other transparent ports once again tagged with VLAN ID "0" as long as they are not forwarded to a standard VLAN by a protocol or subnet rule.
  - Frames tagged with the VLAN ID of the transparent VLAN and that are received at these ports are forwarded to all transparent ports once again tagged with the VLAN ID of the transparent VLAN.
  - Other frames are forwarded according to the normal VLAN rules and a transparent port behaves like an untagged member in this VLAN.
  - All ports that were not members or untagged members in the relevant VLAN are automatically set to the "Forbidden" status.
  - As long as a VLAN is configured as a transparent VLAN, the ports belonging to this VLAN cannot be modified.
  - You can only configure one a transparent VLAN.
- List of ports
  Specify the use of the port. The following options are available:
  - "-"
    The port is not a member of the specified VLAN.
    With a new definition, all ports have the identifier "-".
  - M
    The port is a member of the VLAN. Frames sent in this VLAN are forwarded with the corresponding VLAN tag.
  - R
    The port is a member of the VLAN. A GVRP frame is used for the registration.
  - U (upper case)
    The port is an untagged member of the VLAN. Frames sent in this VLAN are forwarded without the VLAN tag. Frames without a VLAN tag are sent from this port.
  - u (lower case)
    The port is an untagged member of the VLAN, but the VLAN is not configured as a port VLAN. Frames sent in this VLAN are forwarded without the VLAN tag.
  - F
    The port is not a member of the specified VLAN and it is not possible for the VLAN to be registered dynamically at this port using GVRP. You can configure other settings in "Layer 2 > VLAN > Port-based VLAN".

– T
This option is only displayed and cannot be selected in the WBM. This port is a trunk port making it a member in all VLANs. You configure this function in the CLI (Command Line Interface) using the "`switchport mode trunk`" command.

Changing Base Bridge mode with SCALANCE XB-200

VLAN-unaware (802.1D transparent bridge) → VLAN-aware (802.1Q VLAN bridge)

If you change the Base bridge mode from VLAN-unaware to VLAN aware, this has the following effects

- All static and dynamic unicast entries are deleted.
- All static and dynamic multicast entries are deleted.

VLAN-aware (802.1Q VLAN bridge) → VLAN-unaware (802.1D transparent bridge)

If you change the Base bridge mode from VLAN-aware to VLAN-unaware, this has the following effects

- All static and dynamic VLAN information is deleted.
- A management VLAN is created: VLAN 1:
- All static and dynamic unicast entries are deleted.
- All static and dynamic multicast entries are deleted.

GVRP

Configuration of GVRP functionality

Using GVRP frame, a different device can register at the port of the device for a specific VID. A different device, can, for example be an end device or a switch. The device can also send GVRP frames via this port. On this page, you can enable each port for GVRP functionality.

Settings

- **GVRP**
Enable or disable the "GVRP" function.

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports of table 2.
- **Setting**
  Select the setting. You have the following setting options:
  - Enabled
    Enables the sending of GVRP frames.
  - Disabled
    Disables the sending of GVRP frames.
  - No Change
    No change in table 2.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports.
- **Setting**
  Enable or disable the sending GVRP frames.

Port-based VLAN

Processing received frames

On this page, you specify the configuration of the port properties for receiving frames.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports.
- **Priority / Port VID / Acceptable Frames / Ingress Filtering**
  Select the setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports and link aggregations.
- **Priority**
  Select the priority assigned to untagged frames.
  The CoS priority (Class of Service) used in the VLAN tag. If a frame is received without a tag, it will be assigned this priority. This priority specifies how the frame is further processed compared with other frames.
  There are a total of eight priorities with values 0 to 7, where 7 represents the highest priority (IEEE 802.1p Port Priority).
- **Port VID**
  Select the VLAN ID. Only VLAN IDs defined on the "VLAN > General" page can be selected.
  If a received frame does not have a VLAN tag, it has a tag with the VLAN ID specified here added to it and is sent according to the rules at the port.
- **Acceptable Frames**
  Specify which types of frames will be accepted. The following alternatives are possible:
  - Tagged Frames Only
    The device discards all untagged frames. Otherwise, the forwarding rules apply according to the configuration.
  - All
    The device forwards all frames
- **Ingress Filtering**
  Specify whether the VID of received frames is evaluated.
  The following options are available:
  - Enabled
    The VLAN ID of received frames decides whether they are forwarded: To forward a VLAN tagged frame, the receiving port must be a member in the same VLAN. Frames from unknown VLANs are discarded at the receiving port.
  - Disabled
    All frames are forwarded.

Protocol-based VLAN group

On this page, you specify groups and assign a protocol to them.

Settings

- **Protocol Based VLAN**
  Enable or disable the protocol-based VLAN assignment.
- **Protocol Value**
  Enter the hexadecimal protocol value.
  A few examples are shown below:
  - PROFINET: 88:92
  - IP: 08:00
  - Novell: 81:37
  - netbios: f0:f0
  - appletalk: 80:9b
- **Group ID**
  Enter the ID of the group.

The table has the following columns:

- **Protocol Value**
  Shows the file value.
- **Group ID**
  Shows the group ID.

Protocol-based VLAN port

On this page, you specify which protocol and which VLAN is assigned to the individual port.

Settings

- **Port**
  Select the required port. All available ports and the link aggregations can be selected.
- **Group ID**
  Select the group ID from the drop-down list. Specify the ID in "Layer 2 > VLAN > Protocol Based VLAN Group".

The table has the following columns:

- **Port**
  All available ports and the link aggregations are shown.
- **Group ID**
  Shows the group ID assigned to the port.
- **VLAN ID**
  Select the required VLAN ID to be assigned to the port.

IPv4 subnet-based VLAN

On this page, you specify which VLAN ID is assigned to the subnet.

Settings

- **Subnet Based VLAN**
  Enable or disable the subnet-based VLAN assignment.
- **Port**
  Select the port. All available ports and the link aggregations can be selected.
- **Subnet Address**
  Enter the IP address of the subnet.
  Example: 192.168.10.0 for the network 192.168.10.x with nodes 192.168.10.1 to 192.168.10.254.
- **Subnet Mask**
  Enter the subnet mask.

The table has the following columns:

- **Port**
  All available ports and the link aggregations are shown.
- **Subnet Address**
  Shows the subnet assigned to the port.
- **Subnet Mask**
  Shows the subnet mask.
- **VLAN ID**
  Select the VLAN ID you want to assign to the port or the subnet.

Private VLAN

General

Private VLAN configuration page

On this page you define the types of the PVLANs and assign secondary PVLANs to a primary PVLAN.

Note

All secondary PVLANs must be known on all devices of a PVLAN. Even if a device has no host port in a secondary PVLAN, the secondary PVLAN must be known on the device.

Description of the displayed boxes

The table has the following columns:

- **VLAN ID**
  Shows the VLAN ID.
- **Private VLAN Type**
  Specify the type of PVLAN:
  - Primary
    Mit diesem Typ definieren Sie ein Primary PVLAN. In a PVLAN you can only define one primary PVLAN. The primary PVLAN uses the VLAN ID of the VLAN.
  - Isolated
    With this type, you define a secondary PVLAN. Only one device can exist in this secondary PVLAN. The secondary PVLAN has a specific VLAN ID.
  - Community
    With this type, you define a secondary PVLAN. The devices in this secondary PVLAN can communicate with each other via layer 2. The secondary PVLAN has a specific VLAN ID.
- **Primary VLAN ID**
  With secondary PVLANs select the VLAN ID of the primary PVLAN.

IP Interface Mapping

VLAN configuration page

On this page you specify from which secondary PVLANs the IP interface of the primary PVLAN will be reachable.

Description of the displayed boxes

The page contains the following boxes:

- **Interface**
  Select a primary PVLAN with an IP interface.
- **Secondary VLAN ID**
  Select a secondary VLAN ID from which the IP interface of the primary PVLAN will be reachable.

The table has the following columns:

- **Select**
  Select the row you want to delete.
- **Interface**
  Shows the IP interface.
- **Secondary VLAN-ID**
  Shows the secondary VLAN-ID of the secondary PVLAN from which the IP interface of the primary PVLAN is reachable.

Mirroring

Basics

Mirroring

The device provides the option of simultaneously channeling incoming or outgoing data streams via other interfaces for analysis or monitoring. This has no effect on the monitored data streams. This procedure is known as mirroring. In this menu section, you enable or disable mirroring and set the parameters.

Mirroring ports

Mirroring a port means that the data traffic at a port (mirrored port) of the IE switch is copied to another port (monitor port). You can mirror one or more ports to a monitor port.

If a protocol analyzer is connected to the monitor port, the data traffic at the mirrored port can be recorded without interrupting the connection. This means that the data traffic can be investigated without being affected. This is possible only if a free port is available on the device as the monitor port.

General

Mirroring in general

On this page, you can enable or disable the mirroring function and make the basic settings.

Note

Mirroring a port does not work beyond switch core boundaries. Refer to the operating instructions of the device.

Note

You need to disable port mirroring if you want to connect a normal end device to the monitor port. This does not apply to the function extender BUS ANALYZER Agent XM-400.

Note the data rate

If the maximum data rate of the mirrored port is higher than that of the monitor port, data may be lost and the monitor port no longer reflects the data traffic at the mirrored port. Several ports can be mirrored to one monitor port at the same time.

RSPAN

With RSPAN (Remote Switched Port Analyzer) you can forward the data traffic of a mirroring session to the monitor port via a VLAN. On the RSPAN VLAN, the mirrored data traffic is not disturbed by other data.

Frames addressed directly to the mirroring source cannot be mirrored on the RSPAN destination port.

Settings

- **Mirroring**
  Enable or disable mirroring of the data traffic.
- **Monitor Barrier**
  Enable or disable the setting to restrict communication via the monitor port.

Note

Effects of monitor barrier

If you enable this setting, the management of the switch via the monitor port is no longer reachable. The following port-specific functions are changed:

- DCP forwarding is turned off
- LLDP is turned off
- Unicast, multicast and broadcast blocking is turned on

The previous statuses of these functions are no longer restored after disabling monitor barrier again. They are reset to the default values and may need to be reconfigured.

You can reconfigure these functions manually even if monitor barrier is turned on. The data traffic on the monitor port is also allowed again. If you do not require this, make sure that only the data traffic you want to monitor is forwarded to the interface.

If mirroring is disabled, the listed port-specific functions are reset to the default values. This reset takes place regardless of whether the functions were configured manually or automatically by enabling "Monitor Barrier".

Note

Function extender BUS ANALYZER Agent XM-400

If the destination port is a port of the function extender BUS ANALYZER Agent XM-400 the monitor barrier option is always activated. It is activated regardless of whether the check box was disabled or enabled.

- **RSPAN VLAN ID**
  Select a VLAN.

This table has the following columns:

- **Session ID**
  The Session ID is assigned automatically when a new entry is created.
- **Session Type**
  Select the required entry:
  - '-'
    None
  - Port-based
    Port-based mirroring
  - VLAN
    VLAN-based mirroring.
  - MAC ACL
    Mirroring of the MAC Access Control List.
  - IP ACL
    Mirroring of the IP Access Control List.

  Note

  If you have created a session of the type "VLAN" "MAC ACL" or "IP ACL", you cannot create any further sessions. You can create up to 7 sessions of the type "Port-based".

  If you change the "Session Type" of an existing session, all previous configurations of this session are lost.

- **Status**
  Shows whether or not mirroring is active.
- **Hardware Index**
  If in a VLAN you select more than one source port for the port-based egress mirroring, unknown unicast and multicast frames as well as broadcast frames are forwarded only once to the destination port. With several sessions, the corresponding frames are only visible in one session. They are only mirrored on the one destination port with the lowest hardware index.
- **Dest. Port**
  Select the destination port to be mirrored to in this session.

  Note

  Function extender BUS ANALYZER Agent XM-400

  If you connect a function extender BUS ANALYZER Agent XM-400 with a SCALANCE XM-400 basic device as of firmware version 5.1, you can select up to four ports of the function extender BUS ANALYZER Agent XM-400 as destination ports. Function extender BUS ANALYZER Agent XM-400 supports port-based, VLAN-based, MAC ACL-based and IP ACL-based mirroring.

- **RSPAN**
  Enable or disable RSPAN for a session.

Port

Mirroring ports

You can only configure the settings on this page if you have already generated a session ID with the session type "Port-based" on the "General" tab.

Settings

- **Session ID**
  Select the session you want to monitor. Up to 7 parallel sessions are possible and their ports must not overlap.

The table has the following columns:

- **Port**
  Shows the port to be monitored.
- **Ingress Mirroring**
  Enable or disable listening in on incoming packets at the required port.
- **Egress Mirroring**
  Enable or disable listening in on outgoing packets at the required port.

VLAN

VLAN sources of the port mirroring

You can only configure the settings on this page if you have already generated a session ID with the session type "VLAN" on the "General" tab.

On this page, you specify the VLAN whose incoming data traffic will be mirrored to the monitor port.

It can happen that data packets are visible on the monitor port that were not received in the defined VLAN. These data packets come from functions that are enabled on the device, e.g. SIMATIC time client. To avoid these data packets when VLAN mirroring, disable the relevant functions on the device before a recording.

Settings

- **Session ID**
  Select the session you want to monitor. Only one session is possible.
- **VLAN ID**
  Enter the VLAN ID in the "VLAN ID" input box.

The table "Ingress Mirroring " has the following columns:

- **VLAN-ID**
  Shows the VLAN ID for which the incoming frames are mirrored. The VLAN ID can only be assigned once when you create a new data record and can then no longer be changed. To make a change, the entire data record must be deleted and created again.

MAC Flow

You can only configure the settings on this page if you have already generated a session ID with the session type "MAC ACL" on the "General" tab.

The MAC ACL filter decides which data is available at the monitor port.

Settings

- **Session ID**
  Select the session number of the port mirroring.

The table has the following columns:

- **ACL Filter Number**
  Shows the number of the ACL filter. You configure the MAC ACL filter in "Security > MAC ACL".
- **Ingress Mirroring**
  Shows whether incoming packets are mirrored.

  Note

  Rules

  The selected rule only becomes active when you specify with which ACL rules the incoming packets will be filtered for at least one interface. You configure the settings in "Security > MAC ACL > Ingress Rules".

- **Source MAC**
  Shows the MAC address of the sender.
- **Dest. MAC**
  Shows the MAC address of the recipient.
- **Ingress Interfaces**
  Shows all interfaces to which this rule applies. The ACL filter decides which incoming data streams are mirrored on the monitor port (destination port).
- **Egress Interfaces**
  Shows all interfaces to which this rule applies.

IP Flow

You can only configure the settings on this page if you have already generated a session ID with the session type "IP ACL" on the "General" tab.

The ACL filter decides which data is available at the monitor port.

Settings

- **Session ID**
  Select the session ID. Only session IDs of the session type "IP ACL" are available.

The table has the following columns:

- **ACL Filter Number**
  Shows the number of the ACL filter.
- **Ingress Mirroring**
  Shows whether incoming packets are mirrored.

  Note

  Rules

  The selected rule only becomes active when you specify with which ACL rules the incoming packets will be filtered for at least one interface. You configure the settings in "Security > IP ACL > Ingress Rules".

- **Source IP**
  Shows the IPv4 address of the sender.
- **Source Subnet Mask**
  Shows the subnet mask of the sender.
- **Dest. IP**
  Shows the IPv4 address of the recipient.
- **Dest. Subnet Mask**
  Shows the subnet mask of the recipient.
- **Ingress Interfaces**
  Shows all interfaces to which this rule applies. The ACL filter decides which incoming data streams are mirrored on the monitor port (destination port).
- **Egress Interfaces**
  Shows all interfaces to which this rule applies. The ACL filter decides which outgoing data streams are mirrored on the monitor port (destination port).

Dynamic MAC Aging

The device automatically learns the source addresses of the connected nodes.

This information is used to forward data frames to the nodes specifically involved. The network load for the other nodes is reduced.
If a device does not receive a frame whose source address matches a learnt address within a certain time, the learnt address is deleted. This mechanism is known as aging. Aging prevents frames being forwarded incorrectly, for example when an end device (for example a programming device) is connected to a different port. If the check box is not enabled, a device does not delete learned addresses automatically.

Settings

- **Dynamic MAC Aging**
  Enable or disable the function for automatic aging of learned MAC addresses:
- **Aging Time [s]**
  Enter the time in seconds. After this time, a learned address is deleted if the device does not receive any further frames from this sender address.

Ring Redundancy

Ring

Rules for ring redundancy

Factory settings

- The factory setting defines MSTP as the redundancy method.
- With SCALANCE XM-400, the factory setting defines ports P1.1 and P1.2 as ring ports.
- With SCALANCE XM-500, the factory setting defines ports P0.1 and P0.2 as ring ports.
  Ring ports

Ring ports

The predefined ring ports of SCALANCE XR-500 are 10 Gbps ports. If a 1000 Mbps pluggable transceiver is plugged into one of the 10 Gbps ring ports, you cannot enable ring redundancy. Remove the 1000 Mbps pluggable transceiver.

Enabling ring redundancy

You can enable ring redundancy as follows:

- using the WBM
- using the CLI
- using the SELECT/SET button
- using a PNIO configuration download

Settings

- **Ring Redundancy**
  If you enable the "Ring Redundancy" check box, you turn ring redundancy on. The ring ports set on this page are used.
- **Ring redundancy mode**
  Specify the ring redundancy mode. The following options are available:
  - **"-" (disabled)**
    The redundancy function is disabled.
  - **Automatic Redundancy Detection**
    Select this setting to create an automatic configuration of the redundancy mode.
    In the "Automatic Redundancy Detection" mode, the device automatically detects whether or not there is a device with the "HRP Manager" role in the ring. If this is the case, the device adopts the role of "HRP Client". If no HRP manager is found, all devices with the "Automatic Redundancy Detection" or "MRP Auto Manager" setting negotiate among themselves to establish which device adopts the role of "MRP Manager". The device with the lowest MAC address will always become "MRP Manager". The other devices automatically set themselves to "MRP Client" mode.
  - **MRP Auto Manager**
    Automatic media redundancy manager
  - **MRP Client**
    Media redundancy client
  - **HRP Client**
    High Speed Redundancy Protocol client
  - **HRP Manager**
    High Speed Redundancy Protocol manager
- **Ring ports**
  Specify the ports to be used as ring ports in media redundancy in ring topologies.
  The ring port you select in the left-hand drop-down menu is the "Isolated Port" in HRP.

Note

If you restore the factory defaults, the default redundancy mode "Automatic Redundancy Detection" becomes active.

The ring port configuration is also reset to the factory default ports. If other ports were used previously as ring ports, with the appropriate attachment, a previously correctly configured device can cause circulating frames and therefore the failure of the data traffic.

- **Observer**
  Enable or disable the observer. The "Observer" function is only available in HRP rings. The ring port selected in the left-hand drop-down menu is connected to the "Isolated Port" of an HRP manager.
  The observer monitors malfunctions of the redundancy manager or incorrect configurations of an HRP ring.
  If the observer is enabled, it can interrupt the connected ring if errors are detected. To do this, the observer switches a ring port to the "blocking" status. When the error is resolved, the observer enables the port again.
- **Restart Observer** (only available online)
  If numerous errors occur in quick succession, the observer no longer enables its port automatically. The ring port remains permanently in the "blocking" status. This is signaled by the error LED and a message text.
  After the errors have been eliminated, you can enable the port again using the "Restart Observer" button.

Restoring factory settings

If you have restored the factory defaults (Restore Factory Defaults and Restart), ring redundancy is disabled and the default ports are used as the ring ports. This can lead to circulating frames and failure of the data traffic if other settings were used in a previous configuration.

Changing over the status of the ring ports with the redundancy manager

If you configure a redundancy manager, set the status of the ring ports. The first ring port changes to the “blocking" status and the second ring port to the “forwarding" status. As long as ring redundancy is enabled, you cannot change the status of these ring ports.

Note

Make sure that you first open the ring so that there are no circulating frames.

Standby

Redundant linking of rings

Standby redundancy allows the redundant linking of HRP rings. To establish a standby connection, configure two neighboring devices within a ring as standby master or standby slave. The standby master and the standby slave must be connected via parallel cables to two devices in another ring. In problem-free operation, messages are exchanged between the two rings via the master. If the master's line is disturbed, the slave takes over the forwarding of messages between the two rings. Enable standby redundancy for both standby partners and select the ports via which the device is connected to the rings you want to link to. For the "Standby Connection Name", a name unique within the ring must be assigned for both partners. This identifies the two devices as standby partners that belong together.

Note

To be able to use the function, HRP must be activated.

The standby manager always requires an activated HRP client or HRP manager.

Note

Standby master coupling with an optical 100 Mbps connection

While the coupling partner of a standby master with an optical 100 Mbps connection starts up again, physically disconnect the connection between the standby master and its coupling partner.

Settings

- **Standby**
  Enable or disable this function.
- **Standby Port**
  Select the port to be standby port. The link to the other ring is via the standby port.
  Note

  Note

  Standby ports in the spanning tree

  Before you enable the port as the standby port, you need to disable the port in the spanning tree.

  1. Change to the page "Layer 2 > Spanning Tree > CIST Port".

  2. Disable the ports in the spanning tree you want to configure as standby ports.

  The standby port is involved in the redirection of data traffic. In there are no problems, only the standby port of the master is enabled and handles the data traffic into the connected HRP ring or HRP bus.
  If the master or the Ethernet connection (link) of one of the standby ports of the master fails, the standby port of the master will be disabled and the standby port of the slave enabled. As a result, a functioning Ethernet connection to the connected network segment (HRP ring or HRP linear bus) is restored.

- **Standby Connection Name**
  This name defines the master/slave device pair. Both devices must be located in the same ring.
  Here, enter the name for the standby connection. This must be identical to the name entered on the standby partner. You can select any name to suit your purposes, however, you can only use the name for one pair of devices in the entire network.
- **Force device to Standby Master**
  When selected, the device is configured as standby master regardless of its MAC address.
  - If this check box is not selected for either of the devices for which the standby master is enabled, then assuming that no error has occurred, the device with the higher MAC address adopts the role of standby master.
  - If the option is selected for both devices or if the "Force device to Standby Master" property is supported by only one device, the standby master is also selected based on the MAC address.

  This type of assignment is important in particular when a device is replaced. Depending on the MAC addresses, the previous device with the slave function can take over the role of the standby master.

  Note

  If two devices are linked by the standby function, the "Standby" function must be enabled on both devices.

- **Wait for Standby Partner**
  - Enabled
    A standby connection is enabled only after the standby master and the standby slave as well as their standby partners have established a connection. This ensures that the redundant connection is really available before communication via a standby connection is enabled.
  - Disabled
    A standby connection is enabled even if the standby master has not yet established a connection to the standby slave.
    This can lead to circulating frames and failure of the data traffic if another standby connection has already been enabled. Multiple standby connections can, for example, result due to configuration errors if different standby connection names were assigned to the standby master and standby slave.

Spanning Tree

General

General settings of MSTP

On this page, you configure the settings for MSTP. As default, Rapid Spanning Tree is enabled that can be set to the MSTP, RSTP or STP compatible mode with a switch.

On the configuration pages of these functions, you can make detailed settings.

Depending on the compatibility mode, you can configure the corresponding function on the relevant configuration page.

Settings

- **Spanning Tree**
  Enable or disable Spanning Tree.
- **Protocol Compatibility**
  Select the compatibility mode of MSTP. For example if you select RSTP, MSTP behaves like RSTP.
  The following settings are available:
  - STP
  - RSTP
  - MSTP

See also

Link aggregation (Page 1424)

MST port (Page 1419)

MST General (Page 1419)

CIST general/ST general

The page consists of the following parts.

- The left-hand side of the page shows the configuration of the device.
- The central part shows the configuration of the root bridge that can be derived from the spanning tree frames received by a device.
- The right-hand side shows the configuration of the regional root bridge that can be derived from the MSTP frames. The displayed data is only visible if you have enabled "Spanning Tree" on the "General" page and if "MSTP" is set for "Protocol Compatibility". This also applies to the "Bridge Max Hop Count" parameter. If the device is a root bridge, the information on the left and right matches.

Settings

- **Bridge Priority / Root Priority** (only available online)
  Which device becomes the root bridge is decided based on the bridge priority. The bridge with the highest priority becomes the root bridge. The lower the value, the higher the priority. If several devices in a network have the same priority, the device whose MAC address has the lowest numeric value will become the root bridge. Both parameters, bridge priority and MAC address together form the bridge identifier. Since the root bridge manages all path changes, it should be located as centrally as possible due to the delay of the frames.
- **Bridge Address / Root Address**(only available online)
  The bridge address shows the MAC address of the device and the root address shows the MAC address of the root bridge.

- **Root Port** (only available online)
  Shows the port via which the switch communicates with the root bridge.
- **Root Cost** (only available online)
  The path costs from this device to the root bridge.
- **Topology Changes** / **Last Topology Change** (only available online)
  The entry for the device shows the number of reconfigurations due to the spanning tree mechanism since the last startup. For the root bridge, the time since the last reconfiguration is displayed as follows:
  - Seconds: sec unit after the number
  - Minutes: min unit after the number
  - Hour: hr unit after the number
- **Bridge Hello Time [s] / Root Hello Time[s]** (only available online)
  Each bridge regularly sends configuration frames (BPDUs). The interval between two such frames is the Hello time. The default for this parameter is 2 seconds.
- **Bridge Forward Delay [s] / Root Forward Delay [s]**
  New configuration data is not used immediately by a bridge but only after the period specified in the parameter. This ensures that operation is only started with the new topology after all the bridges have the required information.
- **Bridge Max Age / Root Max Age** (only available online)
  When the max age timer elapses the received BPDU is discarded to be accepted as valid by the switch.
- **Bridge Max Hop Count**
  This parameter specifies how many MSTP nodes a BPDU may pass through. If an MSTP BPDU is received and has a "Bridge Max Hop Count" that exceeds the value configured here, it is discarded.
- **Regional root priority** (only available online)
  For a description, see Bridge Priority / Root Priority
- **Regional root address** (only available online)
  The MAC address of the device.
- **Regional root costs** (only available online)
  Shows the path costs from this device to the regional root bridge.
- **Region Name**
  Enter the name of the MSTP region to which this device belongs. As default, the MAC address of the device is entered here. This value must be the same on all devices that belong to the same MSTP region.
- **Region Version**
  Enter the version number of the MSTP region in which the device is located. This value must be the same on all devices that belong to the same MSTP region.
- **Reset Counter** (only available online)
  Click this button to reset the counters on this page to zero.

CIST Port/ST Port

MSTP-CIST port configuration

When the page is called, the table displays the current status of the configuration of the port parameters.

To configure them, click the relevant cells in the port table.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the setting is valid for all ports of table 2.
- **Spanning Tree Status**
  Select the setting. If "No Change" is selected, the entry in table 2 remains unchanged.

Table 2 has the following columns:

- **Port**
  Shows the available ports and interfaces.
  - Port X
  - WLAN X
  - VAP X.Y
  - WDS X.Y
- **Spanning Tree Status**
  Specify whether or not the port is integrated in the spanning tree.

  Note

  If you disable the "Spanning Tree Status" setting for a port, this may cause the formation of loops. The topology must be kept in mind.

- **Priority**
  Enter the priority of the port. The priority is only evaluated when the path costs are the same. The value must be divisible by 16. If the value that cannot be divided by 16, the value is automatically adapted.
- **Cost Calc.**
  Enter the path cost calculation. If you enter the value "0" here, the automatically calculated value is displayed in the "Path Costs" box.

- **Path Costs** (only available online)
  This parameter is used to calculate the path that will be selected. The path with the lowest value is selected as the path. If several ports of a device have the same value for the path costs, the port with the lowest port number is selected.
  If The "Cost Calc." is "0", the automatically calculated value is shown. Otherwise, the value of the "Cost Calc." box is displayed. The calculation of the path costs is largely based on the transmission speed. The higher the achievable transmission speed is, the lower the value of the path costs.
  Typical values for path costs with rapid spanning tree:
  - 10,000 Mbps = 2,000
  - 1000 Mbps = 20,000
  - 100 Mbps = 200,000
  - 10 Mbps = 2,000,000

  The values can, however, also be set individually.
- **Status** (only available online)
  Displays the current status of the port. The values are only displayed and cannot be configured. The "Status" parameter depends on the configured protocol. The following is possible for status:
  - Disabled

    The port only receives and is not involved in STP, MSTP and RSTP.
  - Discarding
    In the "Discarding" mode, BPDU frames are received. Other incoming or outgoing frames are discarded.
  - Listening

    In this status, BPDUs are both received and sent. The port is involved in the spanning tree algorithm.
  - Learning

    Stage prior to the forwarding status, the port is actively learning the topology (in other words, the node addresses).
  - Forwarding

    Following the reconfiguration time, the port is active in the network; it receives and forwards data frames.

- **Fwd. Trans** (only available online)
  Specifies the number of changes from the "Discarding" status to the "Forwarding" status
- **Edge type**
  Specify the type of edge port. You have the following options:
  - "-"
    Edge port is disabled. The port is treated as a "no edge port".
  - Admin
    Select this option when there is always an end device on this port. Otherwise a reconfiguration of the network will be triggered each time a connection is changed.
  - Auto
    Select this option if you want a connected end device to be detected automatically at this port. When the connection is established the first time, the port is treated as a "no Edge Port".
  - Admin/Auto
    Select these options if you operate a combination of both on this port. When the connection is established the first time, the port is treated as an Edge Port.
- **Edge** (only available online)
  Shows the status of the port.
  - Enabled

    An edge port is connected to this port.
  - Disabled
    There is a spanning tree or rapid spanning tree device at this port.

  With an end device, a switch can change over the port faster without taking into account spanning tree frames. If a spanning tree frame is received despite this setting, the port automatically changes to the "Disabled" setting for switches.
- **P.t.P. Type**
  Select the required option. The selection depends on the port that is set.
  - "-"
    Point to point is calculated automatically. If the port is set to half duplex, a point-to-point link is not assumed.
  - P.t.P.

    Even with half duplex, a point-to-point link is assumed.
  - Shared Media

    Even with a full duplex connection, a point-to-point link is not assumed.

  Note

  Point-to-point connection means a direct connection between two devices. A shared media connection is, for example, a connection to a hub.

- **P.t.P.**
  - Enabled
    Shows that a point-to-point link exists.
  - Disabled
    Shows that no point-to-point link exists.
- **Hello Time**
  Enter the interval after which the bridge sends configuration BPDUs

Note

The port-specific setting of the Hello time is only possible in MSTP compatible mode.

MST General

Multiple Spanning Tree configuration

With MSTP, in addition to RSTP, several VLANs can be managed in a LAN with separate RSTP trees.

Settings

- **MSTP Instance ID**
  Enter the number of the MSTP instance.

The table has the following columns:

- **Select**
  Select the line to be deleted.
- **MSTP Instance ID**
  Shows the number of the MSTP instance.
- **Root Address**
  Shows the MAC address of the root bridge
- **Root Priority**
  Shows the priority of the root bridge.
- **Bridge Priority**
  Enter the bridge priority. The value for the bridge priority is a whole multiple of 4096.
- **VLAN ID**
  Enter the VLAN ID. Here, you can also specify ranges with Start ID, "-", End ID. Several ranges or IDs are separated by ",".

MST port

Configuration of the Multiple Spanning Tree port parameters

On this page, you set the parameters for the ports of the configured multiple spanning tree instances.

Settings

- **MSTP Instance ID**
  Select the ID of the MSTP instance.

Table 1 has the following columns

- **1st column**
  Shows that the setting is valid for all ports.
- **MSTP Status**
  Select the setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2

Table 2 has the following columns:

- **Port**
  Shows all available ports and link aggregations.
- **MSTP Instance ID**
  ID of the MSTP instance.
- **MSTP Status**
  Enable or disable MSTP for this port.
- **Priority**
  Enter the priority of the port. The priority is only evaluated when the path costs are the same. The value must be divisible by 16. If the value that cannot be divided by 16, the value is automatically adapted.
- **Cost Calc.**
  Enter the path cost calculation in the input box. If you enter the value "0" here, the automatically calculated value is displayed in the next box "Path Costs".
- **Path Costs** (only available online)
  This parameter is used to calculate the path that will be selected. The path with the lowest value is selected as the path. If several ports of a device have the same value for the path costs, the port with the lowest port number is selected.
  If the value for the Cost Calc is "0" the automatically determined value is displayed. Otherwise, the value of the "Cost Calc box is displayed. The calculation of the path costs is largely based on the transmission speed. The higher the achievable transmission speed is, the lower the value of the path costs.
  Typical values for path costs with rapid spanning tree:
  - 10,000 Mbps = 2,000
  - 1000 Mbps = 20,000
  - 100 Mbps = 200,000
  - 10 Mbps = 2,000,000

  The values can, however, also be set individually.

- **Status** (only available online)
  Displays the current status of the port. The values are only displayed and cannot be configured. The following is possible for status:
  - Discarding
    The port exchanges MSTP information but is not involved in the data traffic.
  - Blocked
    In the blocking mode, BPDU frames are received.
  - Forwarding
    The port receives and sends data frames.
- **Fwd. Trans**(only available online)
  Specifies the number of status changes Discarding - Forwarding or Forwarding - Discarding for a port.

Enhanced Passive Listening Compatibility

Enabling the function

On this page, you can enable passive listening compatibility.

Settings

- **Enhanced Passive Listening Compatibility**
  Enable or disable this function for the entire device.

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports of table 2.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Displays the port of the device.
- **Setting**
  - Enabled
    Enables the function.
  - Disabled
    Disables the function.

Loop detection

With the "Loop Detection" function, you specify the ports for which loop detection will be activated. The ports involved send special test frames - the loop detection frames. If these frames are sent back to the device, there is a loop.

A "local loop" involving this device means that the frames are received again at a different port of the same device. If the sent frames are received again at the same port, there is a "remote loop" involving other network components.

Note

A loop is an error in the network structure that needs to be eliminated. The loop detection can help to find the errors more quickly but does not eliminate them. The loop detection is not suitable for increasing network availability by deliberately including loops.

Note

Note that loop detection is only possible at ports that were not configured as ring ports or standby ports.

Settings

- **Loop Detection**
  Enable or disable loop detection.
- **VLAN Loop Detection**
  Enable or disable loop detection in a VLAN.

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports of table 2
- **Threshold / Remote Reaction / Local Reaction**
  Select the required settings. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports.
- **Setting**
  Specify how the port handles loop detection frames.

  Note

  Test frames create additional network load. We recommend that you only configure individual switches, for example at branch points of the ring, as "sender" and the others as "forwarder".

  – sender
    Loop detection frames are sent out and forwarded.
  – forwarder
    Loop detection frames from other devices are forwarded.
  – blocked
    The forwarding of loop detection frames is blocked.
- **Threshold**
  By entering a number, specify the number of received loop detection frames as of which a loop is assumed.
- **Remote Reaction**
  Specify how the port will react if a remote loop occurs.
  – no action
    A loop has no effect on the port.
  – disable
    The port is blocked.
- **Local Reaction**
  Specify how the port will react if a local loop occurs.
  – no action
    A loop has no effect on the port.
  – disable
    The port is blocked
- **Status** (only available online)
  This box shows whether loop detection is enabled or disabled for this port.
- **Source Port** (only available online)
  Shows the receiving port of the loop detection frame that triggered the last reaction
- **Source VLAN** (only available online)
  Shows the VLAN ID of the loop detection frame that triggered the last reaction.
  This requires that the "VLAN Loop Detection" setting is selected.
- **Reset** (only available online)
  After a loop in the network has been eliminated, click this button to reset the port again.

Note

Changing the configured port status with loop detection

The configuration of the port status can be changed with the "Loop Detection" function. If, for example, the administrator has disabled a port, the port can be enabled again after a device restart with "Loop Detection" (enabled). The port status "link down" is not changed by "Loop Detection".

Note

Effects of configuration using STEP 7

The configuration of Spanning Tree can be changed if you configure "Loop Detection" using STEP 7.

If you enable "Loop Detection" on a port with STEP 7, Spanning Tree is automatically disabled on this port. If you disable "Loop Detection" for the port again with STEP 7 Spanning Tree is not automatically enabled. Enable Spanning Tree with the WBM or CLI.

Link aggregation

Bundling network connections for redundancy and higher bandwidth

A link aggregation according to IEEE 802.3ad allows several connections between neighboring devices to be bundled to achieve higher bandwidths and to protect against failure.

Ports on both partner devices are included in link aggregations and the devices are then connected via these ports. To assign ports (in other words links) correctly to a partner device, the Link Aggregation Control Protocol (LACP) from the IEEE 802.3ad standard is used.

Note

When a port is assigned to a link aggregation but is not active (e.g. link down), the values displayed may differ from the values configured for the link aggregation. If the port in the link aggregation becomes active, individual port configurations such as DCP forwarding are overwritten with the configured values of the link aggregation.

Settings

The table has the following columns:

- **Port**
  Shows the virtual port number of this link aggregation. This identifier is assigned internally by the firmware.
- **Link Aggregation Name**
  Enter a name for the link aggregation. This name can be specified by the user during configuration. The name is not absolutely necessary but can be useful to distinguish between the various link aggregations.

- **MAC Address**
  Shows the MAC address.
- **Status**
  Enable or disable link aggregation.
- **MTU**
  Specify the packet size.
- **LACP**
  - on
    Enables the sending of LACP frames.
  - off
    Disables the sending of LACP frames.
- **Frame Distribution**
  Set the type of distribution of packets on the individual links of an aggregation.
  - Destination&Source Mac
    The distribution is based on a combination of the destination and source MAC address.
  - Destination&Source IP-MAC
    The distribution is based on a combination of the destination and source IP address and MAC address.

- **VLAN Mode**
  Specify how the link aggregation is entered in a VLAN:
  - Hybrid
    The link aggregation sends tagged and untagged frames. It is not automatically a member of a VLAN.
  - Trunk
    The link aggregation only sends tagged frames and is automatically a member of all VLANs.
- **Port**
  Shows the ports that belong to this link aggregation. The following values can be selected from the drop-down list:
  - "-" (disabled)
    Link aggregation is disabled.
  - "a" (active)
    The port sends LACP frames and is only involved in the link aggregation when LACP frames are received.
  - "p" (passive)
    The port is only involved in the link aggregation when LACP frames are received.
  - o" (on)
    The port is involved in the link aggregation and does not send any LACP frames.

Note

Within a "link aggregation", only ports with the following configuration are possible:

- all ports with "o"
- all ports with "a" or "p".

Note

If you add a configured port to a link aggregation, the port adopts the configuration of the link aggregation. If you take the port out of the link aggregation, the settings of the port are reset to the factory settings.

See also

General (Page 1413)

DCP forwarding

Applications

The DCP protocol is used by STEP 7 and the PST Tool for configuration and diagnostics. When shipped, DCP is enabled on all ports; in other words, DCP frames are forwarded at all ports. With this option, you can disable the sending of these frames for individual ports, for example to prevent individual parts of the network from being configured with the PST Tool or to divide the full network into smaller parts for configuration and diagnostics.

Note

PNIO configuration

Since DCP is a PROFINET protocol, the configuration created here is only effective with the VLAN associated with the TIA interface.

All the ports of the device are displayed on this page. After each displayed port, there is a drop-down list for function selection.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the setting is valid for all ports of table 2.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports.
- **Setting**
  Specify whether the port should block or forward outgoing DCP frames. The following options are available:
  - forward
    DCP frames are forwarded at this port.
  - block
    No outgoing DCP frames are forwarded at this port. It is nevertheless still possible to receive via this port.

LLDP

Identifying the network topology

LLDP (Link Layer Discovery Protocol) is defined in the IEEE 802.AB standard.

LLDP is a method used to discover the network topology. Network components exchange information with their neighbor devices using LLDP.

Network components that support LLDP have an LLDP agent. The LLDP agent sends information about itself and receives information from connected devices at periodic intervals. The received information is stored in the MIB.

Applications

PROFINET uses the LLDP protocol for topology diagnostics. In the default setting, LLDP is enabled for all ports; in other words, LLDP frames are sent and received on all ports. With this function, you have the option of enabling or disabling sending and/or receiving per port.

Settings

Table 1 has the following columns:

- 1st column
  Shows that the setting is valid for all ports of table 2.
- Setting
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- Transfer to table
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- Port
  Shows the port.
- Setting
  Select whether or not the port will send or receive LLDP frames. The following options are available:
  - Rx
    This port can only receive LLDP frames.
  - Tx
    This port can only send LLDP frames.
  - Rx & Tx
    This port can receive and send LLDP frames.
  - "-" (disabled)
    This port can neither receive nor send LLDP frames.

FMP

Requirements

- To be able to use the Fiber Monitoring (FM) function, enable LLDP. The FM information is appended to the LLDP packets.
- You can only use Fiber Monitoring with transceivers capable of diagnostics. Devices and modules with transceivers capable of diagnostics have the supplement "FM" in the name.

Monitoring optical links

With Fiber Monitoring, you can monitor optical links.

If you enable FM on an optical port, diagnostics information is sent via the port. In addition to sending, there is also a check to establish whether information is being received.

Regardless of whether the IE switch receives diagnostics information, it monitors the received power and compares it with selectable power limits.

If you connect two IE switches via optical ports with FM enabled, they exchange their diagnostics information. This also allows a power loss to be calculated. The power loss is also compared with selectable power limits.

If the value of the received power or the power loss falls below a specified limit, an event is triggered. In "System > Events > Configuration", you can specify how the IE switch indicates the event.

Settings

- **Port**
  Shows the available optical ports that support fiber monitoring. This depends on the transceivers.
- **State**
  Enable or disable FM. As default, the function is enabled.
- **Rx Power [dBm] maintenance required (warning)**
  Specify the value at which you are informed of the deterioration of the received power by a message of the severity level “Warning"
  The default value depends on the relevant pluggable transceiver.
- **Rx Power [dBm] maintenance demanded (critical)**
  Specify the value at which you are informed of the deterioration of the received power by a message of the severity level “Critical"
  The default value depends on the relevant pluggable transceiver.
- **Power Loss [dB] maintenance required (warning)**
  Enter the value at which you want to be informed about the power loss of the connection the first time.
- **Power Loss [dB] maintenance demanded (critical)**
  Enter the value at which you want to be informed about the power loss of the connection the second time.

Unicast

Filter

Address filtering

This page shows the current content of the unicast filter table. This table lists the source addresses of unicast address frames. The displayed entries were made statically, in other words, the user set them.

On this page, you also define the static unicast filters.

Settings

- **VLAN ID**
  Select the VLAN ID in which you configure a new static MAC address. If nothing is set, "VLAN1" is set as the basic setting.
- **MAC Address**
  Enter the MAC address here.

This table contains the following columns:

- **VLAN ID**
  Shows the VLAN-ID assigned to this MAC address.
- **MAC Address**
  Shows the MAC address of the node that the device has learned or the user has configured.
- **Status**
  Shows the status of each address entry:
  - static
    Configured by the user. Static addresses are stored permanently; in other words, they are not deleted when the aging time expires or when the switch is restarted.
- **Port**
  Shows the port via which the node with the specified address can be reached. Frames received by the device whose destination address matches this address will be forwarded to this port.

Note

You can only specify **one** port for unicast addresses.

Learning

Starting/stopping learning

Note

The page is only available if there is an online connection to the device.

With the automatic learning function, all connected devices are automatically entered in the unicast filter table. As long as the "Start learning" function is enabled, all learned unicast addresses are created immediately as static unicast entries.
The learning process is ended only after clicking the "Stop learning" button. With this method, learning can take a few minutes or several hours in larger networks before all nodes have really been learned. Only nodes that send packets during the learning phase are found.
By subsequently enabling the Port Lock function, only packets from the nodes known after the end of the learning phase (static unicast entries) will be accepted at the relevant ports.

Note

If the Port Lock function was already active on individual ports prior to the automatic learning phase, no addresses will be learned on these ports. This makes it possible to restrict learning to certain ports. To do this, first enable the Port Lock function of the ports that are not intended to learn addresses.

Settings

- **Start learning**
  Click the "Start learning" button to start the learning phase.
  The device now enters the addresses of connected devices until you stop the function.
- **Stop learning**
  Click the "Stop Learning" button to stop the learning phase.
  The learned entries are stored.
- **Deleting all static unicast addresses**
  Click the "Clear all static unicast addresses" button to delete all static entries.
  In large networks with numerous nodes, automatic learning may lead to a lot of undesired static entries. To avoid having to delete these individually, this button can be used to delete all static entries. This function is disabled during automatic learning.

  Note

  Depending on the number of entries involved, deleting may take some time.

Locked ports

Activating the access control

On this page, you can block individual ports for unknown nodes.

If the Port Lock function is enabled, packets arriving at this port from unknown MAC addresses are discarded immediately. Packets from known nodes are accepted by the port.
Since ports with the port lock function enabled cannot learn any MAC addresses, learned addresses on these ports are automatically deleted after the port lock function is enabled.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the setting is valid for all ports of table 2.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports.
- **Setting**
  Enable or disable access the port lock function for the port.

Blocking

Blocking forwarding of unknown unicast frames

On this page, you can block the forwarding of unknown unicast frames for individual ports.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the setting is valid for all ports of table 2.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports.
- **Setting**
  Enable or disable the blocking of unicast frames

Multicast

Groups

Multicast applications

In the majority of cases, a frame is sent with a unicast address to a particular recipient. If an application sends the same data to several recipients, the amount of data can be reduced by sending the data using one multicast address. For some applications, there are fixed multicast addresses (NTP, IETF1 Audio, IETF1 Video etc.).

Reducing network load

In contrast to unicast frames, multicast frames represent a higher load for the device. Generally, multicast frames are sent to all ports. There are three ways of reducing the load caused by multicast frames:

- Static entry of the addresses in the multicast filter table.
- Dynamic entry of the addresses by listening in on IGMP parameter assignment frames (IGMP Configuration).
- Active dynamic assignment of addresses by GMRP frames.

The result of all these methods is that multicast frames are sent only to ports for which an appropriate address is entered.

“Information > Multicast" shows the multicast frames currently entered in the filter table and their destination ports. The displayed entries were made statically, in other words, the user set them.

Settings

- VLAN ID
  Select the VLAN ID to be assigned to the MAC multicast address.
- MAC Address
  Here you enter a new MAC multicast address you want to configure.

The table has the following columns:

- VLAN ID
  Shows VLAN ID of the VLAN to which the MAC multicast address is assigned.
- MAC Address
  Shows the MAC multicast address that the device has learned or the user has configured.

- **Status**
  Shows the status of each address entry.
  - Static
    The address was entered statically by the user. Static addresses are stored permanently; in other words, they are not deleted when the aging time expires or when the device is restarted. These must be deleted by the user.
- **List of ports**
  There is a column for each slot. Within a column, the multicast group to which the port belongs is shown. The following selections are possible:
  - M (Member)
    Multicast frames are sent via this port.
  - F (Forbidden)
    Not a member of the multicast group. Moreover, this address must not be an address learned dynamically with GMRP or IGMP/MLD.
  - -
    Not a member of the multicast group. No multicast frames with the defined multicast MAC address are sent via this port

IGMP

Function

IE switches support "IGMP snooping" and the IGMP querier function. If "IGMP snooping" is enabled, IGMP frames are evaluated and the multicast filter table is updated with this information. If "IGMP Querier is also enabled, IE switches also send IGMP queries that trigger responses from IGMP-compliant nodes.

IGMP Snooping Aging Time

In this menu, you can configure the aging time for IGMP Configuration. When the time elapses, entries created by IGMP are deleted from the address table if they are not updated by a new IGMP frame.

This applies to all ports; a port-specific configuration is not possible.

IGMP Snooping Aging Time depending on the querier

SCALANCE XR500 as IGMP querier:

If a SCALANCE XR500 is used as an IGMP querier, the query interval is 125 seconds. For the "IGMP Snooping Aging Time", set at least 250 seconds.

Other IGMP queriers

If a different IGMP querier is used, the value of the "IGMP Snooping Aging Time" should be at least twice as long as the query interval.

Settings

- **IGMP Snooping**
  Enable or disable IGMP (Internet Group Management Protocol). The function allows the assignment of IP addresses to multicast groups. If the check box is selected, IGMP entries are included in the table and IGMP frames are forwarded.
- **IGMP Snooping Aging Time**
  Enter the value for the aging time in seconds in this box.
- **IGMP Querier**
  Enable or disable "IGMP Querier". The device sends IGMP queries.

GMRP

Enabling GMRP

By selecting the check box, you specify whether or not GMRP is used for each individual port. If "GMRP" is disabled for a port, no registrations are made for it and it cannot send GMRP frames.

Settings

- **GMRP**
  Enable or disable the GMRP function.

Table 1 has the following columns:

- **1st column**
  Shows that the setting is valid for all ports of table 2.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the settings are adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports and the link aggregations.
- **Setting**
- Enable or disable GMRP for the port or for the link aggregation.

Blocking

On this page, you can block the forwarding of unknown multicast frames for individual ports.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the setting is valid for all ports of table 2.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports.
- **Setting**
  Enable or disable the blocking of multicast frames.

MLD

Mulitcast Listener Discovery

Devices support "MLD snooping" and the "MLD querier" function. If "MLD snooping" is enabled, MLD frames are evaluated and the multicast filter table is updated with this information. If "IGMP querier is also enabled, IE switches also send MLD queries that trigger responses from MLD-compliant nodes.

With MLD snooping, MLD frames are only forwarded to the intended multicast listeners instead of being flooded to all ports.

For MLD snooping to work, you need to enable the function globally and in the VLANs.

MLD Snooping Aging Time

On this page, you can configure the aging time for MLD configuration. When the time elapses, entries created by MLD are deleted from the address table if they are not updated by a new MLD packet.

This applies to all VLANS; a VLAN-specific configuration is not possible.

Settings

- **MLD Snooping**
  Enable or disable MLD on the device.
- **MLD Snooping Aging Time**
  Enter the time after which entries generated by MLD are deleted from the multicast filter table. Assuming that this is not updated by a new MLD packet.
  This applies to all VLANS; a VLAN-specific configuration is not possible

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all VLANs of table 2.
- **Snooping / Querier**
  Select the setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Copy to Table**
  If you click the button, the settings are adopted for all ports of table 2.

Table 2 has the following columns:

- **VLAN**
  Shows the VLAN to which the settings relate
- **Snooping**
  Enable or disable MLD for the required VLAN. When enabled, the MLD entries are included the multicast filter table and MLD packets are forwarded.
- **Querier**
  Enable or disable sending of MLD queries

Broadcast

Blocking the forwarding of broadcast frames

On this page, you can block the forwarding of broadcast frames for individual ports.

Note

Some communication protocols work only with the support of broadcast. In these cases, blocking can lead to loss of data communication. Block broadcast only when you are sure that you do not need it.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the setting is valid for all ports of table 2.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  All available ports and the link aggregations are shown.
- **Setting**
  Enable or disable the blocking of broadcast frames.

PTP

General

Note

The following devices support time-of-day synchronization with PTP:

- SCALANCE XR528-6M
- SCALANCE XR552-12M

IEEE 1588 with SCALANCE devices

The IEEE 1588v2 standard defines mechanisms with which highly precise time-of-day synchronization of devices in a network can be achieved. SCALANCE devices with suitable hardware support time synchronization according to IEEE 1588v2.

The functionality is disabled on these devices when they are shipped and following a "Reset to factory defaults". To be able to use PTP, enable this function and configure every port that is on the synchronization path as well as ports that are blocked due to redundancy mechanisms. PTP can also be used with redundancy mechanisms in the ring such as HRP, standby linking of rings, MRP and RSTP.

Setting

- **1588 Mode**
  You can make the following settings:
  - **off**
    The device does not process any PTP messages. PTP messages are, however, forwarded according to the rules of the device.
  - **transparent**
    The device adopts the function of a transparent clock and forwards PTP messages to other nodes while at the same time making entries in the correction field of the PTP message.

TC general

1588 Transparent Clock

On this page, you specify the general settings for PTP.

Settings

- **Delay Mechanism**
  Specify the delay mechanism the device will work with:
  - End-to-end(delay request response mechanism will be used)

    Note

    With end-to-end synchronization with more than 2 slaves, freak values > 100 ns can occur in the offset.

  - Peer-to-peer (peer delay mechanism will be used)
- **Domain Number**
  Enter the domain number for the device. The device ignores PTP messages with a different domain number. A SCALANCE device can only be assigned to one synchronization domain.

TC port

On this page, you specify the ports that can process PTP messages.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports of table 2.
- **Setting / Transport Mechanism**
  Select the setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the settings are adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports.
- **Setting**
  The port status. The following entries are possible:
  - Enabled
    The port is not involved in PTP.
  - Disabled
    The port processes PTP messages.

- **Faulty Flag**
  Shows error status relating to PTP.
  - true
    An error occurred.
  - false
    No error has occurred on this port.
- **Transport Mechanism**
  Specifies the protocol for transferring the PTP messages. This protocol must also be supported by the communications partner of the port
  - Ethernet
  - UDP IPv4

RMON

Statistics

Statistics

On this page you can specify the ports for which RMON statistics are displayed. The RMON statistics are shown on the page “Information > Ethernet Statistics” in “Packet Size”, “Frame Type” and “Packet Errror”.

Settings

- **RMON**
  When enabled, Remote Monitoring (RMON) allows diagnostics data to be collected on the device, prepared and read out using SNMP by a network management station that also supports RMON. This diagnostic data, for example port-related load trends, allow problems in the network to be detected early and eliminated.

  Note

  If you disable RMON, these statistics are not deleted but retain their last status.

- **Port**
  Select the ports for which statistics will be displayed.

The table has the following columns:

- **Port**
  Shows the ports for which statistics will be displayed.

History

Samples of the statistics

On this page, you can specify whether or not samples of the statistics are saved for a port. You can specify how many entries should be saved and at which intervals samples should be taken.

Settings

Table 1 has the following columns:

- 1st column
  Shows that the settings are valid for all ports.
- Setting/Buckets/Interval
  Select the required settings. If "No Change" is selected, the entry in table 2 remains unchanged.
- Transfer to table
  If you click the button, the settings are adopted for all ports of table 2.

Table 2 has the following columns:

- Port
  Shows the port to which the settings relate.
- Setting
  Enable or disable the recording of the history on the relevant port.
- Entries
  Enter the maximum number of samples to be stored at the same time.
- Interval [s]
  Enter the interval after which the current status of the statistics will be saved as a sample.

Configuring layer 3 functions for IPv4

Configuration

The page contains the overview of the layer 3 functions for IPv4 of the device. On this page, you enable or disable the required layer 3 function.

The functions "Routing", "VRRP", "RIP" and "OSPF" are only available with layer 3.

Settings

- **Routing**(only available with devices with a layer 3 license)
  - Enabled
    IPv4 routing is enabled. You can only enable the routing function if DHCP is disabled on all configured interfaces.
  - Disabled
    IPv4 routing is disabled. If IPv6 routing is enabled on the device, this is also disabled.
- **DHCP Relay Agent**
  Enable or disable the DHCP relay agent. You can configure other settings in "Layer 3 (IPv4)> DHCP Relay Agent".
- **VRRP** (only available with devices with a layer 3 license)
  Enable or disable routing using VRRP. To use VRRP, first enable the routing function. You can configure other settings in "Layer 3 (IPv4) > VRRP".
- **OSPF** (only available with devices with a layer 3 license)
  Enable or disable routing using VRRP. To use VRRP, first enable the routing function. You can configure other settings in "Layer 3 (IPv4) > VRRP".
- **RIP** (only available with devices with a layer 3 license)
  Enable or disable routing using RIP. To use RIP , first enable the routing function. You can configure other settings in "Layer 3 (IPv4) > RIP".

Subnets

Overview (IPv4)

Subnet

The page shows the subnets for the selected interface. If more than one subnet is available on an interface, the first entry of this interface is of the address type "Primary". All other subnets have the address type "Secondary".

A subnet always relates to an interface. You configure the interface in "Layer 3 (IPv4) > Configuration".

Settings

- **Interface**
  Select the interface on which you want to configure another subnet.

The table has the following columns:

- **Interface**
  Shows the interface.
- **TIA Interface**
  Shows whether or not the entry is a TIA interface.
- **Interface Name**
  Shows the name of the interface.

- **MAC Address**
  Shows the MAC address.
- **IP Address**
  Shows the IPv4 address of the subnet.
- **Subnet Mask**
  Shows the subnet mask.
- **Address Type**
  Shows the address type. The following values are possible:
  - Primary
    The first IPv4 address that was configured on an IP interface.
  - Secondary
    All other IPv4 addresses that were configured on an interface.
- **IP Assignment Method**
  Shows how the IPv4 address is assigned.
  - Static
    The IP address is static. You enter the IP settings in "IP Address" and "Subnet Mask".
  - Automatic (DHCP)
    The device obtains a dynamic IPv4 address from a DHCP server.
- **Address Collision Detection Status**
  If new IPv4 addresses become active in the network, this function checks whether this can result in address collisions. With this function, IPv4 addresses that will be assigned twice can be detected.
  Shows the status of the "Detection of address collisions" function.

  Note

  The function does not run a cyclic check.

  - Idle
    The interface is not enabled and does not have an IP address.
  - Starting
    This status indicates the start-up phase. In this phase, the device initially sends a query as to whether the planned IP address already exists. If the address is not yet been assigned, the device sends the message that it is using this IP address as of now.
  - Conflict
    The interface is not enabled. The interface is attempting to use an IP address that is already been assigned.
  - Defending
    The interface uses a unique IP address. Another interface is attempting to use the same IP address.
  - Active
    The interface uses a unique IP address. There are no collisions.
  - Not supported
    The function for detection of address collisions is not supported.

Configuration

On this page, you configure the subnet. You create the subnet in "Layer 3 > Subnets > Overview".

Settings

- **Interface (Name)**
  Select the required interface.
- **Interface Name**
  Enter a name for the interface.
- **MAC Address** (only available online)
  Shows the MAC address of the selected interface
- **DHCP**
  Enable or disable the DHCP client for this IPv4 interface.

  Note

  If you want to operate the device as a router with several interfaces, disable DHCP on all interfaces.

- **IP Address**
  Enter the IPv4 address of the subnet. The IPv4 address must not be used more than once.
- **Subnet Mask**
  Enter the subnet mask of the subnet you are creating. Subnets on different interfaces must not overlap.
- **Address Type**
  Shows the address type. The following values are possible:
  - Primary
    The first subnet of the interface.
  - Secondary
    All other subnets of the interface.
- **TIA Interface**
  Enable or disable the setting.
  The following conditions apply to the TIA interface:
  - Only interfaces with the address type "Primary" can be enabled as the TIA interface.
  - There must only ever be one TIA interface.
  - There can only be one TIA interface.
  - A TIA interface is always a VLAN interface.

TIA interface and PROFINET interface

The IP address of the TIA interface is linked to the IP address located in the parameter group "Properties > General > PROFINET interface > Ethernet addresses".

Both IP addresses always have the same value. If you change the value of one IP address, the other IP address changes accordingly.

TIA Interface

The TIA interface is the interface via which all PROFINET functions of the IE switch are handled.

TIA interface is the name of the IP address for PROFINET from the perspective of the IE switch.

PROFINET interface

You make the IP setting of the PROFINET device in TIA under PROFINET interface.

PROFINET interface is the name of the IP address for PROFINET from the perspective of the TIA Portal.

Static Routes (IPv4)

Static route

On this page, you create the static IPv4 routes.

Settings

- **Destination Network**
  Enter the network address of the destination that can be reached via this route.
- **Subnet Mask**
  Enter the corresponding subnet mask.
- **Gateway**
  Enter the IPv4 address of the gateway via which this network address is reachable.
- **Metric**
  Enter the metric for the route. The metric corresponds to the quality of a connection, for example speed, costs. If there are several equal routes, the route with the lowest metric value is used.

The table has the following columns:

- **Destination Network**
  Shows the network address of the destination.
- **Subnet Mask**
  Shows the corresponding subnet mask.
- **Gateway**
  Shows the IPv4 address of the next gateway.
- **Interface** (only available online)
  Shows the Interface of the route.
- **Metric**
  Enter the metric for the route. When creating the route, "not used" is entered automatically. The metric corresponds to the quality of a connection, for example speed, costs. If there are several equal routes, the route with the lowest metric value is used.
- **State** (only available online)
  Shows whether or not the route is active.

Route maps

General (IPv4)

Route maps

With route maps, you control how routing information is further processed. You can filter routing information and specify whether the information is further processed, modified or discarded.

Route maps operate according to the following principle:

- Routing information is compared with the filters of the route maps.
- The comparison is continued until the filters of a route map match the properties of an item of information.
- The information is then processed according to the route map settings:
  - The routing information is discarded.
  - The properties of the routing information are changed.

Settings

- **Name**
  Enter the name for the route map.
- **Sequence No**
  Enter a number for the route map.
  You can create several route maps with the same name but with different sequence numbers. The sequence numbers then specify the order in which the route maps are processed.

The table has the following columns:

- **Name**
  Shows the name of the route map.
- **Sequence No**
  Shows the sequence number of the route map.
- **Action**
  Specify what happens to the routing information that matches the settings of the route map:
  - permit
    The routing information is further processed according to the settings you make on the "Set" page.
  - deny
    The routing information is discarded.

Interface & Value Match (IPv4)

On this page, you specify whether or not the routing information for a route map is filtered according to interfaces, metric or tags.

Settings

- **Route Map (name/seq. no.)**
  Select a route map.
  The created route maps are available to you.
- **Type**
  Select the basis for the filtering:
  - Interface
  - Metric
  - Tag
- **Interface**
  Select an interface.
  Only active if you have selected the "Interface" entry for "Type".
- **Metric**
  Enter a value for the metric.
  Only active if you have selected the "Metric" entry for "Type".
- **Tag**
  Enter a value for the tag.
  Only active if you have selected the "Tag" entry for "Type".
- **Route Type**
  Select the type of the route.
  - Local
    The routing information for the route map is filtered according to directly connected routes (local interfaces).
  - Remote
    The routing information for the route map is filtered according to learned or statically configured routes.

  This box is active only if i you have selected the entry "Route Type in the "Type" drop-down list.

The table has the following columns:

- **Type**
  Shows the selected type:
  - Interface
  - Metric
  - Tag
  - Route Type
- **Value**
  Shows the selected interface, the value of the metric or of the tag.

Destination Match (IPv4)

On this page, you specify whether or not the routing information for a route map is filtered based on the destination IPv4 address.

Settings

- **Route Map (name/seq. no.)**
  Select a route map.
- **IP Address**
  Enter the IPv4 address of the destination on which the filtering is based.
- **Subnet Mask**
  Enter the subnet mask of the destination on which the filtering is based.

The table has the following columns:

- **IP Address**
  Shows the IPv4 address of the destination.
- **Subnet mask/prefix**
  Shows the subnet mask of the destination.

Next Hop Match (IPv4)

On this page, you specify whether or not the filtering for a route map will be based on the router to which the routing information is sent next.

Settings

- **Route Map (name/seq. no.)**
  Select a route map.
- **IP Address**
  Enter the IP address of the router to which the routing information will be sent next.

The table has the following columns:

- **IP Address**
  Shows the IP address of the next router.

Set (IPv4)

On this page, you specify whether or not the routing information will be changed by a route map.

If, for example, you have filtered based on a certain metric, you can change the value of the metric here. The routing information is then forwarded with the new value. You can only change the information of a "Permit" route map.

Settings

- **Route Map (name/seq. no.)**
  Select a route map.

The table has the following columns:

- **Name**
  Shows the name of the route map.
- **Sequence No**
  Shows the sequence number of the route map.
- **Metric**
  Enter the new value for the metric with which the routing information will be forwarded.
- **Tag**
  Enter the new value for the tag with which the routing information will be forwarded.

DHCP Relay Agent

General

If the DHCP server is in a different network, the device cannot reach the DHCP server. The DHCP relay agent intercedes between a DHCP server and the device. The DHCP relay agent forwards the port number of the device with the DHCP query to the DHCP server. If a DHCP server is unreachable, the device can switch to a different DHCP server.

Settings

- **DHCP Relay Agent (Opt. 82)**
  Enable or disable the DHCP relay agent.
- **Server IP Address**
  Enter the IPv4 address of the DHCP server.

The table has the following columns:

- **Server IP Address**
  Shows the IPv4 address of the DHCP server.

Option

Parameters of the DHCP relay agent

On this page, you can specify parameters for the DHCP server, for example the circuit ID. The circuit ID describes the origin of the DHCP query, for example which port received the DHCP query.

You specify the DHCP server on the "General" tab.

Settings

Global configuration

- **Circuit ID router index**
  Enable or disable the setting. When enabled, the router index is added to the generated circuit ID.
- **Circuit ID recipient VLAN ID**
  Enable or disable the setting. When enabled, the VLAN ID is added to the generated circuit ID.
- **Circuit ID receiving port**
  Enable or disable the setting. When enabled, the receiving port is added to the generated circuit ID.

  Note

  You need to select a least one option.

- **Remote ID**
  Shows the device ID.

Interface-specific configuration

- **Interface**
  Select the required IPv4 interface.

The table has the following columns:

- **Interface**
  Shows the interface.

  Note

  If you have not created an interface-specific configuration, the global configuration with the MAC address is used as the device ID.

- **Remote ID type**
  Select the type of the device ID. You have the following options:
  - IP Address
    The IPv4 address of the device is used as the device ID.

    Note

    No automatic updating

    There is no link between the "Remote ID" box and the currently set IP address.

    If the IP address is changed, the new IP address is not entered in the "Remote ID" box automatically. Only the value of the "Remote ID Type" box changes to "Free Text". To use the current IP address again, select "IP Address" again in the "Remote ID Type" box.

  - MAC Address
    The MAC address of the device is used as the device ID.
  - Free Text
    If you use "Free Text", you can enter the device name as the device identifier in the "Remote ID" box.
- **Remote ID**
  Enter the device name. Can only be edited if you select the entry "Free Text" for "Remote ID Type".
- **Circuit ID Type**
  Select the type of circuit ID from the drop-down list. You have the following options:
  - Predefined
    The circuit ID is created automatically based on the router index, VLAN ID or port.
  - Free Number
    If you use "Free Number", you can enter the ID for "Circuit ID".
- **Circuit ID**
  Enter the circuit ID. Can only be edited if you select the "Free Number" entry for the "Circuit ID Type".

VRRP

Router

On this page, you create virtual routers. You can configure other parameters in "Layer 3 (IPv4) > VRRP > Configuration".

Note

- VRRP is available only on layer 3.
- VRRP can only be used in conjunction with VLAN interfaces. Router ports are not supported.
- Simultaneous operation of VRRP and VRRPv3 is not possible.
- Select "VRRP" to configure VRRP

Settings

The following settings are available:

- **VRRP**
  Enable or disable routing using VRRP.
- **Reply to pings on virtual interfaces**
  When enabled, the virtual IPv4 addresses also reply to the ping.
- **Track**
  Enable or disable VRID tracking.
  When enabled, all interfaces of a VRID are monitored. When the link of an interface changes from "up" to "down", the priority of all VRRP interfaces with the same VRID is reduced to the value "0".
  When the link of an interface changes back from "down" to "up", the original priority of the VRRP interfaces is restored.
- **Interface**
  Select the required VLAN interface operating as virtual router.
- **VRID**
  Enter the ID of the virtual router. This ID defines the group of routers that form a virtual router (VR). In the group, this is the same. it can no longer be used for other groups.

The table has the following columns:

- **Interface**
  Shows the Interface that functions as the virtual router.
- **VRID**
  Shows the ID of the virtual router.
- **Virtual MAC Address**
  Shows the virtual MAC address of the virtual router.
- **Primary IP Address**
  Shows the numerically lowest IPv4 address in this VLAN. The entry 0.0.0.0 means that the "Primary" address on this VLAN is used. Otherwise all IPv4 addresses configured in this VLAN in "Layer 3 (IPv4) > Subnets" are valid values.

- **Router Status**
  Shows the current status of the virtual router. Possible values are as follows:
  - Master
    The router is the master router and handles the routing functionality for all assigned IP addresses.
  - Backup
    The router is the backup router. If the master router fails, the backup router takes over the tasks of the master router.
  - Initialize
    The virtual router has just been turned on. It will soon change to the "Master" or "Backup" status.
- **Master IP Address** (only available online)
  Shows the IPv4 address of the master router.
- **Priority**
  Shows the priority of the virtual router.
  If an IPv4 address is assigned to the VRRP router that is also actually configured on the local IPv4 interface, the value 255 is entered automatically. All other priorities can be distributed freely among the VRRP routers. The higher the priority, the earlier the VRRP router becomes "Master".
- **Advert. Interval**
  Shows the interval at which the master router sends VRRP packets.
- **Preempt**
  Shows the precedence of a router when changing roles between backup and master.
  - yes
    This router has precedence when changing roles.
  - no
    This router does not have precedence when changing roles.

Configuration

Introduction

On this page, you configure the virtual router.

Note

VRRP is available only on layer 3.

Description of the displayed values

The page contains the following boxes:

- **Interface / VRID**
  Select the ID of the virtual router to be configured.
- **Primary IP Address**
  Select the numerically lowest IPv4 address: If the router becomes master router, the router uses this IPv4 address.

  Note

  If you only configure one subnet on this VLAN, no entry is necessary. The entry is then 0.0.0.0.
  If you configure more than one subnet on the VLAN and you want a specific IPv4 address to be used as the source address for VRRP packets, select the IPv4 address from the drop-down list. Otherwise, the numerically lowest IPv4 address will be used.

- **Master**
  If this option is enabled, the numerically lowest IPv4 address is entered for "Associated IP Address". This means that the highest priority IPv4 address of the VRRP router is used as the virtual IP address of the virtual master router. The backup routers in this group must disable the option and use the IPv4 address of the router in "Associated IP address".
- **Priority**
  Enter the priority of this virtual router.
  If an IPv4 address is assigned to the VRRP router that is also actually configured on the local IPv4 interface, the value 255 is entered automatically. All other priorities can be distributed freely among the VRRP routers. The higher the priority, the earlier the VRRP router becomes "Master".
- **Advertisement Interval**
  Enter the interval in seconds after which a master router sends a VRRP packet again.
- **Preempt lower priority Master**
  Allow the precedence when changing roles between backup and master based on the selection process.
- **Track ID**
  Select a track ID.
- **Decrement Priority**
  Enter the value by which priority of the VRRP interface will be reduced.
- **Current Priority**
  Shows the priority of the VRRP interface after the monitored interface has changed to the "down" status.

Address overview

Overview

This page shows which IPv4 addresses the virtual router monitors.

Note

VRRP is available only on layer 3.

Settings

The table has the following columns:

- **Interface**
  Shows the Interface that functions as the virtual router.
- **VRID**
  Shows the ID of this virtual router.
- **Number of Addresses**
  Shows the number of IPv4 addresses.
- **Associated IP Address (1) ... Associated IP Address (4)**
  Shows the router IPv4 addresses monitored by this virtual router. If a router takes over the role of master, the routing function is taken over by this router for all these IPv4 addresses.

Address configuration

Note

VRRP is available only on layer 3.

Settings

The following settings are possible:

- **Interface / VRID**
  Select the required virtual router.
- **Associated IP Address**
  Enter the IPv4 address that the virtual router will monitor.

The table has the following columns:

- **Associated IP Address**
  Shows the IPv4 addresses that the virtual router monitors.

Interface Tracking

Introduction

On this page, you configure the monitoring of interfaces.

When the link of a monitored interface changes from "up" to "down", the priority of the assigned VRRP interface is reduced. You configure the value by which the priority is reduced on the page "Layer 3 > VRRP/VRRPv3 > Configuration".

When the link of the interface changes back from "down" to "up", the original priority of the VRRP interface is restored.

Note

This function is available only with layer 3.

Description of the displayed values

The page contains the following boxes:

- **Interface**
  From the drop-down list, select the interface to be monitored.
- **Track ID**
  Enter a track ID.
- **Track ID**
  Select a track ID.
- **Track Interface Count**
  Enter how many monitored interfaces need to change to the "down" status, before the priority is changed.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Track ID**
  Shows the track ID.
- **Interface**
  Shows the interface that is being monitored.

VRRPv3

Router (IPv4)

Introduction

On this page, you create new virtual routers. You can configure other parameters in "Layer 3 (IPv4) > VRRPv3 > Configuration".

Note

- This function is available only with layer 3.
- Simultaneous operation of VRRP and VRRPv3 is not possible.
- Select the "VRRPv3" check box to configure VRRPv3.
- You can use VRRPv3 on VLAN interfaces. Router ports are not supported.

Settings

- **VRRPv3**
  Enable or disable routing using VRRPv3. Can only be enabled when "Layer 3 (IPv6) > Configuration" "IPv6 Routing" is enabled.
- **Reply to pings on virtual interfaces**
  When enabled, the virtual IPv4 addresses also reply to the ping.
- **Track**
  Enable or disable VRID tracking.
  When enabled, all interfaces of a VRID are monitored. When the link of an interface changes from "up" to "down", the priority of all VRRP interfaces with the same VRID is reduced to the value "0".
  When the link of an interface changes back from "down" to "up", the original priority of the VRRP interfaces is restored.
- **Interface**
  Select the required VLAN interface operating as virtual router.
- **VRID**
  Enter the ID of the virtual router. This ID defines the group of routers that form a virtual router (VR). In the group, this is the same. It can no longer be used for other groups.

The table has the following columns:

- **Interface**
  Shows the IPv4 interface that functions as the virtual router.
- **VRID**
  Shows the ID of the virtual router.
- **Virtual MAC Address**
  Shows the virtual MAC address of the virtual router.

- **Primary IP Address**
  Shows the numerically lowest IPv4 address in this VLAN. The entry 0.0.0.0 means that the "Primary" address on this VLAN is used. Otherwise all IPv4 addresses configured on this VLAN in the "Layer 3 (IPv4) > Subnets" menu are valid values.
- **Router State**
  Shows the current status of the virtual router. Possible values are:
  - Master
    The router is the master router and handles the routing functionality for all assigned IPv4 addresses.
  - Backup
    The router is the backup router. If the master router fails, the backup router takes over the tasks of the master router.
  - Initialize
    The virtual router has just been turned on. It will soon change to the "Master" or "Backup" state.
- **Master IP Address**
  Shows the IPv4 address of the master router.
- **Priority**
  Shows the priority of the virtual router.
  If an IPv4 address is assigned to the VRRP router that is also actually configured on the local IPv4 interface, the value 255 is entered automatically. All other priorities can be distributed freely among the VRRP routers. The higher the priority, the earlier the VRRP router becomes "Master".
- **Advert. Interval**
  Shows the interval at which the master router sends VRRPv3 packets.
- **Preempt**
  Shows the precedence of a router when changing roles between backup and master.
  - yes
    This router has precedence when changing roles.
  - no
    This router does not have precedence when changing roles.

Configuration (IPv4)

Introduction

On this page, you configure the virtual router.

Note

This function is available only with layer 3.

Settings

The page contains the following:

- **Interface / VRID**
  Select the ID of the virtual router to be configured.
- **Primary Address**
  Select the primary IPv4 address. If the router becomes master router, the router uses this IPv4 address.

  Note

  If you only configure one subnet on this VLAN, no entry is necessary. The entry is then 0.0.0.0.
  If you configure more than one subnet on the VLAN and you want a specific IPv4 address to be used as the source address for VRRP packets, select the IPv4 address. Otherwise, the numerically lowest IPv4 address will be used.
- **Master**
  If enabled, the numerically lowest IPv4 address is entered for "Associated IP Address". This means that the numerically lowest IPv4 address of the VRRP router is used as the virtual IP address of the virtual master router. The backup routers in this group must disable the option and use the IPv4 address of the router in "Associated IP address".
- **Priority**
  Enter the priority of this virtual router. Valid values are 1-254.
  If an IPv4 address is assigned to the VRRPv3 router that is also actually configured on the local IPv4 interface, the value 255 is entered automatically. All other priorities can be distributed freely among the VRRPv3 routers. The higher the priority, the earlier the VRRPv3 router becomes "Master".
- **Advertisement interval**
  Enter the interval in seconds after which a master router sends a VRRPv3 packet again.
- **Preempt lower priority Master**.
  Allow precedence when changing roles between backup and master based on the selection process.
- **VRRP Compatible Mode**
  When enabled, the VRRPv3 router also sends and receives VRRPv2 frames in addition to VRRPv3 frames for configured IPv4 addresses. Only necessary when some VRRP routers do not support VRRPv3
- **Track ID**
  Select a track ID.
- **Decrement Priority**
  Enter the value by which priority of the VRRP interface will be reduced.
- **Current Priority**
  Shows the priority of the VRRP interface after the monitored interface has changed to the "down" status.

Addresses Overview (IPv4)

Overview

This page shows which IPv4 addresses the virtual router monitors. Each virtual router can monitor a maximum of 10 IPv4 addresses.

Note

This function is available only with layer 3.

Settings

The table has the following columns:

- **Interface**
  Shows the Interface that functions as the virtual router.
- **VRID**
  Shows the ID of this virtual router.
- **Number of Addresses**
  Shows the number of IPv4 addresses.
- **Associated IP Address (1) ...Associated IP Address (10)**
  Shows the router IPv4 addresses monitored by this virtual router. If a router takes over the role of master, the routing function is taken over by this router for all these IPv4 addresses.

Addresses Configuration (IPv4)

Creating or changing the monitored IP addresses

On this page, you can create, modify or delete the IPv4 addresses to be monitored.

Note

This function is available only with layer 3.

Settings

The page contains the following:

- **Interface / VRID**
  Select the ID of the virtual router.
- **Associated IP Address**
  Enter the IPv4 address that the virtual router will monitor.

The table has the following columns:

- **Associated IP Address**
  Shows the IPv4 addresses that the virtual router monitors.

Interface Tracking (IPv4)

Introduction

On this page, you configure the monitoring of interfaces.

When the link of a monitored interface changes from "up" to "down", the priority of the assigned VRRP interface is reduced. You configure the value by which the priority is reduced on the page "Layer 3 > VRRP/VRRPv3 > Configuration".

When the link of the interface changes back from "down" to "up", the original priority of the VRRP interface is restored.

Note

This function is available only with layer 3.

Description of the displayed values

The page contains the following boxes:

- **Interface**
  From the drop-down list, select the interface to be monitored.
- **Track ID**
  Enter a track ID.
- **Track ID**
  Select a track ID.
- **Track Interface Count**
  Enter how many monitored interfaces need to change to the "down" status, before the priority is changed.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Track ID**
  Shows the track ID.
- **Interface**
  Shows the interface that is being monitored.

OSPFv2

Configuration

Introduction

On this page, you configure routing with OSPF.

Note

OSPF is available only on layer 3.

Settings

- **OSPFv2**
  Enable or disable routing using OSPFv2.
- **Router ID**
  Enter the name of one of the OSPFv2 interfaces. The name is entered in the IP address format and does not need to match the local IP address.
- **OSPFv2 RFC1583 Compatibility**
  Enable the option if you still have old OSPFv2 routers in operation that are not compatible with RFC 2328.
- **Border Router**
  Displays the status of the OSPFv2 router. If the local system is an active member in at least 2 areas, this is an area border router.
- **AS Border Router**
  Specify whether the router is an AS border router. An AS border router intercedes between multiple autonomous systems, for example if you have an additional RIP network. An AS border router is also necessary to add and to distribute static routes.
- **New LSA Received** (only available online)
  Shows the number of received LSAs.
  Updates and local LSAs are not counted.
- **New LSA Configured** (only available online)
  Number of different LSAs sent by this local system.
- **External LSA Maximum**
  To limit the number of entries of external LSAs in the database, enter the maximum number of external LSAs.
- **Exit Interval [s]**
  Enter the interval after which the OSPFv2 router once again attempts to come out of the overflow status. A 0 means that the OSPFv3 router attempts to exit the overflow status only following a restart.
- **Inbound Filter**
  Select a route map that filters inbound routes.

- **Redistribute Routes**
  Specify which known routes are distributed using OSPFv2. You can make different decisions for the route types Default, Connected and Static.

  Note

  The settings can only be enabled on an AS border router. Enabling the Default and Static options, in particular, can cause problems if they are enabled at too many points in the network, for example, forwarding loops.

- **Route Map**
  Select a route map that filters which routes are forwarded using OSPFv2.

Areas

Overview

An autonomous system can be divided into smaller areas.

On this page, you can view, create, modify or delete the areas of the router.

Note

OSPF is available only on layer 3.

Settings

- **Area ID**
  Enter the designation of the area. The database is synchronized for all routers of an area. The area ID must be unique in the network.
  The area identifier is a 32-bit number with the following format: x.x.x.x where x = 0 ... 255
  The area ID 0.0.0.0 is reserved for the backbone area and cannot be deleted.

This table contains the following columns:

- **Area ID**
  Shows the ID of the area.
- **Area Type**
  Select the area type from the drop-down list.
  - Normal
  - Stub
  - NSSA
  - Backbone
- **Summary**
  Specify whether summary LSAs are generated for this area.
  - Summary: Summary LSAs are sent to the area.
  - No Summary: Summary LSAs are not sent to the area.

- **Metric**
  Displays the costs for the OSPFv2 interface.
- **Updates**
  Shows the number of recalculations of the routing tables.
- **LSA Count**
  Shows the number of LSAs in the database.
- **Area BR**
  Shows the number of reachable area border routers (ABR) within this area.
- **AS BR**
  Shows the number of reachable autonomous system border routers (ASBR) in this area.

Area range

Creating a new OSPFv2 area range

On this page, networks can be grouped together under one area ID. The method is used only with area border routers. This means that an area border router only advertises one route for grouped areas to the outside.

Note

OSPF is available only on layer 3.

Settings

- **Area ID**
  Select the area ID of the area. You configure the ID in "Layer 3 > OSPFv2 > Areas".
- **Subnet Address**
  Enter the IPv4 address of the network that will be grouped.
- **Subnet Mask**
  Enter the subnet mask of the network that will be grouped.

This table contains the following columns:

- **Area ID**
  Shows the ID of the area.
- **Subnet Address**
  Shows the IPv4 address of the network that will be grouped with other networks.
- **Subnet Mask**
  Shows the subnet mask of the network that will be grouped with other networks.
- **Advertise**
  Enable this setting to advertise the grouped network.

Interfaces

Overview

On this page, you can configure OSPFv2 interfaces.

Note

OSPF is available only on layer 3.

Settings

- **IP Address**
  Select the IPv4 address of the OSPFv2 interface.
- **Area ID**
  Select the area ID of the area with which the OSPFv2 interface is connected.

  Note

  For the secondary interface select the same Area ID as for the corresponding primary interface.
  The information whether a primary or secondary interface is involved can be found in the "Address Type" column on the "Layer 3 (IPv4) > Subnets > Overview" page.

This table contains the following columns:

- **IP Address**
  Shows the IPv4 address of the OSPFv2 interface
- **Area ID**
  Select the area ID of the area with which the OSPFv2 interface is connected.
- **OSPF Status**
  Specify whether or not OSPFv2 is active on the interface.
  - Enabled: OSPFv2 is enabled on the interface.
  - Disabled: OSPFv2 is disabled on the interface.
- **Metric**
  Enter the costs for the OSPFv2 interface.
- **Priority**
  Enter the router priority. The priority is only relevant for selecting the designated router or designated border router. This parameter can be selected differently on routers within the same subnet.
- **Trans. Delay**
  Enter the required delay when sending a connection update.
- **Retrans. Delay**
  Enter the time after which an OSPFv2 packet is transferred again if no confirmation was received.

- **Hello Interval**
  Enter the interval between two Hello packets.
- **Dead Interval**
  Enter the interval after which the neighbor router is marked as "failed" if no more Hello packets are received from it during this time.

Interface authentication

Configuring the interface login

On this page, you define the authentication of the interface.

Note

OSPF is available only on layer 3.

Settings

- **OSPF Interface**
  Select the OSPFv2 interface for which you want to configure authentication.
- **Authentication type**
  Select the authentication method of the OSPF interface. You have the following options:
  - None: No authentication
  - simple: Authentication using an unencrypted password
  - MD5: Authentication using MD5

Simple authentication

- **Password**
  Enter a password.
- **Confirmation**
  Confirm the entered password.

MD5 authentication

- **Authentication Key ID**
  Enter the identifier of the MD5 authentication key.
  Enter the ID for MD5 authentication with which the password will be used as a key. Since the key ID is transferred with the protocol, the same key must be stored under the same key ID on all neighboring routers.

The table has the following columns:

- **Authentication Key ID**
  Can only be edited if you set the MD5 authentication method. It is only possible to use several keys there.
- **MD5 Key**
  Enter the MD5 key.

- **Confirm MD5 Key**
  Confirm the entered key.
- **Youngest Key ID** (only available online)
  Shows whether or not the MD5 key is the latest key ID.

Virtual links

Overview

Due to the protocol, each area border router must have access to the backbone area. If a router is not connected directly to the backbone area, a virtual link to it is created.

Note

OSPF is available only on layer 3.

Note

Note that when creating a virtual link both the transit area and the backbone area must already be configured.

the virtual link must be configured identically at both ends.

Settings

- **Virtual links are not effective unless the device is an ABR.** (only available online)
  This is displayed when at least one virtual link entry is configured and the device is not an area border router.
- **Neighbor Router ID**
  Enter the ID of the neighbor router at the other end of the virtual connection.
- **Transit Area ID**
  Select the ID of the area that connects the two routers.

This table contains the following columns:

- **Transit Area ID**
  Shows the ID via which the two routers are connected.
- **Neighbor Router ID**
  Shows the ID of the neighbor router at the other end of the virtual connection.
- **Virt. Link Status**
  Specify the status of the virtual link. The following statuses are possible:
  - down: The virtual link is inactive.
  - point-to-point: The virtual link is active.
- **Trans. Delay**
  Enter the required delay when sending a link update packet.
- **Retrans. Delay**
  Enter the time after which a packet is transferred again if no confirmation was received.

- **Hello Interval**
  Enter the interval between two Hello packets.
- **Dead Interval**
  Enter the interval after which the neighbor router counts as "failed" if no more Hello packets are received from it during this time.

Virtual Link Authentication

Configuring the virtual link login

On this page, you define the authentication for the virtual link.

Note

OSPF is available only on layer 3.

Settings

- **Virtual Link (Area/Neighbor)**
  Select the virtual link for which you want to configure authentication.
- **Authentication type**
  Select the authentication method of the OSPF interface. You have the following options:
  - None: No authentication
  - simple: Authentication using an unencrypted password.
  - MD5: Authentication using MD5

Simple authentication

- **Password**
  Enter the password.
- **Confirmation**
  Confirm the entered password.

MD5 authentication

- **Authentication Key ID**
  Enter the identifier of the MD5 authentication key.
  Enter the ID for MD5 authentication with which the password will be used as a key.
  Since the key ID is transferred with the protocol, the same key must be stored under the same key ID on all neighboring routers.

The table has the following columns:

- **Authentication Key ID**
  Can only be edited if you set the MD5 authentication method. It is only possible to use several keys there.
- **MD5 Key**
  Enter the MD5 key.

- **Confirm MD5 Key**
  Confirm the entered key.
- **Youngest Key ID**
  Shows whether or not the MD5 key is the latest key ID.

RIPv2

Configuration

On this page, you configure routing with RIP.

Note

RIPv2 is available only on layer 3.

Settings

- **RIPv2**
  Enable or disable routing using RIPv2.
- **Inbound Filter**
  Select a route map that filters inbound routes.
- **Redistribute Routes**
  Specify which known routes are distributed using RIPv2.
  The following types of route exist:
  - Static Default
  - Connected
  - Static
  - OSPF
- **Route Map**
  Select a route map that filters which routes are forwarded using RIPv2.

Interfaces

Overview

On this page, you can configure RIPv2 interfaces.

Note

RIPv2 is available only on layer 3.

Settings

- **IP Address**
  Select the IPv4 address of the RIPv2 interface.

This table contains the following columns:

- **IP Address**
  Shows the IPv4 address of the RIPv2 interface
- **Send Updates**
  Select the way in which updates are sent:
  - no send
    No updates are sent.
  - RIPv1
    Updates for RIPv1 are sent.
  - RIPv1-compat.
    RIPv2 updates are sent as broadcasts according to the rules of RIPv1.
  - RIPv2
    Updates for RIPv2 are sent as multicasts.
  - RIPv1 demand/RIPv2 demand
    RIP packets are sent only as a response to an explicit query.
- **Receive Updates**
  Select the form in which received updates are accepted:
  - no receive
    No updates are received.
  - RIPv1
    Only updates of RIPv1 are received.
  - RIPv2
    Only updates of RIPv2 are received.
  - RIPv1/v2
    Updates of RIPv1 and RIPv2 are received.
- **Default Metric**
  Enter the costs for the RIPv2 interface.

NAT

NAT

On this WBM page, you specify the basic settings for NAT.

Description

The page contains the following boxes:

- **NAT**
  Enable or disable NAT/NAPT for the entire device. When enabled, the device operates as a NAT router.
- **Idle Timeout [s]**
  Enter the required time. If no data exchange takes place, when this time has elapsed ...
- **TCP Timeout [s]**
  Enter the required time. If no data exchange takes place, the TCP connection is deleted from the "NAT Translations" table when this time has elapsed.
- **UDP Timeout [s]**
  Enter the required time. If no data exchange takes place, the UDP connection is deleted from the "NAT Translations" table when this time has elapsed.
- **Interface**
  Select the an IP interface from the drop-down list:
- **NAT**
  Enable or disable NAT for an IP interface.
  An entry is created automatically in the "Pool" tab. The device can be reached from the external network using the IP address of the IP interface.
  If you disable NAT for an IP interface and there are no configurations on the NAT interface, the entry is automatically deleted from the table.
  NAPT
  Enable or disable NAPT for an IP interface.

The table has the following columns:

- **Interface**
  Interface to which the setting relates.
- **NAT**
  Shows whether NAT is enabled or disabled for the selected IP interface.
  NAT is only enabled, when you have enabled NAT for the entire device.
- **NAPT**
  Shows whether NAPT is enabled or disabled for the selected IP interface.
  NAPT is only enabled, when you have enabled NAT for the entire device.
  If you do not create further configurations for NAPT, the dynamic port translation is enabled. As default, the device cannot be reached from an external network. If the device wants to communicate in an external network, the inside local address and the IP address of the IP interface have a port added and the device is assigned as inside local and inside global address. Using this inside global address, the device can be reached from the external network until the timer of the connection elapses.

Static

On this WBM page, you configure static address translatiions.

You specify which inside global address the inside local address of a device will be converted to and vice versa. This variant allows connection establishment in both directions. The device in the internal network can be reached from the external network.

Description

The page contains the following boxes:

- **Interface**
  Select the an IP interface from the drop-down list:
- **Inside Local Address**
  Enter the actual address of the device that should be reachable from external.
- **Inside Global Address**
  Enter the address at which the device can be reached from external.

The table has the following columns:

- **1st column**
  Select the check box in the row to be deleted.
- **Interface**
  Interface to which the setting relates.
- **Inside Local Address**
  Shows the actual address of the device that should be reachable from external.
- **Inside Global Address**
  Shows the address at which the device can be reached from external.

Pool

On this WBM page, you configure static address translations.

As default, the device cannot be reached from an external network. If the device wants to communicate in an external network, an inside global address is assigned to it dynamically. Using this inside global address, the device can be reached from the external network until the timer of the connection elapses.

Description

The page contains the following boxes:

- **Interface**
  Select the an IP interface from the drop-down list:
- **Inside Global Address**
  Enter the start address for the dynamic assignment of addresses at which devices will be reachable from external.
- **Inside Local Address Mask**
  Enter the address mask of the external subnet.

The table has the following columns:

- **1st column**
  Select the check box in the row to be deleted.
- **Interface**
  Interface to which the setting relates.

- **Inside Global Address**
  Shows the start address for the dynamic assignment of addresses at which devices will be reachable from external.
- **Inside Local Address Mask**
  Shows the address mask of the external subnet.

NAPT

On this WBM page, you configure static address translations.

Description

The page contains the following boxes:

- **Interface**
  Select the an IP interface from the drop-down list:
- **Inside Local Address**
  Enter the actual address of the device that should be reachable from external.
- **Service**
  Select the service for which the port translation is valid.
  When you select a service, the same port is entered in the Start Port and End Port boxes.
  if you change the start port, the end port is changed accordngly.
  if you select the entry "-", you can enter the start and end port freely.
- **Start Port**
  Enter a port.
- **End Port**
  Depending on your selection in the "Service" drop down list, you can enter a port or a port is displayed.
  If you enter different ports in the Start Port and End Port boxes, the same port range is entered in the Inside Global Port box. A port range can only be translated to the same port range.
  If you enter the same port in the Start Port and End Port boxes, you can enter any Inside Global Port.
- **Inside Global Port**
  Depending on your selection in the "Service" drop down list, you can enter a port or a port is displayed.
- **Protocol**
  Select the protocol for which the port translation is valid.
- **Description**
  Enter a description for the port translation.

The table has the following columns:

- **1st column**
  Select the check box in the row to be deleted.
- **Interface**
  Interface to which the setting relates.

- **Inside Local Address**
  Shows the actual address of the device that should be reachable from external.
- **Start Port**
  Shows the start port that will be assigned to the inside local address.
- **End Port**
  Shows the end port that will be assigned to the inside local address.
- **Protocol**
  Shows the protocol for which the port translation applies.
- **Inside Global Address**
  Shows the address at which the device will be reached from external.
- **Inside Global Port**
  Shows the port that will be assigned to the Inside Global Address.
- **Description**
  Shows a a description for the port translation.

Configuring layer 3 functions for IPv6

Configuration

Introduction

The page contains the overview of the layer 3 functions for IPv6. On this page, you enable or disable the required layer 3 function.

The functions "IPv6 Routing", "VRRPv3" and "OSPFv3" are only available with Layer 3 (IPv6).

Settings

- **IPv6 Routing** (only available with devices with a layer 3 license)
  - Enabled
    IPv6 routing is enabled. If IPv4 routing is not enabled on the device this is also enabled with this function.
  - Disabled
    IPv6 routing is disabled.
- **VRRPv3** (only available with devices with a layer 3 license)
  Enable or disable routing using VRRPv3. To use VRRPv3 , first enable the IPv6 routing function. You can configure other settings in "Layer 3 (IPv6) > VRRPv3".
- **OSPFv3** (only available with devices with a layer 3 license)
  Enable or disable routing using OSPFv3. You can configure other settings in "Layer 3 (IPv6) > OSPFv3".

Subnets (IPv6)

Connected Subnets

On this page, you can enable IPv6 on the interface. The interface is also called an IPv6 interface. An IPv6 interface can have several IPv6 addresses.

Settings

The page contains the following:

- **Interface**
  Select the IP interface for which IPv6 will be enabled.
- **IPv6 Enable**
  Enable or disable IPv6 for this interface.

  Note

  Disabling IPv6

  As soon as IPv6 is enabled on the IP interface, IPv6 can only be disabled by removing the interface.

- **IPv6 Address**
  Enter the IPv6 address. The entry depends on the selected address type.
- **Prefix Length**
  Enter the number of left-hand bits belonging to the prefix
- **IPv6 Address Type**
  Select the address type.
  - Unicast
  - Anycast
  - Link Local IPv6 address is only valid on the link

The table has the following columns:

- **Interface Name**
  Shows the name of the interface.
- **IPv6 Address**
  Shows the IPv6 address of the subnet.
- **Prefix Length**
  Shows the prefix length.
- **IPv6 Address Type**
  Displays the address type. The following values are possible:
  - Unicast
  - Anycast
  - Link Local

DHCPv6 client

DHCPv6 client

On this page, you specify whether the DHCPv6 client receives not only the configuration settings but also the IPv6 address or supports the "prefix delegation" function.

Prefix delegation

The DHCPv6 server delegates the distribution of the IPv6 prefix to the DHCPv6 client. To do this an IPv6 interface must be configured on the DHCPv6 client as a PD client. Via the PD client, the DHCPv6 client obtains an IPv6 prefix from the DHCPv6 server and stores it under a prefix name. The IPv6 interfaces of the DHCPv6 client the should adopt the prefix are configured as so-called PD sub clients.

You configure the PD sub-clients in "Layer 3 (IPv6)" > "DHCPv6 Client" > "DHCPv6 PD Sub Client".

Settings

- **Interface**
  Select the IP interface via which the DHCPv6 client functions.
- **Enable**
  Enable or disable the DHCPv6 client for the relevant interface.
- **Mode**
  Specify the procedure.
  - Status dependent Obtains the IPv6 address and the configuration file from the DHCPv6 server
  - Stateless: Obtains only the configuration file from the DHCPv6 server
  - Prefix Delegation: The DHCPv6 client takes over the distribution of the IPv6 prefix
- **Prefix Name**
  Enter the prefix name that will be further distributed by the DHCPv6 router. Can only be edited with the "Prefix Delegation" procedure. The DHCPv6 router stores IPv6 prefix of the DHCPv6 server under this name.
- **Rapid Commit**
  When enabled the procedure for the IPv6 address assignment is shortened. Instead of 4 DHCPv6 messages (SOLICIT, ADVERTISE , REQUEST, REPLY) only 2 DHCPv6 messages (SOLICIT, REPLY) are used. You will find further information on the messages in RFC 3315.

DHCPv6 PD Sub Client

On this page you specify which IPv6 interfaces of the DHCPv6 client will adopt the prefix.

This achieved by assigning the prefix name followed by a prefix and the required prefix length to them as the IP address. Here, the first bits of the configured prefix are replaced by the values stored in the prefix name.

The PD sub client forms the IPv6 address from the prefix name, prefix and prefix length.

Example:

Under the prefix name "test", the following IPv6 prefix is stored: 2001:DB8:1234::/48

Prefix = 2002:AF9:5678::1

Prefix length = 64

From these settings the PD sub client forms the following IPv6 address: 2001:DB8:1234::1

Settings

- **Interface**
  Select the interface that functions as the PD sub client. Only IP interfaces can be selected.
- **Prefix Name**
  Enter the prefix name that will be further distributed by the PD router. The PD router stores IPv6 prefix of the DHCPv6 server under this name.
- **Prefix**
  Specify the network identifier of the host.
- **Prefix Length**
  Enter the number of left-hand bits belonging to the prefix

This table contains the following columns

- **Interface Name**
  Shows the name of the interface
- **Prefix Name**
  Shows the name of the prefix.
- **Prefix**
  Shows the prefix.
- **Prefix Length**
  Shows the prefix length.
- **Generated Address** (only available online)
  Shows the IPv6 address formed from the prefix name, prefix and prefix length.

Static Routes (IPv6)

On this page, you configure static IPv6 routes.

Settings

The page contains the following:

- **Destination Network**
  Enter the network address of the destination that can be reached via this route.
- **Prefix Length**
  Enter the number of left-hand bits belonging to the prefix
- **Gateway**
  Enter the IPv4 address of the gateway via which this network address is reachable.

- **Metric**
  Enter the metric for the route. The metric corresponds to the quality of a connection, based for example on speed or costs. If there are several equal routes, the route with the lowest metric value is used.
- **Interface**
  Specify the IPv6 interface via which the network address of the destination is reached.

This table contains the following columns:

- **Destination Network**
  Shows the IPv6 network address of the destination.
- **Prefix Length**
  Shows the prefix length.
- **Gateway**
  Shows the IPv6 address of the next gateway.
- **Interface**
  Shows the Interface of the route.
- **Metric**
  Enter the metric for the route. When creating the route, "not used" is entered automatically. The metric corresponds to the quality of a connection, based for example on speed or costs. If there are several equal routes, the route with the lowest metric value is used.
- **Status**
  Shows whether or not the route is active.

Route maps

General (IPv6)

Route maps

With route maps, you control how routing information is further processed. You can filter routing information and specify whether the information is further processed, modified or discarded.

Route maps operate according to the following principle:

- Routing information is compared with the filters of the route maps.
- The comparison is continued until the filters of a route map match the properties of an item of information.
- The information is then processed according to the route map settings:
  - The routing information is discarded.
  - The properties of the routing information are changed.

Settings

- **Name**
  Enter the name for the route map.
- **Sequence No**
  Enter a number for the route map.
  You can create several route maps with the same name but with different sequence numbers. The sequence numbers then specify the order in which the route maps are processed.

The table has the following columns:

- **Name**
  Shows the name of the route map.
- **Sequence No**
  Shows the sequence number of the route map.
- **Action**
  Specify what happens to the routing information that matches the settings of the route map:
  - permit
    The routing information is further processed according to the settings you make on the "Set" page.
  - deny
    The routing information is discarded.

Interface & Value Match (IPv6)

On this page, you specify whether or not the routing information for a route map is filtered according to interfaces, metric or tags.

Settings

- **Route Map (name/seq. no.)**
  Select a route map.
  The created route maps are available to you.
- **Type**
  Select the basis for the filtering:
  - Interface
  - Metric
  - Tag
- **Interface**
  Select an interface.
  Only active if you have selected the "Interface" entry for "Type".
- **Metric**
  Enter a value for the metric.
  Only active if you have selected the "Metric" entry for "Type".

- **Tag**
  Enter a value for the tag.
  Only active if you have selected the "Tag" entry for "Type".
- **Route Type**
  Select the type of the route.
  - Local
    The routing information for the route map is filtered according to directly connected routes (local interfaces).
  - Remote
    The routing information for the route map is filtered according to learned or statically configured routes.

  This box is active only if i you have selected the entry "Route Type in the "Type" drop-down list.

The table has the following columns:

- **Type**
  Shows the selected type:
  - Interface
  - Metric
  - Tag
  - Route Type
- **Value**
  Shows the selected interface, the value of the metric or of the tag.

Destination Match (IPv6)

On this page, you specify whether or not the routing information for a route map is filtered based on the destination IPv6 address.

Settings

- **Route Map (Name/Seq. No.)**
  Select a route map.
- **IP Address**
  Enter the IPv6 address of the destination on which the filtering is based.
- **Prefix Length**
  Enter the number of left-hand bits belonging to the prefix

The table has the following columns:

- **IP Address**
  Shows the IPv6 address of the destination.
- **Prefix Length**
  Shows the prefix length.

Next Hop Match (IPv6)

On this page, you specify whether or not the filtering for a route map will be based on the router to which the routing information is sent next.

Settings

- **Route Map (name/seq. no.)**
  Select a route map.
- **IP Address**
  Enter the IP address of the router to which the routing information will be sent next.

The table has the following columns:

- **IP Address**
  Shows the IP address of the next router.

Set (IPv6)

On this page, you specify whether or not the routing information will be changed by a route map.

If, for example, you have filtered based on a certain metric, you can change the value of the metric here. The routing information is then forwarded with the new value. You can only change the information of a "Permit" route map.

Settings

- **Route Map (name/seq. no.)**
  Select a route map.

The table has the following columns:

- **Name**
  Shows the name of the route map.
- **Sequence No**
  Shows the sequence number of the route map.
- **Metric**
  Enter the new value for the metric with which the routing information will be forwarded.
- **Tag**
  Enter the new value for the tag with which the routing information will be forwarded.

DHCPv6 Relay Agent

Interfaces

A DHCPv6 relay agent forwards the DHCP queries of the DHCPv6 client to the DHCPv6 server. When forwarding the remote ID is inserted in the DHCP query. The remote ID is the identifier of the interface. With this, the DHCPv6 agent can send the reply of the DHCPv6 server to the required DHCPv6 client.

Settings

- **Remote ID (Opt. 37)**
  When enabled the DHCP relay agent option 37 (Remote ID) is used. Specify the type of identifier in "Remote ID Type".
- **Interface**
  Select the IPv6 interface.

The table has the following columns:

- **Interface**
  Interface to which the settings relate.
- **Current Remote ID**
  Shows which remote ID is currently being used.
- **Remote ID Type**
  Select hat is used as the identifier for the IPv6 interface. The following options are available:
  - DUID
    DHCP Unique Identifier
  - System Name
    System name of the device
  - Management IP
    Management IPv6 address of the device
  - User Defined
    User-defined entry
- **DHCP Unique Identifier**
  Enter the DUID (DHCP Unique Identifier).
- **User Defined**
  Enter a user-defined entry as the identifier.

Server Addresses

On this page you enable the DHCPv6 relay agent on the interface. You also specify which DHCPv6 server can be reached via it.

Settings

- **Relay Interface**
  Select the interface on which the DHCPv6 relay agent accepts DHCPv6 requests.
- **Server IP Address**
  Enter the IPv6 address of the DHCPv6 server to which the DHCPv6 requests will be forwarded. The address can be either the unicast or the link local multicast address. With the link local multicast address you need to specify an interface, via which the DHCPv6 server can be reached. You configure the interface in "Layer 3 (IPv6) > DHCPv6 Relay Agent > Outgoing Interfaces".

The table has the following columns:

- **Server IP Address**
  Shows the IP64 address of the DHCPv6 server.

Outgoing Interfaces

Outgoing Interfaces

The link local multicast address (ff02::1:2) of the DHCPv6 server can only be reached by devices located on the same link. On this page, you specify the IPv6 interface via which this DHCPv6 server can be reached.

Settings

- **Relay Interface**
  Select the IPv6 interface on which the DHCPv6 relay agent accepts DHCPv6 requests.
- **Server IP Address**
  Select the IPv6 address of the DHCPv6 server.
- **Outgoing Interface**
  Select the interface via which the DHCPv6 server can be reached.

The table has the following columns:

- **Server IP Address**
  Shows the IPv6 address of the DHCPv6 server.
- **Outgoing Interface**
  Shows the IPv6 interface via which the DHCPv6 server can be reached.

VRRPv3

Router (IPv6)

Introduction

On this page, you create new virtual routers. You can configure other parameters in "Layer 3 (IPv6) > VRRPv3 > Configuration".

Note

- This function is available only with layer 3.
- Simultaneous operation of VRRP and VRRPv3 is not possible.
- Select the "VRRPv3" check box to configure VRRPv3.
- You can use VRRPv3 on VLAN interfaces. Router ports are not supported.

Settings

- **VRRPv3**
  Enable or disable routing using VRRPv3. Can only be enabled when "Layer 3 (IPv6) > Configuration" "IPv6 Routing" is enabled.
- **Track**
  Enable or disable VRID tracking.
  When enabled, all interfaces of a VRID are monitored. When the link of an interface changes from "up" to "down", the priority of all VRRP interfaces with the same VRID is reduced to the value "0".
  When the link of an interface changes back from "down" to "up", the original priority of the VRRP interfaces is restored.
- **Interface**
  Select the required VLAN interface operating as virtual router.
- **VRID**
  Enter the ID of the virtual router. This ID defines the group of routers that form a virtual router (VR). In the group, this is the same. It can no longer be used for other groups.

The table has the following columns:

- **Interface**
  Shows the Interface that functions as the virtual router.
- **VRID**
  Shows the ID of the virtual router.
- **Virtual MAC Address** (only available online)
  Shows the virtual MAC address of the virtual router.
- **Primary IP Address**
  Shows the numerically lowest IPv6 address in this VLAN. The entry "::" means that the "Primary" address is used on this VLAN. Otherwise all IPv6 addresses configured on this VLAN in the "Layer 3 (IPv6) > Subnets" menu are valid values.
- **Router State** (only available online)
  Shows the current status of the virtual router. Possible values are:
  - Master
    The router is the master router and handles the routing functionality for all assigned IPv4 addresses.
  - Backup
    The router is the backup router. If the master router fails, the backup router takes over the tasks of the master router.
  - Initialize
    The virtual router has just been turned on. It will soon change to the "Master" or "Backup" state.
- **Master IP Address** (only available online)
  Shows the IPv6 address of the master router.
- **Priority** (only available online)
  Shows the priority of the virtual router.
  The current master router is automatically given 255. All other priorities can be distributed freely among the VRRPv3 routers. The higher the priority, the earlier the VRRPv3 router becomes "Master".

- **Advert. Internal**
  Shows the interval at which the master router sends VRRPv3 packets.
- **Preempt**
  Shows the precedence of a router when changing roles between backup and master.
  - yes
    This router has precedence when changing roles.
  - no
    This router does not have precedence when changing roles.
- **Reply to pings on virtual interfaces (Accept Mode)**
  Shows the precedence of a router when changing roles between backup and master.
  - Yes
    Virtual IP addresses also reply to pings.
  - No
    Virtual IP addresses do not reply to pings.

Configuration (IPv6)

Introduction

On this page, you configure the virtual router.

Note

This function is available only with layer 3.

Settings

- **Interface / VRID**
  Select the ID of the virtual router to be configured.
- **Primary Address**
  Select the primary IPv6 address. If the router becomes master router, the router uses this IPv4 address.

  Note

  If you only configure one subnet on this VLAN, no entry is necessary. The entry is then **::**. If you configure more than one subnet on the VLAN and you want a specific IPv6 address to be used as the source address for VRRPv3 packets, select the IPv6 address. Otherwise, the IPv6 address with priority will be used.
- **Master**
  If this option is enabled, the numerically lowest IPv6 address is entered for "Associated IP Address". This means that the highest priority IPv6 address of the VRRP router is used as the virtual IPv6 address of the virtual master router. The backup routers in this group must disable the option and use the IPv6 address of the router in "Associated IP Address".

- **Priority**
  Enter the priority of this virtual router.
  The current master router is always given 255. All other priorities can be distributed freely among the redundant routers. The higher the priority, the earlier the router becomes "Master".
- **Advertisement interval**
  Enter the interval in seconds after which a master router sends a VRRPv3 packet again.
- **Preempt lower priority Master**.
  Allow precedence when changing roles between backup and master based on the selection process.
- **Reply to pings on virtual interfaces (Accept Mode)**
  When enabled, the virtual IPv6 addresses also reply to the ping.
- **Track ID**
  Select a track ID.
- **Decrement Priority**
  Enter the value by which priority of the VRRP interface will be reduced.
- **Current Priority**
  Shows the priority of the VRRP interface after the monitored interface has changed to the "down" status.

Addresses Overview (IPv6)

Overview

This page shows which IPv6 addresses the virtual router monitors.

Note

This function is available only with layer 3.

Displayed values

The table has the following columns:

- **Interface**
  Shows the Interface that functions as the virtual router.
- **VRID**
  Shows the ID of this virtual router.
- **Number of Addresses**
  Shows the number of IPv6 addresses.
- **Associated IP Address (1) ...Associated IP Address (10)**
  Shows the router IPv6 addresses monitored by this virtual router. If a router takes over the role of master, the routing function is taken over by this router for all these IPv6 addresses.

Addresses Configuration (IPv6)

Creating or changing the monitored IP addresses

On this page, you can create, modify or delete the IPv4 addresses to be monitored.

Note

This function is available only with layer 3.

Settings

- **Interface / VRID**
  Select the ID of the virtual router.
- **Associated IP Address**
  Enter the IPv6 address that the virtual router will monitor.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted
- **Associated IP Address**
  Shows the IPv6 addresses that the virtual router monitors.

Interface Tracking (IPv6)

Introduction

On this page, you configure the monitoring of interfaces.

When the link of a monitored interface changes from "up" to "down", the priority of the assigned VRRP interface is reduced. You configure the value by which the priority is reduced on the page "Layer 3 > VRRP/VRRPv3 > Configuration".

When the link of the interface changes back from "down" to "up", the original priority of the VRRP interface is restored.

Note

This function is available only with layer 3.

Description of the displayed values

The page contains the following boxes:

- **Interface**
  From the drop-down list, select the interface to be monitored.
- **Track ID**
  Enter a track ID.

- **Track ID**
  Select a track ID.
- **Track Interface Count**
  Enter how many monitored interfaces need to change to the "down" status, before the priority is changed.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Track ID**
  Shows the track ID.
- **Interface**
  Shows the interface that is being monitored.

OSPFv3

Configuration

Introduction

On this page, you configure the routing using OSPFv3.

Note

This function is available only with layer 3.

Settings

- **OSPFv3**
  Enable or disable routing using OSPFv3.
- **Router ID**
  Enter the name of one of the OSPFv3 interfaces. The router ID is specified in IPv4 format and must be unique in the network.
- **Border Router**
  Displays the status of the OSPFv3 router. If the local system is an active member in at least 2 areas, this is an area border router.
- **AS Border Router**
  Specify whether the router is an AS border router. An AS border router intercedes between multiple autonomous systems, for example if you have an additional RIP network. An AS border router is also necessary to add and to distribute static routes.
- **New LSA Received**
  Shows the number of received LSAs. Updates and its own LSAs are not counted.
- **New LSA Configured**
  Shows the number of different LSAs sent by this local system.

- **External LSA Maximum**
  To limit the number of entries of external LSAs in the database, enter the maximum number of external LSAs.
- **Exit Interval [s]**
  Enter the interval after which the OSPFv3 router once again attempts to come out of the overflow status. A 0 means that the OSPFv3 router attempts to exit the overflow status only following a restart.
- **Inbound Filter**
  Select a route map that filters inbound routes.
- **Redistribute Routes**
  Specify which known routes are distributed using OSPFv3. You make different decisions for the route types Connected, Static and RIPng.

  Note

  The options can only be enabled on an AS border router. Enabling the Default and Static options, in particular, can cause problems if they are enabled at too many points in the network, for example, forwarding loops.

- **Route Map**
  Select a route map that filters which routes are forwarded using RIPng.

Areas

Overview

An autonomous system can be divided into smaller areas.

On this page, you can view, create, modify or delete the areas of the router.

Note

This function is available only with layer 3.

Settings

- **Area ID**
  Enter the designation of the area. The database is synchronized for all routers of an area. The area identifier must be unique in the network.
  The area identifier is a 32-bit number with the following format: x.x.x.x where x = 0 ... 255
  The area identifier 0.0.0.0 is reserved for the backbone area and cannot be deleted.

This table contains the following columns:

- **Area ID**
  Shows the ID of the area.
- **Area Type**
  Specify the type of area.
  - Normal
  - Stub
  - NSSA
  - Backbone
- **Summary**
  Specify whether summary LSAs are generated for this area.
  - Summary: Summary LSAs are generated and sent to the area.
  - No Summary: Summary LSAs are not generated and sent to the area.
- **Metric**
  Displays the costs for the OSPFv3 interface.
- **Updates**
  Shows the number of recalculations of the IPv6 routing tables.
- **LSA Count**
  Shows the number of LSAs in the database.
- **Area BR**
  Shows the number of reachable area border routers (ABR) within this area.
- **AS BR**
  Shows the number of reachable autonomous system border routers (ASBR) in this area.

Area Range

Creating a new OSPFv3 area range

On this page, you can group up to 4 networks under one area ID. The method is used only with area border routers. This means that an area border router only advertises one route for grouped areas to the outside.

Note

This function is available only with layer 3.

Settings

- **Area ID**
  Select the ID of the area. You specify the ID in "Areas".
- **Prefix**
  Enter the IPv6 address of the network that will be grouped.
- **Prefix Length**
  Enter the number of bits belonging to the prefix that will be grouped.

This table contains the following columns:

- **Area ID**
  Shows the ID of the area.
- **Prefix**
  Shows the IPv6 address of the network that will be grouped.
- **Prefix Length**
  Enter the number of bits that will belong to the prefix.
- **Advertise**
  Enable this option to advertise the grouped network.

Interfaces

Overview

On this page, you can configure OSPFv3 interfaces.

Note

This function is available only with layer 3.

Settings

- **Interface**
  Select the IPv6 interface on which OSPFv3 will be enabled.
- **Area ID**
  Select the area ID of the area with which the OSPFv3 interface is connected.

  Note

  For the secondary interface select the same Area ID as for the corresponding primary interface.
  The information whether a primary or secondary interface is involved can be found in "Layer 3 (IPv6) > Subnets".

This table contains the following columns:

- **Interface**
  Shows the OSPFv3 interface.
- **Area ID**
  Select the area ID of the area with which the OSPFv3 interface is connected.
- **OSPF Status**
  Specify whether or not OSPFv3 is active on the interface.
  - Enabled: OSPFv3 is enabled on the interface.
  - Disabled: OSPFv3 is disabled on the interface.
- **Metric**
  Enter the costs for the OSPFv3 interface.
- **Priority**
  Enter the router priority. The priority is only relevant for selecting the designated router or designated border router. This parameter can be selected differently on routers within the same subnet.
- **Trans. Delay**
  Enter the required delay when sending a connection update.
- **Retrans. Interval**
  Enter the time after which an OSPFv3 packet is transferred again if no confirmation was received.
- **Hello Interval**
  Enter the interval between two Hello packets.
- **Dead Interval**
  Enter the interval after which the neighbor router is marked as "failed" if no more Hello packets are received from it during this time.

Virtual links

Overview

Due to the protocol, each area border router must have access to the backbone area. If a router is not connected directly to the backbone area, a virtual link to it is created.

Note

This function is available only with layer 3.

Note

Note that when creating a virtual link both the transit area and the backbone area must already be configured.

A virtual link must be configured identically at both ends.

Settings

- **Neighbor Router ID**
  Enter the ID of the neighbor router at the other end of the virtual connection.
- **Transit Area ID**
  Select the ID of the area that connects the two routers.

This table contains the following columns:

- **Transit Area ID**
  Shows the ID via which the two routers are connected.
- **Neighbor Router ID**
  Shows the ID of the neighbor router at the other end of the virtual connection.
- **Virt. Link Status**
  Specify the status of the virtual link. The following states are possible:
  - down: The virtual link is inactive.
  - point-to-point: The virtual link is active.
- **Trans. Delay**
  Enter the required delay when sending a connection update.
- **Retrans. Interval**
  Enter the time after which a packet is transferred again if no confirmation was received.
- **Hello Interval**
  Enter the interval between two Hello packets.
- **Dead Interval**
  Enter the interval after which the neighbor router counts as "failed" if no more Hello packets are received from it during this time.

RIPng

Configuration

On this page, you configure routing with RIPng.

Note

This function is available only with layer 3.

Settings

- **Inbound Filter**
  Select a route map that filters inbound routes.
- **Redistribute Routes**
  Specify which known routes are distributed using RIPng.
  The following types of route exist:
  - Connected
  - Static
  - OSPFv3
- **Route Map**
  Select a route map that filters which routes are forwarded using RIPng.

Interfaces

Overview

On this page, you specify the interfaces that support RIPng.

Note

This function is available only with layer 3.

Settings

This table contains the following columns:

- **Interface**
  Shows the available IPv6 interface.
- **Admin State**
  Enable or disable the RIPng routing for the interface.
- **Default Metric**
  Enter the costs for the RIPng interface.

Configuring security functions

Users

Local Users

Note

The page is only available if there is an online connection to the device.

Local users

On this page, you create local users with the corresponding rights.

When you create or delete a local user this change is also made automatically in the table "External User Accounts". If you want to make change explicitly for the internal or external user table, use the CLI commands.

Note

You can create up to 16 additional user accounts.

Note

The values displayed depend on the rights of the logged-in user.

Settings

- **User Account**
  Enter the name for the user. The name must meet the following conditions:
  - It must be unique.
  - It must be between 1 and 255 characters long.
  - The following characters must not be included: § ? " ; : < =

  Note

  User name cannot be changed

  After creating a user, the user name can no longer be modified.

  If a user name needs to be changed, the user must be deleted and a new user created.

  Deleting users

  The preset users as well as logged in users cannot be deleted or changed.

- **Password Policy**
  Shows which password policy is being used.
- **Password**
  Enter the password for the user.

  Note

  Password Policy: high

  Note the following password policies:
  - Password length: at least 8 characters, maximum 128 characters
  - At least 1 uppercase letter
  - At least 1 special character
  - At least 1 number

- **Password Confirmation**
  Enter the password again to confirm it.
- **Role**
  Select a role.
  You can choose between system-defined and self-defined roles, refer to the page "Security > Users > Roles.".

The table contains the following columns:

- **User Account**
  Shows the user name.
- **Role**
  Shows the role of the user.
- **Description**
  Displays a description of the user account. The description text can be up to 100 characters long.

Roles

Note

The page is only available if there is an online connection to the device.

Roles

On this page, you create roles that are valid locally on the device.

Note

The values displayed depend on the rights of the logged-in user.

Note

You can create up to 28 additional roles.

Settings

- **Role Name**
  Enter the name for the role. The name must meet the following conditions:
  - It must be unique.
  - It must be between 1 and 64 characters long.

  Note

  Role name cannot be changed

  After creating a role, the name of the role can no longer be changed.

  If a name of a role needs to be changed, the role must be deleted and a new role created.

  Deleting roles

  Predefined roles and assigned roles cannot be deleted or modified.

The table contains the following columns:

- **Role**
  Shows the name of the role.
- **Function Right**
  Select the function rights of the role.
  - 1
    Users with this role can read device parameters but cannot change them.
  - 15
    Users with this role can both read and change device parameters.

  Note

  Function right cannot be changed

  If you have assigned a role, you can no longer change the function right of the role.

  If you want to change the function right of a role, follow the steps outlined below:

  1. Delete all assigned users.
  2. Change the function right of the role:
  3. Assign the role again.

- **Description**
  Enter a description for the role. With predefined roles a description is displayed. The description text can be up to 100 characters long.

Groups

Note

The page is only available if there is an online connection to the device.

User Groups

On this page you link a group with a role.

In this example the group "Administrators" is linked to the "admin"role: The group is defined on a RADIUS server. The role is defined locally on the device. When a RADIUS server authenticates a user and assigns the user to the "Administrators" group, this user is given rights of the "admin" role.

Note

You can create up to 32 groups.

Settings

- **Group Name**
  Enter the name of the group. The name must match the group on the RADIUS server. The name must meet the following conditions:
  - It must be unique.
  - It must be between 1 and 64 characters long.

The table contains the following columns:

- **Group**
  Shows the name of the group.
- **Role**
  Select a role. Users who are authenticated with the linked group on the RADIUS server receive the rights of this role locally on the device.
  You can choose between system-defined and self-defined roles, refer to the page "Security > Users > Roles.".
- **Description**
  Enter a description for the link of the group.to a role.

Passwords

Passwords

Note

The page is only available if there is an online connection to the device.

On this page, you can change passwords. If you are logged on with the "admin" role, you can change the passwords for all user accounts. If you are logged on with the "user" role, you can only change your own password.

Note

If you log on the first time or log on after a "Reset to factory defaults and restart", you will be prompted to change the password.

Note

If you are logged in via a RADIUS server, you cannot change any local device passwords.

Note

Changing the password in "Trial" configuration mode

Even if you change the password in "Trial" configuration mode, this change is saved immediately.

Note

The factory settings for the passwords when the devices ship are as follows:

- admin: admin

When you log in for the first time or following a "Restore Factory Defaults and Restart", with the preset user "admin" you will be prompted to change the password.

Settings

- **Current User**
  Shows the user that is currently logged in.
- **Current User Password**
  Enter the password for the currently logged in user.
- **User Account**
  Select the user whose password you want to change.
- **Password Policy**
  Shows which password policy is being used when assigning new passwords.
  - High
    Password length: at least 8 characters, maximum 128 characters
    at least 1 uppercase letter
    at least 1 special character
    at least 1 number
  - Low
    Password length: at least 6 characters, maximum 128 characters
- **New Password**
  Enter the new password again to confirm it.
- **Password Confirmation**
  Enter the new password again to confirm it.

Options

On this page you specify which password policy will be used when assigning new passwords.

Description

- **Password Policy**
  Shows which password policy is currently being used
- **New Password Policy**
  Select the required setting from the drop-down list.
  - High
    Password length: at least 8 characters, maximum 128 characters
    at least 1 uppercase letter
    at least 1 special character
    at least 1 number
  - Low
    Password length: at least 6 characters, maximum 128 characters

AAA

General

The term "AAA" used in the menu stands for "Authentication, Authorization, Accounting" and is used to identify and allow network nodes, to provide them with the appropriate services and to determine the amount of use.

On this page, you configure the login.

Settings

Note

To be able to use the login authentication "RADIUS", "Local and RADIUS" or "RADIUS and fallback Local" a RADIUS server must be stored and configured for user authentication.

- **Login Authentication**
  Specify how the login is made:
  - Local
    The authentication must be made locally on the device.
  - RADIUS
    The authentication must be handled via a RADIUS server.
  - Local and RADIUS
    The authentication is possible both with the users that exist on the device (user name and password) and via a RADIUS server.
    The user is first searched for in the local database. If the user does not exist there, a RADIUS query is sent.
  - RADIUS and fallback Local
    The authentication must be handled via a RADIUS server.
    A local authentication is performed only when the RADIUS server cannot be reached in the network.

RADIUS client

The term "AAA" used in the menu stands for "Authentication, Authorization, Accounting" and is used to identify and allow network nodes, to provide them with the appropriate services and to determine the amount of use.

Authentication over an external server

The concept of RADIUS is based on an external authentication server.

Each row of the table contains access data for one server. In the search order, the primary server is queried first. If the primary server cannot be reached, secondary servers are queried in the order in which they are entered.

If no server responds, there is no authentication.

Settings

The page contains the following boxes:

- **RADIUS Authorization Mode**
  For the login authentication, the RADIUS authorization mode specifies how the rights are assigned to the user with a successful authentication (Page 1272).
  - conventional
    In this mode the user is logged in with administrator rights if the server returns the value "Administrative User" to the device for the attribute "Service Type". In all other cases the user is logged in with read rights.
  - Vendor specific
    In this mode the assignment of rights depends on whether and which group the server returns for the user and whether or not there is an entry for the user in the table "External User Accounts".

The table has the following columns:

- **Auth. Server Type**
  Select which authentication method the server will be used for.
  - Login
    The server is used only for the login authentication.
  - 802.1X
    The server is used only for the 802.1X authentication.
  - Login & 802.1X
    The server is used for both authentication procedures.
- **RADIUS Server Address**
  Enter the IPv4 address or the FQDN (Fully Qualified Domain Name) of the RADIUS server.
- **Server Port**
  Here, enter the input port on the RADIUS server. As default, input port 1812 is set.
- **Shared Secret**
  Enter your access ID.
- **Confirm Shared Secret**
  Enter your access ID again as confirmation.
- **Max. Retrans**
  Enter the maximum number of query attempts before another configured RADIUS server is queried or the login counts as having failed. As default, 3 is set.
- **Primary Server**
  Specify whether or not this server is the primary server. You can select one of the options "yes" or "no".
- **Test**
  With this button, you can test whether or not the specified RADIUS server is available. The test is performed once and not repeated cyclically.
- **Test Result**
  Shows whether or not the RADIUS server is available:
  - Failed, no test packet sent
    The IP address is not reachable.
    The IP address is reachable, the RADIUS server is, however, not running.
  - Reachable, key not accepted
    The IP address is reachable, the RADIUS server does not, however accept the shared secret.
  - Reachable, key accepted
    The IP address is reachable, the RADIUS server accepts the specified shared secret.

  The test result is not automatically updated. To delete the test result click the "Refresh" button.

802.1x-Authenticator

Setting up network access

An end device can only access the network after the device has verified the login data of the device with the authentication server. Both the end device and the authentication server must support the EAP protocol (Extensive Authentication Protocol).

Enabling authentication for individual ports

By selecting the check box, you specify whether or not network access protection according to IEEE 802.1x is enabled on this port.

Settings

- MAC Authentication
  Enable or disable MAC authentication for the device.
- Guest VLAN
  Enable or disable the "Guest VLAN" function for the device.

Table 1 has the following columns:

- 1st column
  Shows that the settings are valid for all ports of table 2.
- 802.1X Auth. Control
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- 802.1x Re-authentication
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- MAC Authentication
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- Adopt RADIUS VLAN Assignment
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- MAC Auth. Max Permitted Addresses
  Enter how many end devices are allowed to be connected to the port at the same time. If "No Change" is entered, the entry in table 2 remains unchanged
- Guest VLAN
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- Guest VLAN ID
  Enter the VLAN ID of the port. If "No Change" is entered, the entry in table 2 remains unchanged

- **Guest VLAN max. Permitted Addresses**
  Enter how many end devices are allowed in the guest VLAN at the same time.
- **Transfer to table**
  If you click the button, the settings are adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows all available ports.
- **802.1X Auth. Control**
  Specify the authentication of the port:
  - Force Unauthorized
    Data traffic via the port is blocked.
  - Force Authorized
    Data traffic via the port is allowed without any restrictions.
    Default setting
  - Auto
    End devices are authenticated on the port with the "802.1x" method.
    The data traffic via the port is permitted or blocked depending on the authentication result.
- **802.1x Re-authentication**
  Enable this option if you want reauthentication of an already authenticated end device to be repeated cyclically.
- **MAC Authentication**
  Enable this option if you want end devices to be authenticated with the "MAC Authentication" method.

  Note

  At the port level, you cannot enable "MAC Authentication" and "Guest VLAN" at the same time.

  If "Auto is configured for "802.1x Auth. Control and the " MAC Authentication is enabled, the timeout for the "802.1X procedure is 5 seconds. If manual input is necessary at a port for the authentication with the 802.1X" procedure, the 5 seconds are not adequate. To be able to run authentication using "802.1X", disable the MAC authentication on this port.
- **Adopt RADIUS VLAN Assignment**
  The RADIUS server informs the IE switch of the VLAN to which the port will belong. Enable this option if you want the information of the server to be taken into account.
  The port can only e assigned to the VLAN, if the VLAN has been created on the device. Otherwise Authentication (Page 1272) is rejected.
- **MAC Auth. Max Permitted Addresses**
  Enter how many end devices are allowed to be connected to the port at the same time.

- **Guest VLAN**
  Enable this option if you want the end device to be permitted in the guest VLAN if authentication fails.
  The port can only e assigned to the VLAN, if the VLAN has been created on the device. Otherwise Authentication (Page 1272) is rejected.
  This function is also known as "Authentication failed VLAN".

  Note

  At the port level, you cannot enable "MAC Authentication" and "Guest VLAN" at the same time.

- **Guest VLAN ID**
  Enter the VLAN ID of the port.
- **Guest VLAN max. Permitted Addresses**
  Enter how many end devices are allowed in the guest VLAN at the same time.
- **MAC Auth. Currently Permitted Addresses**
  Shows the number of currently connected end devices.
- **802.1X Auth.-Status**
  Shows the status of the authentication of the port:
  - Authorize
  - Do not authorize
- **MAC Auth. Currently Blocked Addresses**
  Shows the number of currently blocked end devices.
- **Guest VLAN Currently Permitted Addresses**
  Shows how many end devices are currently allowed in the guest VLAN.

MAC ACL

Rule configuration

On this page, you specify the access rules for the MAC-based Access Control List. Using the MAC-based ACL, you can specify whether frames of certain MAC addresses are forwarded or discarded.

Settings

- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Source MAC**
  Enter the unicast MAC address of the source.
  Destination MAC Address
  Enter the unicast MAC address of the destination.

- **Action**
  Select the action. The following is possible:
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.
- **Ingress Interfaces**
  Shows a list of all interfaces to which this rule applies.
- **Egress Interfaces**
  Shows a list of all interfaces to which this rule applies.

Note

Entering the MAC addresses

If you enter the address "00:00:00:00:00:00" for the source and/or destination MAC address, the rule created in this way applies to all source or destination MAC addresses.

Ingress Rules

On this page, you specify the ACL rule according to which incoming frames are handled by the port.

Settings

- **Interface**
  Select the required interface.
- **Add Rule**
  Select the ACL rule to be assigned to the interface. You specify the ACL rule on the "Rules Configuration" page.
- **Add**
  To permanently assign the ACL rule to the interface, click the "Add" button. The configuration is shown in the table.
- **Remove Rule**
  Select the ACL rule to be deleted.
- **Remove**
  To remove the ACL rule from the interface, click the "Remove" button.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Source MAC**
  Shows the unicast MAC address of the source.

- **Destination MAC Address**
  Shows the unicast MAC address of the destination.
- **Action**
  Shows the action:
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.

Egress Rules

On this page, you specify the ACL rule according to which outgoing frames are handled by the port.

Settings

- **Interface**
  Select the required interface.
- **Add Rule**
  Select the ACL rule to be assigned to the interface. You specify the ACL rule on the "Rules Configuration" page.
- **Add**
  To permanently assign the ACL rule to the interface, click the "Add" button. The configuration is shown in the table.
- **Remove Rule**
  Select the ACL rule to be deleted.
- **Remove**
  To remove the ACL rule from the interface, click the "Remove" button.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Source MAC**
  Shows the unicast MAC address of the source.
- **Destination MAC Address**
  Shows the unicast MAC address of the destination.
- **Action**
  Shows the action
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.

IP ACL

Rule configuration

On this page, you specify the rules for the IP-based ACL.

Settings

The table has the following columns:

- Rule Number
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- Source IP
  Enter the IPv4 address of the source.
- Source Subnet Mask
  Enter the subnet mask in which the source is located.
- Dest. IP
  Enter the IP address of the destination.
- Dest. Subnet Mask
  Enter the subnet mask in which the destination is located.
- Action
  Select whether the frame is forwarded or rejected when it corresponds to the ACL rule.
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.
- Ingress Interfaces
  Shows a list of all interfaces to which this rule applies.
- Egress Interfaces
  Shows a list of all interfaces to which this rule applies.

Protocol configuration

On this page, you specify the rules for protocols.

Settings

The table has the following columns:

- Rule Number
  Shows the number of the protocol rule. When you create a rule, a new row with a unique number is created.
- Protocol
  Select the protocol for which this rule is valid.

- **Protocol Number**
  Enter a protocol number to define further protocols.
  This box can only be edited if you have set "Other Protocol" for the protocol.
- **Source Port Min.**
  Enter the lowest possible port number of the source port.
  This box can only be edited if you have set "TCP" or "UDP" for the protocol.
- **Source Port Max.**
  Enter the highest possible port number of the source port.
  This box can only be edited if you have set "TCP" or "UDP" for the protocol.
- **Dest. Port Min.**
  Enter the lowest possible port number of the destination port.
  This box can only be edited if you have set "TCP" or "UDP" for the protocol.
- **Dest. Port Max.**
  Enter the highest possible port number of the destination port.
  This box can only be edited if you have set "TCP" or "UDP" for the protocol.
- **Message Type**
  Enter a message type to decide the format of the message.
  This box can only be edited if you have set "ICMP" for the protocol.
- **Message Code**
  Enter a message code to specify the function of the message.
  This box can only be edited if you have set "ICMP" for the protocol.
- **DSCP**
  Enter a value for classifying the priority.
  This box cannot be edited if you have set "ICMP" for the protocol.

Ingress Rules

On this page, you specify the ACL rule according to which incoming frames are handled by the port or VLAN.

Settings

- **Ports**
  Select the required port: a port or a VLAN interface.
  To select a VLAN interface, an IP interface must be configured.

  Note

  If you use a VLAN interface, the ACL rule applies to all ports that belong to the VLAN.

- **Add Rule**
  Select the ACL rule to be assigned to the port. You specify the ACL rule on the "Rules Configuration" page.
- **Add**
  To permanently assign the ACL rule to the port, click the "Add" button. The configuration is shown in the table.

- **Remove Rule**
  Select the ACL rule to be deleted.
- **Remove**
  To remove the ACL rule from the port, click the "Remove" button.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Protocol**
  Shows the protocol for which this rule is valid.
- **Protocol Number**
  Shows the protocol number.
- **Source IP**
  Shows the IP address of the source.
- **Source Subnet Mask**
  Shows the subnet mask in which the source is located.
- **Dest. IP**
  Shows the IP address of the destination.
- **Dest. Subnet Mask**
  Shows the subnet mask of the destination.
- **Action**
  Select whether the frame is forwarded or rejected when it corresponds to the ACL rule.
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.
- **Source Port Min.**
  Shows the lowest possible port number of the source port.
- **Source Port Max.**
  Shows the highest possible port number of the source port.
- **Dest. Port Min.**
  Shows the lowest possible port number of the destination port.
- **Dest. Port Max.**
  Shows the highest possible port number of the destination port.
- **Message Type**
  Shows a message type to decide the format of the message.
- **Message Code**
  Shows a message code to specify the function of the message.
- **DSCP**
  Shows the value for classifying the priority.

Egress Rules

On this page, you specify the ACL rule according to which outgoing frames are handled by the port.

Settings

- **Ports**
  Select the required port: a port or a VLAN interface.
  To select a VLAN interface, an IP interface must be configured.

  Note

  If you use a VLAN interface, the ACL rule applies to all ports that belong to the VLAN.

- **Add Rule**
  Select the ACL rule to be assigned to the port. You specify the ACL rule on the "Rules Configuration" page.
- **Add**
  To permanently assign the ACL rule to the port, click the "Add" button. The configuration is shown in the table.
- **Remove Rule**
  Select the ACL rule to be deleted.
- **Remove**
  To remove the ACL rule from the port, click the "Remove" button.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Protocol**
  Shows the protocol for which this rule is valid.
- **Protocol Number**
  Shows the protocol number.
- **Source IP**
  Shows the IP address of the source.
- **Source Subnet Mask**
  Shows the subnet mask in which the source is located.
- **Dest. IP**
  Shows the IP address of the destination.
- **Dest. Subnet Mask**
  Shows the subnet mask of the source.

- **Action**
  Select whether the frame is forwarded or rejected when it corresponds to the ACL rule.
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.
- **Source Port Min.**
  Shows the lowest possible port number of the source port.
- **Source Port Max.**
  Shows the highest possible port number of the source port.
- **Dest. Port Min.**
  Shows the lowest possible port number of the destination port.
- **Dest. Port Max.**
  Shows the highest possible port number of the destination port.
- **Message Type**
  Shows a message type to decide the format of the message.
- **Message Code**
  Shows a message code to specify the function of the message.
- **DSCP**
  Shows the value for classifying the priority.

Management ACL

On this page, you can increase the security of your device. To specify which station with which IPv4 address is allowed to access your device, configure the IPv4 address or an entire address range.

You can select the protocols and the ports with which the station is allowed to access the device. You define the VLAN in which the station may be located. This ensures that only certain stations within a VLAN have access to the device.

Note

If you enable this function, note the following

A bad configuration on the "Management Access Control List" page can result in you being unable to access the device. You should therefore configure an access rule that allows access to the management before you enable the function.

Settings

- **Management ACL**
  Enable or disable the function. As default, the function is disabled.

  Note

  If the function is disabled, there is unrestricted access to the management of the IE switch. The configured access rules are only taken into account when the function is enabled.

- **IP Address**
  Enter the IPv4 address or the network address for which the rule will apply. If you use the IPv4 address 0.0.0.0, the settings apply to all IPv4 addresses.
- **Subnet Mask**
  Enter the subnet mask. The subnet mask 255.255.255.255 is for a specific IP address. If you want to allow a subnet, for example a C subnet, enter 255.255.255.0. The subnet mask 0.0.0.0 applies to all subnets.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **IP Address**
  Shows the IPv4 address.
- **Subnet Mask**
  Shows the subnet mask.
- **VLANs Allowed**
  Enter the number of the VLAN in which the device is located. The station can only access the device if it is located in this configured VLAN. If this input box remains empty, there is no restriction relating to the VLANs.
- **SNMP**
  Specify whether the station (or the IPv4 address) can access the device using the SNMP protocol.
- **TELNET**
  Specify whether the station (or the IPv4 address) can access the device using the TELNET protocol.
- **HTTP**
  Specify whether the station (or the IPv4 address) can access the device using the HTTP protocol.
- **HTTPS**
  Specify whether the station (or the IPv4 address) can access the device using the HTTPS protocol.
- **SSH**
  Specify whether the station (or the IPv4 address) can access the device using the SSH protocol.
- **Px.y**
  Specify whether the station (or the IPv4 address) can access this device via this port (slot.port).

Configuring SCALANCE W

IP address assignment

Configuration options

When shipped and after restoring to the factory settings, the device does not have an IP address.

The following options are available to assign an IP address to the device:

- DHCP (default setting)
- Primary Setup Tool
  For further information, refer to the configuration manual "Primary Setup Tool". You will find the configuration manual at Siemens Industry Automation and Drives Service & Support on the Internet under the entry ID 19440762 (http://support.automation.siemens.com/WW/view/en/19440762).
- STEP 7
- CLI via the serial interface
  For further information, refer to the configuration manuals for the Command Line Interface". You will find the configuration manuals at Siemens Industry Automation and Drives Service & Support on the Internet under the following entry IDs.
  - SCALANCE W780/W740 Command Line Interface - entry ID 62515451 (http://support.automation.siemens.com/WW/view/en/62515451)
  - SCALANCE W770/W730 Command Line Interface - entry ID 89534887 (http://support.automation.siemens.com/WW/view/en/89534887)
  - SCALANCE W760/W720 Command Line Interface - entry ID 89534699 (http://support.automation.siemens.com/WW/view/en/89534699)

Requirement

- "Set IP address on device" is enabled in the properties of the device. You will find more detailed information in "Addressing PROFINET devices".

Interfaces and system functions

SCALANCE W780 / W740

Availability of the interfaces

The following table shows the availability of the physical and logical interfaces. Note that in this table all interfaces are listed. Depending on the system function, some interfaces are not available. On the WBM pages you can only select the available interfaces.

We reserve the right to make technical changes.

	Client device **W748-1 M12** **W748-1 RJ-45**	**Access points** **W786-1 RJ-45** **W788-1 M12** **W788-1 RJ-45**	**Access points** **W786-2 RJ-45** **W786-2IA RJ-45** **W786-2 SFP** **W788-2 M12** **W788-2 M12 EEC** **W788-1 RJ-45** **W788-2 RJ-45**
Wireless interface (WLAN)	WLAN 1	WLAN 1	• WLAN 1 (in client mode only one WLAN interface is available) • WLAN 2
IP interface: LAN interface VLAN	 P1 ManagementVLAN	 P1 ManagementVLAN	 P1 ManagementV-LAN
VAP interface [1)]	-	VAP 1.Y Y = 1 ... 8	VAP X.Y X = 1 ... 2 Y = 1 ... 8
WDS interface [1)]	_	WDS 1.Y Y = 1 ... 8	WDS X.Y X = 1 ... 2 Y = 1 ... 8
VLAN	24	24	24

[1)] only in access point mode

Availability of the system functions

The following table shows the availability of the system functions on the devices. Note that all functions are described in this configuration manual and in the online help. Depending on the mode and the KEY-PLUG, some functions are not available.

We reserve the right to make technical changes.

			Access point mode	Access points in client mode. Client device
Information	**Security**	Inter AP blocking	✓ W780 iFeatures (MLFB 6GK5 907-8PA00) W700 Security (MLFB 6GK5907-0PA00)	-
	WLAN	AP overview	✓	-
		Client list	✓	-
		WDS list	✓	-
		AP overlap	✓	-
		Force roaming	✓	-
		Client overview	-	✓
		Available APs	-	✓
		IP assignment	-	✓
		Background noise	✓	✓
	WLAN statistics	Error	✓	✓
		Management sent	✓	✓
		Management received	✓	✓
		Data sent	✓	✓
		Data received	✓	✓
	WLAN iFeatures	iREF client list	✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-
		iREF WDS list	✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-
		AeroScout	✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-
System		PROFINET	✓	-✓
		EtherNet/IP	✓	✓

			Access point mode	**Access points in client mode. Client device**
Interfaces	**WLAN**	Basic	✓	-✓
		Expansions	✓	✓
		Antennas	✓	✓
		Permitted channels	✓	✓
		802.11n	✓	✓
		AP	✓	-
		AP WDS	✓	-
		AP 802.11a/b/g data rates	✓	-
		AP 802.11n data rates	✓	-
		Client 802.11a/b/g data rates	-	✓
		Client 802.11n data rates	-	✓
		Force roaming	✓	-
		Signal recorder	-	✓
		Spectrum analyzer	✓	-
Layer 3	**NAT**	Basic	-	✓
		NAPT	-	✓

			Access point mode	Access points in client mode. Client device
Security	**WLAN**	Basic	✓	✓
		AP communication	✓	-
		AP RADIUS authenticator	✓	-
		Client RADIUS supplicant	-	✓
		Key	✓	✓
	Inter AP Blocking	Basic	✓ W780 iFeatures (MLFB 6GK5 907-8PA00) W700 Security (MLFB 6GK5907-0PA00)	-
		Allowed IP addresses	✓ W780 iFeatures (MLFB 6GK5 907-8PA00) W700 Security (MLFB 6GK5907-0PA00)	-

			Access point mode	Access points in client mode. Client device
iFeatures	iPCF		✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	✓ Access point in client mode: W780 iFeatures (MLFB 6GK5 907-8PA00 Client: W740 iFeatures (MLFB 6GK5 907-4PA00)
	iPCF-MC		✓ Only dual APs W780 iFeatures (MLFB 6GK5 907-8PA00)	✓ Access point in client mode: W780 iFeatures (MLFB 6GK5 907-8PA00 Client: W740 iFeatures (MLFB 6GK5 907-4PA00)
	iPRP		✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	✓ Access point in client mode: W780 iFeatures (MLFB 6GK5 907-8PA00 Client: W740 iFeatures (MLFB 6GK5 907-4PA00)
	iREF		✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-
	AeroScout		✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-

SCALANCE W770 / W730

Availability of the interfaces

The following table shows the availability of the physical and logical interfaces. Note that in this table all interfaces are listed. Depending on the system function, some interfaces are not available. On the WBM pages you can only select the available interfaces.

We reserve the right to make technical changes.

	Client device SCALANCE W734-1 RJ-45	Access points W774-1 RJ-45
Wireless interface (WLAN)	WLAN 1	WLAN 1
IP interface: LAN interface VLAN	 P1 ManagementVLAN	 P1 ManagementVLAN
VAP interface [1)]	-	VAP 1.Y Y = 1 ... 8

	Client device **SCALANCE W734-1 RJ-45**	**Access points** **W774-1 RJ-45**
WDS interface [1]	_	WDS 1.Y Y = 1 ... 8
VLAN	16	16

[1] only in access point mode

Availability of the system functions

The following table shows the availability of the system functions on the devices. Note that all functions are described in this configuration manual and in the online help. Depending on the mode and the KEY-PLUG, some functions are not available.

We reserve the right to make technical changes.

			Access point mode	**Access points in client mode** **Client devices**
Information	**Security**	Inter AP blocking	✓ W780 iFeatures (MLFB 6GK5 907-8PA00) W700 Security (MLFB 6GK5907-0PA00)	-
	WLAN	Overview AP	✓	-
		Client List	✓	-
		WDS List	✓	-
		Overlap AP	✓	-
		Force Roaming	✓	-
		Overview Client	-	✓
		Available AP	-	✓
		IP Mapping	-	✓
	WLAN Statistics	Faults	✓	✓
		Management Sent	✓	✓
		Management Received	✓	✓
		Data Sent	✓	✓
		Data Received	✓	✓
	WLAN iFeatures	iREF Client List	✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-
		iREF WDS List	✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-
		AeroScout	✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-
System		PROFINET	✓	-✓
		EtherNet/IP	✓	✓

			Access point mode	Access points in client mode Client devices
Interfaces	**WLAN**	Basic	✓	-✓
		Expansions	✓	✓
		Antennas	✓	✓
		Allowed Channels	✓	✓
		802.11n	✓	✓
		AP	✓	-
		AP WDS	✓	-
		AP 802.11a/b/g Rates	✓	-
		AP 802.11n Rates	✓	-
		Client 802.11a/b/g Rates	-	✓
		Client 802.11n Rates	-	✓
		Force Roaming	✓	-
		Signal recorder	-	✓
		Spectrum Analyzer	✓	-
Layer 3	**NAT**	Basic	-	✓
		NAPT	-	✓
Security	**WLAN**	Basic	✓	✓
		AP Communication	✓	-
		AP RADIUS Authenticator	✓	-
		Client RADIUS Supplicant	-	✓
		Keys	✓	✓
	Inter AP Blocking	Basic	✓ W780 iFeatures (MLFB 6GK5 907-8PA00) W700 Security (MLFB 6GK5907-0PA00)	-
		Allowed Addresses	✓ W780 iFeatures (MLFB 6GK5 907-8PA00) W700 Security (MLFB 6GK5907-0PA00)	-

			Access point mode	Access points in client mode Client devices
iFeatures	iPCF	iPCFv1	✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	✓ Access point in client mode: W780 iFeatures (MLFB 6GK5 907-8PA00 Client: W740 iFeatures (MLFB 6GK5 907-4PA00)
	iPCF-MC		✓ Only dual APs W780 iFeatures (MLFB 6GK5 907-8PA00)	✓ Access point in client mode: W780 iFeatures (MLFB 6GK5 907-8PA00 Client: W740 iFeatures (MLFB 6GK5 907-4PA00)
	iPRP		✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	✓ Access point in client mode: W780 iFeatures (MLFB 6GK5 907-8PA00 Client: W740 iFeatures (MLFB 6GK5 907-4PA00)
	iREF		✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-
	AeroScout		✓ W780 iFeatures (MLFB 6GK5 907-8PA00)	-

SCALANCE W760 / W720

Availability of the interfaces

The following table shows the availability of the physical and logical interfaces. Note that in this table all interfaces are listed. Depending on the system function, some interfaces are not available. On the WBM pages you can only select the available interfaces.

We reserve the right to make technical changes.

	Client device W722-1 RJ45 W721-1 RJ45	Access points W761-1 RJ45
Wireless interface (WLAN)	WLAN 1	WLAN 1
IP interface: LAN interface VLAN	 P1 ManagementVLAN	 P1 ManagementVLAN
VAP interface [1)]	-	VAP 1.1

	Client device **W722-1 RJ45** **W721-1 RJ45**	**Access points** **W761-1 RJ45**
WDS interface [1]	_	WDS 1.1
VLAN	3	3

[1] only in access point mode

Availability of the system functions

The following table shows the availability of the system functions on the devices. Note that all functions are described in this configuration manual and in the online help. Depending on the mode and the KEY-PLUG, some functions are not available.

We reserve the right to make technical changes.

			Access point mode	**Access points in client mode** **Client devices**
Information	**WLAN**	Overview AP	✓	-
		Client List	✓	-
		WDS List	✓	-
		Overlap AP	✓	-
		Force Roaming	✓	-
		Overview Client	-	✓
		Available AP	-	✓
		IP Mapping	-	✓
	WLAN Statistics	Faults	✓	✓
		Management Sent	✓	✓
		Management Received	✓	✓
		Data Sent	✓	✓
		Data Received	✓	✓
System		PROFINET	✓	✓
		EtherNet/IP	✓	✓

			Access point mode	Access points in client mode Client devices
Interfaces	**WLAN**	Basic	✓	✓
		Advanced	✓	✓
		Antennas	✓	✓
		Allowed Channels	✓	✓
		802.11n	✓	✓
		AP	✓	-
		AP WDS	✓	-
		AP 802.11a/ b/g Rates	✓	-
		AP 802.11n Rates	✓	-
		Force Roaming	✓	-
		Signal re-corder	-	✓
		Spectrum Analyzer	✓	-
Layer 3	**NAT**	Basic	-	✓
		NAPT	-	✓
Security	**WLAN**	Basic	✓	✓
		AP Commu-nication	✓	-
		AP RADI-US Authen-ticator	✓	-
		Client RA-DIUS Sup-plicant	-	✓
		Keys	✓	✓
iFeatures	**iPCF**		-	✓ Only W722-1 RJ45 This can be integrated in a WLAN with iPCF, iPCF-MC.
	iPCF-MC		-	✓ Only W722-1 RJ45 This can be integrated in a WLAN with iPCF, iPCF-MC.
	iPRP		-	✓ Only W722-1 RJ-45

IPv6 support

Support of IPv6

The following system functions do not support IPv6 addresses:

- Inter AP blocking
- Force roaming
- IP ACL
- Management ACL

Editing properties and parameters

Editing options

You have the following options for editing properties and parameters:

- Hardware and network editor
  Once you have inserted the network component, you can edit the properties and parameters, for example the device name. You will find more detailed information in "Hardware and network editor".
- Web Based Management (WBM)
  The parameters and properties are accessible using the supplied HTML pages (WBM pages). Each WBM page has its own help page that describes the properties and parameters. For further information, refer to the configuration manuals for Web Based Management.
  - SCALANCE W780/W740 Web Based Management - entry ID: 62382125 (http://support.automation.siemens.com/WW/view/en/62382125)
  - SCALANCE W770/W730 Web Based Management - entry ID: 89535155 (http://support.automation.siemens.com/WW/view/en/89535155)
  - SCALANCE W760/W720 Web Based Management - entry ID: 89535534 (http://support.automation.siemens.com/WW/view/en/89535534)
- Command Line Interface
  All the configuration settings for the device can be made using the CLI. The CLI provides the same options as Web Based Management (WBM). For further information, refer to the configuration manuals for the Command Line Interface.
  - SCALANCE W780/W740 Command Line Interface - entry ID: 62515451 (http://support.automation.siemens.com/WW/view/en/62515451)
  - SCALANCE W770/W730 Command Line Interface - entry ID: 89534887 (http://support.automation.siemens.com/WW/view/en/89534887)
  - SCALANCE W760/W720 Command Line Interface - entry ID: 89534699 (http://support.automation.siemens.com/WW/view/en/89534699)

Availability

The availability of the settings depends

- on the device mode
- on the configuration mode
- Device

Device mode

With SCALANCE W the following device modes are available:

- Access point mode
- Client mode

Configuration mode

The following configuration modes are available:

- Offline configuration
  You can configure a PC station the first time offline. In this mode only the settings that require no connection to the device are available.
- Online configuration mode
  If there is a connection to the device and Web Based Management has started, there are additional pages available in the work area. In the online help, these pages include the note "This page is only available if there is an online connection to the device". With some settings, the additional information "only available online" is included.
  Web Based Management can be started in the following ways:
  - Select "Management" in the navigation panel and click on "Web Based Management".
  - Start the Web browser and enter the IP address or the URL of the device in the address box.

Device

Some of the settings depend on the device. The names used in the online help have the following meaning:

SCALANCE W740	The chapter described / the section described / the setting described is only relevant for the following devices: SCALANCE W748-1 RJ-45 SCALANCE W748-1 M12
SCALANCE W780	The chapter described / the section described / the setting described is only relevant for the following devices: SCALANCE W788-1 M12 SCALANCE W788-2 M12 SCALANCE W788-2 M12 EEC SCALANCE W788-1 RJ-45 SCALANCE W788-2 RJ-45 SCALANCE W786-1 RJ-45 SCALANCE W786-2 RJ-45 SCALANCE W786-2IA RJ-45 SCALANCE W786-2 SFP

SCALANCE W770	The chapter described / the section described / the setting described is only relevant for the following devices: SCALANCE W774-1 RJ-45
SCALANCE W730	The chapter described / the section described / the setting described is only relevant for the following devices: SCALANCE W734-1 RJ-45
SCALANCE W760	The chapter described / the section described / the setting described is only relevant for the following devices: SCALANCE W761-1 RJ-45
SCALANCE W720	The chapter described / the section described / the setting described is only relevant for the following devices: SCALANCE W721-1 RJ-45 SCALANCE W722-1 RJ-45
Access points	SCALANCE W788-1 M12 SCALANCE W788-2 M12 SCALANCE W788-2 M12 EEC SCALANCE W788-1 RJ-45 SCALANCE W788-2 RJ-45 SCALANCE W786-1 RJ-45 SCALANCE W786-2 RJ-45 SCALANCE W786-2IA RJ-45 SCALANCE W786-2 SFP SCALANCE W774-1 RJ-45 SCALANCE W761-1 RJ-45
Client	SCALANCE W748-1 RJ-45 SCALANCE W748-1 M12 SCALANCE W734-1 RJ-45 SCALANCE W721-1 RJ-45 SCALANCE W722-1 RJ-45

Creating and deleting an entry

As an example, an entry will be created and deleted on the Syslog client. The procedure is valid on all pages.

Creating a new entry

1. Select the device in the network or device view.
2. Open the properties of the device in the Inspector window.
3. In the Inspector window go to System > Syslog client.
4. Enter the IP address for the Syslog server.
5. Click in the table. Select the entry "New entry" in the shortcut menu. A new entry is created in the table.

Deleting an entry

1. Select the required entry in the table.
2. Select the "Delete" entry in the shortcut menu.

Buttons you require often

- **Updating the display with "Refresh"**
  Pages that display current parameters have a "Refresh" button at the lower edge of the page. Click this button to request up-to-date information from the device for the current page.
- **Save the settings for all ports with "Transfer to table"**
  Pages on which you can configure several ports have 2 tables. In the first table, you can make the settings for all ports and transfer these to the second table. In the last column of the first table, there is the "Transfer to table" button. Click this button to save the settings you have made for all ports.

Configuration limits

SCALANCE W780 / W740

Configuration limits of the device

The following table lists the configuration limits for Web Based Management and the Command Line Interface of the device.

Depending on your device, some functions are not available.

	Configurable function	Maximum number
System	Syslog server	3
	DNS server	3
	SMTP server	2
	SNMPv1 trap recipient	10
	SNTP server	2
	NTP server	1
	DHCP pools	1
	IPv4 addresses managed by the DHCP server (dynamic + static)	100
	DHCP static assignments per DHCP pool	20
	DHCP options	20
Interfaces	Force destination addresses for roaming	10
Layer 2	Virtual LANs (port-based; including VLAN 1)	24
	Multiple Spanning Tree instances	16

	Configurable function	Maximum number
Security	IP addresses from RADIUS servers	• AAA: 4 • WLAN: 2
	Management ACLs (access rules for management)	10
	MAC ACL rule configuration	20
	Ingress and egress rules for MAC ACL (total)	40 per interface (20 ingress rules / 20 egress rules) • Client: 80 (P1, WLAN) • Access point: 680 (P1, WDS 1,Y, VAP 1,Y) • Dual access point: 1320 (P1, WDS X,Y, VAP X,Y)
	IP ACL rule configuration	20
	Ingress and egress rules for port ACL IP (total)	40 per interface (20 ingress rules / 20 egress rules) • Client: 120 (P1, WLAN, management VLAN) • Access point: 720 (P1, WDS 1,Y, VAP 1,Y, management VLAN) • Dual access point: 1360 (P1, WDS X,Y, VAP X,Y, management VLAN)
	User roles	28
	User groups	32
	Users	28

SCALANCE W770 / W730

Configuration limits of the device

The following table lists the configuration limits for Web Based Management and the Command Line Interface of the device.

Depending on your device, some functions are not available.

	Configurable function	Maximum number
System	Syslog server	3
	SMTP server	3
	SNMPv1 trap recipient	10
	SNTP server	2
	NTP server	1
	DHCP pools	1
	IPv4 addresses managed by the DHCP server (dynamic + static)	100
	DHCP static assignments per DHCP pool	20
	DHCP options	20
Interfaces	Force destination addresses for roaming	10
Layer 2	Virtual LANs (port-based; including VLAN 1)	16
	Multiple Spanning Tree instances	16
Security	IP addresses from RADIUS servers	• AAA: 4 • WLAN: 2
	Management ACLs (access rules for management)	10
	MAC ACL rule configuration	20
	Ingress and egress rules for MAC ACL (total)	40 per interface (20 ingress rules / 20 egress rules) • Client: 80 (P1, WLAN) • Access point: 680 (P1, WDS 1.Y, VAP 1.Y)
	IP ACL rule configuration	20
	Ingress and egress rules for port ACL IP (total)	40 per interface (20 ingress rules / 20 egress rules) • Client: 120 (P1, WLAN, management VLAN) • Access point: 720 (P1, WDS 1.Y, VAP 1.Y, management VLAN)
	User roles	28
	User groups	32
	Users	28

SCALANCE W760 / W720

Configuration limits of the device

The following table lists the configuration limits for Web Based Management and the Command Line Interface of the device.

Depending on your device, some functions are not available.

	Configurable function	Maximum number
System	Syslog server	3
	SMTP server	3
	SNMPv1 trap recipient	10
	SNTP server	2
	NTP server	1
	DHCP pools	1
	IPv4 addresses managed by the DHCP server (dynamic + static)	100
	DHCP static assignments per DHCP pool	20
	DHCP options	20
Interfaces	Force destination addresses for roaming	10
Layer 2	Virtual LANs (port-based; including VLAN 1)	3
	Multiple Spanning Tree instances	16
Security	IP addresses from RADIUS servers	• AAA: 4 • WLAN: 2
	Management ACLs (access rules for management)	10
	MAC ACL rule configuration	20
	Ingress and egress rules for MAC ACL (total)	40 per interface (20 ingress rules / 20 egress rules) • Client: 80 (P1, WLAN) • Access point: 680 (P1, WDS 1.Y, VAP 1.Y)
	IP ACL rule configuration	20
	Ingress and egress rules for IP ACL IP (total)	40 per interface (20 ingress rules / 20 egress rules) • Client: 120 (P1, WLAN, management VLAN) • Access point: 720 (P1, WDS 1.Y, VAP 1.Y, management VLAN) • Dual access point: 1360 (P1, WDS X.Y, VAP X.Y, management VLAN)
	User roles	28
	User groups	32
	Users	28

Useful information

IP addresses

IPv4 / IPv6

What are the essential differences?

	IPv4	IPv6
IP configuration	• DHCP server • Manual	• Stateless Address Autoconfiguration (SLAAC): Stateless autoconfiguration using NDP (Neighbor Discovery Protocol) – Creates a link local address for every interface that does not require a router on the link. – Checks the uniqueness of the address on the link that requires no router on the link. – Specifies whether the global addresses are obtained via a status-free mechanism, a mechanism with status or via both mechanisms. (Requires a router on the link.) • Manual • DHCPv6 (status dependent)
Available IP addresses	32-bit: 4, 29 * 10^9 addresses	128-bit: 3, 4 * 10^{38} addresses
Address format	Decimal: 192.168.1.1 with port: 192.168.1.1:20	Hexadecimal: 2a00:ad80::0123 with port: [2a00:ad80::0123]:20
Loopback	127.0.0.1	::1
IP addresses of the interface	4 IP addresses	Multiple IP addresses • LLA: A link local address (formed automatically) fe80::/128 per interface • ULA: Several unique local unicast addresses per interface • GUA: Several global unicast addresses per interface
Header	• Checksum • Variable length • Fragmentation in the header • No security	• Checking at a higher layer • Fixed size • Fragmentation in the extension header
Fragmentation	Host and router	Only endpoint of the communication
Quality of service	Type of Service (ToS) for prioritization	The prioritization is specified in the header field "Traffic Class".

	IPv4	IPv6
Types of frame	Broadcast, multicast, unicast	Multicast, unicast, anycast
Identification of DHCP clients/ server	Client ID: MAC address	DUID + IAID(s) = exactly one interface of the host DUID = DHCP unique identifier Identifies server and clients uniquely and should not change, not even when replacing network components! IAID = Identity Association Identifier At least one per interface is generated by the client and remains unchanged when the DHCP client restarts Three methods of obtaining the DUID • DUID-LLT • DUID-EN • DUID-LL
DHCP	via UDP with broadcast	via UDP with unicast RFC 3315, RFC 3363 **Stateful DHCPv6** Status-dependent configuration in which the IPv6 address and the configuration settings are transferred. Four DHVPv6 messages are exchanged between client and server: 1. SOLICIT: Sent by the DHCPv6 client to localize DHCPv6 servers. 2. ADVERTISE The available DHCPv6 servers reply to this. 3. REQUEST The DHCPv6 client requests an IPv6 address and the configuration settings from the DHCPv6 server. 4. REPLY The DHCPv6 server sends the IPv6 address and the configuration settings. If the client and server support the function "Rapid commit" the procedure is shortened to two DHCPv6 messages SOLICIT and REPLY . **Stateless autoconfiguration** In stateless DHCPv6, only the configuration settings are transferred. **Prefix delegation** The DHCPv6 server delegates the distribution of IPv6 prefixes to the DHCPv6 client. The DHCPv6 client is also known as PD router.
Resolution of IP addresses in hardware addresses	ARP (Address Resolution Protocol)	NDP (Neighbor Discovery Protocol)

IPv4 address

Structure of an IPv4 address

Address classes

IP address range	Max. number of networks	Max. number of hosts/network	Class	CIDR
1.x.x.x through 126.x.x.x	126	16777214	A	/8
128.0.x.x through 191.255.x.x	16383	65534	B	/16
192.0.0.x through 223.255.255.x	2097151	254	C	/24
224.0.0.0 - 239.255.255.255	Multicast applications		D	
240.0.0.0 - 255.255.255.255	Reserved for future applications		E	

An IP address consists of 4 bytes. Each byte is represented in decimal, with a dot separating it from the previous one. This results in the following structure, where XXX stands for a number between 0 and 255:

XXX.XXX.XXX.XXX

The IP address is made up of two parts, the network ID and the host ID. This allows different subnets to be created. Depending on the bytes of the IP address used as the network ID and those used for the host ID, the IP address can be assigned to a specific address class.

Subnet mask

The bits of the host ID can be used to create subnets. The leading bits represent the address of the subnet and the remaining bits the address of the host in the subnet.

A subnet is defined by the subnet mask. The structure of the subnet mask corresponds to that of an IP address. If a "1" is used at a bit position in the subnet mask, the bit belongs to the corresponding position in the IP address of the subnet address, otherwise to the address of the computer.

Example of a class B network:

The standard subnet address for class B networks is 255.255.0.0; in other words, the last two bytes are available for defining a subnet. If 16 subnets must be defined, the third byte of the subnet address must be set to 11110000 (binary notation). In this case, this results in the subnet mask 255.255.240.0.

To find out whether two IP addresses belong to the same subnet, the two IP addresses and the subnet mask are ANDed bit by bit. If both logic operations have the save result, both IP addresses belong to the same subnet, for example, 141.120.246.210 and 141.120.252.108.

Outside the local area network, the distinction between network ID and host ID is of no significance, in this case packets are delivered based on the entire IP address.

Note

In the bit representation of the subnet mask, the "ones" must be set left-justified; in other words, there must be no "zeros" between the "ones".

IPv6 addresses

IPv6 terms

Network node

A network node is a device that is connected to one or more networks via one or more interfaces.

Router

A network node that forwards IPv6 packets.

Host

A network node that represents an end point for IPv6 communication relations.

Link

A link is, according to IPv6 terminology, a direct layer 3 connection within an IPv6 network.

Neighbor

A network node located on the same link as the network node.

IPv6 interface

Physical or logical interface on which IPv6 is activated.

Path MTU

Maximum permitted packet size on a path from a sender to a recipient.

Path MTU discovery

Mechanism for determining the maximum permitted packet size along the entire path from a sender to a recipient.

LLA

Link local address FE80::/10

As soon as IPv6 is activated on the interface, a link local address is formed automatically. Can only be reached by accounts located on the same link.

ULA

Unique Local Address

Defined in RFC 4193. Via this address, the IPv6 interface can be reached in the LAN.

GUA

Global Unicast Address Via this address, the IPv6 interface can be reached, e.g. via the Internet.

Interface ID

The interface ID is formed with the EUI-64 method or manually.

EUI-64

Extended Unique Identifier (RFC 4291); method for forming the interface ID. In Ethernet, the interface ID is formed from the MAC address of the interface. Divides the MAC address into

the manufacturer-specific part (OUI) and the network-specific part (NIC) and inserts FFFE between the two parts.

Example:

MAC address = AA:BB:CC:DD:EE:FF

OUI = AA:BB:CC

NIC = DD:EE:FF

EUI-64 = OUI + **FFFE** + NIC = AA:BB:CC:**FF:FE**:DD:EE:FF

Scope

Defines the range of the IPv6 address.

Structure of an IPv6 address

IPv6 address format - notation

IPv6 addresses consist of 8 fields each with four-character hexadecimal numbers (128 bits in total). The fields are separated by a colon.

Example:

fd00:0000:0000:ffff:02d1:7d01:0000:8f21

Rules / simplifications:

- If one or more fields have the value 0, a shortened notation is possible.
  The address fd00:**0000:0000**:ffff:02d1:7d01:0000:8f21 can also be shortened and written as follows:
  fd00**::**ffff:02d1:7d01:0000:8f21
  To ensure uniqueness, this shortened form can only be used once within the entire address.
- Leading zeros within a field can be omitted.
  The address fd00:0000:0000:ffff:**02d1**:7d01:0000:8f21 can also be shortened and written as follows:
  fd00**::**ffff:**2d1**:7d01:0000:8f21
- Decimal notation with periods
  The last 2 fields or 4 bytes can be written in the normal decimal notation with periods.
  Example: The IPv6 address fd00::ffff.125.1.0.1 is equivalent to fd00::ffff:7d01:1

Structure of the IPv6 address

The IPv6 protocol distinguishes three types of address: Unicast , anycast and multicast. The following section describes the structure of the global unicast addresses.

IPv6 prefix		Suffix
Global prefix: n bits	Subnet ID m bits	Interface ID 128 - n - m bits
Assigned address range	Description of the location, also subnet prefix or subnet	Unique assignment of the host in the network. The ID is generated from the MAC address.

The prefix for the link local address is always fe80:0000:0000:0000. The prefix is shortened and noted as follows: fe80::

IPv6 prefix

Specified in: RFC 4291

The IPv6 prefix represents the subnet identifier.

Prefixes and IPv6 addresses are specified in the same way as with the CIDR notation (Classless Inter-Domain Routing) for IPv4.

Design

IPv6 address / prefix length

Example

IPv6 address: 2001:0db8:1234::1111/48

Prefix: 2001:0db8:1234::/48

Interface ID: ::1111

Entry and appearance

The entry of IPv6 addresses is possible in the notations described above. IPv6 addresses are always shown in the hexadecimal notation.

VLAN

VLAN tagging

Expansion of the Ethernet frames by four bytes

For CoS (Class of Service, frame priority) and VLAN (virtual network), the IEEE 802.1Q standard defined the expansion of Ethernet frames by adding the VLAN tag.

Note

The VLAN tag increases the permitted total length of the frame from 1518 to 1522 bytes. The end nodes on the networks must be checked to find out whether they can process this length / this frame type. If this is not the case, only frames of the standard length may be sent to these nodes.

The additional 4 bytes are located in the header of the Ethernet frame between the source address and the Ethernet type / length field:

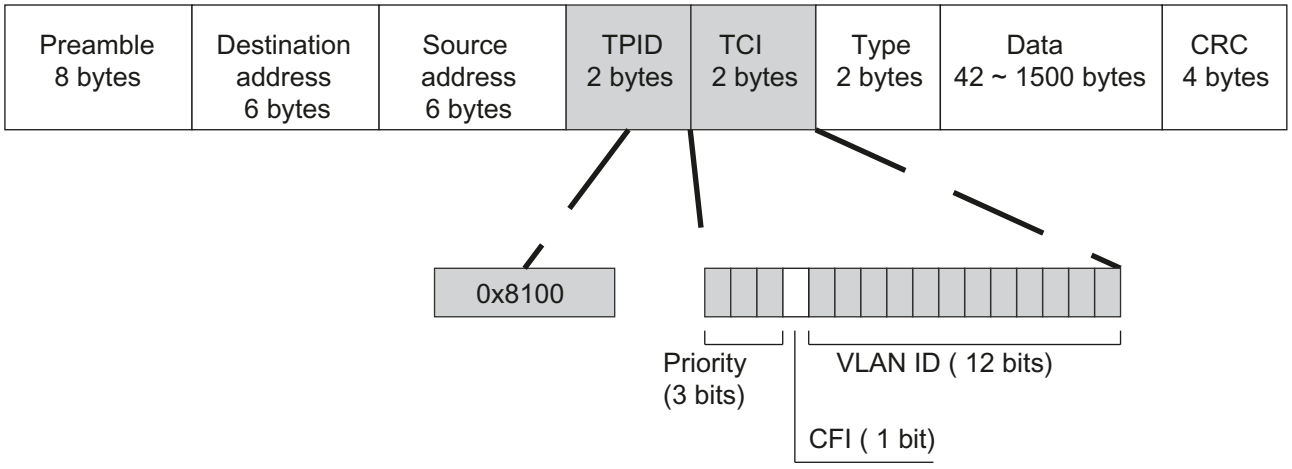


Figure 10-17 Structure of the expanded Ethernet frame

The additional bytes contain the tag protocol identifier (TPID) and the tag control information (TCI).

Tag protocol identifier (TPID)

The first 2 bytes form the Tag Protocol Identifier (TPID) and always have the value 0x8100. This value specifies that the data packet contains VLAN information or priority information.

Tag Control Information (TCI)

The 2 bytes of the Tag Control Information (TCI) contain the following information:

QoS Trust

The tagged frame has 3 bits for the priority that is also known as Class of Service (CoS), see also IEEE 802.1Q.

CoS bits	Priority	Type of the data traffic
000	0 (lowest)	Background
001	1	Best Effort
010	2	Excellent Effort
011	3	Critical Applications
100	4	Video, < 100 ms delay (latency and jitter)
101	5	Voice (language), < 10 ms delay (latency and jitter)
110	6	Internetwork Control
111	7 (highest)	Network Control

The prioritization of the data packets is possible only if there is a queue in the components in which they can buffer data packets with lower priority.

The device has multiple parallel queues in which the frames with different priorities can be processed. As default, first, the frames with the highest priority are processed. This method ensures that the frames with the highest priority are sent even if there is heavy data traffic.

Canonical Format Identifier (CFI)

The CFI is required for compatibility between Ethernet and the token Ring.
The values have the following meaning:

Value	Meaning
0	The format of the MAC address is canonical. In the canonical representation of the MAC address, the least significant bit is transferred first. Standard-setting for Ethernet switches.
1	The format of the MAC address is not canonical.

VLAN ID

In the 12-bit data field, up to 4096 VLAN IDs can be formed. The following conventions apply:

VLAN ID	Meaning
0	The frame contains only priority information (priority tagged frames) and no valid VLAN identifier.
1- 4094	Valid VLAN identifier, the frame is assigned to a VLAN and can also include priority information.
4095	Reserved

VLAN

Network definition regardless of the spatial location of the nodes

VLAN (Virtual Local Area Network) divides a physical network into several logical networks that are shielded from each other. Here, devices are grouped together to form logical groups. Only nodes of the same VLAN can address each other. Since multicast and broadcast frames are only forwarded within the particular VLAN, they are also known as broadcast domains.

The particular advantage of VLANs is the reduced network load for the nodes and network segments of other VLANs.

To identify which frame belongs to which VLAN, the frame is expanded by 4 bytes. This expansion is also known as VLAN tag and includes not only the VLAN ID but also priority information.

Note

When the device receives a doubly tagged frame, the first tag (vlan) is evaluated for the path and the second tag for the priority.

Options for the VLAN assignment

There are various options for the assignment to VLANs:

- Port-based VLAN
  Each port of a device is assigned a VLAN ID. You configure port-based VLAN in "Layer 2 > VLAN".
- Protocol-based VLAN
  Each port of a device is assigned a protocol group.
- Subnet-based VLAN
  The IP address of the device is assigned a VLAN ID.

MAC-based communication

Frames sent by the client to the access point always have the MAC address of the WLAN client as the source MAC address. In the "learning table" of the access point there is therefore only the MAC address of the WLAN client.

MAC mode "Automatic", "Manual" and "Own"

If the MAC address of a device connected to the client is adopted (Automatic) or is set manually (Manual), both the MAC-based and the IP-based frames find their destination for precisely this device. If the MAC address of the Ethernet interface of the WLAN client is used (Own), the MAC-based and IP-based frames only reach the WLAN client.

The access point checks whether the destination MAC address matches the MAC addresses of the connected clients. Since a WLAN client can only use a MAC address, communication at the MAC address level (ISO/OSI layer 2) can be to a maximum of one node downstream from the client or the client itself.

With IP Mapping, several nodes downstream from a client can be addressed based on the IP protocol. The IP packets are broken down according to an internal table and forwarded to the connected devices.

Maximum possible number of Ethernet nodes with layer 2 communication downstream from the client: 1

Notes on the "Automatic" setting:

- As long as there is no link on the Ethernet interface, the device uses the MAC address of the Ethernet interface so that it can be reached in this status. In this status, the device can be found using the Primary Setup Tool and configured with WBM or CLI.
- As soon as there is a link on the Ethernet interface, the device adopts the source MAC address of the first received frame.

Note

From the moment that the device adopts another MAC address (manually or automatically), the device no longer responds to queries of the Primary Setup Tool when the query is received over the WLAN interface. Queries of the PST over the Ethernet interface continue to be replied to.

MAC mode "Layer 2 Tunnel"

With the setting "Layer 2 Tunnel", the client provides information about the devices downstream from it when it logs in to an access point. This makes it possible to enter the MAC addresses of these devices in the "learning table" of the access point. The access point can forward MAC-based frames for the devices downstream from the client to the appropriate client.

In much the same way as with WDS, a separate port is created for the L2T client over which the Ethernet frames are sent without changing the destination MAC address.

Maximum possible number of Ethernet nodes downstream from the client: 8

IEEE 802.11n

Overview

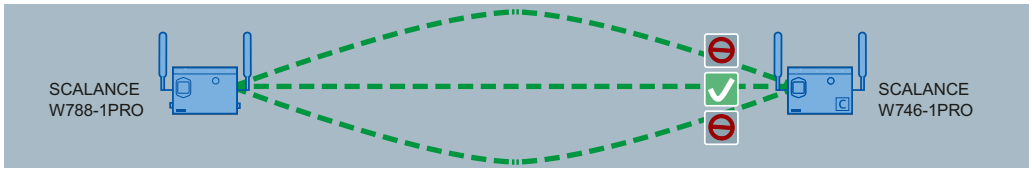
The standard IEEE 802.11n is an expansion of the 802.11 standard and was approved in 2009. Previous standards worked either in the 2.4 GHz frequency band (IEEE 802.11g /b) or in the 5 GHz frequency band (IEEE 802.11a). IEEE 802.11n can operate in both frequency band. In the IEEE 802.11n standard, there are mechanisms implemented in PHY and MAC layers that increase the data throughput and improve the wireless coverage.

- MIMO antenna technology
- Maximum ratio combining (MRC)
- Spatial multiplexing
- Channel bonding
- Frame aggregation
- Accelerated guard interval
- Modulation and coding scheme
- Data throughput rates up to 450 Mbps (gross)
  This is not possible on all SCALANCE W700 devices.

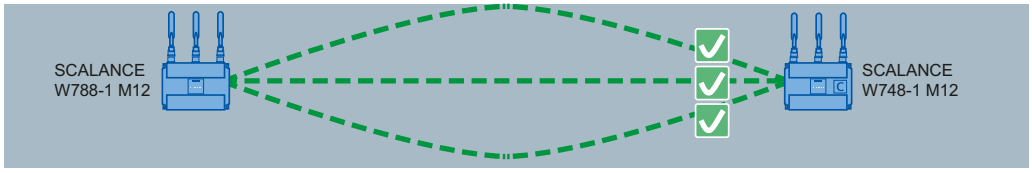
MIMO antenna technology

MIMO (Multiple Input - Multiple Output) is based on an intelligent multiple antenna system. The transmitter and the receiver have several spatially separate antennas. The spatially separate antennas transmit the data streams at the same time. Up to four data streams are possible. The data streams are transmitted over spatially separate paths and return over different paths due to diffraction, refraction, fading and reflection (multipath propagation). The multipath propagation means that at the point of reception a complex, space- and time-dependent pattern results as a total signal made up of the individual signals sent. MIMO uses this unique pattern by detecting the spatial position of characteristic signals. Here, each spatial position is different from the neighboring position. By characterizing the individual senders, the recipient is capable of separating several signals from each other.

Multipath propagation with IEEE 802.11a/b/g



Multipath propagation IEEE 802.11n (MIMO)



Maximum ratio combining (MRC)

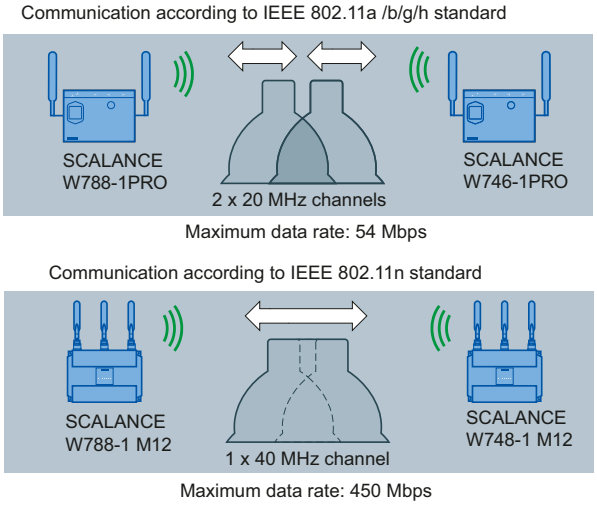
In a multiple antenna system, the wireless signals are received by the individual antennas and combined to form one signal. The MRC method is used to combine the wireless signals. The MRC method weights the wireless signals according to their signal-to-noise ratio and combines the wireless signals to form one signal. The signal-to-noise ratio is improved and the error rate is reduced.

Spatial mutliplexing

With spatial multiplexing, different information is sent using the same frequency. The data stream is distributed over n transmitting antennas; in other words, each antenna sends only 1/n of the data stream. The division of the data stream is restricted by the number of antennas. At the receiver end, the signal is reconstructed.
Due to the spatial multiplexing, there is a higher signal-to-noise ratio and a higher data throughput.

Channel bonding

With IEEE 802.11n, data can be transferred via two directly neighboring channels. The two 20 MHz channels are put together to form one channel with 40 MHz. This allows the channel bandwidth to be doubled and the data throughput to be increased.
To be able to use channel bonding, the recipient must support 40 MHz transmissions. If the recipient does not support 40 MHz transmissions, the band is automatically reduced to 20 MHz. This means that IEEE 802.11n can also communicate with IEEE 802.11a/b/g devices.
The channel bundling is set on the "AP" WBM page with the "HT Channel Width [MHz]" parameter.



Frame aggregation

With IEEE 802.11n, it is possible to group together individual data packets to form a single larger packet; this is known as frame aggregation. There are two types of frame aggregation: Aggregated MAC Protocol Data Unit (A-MPDU) and Aggregated MAC Service Data Unit (A-MSDU).
The frame aggregation reduces the packet overheads. Frame aggregation can only be used if the individual data packets are intended for the same receiving station (client).
The SCALANCE W700 devices support both types of frame aggregation. You specify the settings for the A-MPDU data packet on the "AP 802.11n" WBM page.

Accelerated guard interval

The guard interval prevents different transmissions being mixed together. In telecommunications, this mixing is also known as intersymbol interference (ISI).
When the send time has elapsed, a send pause (guard interval) must be kept to before the next transmission begins.

The guard interval of IEEE 802.11a /b/g is 800 ns. IEEE 802.11n can use the reduced guard interval of 400 ns. You specify the guard interval on the "AP 802.11n" WBM page.

Modulation and coding schemes

The IEEE 802.11n standard supports different data rates. The data rates are based on the number of spatial streams, the modulation method and the channel coding. The various combinations are described in modulation and coding schemes.

PROFINET

PROFINET

PROFINET is an open standard (IEC 61158/61784) for industrial automation based on Industrial Ethernet. PROFINET uses existing IT standards and allows end-to-end communication from the field level to the management level as well as plant-wide engineering. PROFINET also has the following features:

- Use of TCP/IP
- Automation of applications with real-time requirements
  - Real-Time (RT) communication
  - Isochronous Real-Time (IRT) communication
- Seamless integration of fieldbus systems

You configure PROFINET in "System > PROFINET (Page 1374)".

PROFINET IO

Within the framework of PROFINET, PROFINET IO is a communications concept for implementing modular, distributed applications. PROFINET IO is implemented by the PROFINET standard for programmable controllers (IEC 61158-x-10).

EtherNet/IP

EtherNet/IP

EtherNet/IP (Ethernet/Industrial Protocol) is an open industry standard for industrial real-time Ethernet based on TCP/IP and UDP/IP. With EtherNet/IP, Ethernet is expanded by the Common Industrial Protocol (CIP) at the application layer. In EtherNet/IP, the lower layers of the OSI reference model are adopted by Ethernet with the physical, network and transport functions.

You configure EtherNet/IP in "System > EtherNet/IP (Page 1375)".

Common Industrial Protocol

The Common Industrial Protocol (CIP) is an application protocol for automation that supports transition of the field buses in Industrial Ethernet and in IP networks. This industry protocol is used by field buses/industrial networks such as DeviceNet, ControlNet and EtherNet/IP at the application layer as an interface between the deterministic fieldbus world and the automation application (controller, I/O, HMI, OPC, ...). The CIP is located above the transport layer and expands the pure transport services with communications services for automation engineering. These include services for cyclic, time-critical and event-controlled data traffic. CIP distinguishes between time-critical I/O messages (implicit messages) and individual query/ response frames for configuration and data acquisition (explicit messages). CIP is object-oriented; all data "visible" from the outside is accessible in the form of objects. CIP has a common configuration basis: EDS (Electronic Data Sheet).

Electronic Data Sheet

Electronic Data Sheet (EDS) is an electronic datasheet for describing devices.

The EDS required for EtherNet/IP operation can be found in "System > Load&Save".

NAT/NAPT

What is NAT?

With Network Address Translation (NAT), the IPv4 address in a data packet is replaced by another. NAT is normally used on a gateway between a private LAN and an external network with globally valid IPv4 addresses. A local IPv4 address of the internal LAN is changed to an external global IPv4 address by a NAT device at the gateway.

To translate the internal into the global IPv4 address, the NAT device maintains a translation list. The address assignment is automatic. You configure the address assignment in "Layer 3 > NAT > Basic".

What is NAPT?

In "Network Address Port Translation" (NAPT) or "Port Address Translation" (PAT), several internal source IPv4 addresses are translated into the same external source IPv4 address. To identify the individual source nodes, the port of the source device is also stored in the translation list of the NAT gateway and translated for the external address.

If several local clients send a query to the same external destination IPv4 address via the NAT gateway, the gateway enters its own external source IPv4 address in the header of these data packets. Since the forwarded data packets have the same global source IPv4 address, the NAT gateway assigns the data packets to the clients using different port number.

Note

NAT/NAPT is possible only on layer 3 of the ISO/OSI reference model. To use the NAT function, the networks must use the IP protocol.

When using the ISO protocol that operates at layer 2, it is not possible to use NAT.

If a client from the global network wants to use a service in the internal network, the translation list for the static address assignment needs to be configured. You configure the translation list for NAPT in "Layer 3 > NAT > NAPT".

iPCF / iPCF-MC

The wireless range of an IWLAN system can be expanded by using multiple access points. If a client moves from the area covered by one access point to the area covered by another access point, the wireless link is maintained after a short interruption (roaming).

In an industrial environment, there are applications that require a deterministic response when there are large numbers of nodes and when roaming to another cell require handover times of less than 100 milliseconds.

- **iPCF** (industrial Point Coordination Function)
  iPCF ensures that the entire data traffic of a cell is ordered, controlled by the access point. Even with large numbers of nodes, collisions can also be avoided. iPCF also allows fast cell changes.
  You configure iPCF in "iFeatures > iPCF (Page 1729)".
- **iPCF-MC** (industrial Point Coordination Function - Management Channel)
  iPCF-MC was developed to make the advantages of iPCF available to fully mobile nodes that communicate without being dependent on RCoax cable or directional antennas. With iPCF-MC, the client also searches for potentially suitable access points when it receives iPCF queries from the access point and the existing connection to an access point is working problem-free. This means that if a change to a different access point is necessary, this is achieved extremely quickly. In contrast to iPCF, the handover times for iPCF-MC are not dependent on the number of wireless channels being used.
  You configure iPCF-MC in "iFeatures > iPCF-MC (Page 1731)".

iPCF / iPCF-MC - how it works

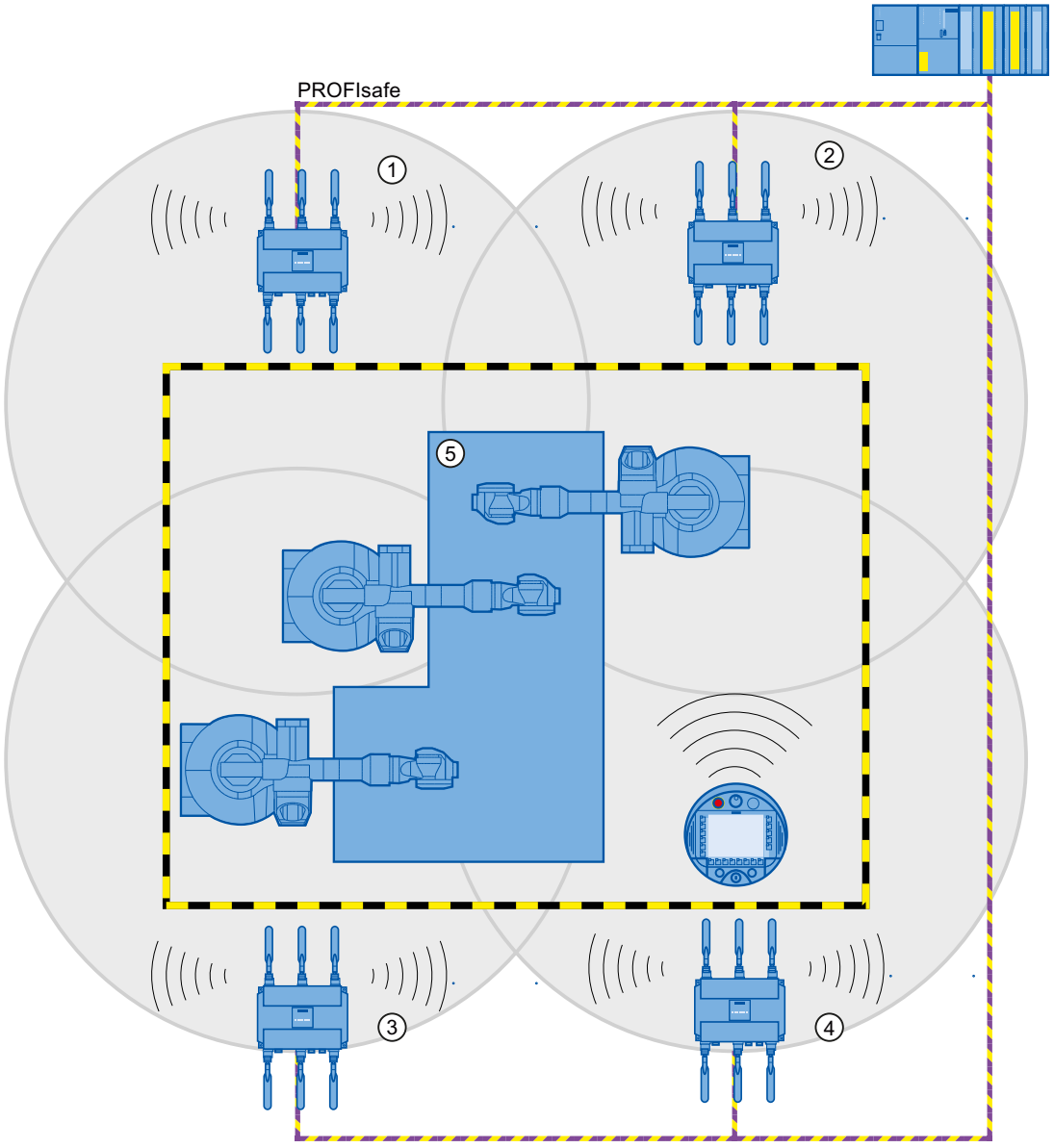
The access point checks all nodes in the wireless cell cyclically. At the same time, the scan includes the downlink traffic for this node. In the reply, the node sends the uplink data. The access point scans a new node at least every 5 ms.

The scan of a node is seen by all other nodes in the cell. This allows a client to detect the quality of the wireless link to the access point even when it is not communicating with the access point itself. If the client does not receive any frames from the access point for a certain time, it starts to search for a new access point.

In iPCF mode, both the search for a new access point and the registration with this access point have been optimized in terms of time. Handover times significantly below 50 ms are achieved.

Stable PNIO communication is only possible when a WLAN client is in a cell with more than 60 % (-65 dBm) signal strength at all times. This can be checked by activating and deactivating the various wireless cells.

This does not mean that the client needs to change when there is a signal strength < 60 % (< -65 dBm). Make sure that access points are available with adequate signal strength.



① Wireless cell of access point 1
② Wireless cell of access point 2
③ Wireless cell of access point 3
④ Wireless cell of access point 4
⑤ Plant

Figure 10-18 Configuration example of iPCF-MC

Restrictions

- iPCF and iPCF-MC are developments of Siemens AG and function only with nodes on which iPCF / iPCF-MC is implemented.
- With an access point with several WLAN interfaces, it is possible to use both iPCF as well as standard WLAN at the same time.
- Access points with a WLAN interface cannot take part in the iPCF-MC procedures, iPCF is, however, possible.

Requirements for iPCF-MC

iPCF-MC uses the two wireless interface of the access point in different ways: One interface works as the management interface and sends a beacon every five milliseconds. The other interface transfers the user data.

The following requirements must be met before you can use iPCF-MC:

- Only SCALANCE W700 devices with two WLAN interfaces can be used as access points
- The data interface (WLAN1) and management interface (WLAN2) must be operated in the same frequency band and must match in terms of their wireless coverage. iPCF-MC will not work if the two wireless interfaces are equipped with directional antennas that cover different areas.
- The management interfaces of all access points to which a client can change must use the same channel. A client scans only this one channel to find accessible access points.
- Transmission based on IEEE 802.11h (DFS) cannot be used for the management interface. 802.11h (DFS) is possible for the data interface.
- A client must support this feature on its WLAN interface.

iREF

How it works

If an access point has several activated antennas, the transmit power is distributed equally on these antennas. The transmit power is subject to country-specific legal restrictions. The maximum permitted power depends on the gain of the connected antennas. If the connected antennas have different gains, the maximum antenna gain effectively restricts the permitted transmit power.

iREF (industrial Range Extension Function) ensures that the data traffic from the access point to each individual client is handled via the most suitable antenna. Which antenna is most suitable is determined by the access point based on the RSSI values of received packets.

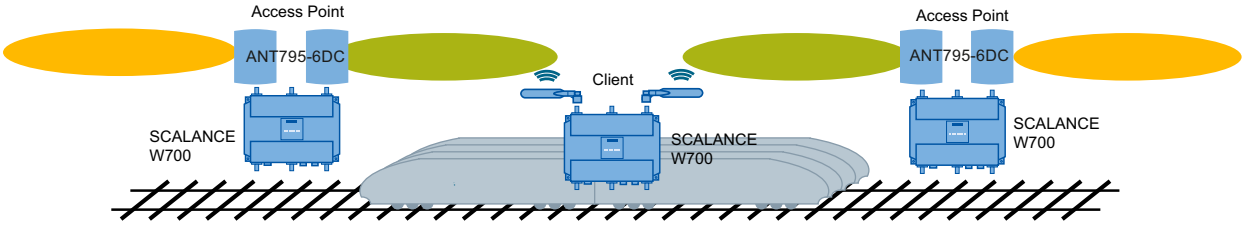
Taking into account antenna gain and possible cable losses, packets are only sent via the antennas with which the maximum signal strength at the client end can be expected.

During this time the other antennas are inactive and the legally permitted transmit power is available for the selected antenna. The inactive antennas do not restrict the permitted transmit power.

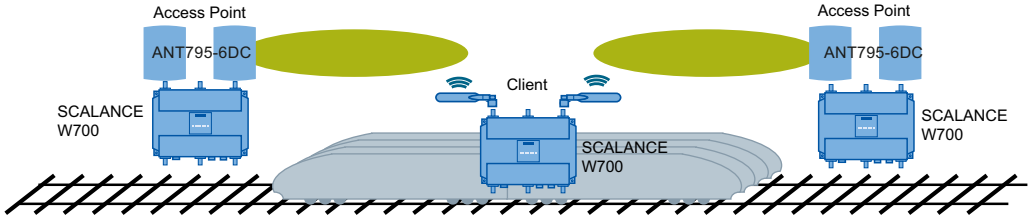
In particular in applications in which MIMO cannot be used or brings no advantage, this allows data to be transmitted at the highest possible data transmission rate.

You configure iREF in "iFeatures > iREF"

without iREF



where iREF



Requirement

- To be able to use iREF, the SCALANCE W700 device must have at least 2 activated antennas.

Restrictions

- A maximum data rate of only up to 150 Mbps (MCS 0 - 7 or 1 x spatial stream) is possible
- iREF cannot be used along with other iFeatures (for example iPCF or iPCF-MC)

Advantages

- Due to the directional data transmission and dynamic deactivation of antennas that do not radiate in the direction of the particular client, interference can be reduced.
- The signal strength is improved because the active antenna always has the maximum permitted transmit power available.

iPRP

The "Parallel Redundancy Protocol" (PRP) is a redundancy protocol for cabled networks. It is defined in Part 3 of the IEC 62439 standard.

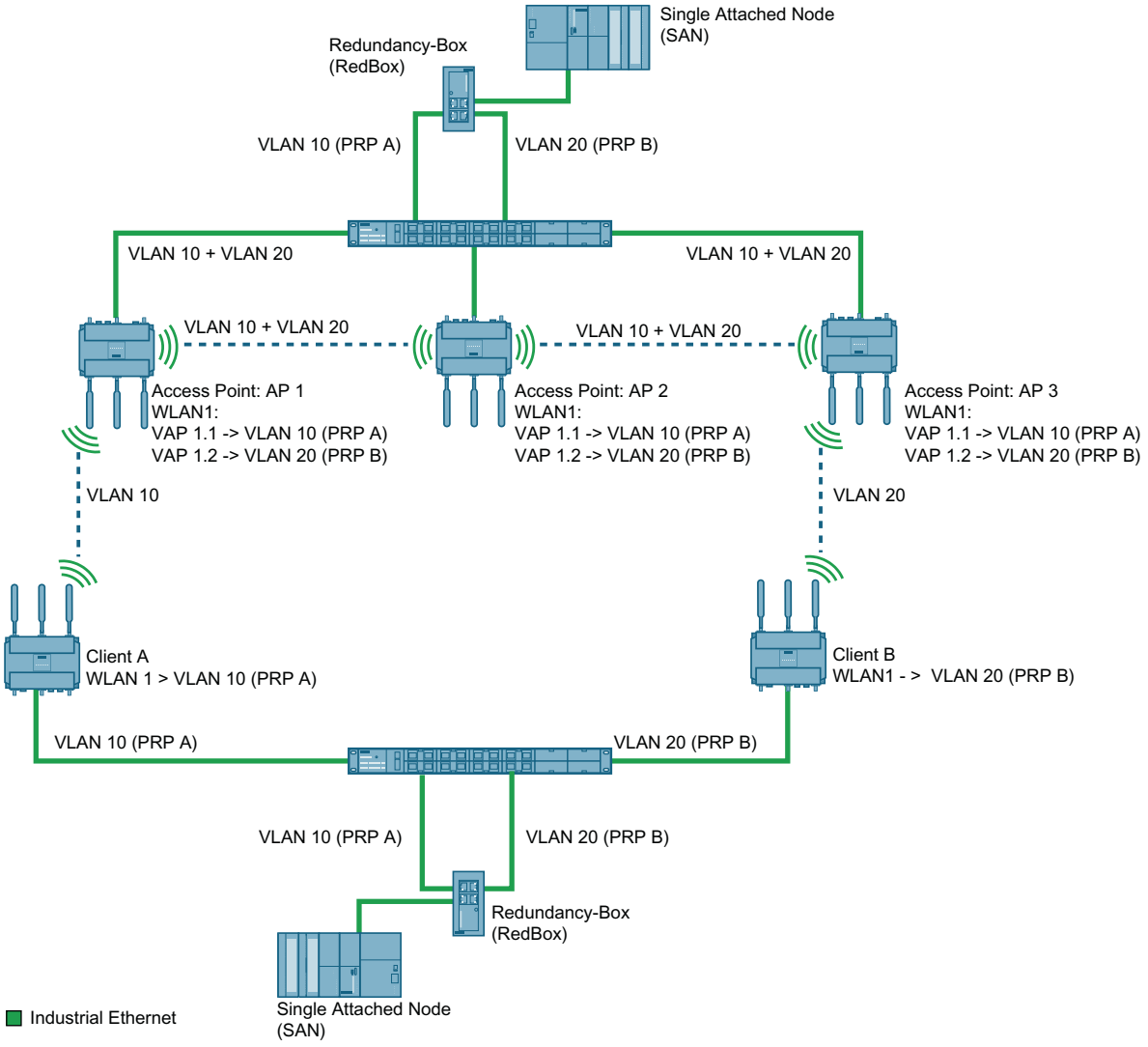
With the "industrial Parallel Redundancy Protocol" (iPRP) the PRP technology can be used in wireless networks. iPRP improves the availability of wireless communication.

How it works

A PRP network consists of two completely independent networks. If one network is disrupted, the frames are sent without interruption/reconfiguration via the parallel redundant network. To achieve this the Ethernet frames are sent to the recipient in duplicate via both networks. Devices capable of PRP have at least two separate Ethernet interfaces that are connected to independent networks.

With devices not capable of PRP a redundancy box (RedBox) is connected upstream. This allows access for so-called Single Attached Nodes (SAN) to PRP networks. The RedBox duplicates every Ethernet frame to be sent and among other things adds the VLAN ID and a sequence number. The RedBox sends a copy of the frame at the same time on both Ethernet interfaces via both networks.

With PRP the SCALANCE W700 devices PRP can be used in wireless networks.



The access points (AP 1, AP 2 and AP 3) and the RedBox A are connected to each other via a switch. The PRP frames are sent to the access points via both networks (PRP A, PRP B). The access points receive the PRP frame.

The PRP frames are sent at the same time on two different wireless links to the recipients. The clients A and B are connected to different access points at the same time. The clients are never connected to the access point via the same interface. The clients and the RedBox B are connected via a switch.

Note

The interface of the switch and of the client must be members of the same VLAN (PRP network). For client A, VLAN 10 (PRP A) is set in the PRP network. The Ethernet interface of the switch may only be a member in VLAN 10 but not in VLAN 20 of client B.

The RedBox B forwards the first PRP frame to arrive and discards the second PRP frame.

The redundant partners (here: AP1 and AP3 or client A and client B) communicate with each other to prevent the two redundant PRP frames from arriving at the RedBox with too great a time difference.

If for example communication between AP1 and client A is not possible, the PRP frame comes to its destination via redundant client B.

You configure iPRP in "iFeatures > iPRP (Page 1732)".

Requirement

- The base bridge mode "802.1Q VLAN Bridge" is set.
- The VLANs have been created.
- The VLANs are configured on the same interface.
- Access point mode: The VAP interface is enabled.
- Client mode: In MAC mode "Layer 2 Tunnel" is set.
- Depending on the configuration the clients can communicate with every access point.

AeroScout

AeroScout tags

SCALANCE W700 devices support tags of the AeroScout company. Tags are battery-operated RFID sensors that send their data cyclically as multicast frames.

Among other things, AeroScout tags have the following features:

- **Ambient temperature**
  If a tag is fitted to a SCALANCE W700 device or material, it is possible to monitor whether a selected ambient temperature is being maintained.
- **Motion**
  Here, a tag can also supply information indicating whether it is in motion or stationary. The areas of material flow and material handling engineering represent possible applications for this function.

- **Button**
  Regardless of the frames sent cyclically, a user can also send a message by pressing a button.
- **LED**
  This provides information on the operating status of the tag.

Note

For more detailed information, please refer to the AeroScout documentation (www.aeroscout.com).

How it works

The tag sends its data as AeroScout frames. The tags and the access points communicate in the 2.4 GHz band.

If the WLAN interface of the access point receives the AeroScout frame, this is converted into a UDP datagram. The SCALANCE W700 device forwards the UDP datagram along with the information about the signal strength (RSSI) to a PC. The AeroScout Engine runs on the PC and evaluates the received information.

Note

It is **not** advisable to use PROFINET communication and AeroScout together on one wireless interface.

Accuracy of localization

To achieve optimum precision in the localization of AeroScout Tags,

- we recommend the use of antennas with omnidirectional characteristics
- if the signals should be received by at least three access points.

SNMP

Introduction

With the aid of the Simple Network Management Protocol (SNMP), you monitor and control network components from a central station, for example routers or switches. SNMP controls the communication between the monitored devices and the monitoring station.

Tasks of SNMP:

- Monitoring of network components
- Remote control and remote parameter assignment of network components
- Error detection and error notification

In versions v1 and v2c, SNMP has no security mechanisms. Each user in the network can access data and also change parameter assignments using suitable software.

For the simple control of access rights without security aspects, community strings are used.

The community string is transferred along with the query. If the community string is correct, the SNMP agent responds and sends the requested data. If the community string is not correct, the SNMP agent discards the query. Define different community strings for read and write permissions. The community strings are transferred in plain text.

Standard values of the community strings:

- public
  has only read permissions
- private
  has read and write permissions

Note

Because the SNMP community strings are used for access protection, do not use the standard values "public" or "private". Change these values following the initial commissioning.

Further simple protection mechanisms at the device level:

- Allowed Host
  The IP addresses of the monitoring systems are known to the monitored system.
- Read Only
  If you assign "Read Only" to a monitored device, monitoring stations can only read out data but cannot modify it.

SNMP data packets are not encrypted and can easily be read by others.

The central station is also known as the management station. An SNMP agent is installed on the devices to be monitored with which the management station exchanges data.

The management station sends data packets of the following type:

- GET
  Request for a data record from the SNMP agent
- GETNEXT
  Calls up the next data record.
- GETBULK (available as of SNMPv2c)
  Requests multiple data records at one time, for example several rows of a table.
- SET
  Contains parameter assignment data for the relevant device.

The SNMP agent sends data packets of the following type:

- RESPONSE
  The SNMP agent returns the data requested by the manager.
- TRAP
  If a certain event occurs, the SNMP agent itself sends traps.

SNMPv1/v2c/v3 use UDP (User Datagram Protocol) and use the UDP ports 161 and 162. The data is described in a Management Information Base (MIB).

SNMPv3

Compared with the previous versions SNMPv1 and SNMPv2c, SNMPv3 introduces an extensive security concept.

SNMPv3 supports:

- Fully encrypted user authentication
- Encryption of the entire data traffic
- Access control of the MIB objects at the user/group level

With the introduction of SNMPv3 you can no longer transfer user configurations to other devices without taking special action, e.g. by loading a configuration file or replacing the C-PLUG.

According to the standard, the SNMPv3 protocol uses a unique SNMP engine ID as an internal identifier for an SNMP agent. This ID must be unique in the network. It is used to authenticate access data of SNMPv3 users and to encrypt it.

Depending on whether you have enabled or disabled the “SNMPv3 User Migration” function, the SNMP engine ID is generated differently.

Restriction when using the function

Use the "SNMPv3 User Migration" function only to transfer configured SNMPv3 users to a substitute device when replacing a device.
Do not use the function to transfer configured SNMPv3 users to multiple devices. If you load a configuration with created SNMPv3 users on several devices, these devices use the same SNMP engine ID. If you use these devices in the same network, your configuration contradicts the SNMP standard.

Compatibility with predecessor products

You can only transfer SNMPv3 users to a different device if you have created the users as migratable users. To create a migratable user the "SNMPv3 User Migration" function must be activated when you create the user.

Spanning Tree

Avoiding loops

The Spanning Tree algorithm detects redundant physical network structures and prevents the formation of loops by disabling redundant paths. It evaluates the distance and performance of a connection or bases the decisions on settings made by the user. Data is then exchanged only over the remaining connection paths.

If the preferred data path fails, the Spanning Tree algorithm then searches for the most efficient path possible with the remaining nodes.

Root bridge and bridge priority

The identification of the most efficient connection is always related to the root bridge, a network component that can be considered as a root element of a tree-like network structure. With the "Bridge Priority" parameter, you can influence the selection of the root bridge. The computer with the lowest value set for this parameter automatically becomes the root bridge. If two computers have the same priority value, the computer with the lower MAC address becomes the root bridge.

Response to changes in the network topology

If nodes are added to a network or drop out of the network, this may affect the optimum path selection for data packets. To be able to respond to such changes, the root bridge sends configuration messages (BPDUs) at regular intervals. You can set the interval between two configuration messages with the "Hello Time" parameter.

Keeping configuration information up to date

With the "Max Age" parameter, you set the maximum age of configuration information. If a bridge has information that is older than the time set in Max Age, it discards the message and initiates recalculation of the paths.

New configuration data is not used immediately by a bridge but only after the period specified in the "Forward Delay" parameter. This ensures that operation is started with the new topology only after all the bridges have the required information.

RSTP, MSTP, CIST

Rapid Spanning Tree Protocol (RSTP)

One disadvantage of STP is that if there is a disruption or a device fails, the network needs to reconfigure itself: The devices start to negotiate new paths only when the interruption occurs. This can take up to 30 seconds. Fur this reason, STP was expanded to create the "Rapid Spanning Tree Protocol" (RSTP, IEEE 802.1w). This differs from STP essentially in that the devices are already collecting information about alternative routes during normal operation and do not need to gather this information after a disruption has occurred. This means that the reconfiguration time for an RSTP controlled network can be reduced to a few seconds.
This is achieved by using the following functions:

- Edge ports (end node port)
  Edge ports are ports connected to an end device.
  A port that is defined as an edge port is activated immediately after connection establishment. If a spanning tree BPDU is received at an edge port, the port loses its role as edge port and it takes part in (R)STP again. If no further BPDU is received after a certain time has elapsed (3 x hello time), the port returns to the edge port status.
- Point-to-point (direct communication between two neighboring devices)

  By directly linking the devices, a status change (reconfiguration of the ports) can be made without any delays.

- Alternate port (substitute for the root port)

  A substitute for the root port is configured. If the connection to the root bridge is lost, the device can establish a connection over the alternate port without any delay due to reconfiguration.

- Reaction to events

  Rapid spanning tree reacts to events, for example an aborted connection, without delay. There is no waiting for timers as in spanning tree.

- Counter for the maximum bridge hops
  The number of bridge hops a package is allowed to make before it automatically becomes invalid.

In principle, therefore with rapid spanning tree, alternatives for many parameters are preconfigured and certain properties of the network structure taken into account to reduce the reconfiguration time.

Multiple Spanning Tree Protocol (MSTP)

The Multiple Spanning Tree Protocol (MSTP) is a further development of the Rapid Spanning Tree Protocol. Among other things, it provides the option of operating several RSTP instances within different VLANs or VLAN groups and, for example, making paths available within the individual VLANs that the single Rapid Spanning Tree Protocol would globally block.

Common and Internal Spanning Tree (CIST)

CIST identifies the internal instance used by the switch that is comparable in principle with an internal RSTP instance.

Show information

Versions

This page shows the versions of the hardware and software of the device.

Note

The page is only available if there is an online connection to the device.

Displayed values

Table 1 contains the following columns:

- **Hardware**
  - Basic Device
    Shows the basic device.
  - WLAN X
    Shows the available WLAN interfaces. A device can have up to two WLAN interfaces.
- **Name**
  Shows the name of the device or module.
- **Revision**
  Shows the hardware version of the device.
- **Article number**
  Shows the article number of the device or described module. For the wireless card, only one version is then displayed if the WLAN interface is enabled.

Table 2 has the following columns:

- **Software**
  - Firmware
    Shows the current firmware version. If a new firmware file was downloaded and the device has not yet restarted, the firmware version of the downloaded firmware file is displayed here. After the next restart, the downloaded firmware is activated and used.
  - Bootloader
    Shows the version of the boot software stored on the device.
  - Firmware_Running
    Shows the firmware version currently being used on the device
- **Description**
  Shows the short description of the software.
- **Version**
  Shows the version number of the software version.
- **Date**
  Shows the date on which the software version was created.

I&M

Note

The page is only available if there is an online connection to the device.

This page contains information about device-specific vendor and maintenance data such as the article number, serial number, version number etc. You cannot configure anything on this page.

Description of the displayed values

The table has the following rows:

- **Manufacturer ID**
  Shows the manufacturer ID.
- Article number
  Shows the article number.
- **Serial number**
  Shows the serial number.
- **Hardware revision**
  Shows the hardware version.
- **Software version**
  Shows the software version.
- **Revision counter**
  Shows the revision counter: Counter for revisions since the initial commissioning
- **Revision Date**
  Date and time of the last revision
- **Function tag**
  Shows the function tag (plant designation) of the device. The plant designation (HID) is created during configuration of the device with HW Config of STEP 7.
- **Location tag**
  Shows the location tag of the device. The location identifier (LID) is created during configuration of the device with HW Config of STEP 7.
- **Date**
  Shows the date created during configuration of the device with HW Config of STEP 7.
- **Description**
  Shows the description created during configuration of the device with HW Config of STEP 7.

ARP / neighbor table

IPv6 Neighbor Table

Assignment of MAC address and IPv6 address

Via the IPv6 neighbor table, there is a unique assignment of MAC address to IPv6 address. This assignment is kept by each network node in its own separate neighbor table.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Displays the interface via which the row entry was learnt.
- **MAC Address**
  Shows the MAC address of the destination or source device.
- **IP Address**
  Shows the IPv6 address of the destination device.
- **Media Type**
  Shows the type of connection.
  - Dynamic
    The device recognized the address data automatically.
  - Static
    The addresses were entered as static addresses.

ARP table

Note

The page is only available if there is an online connection to the device.

Assignment of MAC address and IP address

With the Address Resolution Protocol (ARP), there is a unique assignment of MAC address to IPv4 addresses. This assignment is kept by each network node in its own separate ARP table. The page shows the ARP table of the device.

Displayed values

The table has the following columns:

- **Interface**
  Shows the interface via which the row entry was learnt.
- **MAC Address**
  Shows the MAC address of the destination or source device.
- **IP address**
  Shows the IP address of the target device.
- **Media Type**
  Shows the type of connection.
  - Dynamic
    The device recognized the address data automatically.
  - Static
    The addresses were entered as static addresses

Log tables

Event log

Logging events

Note

The page is only available if there is an online connection to the device.

The device allows you to log events that occur, some of which you can specify in " System > Events". This, for example, allows you to record when an authentication attempt failed or when the connection status of a port has changed. The content of the events log table is retained even when the device is turned off.

Settings

- **Severity Filter**
  You can filter the entries in the table according to severity. Select the required entries in the check boxes above the table.
  - Info
    Information
  - When this parameter is enabled, all entries of the category "Info" are displayed.
  - Warning
    Warning
    When this parameter is enabled, all entries of the category "Warning" are displayed.
  - Critical
    Critical
    When this parameter is enabled, all entries of the category "Critical" are displayed.

  The table has the following columns:

  - **Restart**
    Counts the number of restarts since you last reset to factory settings and shows the device restart after which the corresponding event occurred.
  - **System Up Time**
    Shows the time the device has been running since the last restart when the described event occurred.
  - **System Time**
    Shows the date and time when the described event occurred.
    If the system time is set, the time is also displayed at which the event occurred.
  - **Severity**
    Shows the severity of the message.
  - **Log Message**
    Displays a brief description of the event that has occurred. You will find the list of possible messages in Appendix D of the configuration manual.

Buttons

- **Delete**
  Click this button to delete the content of the event log file. All entries are deleted regardless of what you have selected under "Severity Filters".
  The display is also cleared. The restart counter is only reset after you have restored the device to the factory settings and restarted the device.

  Note

  The table can contain 400 entries for each severity. The number of entries in this table is restricted to 1200. When this number is reached, the oldest entries of the relevant severity are discarded. The table remains permanently in memory.

WLAN authentication log

Logging authentication attempts

This page shows a table with information on successful or failed authentication attempts.

Note

The page is only available if there is an online connection to the device.

Displayed values

- **Severity Filters**
  You can filter the entries in the table according to severity. To display all the entries, enable or disable all parameters.
  - Info
    Information
    When this parameter is enabled, all entries of the category "Info" are displayed.
  - Warning
    Warnings
    When this parameter is enabled, all entries of the category "Warning" are displayed.
  - Critical
    Critical
    When this parameter is enabled, all entries of the category "Critical" are displayed.

The table has the following columns:

- **Restart**
  Counts the number of restarts since you last reset to factory settings and shows the device restart after which the corresponding event occurred.
- **System Up Time**
  Shows the time the device has been running since the last restart when the described fault occurred.
- **System Time**
  Shows the time at which the described error occurred.

- **Severity**
  Shows the severity of the message.
- **Log Message**
  Displays a brief description of the event that has occurred. You will find the list of possible messages in Appendix D of the configuration manual.

If the system time is set, the time is also displayed at which the event occurred.

Buttons

- **Clear**
  Click this button to delete the content of the log file. All entries are deleted regardless of what you have selected under "Severity Filters".
  The display is also cleared. The restart counter is only reset after you have restored the device to the factory settings and restarted the device.

  Note

  The table can contain 400 entries for each severity. The number of entries in this table is restricted to 1200. When this number is reached, the oldest entries are discarded. The table remains permanently in memory.

- **Show all**
  Click this button to display all the entries on the WBM page. Note that displaying all messages can take some time.
- **Next**
  Click this button to go to the next page.
- **Prev**
  Click this button to go to the previous page.

Fault

Error status

This page displays any errors that occur. Errors of the "Cold/Warm Start" event can be deleted following confirmation. If there are no more unanswered error/fault messages, the fault LED goes off.

The time calculation always begins after the last system start. When the system is restarted, a new entry with the type of restart is created in the fault memory.

Note

The page is only available if there is an online connection to the device.

Displayed values

- **No. of Signaled Faults**
  Indicates how often the fault LED lit up and not how many faults occurred.
- **"Reset counters" button**
  The number is reset with this button.

The table contains the following columns:

- **Fault Time**
  Shows the time the device has been running since the last restart when the described fault occurred.
- **Fault Description**
  Display of the fault status for the device.
- **Clear Fault State**
  To delete files of the "Cold/warm start" event, click the "Clear Fault State" button.

Spanning Tree

Note

The page is only available if there is an online connection to the device.

The page shows the current information about the Spanning Tree and the settings of the root bridge.

- If spanning tree is turned off, only the basic information about this device is displayed.
- If Spanning Tree is turned on, the status information about the selected instance is displayed and the information about the configured ports is shown in the table. The information depends on the selected Spanning Tree mode.

Displayed values

- **Spanning Tree Mode**
  Shows the set mode. You specify the mode in "Layer 2 > MSTP > General".
  The following values are possible:
  - '-'
  - STP
  - RSTP
  - MSTP
- **Instance ID**
  Shows the number of the instance. The parameter depends on the configured mode.

- **Bridge Priority / Root Priority**
  Which device becomes the root bridge is decided based on the bridge priority. The bridge with the highest priority (in other words, with the lowest value for this parameter) becomes the root bridge. If several devices in a network have the same priority, the device whose MAC address has the lowest numeric value will become the root bridge. Both parameters, bridge priority and MAC address together form the bridge identifier. Since the root bridge manages all path changes, it should be located as centrally as possible due to the delay of the frames. The value for the bridge priority is a whole multiple of 4096 with a range of values from 0 to 32768.
- **Bridge Address / Root Address**
  The bridge address shows the MAC address of the device and the root address shows the MAC address of the root bridge.
- **Root Cost**
  Shows the path costs from the device to the root bridge.
- **Bridge Status**
  Shows the status of the bridge, e.g. whether or not the device is the root bridge
- **Regional root priority** (available only with MSTP)
  For a description, see Bridge Priority / Root Priority
- **Regional root address** (available only with MSTP)
  Shows the MAC address of the device.
- **Regional root costs** (available only with MSTP)
  Shows the path costs from the regional root bridge to the root bridge.

The table has the following columns:

- **Port**
  Shows the port via which the device communicates.
- **Role**
  Shows the status of the port. The following values are possible:
  - Disabled
    The port was removed manually from the spanning tree and will no longer be taken into account by the spanning tree.
  - Designated
    The ports leading away from the root bridge.
  - Alternate
    The port with an alternative route to a network segment
  - Backup
    If a switch has several ports to the same network segment, the "poorer" Port becomes the backup port.
  - Root
    The port that provides the best route to the root bridge.
  - Master
    This port points to a root bridge located outside the MST region.

- **Status**
  Displays the current status of the port. The values are only displayed. The parameter depends on the configured protocol. The following statuses are possible:
  - Discarding
    The port receives BPDU frames. Other incoming or outgoing frames are discarded.
  - Listening
    The port receives and sends BPDU frames. The port is involved in the spanning tree algorithm. Other outgoing and incoming frames are discarded.
  - Learning
    The port actively learns the topology; in other words, the node addresses. Other outgoing and incoming frames are discarded.
  - Forwarding
    Following the reconfiguration time, the port is active in the network. The port receives and sends data frames.
- **Oper. Version**
  Describes the type of spanning tree in which the port operates
- **Priority**
  If the path calculated by the spanning tree is possible over several ports of a device, the port with the highest priority (in other words the lowest value for this parameter) is selected. A value between 0 and 240 can be entered for the priority in steps of 16. If you enter a value that cannot be divided by 16, the value is automatically adapted.
- **Path costs**
  This parameter is used to calculate the path that will be selected. The path with the lowest value is selected. If several ports of a device have the same value, the port with the lowest port number will be selected.
  If the value "Cost Calc." is "0", the automatically calculated value is shown. Otherwise, the value of "Cost Calc." is displayed.
  The calculation of the path costs is largely based on the transmission speed. The higher the achievable transmission speed is, the lower the value of the path costs.
  Typical values for path costs with rapid spanning tree:
  - 10,000 Mbps = 2,000
  - 1000 Mbps = 20,000
  - 100 Mbps = 200,000
  - 10 Mbps = 2,000,000.

- **Edge type**
  Shows the type of the connection. The following values are possible:
  - Edge Port

    An edge port is connected to this port.
  - No Edge Port
    There is a spanning tree or rapid spanning tree device at this port.
- **P.t.P. Type**
  Shows the type of point-to-point link. The following values are possible:
  - P.t.P.
    With half duplex, a point-to-point link is assumed.
  - Shared Media

    With a full duplex connection, a point-to-point link is not assumed.

Note

Point-to-point connection means a direct connection between two devices.

A shared media connection is, for example, a connection to a hub.

Ethernet Statistics

Interface statistics

Interface statistics

The page shows the statistics from the interface table of the Management Information Base (MIB).

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **In Octet**
  Shows the number of received bytes.
- **Out Octet**
  Shows the number of sent bytes.
- **In Unicast**
  Shows the number of received unicast frames.

- **In Non Unicast**
  Shows the number of received frames that are not of the type unicast.
- **Out Unicast**
  Shows the number of sent unicast frames.
- **Out Non Unicast**
  Shows the number of sent frames that are not of the type unicast.
- **In Errors**
  Shows the number of all possible RX errors, refer to the tab "Packet Error".

Packet Size

Frames sorted by length

This page displays how many frames of which size were sent and received at each port.

Note

The page is only available if there is an online connection to the device.

The displayed values are transferred by RMON.

On the page "Layer 2 > RMON > Statistics", you can set the ports for which values will be displayed.

Displayed values

The table has the following columns:

- **Port**
  Shows the available ports and link aggregations. The port is made up of the module number and the port number, for example port 0.1 is module 0, port 1.

  Note
  Display of frame statistics

  In the statistics relating to frame lengths, note that both incoming and outgoing frames are counted.

- **Frame lengths**
  The other columns after the port number contain the absolute numbers of frames according to their frame length.
  The following frame lengths are distinguished:
  - 64 bytes
  - 65 - 127 bytes
  - 128 - 255 bytes
  - 256 - 511 bytes
  - 512 - 1023 bytes
  - 1024 - max.

Note
Data traffic on blocked ports

For technical reasons, data packets can be indicated on blocked ports.

Button

Reset Counter

Click "Reset Counter" to reset all counters. The counters are reset by a restart.

Frame type

Received frames sorted by type

This page displays how many frames of the type "unicast", "multicast", and "broadcast" were received at each port.

You cannot configure anything on this page.

The displayed values are transferred by RMON. On the page "Layer 2 > RMON > Statistics", you can set the ports for which values will be displayed.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Port**
  Shows the available ports and link aggregations. The port is made up of the module number and the port number, for example port 0.1 is module 0, port 1.
- **Unicast / Multicast / Broadcast**
  The other columns after the port number contain the absolute numbers of the incoming frames according to their frame type "unicast", "multicast" and "broadcast".

Button

Reset Counter

Click "Reset Counter" to reset all counters. The counters are reset by a restart.

Packet Errors

Received bad frames

This page shows how many bad frames were received per port.

You cannot configure anything on this page.

The displayed values are transferred by RMON. On the page "Layer 2 > RMON > Statistics", you can set the ports for which values will be displayed.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Port**
  Shows the available ports and link aggregations.
- **Error types**
  The other columns after the port number contain the absolute numbers of the incoming frames according to their error type.
  In the columns of the table, a distinction is made according to the following error types:
  - CRC

    Packets whose content does not match the CRC checksum.
  - Undersize
    Packets with a length less than 64 bytes.
  - Oversize
    Frames discarded because they were too long.
  - Fragments

    Packets with a length less than 64 bytes and a bad CRC checksum.
  - Jabbers
    VLAN-tagged packets with an incorrect CRC checksum that were discarded because they were too long.
  - Collisions
    Collisions that were detected.

Button

Reset Counter

Click "Reset Counter" to reset all counters. The counters are reset by a restart.

Learning table

Address filtering

This page shows the current content of the learning table. This table lists the source addresses of unicast address frames.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **VLAN-ID**
  Shows the VLAN ID of the node.

  Note

  This column appears in the table only if a VLAN is configured.

- **MAC Address**
  Shows the MAC address of the node.
- **Status**
  Shows the status of each address entry:
  - Dynamic
    The specified address was learned by receiving a frame from this node and will be deleted when the aging time expires if no further frames are received from this node.
  - Invalid
    These values are not evaluated.
- **Port**
  Shows the port via which the node with the specified address can be reached. Frames received by the device whose destination address matches this address will be forwarded to this port.

Buttons

- **Show all**
  Click this button to display all the entries on the page. Note that displaying all messages can take some time.
- **Next**
  Click this button to go to the next page.
- **Prev**
  Click this button to go to the previous page.

Drop-down list for page change

From the drop-down list, select the page you want to go to.

IPv6 routing table

Introduction

This page shows the IPv6 routes currently being used.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Destination Network**
  Shows the destination address of this route.
- **Prefix Length**
  Shows the prefix length of this route.
- **Gateway**
  Shows the gateway for this route.
- **Interface**
  Shows the interface for this route.
- **Metric**
  Shows the metric of the route. The higher value, the longer packets require to their destination.
- **Routing Protocol**
  Shows the routing protocol from which the entry in the routing table originates. The following entries are possible:
  - connected: Connected routes
  - Static: Static routes
  - RIPng: Routes via RIPng
  - OSPFv3: Routes via OSPFv3
  - other: Other routes

DHCP Server

This page shows whether IPv4 addresses were assigned to the devices by the DHCP server.

Note

The page is only available if there is an online connection to the device.

Description

- **IP Address**
  Shows the IPv4 address assigned to the DHCP client.
- **Pool ID**
  Shows the number of the IPv4 address band.
- **Identification method**
  Shows the method according to which the DHCP client is identified.
- **Identification value**
  Shows the MAC address or he client ID of the DHCP client.
- **Remote ID**
  Shows the remote ID of the DHCP client.

- **Circuit ID**
  Shows the circuit ID of the DHCP client.
- **Assignment method**
  Shows whether the IPv4 address was assigned statically or dynamically. You configure the static entries in "System > DHCP > Static Leases".
- **Assignment status**
  Shows the status of the assignment.
  - Assigned
    The assignment is used.
  - Not used
    The assignment is not used.
  - Probing
    The assignment is being checked.
  - Unknown
    The status of the assignment is unknown.
- **Expire Time**
  Shows how long the assigned IPv4 address is still valid. Once this period has elapsed, the DHCP client must either request a new IPv4 address or extend the lease time of the existing IPv4 address.

Security

Overview

This page shows the security settings and the local and external user accounts.

Note

The page is only available if there is an online connection to the device.

The values displayed depend on the rights of the logged-on user.

Displayed values

Services

The "Services" list shows the security settings.

- **Telnet Server**
  You configure the setting in "System > Configuration".
  - Enabled: Unencrypted access to the CLI.
  - Disabled: No unencrypted access to the CLI.
- **SSH Server**
  You configure the setting in "System > Configuration".
  - Enabled: Encrypted access to the CLI.
  - Disabled: No encrypted access to the CLI.
- **Web Server**
  You configure the setting in "System > Configuration".
  - HTTP/HTTPS: Access to the WBM is possible with HTTP and HTTPS.
  - HTTPS: Access to the WBM is now only possible with HTTPS.
- **SNMP**
  You can configure setting in "System > SNMP > General".
  - "-" (SNMP disabled)
    Access to device parameters via SNMP is not possible.
  - SNMPv1/v2c/v3
    Access to device parameters is possible with SNMP versions 1, 2c or 3.
  - SNMPv3
    Access to device parameters is possible only with SNMP version 3.
- **Management ACL**
  You configure the setting in "Security > Management ACL".
  - Disabled: No access restriction
    The access control is disabled.
  - Enabled: No access restriction
    The access control is enabled but no access rules have been defined.
  - Enabled: Restricted access only
    The access control is enabled and access rules have been defined.

- **Login Authentication**
  You configure the setting in "Security > AAA > General".
  - Local
    The authentication must be made locally on the device.
  - RADIUS
    The authentication must be handled via a RADIUS server.
  - Local and RADIUS
    The authentication is possible both with the users that exist on the device (user name and password) and via a RADIUS server.
    The user is first searched for in the local database. If the user does not exist there, a RADIUS query is sent.
  - RADIUS and fallback local
    The authentication must be handled via a RADIUS server.
    A local authentication is performed only when the RADIUS server cannot be reached in the network.
- **Password Policy**
  Shows which password policy is currently being used.

Local and external user accounts

You configure local user accounts and roles in "Security > User Accounts"

When you create a local user account an external user account is generated automatically. Local user accounts involve users each with a password for logging in on the device.

In the table "External User Accounts" a user is linked to a role, for example the user "Observer" with the role "user" In this case, the user is defined on a RADIUS server. The roll is defined locally on the device. When a RADIUS server authenticates a user, the corresponding group however is unknown or does not exist, the device checks whether or not there is an entry for the user in the table "External User Accounts". If an entry exists, the user is logged in with the rights of the associated role. If the corresponding group is known on the device, both tables are evaluated. The user is assigned the role with the higher rights.

Note

The table "External User Accounts" is only evaluated if you have set "Vendor Specific" in the RADIUS Authorization Mode".

With CLI you can access external user accounts.

The table "Local User Accounts" has the following columns:

- **User Account**
  Shows the name of the local user.
- **Role**
  Shows the role of the user. You can obtain more information on the function rights of the role in "Information > Security > Roles".

The table "External User Accounts" has the following columns:

- **User Account**
  Shows the name of the user on the RADIUS server.
- **Role**
  Shows the role assigned to the user on the device. You can obtain more information on this in "Information > Security > Roles".

Supported Function Rights

The page shows the function rights available locally on the device.

Note

The page is only available if there is an online connection to the device.

The values displayed depend on the role of the logged-on user.

Displayed values

- **Function Right**
  Shows the number of the function right. Different rights relating to the device parameters are assigned to the numbers.
- **Description**
  Shows the description of the function right.

Groups

This page shows which group is linked to which role. The group is defined on a RADIUS server. The roll is defined locally on the device.

Note

The page is only available if there is an online connection to the device.

The values displayed depend on the role of the logged-on user.

Displayed values

The table has the following columns:

- **Group**
  Shows the name of the group. The name matches the group on the RADIUS server.
- **Role**
  Shows the role. Users who are authenticated with the linked group on the RADIUS server receive the rights of this role locally on the device.
- **Description**
  Shows a a description for the link.

Roles

The page shows the roles valid locally on the device.

Note

The page is only available if there is an online connection to the device.

The values displayed depend on the role of the logged-on user.

Displayed values

This table contains the following columns:

- **Role**
  Shows the role.
- **Function Right**
  Shows the function rights of the role:
  - 1
    Users with this role can read device parameters but cannot change them.
  - 15
    Users with this role can both read and change device parameters.
  - 0
    This is a role that the device assigns internally when the user could not be authenticated. The user is denied access to the device.
- **Description**
  Shows a description of the role.

Inter AP blocking

the page shows a list of the devices with which the clients are allowed to communicate.

Note

The page is only available

- with SCALANCE W780 / W770
- if there is an online connection to the device.
- in access point mode.
- with KEY-PLUG inserted: W780 iFeatures (MLFB 6GK5 907-8PA00) or W700 Security (MLFB 6GK5907-0PA00)

Settings

The table has the following columns:

- **Radio**
  Shows the available WLAN interfaces to which the settings relate.
- **Port**
  Shows the VAP interface to which the settings relate.
- **MAC address**
  Shows the MAC address of the device with which the client may communicate.
- **IP address**
  Shows the IP address of the device with which the client may communicate.
- **IP address resolution**
  Shows the IP address with which the permitted IP address is resolved.

WLAN

AP overview

Overview of the configuration

This page shows these settings/properties of the WLAN or the WLAN interface.

Note

The page is only available

- if there is an online connection to the device.
- in access point mode.

Displayed values

Table 1 has the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **WLAN Mode**
  Shows the transmission standard. If DFS is activated, the transmission standard "802.11h" is not shown additionally but only the configured transmission standard "802.11a".
- **Configured Channel**
  Shows the configured channel. If "Auto" is displayed, the access point searches for a free channel itself.

- **Alternative DFS Channel**
  If the DFS function is enabled, the configured alternative channel of the access point is displayed.
  If "Auto" is displayed, the access point searches for an alternative channel itself.
  If the DFS function is activated and the access point browses for primary users for 60 seconds before starting communication with the selected channel, the text "scanning ..." is displayed instead of the channel.
- **Operational channel**
  Shows the channel of the access point via which the access point communicates.
- **HT Channel Width**
  Shows the channel bandwidth.
  - 20
    Channel bandwidth 20 MHz
  - 40 up
    Channel bandwidth 40 MHz. The configured channel and the neighboring channel above it are used.
  - 40 down
    Channel bandwidth 40 MHz. The configured channel and the neighboring channel below it are used.

  Note

  Channel bandwidth 40 MHz and frequency band 2.4 GHz

  If the access point detects another access point on the configured channel or on neighboring channels, the access point changes the channel bandwidth from 40 MHz to 20 MHz. If you set a "free" channel on the access point, the access point uses the channel bandwidth 40 MHz.

- **iFeatures**
  Shows which iFeatures are used.
  - -
    iFeatures are not used.
  - iPCF
  - iPCF-MC
  - iREF
- **Status**
  Shows the status of the WLAN interface.
  - enabled
    The WLAN interface is enabled.
  - disabled
    The WLAN interface is disabled.

Table 2 has the following columns:

- **Radio**
  Shows the available WLAN interfaces in this column.
- **Port**
  Shows the port of the virtual access point.
- **MAC Address**
  Shows the MAC address of the virtual access point.
- **SSID**
  Shows the SSID.
- **Security**
  Shows which authentication method is used.
  If the authentication method "Open System + Encryption" or "Shared Key" is used, the "Encrypted (WEP/AES)" authentication method is displayed for both.
- **Status**
  Shows the status of the WLAN interface.
  - enabled
    The WLAN interface is enabled.
  - disabled
    The WLAN interface is disabled.

Client list

Logged-on clients

The page the clients logged on with the access point as well as additional information, for example status, signal strength, MAC address.

Note

The page is only available

- if there is an online connection to the device.
- in access point mode.

Displayed values

- **Logged-on Clients**
  Shows the number of clients logged on to the access point.

The table has the following columns:

- **AID** (Associated ID)
  Shows the connection ID of the client. If the client connects to the access point via the VAP interface, the client is assigned a connection ID. The connection ID is unique within a VAP interface. If two clients log on at different VAP interfaces, both clients can receive the same ID.
- **Radio**
  Shows the available WLAN interfaces.
- **Port**
  Shows the VAP interface.
- **Type**
  Shows the client type, for example "Sta" stands for IEEE 802.11 standard client.
- **MAC Address**
  Shows the MAC address of the client.
- **System Name**
  Shows the system name of the client if the client communicates this to the access point. Not all clients support this parameter.
- **Channel**
  Shows the channel over which the client communicates with the access point.
- **Signal Strength [dBm]**
  Shows the signal strength of the connected client in bBm.
- **Signal Strength [%]**
  Shows the signal strength of the connected client as a percentage.
- **Age [s]**
  Shows the time that has elapsed since the last client activity.
- **Security**
  Shows which authentication method is used.
  If the authentication method "Open System + Encryption" or "Shared Key" is used, the "Encrypted (WEP/AES)" authentication method is displayed for both.
- **WLAN Mode**
  Shows the transmission standard. If DFS is activated, the transmission standard "802.11h" is not shown additionally but only the configured transmission standard "802.11a".
- **Maximum Data Rate [Mbps]**
  Shows the maximum data rate in megabits per second.
- **Status**
  Shows the current status of the connection, for example connected means that the client is connected to the AP and is ready to communicate with the AP.

WDS list

Communication between access points

In normal operation, the access point is used as an interface to a network and communicates with clients. There are, however, situations in which several access points need to communicate with each other, for example to extend wireless coverage or to set up a wireless backbone. This mode is possible with WDS (Wireless Distributed System).

Note

The page is only available

- if there is an online connection to the device.
- in access point mode.

This page shows information about the WDS connections of the access point.

Displayed values

The table has the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **Port**
  Shows the port.
- **BSSID**
  Shows the MAC address of the WDS partner.
- **WDS ID**
  Shows the name of the WDS partner.
- **Channel**
  Shows the channel over which the access point communicates with the WDS partner.
- **Signal Strength [dBm]**
  Shows the signal strength of the connected access point in bBm.
- **Signal Strength [%]**
  Shows the signal strength of the connected access point as a percentage.
- **Security**
  Shows which authentication method is used.
  If the authentication method "Open System + Encryption" or "Shared Key" is used, the "Encrypted (WEP/AES)" authentication method is displayed for both.
- **Max. Data Rate [Mbps]**
  Shows the maximum data rate in megabits per second.
- **Status**
  Shows the current status of the WDS connection.

AP overlap

Overlapping channels

Note

The page is only available

- if there is an online connection to the device.
- in access point mode.

For optimum data throughput, it is important that the set wireless channel is not used by other access points. In the 2.4 GHz band (802.11b or 802.11g), there is overlapping of the channels so that an access point occupies not only the set channel but also the two or three adjacent channels. You should therefore make sure that there is adequate channel spacing to neighboring access points.

This page shows all access points that are visible on the set or adjacent channels (at 2.4 GHz). If entries exist here, the maximum data throughput of the access point and the availability of the communication link to the access point is potentially impaired.

Displayed values

Table 1 has the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **Aging Time [min]**
  Specify the life time of the entries in the list. If an access point is inactive for longer than the set time, it is removed from the list.

  Note

  Changing the aging time

  The aging time is a WLAN setting. For this reason, if a change is made, the WLAN connection is briefly interrupted to accept the new value.

The table has the following columns:

- **Radio**
  Shows the available WLAN interfaces in this column.
- **Type**
  Shows the mode of the WLAN interface.
- **SSID**
  Shows the SSID of the access point.
- **BSSID**
  Shows the MAC address of the access point.
- **System Name**
  Shows the system name of the device. The entry depends on the access point. Not all access points support this parameter.

- **Channel**
  Shows the channel over which the client communicates with the access point.
- **Signal Strength [dBm]**
  Shows the signal strength of the client in bBm.
- **Signal Strength [%]**
  Shows the signal strength of the client as a percentage.
- **Age [s]**
  Shows the time that has elapsed since the last access point activity.
- **Security**
  Shows which authentication method is used.
  If the authentication method "Open System + Encryption" or "Shared Key" is used, the "Encrypted (WEP/AES)" authentication method is displayed for both.
- **WLAN Mode**
  Shows the transmission standard. If DFS is activated, the transmission standard "802.11h" is not shown additionally but only the configured transmission standard "802.11a" or "802.11n".

Client overview

Overview of the configuration

The page shows an overview of the existing clients and their configuration.

Note

The page is only available

- if there is an online connection to the device.
- for clients or access points in client mode.

Displayed values

- **Radio**
  Shows the available WLAN interfaces in this column.
- **WLAN Mode**
  Shows the transmission standard.

- **MAC Mode**
  Shows how the MAC address is assigned to the interface.
  - Automatic
    The client automatically adopts the source MAC address of the first frame that it receives over the Ethernet interface.
  - Manual
    The address was entered manually
  - Own
    The client uses the MAC address of the Ethernet interface for the WLAN interface.
  - Layer 2 Tunnel
    The client uses the MAC address of the Ethernet interface for the WLAN interface. The network is also informed of the MAC addresses connected to the Ethernet interface of the client. Up to eight MAC addresses can be used.
- **MAC Address**
  Shows the MAC address of the WLAN interface.
- **Operational channel**
  Shows the channel of the access point with which the client is connected.
- **HT Channel Width [MHz]**
  Shows the channel bandwidth.
  - 20
    Channel bandwidth 20 MHz
  - 40 up
    Channel bandwidth 40 MHz. The configured channel and the neighboring channel above it are used.
  - 40 down
    Channel bandwidth 40 MHz. The configured channel and the neighboring channel below it are used.

  Note

  Channel bandwidth 40 MHz and frequency band 2.4 GHz

  If the access point detects another access point on the configured channel or on neighboring channels, the access point changes the channel bandwidth from 40 MHz to 20 MHz. If you set a "free" channel on the access point, the access point uses the channel bandwidth 40 MHz.
- **Connected BSSID**
  This box shows the MAC address of the access point with which the client is connected.
- **Connected SSID**
  This box shows the SSID of the access point with which the client is connected.
- **Security**
  Shows which authentication method is used.
  If the authentication method "Open System + Encryption" or "Shared Key" is used, the "Encrypted (WEP/AES)" authentication method is displayed for both.

- **iFeatures**
  Shows which iFeatures are used.
  - -
    iFeatures are not used.
  - iPCF
  - iPCF-MC
- **Status**
  Shows the status of the WLAN interface.
  - enabled
    The WLAN interface is enabled.
  - disabled
    The WLAN interface is disabled.

Available APs

Available access points

Note

The page is only available

- if there is an online connection to the device.
- for clients or access points in client mode.

This page shows all the access points visible to the client. The list also includes the access points to which the client cannot connect due to its configuration.

Note

Display when iPCF mode is activated

If the iPCF mode is active, the display is different. Since the client does not run a background scan in this case, only the access point with which the client is currently connected is displayed.

Displayed values

The table has the following columns:

- **Radio**
  Shows the WLAN interface visible to the access point.
- **SSID**
  Shows the SSID of the access point.
- **BSSID**
  Shows the MAC address of the access point.

- **System Name**
  Shows the system name of the access point. The entry depends on the access point. Not all access points support this parameter.
- **Channel**
  Shows the channel on which the access point transmits or communicates.
- **Signal Strength [dBm]**
  Shows the signal strength of the access point in bBm.
- **Signal Strength [%]**
  Shows the signal strength of the access point as a percentage.
- **Type**
  Shows the mode of the WLAN interface.
- **Security**
  Shows which authentication method is used.
  If the authentication method "Open System + Encryption" or "Shared Key" is used, the "Encrypted (WEP/AES)" authentication method is displayed for both.
- **WLAN Mode**
  Shows the transmission standard. If DFS is activated, the transmission standard "802.11h" is not shown additionally but only the configured transmission standard "802.11a" or "802.11n".
- **Status**
  Shows the status of the access point, for example whether or not the access point is available.

Force Roaming

Note

The page is only available

- if there is an online connection to the device.
- in access point mode.

The device monitors the connection to IP addresses cyclically. To achieve this, the device sends echo messages (pings) to the configured destination IP address at regular intervals.

This WBM page shows the current status of the connection. It also shows whether there is roaming.

Displayed values

The table has the following columns:

- **Port**
  Shows the available VAP interfaces.
- **Destination Address / State**
  Shows which destination address is monitored and the status of the connection. You configure the destination address in "Interfaces > WLAN > Force Roaming".
  - not configured: No destination address is configured.
  - idle: The configuration is incomplete.
  - up:The destination address is reachable.
  - down: The destination address is unreachable.
- **Force Roaming on IP down**
  Indicates whether roaming is currently being performed.
  - inactive: No roaming is being performed.
  - active: Roaming is being performed. None of the destination addresses is reachable.

IP mapping table

WLAN access for several devices via a client

Note

The page is only available

- if there is an online connection to the device.
- for clients or access points in client mode.

You can make WLAN access available for several devices with one client if you use IP mapping. This means that you do not need to equip every device with its own WLAN client. This is possible only if the connected devices are addressed only by IP frames. Communication at MAC address level (ISO/OSI layer 2) can

- be established with one component whose MAC address is configured on the client,
- be established with a maximum of eight components if the "Layer 2 Tunnel" MAC mode is selected.

The "Layer 2 Tunnel" setting meets the requirements of industrial applications in which MAC address-based communication takes place with several devices downstream from the client. Clients with this setting cannot connect to standard Wi-Fi devices and access points with firmware V3.0 or older.

The client maintains a table with the assignment of MAC address and IP address to be able to send incoming IP frames to the correct MAC address. The page shows the IP mapping table of the device.

Note

Display of the IP mapping table

If "Layer 2 Tunnel" is configured on the device for the MAC mode, the IP mapping table is not displayed.

Displayed values

- **MAC address**
  The MAC address of a device located downstream from the WLAN client from the perspective of the access point.
- **IP address**
  The IP address managed for this device by the WLAN client.
- **Type**
  There are two options for the type:
  - system
    The information relates to the WLAN client itself.
  - learned
    The information relates to a device downstream from the WLAN client.

MAC mode

Frames sent by the client to the access point always have the MAC address of the WLAN client as the source MAC address. In the "learning table" of the access point there is therefore only the MAC address of the WLAN client.

If there are further SCALANCE W700 devices downstream from the client, the "Automatic" option should not be enabled. In this case, the MAC address would be assigned indiscriminately to the first SCALANCE W700 device that signals over Ethernet. If there is only IP communication between the access point and the client, the default setting "Own" can be retained. If MAC address-based frames are also to be sent by SCALANCE W700 devices downstream from the client, you need to select the settings "Automatic", "Manual" or "Layer 2 Tunnel".

Background noise

The page displays the background noise of the channel.

Note

The page is only available if there is an online connection to the device.

Description

- **Connector**
  Shows the name of the relevant antenna connector.
- **Channel [dBm]**
  Shows the background noise of the set channel.
- **Extended Channel [dBm]**
  Shows the background noise of the extended channel (HT-40).

WLAN statistics

Fault

The page show how many bad frames were received or sent per WLAN interface. If an increased number of errors occurs, you should check the settings for the WLAN interface(s), the setup of the devices and the connection quality.

Note

The page is only available if there is an online connection to the device.

Displayed values

The Transmission Errors table has the following columns:

- **Interface**
  Shows the WLAN interface to which the entries apply.
- **Error types**
  The other columns after the WLAN interface contain the absolute numbers of the frames sent according to their error type.
  The columns of the table distinguish the following error types:
  - Transmission errors
    Shows the number and percentage of bad frames that were sent.
  - Discarded frames
    Shows the number and percentage of frames that were discarded. Despite all the retries, the frame could not be successfully sent.
    The frame has not yet been sent and the recipient has logged off in the meantime.
  - Retries
    Shows the number and percentage of frames sent successfully that required one or more retries.

The "Received Errors" table has the following columns:

- **Interface**
  Shows the WLAN interface to which the entries apply.
- **Error types**
  The other columns after the WLAN interface contain the absolute numbers of the frames received according to their error type.
  The columns of the table distinguish the following error types:
  - Received errors
    Shows the number and percentage of bad frames that were received.
  - Duplicate frames
    Shows the number and percentage of frames that were received twice.
  - Encryption errors
    Shows the number and percentage of incorrectly encrypted frames.
  - FCS errors
    Shows the number and percentage of frames in which the checksum was incorrect.

Sent management frames

The page shows how many frames in response to logging on or logging off were counted per VAP interface.

Note

The page is only available

- if there is an online connection to the device.
- in access point mode.

Displayed values

The table has the following columns:

- Interface
  Shows the VAP interface to which the entries apply.
- Frames
  - Management Frames
    Shows the number of management frames
  - Association Requests
    Shows the number of requesting association frames relevant for a logon.
  - Association Responses
    Shows the number of responding association frames relevant for a logon.
  - Disassociation Requests
    Shows the number of requesting disassociation frames relevant for a logoff.
  - Authentication Requests
    Shows the number of requesting Authentication frames relevant for a logon.
  - Authentication Responses
    Shows the number of responding authentication frames relevant for a logon.
  - Deauthentication Requests
    Shows the number of requesting deauthentication frames relevant for a logon.

Management Frames Received

The page shows how many frames in response to logging on or logging off were counted per VAP interface.

Note

The page is only available

- if there is an online connection to the device.
- in access point mode.

Displayed values

The table has the following columns:

- **Interface**
  Shows the VAP interface to which the entries apply.
- **Frame**
  - Management Frames
    Shows the number of management frames
  - Association Requests
    Shows the number of requesting association frames relevant for a logon.
  - Association Responses
    Shows the number of responding association frames relevant for a logon.
  - Disassociation Requests
    Shows the number of requesting disassociation frames relevant for a logoff.
  - Authentication Requests
    Shows the number of requesting Authentication frames relevant for a logon.
  - Authentication Responses
    Shows the number of responding authentication frames relevant for a logon.
  - Deauthentication Requests
    Shows the number of requesting deauthentication frames relevant for a logon.

Sent data frames

The page shows how many frames were sent per VAP interface.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Shows the VAP interface to which the entries apply.
- **Frame types**
  The other columns after the VAP interface contain the absolute numbers of the sent frames according to the frame types.
  In the columns of the table, a distinction is made according to the following frame types:
  - Data Frames
    Shows the number of sent data frames.
  - Multicast/Broadcast Frames
    Shows the number of sent multicast and broadcast frames.
  - Unicast Frames
    Shows the number of unicast frames sent.
  - Average Data Rate
    Shows average data rate of the last data frames sent.

Received data frames

The page shows how many frames were received per VAP interface.

Note

The page is only available if there is an online connection to the device.

Displayed values

The table has the following columns:

- **Interface**
  Shows the VAP interface to which the entries apply.
- **Frame types**
  The other columns after the VAP interface contain the absolute numbers of the received frames according to the frame types.
  In the columns of the table, a distinction is made according to the following frame types:
  - Data Frames
    Number of received data frames.
  - Multicast/Broadcast Frames
    Shows the number of received multicast and broadcast frames.
  - Unicast Frames
    Shows the number of received unicast frames.
  - Average Data Rate
    Shows average data rate of the last data frames received.

WLAN iFeatures

iREF Client List

The page shows the antenna connector via which the clients logged on to the access point communicate. Other information such as the signal strength and the MAC address of the WLAN interface is also shown.

Note

The page is only available

- for SCALANCE W780
- for SCALANCE W780
- if there is an online connection to the device.
- in access point mode.
- with KEY-PLUG inserted: W780 iFeatures (MLFB 6GK5 907-8PA00)

Displayed values

- **Associates Stations**
  Shows the number of clients logged in to the access point.

The table has the following columns:

- **AID** (Associated ID)
  Shows the connection ID of the client. If the client connects to the access point via the VAP interface, the client is assigned a connection ID. The connection ID is unique within a VAP interface. If two clients log on at different VAP interfaces, both clients can receive the same ID.
- **Radio**
  Shows the available WLAN interfaces.
- **Port**
  Shows the VAP interface.
- **MAC Address**
  Shows the MAC address of the client.
- **System Name**
  Shows the system name of the client if the client communicates this to the access point. Not all clients support this parameter.
- **TX Chain**
  Shows the antenna connector over which the client communicates with the access point.
- **Signal Strength [dBm]**
  Shows the signal strength of the connected client in bBm.
- **Signal Strength [%]**
  Shows the signal strength of the connected client as a percentage.
- **Age [s]**
  Shows the age of the listed client.

iREF WDS List

The page shows the access points logged on to the access point via a WDS link. Information such as the antenna used and the signal strength of the WLAN interface is shown.

Note

The page is only available

- SCALANCE W780
- SCALANCE W770
- if there is an online connection to the device.
- in access point mode.
- with KEY-PLUG inserted: Access point W780 iFeatures (MLFB 6GK5 907-8PA00)

Displayed values

- **Connected WDS partners**
  Shows the number of access points logged in to the access point

The table has the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **Port**
  Shows the WDS interface.
- **BSSID**
  Shows the MAC address of the WDS partner.
- **WDS ID**
  Shows the name of the WDS partner.
- **TX Chain"**
  Shows the antenna connector over which the two access points communicate with each other.
- **Signal Strength [dBm]**
  Shows the signal strength of the connected access point in bBm.
- **Signal Strength [%]**
  Shows the signal strength of the connected access point as a percentage.

AeroScout

Note

The page is only available

- for SCALANCE W780 and SCALANCE W774-1 RJ-45
- if there is an online connection to the device.
- in access point mode.
- with KEY-PLUG inserted: Access points W780 iFeatures (MLFB 6GK5 907-8PA00)

Note

AeroScout and iPCF mode

The AeroScout function cannot be combined with other iFeatures (iPCF, iPCF-MC iREF). AeroScout can only be used in the 2.4 GHz band according to IEEE 802.11g, IEEE 802.11n and IEEE 802.11n-only.
For more detailed information, please refer to the AeroScout documentation of (www.aeroscout.com).

Displayed values

- **Tag Information forwarding**
  Shows the setting of the management program.
  In the management program that evaluates the AeroScout frames, you can specify whether or not an IWLAN device will forward frames.

  Note

  No evaluation or processing

  With a suitable configuration, the SCALANCE W device forwards AeroScout frames but does not process or evaluate them itself. This is done only in the "AeroScout System Manager" program.

- **AeroScout Status**
  Shows whether or not AeroScout is activated. You can set this for each WLAN interface of the device in "iFeatures > AeroScout". If you have enabled forwarding, "active" is displayed here, otherwise "inactive"
- **Engine Port**
  Shows the port at which the device expects UDP datagrams from the AeroScout Engine.
- **Response Port**
  Shows the port via which the device forwards the received AeroScout frames.
- **Response IP**
  Shows the IP address of the PC on which the AeroScout Engine for evaluation of the AeroScout frames is running.
- **Multicast Address**
  Shows the multicast address that the tag uses. The multicast address is configured in the AeroScout Engine.

- **Acknowledgments Sent**
  Shows the number of acknowledgments sent by the device. The number reflects the received UDP datagrams.
- **Messages Dropped**
  Shows the number of AeroScout frames that were discarded by the device. If, for example, an AeroScout tag is configured so that it sends on channel 1, the device does not forward a frame received on channel 6.

Configuring system functions

Configuration

Specify the services that access the device. With some services, there are further configuration pages on which more detailed settings can be made.

Settings

The page contains the configuration overview of the access options of the device.

- **Telnet Server**
  Enable or disable the "Telnet Server" service for unencrypted access to the CLI.
- **SSH server**
  Enable or disable the "SSH Server" service for encrypted access to the CLI.
- **HTTPS Server only**
  Enable or disable access using HTTP.
- **DNS client**
  Enable or disable the DNS client. You can configure other settings in "System > DNS Client".
- **SMTP Client**
  Enable or disable the SMTP client. You can configure other settings in "System > SMTP Client".
- **Syslog Client**
  Enable or disable the system event client. You can configure other settings in "System > Syslog Client".
- **DCP Server**
  Specify whether or not the device can be accessed with DCP (Discovery and Configuration Protocol):
  - "-" (disabled)
    DCP is disabled. Device parameters can neither be read nor modified.
  - Read/write access
    With DCP, device parameters can be both read and modified.
  - Read-Only
    With DCP, device parameters can be read but cannot be modified.

- **Time**
  Select the required setting. The following settings are possible:
  - Manual
    The system time is set manually.
  - SIMATIC Time
    The system time is set using a SIMATIC time transmitter. You can configure other settings in "System > System Time > SIMATIC Time Client".
  - SNTP Client
    The system time is set via an SNTP server. You can configure other settings in "System > System time > SNTP client".
  - NTP Client
    The system time is set via an NTP server. You can configure other settings in "System > System Time > NTP Client".
- **SNMP**
  Select the required protocol. The following settings are possible:
  - "-" (SNMP disabled)
    Access to device parameters using SNMP is not possible.
  - SNMPv1/v2c/v3
    Access to device parameters is possible with SNMP versions 1, 2c or 3. You can configure other settings in "System > SNMP > General".
  - SNMPv3
    Access to device parameters is possible with SNMP version 3. You can configure other settings in "System > SNMP > General".
- **SNMPv1/v2 Read-Only**
  Enable or disable write access to SNMP variables with SNMPv1/v2c.
- **SNMPv1 Traps**
  Enable or disable the sending of SNMP traps (alarm frames). You can configure other settings in "System > SNMP > Traps".
- **DHCP Client**
  Enable or disable the DHCP client. You can configure other settings in "System > DHCP client".
- **DHCPv6 Client**
  Enable or disable the DHCPv6 client.

- **SINEMA configuration interface**
  If the SINEMA configuration interface is enabled, you can download configurations to the device via the TIA Portal.
- **Configuration Mode**
  Select the required mode. The following modes are possible:
  - Automatic Save
    Automatic backup mode. Approximately 1 minute after the last parameter change or when you restart the device, the configuration is automatically saved. In addition to this, the following message appears in the display area "Changes will be saved automatically in x seconds. Press 'Write Startup Config' to save immediately".

    Note

    Interrupting the save

    Saving starts only after the timer in the message has elapsed. How long saving takes depends on the device.

    During the save, the message "Saving configuration data in progress. Please do not switch off the device" is displayed.

    - Do not switch off the device immediately after the timer has elapsed.
  - Trial
    Trial mode. In Trial mode, although changes are adopted, they are not saved in the configuration file (startup configuration).
    To save changes in the configuration file, use the "Write Startup Config" button. The "Write Startup Config" button is displayed when you set trial mode. In addition to this after every parameter change the following message is displayed in the display area: "Trial Mode Active – Press "Write Startup Config" button to make your settings persistent" as soon as there are unsaved modifications. This message can be seen on every WBM page until the changes made have either been saved or the device has been restarted.

General

Device

This page contains the general device information.

Settings

- **Current System Time**(Only available online)
  Shows the current system time. The system time is either set by the user or by a time-of-day frame: either SINEC H1 time-of-day frame, NTP or SNTP. (readonly)
- **System Up Time**(only available online)
  Shows the operating time of the device since the last restart. (readonly)
- **Device Type**(only available online)
  Shows the type designation of the device. (readonly)

- **System Name**
  You can enter the name of the device. The entered name is displayed in the selection area. A maximum of 255 characters are possible. The system name is also displayed in the CLI input prompt. The number of characters in the CLI input prompt is limited. The system name is truncated after 16 characters.
- **System Contact**
  You can enter the name of a contact person responsible for managing the device.
- **Location**
  You can enter the location of the device. The entered installation location is displayed in the selection area.

Note

The ASCII characters 0x20 to 0x7e are used in the input boxes.

At the start and end of the input boxes"System name", "System Contact" and "System Location", the characters "<", ">" and "space" are not permitted.

Coordinates

On this page, you configure the geographic coordinates (latitude, longitude and the height above the ellipsoid according to WGS84). These boxes are purely for information with a maximum length of 32 characters.

Getting the coordinates

Use suitable maps for obtaining the geographic coordinates of the device.

The geographic coordinates can also be obtained using a GPS receiver. The geographic coordinates of these devices are normally displayed directly and only need to be entered in the input boxes of this page.

Settings

- **Latitude**
  Enter the north or south latitude for the location of the device.
  For example, +49° 1´ 31.67" means that the device is located at 49 degrees, 1 arc minute and 31.67 arc seconds north latitude.
  A south latitude is shown by a preceding minus character.
  You can also append the letters N (north latitude) or S (south latitude) to the numeric information (49° 1´ 31.67" N).
- **Longitude**
  Enter the value of the eastern or western longitude of the location of the device.
  For example, +8° 20´ 58.73" means that the device is located at 8 degrees, 20 minutes and 58.73 seconds east.
  A western longitude is indicated by a preceding minus sign.
  You can also add the letter E (eastern longitude) or W (western longitude) to the numeric information (8° 20´ 58.73" E).
- **Height**
  Enter the value of the height above sea level in meters.
  For example, 158 m means that the device is located at a height of 158 m above sea level.
  Heights below sea level (for example the Dead Sea) are indicated by a preceding minus sign.

Agent IPv4

Configuration of the IP addresses

On this page, you configure the IPv4 address for the device.

Settings

- **IP Assign Method**
  Shows how the IPv4 address is assigned.
  - Static
    The IPv4 address is static. You enter the IP settings in "IP Address" and "Subnet Mask".
  - Dynamic (DHCP)
    The device obtains a dynamic IPv4 address from a DHCP server.
- **IP Address**
  Enter the IP address of the device. If you change the IPv4 address, the Web browser should automatically set itself to the new address. If this does not happen, enter the new address in the Web browser manually.
- **Subnet Mask**
  Enter the subnet mask of the device.
- **Default gateway**
  Enter the IPv4 address of the default gateway to be able to communicate with devices in another subnet, for example diagnostics stations, e-mail server.

- **Agent VLAN ID**
  Select the VLAN ID from the drop-down list. The drop-down list is only available if "802.1 Q VLAN Bridge" is set for the "Base Bridge Mode" parameter. You configure the parameter in "Layer 2 > VLAN > General". You can only select VLANs that have already been configured.

  Note
  Changing the agent VLAN ID

  If the configuration PC is connected directly to the device via Ethernet and you change the agent VLAN ID, the device is no longer reachable via Ethernet following the change.

- **MAC Address** (only available online)
  Shows the MAC address of the device. The MAC address is linked to the hardware and cannot be modified.

Agent IPv6

Agent IPv6

Configuration of the IP addresses

On this page, enable IPv6 on the management VAN. This VLAN interface is also called an IPv6 interface. An IPv6 interface can have several IPv6 addresses.

Description

The page contains the following:

- **Interface**
  Shows the VLAN interface on which IPv6 will be enabled.
- **IPv6 Enable**
  Enable or disable IPv6 on the interface. When you enable the setting and accept it, the link local address is created automatically.
- **IPv6 Address**
  Enter the IPv6 address. The entry depends on the selected address type.
- **Prefix Length**
  Enter the number of left-hand bits belonging to the prefix
- **IPv6 Address Type**
  Select the address type:
  - Unicast
  - Link Local: IPv6 address is only valid on the link.

- **Address Configuration**
  Specify the mechanism for the address configuration:
  - Automatic (default)
    The IPv6 address is created using a stateless mechanism or a stateful mechanism.
  - DHCPv6
    Stateful: Obtains the IPv6 address and the configuration file from the DHCPv6 server.
  - SLAAC (Stateless Address Auto Configuration)
    Stateless autoconfiguration using NDP (Neighbor Discovery Protocol)
  - Static
    Enter a static IPv6 address.
- **Rapid Commit**
  When enabled the procedure for the IPv6 address assignment is shortened. Instead of 4 DHCPv6 messages (SOLICIT, ADVERTISE , REQUEST, REPLY) only 2 DHCPv6 messages (SOLICIT, REPLY) are used. You will find further information on the messages in RFC 3315.

The table has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Interface Name**
  Shows the name of the VLAN interface.
- **IPv6 Address**
  Shows the IPv6 address.
- **Prefix Length**
  Shows the prefix length.
- **IPv6 Address Type**
  Displays the address type. The following values are possible:
  - Unicast
  - Link Local

IPv6 default routes

On this page, you configure the default IPv6 routes.

Description

The page contains the following:

- **Destination Network**
  Enter the network address of the destination that can be reached via this route.
- **Prefix Length**
  Enter the number of left-hand bits belonging to the prefix
- **Gateway**
  Enter the IPv4 address of the gateway via which this network address is reachable.

- **Metric**
  Enter the metric for the route. The metric corresponds to the quality of a connection, based for example on speed or costs. If there are several equal routes, the route with the lowest metric value is used.
  Range of values: 1 - 254
- **Interface**
  Specify the interface via which the network address of the destination is reached.

This table contains the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Destination Network**
  Shows the network address of the destination.
- **Prefix Length**
  Shows the prefix length.
- **Gateway**
  Shows the IPv6 address of the next gateway.
- **Interface**
  Shows the Interface of the route.
- **Metric**
  Enter the metric for the route. When creating the route, "not used" is entered automatically. The metric corresponds to the quality of a connection, based for example on speed or costs. If there are several equal routes, the route with the lowest metric value is used.
  Range of values: 1 - 254
- **Status**
  Shows whether or not the route is active.

DNS client

The DNS server (Domain Name System) assigns a domain name to an IP address so that a device can be uniquely identified.

If this setting is enabled, the device can communicate with a DNS server as a DNS client.

Note

Only resource records (RR) of type A (IPv4 address of a host) are supported.

Settings

- **DNS client**
  Enable or disable depending on whether the device should operate as a DNS client.
- **DNS server used**
  - learned only
    The device uses only the DNS servers assigned by DHCP.
  - manual only
    The device uses only the manually configured DNS servers. The DNS servers must be connected to the Internet.
  - all
    The device uses all available DNS servers.
- **DNS Server Address**
  Enter the IP address of the DNS server.

The table has the following columns:

- **DNS Server Address**
  Shows the IP address of the DNS server.
- **Origin**
  This shows whether the DNS server was configured manually or was assigned by DHCP.

Restart

On this page, there is a button with which you can restart the device and various options for resetting to the device defaults.

Note

The page is only available if there is an online connection to the device.

Note

Note the following points about restarting a device:

- You can only restart the device with administrator privileges.
- A device should only be restarted with the buttons of this page or with the appropriate CLI commands and not by a power cycle on the device.
- Any modifications you have made only become active on the device after clicking the "Set Values" button on the relevant page. If the device is in "Trial Mode", configuration modifications must be saved manually before a restart. In "Automatic Save" mode, the last changes are saved automatically before a restart.

Settings

The following options are available for restarting:

- **Restart**
  Click this button to restart the system. You must confirm the restart in a dialog box. During a restart, the device is reinitialized, the internal firmware is reloaded, and the device runs a self-test. The learned entries in the address table are deleted. You can leave the browser window open while the device restarts. You then need to log in again.
- **Restore Memory Defaults and Restart**
  Click on this button to restore the factory configuration settings with the exception of the following parameters and to restart:
  - IP addresses
  - Subnet mask
  - IP address of the default gateway
  - DHCP client ID
  - DHCP
  - System name
  - System location
  - System contact
  - Mode of the device
- **Restore Factory Defaults and Restart**
  Click on this button to restore the factory configuration settings. The protected defaults are also reset.
  An automatic restart is triggered.

Note

By resetting all the defaults to the factory configuration settings, the IP address is also lost. Following this, the device can only be accessed using the Primary Setup Tool or using DHCP.

With the appropriate attachment, a previously correctly configured device can cause circulating frames and therefore the failure of the data traffic.

Commit Control

Description

Note

The page is only available if there is an online connection to the device.

On this page, you specify when the WLAN settings become effective on the device.
If you change a WLAN setting and confirm the change, this change is adopted and takes effect

immediately. To do this, the WLAN connection is briefly interrupted. This means that you can lose the WLAN connection to your device before it is fully configured.

With the "Manual Commit" setting, you have the opportunity of first fully configuring the device. The changes are accepted, but are not active immediately. The changes only take effect when you confirm the changes with the "Set Values" button.

Note

If you configure the device via the WLAN interface, we recommend that you use the "Manual Commit" setting. Check the parameters again before you confirm the changes with the "Set Values" button.

Settings

- **Commit Mode**
  Select the required setting.
  - Automatic Commit
    Each change in the WLAN settings is adopted and is immediately effective when you click the "Set Values" button. With its default setting, the SCALANCE W700 device is set to "Automatic Commit".
  - Manual Commit
    The changes are accepted, but do not take effect immediately. The changes only take effect when you click the "Set Values" button. The "Commit Changes" button is displayed when you set "Manual Commit". In addition to this the message “Manual Commit Mode active - Click the "Commit Changes" button to to provide current configuration to driver” appears as soon as there are WLAN changes. This message can be seen on every WBM page until the changes made have either taken effect or the device has been restarted.

    Note

    When the changes take effect, the WLAN connection to all WLAN interfaces will be interrupted for a short time. The WLAN driver is started with the new settings.

Load & save

Overview of the file types

Table 10-106 HTTP

File type	Description	Down-load	Save	Delete
Config	Start configuration	X	X	--
ConfigPack	Detailed configuration information. for example, start configuration, users, certificates, firmware of the device (if saved as well). For more detailed information on creating and using the ConfigPack incl. firmware, refer to the section "Maintenance".	X	X	--
CountryList	The zip file contains the country list as a csv and as a pdf file.	--	X	--
Debug	This file contains information for Siemens Support.	--	X	X
EDS	Electronic Data Sheet (EDS) Electronic data sheets for describing devices in the EtherNet/IP mode	--	X	--
Firmware	Loading firmware updates	X	X	--
GSDML	Information on the device properties (PROFINET)	--	X	--
HTTPS Cert	HTTPS certificate Maximum file size: 8192 bits	X	X	X
LogFile	File with entries from the event log table	--	X	--
MIB	Private MSPS MIB file "Scalance_w_msps.mib"	--	X	--
RunningCLI	This file contains an overview of the current configuration in the form of CLI commands. You can download the text file. The file is not intended to be uploaded again unchanged.	--	X	--
Script	CLI script file	X	--	--
StartupInfo	Startup log file	--	X	--
Users	File with user names and passwords	X	X	--
WLANAuth-log	File with entries from the WLAN Authentication Log (information on successful or failed authentication attempts)	--	X	--
WLANCert (in client mode only)	User certificate. You can specify a password for the user certificate on the WBM page "Load&Save > Password". Maximum file size: 8192 bits	X	X	X
WLANServ-Cert (in client mode only)	Server certificate Maximum file size: 8192 bits	X	X	X

File type	Description	Down-load	Save	Delete
WLANSi-gRec (in client mode only)	The zip file contains the following: • csv file with the measured values of the signal recorder • pdf file with the measured values and an additional graphic representation of the measured values. You will find information about the measured values and their graphic representation in the section "AU-TOHOTSPOT".	--	X	X
WLANSpec-trumAnalyz-er (Only in ac-cess point mode)	The Zip file contains a csv file with the measured values of the spectrum analyzer. You will find information about the measured values and their graphic representation in the section "AU-TOHOTSPOT".	--	X	X

Table 10-107 TFTP

File type	Description	Save	Down-load
Config	Start configuration	X	X
ConfigPack	Detailed configuration information. for example, start configuration, users, certificates, firmware of the de-vice (if saved as well). For more detailed information on creating and using the ConfigPack incl. firmware, refer to the section "Maintenance".	X	X
CountryList	The zip file contains the country list as a csv and as a pdf file.	X	--
Debug	This file contains information for Siemens Support.	X	--
EDS	Electronic Data Sheet (EDS) Electronic data sheets for describing devices in the EtherNet/IP mode	X	--
Firmware	Loading firmware updates	X	X
GSDML	Information on the device properties (PROFINET)	X	--
HTTPS Cert	HTTPS certificate Maximum file size: 8192 bits	X	X
LogFile	File with entries from the event log table	X	--
MIB	Private MSPS MIB file "Scalance_w_msps.mib"	X	--
RunningCLI	This file contains an overview of the current configura-tion in the form of CLI commands. You can download the text file. The file is not intended to be uploaded again unchanged.	X	--
Script	CLI script file	--	X
StartupInfo	Startup log file	X	--
Users	File with user names and passwords	X	X

File type	Description	Save	Down-load
WLANAuthlog	File with entries from the WLAN Authentication Log (information on successful or failed authentication attempts)	X	--
WLANCert (in client mode only)	User certificate. You can specify a password for the user certificate on the WBM page "Load&Save > Password". Maximum file size: 8192 bits	X	X
WLANServerCert (in client mode only)	Server certificate Maximum file size: 8192 bits	X	X
WLANSigRec (in client mode only)	The zip file contains the following: • csv file with the measured values of the signal recorder • pdf file with the measured values and an additional graphic representation of the measured values. You will find information about the measured values and their graphic representation in the section "AUTOHOTSPOT".	X	--
WLANSpectrumAnalyzer (Only in access point mode)	The Zip file contains a csv file with the measured values of the spectrum analyzer. You will find information about the measured values and their graphic representation in the section "AUTOHOTSPOT".	X	--

Uploading and saving using HTTP

Loading and saving data via HTTP

This page allows you to store device data in an external file on your client PC or to load such data from an external file from the client PC to the devices.
This means, for example, that you can also load new firmware from a file located on your client PC.

Note

The page is only available if there is an online connection to the device.
This page is available both for connections using HTTP and for connections using HTTPS.

Firmware

The firmware is signed and encrypted. This ensures that only firmware created by Siemens can be downloaded to the device.

Note

Incompatibility with predecessor versions

During the installation of a previous version, the configuration data and log files can be lost. In this case, the device starts up with the factory settings after the firmware has been installed.

Incompatibility with previous firmware versions with/without PLUG inserted

During the installation of a previous version, the configuration data can be lost. In this case, the device starts up with the factory settings after the firmware has been installed.

In this situation, if a PLUG is inserted in the device, following the restart, this has the status "Not Accepted" since the PLUG still has the configuration data of the previous more up-to-date firmware. This allows you to return to the previous, more up-to-date firmware without any loss of configuration data. If the original configuration on the PLUG is no longer required, the PLUG can be deleted or rewritten manually using the WBM page "System > PLUG".

Configuration files

Note

Configuration files and trial mode/automatic save mode

In "Automatic Save" mode, the data is saved automatically before the configuration files (ConfigPack and Config) are transferred.
In trial mode, although the changes are adopted, they are not saved in the configuration files (ConfigPack and Config). Use the "Write Startup Config" button on the "System > Configuration" WBM page to save changes in the configuration file.

CLI script file

You can download existing CLI configurations (RunningCLI) and upload your own CLI scripts (Script).

Note

The downloadable CLI script is not intended to be uploaded again unchanged.

Settings

The table has the following columns:

- **Type**
  Shows the designation of the file.
- **Description**
  Shows the short description of the file type.

- **Load**
  With this button, you can upload files to the device. The button can be enabled, if this function is supported by the file type.
- **Save**
  With this button, you can save files from the device. The button can only be enabled if this function is supported by the file type and the file exists on the device.
- **Delete**
  With this button, you can delete files from the device. The button can only be enabled if this function is supported by the file type and the file exists on the device.

Note

Following a firmware update, delete the cache of your Internet browser.

Uploading and saving using TFTP

Loading and saving data via a TFTP server

On this page, you can configure the TFTP server and the file names. You can also store device data in an external file on a TFTP server or load such data from an external file from the TFTP server to the devices. This means, for example, that you can also load new firmware from a file located on a TFTP server.

Firmware

The firmware is signed and encrypted. This ensures that only firmware created by Siemens can be downloaded to the device.

Note

Incompatibility with predecessor versions

During the installation of a previous version, the configuration data and log files can be lost. In this case, the device starts up with the factory settings after the firmware has been installed.

Incompatibility with previous firmware versions with/without PLUG inserted

During the installation of a previous version, the configuration data can be lost. In this case, the device starts up with the factory settings after the firmware has been installed.

In this situation, if a PLUG is inserted in the device, following the restart, this has the status "Not Accepted" since the PLUG still has the configuration data of the previous more up-to-date firmware. This allows you to return to the previous, more up-to-date firmware without any loss of configuration data. If the original configuration on the PLUG is no longer required, the PLUG can be deleted or rewritten manually using the WBM page "System > PLUG".

Configuration files

Note

Configuration files and Trial mode/Automatic Save

In "Automatic Save" configuration mode, the data is saved automatically before the configuration files (ConfigPack and Config) are transferred.
In trial mode , although the changes are adopted, they are not saved in the configuration files (ConfigPack and Config). Use the "Write Startup Config" button on the "System > Configuration" WBM page to save changes in the configuration files.

CLI script file

You can download existing CLI configurations (RunningCLI) and upload your own CLI scripts (Script).

Note

The downloadable CLI script is not intended to be uploaded again unchanged.

Settings

- **TFTP Server IP Address**
  Enter the IP address or the FQDN of the TFTP server with which you exchange data.
- **TFTP Server Port**
  Here, enter the port of the TFTP server via which data exchange will be handled. If necessary, you can change the default value 69 to your own requirements.

The table has the following columns:

- **File type**
  Shows the designation of the file.
- **Description**
  Shows the short description of the file type.

- **File name**
  A file name is preset here for every file type.

  Note
  Changing the file name

  You can change the file name preset in this column. After clicking the "Set Values" button, the changed name is stored on the device and can also be used with the Command Line Interface.

- **Select action** (only available online)
  Select the required action. The selection depends on the selected file type, for example you can only save the log file.
  The following actions are possible:
  - Save file
    With this selection, you save a file on the TFTP server.
  - Load file
    With this selection, you load a file from the TFTP server.

Passwords

Password for certificates

There are files to which access is password protected. For example to use the HTTPS certificate, you need to specify the corresponding password on this WBM page.

Settings

- **Type**
  Shows the file type
- **Description**
  Shows the short description of the file type.
- **Enabled**
  When selected, the file is used. Can only be enabled if the password is configured.
- **Password**
  Enter the password for the file.

- **Password Confirmation**
  Confirm the new password.
- **Status**
  Shows whether the password for the file match the device.
  - Valid
    The "Enabled" check box is selected and the password matches the file.
  - Invalid
    The "Enabled" check box is selected but the password does not match the file or no file has been loaded yet.
  - '-'
    The password cannot be evaluated or is not yet being used. The "Enabled" check box is not selected.

Events

Configuration

On this page, you specify how the device reacts to system events. By enabling the appropriate options, you specify how the device reacts to events. To enable or disable the options, click the relevant settings of the columns.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all events of table 2.
- **E-mail / Trap / Log Table / Syslog / Faults**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Event**
  The column contains the following values:
  - **Cold/Warm Restart**
    The device was turned on or restarted by the user.
  - **Link Change**
    This event occurs only when the port status is monitored and has changed, see "System > Fault Monitoring > Link Change".
  - Authentication error
    This event occurs when attempting access with a bad password.
  - Power Change
    This event occurs when power supply lines 1 and 2 are monitored. It indicates that there was a change to line 1 or line 2. The event occurs when the PoE power supply has failed, see "System > Fault Monitoring > Power Supply".
  - Spanning Tree Change
    The STP or RSTP or MSTP topology has changed.
  - Fault State Change
    The fault status has changed. The fault state can relate to the activated port monitoring, the response of the signaling contact or the power supply monitoring.
  - Overlap AP Detection (only in access point mode)
    This event is triggered when there is an entry in the Overlap AP list.
  - WDS (Only in access point mode)
    The connection status of a WDS link has changed.
  - DFS (Only in access point mode)
    This event occurs if a radar signal was received or the DFS scan was started or stopped.
  - WLAN General (Only in access point mode)
    This event occurs if a the channel bandwidth has changed.
  - WLAN Authentication Log
    Forwarding of the entries from the WLAN authentication log to the system protocol server.
  - WLAN De/Authentication (Only in client mode)
    With successful or failed WLAN authentication attempts.
  - iPCF Cycle Time (Only in access point mode)
    Only available when the KEY-PLUG is inserted.
    This event occurs if too many clients are logged on for the set iPCF cycle time or if some clients were not reached in one cycle.
  - iPCF Poll Size
    Only available when the KEY-PLUG is inserted.
    This event occurs if the PROFINET data size is too large for transfer.
- **E-mail**
  The device sends an e-mail. This is only possible if the SMTP server is set up and the "SMTP client" function is enabled.

- **Trap**
  The device sends an SNMP trap. This is only possible if "SNMPv1 Traps" is enabled in "System > Configuration".
- **Log Table**
  The device writes an entry in the event log table.
- **Syslog**
  The device writes an entry to the system log server. This is only possible if the system log server is set up and the "Syslog client" function is enabled.
- **Faults**
  The device triggers an error. The error LED lights up.
- **E-mail**
  The device sends an e-mail. This is only possible if the SMTP server is set up and the "SMTP client" function is enabled.
- **Trap**
  The device sends an SNMP trap. This is only possible if "SNMPv1 traps" is enabled in "System > Configuration".
- **Log table**
  The device writes an entry in the log table.
- **Syslog**
  The device writes an entry to the system log server. This is only possible if the system log server is set up and the "Syslog client" function is enabled.
- **Fault**
  The device triggers a fault. The error LED lights up

Severity filter

On this page, set the threshold levels for sending system event notifications.

Settings

The table has the following columns:

- **Client Type**
  Select the client type for which you want to make settings:
  - E-mail
    Sending system event messages by e-mail
  - Log table
    Entry of system events in the log table.
  - Syslog
    Entering system events in the Syslog file
  - WLAN Authentication Log
    Forwarding of the entries from the WLAN Authentication Log to the system protocol server.
    Can only be configured with SCALANCE W770 and SCALANCE W780
- **Severity**
  Select the required level. The following settings are possible:
  - Info
    System events are processed as of the severity level "Info".
  - Warning
    System events are processed as of the severity level "Warning".
  - Critical
    System events are processed as of the severity level "Critical".

SMTP client

The device provides the option of automatically sending an e-mail if an alarm event occurs (for example to the network administrator). The e-mail contains the identification of the sending device, a description of the cause of the alarm in plain language, and a time stamp. This allows centralized network monitoring to be set up for networks with few nodes based on an E-mail system. When an e-mail error message is received, the WBM can be started by the Internet browser using the identification of the sender to read out further diagnostics information.

On this page, you can configure up to three SMTP servers and the corresponding e-mail addresses.

Settings

- **SMTP Client**
  Enable or disable the SMTP client.
- **Sender Email Address**
  Enter the name of the sender to be included in the e-mail, for example the device name. This setting applies to all configured SMTP servers.
- **Send Test Mail**
  Send a test e-mail to check your configuration.

- **SMTP Port**
  Enter the port via which your SMTP server can be reached.
  Factory settings: 25
  This setting applies to all configured SMTP servers.
- **SMTP Server Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the SMTP server.

This table contains the following columns:

- **SMTP Server Address**
  Shows the IP address or the FQDN (Fully Qualified Domain Name) of the SMTP server.
- **Email address of the recipient**
  Enter the e-mail address to which the device sends an e-mail if a fault occurs.

Syntax of email addresses

The following conditions apply to e-mail addresses:

- Alphanumeric characters are permitted.
- The following special characters are permitted:
  - @
  - _ (underscore)
  - - (hyphen)
  - . (period)
- A @ character must be included.
- Only one @ may be used.
- The characters "@" and "." must not be the first or last character.

DHCPv4

DHCP client

If the device is configured as a DHCP client, it starts a DHCP query. As the reply to the query the device receives an IPv4 address from the DHCP server. The server manages an address range from which it assigns IPv4 addresses. It is also possible to configure the server so that the client always receives the same IPv4 address in response to its request.

Settings

- **DHCP client configuration file request (opt 66, 67)**
  Select this option if you want the DHCP client to use options 66 and 67 to download and then enable a configuration file.
- **DHCP Mode**
  Select the DHCP mode. The following modes are possible:
  - via MAC Address
    Identification is based on the MAC address.
  - via DHCP Client ID
    Identification is based on a freely defined DHCP client ID.
  - Via System Name
    Identification is based on the system name. If the device name is 255 characters long, the last character is not used for identification.
  - via PROFINET Name of Station
    Identification is based on the PROFINET name of the station.

This table contains the following columns:

- **Interface**
  Interface to which the setting relates.
- **DHCP**
  Enable or disable the DHCP client for the relevant interface.

DHCP server

Note

The page is only available for

- SCALANCE W780 / W770 / W740 / W730, W761-1 RJ-45 and W722-1 RJ-45

You can operate the device as a DHCP server. This allows IPv4 addresses to be assigned automatically to the connected devices. The IPv4 addresses are either distributed dynamically from an IPv4 address band you have specified or a specific IPv4 address (static) can be assigned to a particular device.

On this page, specify the IPv4 address band from which the device receives any IPv4 address. You configure the static assignment of the IPv4 addresses in "Static IP addresses".

Requirement

- In access point mode
  - The connected devices are configured so that they obtain the IPv4 address from a DHCP server.
- In client mode
  - The connected devices are configured so that they obtain the IPv4 address from a DHCP server.
  - NAT is enabled. You enable NAT in "Layer 3 > NAT".

Settings

- **DHCP server**
  Enable or disable the DHCP server on the device.

  Note

  To avoid conflicts with IPv4 addresses, only one device may be configured as a DHCP server in the network.

  Note

  Access point

  With an access point, the "DHCP Server" function is only possible on the VLAN assigned to the management (agent VLAN ID).

- **Prior to offering, check the address with ICMP echo**
  When selected, the DHCP server checks whether or not the IPv4 address has already been assigned. To do this the DHCP server sends ICMP echo messages (ping) to the IPv4 address. If no reply is received, the DHCP server can assign the IPv4 address.

  Note

  If there are devices in your network on which the echo service is disabled as default, there may be conflicts with the IPv4 addresses. To avoid this, assign these devices an IPv4 address outside the IPv4 address band.

The table has the following columns:

- **Pool ID**
  Shows the number of the IPv4 address band.

  Note

  Only one pool ID (ID = 1) can be created.

- **Enable**
  Specify whether or not this IPv4 address band will be used.

  Note

  If the IPv4 address band is enabled, neither DHCP options nor static PPv4 addresses can be configured.

- **Interface**
  Specify the interface via which the IPv4 addresses are dynamically assigned.
  The requirement for the assignment is that the IPv4 address of the interface is located within the IPv4 address band. If this is not the case, the interface does not assign any IPv4 addresses.
- **Subnet**
  Enter the network address range that will be assigned to the devices. Use the CIDR notation.
- **Lower IP address**
  Enter the IPv4 address that specifies the start of the dynamic IPv4 address band. The IPv4 address must be within the network address range you configured for "Subnet".
- **Upper IP address**
  Enter the IPv4 address that specifies the end of the dynamic IPv4 address band. The IPv4 address must be within the network address range you configured for "Subnet".
- **Lease Time [sec]**
  Specify for how many seconds the assigned IPv4 address remains valid. Once this period has elapsed, the device must either request a new IPv4 address or extend the lease time of the existing IPv4 address.

DHCP options

Note

The page is only available for

- SCALANCE W780 / W770 / W740 / W730, W761-1 RJ-45 and W722-1 RJ-45

On this page you specify which DHCP options the DHCP server supports. The various DHCP options are defined in RFC 2132.

Settings

- **Pool ID**
  Select the required IPv4 address band.
- **Option Code**
  Enter the number of the required DHCP option. The various DHCP options are defined in RFC 2132. The DHCP options 1, 3, 6, 66 and 67
  are created automatically when the IPv4 address band is created. With the exception of option 1, the options can be deleted.
  With the DHCP option 3, the internal IPv4 address of the device is automatically set as a DHCP parameter.

  Note

  Not supported DHCP servers

  The DHCP options 50 - 60 and 255 are not supported.

The table has the following columns:

- **Pool ID**
  Shows the number of the IPv4 address band.
- **Option Code**
  Shows the number of the DHCP option.
- **Use Interface IP**
  Specify whether or not the internal IPv4 address of the device will be used.
- **Value**
  Enter the DHCP parameter that is transferred to the DHCP client. The content depends on the DHCP option.
  - DHCP option 67 (boot file name)
    Enter the name of the boot file in the string format.
  - DHCP options 3 (Router) and 6 (DNS):
    Enter the DHCP parameter as an IPv4 address, e.g. 192.168.100.2. With DHCP option 6, you can specify several IPv4 addresses separated by commas.
  - DHCP option 66 (TFTP Server):
    Enter the DHCP parameter as an IPv4 address, e.g. 192.168.100.2 or the FQDN name.
  - All other DHCP options
    Enter the DHCP parameter in hexadecimal, e.g. the IPv4 address 192.168.100.2 corresponds to "C0A86402".

Static IP addresses

Note

The page is only available for

- SCALANCE W780 / W770 / W740 / W730, W761-1 RJ-45 and W722-1 RJ-45

On this page you specify that devices with a certain MAC address are assigned to the selected IPv4 address.

Settings

- **Pool ID**
  Select the required IPv4 address band.
- **Client Identification Method**
  Select the method according to which a client is identified.
  - Ethernet MAC
    The client is identified by its MAC address.
  - Client ID
    The client is identified by a freely defined DHCP client ID.
- **Value**
  Enter the MAC address.

The table has the following columns:

- **Pool ID**
  Shows the number of the IPv4 address band.

  Note

  Only Pool ID = 1 is supported.

- **Identification Method**
  Shows whether the client is identified by its MAC address or the client ID.
- **Value**
  Shows the MAC address to which the IPv4 address is assigned.
- **IP Address**
  Specify the IPv4 address. The IPv4 address must match the subnet of the IPv4 address band.

SNMP

General

On this page, you make the basic settings for SNMP. Enable the options according to the function you want to use.

Settings

- **SNMP**
  Select the SNMP protocol. The following settings are possible:
  - "-" (disabled)
    SNMP is disabled.
  - SNMPv1/v2c/v3
    SNMPv1/v2c/v3 is supported.
  - SNMPv3
    Only SNMPv3 is supported.
- **SNMPv1/v2 Read-Only**
  If you enable this option, SNMPv1/v2c can only read the SNMP variables.

  Note

  Community String

  For security reasons, do not use the standard values "public" or "private". Change the community strings following the initial installation.

- **SNMPv1/v2c Read Community String**
  Enter the community string for access of the SNMP protocol.
- **SNMPv1/v2c Read/Write Community String**
  Enter the community string for read and write access of the SNMP protocol.

- **SNMPv1 Traps**
  Enable or disable the sending of SNMP traps (alarm frames). On the "Trap" tab, specify the IP addresses of the devices to which SNMP traps will be sent.
- **SNMPv1/v2c Trap Community String**
  Enter the community string for sending SNMPv1/v2 messages.
- **SNMPv3 User Migration**
  - **Enabled**
    If the function is enabled, an SNMP engine ID is generated that can be migrated. You can transfer configured SNMPv3 users to a different device.
    If you enable this function and load the configuration of the device on another device, configured SNMPv3 users are retained.
  - **Disabled**
    If the function is disabled, a device-specific SNMP engine ID is generated. To generate the ID, the agent MAC address of the device is used. You cannot transfer this SNMP user configuration to other devices.
    If you load the configuration of the device on another device, all configured SNMPv3 users are deleted.
- **SNMP Engine ID**
  Shows the SNMP engine ID.

Traps

If an alarm event occurs, a device can send SNMP traps (alarm frames) to up to ten different management stations at the same time. Traps are only sent if the events specified in "Events" occur.

Note

SNMP traps are sent only when the "SNMPv1 Traps" setting was selected in "SNMP > General".

Settings

- **Trap Receiver Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the station to which the device sends SNMP traps. You can specify up to ten different recipients.

The table has the following columns:

- **Trap Receiver Address**
  If necessary, change the IP addresses or the FQDN of the stations.
- **Trap**
  Enable or disable the sending of SNMP traps. Stations that are entered but not selected do not receive SNMP traps.

v3 Groups

Security settings and assigning permissions

SNMP version 3 allows permissions to be assigned, authentication, and encryption at protocol level. The security levels and read/write permissions are assigned according to groups. The settings automatically apply to every member of a group.

Settings

- **Group Name**
  Enter the name of the group. The maximum length is 32 characters.
- **Security Level**
  Select the security level (authentication, encryption) valid for the selected group. You have the following options for the security levels:
  - no Auth/no Priv
    No authentication enabled / no encryption enabled.
  - Auth/no Priv
    Authentication enabled / no encryption enabled.
  - Auth/Priv
    Authentication enabled / encryption enabled.

The table has the following columns:

- **Group Name**
  Shows the defined group names.

  Note

  Once a group name and the security level have been specified, they can no longer be modified after the group is created. If you want to change the group name or the security level , you will need to delete the group and recreate it and reconfigure it with the new name.

- **Security Level**
  Shows the configured security level.
- **Read**
  Enable or disable read access for the required group.

- **Write**
  Enable or disable write access for the required group.

  Note

  For write access to work, you also need to enable read access.

- **Persistence**
  Shows whether or not the group is assigned to an SNMPv3 user. If the group is not assigned to an SNMPv3 user, no automatic saving is triggered and the configured group is deleted after restarting the device.
  - Yes
    The group is assigned to an SNMPV3 user.
  - No
    The group is not assigned to an SNMPV3 user.

v3 users

User-specific security settings

On the WBM page, you can create new SNMPv3 users and modify or delete existing users. The user-based security model works with the concept of the user name; in other words, a user ID is added to every frame. This user name and the applicable security settings are checked by both the sender and recipient.

Settings

- **User name**
  Enter a freely selectable user name. After you have entered the data, you can no longer modify the name.

The table has the following columns:

- **User name**
  Shows the created users.
- **Group Name**
  Select the group to which the user will be assigned.
- **Authentication Protocol**
  Specify the authentication protocol for which a password will be stored.
  The following settings are available:
  - None
  - MD5
  - SHA
- **Encryption Protocol**
  Specify whether or not a password should be stored for encryption with the DES algorithm. Can only be enabled when an authentication protocol has been selected.

- **Authentication Password**
  Enter the authentication password. This password must have at least 6 characters, the maximum length is 32 characters.
- **Confirm authentication password**
  Confirm the password by repeating the entry.
- **Encryption Password**
  Enter your encryption password. This password must have at least 6 characters, the maximum length is 32 characters.
- **Confirm Encryption Password**
  Confirm the encryption password by repeating the entry.
- **Persistence**
  Shows whether or not the user is assigned to an SNMPv3 group. If the user is not assigned to an SNMPv3 group, no automatic saving is triggered and the configured user is deleted after restarting the device.
  - Yes
    The user is assigned to an SNMPv3 group.
  - No
    The user is not assigned to an SNMPv3 group.

System time

Manual time setting

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

On this page, you set the date and time of the system. For this setting to be used, enable "Manual time setting".

Note

The page is only available if there is an online connection to the device.

Settings

- **Time Manually**
  Enable or disable the manual time setting. If you enable the option, the "System Time" input box can be edited.
- **System Time**
  Enter the date and time in the format "MM/DD/YYYY HH:MM:SS".
- **Use PC Time**
  Click the button to use the time setting of the PC.

- **Last Synchronization Time**
  Shows when the last time-of-day synchronization took place.
  If no time-of-day synchronization was possible, the box displays "Date/time not set".
- **Last Synchronization Mechanism**
  This box displays how the last time-of-day synchronization was performed.
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame.
  - PTP
    Automatic time-of-day synchronization with PTP

DST overview

On this page, you can create new entries for the daylight saving time changeover.

The table provides an overview of the existing entries.

Settings

- **DST No.**
  Shows the number of the entry.
  If you create a new entry, a new line with a unique number is created.
- **Name**
  Shows the name of the entry.
- **Year**
  Shows the year for which the entry was created.
- **Start Date**
  Shows the month, day and time for the start of daylight saving time.
- **End Date**
  Shows the month, day and time for the end of daylight saving time.
- **Recurring Date**
  With an entry of the type "Rule", the period in which daylight saving time is active is displayed consisting of week, day, month and time of day. With an entry of the type "Date" a "-" is displayed.

- **State**
  Shows the status of the entry:
  - enabled
    The entry was created correctly.
  - invalid
    The entry was created new and the start and end date are identical.
- **Type**
  Shows how the daylight saving time changeover is made:
  - Date
    A fixed date is entered for the daylight saving time changeover.
  - Recurring
    A rule was defined for the daylight saving time changeover.

DST configuration

On this page, you can configure the entries for the daylight saving time changeover. As result of the changeover to daylight saving or standard time, the system time for the local time zone is correctly set.

You can define a rule for the daylight saving time changeover or specify a fixed date.

Settings

Note

The content of this page depends on the selection in the "Type" box.

The boxes "DST No.", "Type" and "Name" are always shown.

- **DST No.**
  Select the type of the entry.
- **Type**
  Select how the daylight saving time changeover is made:
  - Date
    You can set a fixed date for the daylight saving time changeover.
    This setting is suitable for regions in which the daylight saving time changeover is not governed by rules.
  - Recurring
    You can define a rule for the daylight saving time changeover.
    This setting is suitable for regions in which the daylight saving time always begins or ends on a certain weekday.
- **Name**
  Enter a name for the entry.

Settings with "Date" selected

You can set a fixed date for the start and end of daylight saving time.

- **Year**
  Enter the year for the daylight saving time changeover.
- **Start Date**
  Enter the following values for the start of daylight saving time:
  - Day
    Specify the day.
  - Hour
    Specify the hour.
  - Month
    Specify the month.
- **End Date**
  Enter the following values for the end of daylight saving time:
  - Day
    Specify the day.
  - Hour
    Specify the hour.
  - Month
    Specify the month.

Settings with "Recurring" selected

You can create a rule for the daylight saving time changeover.

- **Start Date**
  Enter the following values for the start of daylight saving time:
  - Hour
    Specify the hour.
  - Month
    Specify the month.
  - Week
    Specify the week.
    You can select the 1st to 4th or the last week of the month.
  - Weekday
    Specify the weekday.
- **End Date**
  Enter the following values for the end of daylight saving time:
  - Hour
    Specify the hour.
  - Month
    Specify the month.
  - Week
    Specify the week.
    You can select the 1st to 4th or the last week of the month.
  - Weekday
    Specify the weekday.

SNTP client

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

SNTP (Simple Network Time Protocol) is used for synchronizing the time in the network. The appropriate frames are sent by an SNTP server in the network.

Note

To avoid time jumps, make sure that there is only one time server in the network.

Settings

- **SNTP Client**
  Enable or disable automatic time-of-day synchronization using SNTP.
- **Current System Time** (only available online)
  Shows the current date and current normal time received by the IE switch. If you specify a time zone, the time information is adapted accordingly.

- **Last Synchronization Time**(only available online)
  Shows when the last time-of-day synchronization took place.
- **Last Synchronization Mechanism** (only available online)
  Shows how the last time-of-day synchronization was performed. The following methods are possible:
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame
  - PTP
    Automatic time-of-day synchronization with PTP
- **Time Zone**
  Enter the time zone you are using in the format "+/- HH:MM". The time zone relates to UTC standard world time.
  The time in the "Current System Time" box is adapted accordingly.
- **Daylight Saving Time**
  Shows whether the daylight saving time changeover is active.
  - active (offset +1 h)
    The system time was changed to daylight saving time; in other words an hour was added. The normal time including the time zone continues to be displayed in the "Current System Time" box.
    The normal time including the time zone continues to be displayed in the "Current System Time" box.
  - inactive (offset +0 h)
    The current system time is not changed.
- **SNTP Mode**
  Select the synchronization mode. The following types of synchronization are possible:
  - Poll
    If you select this protocol type, the input boxes "SNTP Server Address", "SNTP Server Port" and "Poll Interval[s]" are displayed to allow further configuration. With this type of synchronization, the device is active and sends a time query to the SNTP server.
  - Listen
    With this type of synchronization, the device is passive and "listens" for SNTP frames that deliver the time of day.
- **Poll Interval[s]**
  Here, enter the interval between two-time queries. In this box, you enter the query interval in seconds. Possible values are 16 to 16284 seconds.

- **SNTP Server Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the SNTP server.
- **SNTP Server Port**
  Enter the port of the SNTP server.
  The following ports are possible:
  - 123 (standard port)
  - 1025 to 36564
- **Primary**
  The check mark is set for the SNTP server that you create first. If several SNTP servers have been created, the primary server is queried first.

NTP client

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

If you require time-of-day synchronization using NTP, you can make the relevant settings here.

Note

To avoid time jumps, make sure that there is only one time server in the network.

Settings

- **NTP Client**
  Enable or disable automatic time-of-day synchronization using NTP.
- **Current System Time** (only available online)
  Shows the current date and current normal time received by the IE switch. If you specify a time zone, the time information is adapted accordingly.
- **Last Synchronization Time**(only available online)
  Shows when the last time-of-day synchronization took place.

- **Last Synchronization Mechanism** (only available online)
  This box displays how the last time-of-day synchronization was performed. The following methods are possible:
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame
  - PTP
    Automatic time-of-day synchronization with PTP
- **Time Zone**
  Enter the time zone you are using in the format "+/- HH:MM". The time zone relates to UTC standard world time.
  The time in the "Current System Time" box is adapted accordingly.
- **Daylight Saving Time**
  Shows whether the daylight saving time changeover is active.
  - active (offset +1 h)
    The system time was changed to daylight saving time; in other words an hour was added. The normal time including the time zone continues to be displayed in the "Current System Time" box.
  - inactive (offset +0 h)
    The current system time is not changed.
- **NTP Server Index**
  Select the index of the NTP server. The server with the lowest index is queried first.
- **NTP Server Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the NTP server.
- **NTP Server Port**
  Enter the port of the NTP server.
  The following ports are possible:
  - 123 (standard port)
  - 1025 to 36564
- **Poll Interval[s]**
  Here, enter the interval between two time queries. In this box, you enter the poll interval in seconds. Possible values are 64 to 1024 seconds.

SIMATIC Time Client

There are different methods that can be used to set the system time of the device. Only one method can be active at any one time. If one method is activated, the previously activated method is automatically deactivated.

On this page, you configure time synchronization with the SIMATIC Time Client.

Note

To avoid time jumps, make sure that there is only one time server in the network.

Settings

- **SIMATIC Time Client**
  Enable or disable the SIMATIC time client.
- **Current System Time** (only available online)
  Shows the current system time.
- **Last Synchronization Time**(only available online)
  Shows when the last time-of-day synchronization took place.
- **Last Synchronization Mechanism** (only available online)
  This box displays how the last time-of-day synchronization was performed.
  - Not set
    The time was not set.
  - Manual
    Manual time setting
  - SNTP
    Automatic time-of-day synchronization using SNTP
  - NTP
    Automatic time-of-day synchronization using NTP
  - SIMATIC
    Automatic time-of-day synchronization using the SIMATIC time frame
  - PTP
    Automatic time-of-day synchronization with PTP

Auto logout

On this page, set the times after which there is an automatic logout from WBM or the CLI following user in activity.

Note

No automatic logout from the CLI

If the connection is not terminated after the set time, check the setting of the "keepalive" mechanism on the Telnet client.

If the interval is shorter than the configured time, the connection is kept alive although no user data is transferred. Example: you selected 300 seconds for the automatic logout and 120 seconds is set for the keepalive function. In this case, a packet is sent every 120 seconds that keeps the connection up.

- Disable the keepalive mechanism (interval time = 0)
  or
- Set the interval high enough so that the underlying connection is terminated when there is inactivity.

Settings

- **Web Based Management [s]**
  Enter the time in seconds for the automatic logout from WBM. If you enter the value 0, the automatic logout is disabled.
- **CLI (TELNET, SSH) [s]**
  Enter the time in seconds for the automatic logout from the CLI. If you enter the value 0, the automatic logout is disabled.

Syslog client

Syslog according to RFC 3164 is used for transferring short, unencrypted text messages over UDP in the IP network. This requires a system log server.

Requirements for sending log entries:

- The Syslog function is enabled on the device.
- The Syslog function is enabled for the relevant event.
- There is a Syslog server in your network that receives the log entries. Since this is a UDP connection, there is no acknowledgment to the sender.
- The IP address or the FQDN (Fully Qualified Domain Name) of the Syslog server is entered in the device.

Settings

- **Syslog Client**
  Enable or disable the Syslog function.
- **Syslog Server Address**
  Enter the IP address or the FQDN (Fully Qualified Domain Name) of the Syslog server.

The table contains the following columns:

- **Syslog Server Address**
  Shows the IP address or the FQDN (Fully Qualified Domain Name) of the Syslog server.
- **Server Port**
  Enter the port of the Syslog server being used.

Fault Monitoring

Power supply

Note

This page is not available on the SCALANCE W786-2 SFP.

Configure whether or not the power supply should be monitored by the messaging system. Depending on the hardware variant, there are one or two power connectors (Supply 1 / Supply 2). With a redundant power supply, configure the monitoring separately for each individual feed-in line.

If there is no power on one of the monitored lines (Supply 1 or Supply 2) or when the voltage is too low, a fault is signaled by the signaling system.

Note

You will find the permitted operating voltage limits in the operating instructions of the device.

An error causes the fault LED to light up on the device. Depending on the configuration, the error may trigger a trap, an e-mail or an entry in the event log table.

Settings

- **Line 1**
  Enable or disable the monitoring of power connector 1.
- **Line 2** (not available with W760/W720)
  Enable or disable the monitoring of power connector 2.

- **PoE** (not available with W760/W720)
  Enable or disable the monitoring of the power supply via Ethernet.
- **PROFINET redundancy** (not available with W760/W720)
  With the following devices, you can also configure which power supply will be monitored by PROFINET:
  - SCALANCE W788-x RJ-45 variants
  - SCALANCE W748-1 RJ-45
  - SCALANCE W774-1 (RJ-45 and M12 variant)
  - SCALANCE W734-1 RJ-45

Link Change

On this page, you configure whether or not an error message is triggered if there is a status change on a network connection.

If connection monitoring is enabled, an error is signaled

- when there should be a link on a port and this is missing.
- or when there should not be a link on a port and a link is detected.

An error causes the signaling contact to trigger and the fault LED to light up on the device. Depending on the configuration, the error may trigger a trap, an e-mail or an entry in the event log table.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports.
- **Setting**
  Select the required setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns

- **Port**
  Shows the available ports.
- **Setting**
  Select the setting. You have the following options:
  - Up
    Error handling is triggered when the port changes to the active status.
    (From "Link down" to "Link up")
  - Down
    Error handling is triggered when the port changes to the inactive status.
    (From "Link up" to "Link down")
  - "-" (disabled)
    The error handling is not triggered.

EtherNet/IP

EtherNet/IP

On this page, you configure the mode of EtherNet/IP.

Note

Devices with two Ethernet interfaces

On devices with two Ethernet interfaces only one of the interfaces (P1 or P2) may be used for the EtherNet/IP configuration.

This applies to the following devices:

- SCALANCE W786-2 SFP .
- SCALANCE W774-1 RJ-45
- SCALANCE W774-1 M12 EEC
- SCALANCE W734-1 RJ-45.

Description of the displayed boxes

The page contains the following boxes:

- **EtherNet/IP Device Diagnostics**
  Shows whether EtherNet/IP is enabled ("On") or disabled ("Off").
- **EtherNet/IP Device Diagnostics for next boot**
  Set whether EtherNet/IP will be enabled ("On") or disabled ("Off") after the next device restart.

Note

EtherNet/IP and PROFINET

When EtherNet/IP is turned on, PROFINET is turned off. The switchover from EtherNet/IP and PROFINET has no effect on DCP.

Note

PROFINET AR Status

If a PROFINET connection is established; in other words the PROFINET AR status is "Online", you cannot enable EtherNet/IP.

PROFINET

Settings for PROFINET

This page shows the PROFINET AR status and the device name.

Description of the displayed boxes

The page contains the following boxes:

- **PROFINET Mode**
  Shows whether PROFINET is enabled ("On") or disabled ("Off").
- **PROFINET Mode for next boot**
  Set whether PROFINET will be enabled ("On") or disabled ("Off") after the next device restart.

Note

PROFINET and EtherNet/IP

When PROFINET is turned on, EtherNet/IP is turned off. The switchover from PROFINET and EtherNet/IP has no effect on DCP.

Note

PROFINET AR Status

If a PROFINET connection is established; in other words the PROFINET AR status is "Online", you cannot disable PROFINET.

- **PROFINET AR Status**
  This box shows the status of the PROFINET connection; in other words whether the device is connected to a PROFINET controller "Online " or "Offline".
  Here, online means that a connection to a PROFINET controller exists, that this has downloaded its configuration data to the device and that the device can send status data to the PROFINET controller. In this status known as "in data exchange", the parameters set via the PROFINET controller cannot be configured.
- **PROFINET Name of Station**
  This box displays the PROFINET device name according to the configuration in HW Config of STEP 7.

Note

Devices with two Ethernet interfaces

With devices that have two Ethernet interfaces, only interface P1 should be used for the PROFINET configuration because LLPD frames can only be sent and received via interface P1. They are blocked at interface P2 and are also not forwarded between the interfaces.

This applies to the following devices:

- SCALANCE W786-2 SFP
- SCALANCE W774-1 RJ-45
- SCALANCE W774-1 M12 EEC
- SCALANCE W734-1 RJ-45

SCALANCE W700 and STEP 7

The Ethernet interface can be configured in STEP 7 if the following requirements are met:

- STEP 7 V13 Update 3 with HSP0107 or
- STEP7 version 5.5.4 with GSDML version 2.31

The diagnostics functions can also be used. The WLAN interface cannot be configured with STEP 7.

PROFINET for client devices

If a client is to be used as a PROFINET device, the MAC address of the client must be specified as follows (MAC Mode):

- **Own**
  In the network beyond the device, only IP communication and no PROFINET is possible.
- **Layer 2 Tunnel**
  The client and the devices downstream from it can be used as PROFINET devices.

  Note

  If "Automatic" or "Manual" is configured as the MAC mode for a client, this device cannot be used as a PROFINET device.

PLUG

Configuration

Note

The page is only available if there is an online connection to the device.

This page is only available for SCALANCE W780 / W 770 / W740 / W730.

NOTICE

Do not remove or insert a C-PLUG / KEY-PLUG during operation!

A PLUG may only be removed or inserted when the device is turned off.

The device checks whether or not a PLUG is inserted at one second intervals. If it is detected that the PLUG was removed, there is a restart. If a valid KEY-PLUG was inserted in the device, the device changes to a defined error state following the restart. With SCALANCE W, the available wireless interfaces are deactivated in this case.

If the device was configured at some time with a PLUG, the device can no longer be used without this PLUG. To be able to use the device again, reset the device to the factory settings.

Information about the configuration of the C-PLUG / KEY-PLUG

This page provides detailed information about the configuration stored on the C-PLUG or KEY-PLUG. It is also possible to reset the PLUG to factory defaults or to load it with new contents.

Note

The action is only executed after you click the "Set Values" button.

The action cannot be undone.

If you decide against executing the function after making your selection, click the "Refresh" button. As a result the data of this page is read from the device again and the selection is canceled.

Note

Incompatibility with previous versions with PLUG inserted

During the installation of a previous version, the configuration data can be lost. In this case, the device starts up with the factory settings after the firmware has been installed. In this situation, if a PLUG is inserted in the device, following the restart, this has the status "NOT ACCEPTED" since the PLUG still has the configuration data of the previous more up-to-date firmware. This allows you to return to the previous, more up-to-date firmware without any loss of configuration data.

If the original configuration on the PLUG is no longer required, the PLUG can be deleted or rewritten manually using "System > PLUG".

Settings

The table has the following rows:

- **Status**
  Shows the status of the PLUG. The following are possible:
  - ACCEPTED
    There is a PLUG with a valid and suitable configuration in the device.
  - NOT ACCEPTED
    Invalid or incompatible configuration on the inserted PLUG.
  - NOT PRESENT
    There is no C-PLUG or KEY-PLUG inserted in the device.
  - FACTORY
    PLUG is inserted and does not contain a configuration. This status is also displayed when the PLUG was formatted during operation.
  - MISSING
    There is no PLUG inserted. Functions are configured on the device for which a license is required.
- **Device Group**
  Shows the SIMATIC NET product line that used the C-PLUG or KEY-PLUG previously.
- **Device Type**
  Shows the device type within the product line that used the C-PLUG or KEY-PLUG previously.
- **Configuration Revision**
  The version of the configuration structure. This information relates to the configuration options supported by the device and has nothing to do with the concrete hardware configuration. This revision information does not therefore change if you add or remove additional components (modules or extenders), it can, however, change if you update the firmware.
- **File System**
  Displays the type of file system on the PLUG.

NOTICE
New file system UBI As of firmware version 2.0, UBI is the standard file system for the C-PLUG or KEY-PLUG. If a C-PLUG with the previous file system IECP is detected in such a device, this C-PLUG will be formatted for the UBI file system and the data will be rewritten to the C-PLUG. This change in the file system also occurs following a firmware update to V2.0. A downgrade to the previous version of the firmware then presents a problem. The firmware can neither read nor write the C-PLUG or KEY-PLUG and it is not even possible to "Erase PLUG to factory default".

- **File System Size [bytes]**
  Shows the maximum storage capacity of the file system on the C-PLUG.
- **File System Usage**
  Displays the storage space in use in the file system of the C-PLUG.

- **Firmware on PLUG**
  When enabled, the firmware will be stored on the PLUG. This means that automatic firmware updates/downgrades can be made with the PLUG.
- **Info String**
  Shows additional information about the device that used the PLUG previously, for example, article number, type designation, and the versions of the hardware and software. The displayed software version corresponds to the version in which the configuration was last changed. With the "NOT ACCEPTED" status, further information on the cause of the problem is displayed.
  If a PLUG was configured as a PRESET PLUG this is shown here as additional information in the first row. For more detailed information on creating and using a PRESET PLUG refer to the section "Maintenance (page 363)".
- **Modify PLUG**
  Select the required setting. You have the following options for changing the configuration on the C-PLUG or KEY-PLUG:
  - Write current configuration to PLUG
    This option is available only if the status of the PLUG is "NOT ACCEPTED" or FACTORY. The configuration in the internal flash memory of the device is copied to the PLUG.
  - Erase PLUG to factory default
    Deletes all data from the C-PLUG and triggers low-level formatting.

License

NOTICE
Do not remove or insert a C-PLUG / KEY-PLUG during operation! A PLUG may only be removed or inserted when the device is turned off. The device checks whether or not a PLUG is present at one second intervals. If it is detected that the PLUG was removed, there is a restart. If a valid KEY-PLUG was inserted in the device, the device changes to a defined error state following the restart. With SCALANCE W, the available wireless interfaces are deactivated in this case. If the device was configured at some time with a PLUG, the device can no longer be used without this PLUG. To be able to use the device again, reset the device to the factory settings.

Note

Incompatibility with previous versions with PLUG inserted

During the installation of a previous version, the configuration data can be lost. In this case, the device starts up with the factory settings after the firmware has been installed. In this situation, if a PLUG is inserted in the device, following the restart, this has the status "NOT ACCEPTED" since the PLUG still has the configuration data of the previous more up-to-date firmware. This allows you to return to the previous, more up-to-date firmware without any loss of configuration data.

If the original configuration on the PLUG is no longer required, the PLUG can be deleted or rewritten manually using "System > PLUG".

Information about the license of the KEY-PLUG

A C-PLUG can only store the configuration of a device. In addition to the configuration, a KEY-PLUG also contains a license that enables certain functions of your SIMATIC NET device.

Note

Information only available online

You can only view the information on this page if there is an online connection to the device.

This page is only available for SCALANCE W780 / W 770 / W740 / W730.

Displayed values

- **Status**
  Shows the status of the KEY-PLUG. The following are possible:
  - ACCEPTED
    The KEY-PLUG in the device contains a suitable and valid license.
  - NOT ACCEPTED
    The license of the inserted KEY-PLUG is not valid.
  - NOT PRESENT
    No KEY-PLUG is inserted in the device.
  - MISSING
    There is no KEY-PLUG or a C-PLUG with the status "FACTORY" inserted in the device. Functions are configured on the device for which a license is required.
  - WRONG
    The inserted KEY-PLUG is not suitable for the device.
  - UNKNOWN
    Unknown content of the KEY-PLUG.
  - DEFECTIVE
    The content of the KEY-PLUG contains errors.
- **Article number**
  Shows the article number of the KEY-PLUG. The KEY-PLUG is available for various functional enhancements and for various target systems.
- **Serial Number**
  Shows the serial number of the KEY-PLUG.
- **Info String**
  Shows additional information about the device that used the KEY-PLUG previously, for example, article number, type designation, and the versions of the hardware and software. The displayed software version corresponds to the version in which the configuration was last changed. With the "NOT ACCEPTED" status, further information on the cause of the problem is displayed.

Note

When you save the configuration, the information about whether or not a KEY-PLUG was inserted in the device at the time is also saved. This configuration can then only work if a KEY-PLUG with the same order number / license is inserted. This applies regardless of whether or not iFeatures are configured.

Ping

Reachability of an address in an IP network

Note

The page is only available if there is an online connection to the device.

With the ping function, you can check whether a certain IPv4 address is reachable in the network.

Settings

- **IP Address**
  Enter the IPv4 address of the device.
- **Repeat**
  Enter the number of ping requests.
- **Ping**
  Click this button to start the ping function.
- **Ping Output**
  This box shows the output of the ping function.
- **Delete**
  Click this button to delete the ping output.

Configuring interfaces

Ethernet

Overview

The page shows the configuration for the data transfer for all ports of the device.

Settings

- **Port**
  Shows the configurable ports.
- **Port name**
  Shows the name of the port.
- **Status**
  Shows whether the port is on or off. Data traffic is possible only over an enabled port.
- **OperState**
  Displays the current operational status. The operational status depends on the configured "Status" and the "Link".
  The available options are as follows:
  - up
    You have configured the status "enabled" for the port and the port has a valid connection to the network.
  - down
    You have configured the status "disabled" or "Link down" for the port or the port has no connection.
- **Link** (only available online)
  Shows the connection status to the network. With the connection status, the following is possible:
  - up
    The port has a valid link to the network, a link integrity signal is being received.
  - down
    The link is down, for example because the connected device is turned off.
- **Mode** (only available online)
  Shows the transfer parameters of the port
- **MTU**
  Shows the maximum packet size.
- **Negotiation**
  Shows whether the automatic configuration is enabled or disabled.
- **MAC Address** (only available online)
  Shows the MAC address of the port.

Configuration

On this page, you configure the Ethernet ports of the device.

Note

SCALANCE W786-2 SFP

The two SFP ports of the SCALANCE W786-2 SFP cannot be assigned parameters or diagnosed individually.

Settings

- **Port**
  Select the port to be configured.
- **Status**
  Specify whether the port is enabled or disabled.
  - enabled
    The port is enabled. Data traffic is possible only over an enabled port.
  - disabled
    The port is disabled but the connection remains.
  - Link down
    The port is disabled and the connection to the partner device is terminated.
- **Port name**
  Enter a name for the port.
- **MAC Address** (only available online)
  Shows the MAC address of the port.
- **Mode Type**

  Note

  This parameter cannot be configured for:
  - SCALANCE W786-2 SFP
  - SCALANCE W770
  - SCALANCE W760
  - SCALANCE W730
  - SCALANCE W720

  The value is permanently set to “Auto negotiation“.

  Shows the transmission speed and the transmission method of the port. You make the settings for "Autonegotiation" and "Transmission rate" in the port options.

  Note

  Before the port and partner port can communicate with each other, the settings must match at both ends.

- **Mode** (only available online)
  Shows which transmission speed and which transmission mode is currently being used.
- **Negotiation**
  Shows whether the automatic configuration of the connection to the partner port is enabled or disabled.
- MTU ((Maximial Transmission Unit)
  Enter the packet size above which packets are fragmented.

- **OperState**
  Displays the current operational status. The operational status depends on the configured "Status" and the "Link".
  The available options are as follows:
  - up
    You have configured the status "enabled" for the port and the port has a valid connection to the network.
  - down
    You have configured the status "disabled" or "Link down" for the port or the port has no connection.
- **Link** (only available online)
  Shows the connection status to the network. With the connection status, the following is possible:
  - Up
    The port has a valid link to the network, a link integrity signal is being received.
  - Down
    The link is down, for example because the connected device is turned off.

Changing the port configuration

Note

Optical ports only work with the full duplex mode and at maximum transmission rate. As a result, the following settings cannot be made for optical ports:

- Automatic configuration
- Transmission speed
- Transmission technique

Note

With various automatic functions, the device prevents or reduces the effect on other ports and priority classes (Class of Service) if a port is overloaded. This can mean that frames are discarded even when flow control is enabled.

Port overload occurs when the device receives more frames than it can send, for example as the result of different transmission speeds.

WLAN

Basic

Note

To configure the WLAN interface, you must always specify the country code first. Some parameters are dependent on the country setting, for example the transmission standard.

Settings

- **Country Code**
  Specify the country in which the device will be used.
  You do not need to know the data for the specific country, the channel division and output power are set by the device according to the country you select.

  Note

  Locale setting

  The correct country setting is mandatory for operation complying with the approvals. Selecting a country different from the country of use can lead to legal prosecution.

- **Device Mode**
  Mode of the device. The mode can only be changed for access points.
  The following modes are possible:
  - AP: Access point mode
  - Client: Client mode

  Note

  After changing the mode, a message is displayed. If you confirm the message with "Yes", the configuration settings are updated in the changed mode.

- **Radio**
  Shows the available WLAN interfaces.
- **Enabled**
  Status of the WLAN interface. To enable the WLAN interface, enable the setting.

  Note

  Enabling the WLAN interface

  The WLAN interfaces are disabled when the device is supplied. The WLAN interfaces are can be enabled once the country and the antenna settings are configured.

- **Radio Mode**
  Shows the mode of the WLAN interface.

- **Frequency Band**
  Specify the frequency band. In client mode, dual-frequency operation is also possible.
  - 2.4 GHz
  - 5 GHz
  - 2.4 GHz + 5 GHz (only in client mode)

Note

Configuring WLAN interfaces of the W786-2IA RJ-45 for different frequency bands

If both WLAN interfaces are configured for the same frequency band on this device, there may be mutual influence or interference. This applies in particular when there is a high data throughput.

- **WLAN Mode 2.4 GHz/WLAN Mode 5 GHz**
  Select the required transmission standard for the configured frequency band.
  The selection depends on the country setting.
  - Auto (in client mode only)
    The transmission standard is determined automatically (2.4 GHz + 5 GHz).
  - 802.11a
    The transmission standard IEEE 802.11a (5 GHz) is set.
  - 802.11g
    The transmission standard IEEE 802.11g (2.4 GHz) is set. This transmission standard is downwards compatible with IEEE 802.11b
  - 802.11n
    The transmission standard IEEE 802.11n (2.4 GHz and 5 GHz) is set. This transmission standard is downwards compatible with IEEE 802.11a and IEEE 802.11g.
  - 802.11n only
    The transmission standard IEEE 802.11n (2.4 GHz and 5 GHz) is set. This transmission standard is downwards compatible with IEEE 802.11a and IEEE 802.11g.

Note

Fragmentation length

If you select the transmission standard "802.11n", "802.11n only" or "Auto" (only in client mode), you cannot set the threshold value of the fragmentation length see "Fragmentation Length Threshold" in "Interfaces > WLAN > Advanced".

- **DFS (802.11h)**
  Enables or disables the "Dynamic Frequency Selection (DFS)" function.
  - Enabled
    If you have enabled this function, additional DFS channels from the 5 GHz band are available to you. These channels are country-specific and are subject to certain DFS regulations.
    If the access point discovers a disturbance on the current channel, it changes automatically to an alternative channel and the current channel is blocked for 30 minutes. The disturbance can originate from a primary user or radar interference.
    Before the access point starts the communication on a channel it searches 60 seconds for primary users on the channel. During this time the access point does not send beacons. If signals are found on the channel, the access point changes channel and repeats the check. Only when no signals from primary users are detected after 60 seconds does the access point send on a channel. The access point also searches for primary users during operation.
  - Disabled
    The DFS function is not used.
- **Outdoor Mode**
  - Enabled
    If you have enabled the outdoor mode, you only have the channels available that are permitted for outdoor operation.
  - Disabled
    If you have disabled the outdoor mode, you only have the channels available that are permitted for operation in a building.

- **Maximum Tx Power**
  Specify the transmit power of the device. If the transmit power is set too high the received signal at the client may be overmodulated. Check the received signal strength at the client (dBm). It may be necessary to reduce the transmit power depending on the antennas being used to avoid exceeding the maximum legal transmit power. Reducing the transmit power effectively reduces cell size.

  Note

  The maximum possible transmit power varies depending on the channel and data rate. For more detailed information on transmit power, refer to the documentation "Characteristics radio interface".

  Note

  If both interfaces of access points with two WLAN interfaces are operated in the same frequency range, this may cause wireless interference on one or both interfaces at a transmit power higher than 15 dBm.

- **Tx power check (only available online)**
  Indicates whether the settings that have been made will violate the permitted transmit power restrictions of the selected country. The following parameters influence this calculation: max. Maximum Tx Power, Antenna Gain, Additional Attenuation.
  The following displays can appear:
  - Allowed
    The channels can be used with the current settings.
  - Not Allowed (Some Channels)
    Among the channels, there are some on which the current transmit power exceeds the maximum permitted transmit power.
  - Not Allowed (All Channels)
    No permitted operation is possible. The transmit power is too high.
  - Controlled automatically by iREF.
    Controlled by the iREF function.

Advanced

You only need to adapt the settings on this page if the device cannot be used as it is intended with the default settings.

Settings

- **Radio**
  Shows the available WLAN interfaces.
- **Beacon Interval [ms]** (Only in access point mode)
  Specify the interval (40 - 1000 ms) at which the access point sends beacons. Beacons are packets that are sent cyclically by an access point to inform clients of its existence.

- **DTIM**(Only in access point mode)
  The DTIM interval (1-15) specifies the number of beacons to be sent before the access point sends the collected packets (broadcast, unicast, multicast) to the client.
  - If you enter a "1", the access point transmits broadcast, unicast and multicast packets directly after each beacon (recommended setting for normal network environments).
  - If you entered a "5", this would mean that the access point collects the packets and sends them after every fifth beacon.

  Increasing this value allows a longer sleep mode for the clients but means a greater delay for packets.
- **RTS/CTS Threshold [Bytes]**
  RTS/CTS (Request To Send/Clear To Send) is a method for avoiding collisions. The method is based on the exchange of status information prior to sending the actual data (Hidden node problem). To minimize the network load due to additional protocol traffic, this method is used only as of a certain packet size. You specify the packet size with the RTS/CTS Threshold parameter.
- **Fragmentation Length Threshold [Bytes]**
  Specify the maximum packet size transferred on the wireless link. Large packets are divided up into small packets prior to transmission and then reassembled into the original size after they have been received. This can be beneficial if the transmission quality is poor because larger packets are more difficult to transmit. However fragmentation into smaller packets means a poorer throughput.

  Note

  You can only edit this value if the you have set the transmission standard "802.11g" (2.4 GHz) or "802.11 a" (5 GHz), see "WLAN Mode" in "Interfaces > WLAN > Basic".

- **HW Retries**
  Specify the number of hardware retries. The hardware repetition is performed by the WLAN chip itself when it tries to repeat an unacknowledged packet immediately.
  If all hardware repetitions were unsuccessful, the packet is deleted.

  Note

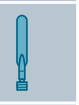
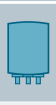
  If you have enabled iPCF or iPCF-MC, you can set up to 16 hardware retries.

- **Multi Radar Detection** (only in access point mode)
  - Enabled
    This function is only available if you have enabled the "DFS" function on the "Basic" page. This function is suitable for systems with several access points connected via an Ethernet network and that send on the same channel. When an access point detects a radar signal it distributes this information to all connected access points. If at least one further access point verifies the radar signal within 40 ms, all connected access points are informed. All the devices sending on this channel change to a different channel. The channel is blocked for 30 minutes for the access points in the network. If you have configured "Auto" for the channel on the "Interfaces > WLAN > AP" page, the function cannot be used reliably. In this case the verification of the radar signals is only possible when at least two connected access points happen to transmit on the same channel. If only one access point detects a signal on a channel, it treats this as a valid radar signal.
  - Disabled
    The function is not used. When an access point detects a radar signal it changes to another channel.. The configured channel is no longer taken into account.
- **Prefer Configured DFS Channel** (only in access point mode)
  - Enabled
    This function is only available if you have enabled the "DFS" function on the "Basic" page. If the configured channel of a WLAN interface was blocked due to radar detection and is released again after 30 minutes the access point changes automatically to the configured channel. Before the access point starts the communication on the configured channel it searches 60 seconds for primary users on the channel. During this time the access point does not send beacons. If signals are found on the channel, the access point changes channel and repeats the check. Only when no signals from primary users are detected after 60 seconds does the access point send on the channel. If you have configured "Auto" for the channel on the "Interfaces > WLAN > AP" page, the device does not have a configured channel to which it can return.
  - Disabled
    The function is not used.

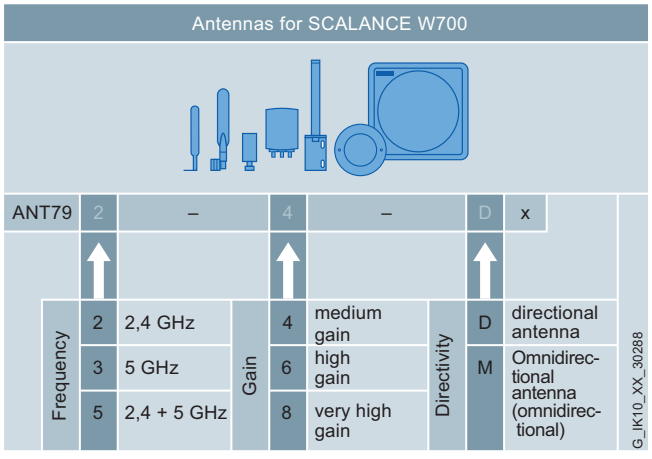
Antennas

Overview

Overview of IWLAN antennas:

Antennas only for SCALANCE W700 to IEEE 802.11n						Antennas for SCALANCE W700 to IEEE 802.11n and IEEE 802.11a/b/g		Antennas only for SCALANCE W700 to IEEE 802.11a/b/g	
with 1 connector		with 2 connectors (dual)		with 3 connectors (MIMO)		with 1 connector		with 1 connector	
	ANT795-4MA		ANT793-6DG		ANT795-6MT		ANT792-4DN		ANT795-4MR
	ANT795-4MC		ANT793-8DJ		ANT793-6DT		ANT793-4MN		ANT795-4MS
	ANT795-4MD		ANT793-8DK		ANT896-6MM		ANT792-6MN		ANT795-6DN
	ANT795-6DC		ANT793-8DL				ANT793-6MN		ANT793-8DN
	ANT793-8DP						ANT792-8DN		
	ANT795-4MX						ANT795-6MN		
	ANT795-MP						RCoax leaky feeder		

The antenna name provides information about the properties of the antennas listed in the IWLAN antenna overview:



Antennas

On this page, you configure the settings for the connected external antennas.

Note

Transmission disruptions without an antenna connected

A WLAN interface can have up to three antenna connectors. A the first connector, there must be always be an antenna connected as soon as the associated WLAN Interface is turned on.

Connectors that are not used must have a 50 Ω terminating resistor fitted.

If no antenna is connected, the relevant WLAN interface must also be disabled for Rx and Tx. Otherwise, there may be transmission disruptions.

Settings

- **Connector**
  Shows the name of the relevant antenna connection.
- **Antenna Type**
  Select the type of external antenna connected to the device. If the type of your external antenna is not available, select the entry "User defined".
  If you terminate an antenna connection using a 50 Ω terminating resistor, select the entry "Not used (Connect 50 Ohm Termination)".
- **Antenna Gain**
  If you select the "User defined" entry for the "Antenna Type", enter the antenna gain manually in the "dBi" unit.
  - **Antenna Gain 2.4 GHz [dBi]**
    Here, enter the antenna gain the antenna has in the 2.4 GHz frequency band.
  - **Antenna Gain 5 GHz [dBi]**
    Here, enter the antenna gain the antenna has in the 5 GHz frequency band.
- **Cable Length [m]**
  Enter the length of the flexible antenna connecting cable between the device and the external antenna in meters.

- **Additional Attenuation [dB]**
  Here, specify the additional attenuation caused, for example by an additional splitter.
- **Antenna Mode**
  Specify the use of the antenna. For antenna connector 1 (R1 A1 and R2 A1), the entry cannot be changed.
  - TX
    For sending only
  - RX
    For receiving only
  - RX\TX
    For receiving and sending

  The following table shows which combinations are possible:

R1 A1 / R2 A1	R1 A2 / R2 A2	R1 A3 / R2 A3
RX\TX	RX\TX	RX\TX
RX\TX	RX\TX	RX
RX\TX	RX	RX
RX\TX	RX\TX	TX
RX\TX	TX	TX
RX\TX	RX\TX	-- 1)
RX\TX	TX	-- 1)
RX\TX	RX	-- 1)
RX\TX	-- 1)	-- 1)

1) Antenna type "50 Ω termination"

Permitted channels

For communication, a specific channel within a frequency band is used. You can either set this channel specifically or configure so that the channel is selected automatically.

Settings

Table 1 contains the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **Use Allowed Channels only**
  If you enable the setting, you restrict the selection of channels via which the AP or the client is allowed to establish the connection.
  In the following tables, you define the
  - channels that the AP can use to establish a wireless cell when the "Auto" channel setting is enabled.
  - the channels on which the client searches for an AP.

  The tables are divided up according to frequency bands.
  When disabled, the channels available based on the settings (country code, antennas, transmit power etc.) are used.

Above the tables for the frequency bands, you will find the following setting:

- **Select / Deselect all**
  - Enabled
    All channels are activated.
  - Disabled
    The first valid channel of the frequency band remains enabled. Enable the required channel.

The tables of the frequency bands have the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **Radio Mode**
  Shows the mode.
- **Channel number**
  To specify the valid channels for the required frequency band, select the appropriate check box for the channel number.
  The table displays the permitted channels of the country. Only the valid channels can be enabled. Invalid channels are grayed out and cannot be enabled.

Note

To specify the channels, the setting "Use Allowed Channels only" must be enabled.

802.11n

Properties of 802.11n

With the IEEE 802.11n standard, it is possible to put together individual data packets in one larger data packet, the A-MPDU and A-MSDU data packets. This achieves a higher data throughput.
On this page, you make the settings for the A-MPDU and A-MSDU data packets. Some of the settings depend on the set transmission standard and the selected channel width.

Settings

The table has the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **A-MPDU**
  Enables or disables the sending of A-MPDU data packets. If this setting is disabled, only A-MPDU data packets are received.
- **A-MPDU Limit [frames]**
  Specify the number of individual subframes grouped together in one A-MPDU data frame.
- **A-MPDU Limit [Bytes]**
  Specify the maximum size of the A-MPDU frame.
- **A-MSDU**
  Aggregated MAC Service Data Unit (A-MSDU)Enables or disables that several MSDUs with the same destination address are bundled into an A-MSDU and are sent together. This reduces the network load. Due to their shorter maximum length A-MSDUs are more suitable for the bundling of several shorter frames. If this check box is disabled, A-MSDU data packets are received but not sent.
- **A-MSDUPacket Size [bytes]**
  Specify the maximum size of the A-MSDU data packet.
  Range of values: 50 - 200 bytes
  Default: 100 bytes
- **Guard Interval [ns](only in access point mode)**
  Select the send pause that must be kept to between two transmitted OFDM symbols. The following settings are possible. The selection depends on the selected transmission standard.
  - 400 (short)/800 (long): The setting 400 ns is optional. Depending on the signal quality, packets can be sent with a send pause of 400 ns or 800 ns.
  - 800 (long): The send pause is 800 ns.

Access point

Note

The page is available only in access point mode.

Settings

Table 1 has the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **Channel**
  Specify the main channel.
  If you want the access point to search for a free channel itself, use "Auto". The selection of channels used by an access point when establishing a wireless cell can be restricted. For "Channel selection", enable the setting "Use Allowed Channels only". If you want to use a fixed channel, select the required channel from the drop-down list.

  Note

  SCALANCE W780/W740: Channel spacing with WLAN interfaces

  If you use a second WLAN interface, make sure that you have adequate channel spacing.

- **Alternative DFS Channel (802.11h)**
  If you have enabled the "DFS" function on the "Basic" page, specify the alternative channel here. If you want the access point to search for a free channel itself, use "Auto".
  If a primary user was detected both on the main and alternative channel, the access point automatically searches for a free channel.
  If you want to use a fixed channel, select the required channel from the drop-down list.
- **HT Channel Width [MHz]**
  You can specify the channel bandwidth only with the IEEE 802.11n transmission standard. The following settings are possible.
  - 20
    Channel bandwidth 20 MHz
  - 40 up
    Channel bandwidth 40 MHz. The configured channel and the neighboring channel above it are used.
  - 40 down
    Channel bandwidth 40 MHz. The configured channel and the neighboring channel below it are used.

    Note

    Channel bandwidth 40 MHz and frequency band 2.4 GHz

    If the access point detects another access point on the configured channel or on neighboring channels, the access point changes the channel bandwidth from 40 MHz to 20 MHz. If you set a "free" channel on the access point, the access point uses the channel bandwidth 40 MHz.

Table 2 has the following columns:

- **Radio**
  Shows the available WLAN interfaces in this column.
- **Available Channels**
  Shows the permitted channels. The display depends on the wireless approvals of the currently selected country and the settings for "Channel selection".

Table 3 has the following columns:

- **Radio**
  Shows the WLAN interface.
- **Port**
  Shows the VAP interface.
- **Enabled**
  To use the required VAP interface, select this check box.
- **SSID**
  Enter the SSID of the WLAN. The length of the character string for SSID it is 1 to 32 characters.
  The ASCII characters 0x20 to 0x7e are used for the SSID.
- **Broadcast SSID**
  - deactived
    The SSID is no longer sent in the beacon frame of the access point. This means that the SSID is not visible for other devices. Only clients that know the SSID of the access point and that are configured with it can connect to the access point. The "Any SSID" option must be disabled on these clients.
  - activated
    The SSID is sent in the Beacon frame of the access point and is visible for other devices. This means that clients on which the "Any SSID" setting is enabled can also connect to the access point.

  Note

  Since no encryption is used for the SSID transfer, this function can only provide basic protection against unauthorized access. The use of an authentication method (for example WPA2 (RADIUS) or WPA2-PSK if this is not possible) provides higher security. You must also expect that certain end devices may have problems with access to a hidden SSID.

- **WDS only**
  If you enable this setting, the access point only supports communication via WDS. In WDS mode, all access points must use the same channel.
- **WDS ID**
  Enter the WDS ID. The WDS ID can be a maximum of 32 characters long.
  To establish a WDS connection, enter this WDS ID for the WDS partner.
  ASCII characters 0x20 to 0x7e are used for the WDS ID.

AP WDS

In normal operation, the access point is used as an interface to a network and communicates with clients. There are, however, situations in which several access points need to communicate with each other, for example to extend wireless coverage or to set up a wireless backbone. This mode is possible with WDS (Wireless Distributed System).

Note

This tab is available only in access point mode.

Settings

- **Radio**
  Shows the available WLAN interfaces.
- **Port**
  Shows the port.
- **Port enabled**
  Enables the WDS interface.
- **Connection over**
  Specify the VAP interface via which the WDS connection is established. Both the MAC address of the VAP as well as security settings (for example WPA2) are used.
- **Partner ID Type**
  Specify the type of WDS communication.
  - MAC Address
    The MAC address is used. The "Partner WDS ID" input box is grayed out. For "Partner MAC", enter the MAC address of the WDS partner.
  - WDS ID
    The WDS ID is used. The "Partner MAC" input box is grayed out. Enter the WDS ID of the WDS partner in "Partner WDS ID". Use this option if you want to replace the access point later using the C-PLUG or KEY-PLUG.
- **Partner MAC**
  Enter the MAC address of the WDS partner. Can only be edited if "MAC Address" is set for the "Partner ID Type".
- **Partner WDS ID**
  Enter the WDS ID of the WDS partner. Can only be edited if "WDS ID" is set for the "Partner ID Type". For the WDS ID, the ASCII characters 0x20 to 0x7e are permitted.

Note

Matching security settings in WDS mode

In WDS mode, make sure that the security settings match up for all devices involved. If settings are incorrect or not compatible on the individual devices, no data exchange is possible due to incorrect authentication. Avoid the "Auto" setting in the security settings of the Basic Wizard, because with this setting, synchronization of the security settings between the access points is not possible.

Note

In WDS operation, the following restrictions apply to all access points involved:

- All access points that will communicate with each other must use the same channel, the same transmission procedure and the same data rate.
- You can select either WEP or WPA(2)-PSK as the encryption method.
  You configure the security settings in the assigned VAP interface: "Security > WLAN > Basic"
  You cannot use authentication with a RADIUS server for a WDS connection.
- In the IEEE 802.11h transmission mode, it is not practical to select the WDS mode. In WDS mode, all access points must use the same channel. If a signal from a primary user is detected by an access point, the channel is changed automatically and the existing connection is then terminated.

AP 802.11a/b/g Rates

Note

The page is only

- available in access point mode.
- Can be configured if "802.11a", "802.11g" or "802.11n" is set for the WLAN mode.

The page shows the available data transmission speeds for the WLAN mode 802.11a/b/g. If necessary, you can change the data transmission speeds. Otherwise, we recommend that you retain the default setting for data transmission speeds. The access point will then use only the selected data transmission speeds for communication with the clients.

Settings

Table 1 has the following columns:

- **Radio**
  Specifies the WLAN interface to which the information relates.
- **Use selected data rates only**
  If you select this setting, you can specify the data transmission speeds for the required WLAN interface.
  If this setting is disabled, the default values are used.

Radio
Select the WLAN interfaces displayed in Table 3 (Data rate).

Table 2 has the following columns:

- **All data rate settings**
  Shows that the setting is valid for all entries in Table 3.
- **Enabled / Basic settings**
  Select the required setting. If "No Change" is selected, the entry in table 3 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 3

Table 3 (Data Rate) has the following columns:

- **Radio**
  Specifies the WLAN interface to which the information relates.
- **Data Rate [Mbps]**
  Shows the supported data transmission speeds in megabits per second.

- **Enabled**
  Enable the setting to assign the required data transmission speed to the WLAN interface.

  Note

  You need to enable at least one data transmission speed.

- **Basic**
  Enable the option to declare the required data transmission speed as "Basic". The setting specifies that a client must be capable of the required speed to be able to connect to the access point. The setting can only be enabled if an available data rate has been selected.

  Note

  At least one data transmission speed needs to be specified as "Initial Setup".

Default Values
The "Default Values" button sets the selection of values to conform with the standard.

AP 802.11n data rates

Note

The page is only

- available in access point mode.
- Can be configured if "802.11n only" or "802.11n" is set for the WLAN mode.

The page shows the available data transmission speeds (MCS = Modulation and Coding Schemes) for the WLAN mode 802.11n. You can select any combination of these data transmission speeds. The access point will then use only the selected data transmission speeds for communication with the clients.

Settings

Table 1 has the following columns:

- **Radio**
  Specifies the WLAN interface to which the information relates.
- **Use selected data rates only**
  If you select this setting, you can specify the data transmission speeds for the required WLAN interface.
  If this setting is disabled, the default values are used. As default, the setting is disabled.

Radio
Select the WLAN interfaces displayed in Table 3 (MCS Index).

Table 2 has the following columns:

- **All data rate settings**
  Shows that the setting is valid for all entries in Table 3.
- **Enabled**
  Select the required setting. If "No Change" is selected, the entry in table 3 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 3.

Table 3 (MCS Index) consists of the following columns:

- **Radio**
  Specifies the WLAN interface to which the information relates.
- **MCS Index**
  Shows the supported MCS indexes. The displayed MCS indexes depend on the settings "Antenna Type" and "Antenna Mode". You will find the settings in "Interfaces > WLAN > Antennas". If, for example, you only use one antenna, only the MCS 0 to 7 are displayed.
- **Streams**
  Shows the maximum possible number of parallel data streams that can be transmitted with the selected MCS index.
- **Data Rate [Mbps]**
  Shows the supported data transmission speeds in megabits per second. The displayed data transmission speeds depend on the settings "Guard Interval" and "HT Channel Width". The "HT Channel Width" setting can be found in "Interfaces > WLAN > Access Point". The "Guard Interval" setting can be found in "Interfaces > WLAN > 802.11n"
- **Enabled**
  Enable the setting to assign the required data transmission speed to the WLAN interface.

Note

You need to enable at least one MCS index.

Default Values
The "Default Values" button sets the selection of values to conform with the standard.

Client

Connecting to a network

On this page, you can specify how the device connects to a network as client.

Note

This page is only available for clients or access points in client mode.

Settings

Table 1 has the following columns:

- **Radio**
  Shows the available WLAN interfaces.

  Note
  WLAN interface disabled

  The WLAN interface will be disabled unless at least one SSID is configured or the setting "Any SSID" is enabled.

- **MAC Mode**
  Specify how the MAC address is assigned to the client. The following are possible:
  - Automatic
    The client automatically adopts the source MAC address of the first frame that it receives over the Ethernet interface.
  - Manual
    If you select Manual, enter the MAC address in the "MAC Address" column.
  - Own
    The client uses the MAC address of the Ethernet interface for the WLAN interface.
  - Layer 2 Tunnel
    The client uses the MAC address of the Ethernet interface for the WLAN interface. The network is also informed of the MAC addresses connected to the Ethernet interface of the client. Up to eight MAC addresses can be used.
- **MAC Address**
  If you have selected "Manual" for "MAC Mode", enter the MAC address of the client.
- **Any SSID**
  - Enabled
    In client mode the SCALANCE W700 device attempts to connect to the access point that corresponds to the security settings of security context 1. The clients can only connect to the access point on which the "Broadcast SSID" option is enabled.
  - Disabled
    The client attempts to connect to the access point from the SSID list whose security settings match one of the defined security contexts.
- **DHCP Renew After Roaming**
  - Enabled
    After changing to a different access point, a check is made to find out whether the IPv4 address of the client is still valid. If he IPv4 address is invalid, a new IPv4 address is requested from the DHCP server.
  - Disabled
    If the client changes to a different access point the IPv4 address is not checked.

- **Roaming Threshold**
  Specify the threshold after which the client roams to the new access point.
  - High
    Changes only at a significantly higher field strength to the AP with the stronger signal.
  - Medium
    Changes at a moderately higher field strength to the AP with the stronger signal.
  - Low
    Changes at a slightly higher field strength to the AP with the stronger signal.
- **Background Scan Mode**
  While the client is connected to an access point, it scans for other access points in the background with which it can connect when necessary. Specify the mode for the scan. The following options are available:
  - always
    If the background scan threshold is undershot, the client searches continuously for access points.
  - idle
    If there is no data transfer for a certain time, a scan is started for further access points.
  - disabled
    As long as the client is connected, there is no scan for further access points.
- **Background Scan Interval [ms]**
  Specify the interval at which further access points are scanned.
- **Background Scan Threshold [dBm]**
  Specify the threshold. If the threshold is undershot, the client searches for further access points.

Table 2 has the following columns:

- **Radio**
  Shows the WLAN interface.
- **Scan Channels**
  Shows the channels on which the client searches for an access point. The display depends on the wireless approvals of the selected country and the settings for "Channel selection".

Table 3 has the following columns:

- **Enabled**
  Enables or disables the relevant SSID.
- **Radio**
  Shows the WLAN interface.

- **SSID**
  Enter the SSID of the access point with which the client will connect.
  For the SSID, ASCII code 0x20 to 0x7e is used.
- **Security**
  Select a security context. You create and configure a security context in "Security > WLAN > Basic".
  Default setting: Context 1

Note

iPCF or iPCF-MC mode activated

If the iPCF or iPCF-MC mode is enabled, you can only select security context 1.

Force Roaming

Note

The page is only available

- in access point mode
- if there is an online connection to the device.

The WLAN interface of the device must be enabled, otherwise the frequency ranges cannot be scanned.

If the interface is no longer available (cable break, network component faild, connector removed), a client connected over the wireless network is not aware of this. The access point can force the logged-on clients to roam by deactivating the relevant interface.

On this page you specify when roaming is performed.

- On connection abort
  If the wired Ethernet interface is no longer available the WLAN interface is turned off. The clients roam and then connect to a different access point. As soon as the first access point reaches the server again it switches its WLAN interfaces active again.
- When the destination address is unreachable.
  Roaming depends on destination addresses. To monitor the device sends pings to the configured destination addresses at regular intervals.
  - VAP interface monitored using a destination address
    If no ping response is received from this destination address, the access point turns the relevant VAP interface off.
  - VAP interface monitored using several destination addresses
    Only when no ping response is received from any of these destination addresses does the access point turn the relevant VAP interface off. As long as at least one destination address is reachable, the VAP interface remains active.

  The access point sends a disassociation frame to the WAN clients connected via this VAP interface. The WLAN clients roam and connect to a different VAP interface. If the address becomes reachable again, the connection can be established again via this VAP interface.

Settings

Table 1 has the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **Force roaming on link down**
  When enabled if there is a connection abort via the Ethernet interface, the WLAN interface is turned off.

The table "Force Roaming on IP down"has the following columns:

- **Select**
  Select the check box in the row to be deleted.
- **Destination Address**
  Enter the IPv4 address or the FQDN (Fully Qualified Domain Name) of the destination whose reachability will be checked.

  Note

  Destination address not in the agent IP subnet

  If the destination address is not in the agent IP subnet, a gateway must be entered for "Layer 2 > Agent IP".

  The Base Bridge mode "802.1Q VLAN Bridge"

  If you have configured the "Base Bridge Mode" "802.1Q VLAN Bridge" in "Layer 2 > VLAN", echo messages (ping packets) are sent into the management VLAN.

- **Interval [ms]**
  Specify the interval at which echo messages (pings) are sent.
- **Max. Lost Packets**
  Specify the maximum number of consecutive lost ping responses. When this number is reached for a destination address, this destination address counts as being unreachable (down).
- **VAP X.Y**
  Specify the VAP interface via which the address is monitored.

Signal recorder

Recording the effective user signal

The signal recorder is used to record the effective user signal between access point and client. Using this data, you can locate areas with an inadequate user signal. The signal recorder can be particularly useful when the client moves along a fixed path.

Note

The page is only available

- for clients or access points in client mode
- if there is an online connection to the device.

The WLAN interface of the device is enabled, otherwise no recording will be made.

Description

The display is divided into two areas.

- Client
  Displays the measurement on the client.
- Access point
  Displays the measurement of the access point with which the client is currently connected. This requires that the setting "Bidirectional Recording" is enabled and that a firmware version > 6.1 is installed on the access point. The access point sends its data to a maximum of 3 clients on which signal recorders are running. The access point data is not displayed on other clients.

Both areas each contain two graphics.

The first graphic contains the following elements:

- Scroll bar
  With the scroll bar, you can look through the entire measurement. To do this you can use the "<<" and ">>" buttons or the arrow keys on the keyboard.
- Bar (left)
  In the bar on the left-hand side the user signal of the client / access point is displayed in real time according to the color scheme. The gray line shows the background noise.
  If the client has an iPCF-MC connection, the user signal of the management channel is shown with a black line.
- Color scheme
  The range > -35 dBm (blue) is the overmodulation range, in other words the WLAN signal is too strong and is received overmodulated. As of approximately -60 dBm (yellow) the WLAN signal is weaker.

- x axis
  The x axis shows the course of the measurement in random samples and seconds.
- Measurement data
  - Client
    The measurement data shows the value of the effective user signal according to the color scheme shown. The gray line shows the background noise.
    If the client changes access points during a measurement (roaming) or reconnects, this is displayed by a vertical black line. On the line the previous AP system name and the BSSID are shown.
    If during a measurement the client has no connection to an access point, no user signal is displayed. To make it clear that there is no connection to an access point, the BSSID is set to 00:00:00:00:00:00 and shown in red.
    If the client has an iPCF-MC connection, the user signal of the management channel is shown with an additional black line.
  - Access point
    The measurement data shows the value of the effective user signal according to the color scheme shown. The gray line shows the background noise.
    If the client changes access points during a measurement (roaming) or reconnects, this is displayed by a vertical black line.
    If the access point does not support the setting "Bidirectional Recording" no user signal is displayed.

The second graphic contains the following elements:

- Bar (left)
  In the bar on the left-hand side the transfer attempts and the data rate of the client / access point are displayed according to the color scheme.
- Color scheme
  The range > -35 dBm (blue) is the overmodulation range, in other words the WLAN signal is too strong and is received overmodulated. As of approximately -60 dBm (yellow) the WLAN signal is weaker. The individual colors are described again under the graphic.
- x axis
  The x axis shows the course of the measurement in random samples and seconds.
- Measurement data
  - Client
    The measurement data shows the transfer attempts according to the color scheme shown. The transfer attempts are shown as a bar. The data rate of the sent data packets is represented as a line. If the client changes access points during a measurement (roaming) or reconnects, this is displayed by a vertical black line.
  - Access point
    The measurement data shows the transfer attempts according to the color scheme shown. The transfer attempts are shown as a bar. The data rate of the sent data packets is represented as a line.
    If the client changes access points during a measurement (roaming) or reconnects, this is displayed by a vertical black line. If the access point does not support the setting "Bidirectional Recording" no data is displayed

Beside the graphics the following values are displayed:

- Current Sample
  The number of the current measurement
- CL Signal [dBm] / AP Signal [dBm]
  The effective user signal of the client / access point in dBm
- CL Noise Floor [dBm] / AP Noise Floor [dBm]
  The background noise of the client / access point in dBm
- CL Retries [%] / AP Retries [%]
  The transfer repetitions of the client / access point as a percentage.
- CL RSSI / AP RSSI
  The raw value of the RSSI (Received Signal Strength Indication) of the client / access point
- CL TX Rate [Mbps] / AP TX Rate [Mbps]
  The average data rate of the sent data packets during the current random test
- AP System Name
  The system name of the access point
- Client Count
  Number of clients connected to the access point.
- BSSID
  The BSSID (Basic Service Set Identification)
- Operative Channel
  The current channel or the channel on which the client is connected to the access point
- Roaming Counter
  The roaming counter shows how often the client has changed access points. After 99999 changes the counter is reset.

Settings

This table contains the following columns:

- **Radio**
  Shows the available WLAN interfaces. Since a client has a WLAN interface, there is only ever one row for "WLAN 1" in this table.
- **Interval [ms]**
  Specify the time interval between acquiring two measured values in milliseconds.
- **Samples**
  Specify how many measurements should be made.
- **Endless**
  If you enable the option checkmark, the number of measurements is unlimited The "Samples" box is grayed out . The signal recorder runs until it is stopped manually or the device is reconfigured.
  You can only select this option starting at a time interval ≥ 10 milliseconds.
- **Bidirectional Recording**
  If you enable the setting the values of the access point are recorded as of a time interval of ≥ 10 milliseconds.

- **Start**
  Click the button in this column to start recording the user signal.

  Note
  - If you start a new recording, the previous recording will be overwritten.
  - If the recording has lasted less than 10 minutes and has not yet been completed (e.g. due to a restart or power down), the measured values are deleted.

  The signal recorder saves the recorded data automatically every 10 minutes. Following a restart, the recording contains all the values up to the last save action.
- **Stop**
  Click the button in this column to stop recording the user signal prematurely. If the specified number of measurements has been made, recording of the user data signal stops automatically.
- **Displayed samples**
  Select how many measured values will be shown in the graphic.

Notes on usage

Note the following tips that will help you to obtain useful measurements with the signal recorder:

- Set a fixed data rate on the access point.
- If you have activated iPCF, set as low a cycle time on the access point as possible for the measurements.
- Make sure that there is enough data communication during the measurement because the statistics functions evaluate incoming data frames.
- The measurement path should be traveled 2 to 3 times with the same parameters to find out whether losses of signal strength always occur at the same position.
- Selective measurements at a fixed position should be made over a longer period of time.

Note
- If you start a new recording, the previous recording will be overwritten.
- When the device restarts, the measured values are deleted.

Change to one of the following menu items to call up the result of the recording:

- System > Load&Save > HTTP
  Click the "Save" button in the "WLANSigRec" table row to save the file "signal_recorder_SCALANCE_W700.zip" in the file system of the connected PC.
- System > Load&Save > TFTP
  If necessary, change the file name "signal_recorder_SCALANCE_W700.zip" in the "WLANSigRec" table row. In the table row "WLANSigRec", select the "Save file" entry from the drop-down list of the last column and click the "Save Values" button.

The downloaded ZIP file contains two files with the results of the recording:

- A PDF file
- A CSV file

Spectrum Analyzer

Note

The page is only available

- in access point mode
- if there is an online connection to the device.

The WLAN interface of the device must be enabled, otherwise the frequency ranges cannot be scanned.

Technical information

The frequency range depends on the configuration.

Parameters		Value
Amplitude accuracy	In 2.4 GHz	3 dBm
	In 5 GHz	7 dBm
Resolution bandwidth		330 KHz
Min. signal strength		-100 dBm
Max. signal strength		0 dBm
Analysis time	At 40 MHz	120 ms
	At 20 MHz	95 ms
Update time		1 s.

Representing signals of the frequency range

Which the spectrum analyzer you can recognize and represent the electromagnetic signals of a frequency range. You can measure the strength of all signals located in the environment of the access point.

Note

When the spectrum analyzer is started, all WLAN connections are terminated on both WLAN interfaces. The access point then also does not send any beacons.

Note

Do not enable the spectrum analyzer if the device is operating productively. This can influence the performance of the device.

Note

The functionality of the spectrum analyzer does not replace a dedicated spectrum analyzer.

Description

The page contains the following graphics:

In all graphics, the lower x axis shows the channels around the selected center frequency for which the measurements are made. The upper x axis shows the frequency range. The display of the y axis depends on the selected graphic.

- Realtime
  The y axis shows the signal strength in dBm.
  The graphic shows the strength of all signals that the access point receives in its environment in the configured frequency range.
  The red line shows the maximum values since the start of the measurement. The white line shows the current values. The green line shows the average values.
- Spectrogram
  The y axis shows the course of the measured values over time from current (0 s) to the values received before 500 s.
  The graphic shows the strength of all signals that the access point receives in its environment in the configured frequency range.
  The color depends on the setting for "Color Scheme".
- Density Chart
  The y axis shows the signal strength in dBm.
  The graphic shows how often signals occur with a certain strength in the configured frequency range.
  The color goes from the lowest value (0%) in black to the highest value (100%) in red.

The page contains the following buttons:

- Expand 
  With this icon you only show one graphic type in large format on the page.
- Reduce 
  With this icon you return to the view with all three graphic types.
- Color Scheme 
  With this icon, you change the colours scheme for the graphic type "Spectrogram":
  - The color goes from the lowest value (-100 dBm) in black to the highest value (0 dBm) in red.
  - The color goes from the lowest value (-100 dBm) in red to the highest value (0 dBm) in black.
- Reset 
  With this icon you reset the maximum and average values of the graphic type "Realtime".

This table contains the following columns:

- Radio
  Shows the WLAN interface to which the information applies.
- State
  Shows the status of the measurement. The following values are possible:
  - Stopped
    The measurement was stopped.
  - Started
    The measurement is running.
- Frequency Band
  Specify the frequency band.

- Center Frequency
  Select the center frequency.
  How many channels are displayed depends on the setting for the "HT Channel Width", refer to the page "Interfaces > WLAN > AP".
- Start
  Click the button in this column to start the measurement.

  Note

  If you start a new measurement, the previous measurement will be overwritten.

- Stop
  Click the button in this column to end the measurement.

You can save the measurement result as a CSV file with the menu “System > Load&Save > WLANSpectrumAnalyzer”.

Measurement results

CSV file

The CSV file contains information on the configuration of the device and detailed information on all individual measurements and is divided into two areas. The first area contains the configured settings:

- System Name
  The system name of the access point
- Device IP
  The IP address of the device
- Device MAC
  The MAC address of the device
- Recording Interval
  The interval between acquisition of two measured values

The second area is a table. The table contains the following for each measured value:

- Sample
  The consecutive number of the measurement
- Timestamp
  The time stamp
- The following columns show all frequencies of the selected frequency band. The cells are only filled for the frequencies for which a value was measured. The measured values show the signal strength in dBm.

Configuring layer 2 functions

VLAN

General

On this page you specify whether or not the device forwards frames with VLAN tags transparently (IEEE 802.1D/VLAN-unaware mode) or takes VLAN information into account (IEEE 802.1Q/VLAN-aware mode). If the device is in the "802.1Q VLAN Bridge" mode, you can define VLANs and specify the use of the ports .

Note

Changing the agent VLAN ID

If the configuration PC is connected directly to the device via Ethernet and you change the agent VLAN ID, the device is no longer reachable via Ethernet following the change.

Settings

The page contains the following boxes:

- **Base Bridge-Mode**
  Select the required mode from the drop-down list. The following modes are possible:

  Note

  Changing Base bridge mode

  Note the section "Changing Base bridge mode". This section describes how a change affects the existing configuration.

  – 802.1 D Transparent Bridge
    Sets the mode "VLAN-unaware" for the device. In this mode, VLAN tags are not taken into account or changed but are forwarded transparently. In this mode, you cannot create any VLANs.
  – 802.1Q VLAN Bridge
    Sets the mode "VLAN-unaware" for the device. In this mode, VLAN tags are not taken into account or changed but are forwarded transparently. In this mode, you cannot create any VLANs.
    Only a management VLAN is available: VLAN 1.

- **VLAN ID**
  Enter the VLAN-ID (a number between 1 and 4094).
  You can create up to 2 (SCALANCE W760), 8 (SCALANCE W770) or 24 (SCALANCE W780) VLANS.

The table has the following columns:

- VLAN ID
  Shows the VLAN ID. The VLAN ID (a number between 1 and 4094) can only be assigned once when creating a new data record and can then no longer be changed. To make a change, the entire data record must be deleted and created again.
- Name
  Enter a name for the VLAN. The name only provides information and has no effect on the configuration.
- Status
  Shows the status type of the entry in the internal port filter table. Here, static means that the address was entered as a static address by the user. The entry GVRP means that the configuration was registered by a GVRP frame. This is, however, only possible if GVRP was enabled for the device.
- List of ports
  Specify the use of the port. The following options are available:
  - "-"
  - The port is not a member of the specified VLAN.
    With a new definition, all ports have the identifier "-".
  - M
    The port is a member of the VLAN. Frames sent in this VLAN are forwarded with the corresponding VLAN tag.
  - R
    The port is a member of the VLAN. A GVRP frame is used for the registration.
  - U (upper case)
    The port is an untagged member of the VLAN. Frames sent in this VLAN are forwarded without the VLAN tag. Frames without a VLAN tag are sent from this port.
  - u (lower case)
    The port is an untagged member of the VLAN, but the VLAN is not configured as a port VLAN. Frames sent in this VLAN are forwarded without the VLAN tag.
  - F
    The port is not a member of the specified VLAN and it is not possible for the VLAN to be registered dynamically at this port using GVRP. You can configure other settings in "Layer 2 > VLAN > Port-based VLAN".

Changing Base bridge mode

VLAN-unaware (802.1D transparent bridge) → VLAN-aware (802.1Q VLAN bridge)

If you change the Base Bridge mode from VLAN-unaware to VLAN aware, this has the following effects

- All static and dynamic unicast entries are deleted.

VLAN-aware (802.1Q VLAN bridge) → VLAN-unaware (802.1D transparent bridge)

If you change the Base Bridge mode from VLAN-aware to VLAN-unaware, this has the following effects:

- All VLAN configurations are deleted.
- A management VLAN is created: VLAN 1.
- All static and dynamic unicast entries are deleted.

802.1Q VLAN Bridge: Important rules for VLANs

Make sure you keep to the following rules when configuring and operating your VLANs:

- Frames with the VLAN ID "0" are handled as untagged frames but retain their priority value.
- As default, all ports on the device send frames without a VLAN tag to ensure that the end node can receive these frames.
- With SCALANCE W devices, the VLAN ID "1" is the default on all ports.
- If an end node is connected to a port, outgoing frames should be sent without a tag (static access port). If, however, there is a further switch at this port, the frame should have a tag added (trunk port).

Port-based VLAN

Processing received frames

On this page, you specify the configuration of the port properties for receiving frames.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the settings are valid for all ports.
- **Priority / Port VID / Acceptable Frames / Ingress Filtering**
  Select the setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Port**
  Shows the available ports and link aggregations.
- **Priority**
  Select the priority assigned to untagged frames.
  The CoS priority (Class of Service) used in the VLAN tag. If a frame is received without a tag, it will be assigned this priority. This priority specifies how the frame is further processed compared with other frames.
  There are a total of eight priorities with values 0 to 7, where 7 represents the highest priority (IEEE 802.1p Port Priority).

- **Port VID**
  Select the VLAN ID. Only VLAN IDs defined on the "VLAN > General" page can be selected. If a received frame does not have a VLAN tag, it has a tag with the VLAN ID specified here added to it and is sent according to the rules at the port.
- **Acceptable Frames**
  Specify which types of frames will be accepted. The following alternatives are possible:
  - Tagged Frames Only
    The device discards all untagged frames. Otherwise, the forwarding rules apply according to the configuration.
  - All
    The device forwards all frames
- **Ingress Filtering**
  Specify whether the VID of received frames is evaluated.
  The following options are available:
  - Enabled
    The VLAN ID of received frames decides whether they are forwarded: To forward a VLAN tagged frame, the receiving port must be a member in the same VLAN. Frames from unknown VLANs are discarded at the receiving port.
  - Disabled
    All frames are forwarded.

Dynamic MAC aging

The device automatically learns the source addresses of the connected nodes.

This information is used to forward data frames to the nodes specifically involved. The network load for the other nodes is reduced.
If a device does not receive a frame whose source address matches a learnt address within a certain time, the learnt address is deleted. This mechanism is known as aging. Aging prevents frames being forwarded incorrectly, for example when an end device (for example a programming device) is connected to a different port. If the check box is not enabled, a device does not delete learned addresses automatically.

Settings

- **Dynamic MAC Aging**
  Enable or disable the function for automatic aging of learned MAC addresses:
- **Aging Time [s]**
  Enter the time in seconds. After this time, a learned address is deleted if the device does not receive any further frames from this sender address.

Spanning Tree

General

General settings of MSTP

This is the basic page for spanning tree. Select the compatibility mode from the drop-down list. As default, Multiple Spanning Tree is enabled.

On the configuration pages of these functions, you can make detailed settings.

Note

Client device not as root bridge

Using the configuration of priorities and path costs, make sure that a client device can never become the root bridge. If a client device becomes the root bridge the Rapid Spanning Tree function no longer works.

Depending on the compatibility mode, you can configure the corresponding function on the relevant configuration page.

Settings

- **Spanning Tree**
  Enable or disable Spanning Tree.

  Note

  Clients or access points in client mode

  If "Automatic" is set for "MAC Mode", spanning tree is not supported.

- **Protocol Compatibility**
  Select the compatibility mode of Spanning Tree. If, for example, you select RSTP, Spanning Tree behaves like RSTP.
  The following settings are available:
  - STP
  - RSTP
  - MSTP

  Note

  If iPCF mode is enabled, only the compatibility modes STP and RSTP are supported.

CIST general

The page consists of the following parts:

- The left-hand side of the page shows the configuration of the device.
- The central part shows the configuration of the root bridge that can be derived from the spanning tree frames received by a device.
- The right-hand side shows the configuration of the regional root bridge that can be derived from the MSTP frames received by a device. The displayed data is only visible if you have enabled "Spanning Tree" on the "General" page and if "MSTP" is set for "Protocol Compatibility". This also applies to the "Bridge Max Hop Count" parameter. If the device is a root bridge, the information on the left and right matches.

Settings

- **Bridge Priority / Root Priority** (only available online)
  Which device becomes the root bridge is decided based on the bridge priority. The bridge with the highest priority becomes the root bridge. The lower the value, the higher the priority. If several devices in a network have the same priority, the device whose MAC address has the lowest numeric value will become the root bridge. Both parameters, bridge priority and MAC address together form the bridge identifier. Since the root bridge manages all path changes, it should be located as centrally as possible due to the delay of the frames.
- **Bridge Address** / **Root Address**(only available online)
  The bridge address shows the MAC address of the device and the root address shows the MAC address of the root bridge.
- **Root Port** (only available online)
  Shows the port via which the switch communicates with the root bridge.
- **Root Cost** (only available online)
  The path costs from this device to the root bridge.
- **Topology Changes** / **Last Topology Change** (only available online)
  The entry for the device shows the number of reconfigurations due to the spanning tree mechanism since the last startup. For the root bridge, the time since the last reconfiguration is displayed as follows:
  - Seconds: "Sec" unit after the number
  - Minutes: "Min" unit after the number
  - Hour: "Hr" unit after the number
- **Bridge Hello Time / Root Hello Time** (only available online)
  Each bridge regularly sends configuration frames (BPDUs). The interval between two such frames is the Hello time. The default for this parameter is 2 seconds.
- **Bridge Forward Delay [s] / Root Forward Delay [s]**
  New configuration data is not used immediately by a bridge but only after the period specified in the parameter. This ensures that operation is only started with the new topology after all the bridges have the required information.
- **Bridge Max Age / Root Max Age** (only available online)
  Bridge Max Age defines the maximum "age" of a received BPDU for it to be accepted as valid by the switch.

The following values are available only in access point mode:

- **Regional root priority** (only available online)
  For a description, see Bridge Priority / Root Priority
- **Regional root address** (only available online)
  The MAC address of the device.
- **Regional root costs** (only available online)
  The path costs from this device to the root bridge.
- **Bridge Max Hop Count**
  This parameter specifies how many MSTP nodes a BPDU may pass through. If an MSTP BPDU is received and has a "Bridge Max Hop Count" that exceeds the value configured here, it is discarded.
- **Region Name**
  Enter the name of the MSTP region to which this device belongs. As default, the MAC address of the device is entered here. This value must be the same on all devices that belong to the same MSTP region.
- **Region Version**
  Enter the version number of the MSTP region in which the device is located. This value must be the same on all devices that belong to the same MSTP region.
- **Layer-2 Tunnel Admin Edge Port** (only available in access point mode)
  Enable this setting if there can be an end device on a layer 2 tunnel port. Otherwise a reconfiguration of the network will be triggered whenever a link to this port is modified. The L2T clients should be interconnected.
- **Layer-2 Tunnel Auto Edge Port**(Only available in access point mode)
  Enable this setting if you want to detect automatically at all layer 2 tunnel ports whether or not an end device is connected.

To reset the counters on this page the following button is available:

- **Reset Counters** (only available online)
  The values are reset with this button.

CIST port

MSTP-CIST port configuration

When the page is called, the table displays the current status of the configuration of the port parameters.

To configure them, click the relevant cells in the port table.

Settings

Table 1 has the following columns:

- **1st column**
  Shows that the setting is valid for all ports of table 2.
- **Spanning Tree Status**
  Select the setting. If "No Change" is selected, the entry in table 2 remains unchanged.

Table 2 has the following columns:

- **Port**
  Shows the available ports and interfaces.
  - Port X
  - WLAN X
  - VAP X.Y
  - WDS X.Y
- **Spanning Tree Status**
  Specify whether or not the port is integrated in the spanning tree.

  Note

  If you disable the "Spanning Tree Status" setting for a port, this may cause the formation of loops. The topology must be kept in mind.

- **Priority**
  Enter the priority of the port. The priority is only evaluated when the path costs are the same. The value must be divisible by 16. If the value that cannot be divided by 16, the value is automatically adapted.
- **Cost Calc.**
  Enter the path cost calculation. If you enter the value "0" here, the automatically calculated value is displayed in the "Path Costs" box.

- **Path Costs** (only available online)
  This parameter is used to calculate the path that will be selected. The path with the lowest value is selected as the path. If several ports of a device have the same value for the path costs, the port with the lowest port number is selected.
  If The "Cost Calc." is "0", the automatically calculated value is shown. Otherwise, the value of the "Cost Calc." box is displayed. The calculation of the path costs is largely based on the transmission speed. The higher the achievable transmission speed is, the lower the value of the path costs.
  Typical values for path costs with rapid spanning tree:
  - 10,000 Mbps = 2,000
  - 1000 Mbps = 20,000
  - 100 Mbps = 200,000
  - 10 Mbps = 2,000,000

  The values can, however, also be set individually.
- **Status** (only available online)
  Displays the current status of the port. The values are only displayed and cannot be configured. The "Status" parameter depends on the configured protocol. The following is possible for status:
  - Disabled

    The port only receives and is not involved in STP, MSTP and RSTP.
  - Discarding
    In the "Discarding" mode, BPDU frames are received. Other incoming or outgoing frames are discarded.
  - Listening

    In this status, BPDUs are both received and sent. The port is involved in the spanning tree algorithm.
  - Learning

    Stage prior to the forwarding status, the port is actively learning the topology (in other words, the node addresses).
  - Forwarding

    Following the reconfiguration time, the port is active in the network; it receives and forwards data frames.

- **Fwd. Trans** (only available online)
  Specifies the number of changes from the "Discarding" status to the "Forwarding" status
- **Edge type**
  Specify the type of edge port. You have the following options:
  - "-"
    Edge port is disabled. The port is treated as a "no edge port".
  - Admin
    Select this option when there is always an end device on this port. Otherwise a reconfiguration of the network will be triggered each time a connection is changed.
  - Auto
    Select this option if you want a connected end device to be detected automatically at this port. When the connection is established the first time, the port is treated as a "no Edge Port".
  - Admin/Auto
    Select these options if you operate a combination of both on this port. When the connection is established the first time, the port is treated as an Edge Port.
- **Edge** (only available online)
  Shows the status of the port.
  - Enabled

    An edge port is connected to this port.
  - Disabled
    There is a spanning tree or rapid spanning tree device at this port.

  With an end device, a switch can change over the port faster without taking into account spanning tree frames. If a spanning tree frame is received despite this setting, the port automatically changes to the "Disabled" setting for switches.
- **P.t.P. Type**
  Select the required option. The selection depends on the port that is set.
  - "-"
    Point to point is calculated automatically. If the port is set to half duplex, a point-to-point link is not assumed.
  - P.t.P.

    Even with half duplex, a point-to-point link is assumed.
  - Shared Media

    Even with a full duplex connection, a point-to-point link is not assumed.

  Note

  Point-to-point connection means a direct connection between two devices. A shared media connection is, for example, a connection to a hub.

- **P.t.P.**
  - Enabled
    Shows that a point-to-point link exists.
  - Disabled
    Shows that no point-to-point link exists.
- **Hello Time**
  Enter the interval after which the bridge sends configuration BPDUs

Note

The port-specific setting of the Hello time is only possible in MSTP compatible mode.

MST General

Multiple Spanning Tree configuration

With MSTP, in addition to RSTP, several VLANs can be managed in a LAN with separate RSTP trees.

Settings

- **MSTP Instance ID**
  Enter the number of the MSTP instance.

The table has the following columns:

- **Select**
  Select the line to be deleted.
- **MSTP Instance ID**
  Shows the number of the MSTP instance.
- **Root Address**
  Shows the MAC address of the root bridge
- **Root Priority**
  Shows the priority of the root bridge.
- **Bridge Priority**
  Enter the bridge priority. The value for the bridge priority is a whole multiple of 4096.
- **VLAN ID**
  Enter the VLAN ID. Here, you can also specify ranges with Start ID, "-", End ID. Several ranges or IDs are separated by ",".

MST port

Configuration of the Multiple Spanning Tree port parameters

On this page, you set the parameters for the ports of the configured multiple spanning tree instances.

Settings

- **MSTP Instance ID**
  Select the ID of the MSTP instance.

Table 1 has the following columns

- **1st column**
  Shows that the setting is valid for all ports.
- **MSTP Status**
  Select the setting. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2

Table 2 has the following columns:

- **Port**
  Shows all available ports and link aggregations.
- **MSTP Instance ID**
  ID of the MSTP instance.
- **MSTP Status**
  Enable or disable MSTP for this port.
- **Priority**
  Enter the priority of the port. The priority is only evaluated when the path costs are the same. The value must be divisible by 16. If the value that cannot be divided by 16, the value is automatically adapted.
- **Cost Calc.**
  Enter the path cost calculation in the input box. If you enter the value "0" here, the automatically calculated value is displayed in the next box "Path Costs".
- **Path Costs** (only available online)
  This parameter is used to calculate the path that will be selected. The path with the lowest value is selected as the path. If several ports of a device have the same value for the path costs, the port with the lowest port number is selected.
  If the value for the Cost Calc is "0" the automatically determined value is displayed. Otherwise, the value of the "Cost Calc box is displayed. The calculation of the path costs is largely based on the transmission speed. The higher the achievable transmission speed is, the lower the value of the path costs.
  Typical values for path costs with rapid spanning tree:
  - 10,000 Mbps = 2,000
  - 1000 Mbps = 20,000
  - 100 Mbps = 200,000
  - 10 Mbps = 2,000,000

  The values can, however, also be set individually.

- **Status** (only available online)
  Displays the current status of the port. The values are only displayed and cannot be configured. The following is possible for status:
  - Discarding
    The port exchanges MSTP information but is not involved in the data traffic.
  - Blocked
    In the blocking mode, BPDU frames are received.
  - Forwarding
    The port receives and sends data frames.
- **Fwd. Trans**(only available online)
  Specifies the number of status changes Discarding - Forwarding or Forwarding - Discarding for a port.

DCP forwarding

Applications

The DCP protocol is used by STEP 7 and the PST Tool for configuration and diagnostics. When shipped, DCP is enabled on all ports; in other words, DCP frames are forwarded at all ports. With this option, you can disable the sending of these frames for individual ports, for example to prevent individual parts of the network from being configured with the PST Tool or to divide the full network into smaller parts for configuration and diagnostics.

Note

Empty table

If you have enabled NAT on the device, the table is empty or will be emptied:

Settings

- **Port**
  Shows the available ports.
- **Setting**
  Specify whether the port should block or forward outgoing DCP frames. The following options are available:
  - block
    No outgoing DCP frames are forwarded via this port. It is nevertheless still possible to receive via this port.
  - forward
    DCP frames are forwarded via this port.

LLDP

Identifying the network topology

LLDP (Link Layer Discovery Protocol) is defined in the IEEE 802.AB standard.

LLDP is a method used to discover the network topology. Network components exchange information with their neighbor devices using LLDP.

Network components that support LLDP have an LLDP agent. The LLDP agent sends information about itself and receives information from connected devices at periodic intervals. The received information is stored in the MIB.

Applications

PROFINET uses the LLDP protocol for topology diagnostics. In the default setting, LLDP is enabled for all ports; in other words, LLDP frames are sent and received on all ports. With this function, you have the option of enabling or disabling sending and/or receiving per port.

Settings

The table has the following columns:

- Port
  Shows the port.
- Setting
  Specify the LLDP functionality. The following options are available:
  - Tx
    This port can only send LLDP frames.
  - Rx
    This port can only receive LLDP frames.
  - Rx & Tx
    This port can receive and send LLDP frames.
  - "-" (disabled)
    This port can neither receive nor send LLDP frames.

Configuring layer 3 functions for IPv4

NAT

Basic

Note

The page is only available

- SCALANCE W740, SCALANCE W730 and SCALANCE W722-1 RJ45
- clients or access points in client mode.

On this page, you specify the basic settings for NAT.

Settings

- **Interface**
  Select the required interface.
- **Enable NAT**
  Enable or disable NAT for the Ethernet interface. Cannot be enabled if Layer 2 Tunnel is set for MAC Mode on the client.
- **TCP Idle Timeout [s]**
  Enter the required time in seconds. If no data exchange takes place, the TCP connection is deleted from the translation table when this time has elapsed.
  The range of values is 1 to 4294967295. Default setting: 86400 seconds
- **UDP Idle Timeout [s]**
  Enter the required time in seconds. If no data exchange takes place, the UDP connection is deleted from the translation table when this time has elapsed.
  The range of values is 1 to 4294967295. Default setting: 300 seconds
- **Local Interface IP address**
  Enter the local IP address of the Ethernet interface. This IP address is the gateway address of the local device.
- **Local Interface Subnet Mask**
  Enter the subnet mask for the local Ethernet.
- **IPv6 Transparent Mode**
  When enabled, the transparent mode is activated. IPv6 frames are forwarded unchanged between Ethernet and WLAN. This assumes that "Own" is not set for MAC mode and IPv6 is turned off.

NAPT

Note

The page is only available

- SCALANCE W740, SCALANCE W730 and SCALANCE W722-1 RJ45
- clients or access points in client mode.

On this page, you define the translation list for communication from the global to the local network.

Settings

- **Interface**
  Interface to which the settings relate. Can only be selected if the device has several interfaces.
- **Traffic Type**
  Specify the protocol for which the address assignment is valid.
- **Global Port**
  Enter the global port. Incoming frames with this port as the destination port are forwarded.

  Note

  If the port is already occupied by a local service, for example Telnet, a warning is displayed. In this case, avoid using TCP port 23 (Telnet), port 22 (SSH), ports 80/443 (http/https: reachability of the client with the WBM) and UDP port 161 (SNMP) as global port.
- **Local IP address**
  Enter the IP address of the node in the local network. The frame is forwarded to this IP address.
- **Local Port**
  Enter the number of the port. This is the new destination port to which the incoming frame will be forwarded. You can enter the same port in the local area network for more than one node.

The table has the following columns:

- **Select**
  Select the lines to be deleted.
- **Interface**
  Shows the interface to which the settings relate.
- **Activate**
  When enabled, the entry for the address assignment is used.
- **Traffic Type**
  Shows the protocol for which the address assignment applies.
- **Dynamic Global IP Address**
  Shows whether or not dynamic address translation is used.
- **Global IP Address**
  Shows the global IP address to which the local IP address will be translated.

- **Global Port**
  Shows the global port.
- **Local IP address**
  Shows the IP address of the node in the local network.
- **Local Port**
  Shows the number of the local port.

Configuring security functions

Passwords

Passwords

Note

The page is only available if there is an online connection to the device.

On this page, you can change passwords. If you are logged on with the "admin" role, you can change the passwords for all user accounts. If you are logged on with the "user" role, you can only change your own password.

Note

If you log on the first time or log on after a "Reset to factory defaults and restart", you will be prompted to change the password.

Note

If you are logged in via a RADIUS server, you cannot change any local device passwords.

Note

Changing the password in "Trial" configuration mode

Even if you change the password in "Trial" configuration mode, this change is saved immediately.

Note

The factory settings for the passwords when the devices ship are as follows:

- admin: admin

When you log in for the first time or following a "Restore Factory Defaults and Restart", with the preset user "admin" you will be prompted to change the password.

Settings

- **Current User**
  Shows the user that is currently logged in.
- **Current User Password**
  Enter the password for the currently logged in user.
- **User Account**
  Select the user whose password you want to change.
- **Password Policy**
  Shows which password policy is being used when assigning new passwords.
  - High
    Password length: at least 8 characters, maximum 128 characters
    at least 1 uppercase letter
    at least 1 special character
    at least 1 number
  - Low
    Password length: at least 6 characters, maximum 128 characters
- **New Password**
  Enter the new password again to confirm it.
- **Password Confirmation**
  Enter the new password again to confirm it.

Options

On this page you specify which password policy will be used when assigning new passwords.

Description

- **Password Policy**
  Shows which password policy is currently being used
- **New Password Policy**
  Select the required setting from the drop-down list.
  - High
    Password length: at least 8 characters, maximum 128 characters
    at least 1 uppercase letter
    at least 1 special character
    at least 1 number
  - Low
    Password length: at least 6 characters, maximum 128 characters

Users

Local users

Note

The page is only available if there is an online connection to the device.

Local users

On this page, you create local users with the corresponding rights.

When you create or delete a local user this change is also made automatically in the table "External User Accounts". If you want to make change explicitly for the internal or external user table, use the CLI commands.

Note

The values displayed depend on the rights of the logged-in user.

Description

The page contains the following:

- **User Account**
  Enter the name for the user. The name must meet the following conditions:
  - It must be unique.
  - It must be between 1 and 250 characters long.
  - The following characters must not be included: § ? " ; : < =

  Note
  User name cannot be changed
  After creating a user, the user name can no longer be modified.
  If a user name needs to be changed, the user must be deleted and a new user created.

  Note
  Default user "user" set in the factory
  As of firmware version 6.0 the default user set in the factory "user" is no longer available when the product ships.
  If you update a device to the firmware V6.0 the default user set in the factory "user" is initially still available. If you reset the device to the factory settings ("Restore Factory Defaults and Restart") the default user set in the factory "user" is deleted.
  You can create new users with the role "user".

- **Password Policy**
  Shows which password policy is being used. You configure the password policy, see the Security > Passwords > Options page
- **Password**
  Enter the password. The strength of the password depends on the set password policy:

  Note
  Password Policy: Low
  - Password length: at least 6 characters, maximum 128 characters

  Password Policy: High
  The password must meet the following conditions:
  - Password length: at least 8 characters, maximum 128 characters
  - At least 1 uppercase letter
  - At least 1 special character
  - At least 1 number

- **Password Confirmation**
  Enter the password again to confirm it.
- **Role**
  Select a role.
  You can choose between default and self-defined roles, refer to the page "Security > Users > Roles.".

The table contains the following columns:

- **Select**
  Select the check box in the row to be deleted.

  Note

  The preset users as well as logged in users cannot be deleted or changed.

- **User Account**
  Shows the user name.
- **Role**
  Shows the role of the user.
- **Description**
  Displays a description of the user account. The description text can be up to 100 characters long.

Roles

Note

The page is only available if there is an online connection to the device.

Roles

On this page, you create roles that are valid locally on the device.

Note

The values displayed depend on the rights of the logged-in user.

Settings

- **Role Name**
  Enter the name for the role. The name must meet the following conditions:
  - It must be unique.
  - It must be between 1 and 64 characters long.

  Note
  Role name cannot be changed

  After creating a role, the name of the role can no longer be changed.

  If a name of a role needs to be changed, the role must be deleted and a new role created.

  Deleting roles

  Predefined roles and assigned roles cannot be deleted or modified.

The table contains the following columns:

- **Role**
  Shows the name of the role.
- **Function Right**
  Select the function rights of the role.
  - 1
    Users with this role can read device parameters but cannot change them.
  - 15
    Users with this role can both read and change device parameters.

  Note
  Function right cannot be changed

  If you have assigned a role, you can no longer change the function right of the role.

  If you want to change the function right of a role, follow the steps outlined below:
  1. Delete all assigned users.
  2. Change the function right of the role:
  3. Assign the role again.

- **Description**
  Enter a description for the role. With predefined roles a description is displayed. The description text can be up to 100 characters long.

Groups

Note

The page is only available if there is an online connection to the device.

User Groups

On this page you link a group with a role.

In this example the group "Administrators" is linked to the "admin"role: The group is defined on a RADIUS server. The role is defined locally on the device. When a RADIUS server authenticates a user and assigns the user to the "Administrators" group, this user is given rights of the "admin" role.

Settings

- Group Name
  Enter the name of the group. The name must match the group on the RADIUS server. The name must meet the following conditions:
  - It must be unique.
  - It must be between 1 and 64 characters long.

The table contains the following columns:

- Group
  Shows the name of the group.
- Role
  Select a role. Users who are authenticated with the linked group on the RADIUS server receive the rights of this role locally on the device.
  You can choose between system-defined and self-defined roles, refer to the page "Security > Users > Roles.".
- Description
  Enter a description for the link of the group.to a role.

AAA

User management

Overview of user management

Access to the device is managed by configurable user settings. Set up users with a password for authentication. Assign a role with suitable rights to the users.

The authentication of users can either be performed locally by the device or by an external RADIUS server. You configure how the authentication is handled on the "Security > AAA > General" page.

Local logon

The local logging on of users by the device runs as follows:

1. The user logs on with user name and password on the device.
2. The device checks whether an entry exists for the user.
   → If an entry exists, the user is logged in with the rights of the associated role.
   → If no corresponding entry exists, the user is denied access.

Login via an external RADIUS server

RADIUS (Remote Authentication Dial-In User Service) is a protocol for authenticating and authorizing users by servers on which user data can be stored centrally.

Depending on the RADIUS authorization mode you have selected on the "Security > AAA > RADIUS Client" page, the device evaluates different information of the RADIUS server.

RADIUS authorization mode "Standard"

If you have set the authorization mode "conventional", the authentication of users via a RADIUS server runs as follows:

1. The user logs on with user name and password on the device.
2. The device sends an authentication request with the login data to the RADIUS server.
3. The RADIUS server runs a check and signals the result back to the device.
   - The RADIUS server reports a successful authentication and returns the value "Administrative User" to the device for the attribute "Service Type".
     → The user is logged in with administrator rights.
   - The RADIUS server reports a successful authentication and returns a different or even no value to the device for the attribute "Service Type".
     → The user is logged in with read rights.
   - The RADIUS server reports a failed authentication to the device:
     → The user is denied access.

RADIUS authorization mode "SiemensVSA"

Requirement

For the RADIUS authorization mode "Siemens VSA" the following needs to be set on the RADIUS server:

- Manufacturer code: 4196
- Attribute number: 1
- Attribute format: Character string (group name)

Procedure

If you have set the authorization mode "SiemensVSA", the authentication of users via a RADIUS server runs as follows:

1. The user logs on with user name and password on the device.
2. The device sends an authentication request with the login data to the RADIUS server.
3. The RADIUS server runs a check and signals the result back to the device.
   Case A: The RADIUS server reports a successful authentication and returns the group assigned to the user to the device.
   - The group is known on the device and the user is not entered in the table "External User Accounts"
     → The user is logged in with the rights of the assigned group.
   - The group is known on the device and the user is entered in the table "External User Accounts"
     → The user is assigned the role with the higher rights and logged in with these rights.
   - The group is not known on the device and the user is entered in the table "External User Accounts"
     → The user is logged in with the rights of the role linked to the user account.
   - The group is not known on the device and the user is not entered in the table "External User Accounts"
     → The user is logged in with the rights of the role "Default".

   Case B: The RADIUS server reports a successful authentication but does not return a group to the device.
   - The user is entered in the table "External User Accounts":
     → The user is logged in with the rights of the linked role "".
   - The user is not entered in the table "External User Accounts":
     → The user is logged in with the rights of the role "Default".

   Case C: The RADIUS server reports a failed authentication to the device:
   - The user is denied access.

General

The term "AAA" used in the menu stands for "Authentication, Authorization, Accounting" and is used to identify and allow network nodes, to provide them with the appropriate services and to determine the amount of use.

On this page, you configure the login.

Settings

Note

To be able to use the login authentication "RADIUS", "Local and RADIUS" or "RADIUS and fallback Local" a RADIUS server must be stored and configured for user authentication.

- **Login Authentication**
  Specify how the login is made:
  - Local
    The authentication must be made locally on the device.
  - RADIUS
    The authentication must be handled via a RADIUS server.
  - Local and RADIUS
    The authentication is possible both with the users that exist on the device (user name and password) and via a RADIUS server.
    The user is first searched for in the local database. If the user does not exist there, a RADIUS query is sent.
  - RADIUS and fallback Local
    The authentication must be handled via a RADIUS server.
    A local authentication is performed only when the RADIUS server cannot be reached in the network.

RADIUS client

The term "AAA" used in the menu stands for "Authentication, Authorization, Accounting" and is used to identify and allow network nodes, to provide them with the appropriate services and to determine the amount of use.

Authentication over an external server

The concept of RADIUS is based on an external authentication server.

Each row of the table contains access data for one server. In the search order, the primary server is queried first. If the primary server cannot be reached, secondary servers are queried in the order in which they are entered.

If no server responds, there is no authentication.

Settings

The page contains the following boxes:

- **RADIUS Authorization Mode**
  For the login authentication, the RADIUS authorization mode specifies how the rights are assigned to the user with a successful authentication.
  - conventional
    In this mode the user is logged in with administrator rights if the server returns the value "Administrative User" to the device for the attribute "Service Type". In all other cases the user is logged in with read rights.
  - Vendor specific
    In this mode the assignment of rights depends on whether and which group the server returns for the user and whether or not there is an entry for the user in the table "External User Accounts".

The table has the following columns:

- **Auth. Server Type**
  Select which authentication method the server will be used for.
  - Login
    The server is used only for the login authentication.
  - 802.1X
    The server is used only for the 802.1X authentication.
  - Login & 802.1X
    The server is used for both authentication procedures.
- **RADIUS Server Address**
  Enter the IPv4 address or the FQDN (Fully Qualified Domain Name) of the RADIUS server.
- **Server Port**
  Here, enter the input port on the RADIUS server. As default, input port 1812 is set.
- **Shared Secret**
  Enter your access ID.
- **Confirm Shared Secret**
  Enter your access ID again as confirmation.
- **Max. Retrans**
  Enter the maximum number of query attempts before another configured RADIUS server is queried or the login counts as having failed. As default, 3 is set.
- **Primary Server**
  Specify whether or not this server is the primary server. You can select one of the options "yes" or "no".
- **Test**
  With this button, you can test whether or not the specified RADIUS server is available. The test is performed once and not repeated cyclically.
- **Test Result**
  Shows whether or not the RADIUS server is available:
  - Failed, no test packet sent
    The IP address is not reachable.
    The IP address is reachable, the RADIUS server is, however, not running.
  - Reachable, key not accepted
    The IP address is reachable, the RADIUS server does not, however accept the shared secret.
  - Reachable, key accepted
    The IP address is reachable, the RADIUS server accepts the specified shared secret.

  The test result is not automatically updated. To delete the test result click the "Refresh" button.

MAC ACL

Rule configuration

On this page, you specify the access rules for the MAC-based Access Control List. Using the MAC-based ACL, you can specify whether frames of certain MAC addresses are forwarded or discarded.

Settings

- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Source MAC**
  Enter the unicast MAC address of the source.
  Destination MAC Address
  Enter the unicast MAC address of the destination.
- **Action**
  Select the action. The following is possible:
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.
- **Ingress Interfaces**
  Shows a list of all interfaces to which this rule applies.
- **Egress Interfaces**
  Shows a list of all interfaces to which this rule applies.

Note

Entering the MAC addresses

If you enter the address "00:00:00:00:00:00" for the source and/or destination MAC address, the rule created in this way applies to all source or destination MAC addresses.

Ingress Rules

On this page, you specify the ACL rule according to which incoming frames are handled by the port.

Settings

- **Interface**
  Select the required interface.
- **Add Rule**
  Select the ACL rule to be assigned to the interface. You specify the ACL rule on the "Rules Configuration" page.
- **Add**
  To permanently assign the ACL rule to the interface, click the "Add" button. The configuration is shown in the table.
- **Remove Rule**
  Select the ACL rule to be deleted.
- **Remove**
  To remove the ACL rule from the interface, click the "Remove" button.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Source MAC**
  Shows the unicast MAC address of the source.
- **Destination MAC Address**
  Shows the unicast MAC address of the destination.
- **Action**
  Shows the action:
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.

Egress Rules

On this page, you specify the ACL rule according to which outgoing frames are handled by the port.

Settings

- **Interface**
  Select the required interface.
- **Add Rule**
  Select the ACL rule to be assigned to the interface. You specify the ACL rule on the "Rules Configuration" page.

- **Add**
  To permanently assign the ACL rule to the interface, click the "Add" button. The configuration is shown in the table.
- **Remove Rule**
  Select the ACL rule to be deleted.
- **Remove**
  To remove the ACL rule from the interface, click the "Remove" button.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Source MAC**
  Shows the unicast MAC address of the source.
- **Destination MAC Address**
  Shows the unicast MAC address of the destination.
- **Action**
  Shows the action
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.

IP ACL

Rule configuration

On this page, you specify the rules for the IP-based ACL.

Settings

The table has the following columns:

- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Source IP**
  Enter the IPv4 address of the source.
- **Source Subnet Mask**
  Enter the subnet mask in which the source is located.
- **Dest. IP**
  Enter the IP address of the destination.

- **Dest. Subnet Mask**
  Enter the subnet mask in which the destination is located.
- **Action**
  Select whether the frame is forwarded or rejected when it corresponds to the ACL rule.
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.
- **Ingress Interfaces**
  Shows a list of all interfaces to which this rule applies.
- **Egress Interfaces**
  Shows a list of all interfaces to which this rule applies.

Protocol configuration

On this page, you specify the rules for protocols.

Settings

The table has the following columns:

- **Rule Number**
  Shows the number of the protocol rule. When you create a rule, a new row with a unique number is created.
- **Protocol**
  Select the protocol for which this rule is valid.
- **Protocol Number**
  Enter a protocol number to define further protocols.
  This box can only be edited if you have set "Other Protocol" for the protocol.
- **Source Port Min.**
  Enter the lowest possible port number of the source port.
  This box can only be edited if you have set "TCP" or "UDP" for the protocol.
- **Source Port Max.**
  Enter the highest possible port number of the source port.
  This box can only be edited if you have set "TCP" or "UDP" for the protocol.
- **Dest. Port Min.**
  Enter the lowest possible port number of the destination port.
  This box can only be edited if you have set "TCP" or "UDP" for the protocol.
- **Dest. Port Max.**
  Enter the highest possible port number of the destination port.
  This box can only be edited if you have set "TCP" or "UDP" for the protocol.
- **Message Type**
  Enter a message type to decide the format of the message.
  This box can only be edited if you have set "ICMP" for the protocol.

- **Message Code**
  Enter a message code to specify the function of the message.
  This box can only be edited if you have set "ICMP" for the protocol.
- **DSCP**
  Enter a value for classifying the priority.
  This box cannot be edited if you have set "ICMP" for the protocol.

Ingress Rules

On this page, you specify the ACL rule according to which incoming frames are handled by the port or VLAN.

Settings

- **Ports**
  Select the required port: a port or a VLAN interface.
  To select a VLAN interface, an IP interface must be configured.

  Note

  If you use a VLAN interface, the ACL rule applies to all ports that belong to the VLAN.

- **Add Rule**
  Select the ACL rule to be assigned to the port. You specify the ACL rule on the "Rules Configuration" page.
- **Add**
  To permanently assign the ACL rule to the port, click the "Add" button. The configuration is shown in the table.
- **Remove Rule**
  Select the ACL rule to be deleted.
- **Remove**
  To remove the ACL rule from the port, click the "Remove" button.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Protocol**
  Shows the protocol for which this rule is valid.
- **Protocol Number**
  Shows the protocol number.
- **Source IP**
  Shows the IP address of the source.
- **Source Subnet Mask**
  Shows the subnet mask in which the source is located.

- **Dest. IP**
  Shows the IP address of the destination.
- **Dest. Subnet Mask**
  Shows the subnet mask of the destination.
- **Action**
  Select whether the frame is forwarded or rejected when it corresponds to the ACL rule.
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.
- **Source Port Min.**
  Shows the lowest possible port number of the source port.
- **Source Port Max.**
  Shows the highest possible port number of the source port.
- **Dest. Port Min.**
  Shows the lowest possible port number of the destination port.
- **Dest. Port Max.**
  Shows the highest possible port number of the destination port.
- **Message Type**
  Shows a message type to decide the format of the message.
- **Message Code**
  Shows a message code to specify the function of the message.
- **DSCP**
  Shows the value for classifying the priority.

Egress Rules

On this page, you specify the ACL rule according to which outgoing frames are handled by the port.

Settings

- **Ports**
  Select the required port: a port or a VLAN interface.
  To select a VLAN interface, an IP interface must be configured.

  Note

  If you use a VLAN interface, the ACL rule applies to all ports that belong to the VLAN.

- **Add Rule**
  Select the ACL rule to be assigned to the port. You specify the ACL rule on the "Rules Configuration" page.
- **Add**
  To permanently assign the ACL rule to the port, click the "Add" button. The configuration is shown in the table.

- **Remove Rule**
  Select the ACL rule to be deleted.
- **Remove**
  To remove the ACL rule from the port, click the "Remove" button.

The table has the following columns:

- **Rule Order**
  Shows the order of the ACL rules.
- **Rule Number**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **Protocol**
  Shows the protocol for which this rule is valid.
- **Protocol Number**
  Shows the protocol number.
- **Source IP**
  Shows the IP address of the source.
- **Source Subnet Mask**
  Shows the subnet mask in which the source is located.
- **Dest. IP**
  Shows the IP address of the destination.
- **Dest. Subnet Mask**
  Shows the subnet mask of the source.
- **Action**
  Select whether the frame is forwarded or rejected when it corresponds to the ACL rule.
  - Forward
    If the frame complies with the ACL rule, the frame is forwarded.
  - Discard
    If the frame complies with the ACL rule, the frame is not forwarded.
- **Source Port Min.**
  Shows the lowest possible port number of the source port.
- **Source Port Max.**
  Shows the highest possible port number of the source port.
- **Dest. Port Min.**
  Shows the lowest possible port number of the destination port.
- **Dest. Port Max.**
  Shows the highest possible port number of the destination port.
- **Message Type**
  Shows a message type to decide the format of the message.
- **Message Code**
  Shows a message code to specify the function of the message.
- **DSCP**
  Shows the value for classifying the priority.

WLAN

Basic (Access Point)

Safety levels

To make the network secure, authentication and encryption are used. On this page, you specify the security settings.

Note

Transmission standard IEEE 802.11 n

The transmission standard IEEE 802.11 n with the setting "802.11n" or "802.11 n only" supports only WPA2/ WPA2-PSK with AES in the security settings.

iPCF or iPCF-MC

If iPCF or iPCF-MC is enabled, only "Open System" with the encryption methods WEP or AES is supported in the security settings.

Settings

The table has the following columns:

- **Interface**
  Shows the available interfaces.
- **Authentication Type**
  Select the type of authentication. The selection depends on the operating mode and the transmission standard.
  - **Open System**
    There is no authentication. Encryption with a fixed (unchanging) WEP key can be selected as an option. To use the key, enable "Encryption". You define the WEP key on the "Keys" page.
    If iPCF mode is enabled, the encryption method AES with 128 bit key length is also supported.
  - **Shared Key**
    With this authentication, a fixed key is stored on the client and access point. This WEP key is then used for authentication and encryption. You define the WEP key on the "Keys" page.

    Note

    If you use "Open System" with "Encryption" or "Shared Key", Key 1 must always be set on the "Keys" page.

  - **WPA (RADIUS)**
    Wi-Fi Protected Access is a method specified by the Wi-Fi Alliance to close security gaps in WEP. Authentication using a server is stipulated (802.1x). The dynamic exchange of keys at each data frame introduces further security.
  - **WPA-PSK**
    WPA Pre Shared Key (WPA-PSK) is a weakened form of WPA. In this method, authentication is not established by a server but is based on a password. This password is configured manually on the client and server.
  - **WPA2 (RADIUS)**
    WPA2 (Wi-Fi Protected Access 2) is a further development of WPA and implements the functions of the IEEE 802.11i security standard. WPA authentication works, however, without the RADIUS server.
  - **WPA2-PSK**
    WPA2-PSK is based on the 802.11i standard. WPA authentication works, however, without a RADIUS server. Instead of this, a WPA(2) key (WPA(2) pass phrase) is stored on each client and access point. The WPA(2) pass phrase is used for authentication and further encryption.
  - **WPA/WPA2-Auto-PSK**
    Setting with which an access point can process both the "WPA-PSK" as well as the "WPA2-PSK" type of authentication. This is necessary when the access point communicates with different clients, some using "WPA-PSK" and others "WPA2-PSK". The same encryption method is set on the clients.

  - **WPA/WPA2-Auto**
    Setting with which an access point can process both the "WPA" as well as the "WPA2" type of authentication. This is necessary when the access point communicates with different clients, some using "WPA" and others "WPA2". The same encryption method is set on the clients

- **Encryption**
  Encryption protects the transferred data from eavesdropping and corruption. You can only disable encryption if you have selected "Open System" for authentication. All other security methods include both authentication and encryption.

- **Cipher**
  Select the encryption method. The selection depends on the transmission standard.

  - **AUTO**
    AES or TKIP is used depending on the capability of the other station.

  - **WEP**
    WEP (Wired Equivalent Privacy)
    A symmetrical stream encryption method with only 40- or 104-bit long keys based on the RC4 algorithm (Ron's Code 4).

  - **TKIP (Temporal Key Integrity Protocol)**
    A symmetrical encryption method with the RC4 algorithm (Ron's Code 4). In contrast to the weak WEP encryption, TKIP uses changing keys derived from a main key. TKIP can also recognize corrupted data frames.

  - **AES (Advanced Encryption Standard)**

    Strong symmetrical block encryption method based on the Rijndael algorithm that further improves the functions of TKIP.

    Note

    To provide better protection of your data against attacks, use WPA2/ WPA2-PSK with AES.

- **WPA(2) Pass Phrase**
  Enter a WPA(2) pass phrase here. This WPA(2) key must be known both at the client end and on the access point and is entered by the user at both ends.
  ASCII code 0x20 to 0x7e is used for the WPA(2) key.

- **WPA(2) Pass Phrase Confirmation**
  Confirm the entered WPA(2) key.

- **Default Key**
  Specify the WEP key used to encrypt the data. You define the WEP key on the "Keys" page.

Options

Which settings are possible depends on the "Authentication Type" you have selected.

Authentication type	Encryption	Cipher	Encryption key source
Open System	disabled	--	--
Open System	Enabled	WEP	Default Key
Open System	Enabled	AES[1)]	Default Key (128-bit)

Authentication type	Encryption	Cipher	Encryption key source
Shared Key	Enabled	WEP	Default Key
WPA (RADIUS)	Enabled	Auto/TKIP/AES	RADIUS Server
WPA-PSK	Enabled	Auto/TKIP/AES	WPA(2) Pass Phrase
WPA2 (RADIUS)	Enabled	Auto/TKIP/AES	RADIUS Server
WPA2-PSK	Enabled	Auto/TKIP/AES	WPA(2) Pass Phrase
WPA/WPA2-AutoPSK	Enabled	Auto/TKIP/AES	WPA(2) Pass Phrase
WPA/WPA2-Auto (RADIUS)	Enabled	Auto/TKIP/AES	RADIUS Server

[1] available only with iPCF or iPCF-MC

Basic (Client)

Safety levels

To make the network secure, authentication and encryption are used. On this page, you specify the security settings.

Note

Transmission standard IEEE 802.11 n

The transmission standard IEEE 802.11 n with the setting "802.11n" or "802.11 n only" supports only WPA2/ WPA2-PSK with AES in the security settings.

iPCF or iPCF-MC mode activated

If iPCF or iPCF-MC mode is enabled, only "Open System" with the encryption method AES is supported in the security settings with Security Context 1.

Settings

The table has the following columns:

- Security context
  Shows the number of the entry. If you create a new entry, a new row with a unique number is created.
  You can create up to 8 security contexts. The security context 1 cannot be deleted.
- Authentication Type
  Select the type of authentication. The selection depends on the operating mode and the transmission standard.
  - Open System
    There is no authentication. Encryption with a fixed (unchanging) WEP key can be selected as an option. To use the key, enable "Encryption". You define the WEP key on the "Keys" page.
    If iPCF or iPCF-MC mode is enabled, only the encryption method AES with 128 bit key length is supported.
  - Shared Key
    In Shared Key authentication, a fixed key is stored on the client and access point. This WEP key is then used for authentication and encryption. You define the WEP key on the "Keys" page.
  - WPA (RADIUS)
    Wi-Fi Protected Access is a method specified by the Wi-Fi Alliance to close security gaps in WEP. Authentication using a server is stipulated (802.1x). The dynamic exchange of keys at each data frame introduces further security.

    Note

    Make the relevant RADIUS settings initially on the page "Security > WLAN > Basic".

  - WPA-PSK
    WPA Pre Shared Key (WPA-PSK) is a weakened form of WPA. In this method, authentication is not established by a server but is based on a password. This password is configured manually on the client and server.
  - WPA2 (RADIUS)
    WPA2 (Wi-Fi Protected Access 2) is a further development of WPA and implements the functions of the IEEE 802.11i security standard. WPA authentication works, however, without the RADIUS server.

    Note

    Make the relevant RADIUS settings initially on the page "Security > WLAN > Client Radius Supplicant".

  - WPA2-PSK
    WPA2-PSK is based on the 802.11i standard. WPA authentication works, however, without a RADIUS server. Instead of this, a WPA(2) key (WPA(2) pass phrase) is stored on each client and access point. The WPA(2) pass phrase is used for authentication and further encryption.

  - **WPA/WPA2-Auto-PSK**
    Setting with which an access point can process both the "WPA-PSK" as well as the "WPA2-PSK" type of authentication. This is necessary when the access point communicates with different clients, some using "WPA-PSK" and others "WPA2-PSK". The same encryption method is set on the clients.
  - **WPA/WPA2-Auto**
    Setting with which an access point can process both the "WPA" as well as the "WPA2" type of authentication. This is necessary when the access point communicates with different clients, some using "WPA" and others "WPA2". The same encryption method is set on the clients

- **Encryption**
  Encryption protects the transferred data from eavesdropping and corruption. You can only disable encryption if you have selected "Open System" for authentication. All other security methods include both authentication and encryption.
- **Cipher**
  Select the encryption method. The selection depends on the transmission standard.
  - **AUTO**
    AES or TKIP is used depending on the capability of the other station.
  - **WEP**
    WEP (Wired Equivalent Privacy)
    A symmetrical stream encryption method with only 40- or 104-bit long keys based on the RC4 algorithm (Ron's Code 4).
  - **TKIP (Temporal Key Integrity Protocol)**
    A symmetrical encryption method with the RC4 algorithm (Ron's Code 4). In contrast to the weak WEP encryption, TKIP uses changing keys derived from a main key. TKIP can also recognize corrupted data frames.
  - **AES (Advanced Encryption Standard)**

    Strong symmetrical block encryption method based on the Rijndael algorithm that further improves the functions of TKIP.

    Note

    To provide better protection of your data against attacks, use WPA2/ WPA2-PSK with AES.

- **WPA(2) Pass Phrase**
  Enter a WPA(2) pass phrase here. This WPA(2) key must be known on both the client and the access point and is entered by the user at both ends.
  For a key with 8 to 63 characters, you can only use the following readable ASCII characters: 0x20 - 0x7e.
  For a key with precisely 64 characters, you can use the following ASCII characters: 0 - 9, a - f and A - F.
- **WPA(2) Pass Phrase Confirmation**
  Confirm the entered WPA(2) key.
- **Default Key**
  Specify the WEP key used to encrypt the data. You define the WEP key on the "Keys" page.

AP communication

Communications options

On this page, you specify the type of communication allowed by the access point.

Note

The page is available only in access point mode.

Settings

Table 1 has the following columns:

- **Port**
  Shows that the settings are valid for all ports of Table 2.
- **within own VAP / with other VAPs / with Ethernet**
  Select the setting you require. If "No Change" is selected, the entry in table 2 remains unchanged.
- **Transfer to table**
  If you click the button, the setting is adopted for all ports of table 2.

Table 2 has the following columns:

- **Radio**
  Shows the available WLAN interfaces.
- **Port**
  Shows the VAP interface.
- **within own VAP**
  - Enabled
    Clients logged on to the same VAP interface of an access point can communicate with each other.
  - Disabled
    Option is disabled.

- **with other VAPs**
  Only available if the device has at least 2 VAP interfaces.
  - Enabled
    Clients logged on to different VAP interfaces of an access point can communicate with each other.
  - Disabled
    Option is disabled.

  Note

  For an access point, "with other VAPs" must be enabled on all WLAN interfaces or on all VAPs to allow communication between clients logged on at different VAP interfaces of the access point.

- **with Ethernet**
  - Enabled
    Clients can communicate via the Ethernet interface of the access point.
  - Disabled
    Option is disabled.

Client RADIUS Supplicant

Client Supplicant

On this page, you configure the settings for the RADIUS authorization of the client.

Note

This page is only available for clients or access points in client mode.

Settings

The table has the following columns:

- **Security context**
  Shows the security context.
- **Dot1x User Name**
  Enter the user name with which the client will log on with the RADIUS server.
- **Dot1x User Password**
  Enter the password for the user name specified above. The client logs on with the RADIUS server using this combination.
  For password assignment, ASCII characters 0x20 to 0x7e are used.
- **Confirm Dot1x user password**
  Confirm the password.

- **Dot1x Check Server Certificate**
  Specify whether or not the RADIUS server identifies itself to the client using a certificate.
- **Dot1x EAP Types**
  Specify the authentication method. The following methods are available:
  - Auto
    EAP-TLS, EAP-TTLS or PEAP is used depending on the capability of the other station.
  - EAP-TLS
    Client logs on using a certificate.
  - EAP-TTLS
    The client logs on with the RADIUS server using the user name and password
  - PEAP
    The client logs on with the RADIUS server using the user name and password.

AP RADIUS Authenticator

Configuration of the RADIUS server

On this page, you define the RADIUS servers and the RADIUS authentication of the access point. You can enter data for two RADIUS servers.

Note

The page is available only in access point mode.

Settings

- **Reauthentication Mode**
  Specify who sets the time after which the clients are forced to reauthenticate.
  - '-' Disabled
    Reauthentication mode is disabled.
  - Server
    Enables time management on the server.
  - Local
    Enables local time management. In "Reauthentication Interval", specify the time of validity.
- **Reauthentication Interval [s]**
  If time management is local, enter the period of validity of the authentication in seconds. The minimum time is 1 minute (enter 60), the maximum time is 12 hours (enter 43200). The default is one hour (3,600 seconds).

The table has the following columns:

- **Server IP Address**
  Here, enter the IP address of the RADIUS server.
- **Server Port**
  Here, enter the input port on the RADIUS server.
- **Shared Secret**
  Enter the password of the RADIUS server.
  For the password, ASCII code 0x20 to 0x7e is used.
- **Confirm Shared Secret**
  Confirm the password.
- **Max. Retransmissions**
  Enter the maximum number of connection attempts.
- **Primary Server**
  Specify whether or not this server is the primary server.
  - yes: Primary server
  - no: Backup server.
- **Status**
  With this setting, you can enable or disable the RADIUS server

Key

Specifying the WEP key

To allow you to enable the encryption for the "Open System" and "Shared Key" authentication methods, you must first enter at least one key in the key table.

Settings

The table has the following columns:

- **Key 1 - 4**
  Enter the WEP or AES key. For the WEP key, characters of the ASCII code from 0x20 to 0x7E or hexadecimal characters from 0x00 to 0xFF are permitted.
  If iPCF or iPCF-MC mode is enabled, only the encryption method AES with 128-bit key length is supported.
  You can choose between the following key lengths:
  - 5 or 13 ASCII or 10 or 26 hexadecimal characters (40/104 bits)
  - 16 ASCII or 32 hexadecimal characters (128 bits)

  Note

  The hexadecimal characters are entered without being preceded by "0x". One hexadecimal character codes four bits. The entries "ABCDE" (ASCII characters) and "4142434445" (hexadecimal characters) are therefore the same because the ASCII character "A" has hexadecimal code "0x41".

- **Key 1 - 4 Confirmation**
  Confirm the WEP key.

Management ACL

On this page, you can increase the security of your device. To specify which station with which IP address is allowed to access your device, configure the IP address or an entire address range.

You can select the protocols and the ports with which the station is allowed to access the device. You define the VLAN in which the station may be located. This ensures that only certain stations within a VLAN have access to the device.

Note

Note that a bad configuration may mean that you can no longer access the device.

Settings

- **Management ACL**
  Enable or disable the function.
- **IP Address**
  Enter the IP address or the network address to which the rule will apply. If you use the IP address 0.0.0.0, the settings apply to all IP addresses.
- **Subnet Mask**
  Enter the subnet mask. The subnet mask 255.255.255.255 is for a specific IP address. If you want to allow a subnet, for example a C subnet, enter 255.255.255.0. The subnet mask 0.0.0.0 applies to all subnets.

The table has the following columns:

- **Select**
  Select the lines to be deleted.
- **Rule Order**
  Shows the number of the ACL rule. When you create a rule, a new row with a unique number is created.
- **IP Address**
  Shows the IP address.
- **Subnet Mask**
  Shows the subnet mask.
- **VLANs Allowed**
  Enter the number of the VLAN in which the device is located. The station can only access the device if it is located in this configured VLAN. If this input box remains empty, there is no restriction.
- **SNMP**
  Specify whether the station or the IP address accesses the device using the SNMP protocol.
- **TELNET**
  Specify whether the station or the IP address accesses the device using the TELNET protocol.
- **HTTP**
  Specify whether the station or the IP address accesses the device using the HTTP protocol.
- **HTTPS**
  Specify whether the station or the IP address accesses the device using the HTTPS protocol.
- **SSH**
  Specify whether the station or the IP address accesses the device using the SSH protocol.
- **Px**
  Specify whether the station or the IP address accesses the device via this port.
- **WLAN1** (In client mode only)
  Specify whether the station or the IP address accesses the device via the WLAN interface.
- **VAPX.Y** (only in access point mode)
  Specify whether the station or the IP address accesses the device via the VAP interface.
- **WDSX.Y** (only in access point mode)
  Specify whether the station or the IP address accesses the device via the WDS interface.

Inter AP blocking

Basic

Note

- This tab is only available.
  - SCALANCE W780 / W770
  - in access point mode
- This WBM page can only be configured with the following KEY-PLUGs:
  - W780 iFeatures (MLFB 6GK5 907-8PA00)
  - W700 Security (MLFB 6GK5907-0PA00)

When should Inter AP blocking be used?

The clients connected to an access point can normally communicate with all devices of the layer 2 network.

With inter AP blocking, the communication of the clients connected to the access point can be restricted. Only the devices whose IP addresses are configured in "Allowed Addresses" on the access point are accessible to the clients. Communication with other nodes in the network is therefore prevented.

Settings

- **Refresh Interval [s]**
  Enter the refresh interval for the ARP table.

The table has the following columns:

- **Radio**
  Specifies the WLAN interface to which the settings relate.
- **Port**
  Specifies the VAP interface to which the settings relate.
- **SSID**
  Specifies the SSID to which the settings relate.
- **Enabled**
  When selected, the access restriction is used. You configure the devices that are accessible for the clients in "Security > Inter AP Blocking > Allow IP Addresses".
- **Block Gratuitous ARP Requests**
  When enabled, gratuitous ARP packets are not forwarded.
- **Block Non-IP Frames**
  When enabled, there is no exchange of non-IP packets, for example layer 2 packets between the client and the devices configured on the access point as permitted communications partners.

Allowed Addresses

On this page you specify which devices are accessible to the clients.

Note

- This tab is only available.
  - SCALANCE W780 / W770
  - in access point mode
- This WBM page can only be configured with the following KEY-PLUGs:
  - W780 iFeatures (MLFB 6GK5 907-8PA00)
  - W700 Security (MLFB 6GK5907-0PA00)

Settings

- **Port**
  Select the required port.
- **IP address**
  Enter the IP address of the devices accessible to the client.

The table has the following columns:

- **Select**
  Select the lines to be deleted.
- **Radio**
  Specifies the WLAN interface to which the settings relate
- **Port**
  Specifies the VAP interface to which the settings relate
- **IP address**
  IP Address of the devices accessible to the client. If necessary, you can change the IP address.
- **Resolver IP Address**
  The IP address with which the allowed IP address is resolved. The entry is necessary if the management IP address is located in a different subnet. If the IP address "0.0.0.0" is configured for "IP address resolution", the management IP address is used for the resolution.

Configuring iFeatures

iPCF

Note

iPCF for clients or access points in client mode

- SCALANCE W740, SCALANCE W734-1 RJ-45 and SCALANCE W722-1 RJ-45
- The setting can only be configured with the following KEY-PLUG:
  - Client: W740 iFeatures (MLFB 6GK5 907-4PA00)

iPCF in access point mode

- SCALANCE W780 and SCALANCE W774-1 RJ-45
- The setting can only be configured with the following KEY-PLUG:
  - Access point: W780 iFeatures (MLFB 6GK5 907-8PA00)

When should iPCF be used?

The use of iPCF is advisable particularly if you have a large number of nodes and want to implement highly deterministic operation. This is necessary, for example with PROFINET or other cyclic protocols.

Note

iPCF, iPCF-MC and iPRP are not compatible with each other and cannot be used at the same time on a device.

Settings

In both modes, the table has the following columns:

- **Radio**
  Specifies the WLAN interface to which the settings relate.
- **Enable iPCF**
  Enable or disable iPCF mode. For PROFINET communication, we recommend that you enable the iPCF mode. By enabling iPCF, the data rates provided by the access point are adapted. We strongly recommend that you retain the default setting for the data rates (802.11 a/b/g = 12 Mbps and 802.11n = MCS 2).

In access point mode, the table has the following additional columns:

- **Protocol Support**
  Specify which protocol is handled with priority by the WLAN interface.
  - PROFINET
    if you set PROFINET, there must be no PROFINET controller downstream from the client.
  - EtherNet/IP
    If you set Ethernet/IP, there must be no scanner downstream from the client.
  - Disabled
    The function is disabled.
- **iPCF Cycle Time [ms]**
  Select the required cycle time from the drop-down list.
  The following points need to be taken into account when setting the cycle time. Otherwise it may not be possible to establish stable communication.
  - There is only one access point in the system; in other words, the clients move only in one wireless cell. In this case, update times >= 16 ms are supported.
  - There are several access points in the system that communicate over different channels. The clients roam between the access points. In this case, select update times >= 32 ms.

  In addition to the guide values shown above, remember that the shortest cycle time to be set is calculated according to the formula "2 ms * max. number of nodes".
- **Scanning Mode**
  The selected setting affects the scanning of the logged-on clients.
  The following settings are available:
  - All Channels
    The client scans all permitted channels and selects the access point with the best signal strength and connects to it.
  - Next Channel
    The client scans the next channel from its permitted channel list. If an access point is there, it connects to it. If it does not find an access point on this channel, it scans the next channel.
- **Signal Quality Threshold**
  Can only be configured if "Next Channel" is set for "Scanning Mode". The client is given a minimum signal strength with which the access point must be seen during the scan so that a connection can be established to it.
  The following threshold values exist for the signal strength:

Level	Signal quality in RSSI	Signal quality in %
1	20	40
2	25	50
3	30	60
4	35	70
5	40	80

iPCF-MC

Requirements to be able to use iPCF-MC:

- The access point has at least two WLAN interfaces (dual AP).
- Access point mode: Only dual APs with KEY-PLUG W780 iFeatures (MLFB 6GK5 907-8PA00)
- Client mode: Client with KEY-PLUG W740 iFeatures (MLFB 6GK5 907-4PA00)
- The management interface and data interface must be operated in the same frequency band and mode and must match in terms of their wireless coverage. iPCF-MC will not work if both wireless interfaces are equipped with directional antennas that cover different areas.
- The management interfaces of all access points to which a client can change must use the same channel. A client scans only this one channel to find accessible access points.
- Transmission based on IEEE801.11h (DFS) cannot be used for the management interface. 801.11h (DFS) is possible for the data interface.
- The client cannot be operated with "Use Allowed Channels only".
- "Force Roaming on link down" is automatically mirrored on the second interface.
- The following applies to clients: All configured and active SSIDs must be assigned to security context 1. An SSID is active when the corresponding check box "Enabled" is selected on the page "Interfaces > WLAN > Client".
- In Japan, iPCF-MC cannot be enabled if the data or management interface uses a frequency of the 4920 MHz - 5080 MH frequency band.

When should iPCF-MC be used?

iPCF was developed to achieve short handover times when roaming between cells. The iPCF-MC technique allows short handover times even for freely mobile clients and when a lot of cells are involved or a large number of channels is being used.

Note

- iPCF, iPCF-MC and iPRP are not compatible with each other and cannot be used at the same time on a device.
- With 11n devices, remember that the assignment of the WLAN interfaces is fixed for iPCF-MC.
  - WLAN1: Data interface
  - WLAN2: Management interface

Settings

The following setting exists in both modes

- **Enable iPCF-MC**
  Enable or disable the iPCF-MC mode of the device.
  For PROFINET communication, we recommend enabling the iPCF-MC mode. By enabling iPCF-MC, the data rates provided by the access point are adapted.
  We strongly recommend that you retain the default setting for the data rates (802.11 a/b/g = 12 Mbps and 802.11n = MCS 2).

The following setting exists only in access point mode

- **iPCF Cycle Time**
  Select the update time configured for the network to which the access point is connected. The lowest value for the update time is 32 ms.

The following setting exists only in client mode

- **Management Scan Period**
  This parameter specifies the time between two client background scans (specified in iPCF cycles). If, for example, you select two, the client runs a background scan only in every second iPCF cycle.
  A lower value for the background scan interval provides the basis for fast roaming, however this means that no high data throughput can be achieved. A higher value should be selected for a high data throughput.

Protocol Support (only in access point mode)

Specify which protocol is handled with priority.

- PROFINET
  if you set PROFINET, there must be no PROFINET controller downstream from the client.
- EtherNet/IP
  If you set Ethernet/IP, there must be no scanner downstream from the client.

iPRP

Note

- SCALANCE W770/W740, SCALANCE W770/W730 and SCALANCE W722-1 RJ45

The following KEY-PLUGs are inserted:

- Access point: W780 iFeatures (MLFB 6GK5 907-8PA00)
- Client: W740 iFeatures (MLFB 6GK5 907-4PA00)

When should iPRP be used?

Note

Use of iPRP with other iFeatures

IPRP and other iFeatures (e.g. iPCF. iPCF-MC,) are not compatible with each other and cannot be used at the same time on a device.

iPRP with oversize frames (jumbo frames)

To be able to use oversize frames, oversize frames (jumbo frames) must be configured on all devices in the network.

With the "industrial Parallel Redundancy Protocol" (iPRP) the PRP technology can be used in a wireless network. With IPRP the PRP frames are transferred parallel via two wireless links. The parallel transfer allows disruptions of the transfer on one wireless link to be compensated on the other.

Requirement

- The Base Bridge mode "802.1Q VLAN Bridge" is set.
- The VLANs have been created.
- Access point mode: The VAP interface is enabled.
- Client mode: For "MAC Mode", Layer 2 Tunnel is set.

Settings

The page contains the following:

- **PRP A**
  Select the VLAN assignment for PRP from the drop-down list.
- **PRP B**
  Select the VLAN assignment for PRP B from the drop-down list.

This table contains the following columns:

- **Interface**
  Shows the available interfaces.

  Note

  Access point

  On the access point the iPRP function is only supported on a VAP interface.

- **Enable iPRP**
  Enable or disable iPRP for the required interface.
- **PRP Network**
  Specify the PRP network in which the interface is a member.

iREF

Note

- The page is available only in access point mode.
- The function is supported by SCALANCE W780 / W770.
- The setting can only be configured with the following KEY-PLUG:
  - Access point: W780 iFeatures (MLFB 6GK5 907-8PA00)

When should iREF be used?

With iREF, the data can be sent with the highest possible transmit power. This applies in particular to use cases in which MIMO cannot be used, is of no advantage or when using sector antennas.

Note

Use of iREF with other iFeatures

iREF and other iFeatures (e.g. iPCF) are not compatible with each other and cannot be used at the same time on a device.

Settings

The table has the following columns:

- **Radio**
  Specifies the WLAN interface to which the settings relate.
- **Enable iREF**
  Enable or disable iREF for the required WLAN interface. The result is shown in "Information > WLAN >iFeatures".

  Note

  To be able to use iREF, there must be at least two antennas.

  The antennas are automatically switched to antenna mode RX/TX as soon as iREF is enabled.

AeroScout

Note

- The page is only available
  - SCALANCE W780 / W770
  - in access point mode
- The setting can only be configured with the following KEY-PLUG:
  - Access point: W780 iFeatures (MLFB 6GK5 907-8PA00)

Note

The AeroScout function cannot be combined with other iFeatures (iPCF,iPCF-MC, iREF).

AeroScout can only be used in the 2.4 GHz band according to IEEE 802.11g, IEEE 802.11n and IEEE 802.11n-only.

For more detailed information, please refer to the AeroScout documentation (www.aeroscout.com).

Settings

The table has the following columns:

- **Radio**
  Specifies the WLAN interface to which the settings relate.
- **Enable AeroScout**
  Enable or disable AeroScout support for the required WLAN interface. The result is shown in "Information > WLAN iFeatures". This page is only available in the WBM.

10.1.4.6 Configuring PROFIBUS DP (S7-1200)

The basics of configuring a DP master system (S7-1200)

Distributed I/O

DP master systems that consist of a DP master and DP slaves which are connected via a bus and communicate with one another via the PROFIBUS DP protocol are referred to as distributed I/O.

Firmware version of the S7-1200 CPU

Use of the PROFIBUS functions with the S7-1200, requires CPUs with firmware version 2.0 or higher.

Configuring distributed I/O

Since DP masters and DP slaves are different devices, these instructions only provide a basic configuration procedure. However, the process for configuring distributed I/O is practically identical to that of non-distributed configuration.

Creating the DP master system in the network view

After you have used dragged a DP master and a DP slave (for example, a CM 1243-5 and a CM 1243-5) from the hardware catalog to the network view, connect them both to a PROFIBUS subnet.

Additional information

Observe additional information on the scope of functions in the manuals of the respective device.

DP slaves within the hardware catalog (S7-1200)

DP slaves within the hardware catalog

You will find the DP slaves in the "Distributed I/O" folder of the hardware catalog. Compact and modular DP slaves are located there:

- Compact DP slaves
  Modules with integrated digital/analog inputs and outputs, for example, ET 200L
- Modular DP slaves
  (Interface modules with S7 modules assigned, for example, ET 200M

The available DP master and the desired functionality will determine which DP slaves can be used.

I slaves within the hardware catalog

The CM 1242-5 is, for example, an DP slave that can be configured as intelligent DP slave. You can find it in the hardware catalog at:

- CM 1242-5
  "PLC > SIMATIC S7 1200 > Communication module > PROFIBUS"

DP/DP coupler in the hardware catalog (S7-1200)

Introduction

A DP/DP coupler connects two PROFIBUS DP networks as a gateway so that the DP master from one network can transfer data to the DP master of the other network.

The maximum amount of data that can be transferred is 244 bytes input data and 244 bytes output data.

DP/DP coupler in the hardware catalog

Details of a DP/DP coupler as gateway between two DM master systems are contained in the hardware catalog in the folder "Other field devices > PROFIBUS-DP > Gateways".

Configuring the DP/DP coupler

DP/DP couplers are configured in both PROFIBUS networks, each in their own master systems.

The input and output areas of both networks must thereby be matched to one another. The output data from one end of the DP/DP coupler are accepted as input data at the other respective end and vice versa.

Configurations involving PROFIBUS DP (S7-1200)

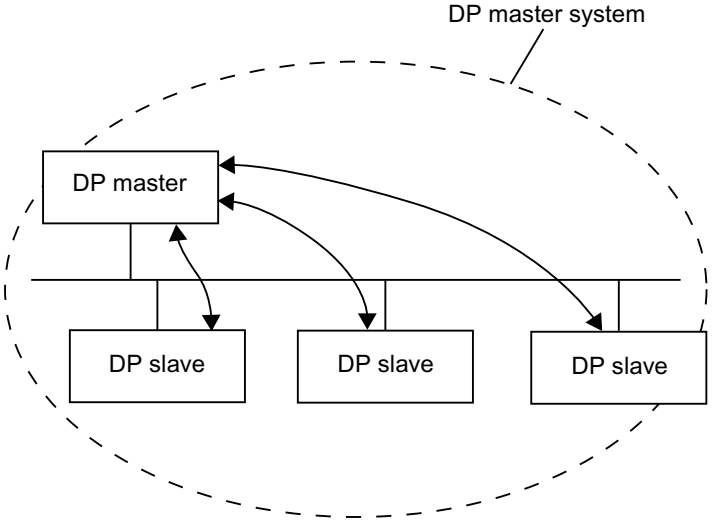
Configurations involving basic DP slaves (S7-1200)

Communication between DP master and DP slave

In the case of a configuration involving simple DP slaves, data is exchanged between the DP master and simple DP slaves, i.e. with I/O modules via the DP master. The DP master sequentially polls each DP slave configured within the DP master system and contained in its polling list. In doing so, it transfers the output data to the slaves and receives their input data in return.

Mono-master system

The configuration with only one DP master is also described as mono-master system. A single DP master with its associated DP slaves is connected to a physical PROFIBUS DP subnet.



Configurations involving intelligent DP slaves (S7-1200)

Definition

DP slaves that feature their own preprocessing program are referred to as intelligent DP slaves (I-slaves).

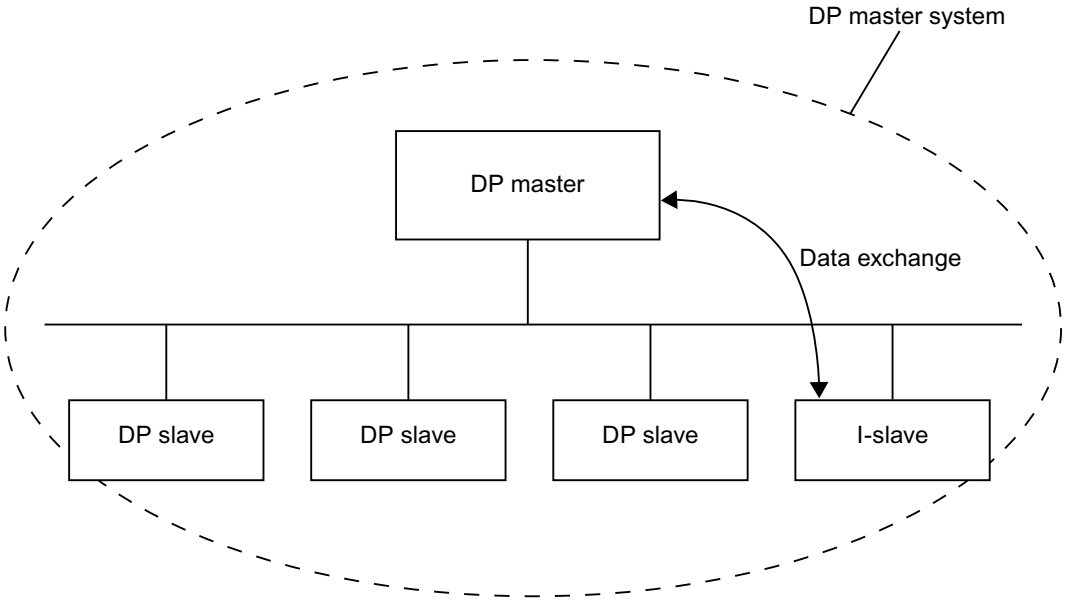
CM 1242-5 is an intelligent DP slave.

I-slave <> DP master data exchange

A higher-level automation system processes the automation task, which is broken down into sub-tasks. The sub-tasks are processed in the lower-level automation systems. The necessary control tasks are processed separately and efficiently in the CPUs as preprocessing programs.

In the case of configurations involving intelligent DP slaves, the DP master only accesses the operand area of the I-slave's CPU, and not the I/O modules of the intelligent DP slave. The operand area must not be assigned to any actual I/O modules in the I-slave. The assignment must be made during I-slave configuration.

The addresses of the data to be exchanged between the master and slave are configured in the transfer area of the I-slave.



Configuring distributed I/O systems (S7-1200)

Creating a DP master system (S7-1200)

Introduction

To create a PROFIBUS DP master system, you need to have one DP master and at least one DP slave. As soon as you connect a DP master to a DP slave via their PROFIBUS DP interfaces, a master-slave link is established.

DP master and DP slave

The DP master must have a PROFIBUS DP interface. When using an S7-1200 CPU with PROFINET interface, you require the communications module CM 1243-5 with PROFIBUS DP interface for use as the DP master.

As DP slave, you can use head modules of the distributed I/O with PROFIBUS DP interface or CPUs as intelligent DP slaves. When using an S7-1200 CPU with PROFINET interface, you require the communications module CM 1242-5 with PROFIBUS DP interface to use the CPU as an intelligent DP slave.

Requirement

- You are in the network view.
- The hardware catalog is open.

Procedure

To create a DP master system, for example with a CPU 1217C , follow these steps:

1. Select a CPU 1217C from the hardware catalog as a potential DP master.
2. Drag the CPU onto the free area within the network view.
3. Select the communications module CM 1243-5 from the hardware catalog.
4. Drag the communications module to the CPU in the network view.
5. Right-click on the DP interface of the communications module.
6. Select "Create master system" from the shortcut menu.

A DP master system with a CPU 1217C is created as the only node via the CM 1243-5 as DP master.

If you connect the DP interface of a DP slave to the DP interface of the DP master, the DP slave will be added automatically to the DP master system. If there is not yet a subnet between the DP master and DP slave, a new subnet is created between the DP master and DP slave.

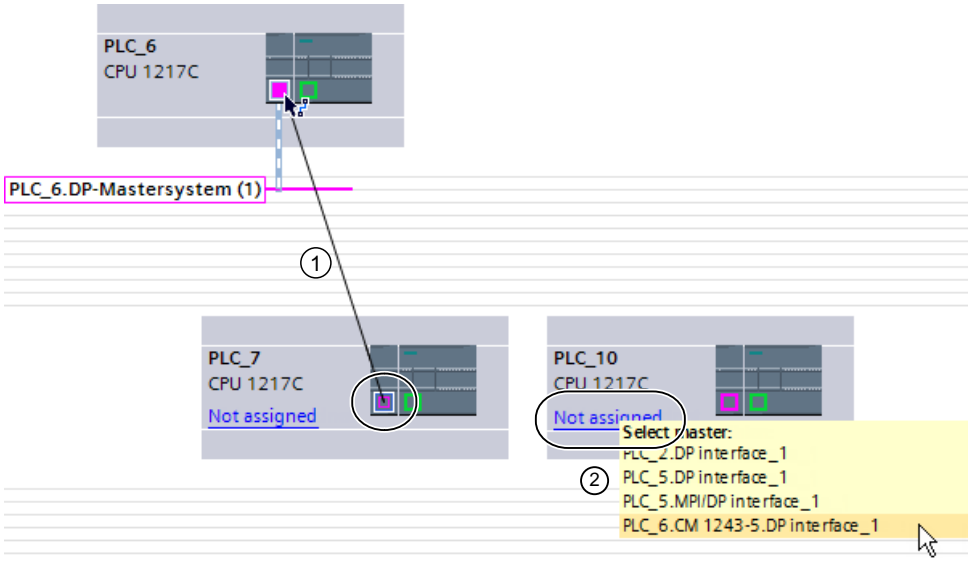
To include, for example, a CPU 1217C with communications module CM 1242-5 as an intelligent DP slave in the DP master system, follow the steps below:

1. Click on the DP interface of the DP master (interface of the CM 1243-5) or DP slave (interface of the CM 1242-5).
2. Hold down the mouse button and draw a connecting line between this DP interface and that of the desired communication partner.

or

1. Click on the hyperlink of the DP slave CPU 1217C with CM 1242-5.
2. Select the required DP master from the list of possible DP masters displayed.

The intelligent DP slave CPU 1217C with CM 1242-5 is included in the DP master system as DP master with CPU 1217C and CM 1243-5 .



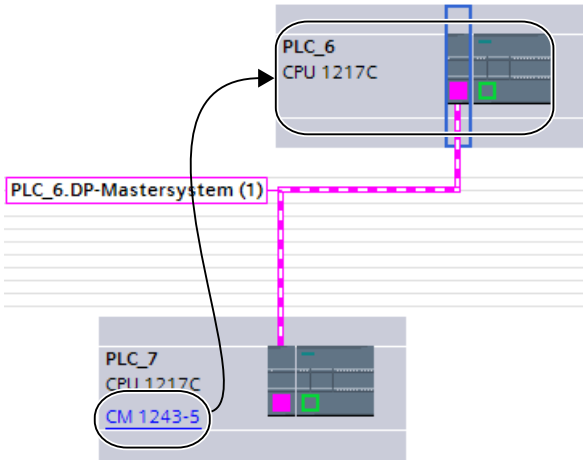
① Drag between the DP interfaces of DP slave to DP master.

② Click on the link of an unassigned DP slave, and a selection of possible DP masters opens.

If necessary, adapt the properties of the PROFIBUS subnet or of the DP master (e.g. PROFIBUS address) in the Inspector window under "Properties".

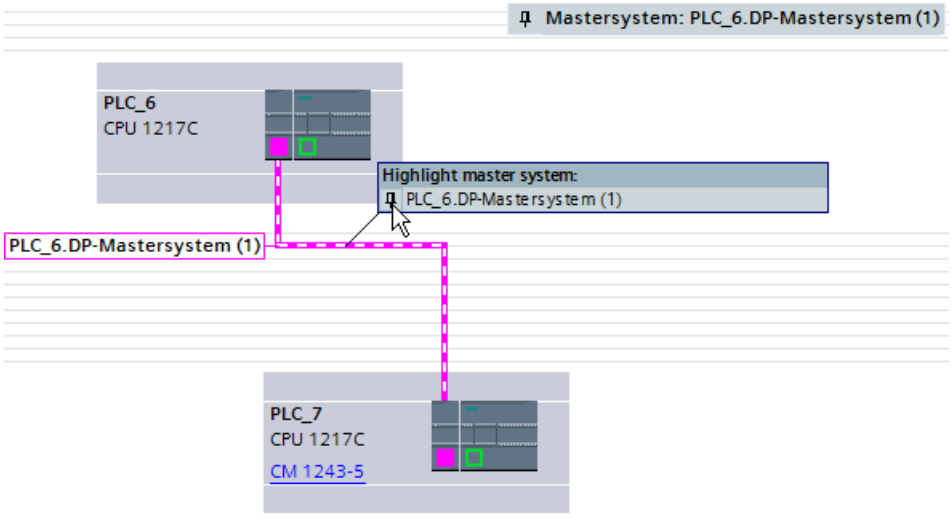
DP master display on the DP slave

When a DP slave is connected to a DP master, the name of the DP master is displayed on the DP slave as a hyperlink. Click the hyperlink to select the associated DP master.



Highlighting applied to the DP master system

When you create a new DP master system, highlighting will be applied to it. This enables you to identify quickly which devices belong to the DP master system. You can also highlight a DP master system yourself by moving the mouse pointer over a subnet. This will result in the names of the available DP master systems being displayed. Click the required DP master system to highlight it.



There are various ways of removing the highlighting from a DP master system:

- Highlight a different DP master system.
- Click on the pin with the name of the DP master system in the top right-hand corner of the network view.

Editing DP master systems and interfaces (S7-1200)

Introduction

Once you have created a DP master system, you also have the option of disconnecting the DP master system from its components. This can result in subnets with DP slaves but without DP master.

Generally, there is no need to edit the interfaces of a DP master.

You can change the name and number on the DP master system.

Disconnection of master or slaves from the DP master system

If you have configured a PROFIBUS CP as a DP master with master system, you can then disconnect the DP master system from the DP master. After this, the device will no longer be connected to the DP master system.

Disconnecting the subnet from a DP master effectively eliminates the master system in the sense that it is no longer assigned to a DP master. However, the individual DP slaves are still interconnected via the subnet.

If you delete the DP slaves or disconnect them from the master system, the master system is then retained on the DP master.

Requirement

- You must be in the network view.
- There has to be a DP master system with one DP master and at least one DP slave.

Disconnecting the DP master from the DP master system

To disconnect the DP master system, proceed as follows:

1. Right-click on the DP master's DP interface.
2. Select "Disconnect from master system" from the shortcut menu.

The selected DP master will be disconnected from the DP master system. A subnet with the DP slaves will be retained.

Adding a DP master to the DP master system

To reassign a DP master to a subnet, proceed as follows:

1. Right-click on the DP master's DP interface.
2. Select "Create master system" from the shortcut menu.
3. Draw connecting lines between the new DP master system and the DP interfaces of the DP slaves.

The DP master combines with the DP slaves to recreate a DP master system.

Editing the properties of a DP master system

To edit the properties of a DP master system, proceed as follows:

1. Move the mouse pointer over a subnet with a DP master system.
2. A message will appear displaying the available DP master systems. Click the one you want to edit. The DP master system will now be color-highlighted.
3. Click on the highlighted DP master system.
4. In the inspector window, edit the DP master system attributes under "Properties > General".

Note

If you click a subnet when no DP master system is highlighted, you will be able to edit the properties of the entire subnet under "Properties" in the inspector window.

Adding DP slaves to the master system and configuring them (S7-1200)

In the network view, you can add various DP slaves from the hardware catalog directly by using the drag-and-drop function or by double-clicking.

Types of DP slaves

For configuration purposes, DP slaves are broken down into the following categories:

- Compact DP slaves
  (Modules with integrated digital/analog inputs and outputs, for example, ET 200L)
- Modular DP slaves
  (interface modules with S5 or S7 modules assigned, for example ET 200M)
- Intelligent DP slaves (I-slaves)
  (SIMATIC S7-300 with, for example, CP 342-5, CPU 315-2 DP, or ET 200S with IM 151-7 CPU)

Slot rules

- Your DP master system should only contain one DP master, but it may contain one or or more DP slaves.
- You may only configure as many DP slaves in a DP master system as are permitted for the specific DP master.

Note

When configuring the DP master system, remember to observe the DP master technical data (max. number of nodes, max. number of slots, max. quantity of user data). User data restrictions may possibly prevent you from being able to use the maximum number of nodes that is theoretically possible.

Requirements

- You are in the network view.
- A DP master system must have been created.

Adding a DP slave to the DP master system

To add a DP slave from the hardware catalog to the DP master system, follow these steps:

1. Select a DP slave from the hardware catalog.
2. Drag-and-drop the DP slave from the hardware catalog into the network view.
3. Draw a connecting line between the DP master's DP interface or a highlighted DP master system and the DP interface of the new DP slave.

A DP master system will be created and the DP slave will be connected to the DP master automatically.

Note

When a DP master system is highlighted, you can double-click the required DP slave in the hardware catalog. This will result in the DP slave being added to the highlighted DP master system automatically.

Disconnecting a DP slave from the DP master system

To disconnect a DP slave from the DP master system, follow these steps:

1. In the network view, right-click on the DP slave's DP interface.
2. From the shortcut menu, select the method for disconnecting from the DP master system:
   - "Disconnect from subnet": The PROFIBUS connection is broken and the device is no longer connected to the DP master system or a subnet.
   - "Disconnect from master system": The DP slave remains connected to the subnet, but is no longer assigned to the DP master system as a DP slave.

The selected DP slave will be disconnected from the DP master system.

You can also disconnect directly using the shortcut menu of the DP slave:

1. In the network view, right-click on the DP slave.
2. Select "Disconnect from DP master/IO system" from the shortcut menu.

The DP slave remains connected to the subnet, but is no longer assigned to the DP master system as a DP slave.

Assigning a DP slave to a new DP master system

To assign an existing DP slave to a new DP master system, follow these steps:

1. Right-click on the DP slave's DP interface.
2. From the shortcut menu, select "Assign to new master".
   It does not matter whether the DP slave concerned is already assigned to another DP master system.
3. From the selection list, select the DP master with the DP master system to which you want to connect the DP slave.
   The selected DP slave is now assigned to a new DP master system.

The "Assign to new subnet" function works in a similar way in that it allows you to connect a DP slave to a new subnet. However, in this case, the DP slave will not be connected to an existing DP master system.

You can also make the assignment directly using the shortcut menu of the DP slave:

1. Right-click the DP slave.
2. From the shortcut menu, select "Assign new DP master/IO system".
3. Select the DP master from the list of available DP masters.

The selected DP slave has now been assigned to a new DP master system.

Note

If there is only one DP master on a subnet, you only have to connect the interface of the DP slave to the subnet of this DP master with drag-and-drop. The DP slave is then connected to the subnet of the DP master immediately, and the DP slave is assigned to this DP master.

Configuring a DP slave

To configure a DP slave, follow these steps:

1. Switch to the DP slave's device view.
2. Select the module you want.
3. Configure the DP slave in the Inspector window.

Hint: Quick configuration of master systems (S7-1200)

If the DP master system has several DP slaves, use drag-and-drop operation to assign to the master in one step all DP slaves that were placed.

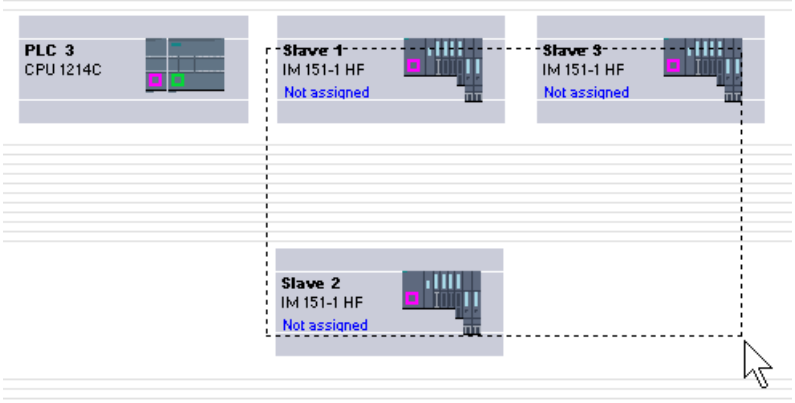
Requirements

DP master and DP slaves are placed in the network view.

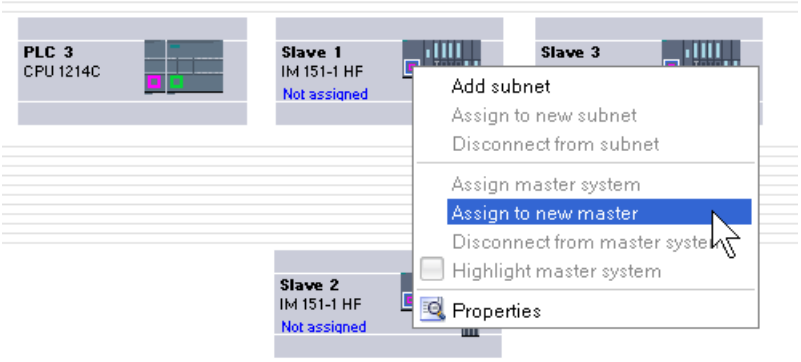
Assigning DP slaves to a DP master system

To do this, follow these steps:

1. Select an appropriate zoom factor so that you can see as many DP slaves as possible in the network view.
2. Arrange the DP slaves in a maximum of two rows.
3. Select all DP interfaces with the mouse cursor (not all devices!). This only works if you begin to drag the mouse cursor outside of the first DP slave and release the mouse button at the last DP slave (selection with the lasso).



4. Select the shortcut menu "Assign to new master" and select the corresponding DP interface for the DP master in the subsequent dialog.



5. The DP slaves are automatically networked with the DP master and combine with it to form a DP master system.

Note

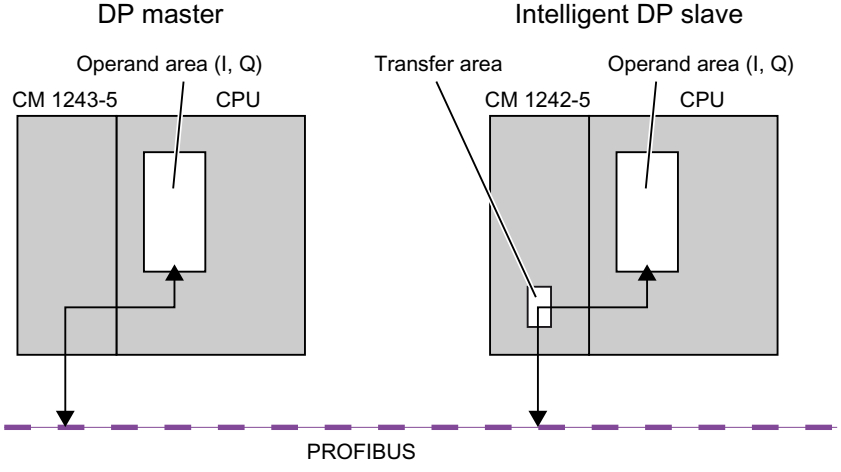
When a DP master system is highlighted, you can double-click on a DP slave in the hardware catalog and thereby quickly add additional DP slaves. This will result in the DP slave being added to the highlighted DP master system automatically.

Configuring intelligent DP slaves (S7-1200)

Adding an I-slave to a DP master system (S7-1200)

Introduction

One of the characteristics of an intelligent DP slave (I-slave) is that the DP master is not provided with I/O data directly by a real I/O, but by a preprocessing CPU. Together with the CP, this preprocessing CPU forms the I-slave.



Difference: DP slave v. intelligent DP slave

In the case of a DP slave, the DP master accesses the distributed I/O directly.

In the case of an intelligent DP slave, the DP master actually accesses a transfer area in the I/O address space of the preprocessing CPU instead of accessing the connected I/O of the intelligent DP slave. The user program running on the preprocessing CPU is responsible for ensuring data exchange between the address area and I/O.

Note

The I/O areas configured for data exchange between the DP master and DP slaves must not be used by I/O modules.

Applications

Configurations involving intelligent DP slaves: I-slave <> DP master data exchange

Procedure

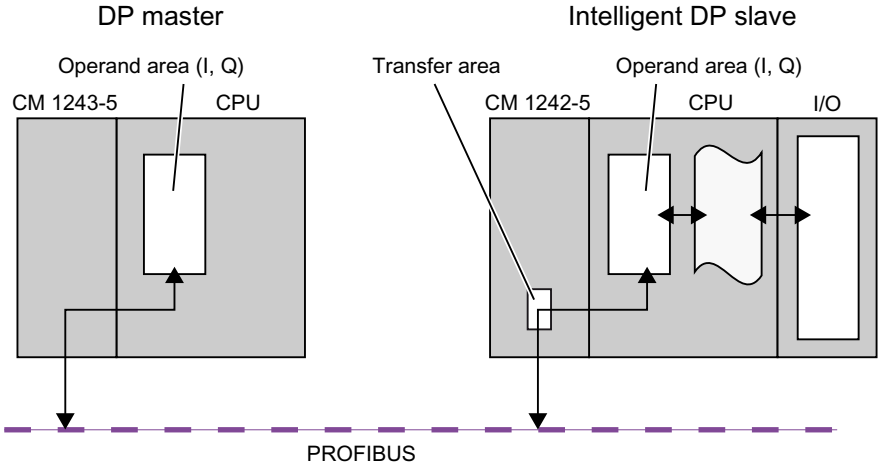
To add an I-slave to a DP master system, follow these steps:

1. In the network view, drag from the hardware catalog to a station one CM 1242-5 as an I-slave and one CM 1243-5 as a DP master.
2. Draw a connecting line between the DP interfaces of both devices.
   This way you connect the I-slave with a DP master in a DP master system.
   Result: You have now set up a DP master system with one DP master and one I-slave.

Configuring access to I slave data (S7-1200)

Data access

The following applies to the CP 1242-5 in its function as I-slave: The addresses for the data transfer area and the address for the I/O modules in the I-slave differ. This means that the start address occupied by an I/O module can no longer be used for the transfer memory. If the higher-level DP master is to access the data of an I/O module in the I-slave, you must configure this data exchange between the I/O module and transfer area in the I-slave user program.



Configuration of the transfer area for the CM 1242-5 (transfer area)

With CM 1242-5, the transfer area for the cyclic PROFIBUS data exchange is configured as transfer area in the parameter group "PROFIBUS interface > Mode > I Slave Communication".

Direct data access from CPU to CPU

Direct data access from CPU to CPU via PROFIBUS is supported by the S7-1200 PROFIBUS CMs only via the PUT/GET services.

Configuring DP slaves as distributed I/O devices (S7-1200)

Configuring an ET 200S (S7-1200)

Slot rules for configuring an ET 200S

The following rules apply when configuring an ET 200S:

- Do not leave any gaps when inserting the ET 200S modules.
- Slot 1: only for PM-E or PM-D Power Modules.
- To the left of an Electronics Module (EM): an EM or a Power Module (PM-E or PM-D) only.

- To the left of Motor Starter (MS): an MS, a PM-D, PM-D Fx (1..x..4) Power Module or a PM-X Power Module only.
- To the left of a PM-X: a motor starter or a PM-D only
- Up to 63 modules and one IM Interface Module are permitted

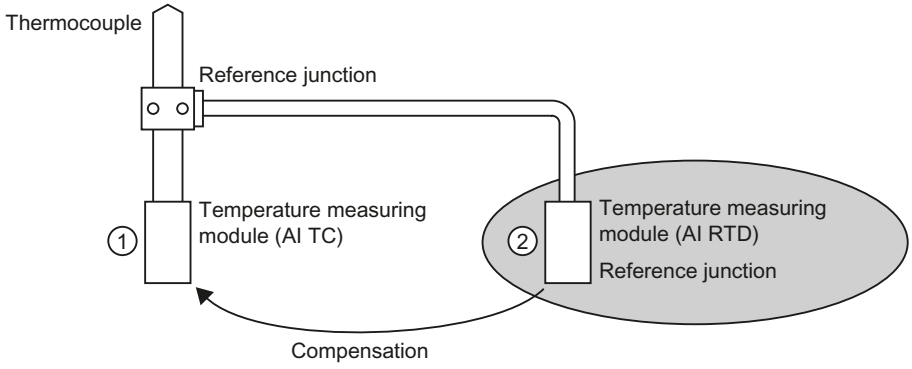
Note

Remember to ensure that the correct PM-E and EM voltage ranges are assigned.

Configuring reference junctions

A reference junction is the connection of a thermocouple to a supply line (generally in the terminal box). The voltage that occurs due to the effects of temperature falsifies the temperature value measured by the module.

On the ET 200S, one channel of the AI RTD module can be programmed as a reference junction. Other AI TC modules can correct their measured values using the temperature at the reference junction as measured by this module.



① Configuring the AI TC:
→ Selection of the reference junction used

② Configuring of the AI RTD:
→ Activation of the reference junction
→ Specifying the slot and channel of the AI RTD

Special characteristics to be noted when assigning parameters for reference junctions

The process of assigning parameters for reference junctions will be explained based on the example of a resistance thermometer with a Pt 100 climatic range that is used for measuring the reference junction temperature.

To assign parameters for the reference junction, follow these steps:

1. In the ET 200S device view, insert an analog electronics module, for example a 2AI RTD HF.
2. Select the module on the rack.

3. Under "Properties > Inputs" in the inspector window, set a channel for the reference junctions function to the "RTD-4L Pt 100 climatic range" measuring range.
4. Select the ET 200S.
5. Under "Properties > Module parameters > Reference junctions" in the inspector window, select the "Reference junction" check box and specify the slot and channel number of the relevant RTD module.
6. Insert the analog electronics module for measuring the temperature using a thermocouple (TC module) and parameterize it with the reference junction number of the RTD module.

Additional information

For additional information on the various types and uses of ET 200S modules, please refer to the operating instructions and the manual titled "ET 200S Distributed I/O System".

For additional information on analog value processing, please see the documentation for the ET 200S distributed I/O system.

Packing addresses (S7-1200)

Introduction

DP slaves and I/O devices from the ET 200S family are configured in the same way as other modular DP slaves and I/O devices. As well as supporting all the standard modular DP slave and I/O device functions, the ET 200S also offers the "Pack addresses" function:

When digital electronics modules requiring an address space of 2 or 4 bits are inserted into the device view, they will initially be spread over a total area of 1 byte. However, the address area actually occupied can be compressed after configuration using the "Pack addresses" function.

	Initial state	After "Pack addresses"
Module	I address	I address
2DI (2-bit)	Byte 10	10.0...10.1
4DI (4-bit)	Byte 11	10.2...10.5

Requirements

- You are in the device view.
- An ET 200S, for example an IM 151-1, must be present.
- A pair of digital electronics modules, for example 2DI AC120V ST, must be inserted into the slots.

Packing addresses

To pack addressed, follow these steps:

1. Select the electronics modules whose addresses are to be packed. The following options are available for selecting multiple electronics modules:
   - Press and hold down <Shift> or <Ctrl> while clicking the relevant electronics modules.
   - Click off the rack and select the required electronics modules by drawing round them with the mouse.
2. Click "Pack addresses" in the shortcut menu for the selected electronics modules.

The address areas for inputs, outputs and motor starters are packed separately. The packed addresses will be displayed in the I address and Q address columns of the device overview.

How packed addresses are generated and structured

If you make use of the "Pack addresses" function, the addresses of the selected electronics modules will be packed in accordance with the following rules:

- The start of the address area is determined by the lowest address of the selected electronics modules: X.0
- If the bit address is not "0", then the next (free) byte address as of which the selected area can be compacted will be selected automatically: (X+n).0
- If no coherent area exists, the addresses will be automatically packed into any available address gaps.

Electronics modules with packed addresses and the same byte address form a packing group.

Unpacking addresses

To unpack addressed, follow these steps:

1. Select one or more electronics modules with packed addresses.
2. Click "Unpack addresses" in the shortcut menu for the selected electronics modules.

The packing groups of the selected electronics modules will be disbanded and the packed addresses for the relevant electronics modules unpacked.

The packing group will also be disbanded and the packed addresses unpacked in the following cases: if you delete electronics modules from a packing group, move electronics modules out of a packing group or insert electronics modules on a free slot within a packing group.

The start addresses of the unpacked electronics modules will be assigned to the next available byte addresses in each case.

Special characteristics of electronics modules with packed addresses

The following special characteristics apply to electronics modules with packed addresses:

- As far as the CPU is concerned, there is no way of assigning a slot for the electronics module. Consequently, the instruction GADR_LGC (SFC 5) outputs error information W#16#8099 "Slot not configured" for the actual slot of the electronic module.
- The instruction LGC_GADR (SFC 49) and SZL-ID W#16#xy91 "module status information" cannot be evaluated for an electronics module.
- The electronics module receives an additional diagnostics address via the DPV1 function, because otherwise the packed addresses would prevent interrupts from being assigned as far as the CPU is concerned.
- The "Insert/remove interrupt" is not possible. This is because the "Pack addresses" and "Insert/remove interrupt" functions are mutually exclusive.

Configuring option handling with standby modules (S7-1200)

You can use the option handling function to prepare the ET 200S with PROFIBUS interface for future expansions (options). This section describes option handling with standby modules.

You do this by assembling, wiring, configuring, and programming the maximum configuration envisaged for the ET 200S and by using cost-effective standby modules (138-4AA00 or 138-4AA10) during assembly until it becomes time to replace them with the necessary electronics modules.

Note

The ET 200S can be completely factory-wired with the master cabling, as no connection exists between a standby module and the terminals of the terminal module (and, in turn, to the process).

Requirement

- ET 200S interface module
  - IM 151-1 STANDARD (6ES7 151-1AA03-0AB0 or higher)
  - IM 151-1 FO STANDARD (6ES7 151-1AB02-0AB0 or higher)
- Power module with option handling
  - PM-E DC24..48V
  - PM-E DC24..48V/AC24..230V

Procedure

To activate option handling, follow these steps:

1. Select the IM 151-1 in the device view and enable it in "Option handling" check box under "Properties > General > Option handling" in the inspector window.
2. Select the numbered check boxes for the slots that are initially to accommodate the standby modules prior to the future electronics modules.
3. Select the power module in the device view and enable it in the "Option handling" check box under "Properties > Addresses" in the inspector window. Reserve the necessary address space for the control and check-back interface in the process image output (PIQ) and process image input (PII).

The assembled standby modules can be replaced with the configured modules at a later date without having to modify the configuration.

Note

The addresses for these interfaces are reserved as soon as you activate option handling on the power module. The "Option handling" function must also be activated on the DP slave (IM 151-1 STANDARD Interface Module). If it is not activated, the addresses reserved for the control and check-back interface will be released again.

Note that activating and deactivating the option handling function repeatedly can change the address of the control and check-back interface.

Option handling may be activated for one PM-E DC24..48V or one PM-E DC24..48V/ AC24..230V Power Module only.

Additional information

For additional information on the assignment and significance of bytes within the process image, option handling with PROFIBUS and the use of standby modules, please refer to the documentation for the ET 200S distributed I/O system.

How option handling works during startup

If "Startup when expected/actual config. differ" is not available, the ET 200S will still start up if a standby module is inserted instead of the configured electronics module and option handling has been activated for the slot concerned.

How option handling works during operation

During operation, the option handling mode varies in accordance with the following:

- Option handling enabled for a slot:
  Either the standby module (option) or the configured electronics module can be plugged into this slot. If any other kind of module is present on this slot, diagnostics will be signaled (no module/incorrect module).
- Option handling disabled for a slot:
  Only the configured electronics module can be plugged into this slot. If any other kind of module is present, diagnostics will be signaled (no module/incorrect module).

Standby module substitute values

- Substitute value for digital inputs: 0
- Substitute value for analog inputs: 0x7FFF

Control and evaluation in the user program

The ET 200S is equipped with a control and feedback interface for the "Option handling" function.

The control interface is located in the process image output (PIQ). Each bit in this address area controls one of the slots from 2 to 63:

- Bit value = 0: Option handling parameterized. Standby modules are permitted.
- Bit value = 1: Option handling canceled. Standby modules are not permitted on this slot.

The check-back interface is located in the process image input (PII). Each bit in this address area provides information about the modules that are actually plugged into slots 1 to 63:

- Bit value = 0: This slot has the standby module, an incorrect module or no module plugged into it.
- Bit value = 1: The configured module is plugged into this slot.

See also

Which modules support option handling? (http://support.automation.siemens.com/WW/view/en/22564754)

Configuring option handling without standby modules (S7-1200)

You can use the option handling function to prepare the ET 200S for future expansions (options) without installing standby modules. This section describes option handling without standby modules.

Note

ET 200S with PROFINET interface

This description refers to the ET 200S with PROFIBUS interface. In principle, option handling for ET 200S with PROFINET interface functions as described here without standby modules. PN interface modules are to be used instead of the DP interface modules listed here. You can find additional information about option handling for ET 200S with PROFINET interface in the corresponding manuals.

Requirement

- ET 200S interface module
  - IM 151-1 HIGH FEATURE (6ES7151-1BA02 or higher)
  - IM 151-1 STANDARD (6ES7 151-1AA05-0AB0 or higher)
- Power module with option handling
  - PM-E DC24..48V
  - PM-E DC24..48V/AC24..230V

Procedure

To activate option handling, follow these steps:

1. Select the IM 151-1 in the device view and enable it in "Option handling" check box under "Properties > General > Option handling" in the inspector window.
2. Select the power module in the device view and enable it in the "Option handling" check box under "Properties > Addresses" in the inspector window. Reserve the necessary address space for the control and check-back interface in the process image output (PIQ) and process image input (PII).
3. Configure the slave's maximum configuration. The selection/clearing of options is controlled via the user program.

   Note

   The addresses for these interfaces are reserved as soon as you activate option handling on the power module. The "Option handling" function must also be activated on the DP slave (IM 151-1 interface module). If it is not activated, the addresses reserved for the control and check-back interface will be released again.

   Note that activating and deactivating the option handling function repeatedly can change the address of the control and check-back interface.

   Option handling may be activated for one PM-E DC24..48V or one PM-E DC24..48V/ AC24..230V Power Module only.

Additional information

For additional information on the assignment and significance of bytes within the process image, option handling with PROFIBUS and the use of standby modules, please refer to the documentation for the ET 200S distributed I/O system.

Control and evaluation in the user program

The ET 200S is equipped with a control and feedback interface for the "Option handling" function.

The control interface is located in the process image output (PIQ). Each bit in this address area controls one of the slots from 1 to 63:

- Bit value = 0: Slot is not available in the actual configuration.
- Bit value = 1: Slot is available in the actual configuration.

The check-back interface is located in the process image input (PII). Each bit in this address area provides information about the modules that are actually plugged into slots 1 to 63:

- Bit value = 0: Slot belongs to an option that is not available or module status is faulty.
- Bit value = 1: The configured module is plugged into this slot.

See also

Sample applications for ET 200S, option handling without standby module (http://support.automation.siemens.com/WW/view/en/29430270)

Configuring the ET 200S in DPV1 mode (S7-1200)

PROFIBUS DPV1 enables you to access extended PROFIBUS functions.

Requirement

- You must be in network view.
- A DP master with DPV1 functionality must be available.
- A master-slave connection must be established with PROFIBUS.

Procedure

To switch the DP slave over to DPV1, follow these steps:

1. Select the DP slave.
2. Under "Properties > Module parameters" in the Inspector window, select "DPV1" mode from the "DP interrupt mode" drop-down list.

or

1. Select the DP master.
2. In the I/O communications table, select the row with the connection between the DP master and the desired DP slave.
3. Under "Properties > Module parameters" in the Inspector window, select "DPV1" mode from the "DP interrupt mode" drop-down list.

Special characteristics

The parameters are subject to interdependencies, which are outlined below:

Parameter	DPV0 mode	DPV1 mode
Operation when target configuration does not match actual configuration	Fully available	Fully available
Diagnostics interrupt (OB 82)	Not available, not set	Fully available
Hardware interrupt (OB 40 to 47)	Not available, not set	Fully available
Insert/remove interrupt (OB 83)	Not available, not set	Only available when addresses are not packed. "Startup when target configuration does not match actual configuration" is activated automatically along with an insert/remove interrupt.

Interrupts in the case of modules with packed addresses

If the module is capable of triggering interrupts and the bit address is not equal to 0 because of packed addresses, you will need to assign a diagnostics address in the ET 200S address dialog.

The diagnostics address is required for the purpose of assigning a DPV1 interrupt to the module as an interrupt trigger. The CPU will only be able to assign an interrupt correctly and store information on the interrupt in the interrupt OB start information/in the diagnostics buffer if the module concerned has this "unpacked" address. In this context, the CPU cannot make use of "packed" addresses.

From the point of view of interrupt processing (interrupt OB), this means the module will have the assigned diagnostics address, but from the point of view of processing the input and output data in the user program, the module will have the packed addresses.

Note

When the module addresses are packed, the insert/remove interrupt for the ET 200S is unavailable.

Using GSD files (S7-1200)

GSD revisions (S7-1200)

What you need to know about GSD revisions

The properties of DP slaves are made available to configuration tools by means of GSD files. Functional enhancements in the area of the distributed I/O will have an effect on the GSD specification, for example, they will require the definition of new keywords. This results in the versioning of the specification. In the case of GSD files, the version of the specification on which a GSD file is based is called a "GSD revision". From GSD revision 1, the GSD revision must be included as a keyword "GSD_revision" in GSD files. GSD files without this keyword will therefore be interpreted by configuration tools as GSD revision "0".

GSD files can be interpreted up to GSD revision 5. This means that DP slaves that support the following functions, for example, will be supported:

- Diagnostic alarms for interrupt blocks
- Isochronous mode and constant bus cycle time
- SYNC/FREEZE
- Clock synchronization for DP slaves

Installing the GSD file (S7-1200)

Introduction

A GSD file (generic station description file) contains all the DP slave properties. If you want to configure a DP slave that does not appear in the hardware catalog, you must install the GSD file provided by the manufacturer. DP slaves installed via GSD files are displayed in the hardware catalog and can then be selected and configured.

Requirement

- The hardware and network editor is closed.
- You have access to the required GSD files in a directory on the hard disk.

Procedure

To install a GSD file, follow these steps:

1. In the "Options" menu, select the "Manage general station description files (GSD)" command.
2. In the "Installed GSDs" tab, select the directory in which the GSD files are stored.
3. Choose one or more files from the list of displayed GSD files.
4. Click on the "Install" button. The selected GSD files are now being installed.

5. To create a log file for the installation, click on the "Save log file" button.
   Any problems during the installation can be tracked down using the log file.
6. Click "Close". You will be notified that the DP slaves from the installed GSD files are entered into the hardware catalog. This process can take a few seconds.

By installing a GSD file you have added the DP slave described in the file to the hardware catalog.

Result

You will find the new DP slaves installed by means of GSD files in the hardware catalog under "Additional field devices > PROFIBUS DP".

GSD files are always saved together with the project, which means all information relevant for display of the device (including symbols) is also available in the saved project.

See also

Overview of hardware and network editor (Page 525)

Deleting GSD file (S7-1200)

Introduction

You can delete installed DP slaves using GSD files. These DP slaves are then no longer displayed in the hardware catalog.

Requirement

- The hardware and network editor is closed.
- The hardware catalog contains DP slaves installed using GSD files.

Procedure

To delete a GSD file, follow these steps:

1. In the "Options" menu, select the "Manage general station description files (GSD)" command.
2. In the "Installed GSDs" tab, select the directory in which the GSD file is stored.
3. Select the file that is to be deleted from the list of displayed GSD files.
4. Click the "Delete" button.

The selected GSD file was deleted and the DP slave is no longer located in the hardware catalog.

Finding unused GSD files (S7-1200)

Introduction

When you add a GSD-based hardware component to your hardware configuration, different GSD files are added to the project data. If the GSD-based hardware component is once again deleted from the configuration, the GSD files remain in the data inventory of the project. The GSD files are no longer needed in the project if the corresponding GSD-based hardware component is not used in the hardware configuration again. You can find files that are no longer needed in the project with a search function and delete them.

Requirement

- A project must be open.
- The project must not be online.
- The project must not be write-protected.

Procedure

To find and delete unused GSD files, follow these steps:

1. In the "Options" menu, select the "Manage general station description files (GSD)" command.
2. Click on the "Find unused GSDs" button in the "GSDs in the project" tab.

   Note

   Unused GSD files are only available if at least one GSD-based hardware component was added to the configuration and then removed from the configuration. Unused GSD files in the project libraries are included in the search. GSD-based hardware components that are still being used in the configuration are not considered to be unused.

   Installed GSD-based hardware components that are included in the hardware catalog but have not been used in the hardware configuration of the project yet, could not have stored unused GSD data in the project.

3. If GSD files that are no longer needed were found in the project data, you can now delete them. To do so, click the "Delete" button.

GSD files that are no longer needed are deleted from the project data. The GSD-based hardware component is still available in the hardware catalog and, if necessary, can be added to the configuration once again. The necessary GSD data are once again copied to the project data in this case.

Note

If you delete a GSD-based hardware component with "Options > Manage general station description files (GSD) > Installed GSDs", it is deleted from the hardware catalog. However, unused GSD files of this hardware component can still be a part of the project data and can be deleted with the function for finding the unused GSD files.

NOTICE
Actions that cannot be undone When you delete GSD files that are not being used as an action, the action stack for actions that can be undone is deleted. This means the action of deleting unused GSD files as well as all previous actions can no longer be undone.

See also

Basics of undoing and redoing actions (Page 316)

Configuring GSD-based DP slave (S7-1200)

DP slaves that you have inserted through installation of a GSD file can be selected as usual via the hardware catalog and inserted in the network view. If you want to insert the modules of the GSD-based DP slaves, you must take into account some particular details.

Requirement

- You have installed a DP slave using a GSD file.
- You have inserted the head module in the network view in the usual manner.
- The device overview opens in the device view.
- The hardware catalog is open.

Procedure

To add the modules of a GSD-based DP slave, proceed as follows:

1. In the hardware catalog, navigate to the modules of the GSD-based DP slave.
   GSD-based DP slaves, also referred to as DP standard slaves, can be found in the "Other field devices" folder of the hardware catalog.
2. Select the desired module.
3. Use drag-and-drop to move the module to a free space in the device overview.
4. Select the module in the device overview to edit parameters.

You have now inserted the module in a free slot of the GSD-based DP slave and can edit its parameters.

Note

You can see only the GSD-based DP slave in the graphic area of the device view. The added modules of GSD-based DP slaves are only found in the device overview.

Preset configuration

For modules with an adjustable preset configuration, you can change this configuration in the inspector window under "Properties > Preset configuration".

10.1.4.7 Configuring PROFINET IO

What you need to know about PROFINET IO

What is PROFINET IO?

PROFINET IO

PROFINET is an Ethernet-based automation standard of PROFIBUS Nutzerorganisation e.V. (PNO) which defines a manufacturer-neutral communication, automation and engineering model.

Objective

The objective of PROFINET is:

- Integrated communication via field bus and Ethernet
- Open, distributed automation
- Use of open standards

Architecture

The PROFIBUS User Organisation e.V. (PNO) has designated the following aspects for PROFINET architecture:

- Communication between controllers as components within distributed systems.
- Communication between field devices, such as I/O devices and drives.

Implementation by Siemens

The demand for "Communication between controllers as components within distributed systems" is implemented by "Component Based Automation" (CBA). Component Based Automation is used to create a distributed automation solution based on prefabricated components and partial solutions.

The demand for "Communication between field devices" is implemented by Siemens with "PROFINET IO". Just as with PROFIBUS DP, the complete configuration and programming of the components involved is possible using the Totally Integrated Automation Portal.

The following sections deal with the configuration of communication between field devices using PROFINET IO.

Overview of RT classes

RT classes in PROFINET IO

PROFINET IO is a scalable, real-time communication system based on Ethernet technology. The scalable approach is reflected in several real-time classes:

- **RT:** Transmission of data in prioritized, non-isochronous Ethernet frames. The required bandwidth is within the free bandwidth area for TCP/IP communication.
- **IRT:** Isochronous transmission of data with high stability for time-critical applications (for example, motion control). The required bandwidth is from the area of bandwidth reserved for cyclic data.

Depending on the device, not all real-time classes are supported.

Which IO controllers and IO devices support which PROFINET functions?

Additional information and overviews

In the following Siemens Industry Online Support article (http://support.automation.siemens.com/WW/view/en/44383954), you will find an overview of the PROFINET IO controllers and IO devices that support the following PROFINET functions:

- Isochronous real-time communication (IRT)
- Prioritized startup
- Media redundancy (MRP)
- PROFIenergy
- Shared device
- I-device
- Isochronous mode of process data

The functions are explained in the following sections, but without naming the respective hardware that supports these functions.

In the hardware catalog, you can also find an overview of the supported functions in the description below the selected components.

You can also find a description of PROFINET in the respective current STEP 7 version here (http://support.automation.siemens.com/WW/view/en/49948856).

Connecting existing bus systems

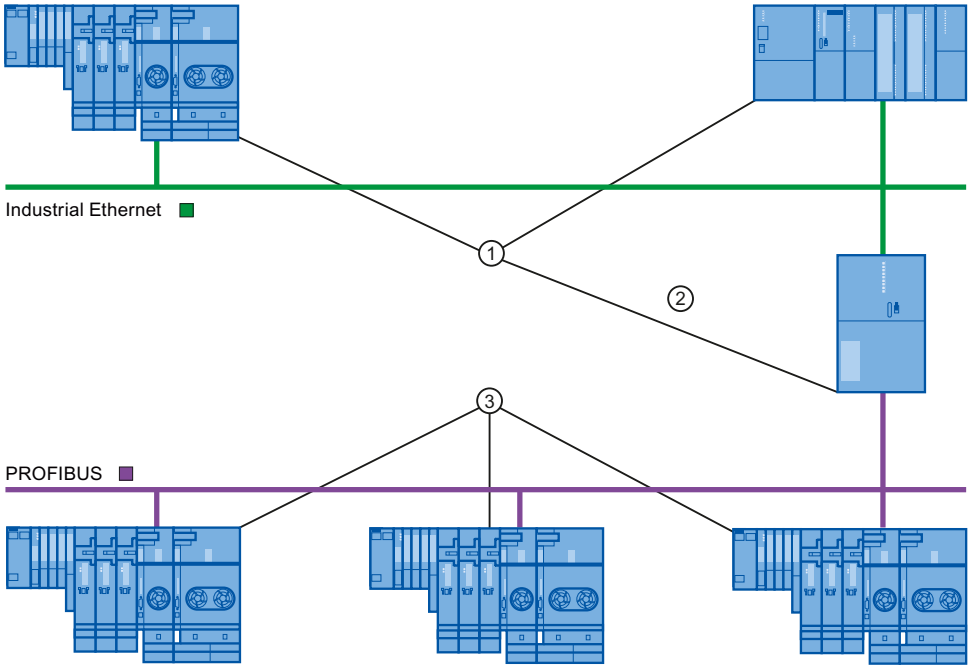
Connection of PROFINET and PROFIBUS

PROFINET IO and PROFIBUS DP can be connected with each other as follows:

- via Industrial Ethernet:
  To connect the two network types, Industrial Ethernet (plant control level) and PROFIBUS (cell level/field level), use the IE/PB link, for example.
- via Industrial Wireless LAN:
  PROFIBUS devices, for example, can be connected to PROFINET IO via a wireless LAN/PB link. This allows existing PROFIBUS configurations to be integrated into PROFINET.

AS interface devices can be connected by an IE/AS-i link PN IO to the interface of a PROFINET device. This allows the existing AS-i network to be integrated into PROFINET.

The following figure shows the connection of a PROFIBUS subnet via a PROFINET device with proxy functions.



① PROFINET devices
② PROFINET device with proxy functions (for example, IE/PB link)
③ PROFIBUS devices

PROFINET device with proxy functions used as proxy for a PROFIBUS device

The PROFINET device with proxy functions is the proxy for a PROFIBUS device on the Ethernet. The proxy functionality enables a PROFIBUS device to communicate with all devices on the PROFINET and not just with its master.

Existing PROFIBUS systems can easily be integrated into PROFINET communication using the proxy functions.

If, for instance, you connect a PROFIBUS device via an IE/PB link to PROFINET, the IE/PB link acts as a proxy for the PROFIBUS components for communicating via PROFINET.

Configuration using IE/PB link PN IO

Configuration using IE/PB Link PN IO

You can use the IE/PB Link PN IO to connect PROFIBUS DP configurations to PROFINET IO.

From the CPU perspective, the PROFIBUS DP slaves are connected to the same network as the IE/PB Link PN IO. These slaves have the same device name and IP address as the IE/PB Link PN IO, but different device numbers. Furthermore, each also has a specific PROFIBUS address.

In the properties of the IE/PB link, the PROFIBUS addresses of the connected DP slaves are displayed in addition to the PROFINET device numbers, as this device has two addressing schemes.

Handling device numbers and PROFIBUS addresses on the master system

During placement, the same number is assigned for the PROFINET device number and the PROFIBUS address.

In the Inspector window under "General Properties > PROFINET device number", you can find an overview of the device numbers used and the PROFIBUS addresses of an IE/PB link. The device number can also be changed here. You can also set that the device number and the PROFIBUS address should or should not always be identical. If the "PROFINET device number=PROFIBUS address" option is activated, you do not have to update the device number when the PROFIBUS address changes.

The PROFIBUS address can be changed in the properties of the PROFIBUS device.

Restrictions

The following restrictions apply to DP slaves configured as described above on the PROFIBUS subnet of an IE/PB link:

- No pluggable IE/PB link
- No pluggable DP/PA link
- No pluggable Y link
- Not CiR-compliant
- No pluggable redundant slaves
- No isochronous transmission / constant bus cycle time can be configured
- SYNC/FREEZE instructions ("DPSYC_FR") of a CPU on the the Ethernet subnet for DP slaves behind the IE/PB-Link are not supported.

See also

Connect the DP slave via the IE/PB Link to a PROFINET IO system (Page 1800)

Configuration using IWLAN/PN link

Maximum number of devices in a IWLAN segment

If an Ethernet subnet is set up as wireless network (IWLAN = Industrial Wireless LAN), cyclic data exchange between IO controllers and IO devices is possible via a wireless route.

On one side of the wireless route there are fixed installed access points (for example, SCALANCE W 788) and on the other side mobile stations (with, for example IWLAN/PB links with PROFIBUS devices).

If the action radius of the mobile stations is large, it may be necessary to install several access points (SCALANCE W 788). Since each access point forms a segment with its wireless range, the IWLAN is made up of a series of segments.

The mobile devices "on the one side" of the wireless link with their IWLAN/PB links can move along the segments.

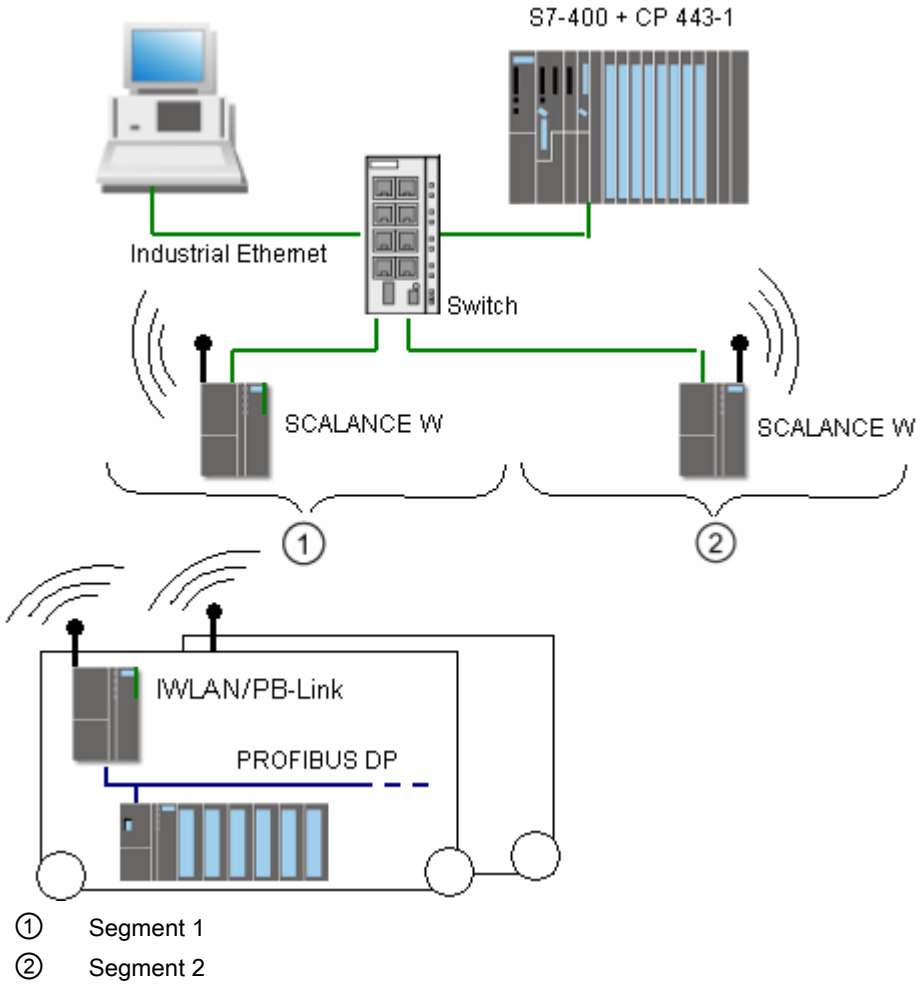
Special feature

If several IWLAN/PB links are located within a segment, they have to share the bandwidth that is available for wireless transmission. This leads to a lengthening of the update time for these devices.

Example

In the following example there are two IO devices (IWLAN/PB link) with a segment.

If no more than a maximum of two IWLAN/PB links are present in a IWLAN segment at the same time, enter a "2".



① Segment 1
② Segment 2

Configure PROFINET IO

Addressing PROFINET devices

Assigning addresses and names to PROFINET devices

In this chapter you will learn which address and naming conventions are valid for the PROFINET devices.

IP addresses

All PROFINET devices work with the TCP/IP protocol and therefore require an IP address for Ethernet operation.

You can set the IP addresses in the module properties. If the network is part of an existing company Ethernet network, ask your network administrator for this data.

The IP addresses of the IO devices are assigned automatically, usually at CPU startup. The IP addresses of the IO devices always have the same subnet mask as the IO controller and are assigned in ascending order, starting at the IP address of the IO controller.

Device names

Before an IO device can be addressed by an IO controller, it must have a device name. This procedure was chosen for PROFINET because names are easier to administer than complex IP addresses.

Both the IO controller as well as IO devices have a device name. When the "Generate PROFINET device name automatically" option is activated, the device name is automatically derived from the name configured for the device (CPU, CP or IM):

- The PROFINET device name is made up of the name of the device (for example, the CPU), the name of the interface (only with multiple PROFINET interfaces) and optionally the name of the IO system:
  <CPU name>.<Name of the interface>.<IO system name>
  You cannot change this name directly. You change the PROFINET device name indirectly, by changing the name of the affected CPU, CP or IM in the general properties of the module. This PROFINET device name is also displayed, for example, in the list of accessible devices. If you want to set the PROFINET device name independently of the module name, you have to deactivate the "Generate PROFINET device name automatically" option.
- A "converted name" is generated from the PROFINET device name. This is the device name that is actually loaded into the device.
  The PROFINET device name is only converted if it does not comply to the rules of IEC 61158-6-10. You cannot change this name directly either.

Rules for the converted name

The rules for the converted name are listed in the following section. If the converted name is **not** different from the name of the module, the name of the module must comply with this rule.

- The name consists of one or more labels , which are separated by a dot [.].
- Restricted to a total of 240 characters (lower case letters, numbers, dash, or dot)
- A name component within the device name, which means a character string between two dots, must not exceed 63 characters.
- A name component consists of the characters [a-z, 0-9].
- The device name must not begin or end with the "-" character.
- The device name must not begin with a number.
- The device name form n.n.n.n (n = 0, ... 999) is not permitted.
- The device name must not begin with the string "port-xyz" or "port-xyz-abcde" (a, b, c, d, e, x, y, z = 0, ... 9).

Example of device names

```
device-1.machine-1.plant-1.vendor
```

If you assign this name to a CPU, for example, STEP 7 will not convert it since it conforms to the rules described above.

Device number

In addition to the device name, a device number is also automatically assigned when an IO device is plugged in. You can change this number.

Devices in the PROFINET subnet

In a PROFINET subnet the maximum allowable number of devices is monitored during configuration.

See also

Assigning the device name and IP address (Page 1769)

Retentivity of IP address parameters and device names (Page 1779)

Assigning the device name and IP address

Assigning an IP address and subnet mask for an IO controller the first time

There are various options for this.

During parameter assignment of the PROFINET interface you must specify if the IP address is set in the project (which means in the hardware configuration) or if the IP address is to be set on the device.

Assignment of an IP address	Comments
Option "IP address is set in the project": The IO controller receives the IP address by loading the hardware configuration, for example, via one of the PROFINET interfaces, by means of the PROFIBUS interface or via the MPI interface.	When the hardware configuration is loaded to the IO controller (e.g. CPU), the IP address and the device name, if set, are also loaded. Example PROFINET interface: 1. Connect your programming device/PC to the same network as the relevant PROFINET device. The interface of the PG/PC must be set to TCP/IP (Auto) mode. 2. Have a list of accessible devices displayed. 3. Select the target device by using its MAC address and load the hardware configuration including the configured IP address (IP address is then saved retentively). If your PROFINET device has an MPI or PROFIBUS DP interface, connect your programming device/PC directly to the PROFINET device via the MPI or PROFIBUS DP interface. The configured IP address is applied when the hardware configuration is loaded.
Option "IP address is set directly at the device": • Assign online • Assignment by user program (instruction IP_CONFIG for S7-300/400, T_CONFIG for S7-1200/1500) • Assign via CPU display (S7-1500) • Higher-level IO controller makes assignment (only with I-devices)	If you have selected this option in the properties of the PROFINET interface, the IP address can be assigned by the online and diagnostics editor, by the primary setup tool, or by the user program ("IP_CONFIG" instruction). This option is set automatically if the option "Multiple use IO system" has been enabled in the properties of the PROFINET IO system (standard machine project). In case of an S7-1200-CPU, make sure that access to the CPU is not protected by a password. If the CPU is write-protected, no IP address and no device name can be assigned directly on the device.

Commissioning a PROFINET interface

Further details of how to commission a PROFINET interface can also be found in the operating instructions for the PROFINET devices of the SIMATIC family.

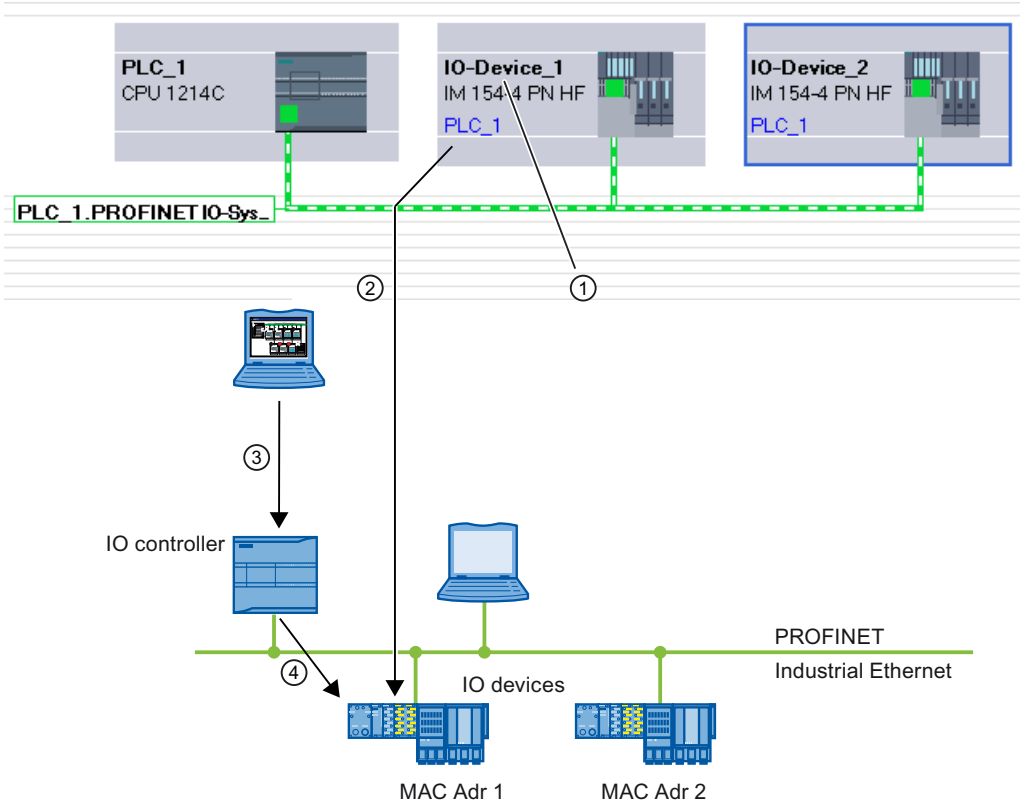
Assignment of device name for IO devices when the "Support device replacement without exchangeable medium" option is selected

For IO controllers with the "Support device replacement without exchangeable medium" option selected, you do not have to assign device names to the IO devices locally, for example in the event of device replacement. Another application is automatic commissioning, in which the CPU automatically assigns the device name and IP address parameters to the IO devices during startup.

Requirement: The ports of the devices are interconnected, and the devices involved support LLDP. The devices have been put into the delivery state or – for S7-1500 CPUs version V1.5 and higher – the "Permit overwriting of device names of all assigned IO devices" option is selected for the IO controller ("Advanced options > Interface options" area of the properties of the PROFINET interface).

Assignment of device name and address for an IO device

The following graphic illustrates the process for assigning the device name and address. This process does not apply when the "Support device replacement without exchangeable medium" option is selected.



① Each device receives a name; STEP 7 automatically assigns an IP address.

② From the name, STEP 7 generates a PROFINET device name that you assign to an IO device online (MAC address) and that is written to the device.

③ You load the configuration to the IO controller.

④ The IO controller assigns the appropriate IP address to the IO device with the assigned PROFINET device name during startup.

Changing the device name and IP address

You can change the name and IP address manually. The device name must first be changed in the configuration in order to subsequently assign it to the IO device via the memory card or online with programming device/PC.

Offline with memory card:

1. Place the configured data (device name: for example, turbo-3) for the IO device on the MMC in the PG/PC. Use the command "SIMATIC Card Reader > Save Device Name to Memory Card" in the "Project" menu for this.
2. Then insert the MMC into the IO device. The IO device automatically adopts the configured device name.

Online with programming device/PC:

1. Connect the programming device/PC directly to the Ethernet subnetwork via the PROFINET interface.
2. In the network view, select the subnet or I/O device, and click the "Assign device name" command:
   - Either in the shortcut menu of the selected subnet / I/O device or
   - in the menu bar of the graphic view on the corresponding button.
3. In the "Assign PROFINET device name" dialog box, select the suitable PG/PC interface to connect to the Ethernet subnet. All configured PROFINET device names are in the top drop-down list. Select a PROFINET device name from it and select the IO device to receive this device name from the table at the bottom. You can filter the display of devices in the table according to various criteria.
4. You can easily identify the device using the "Flash LED" button.
5. Click "Assign name".

The IO controller recognizes the IO device by its device name and automatically assigns the configured IP address to it.

IP address assignment for special IO devices

Special IO devices, for example, SCALANCE X, S7-300 CPs, support the option of assigning the IP addresses not from the IO controller during startup. In this case, the IP address is assigned in a different way. The option is called "IP address is set directly at the device". For additional information, refer to the manual of the respective PROFINET device of the SIMATIC device family.

Another special case is the option "IP address is set directly at the device" in the "IP protocol" area of the Ethernet address properties of an IO device. This option is set automatically when the option "Multiple use IO system" is selected for a standard machine project in the associated PROFINET IO system. In this case, an adapted IP address is not assigned by the IO controller until the IO controller itself has received a local IP address.

Requirement for additional procedures when assigning IP address and device name

If the IO device, as described above, should not obtain the IP address or device name from the IO controller, proceed as follows:

1. Select device or network view.
2. Open the properties of the respective PROFINET device and select the area "PROFINET interface [X1]" > "Ethernet addresses".
3. Select the option "IP address is set directly at the device" under "IP protocol" or the option "PROFINET device name is set directly at the device" under "PROFINET".

Rules

If the "IP address/device name is set directly at the device" option is used for a PROFINET device, note the following:

- The subnet part of the IP address of the IO device must match the subnet part of the IP address of the IO controller.
- The corresponding PROFINET device cannot be used as a gateway.

See also

Enabling device replacement without exchangeable medium (Page 1798)

Example of the assignment of the device name

In this example you assign device names to a PROFINET IO controller and a PROFINET IO device. To make assignment easier, the device names should also contain the names of the PROFINET IO system.

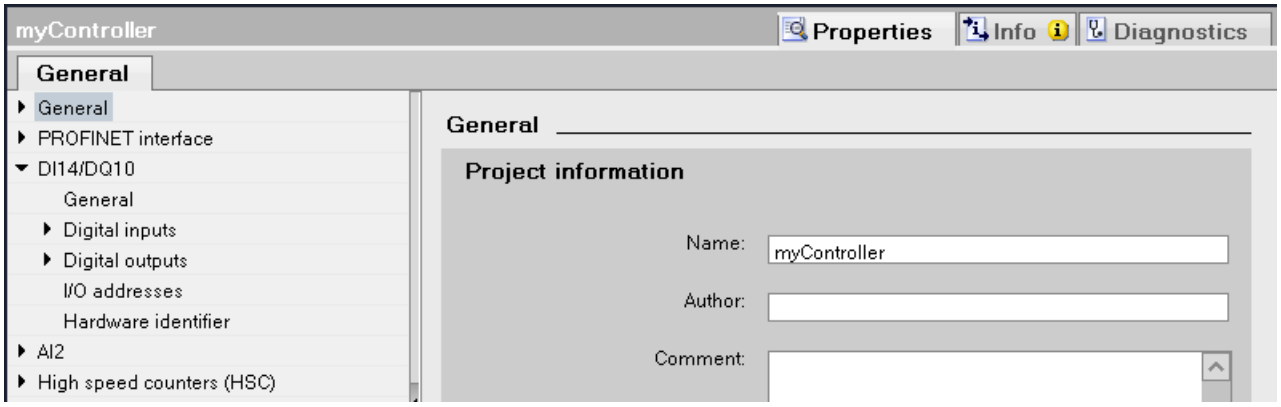
Requirement

- You must be in the network view.
- A CPU 1214C (V2.0 or higher) must be available in the network view.
- An interface module IM 151-3PN exists.
- The PROFINET interfaces of both modules are networked.

Procedure

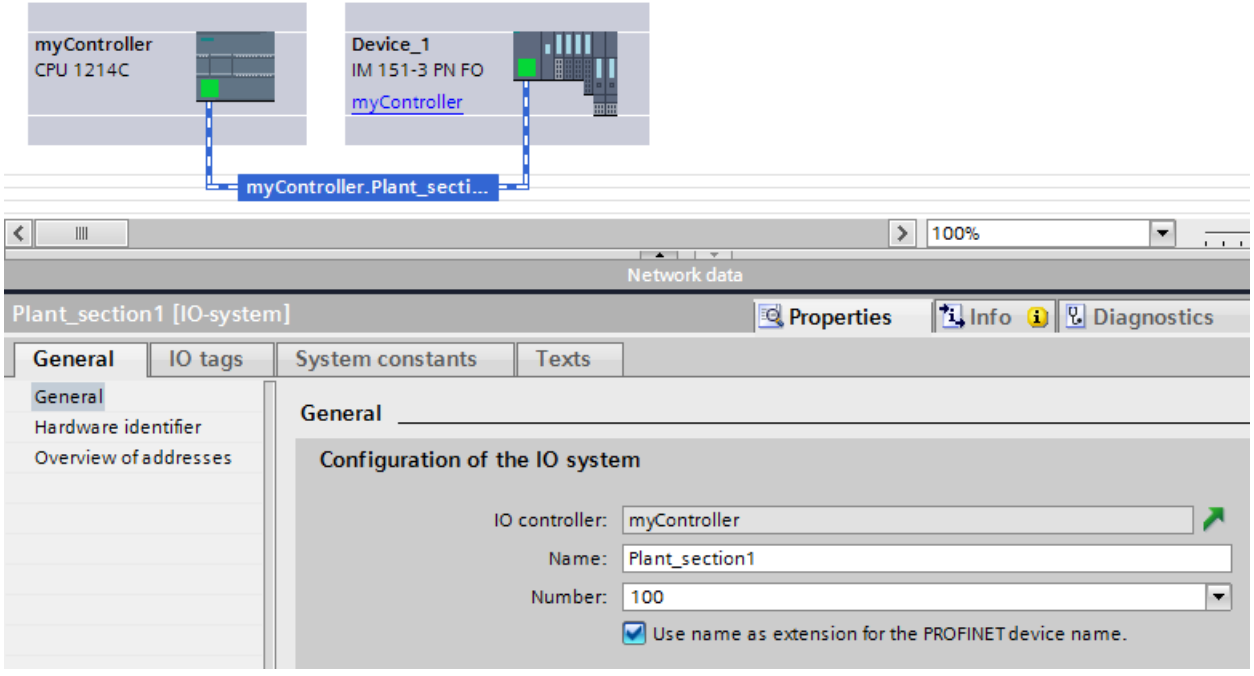
To assign the names, follow these steps:

1. Select the CPU.
   Make sure that you have selected only the CPU and not the complete device!
2. Assign the name "myController" in the Inspector window, under "General".

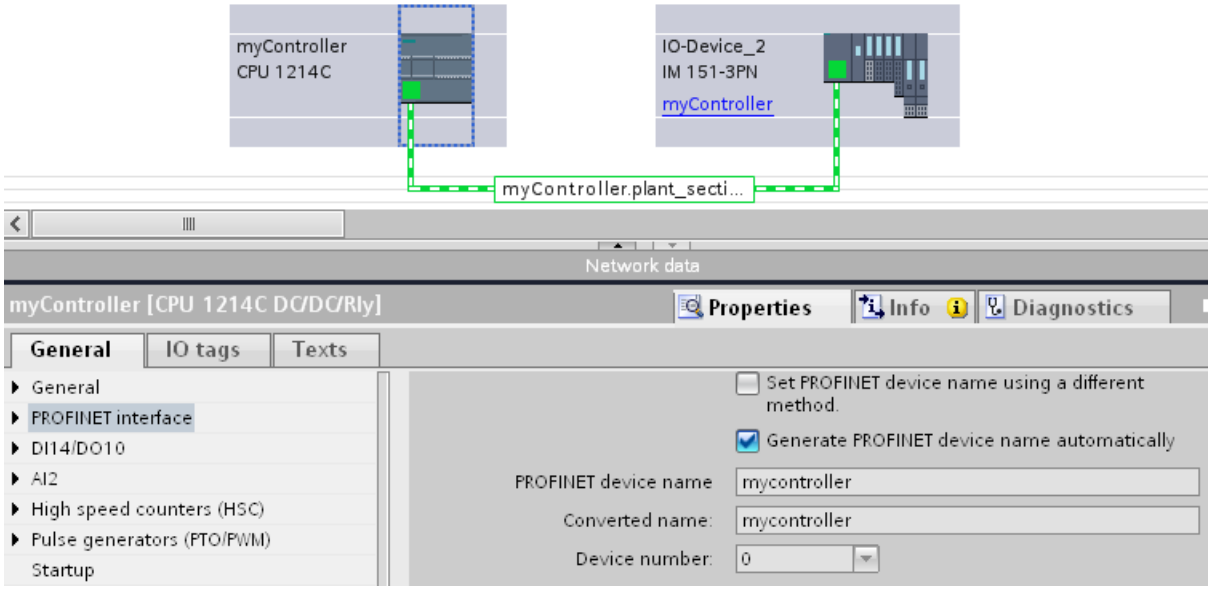


3. Select the interface module.
   Ensure that you have selected only the interface module and not the complete ET 200S device.
4. Assign the name "Device_1" in the Inspector window, under "General".
5. Right-click on the PROFINET IO system and select the "Properties" command.

6. Assign the name "Plant_section1" to the IO system and select the check box "Use name as extension for PROFINET device names".



7. You can find the automatically generated PROFINET device names at the selected device in the Inspector window, at "PROFINET interface".



The PROFINET device name corresponds to the name of the module (with the name of the IO system as extension) with the difference that only lower case text is used. Background: No distinction is made between upper and lower case ("case insensitive") for the storing of the name.

If you want to specify the device name independently of the module name, you have to deactivate the "Generate PROFINET device name automatically" option. The PROFINET device name can be edited in this case.
The converted name is displayed below. This is the name that is automatically generated from the PROFINET device name and satisfies the DNS conventions. If you work with STEP 7, you do not require this name. This name is displayed here as a check and corresponds to the name that is stored in the device. If you work with other tools that are able to record the data exchange and read the actual device names, then you find the converted names.

Other special features

For PROFINET devices with multiple PROFINET interfaces, the name of the interface is attached to the name of the module, separated by a dot.

Example:

- Name of the module: myController
- Name of the interface: Interface_1
- PROFINET device name: mycontroller.interface_1

Assign device name via communication table

Introduction

You can assign the device names of PROFINET IO devices configured offline to the devices online. You can do this in the table area of the network view in the table "I/O communication". You can also assign the device names to several devices at the same time.

"Online assignment" tab

In the I/O communication table, you will find the tabs "Offline configuration" and "Online assignment". In the "Online assignment" tab, you can assign the PROFINET device names that were assigned offline to the corresponding IO devices online. To do this, use the buttons "Check devices" and "Assign now".

The objects displayed in the table of the "Online assignment" tab depend on the setting of the filter function. If only selected objects should be displayed, only objects of the corresponding context are displayed depending on the selection in the network view.

- PROFINET subnet: All connected devices and their PROFINET interfaces
- IO system All devices involved and their PROFINET interfaces
- Sync domain: All devices involved and their PROFINET interfaces
- Devices: The device and any existing PROFINET interfaces
- Other subnets or interfaces such as MPI or PROFIBUS are not displayed

If the display is set for all devices using the filter function, all devices are displayed that have a PROFINET interface, regardless of whether they are connected via a PROFINET subnet or are part of an IO system. Devices without a PROFINET interface, for example only with a DP or MPI interface, are not displayed.

General procedure

To assign PROFINET device names, you must first detect the IO devices available online. With this procedure, it matters whether the MAC addresses are known or unknown. This results in a general procedure in two steps:

1. Detecting the IO devices available online
2. Assigning configured PROFINET device names to the IO devices available online

Requirements

- You are in the network view.
- There is an online connection to the devices.

Procedure (step 1)

To detect IO devices available online from the I/O communication table, follow these steps:

1. Optional: Entered known MAC addresses in the "MAC address" column. After every valid entry, the check box under "Assign device" is selected for the relevant row.

   Note

   You can enter, insert or import the MAC address in different formats. The correct format is automatically entered in the cell. The following entries are supported and then converted to the required format:

   - "08:00:06:BA:1F:20"
   - "08 00 06 BA 1F 20"
   - "080006BA1F20"

   The formats used in the example are automatically converted to "08-00-06-BA-1F-20".

2. Click "Check devices" to start the check of the IO devices available online.
3. Set the PG/PC interface in the dialog window and click "Start".

Intermediate result

After the check, the result is displayed for every device in the table. Online data found is automatically entered in the table and the check box "Assign device" is set to "checked" in the rows in which a MAC address was entered or found online. The result of the check is shown as an icon in the "Status" column.

Status	Meaning
	Agreement: Matching device and compatible type. The PROFINET device name and the device type of the device found online matches the device in the project. No action is required.
	Ready for assignment: The device type of the device found online matches the device in the project. With a known MAC address the device is ready for the assignment of the PROFINET device name.
	Different device type: A matching device was found online, but the device type does not match the device type in the project. Please select a device with a matching device type online.

Status	Meaning
	Different device: No device could be found online that uniquely matches the device in the project. Please select a matching device manually.
	Device cannot be accessed: With a known MAC address the device cannot be accessed online. Please check the connection to the device.

Note

The icon "Ready for assignment" appears when a MAC address exists and matching device data was found, but no PROFINET device name was found online.

You can update the data of the detected devices again via their MAC addresses at any time. To do this, you specify the MAC address and the status of the device is displayed immediately without having to re-detect the device.

Procedure (step 2)

All PROFINET device names configured offline will be assigned to the devices available online in a bulk operation.

1. Click the "Assign now" button.

   Note

   The bulk operation cannot be reversed. A message to this effect appears in a dialog window.

2. Click "Start" in the dialog window to start the assignment of the PROFINET device names.

Result

The PROFINET device names configured offline will be assigned to the devices available online. This relates to devices in whose row the check box under "Assign device" is selected, that have a MAC address and have the status "Ready for assignment".

Importing and exporting data

Using the import and export button, you can import or export the data of the I/O communication table for the online assignment:

- When you export, the currently displayed data of the table is exported to a CSV file. Using the filter function of the table, you can select which data will be exported.
- When you import, the data of the CSV file is written to the table. If there are conflicts with values already existing in the table, you can decide whether the data should be overwritten or whether the import needs to be stopped.

Assign device name via memory card

Introduction

You can configure the device names of PROFINET IO devices offline. To do this, store a configured device name on a memory card and then insert the card into the appropriate IO device.

If an IO device has to be completely replaced due to a device defect, the IO controller automatically reconfigures the new device. Using the memory card, a device can be replaced without a programming device.

Requirements

- The programming device has a card reader for memory cards.
- The IO device must support the assignment of the device name via memory card.
- The station and its PROFINET IO system is configured.

Procedure

To store a device name on a memory card, follow these steps:

1. Insert the memory card into the card reader.
2. Select the IO device whose device name is to be assigned by the memory card.
3. Select the "Card reader > Save Device Name to Memory Card" command in the "Project" menu.
   If the memory card is not empty, a message will be issued informing you of this and you will have the option to delete the card.

Retentivity of IP address parameters and device names

The retentivity of IP address parameters (IP address, subnet mask, router setting) and device name depends on how the address is assigned.

The non-retentive, temporary assignment means:

- IP address parameters and device name remain valid for the following time period:
  - Until the next POWER OFF
  - Until the next memory reset
  - Until termination of the online connection (for example, after loading the program)
    After POWER OFF / POWER ON or a memory reset, the CPU can only be accessed via the MAC address.

If the IP address parameters are not retentive, communication based on the IP protocol can no longer take place after the above described events (for example, after POWER OFF/ POWER ON).

The assignment of a temporary IP address also deletes retentively saved IP address parameters.

Assigning IP address parameters and device name non-retentively

IP address parameters and device name are not retentive in the following cases:

- A temporary IP address that is not retentive is implicitly assigned with the "Accessible devices" function, if the device (e.g. CPU) does not yet have an IP address.
- The device is a "normal" IO controller (i.e., not an I-device), and it is specified in the user program ("IP_Conf" instruction) that the IP address parameters/device name are not to be retentive.

Assigning IP address parameters and device name retentively

IP address parameters and device name are retentive in the following cases:

- In the properties of the PROFINET interface, it is specified that the IP address parameters are set in the project ("Set IP address in the project" option).
- In the properties of the PROFINET interface, it is specified that the IP address is to be set on the device.
  - Once the configuration is loaded, the IP address parameters and the device name are assigned via STEP 7 or a setup tool such as PST (STEP 7: online and diagnostic function "Assign IP address"). The assigned IP address parameters are retentive.
  - The device is a "normal" IO controller (i.e., not an I-device), and it is specified in the user program ("IP_Conf" instruction) that the IP address parameters/device name are to be retentive.

Special features of the I-device

In the properties of the PROFINET interface of the I-device, it is specified that the IP address parameters are to be set on the device. The IP address parameters for the I-device are assigned by the higher-level IO controller.

- If prioritized startup is set, the IP address parameters are retentive.
- If no prioritized startup is set, the IP address parameters are not retentive.

Recommendation

If possible, use the "Set IP address in project" option and specify an appropriate IP address. In this case, the IP address is assigned retentively.

Resetting of retentive IP address parameters and device names

Retentive IP address parameters and device names are reset via the online and diagnostic function "Reset to factory settings".

Note

Consequences of reassignment of IP address parameters on top of existing IP parameters

- The temporary assignment of IP address parameters/device names results in a reset of any retentively saved IP address parameters/device names.
- With a fixed assignment of IP address parameters/device names, previously retentively-saved parameters are replaced by the newly assigned parameters.

Note

Reuse of devices

Execute the "Reset to factory settings" before you install a device with retentive IP address parameters and device name in another subnet or system or before you place it in storage.

Creating a PROFINET IO system

Introduction

To create a PROFINET IO system, you need to have a PROFINET IO controller and at least one PROFINET IO device. As soon as you connect an IO controller to an IO device via their PROFINET interfaces, a controller-device link is established.

IO controller and IO device

As IO controller, you can use the following devices with a PROFINET interface:

- CPU with a permanently integrated or plug-in PROFINET interface
- CP in conjunction with a CPU
- Interface module assigned to a CPU/FM
- Interface module with a PROFINET interface

As IO device, you can use head modules of the distributed I/O with PROFINET interface or CPUs as intelligent IO devices.

With some devices, you can switch between modes (such as "IO controller" and "IO device") under "Mode" in the Inspector window of the PROFINET interface. Select the option button with the required mode.

Requirements

- You are in the network view.
- The hardware catalog is open.

Procedure

To create a PROFINET IO system, for example with a CPU 1217C , follow these steps:

1. Select a CPU 1217C as a potential IO controller from the hardware catalog.
2. Drag the CPU onto the free area within the network view.
3. Right-click on the PROFINET interface of the CPU.
4. Select "Create IO system" from the shortcut menu.

A PROFINET IO system with a CPU 1217C as IO controller will be created as the only node.

If you connect the PROFINET interface of an IO device to the PROFINET interface of the IO controller, the IO slave will be added automatically to the IO system. If there is not yet a subnet between the IO controller and IO device, a new subnet is created between the IO controller and IO device.

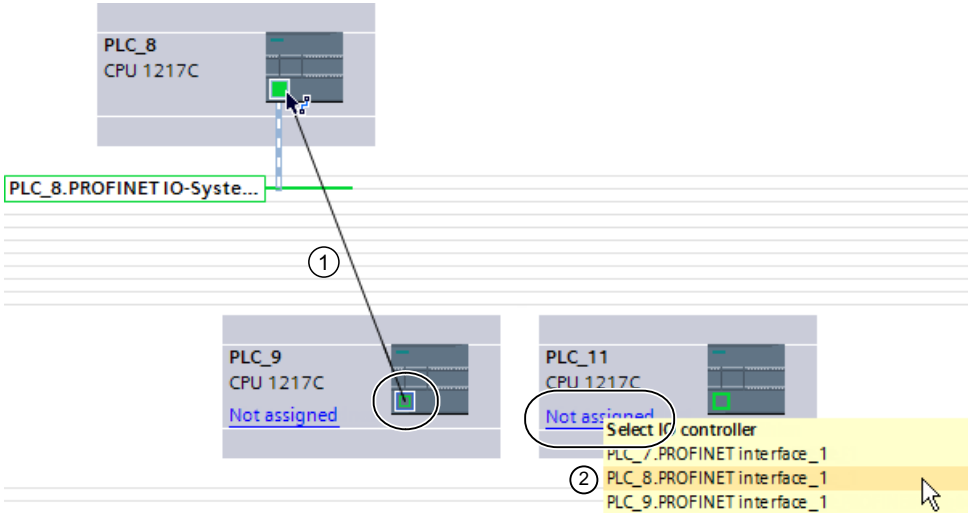
To include a CPU 1217C as an intelligent IO device in the DP master system, for example, follow the steps below:

1. Click on the PROFINET interface of the IO controller or IO device.
2. Hold down the mouse button and drag a connection from the selected PROFINET interface to the PROFINET interface of the desired communications partner.

or

1. Click on the hyperlink of the IO device CPU 1217C.
2. Select the required IO controller from the displayed list of possible IO controllers.

The intelligent IO device CPU 1217C is included in the PROFINET IO system with the CPU 1217C as IO controller.



① Drag between the PROFINET interfaces of IO device to IO controller.

② Click on the link of an unassigned IO device, and a selection of possible IO controller opens.

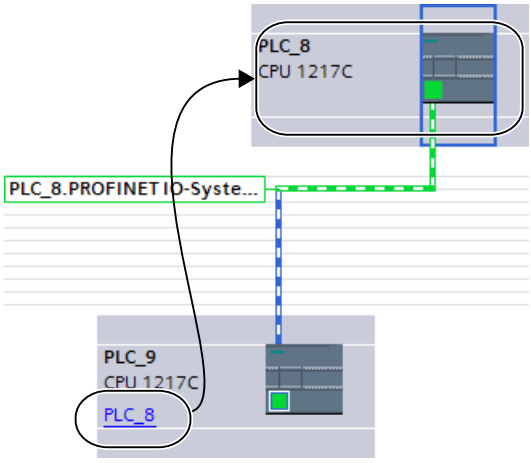
If required, adapt the properties of the Ethernet subnet or the IO controller (for example, IP address) under "Properties" in the inspector window.

Note

If there is only one IO controller on a subnet, you only have to connect the interface of the IO device to the subnet of this IO controller with drag-and-drop. The IO device is then connected to the subnet of the IO controller immediately, and the IO device is assigned to this IO controller.

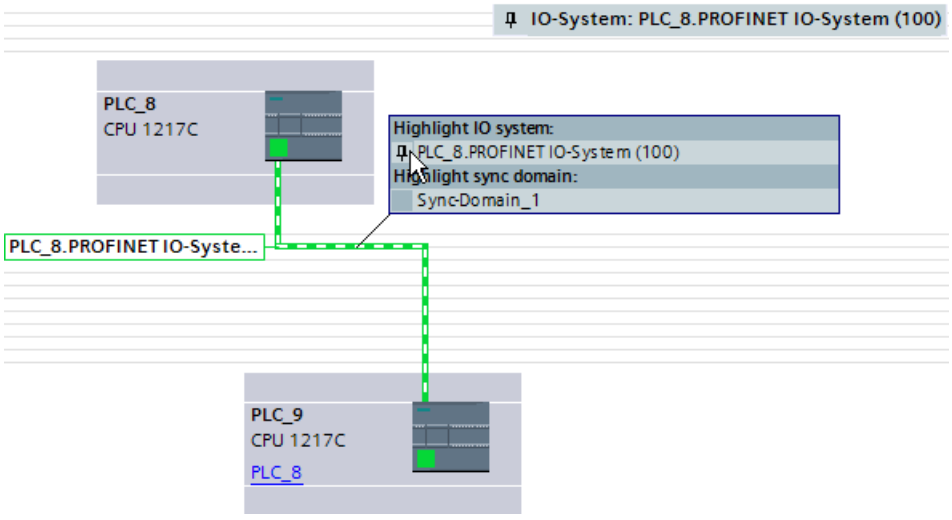
Display of the IO controller in the IO device

When an IO device is connected to an IO controller, the name of the IO controller is displayed on the IO device as a hyperlink. Click the hyperlink to select the assigned IO controller.



Highlighting the PROFINET IO system

When you have created a new PROFINET IO system, the PROFINET IO system will be highlighted. This enables you to identify quickly which devices belong to the PROFINET IO system. You can also highlight a PROFINET IO system yourself by moving the mouse pointer over a subnet. This will result in the names of the available PROFINET IO systems being displayed. If you click on one of the displayed PROFINET IO systems with the mouse, the corresponding PROFINET IO system is highlighted.



There are various ways of removing the highlighting from a PROFINET IO system:

- Highlight a different PROFINET IO system.
- Click on the pin with the name of the PROFINET IO system in the top right-hand corner of the network view.

Handling PROFINET IO systems

Using shortcut menu commands, you can delete PROFINET IO systems, create new ones or even connect the interface to another subnet from within the network view.

An existing PROFINET configuration can thereby be corrected in the network view.

Create new PROFINET IO system for IO controller

To create a new PROFINET IO for an IO controller, proceed as follows:

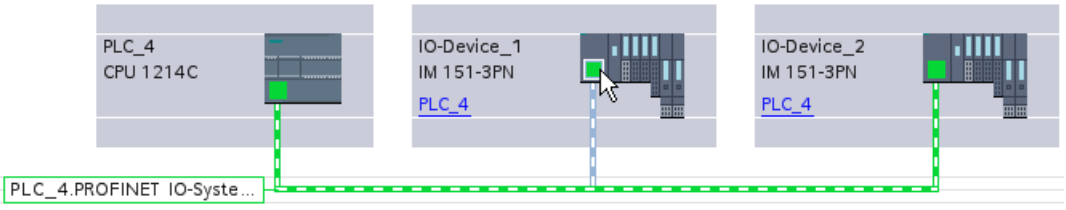
1. Make sure that no IO system is already assigned to the IO controller. If an IO system is already assigned to the IO controller, the "Assign IO system" shortcut menu command is disabled.
2. Select the PROFINET interface and then select the "Assign IO system" shortcut menu command.

A new PROFINET IO system is created at the IO controller and you can assign IO devices to this IO system.

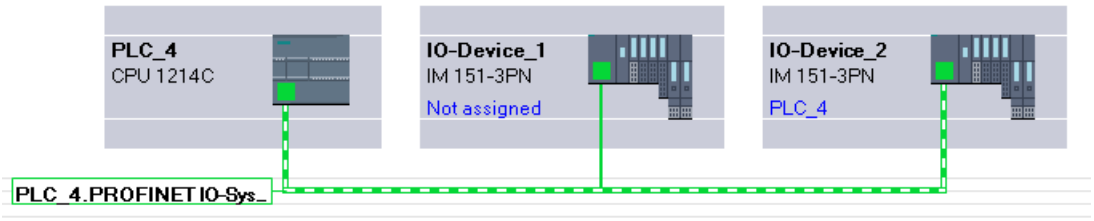
Disconnecting PROFINET IO devices from PROFINET IO system

To disconnect an already networked PROFINET IO device from its PROFINET IO system, follow these steps:

1. Click on the PROFINET interface of an IO device.



2. Select the "Disconnect from IO system" shortcut menu command.
   The IO device that was assigned to this IO system is then no longer assigned to it.



You can create a new IO systems and can assign each of the non-assigned IO devices to an IO controller.

Assign PROFINET IO devices to other IO controllers

Existing PROFINET IO systems can be easily reconfigured in the network view by selecting the IO device or its interface and the required command from the shortcut menu:

When selecting the interface:

1. Select the interface of an IO device and then select the shortcut menu. You have the following options here:
   - Assign a new subnet to the IO device or disconnect it from the existing subnet
   - Assign the IO device to a new IO controller
   - Assign a new IO system to the IO device or disconnect it from the existing subnet
2. To assign another IO controller to the IO device, select the "Assign to new IO controller" shortcut menu command.
   If there is no connection, a subnet is automatically created and the IO device is assigned to the IO system of the new IO controller.

When selecting the entire device:

1. Select the IO device and select the shortcut menu. You have the following options here:
   – Assign the IO device to a new IO controller
   – Disconnect the IO device from the existing subnet
2. To assign the IO device to another IO controller, select the "Assign new DP master/IO controller" shortcut menu command.
   If there is no connection, a subnet is automatically created and the IO device is assigned to the IO system of the new IO controller.

Tip: Quick configuration of IO systems

If the IO system has a lot of IO devices, assign all IO devices placed by drag-and-drop operation to an IO controller on one step.

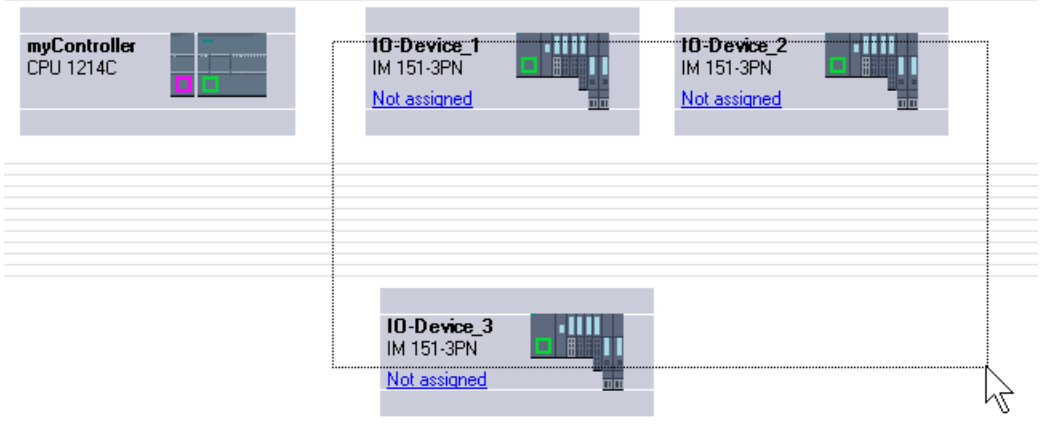
Requirements

IO controller and IO devices are placed in the network view.

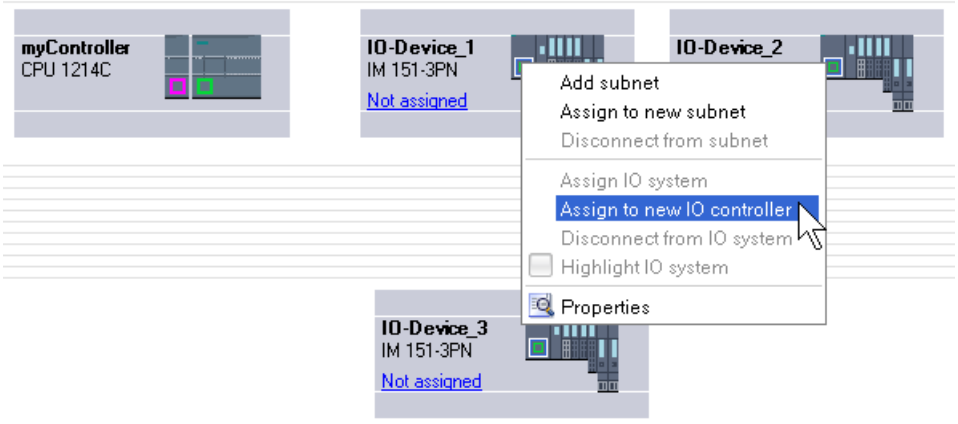
Assign IO devices to an IO system

To do this, follow these steps:

1. Select an appropriate zoom factor so that you can see as many IO devices as possible in the network view.
2. Arrange the IO devices in not more than of two rows.
3. Select all IO interfaces (not all devices) with the mouse cursor. This only works if you begin to drag the mouse cursor outside of the first IO device and release the mouse button at the last IO device (selection with the lasso).



4. Select the shortcut menu "Assign new IO controller" and select the corresponding IO interface of the IO controller in the subsequent dialog.



5. The IO devices are automatically networked with the IO controller and combine with it to form an IO system.

Note

When an IO system is highlighted, you can double-click on an IO device in the hardware catalog and thereby quickly add additional IO devices. Result: The IO device is automatically added to the highlighted IO system.

Interconnecting ports

If an IO device is assigned to an IO controller, this does not yet specify how the ports are connected to each other.

Although a port interconnection is not required to make use of the Ethernet/PROFINET functions, it does offer the following advantages:

- A target topology is specified with the port interconnection. Based on an online-offline comparison, it is possible to conduct a target-actual comparison with all devices that support this function.
- Only with IRT communication: If a port interconnection is configured, STEP 7 can determine the required bandwidth more precisely. As a rule, this leads to a higher performance.

Make sure that no invalid ring structures occur through the interconnection of ports.

Port interconnection is only advisable for devices that support the topology configuration.

Interconnecting ports in the Inspector window

To interconnect ports, follow these steps:

1. Select the Ethernet/PROFINET device or the Ethernet/PROFINET interface.
2. Navigate to the port property "Port interconnection".
   When the Ethernet/PROFINET interface is selected, you can find this setting in the Inspector window as follows: Properties > General > Advanced Options > Port [...] > Port Interconnection.
3. In the "Local port" section, you can find the settings at the local port. In the case of fiber-optic cable you can, for example, set the cable names here.
   In the "Partner port" section, click on the black triangle in the "Partner port" box to display and select the available partner ports.
4. If the port interconnection is a port interconnection with copper as medium and the devices support IRT communication, you can also set cable length and signal transit time.

If the Ethernet/PROFINET interface was disconnected, it is automatically networked by this action. In the properties of the subnet you can set whether this subnet should or should not be used for the networking.

Note

Interconnecting an electric with an optical port

If you want to interconnect an electrical and an optical port, you have to distinguish between RT and IRT communication:

- With RT communication, it is not necessary to configure a media converter.
- With IRT communication, you have to make the interconnection via a media converter.

Information on monitoring the partner port

After you have interconnected the two ports, you receive information on the monitoring of the partner port in a text field. The following displays are possible:

- Monitoring of the partner port is not possible.
- Partner port is being monitored.

It is not possible to monitor the partner port if you, for example, select a deactivated port as partner port. In this case it is not possible to monitor the target topology and the signal propagation time. A device replacement is then only possible with a Micro Memory Card.

See also

Overview (Page 712)

Setting the send clock

Requirements to change the send clock at the PROFINET device

The PROFINET device must be an IO controller as sync master or IRT (Isochronous Realtime) must not be configured in the IO system.

If the PROFINET device is not an IO controller as sync master, the following requirements must be met for the setting of the send clock:

- No device must be configured at the IO system as a sync slave or sync master.
- All devices at the IO system must be unsynchronized.

When IRT is configured with an IO controller as sync master, the send clock can be set at the device or in the sync domain.

Procedure

To set the send clock on the PROFINET device, follow these steps:

1. Select the PROFINET IO controller in the device or network view.
2. Change the value for the shortest possible update interval in the properties of the PROFINET interface under "PROFINET Interface > Advanced options > Real-time settings > IO communication > Send clock".

The send clock is valid for all PROFINET devices at the IO system. If the synchronization role is set to a value other than "Unsynchronized", you can also set the send clock in the sync domain, in other words, centrally at the PROFINET IO system.

1. Select the IO system using the PROFINET subnet.
2. Change the value for the send clock in the properties of the sync domain under "Domain management > Sync domains > Sync_domain_1 > Send clock".

Setting the update time

Update time

An IO device / IO controller in the PROFINET IO system is supplied with new data from the IO controller / IO device within this time period. The update time can be separately configured for each IO device and determines the time interval in which data is transmitted from the IO controller to the IO device (outputs) as well as data from the IO device to the IO controller (inputs).

STEP 7 calculates the update time automatically in the default setting for each IO device of the PROFINET IO system, taking into account the volume of data to be exchanged as well as the set send clock.

Setting the update time

If you do not want to have the update time calculated automatically, you can change the setting.

To change the update time in the Inspector window, proceed as follows:

1. Select the PROFINET interface of the IO device in the network or device view.
2. Specify the update time in the interface properties under "Advanced options > Real time settings > IO cycle". You can specify the update time in two ways:
   - Automatic calculation of the optimal update time.
   - Update time you can set yourself by selecting different values from a drop-down list.
3. If you want to keep the relationship between the send clock and the update time constant, enable the "Adapt update time when send clock changes" option.
   This option ensures that the update time is not set to less than the send clock.

You can also carry out this setting directly in the I/O communication table via the drop-down list in the "Update time mode" column.

Manual setting of the send clock may result in errors if the available bandwidth is not adequate or when other limits/configuration limits are exceeded (for example, too many nodes are configured).

Note

When you change the update time mode from "Can be set" to "Automatic", a manually entered update time is overwritten by the automatically calculated value.

No update time can be calculated

STEP 7 determines the sequence of the cyclic data exchange based on the configuration information (IO controller properties, IO device properties, number and type of the IO devices, consistency of the cyclic user data...). The cyclic data is packed in frames and sent/received successively at calculated time intervals.

The maximum number/size of the frames and the maximum number of intervals available must be sufficient to "accommodate" all the data. The resulting send/receive interval must also be supported by every PROFINET device.

If the limits relating to the amount of cyclic user data/number of frames or relating to the intervals available are exceeded, STEP 7 cannot calculate an update time.

If there is no common basis for the send/receive interval, calculating the update time is also not possible.

If there is a reason preventing the calculation of the update time, STEP 7 reports the cause when compiling the hardware configuration.

How to eliminate the problem:

- Reduce the number of IO devices
- Reduce the number of modules within the IO devices
- If you are using an IE/PB link: Reduce the number of DP slaves downstream from the IE/PB link
- Use a more powerful IO controller or IE/PB link

- Increase the send clock
  - With RT: in the properties of the IO controller
  - With IRT: in the properties of the sync domain
- Check the device properties of the IO devices ('MinDeviceInterval' and the possible scan rates) due to the common basis for the send/receive interval. Replace unsuitable IO devices. This device properties are stored in the GSD file of the IO device.
- With IRT configuration:
  - Check whether the ports of the sync master and sync slaves are interconnected.
  - Check the order of the IO devices: There can be **no** unsynchronized device connected between the sync master and sync slave
    (Example of bad configuration: Sync master --- unsynchronized device --- sync slave).
  - Check whether you have configured more than one sync master.
  - Check the bandwidth remaining for RT data. The bandwidth for RT data available for the transfer can be restricted by IRT communication on the same Ethernet subnet.
- When using I-devices:
  It may not be possible to use the set send clock together with the existing I-device configuration.
  - Configure the I-device without lower-level IO devices and activate the setting "Parameter assignment of PN interface by higher-level IO controller".
  - Change the send clock of the IO controller to an even value (... 0.250, 0.500, 1.000, ...).

Identification of the IO devices involved:

You can identify the IO devices involved for which no update time can be calculated in the "I/O communication" table of the PROFINET IO system in the network view. No entry is made in the "Update time" column for the IO device involved (entry "-").

Setting the watchdog time

Watchdog time

You can configure the watchdog time for PROFINET IO devices.

If the IO device is not supplied with input or output data (IO data) by the IO controller within the watchdog time, it switches to the safe state.

Do not enter the watchdog time directly, but as "Accepted number of update cycles when IO data is missing". This makes setting easier because the update time can be shorter or longer, depending on the power of the IO device or the setting.

The resulting watchdog time is automatically calculated from the "Accepted number of update cycles when IO data is missing".

Configuring the watchdog time

To specify the watchdog time, follow these steps:

1. Select the PROFINET interface of the IO device in the network or device view.
2. In the properties of the interface, navigate to "Advanced options > Realtime settings > IO cycle".
3. Select the required number of cycles from the drop-down list "Trigger watchdog after # cycles with missing IO data".

The watchdog time is subsequently calculated automatically based on the preset factor. It must not be more than 1.92 seconds.

Note

The default setting should only be changed in exceptional cases, for example, during the commissioning phase.

Calculated bandwidth for cyclic IO data

Calculated bandwidth for cyclic IO data

Adherence to the maximum available bandwidth for cyclic IO data is monitored by the system. The maximum bandwidth depends on the send clock cycle. If the send clock cycle is greater than or equal to 1 ms, the maximum bandwidth is 0.5 ms. If the send clock cycle is shorter, the maximum available bandwidth is also reduced.

The bandwidth actually required for cyclic IO data is determined by the system based on the number of configured IO devices and IO modules. Furthermore, the required bandwidth depends on the update time that is used.

In general, the calculated bandwidth increases in the following cases:

- There is a greater number of IO devices
- There is a greater number of IO modules
- The update times are shorter.

Maximum bandwidth for cyclic IO data depending on the send clock

The following table shows how the maximum available bandwidth for cyclic IO data reacts based on the send clock:

Send clock cycle	Maximum bandwidth for cyclic IO data
250 µs – 468.75 µs	<< 125 µs
500 µs – 968.75 µs	= send clock / 2
1 – 4 ms	= 500 µs

Setting port options

Setting the port options

Changing connection settings for the PROFINET IO port

By default, the settings for ports of a PROFINET interface are specified in such a way that problem-free data exchange is ensured in normal situations.

If necessary, you can change the settings. The setting options are described below.

Possible settings for transmission rate/duplex

Depending on the selected device, you can make the following settings for "Transmission rate/ duplex":

- Automatic setting
  Recommended default setting of the port. The transmission settings are automatically "negotiated" with the partner port. The "Enable autonegotiation" option is automatically selected by default.
- TP/ITP at x Mbps full duplex (half duplex)
  Setting of the transmission rate and the full duplex/half duplex mode. The effectiveness depends on the "Enable autonegotiation" setting:
  - Autonegotiation enabled
    You can use both cross cable and patch cable. The port is monitored with this setting.
  - Autonegotiation disabled
    Make sure that you use the correct cables (patch or cross cables). The port is also monitored with this setting.
- Deactivated
  Depending on the device type, the drop-down list box can contain the "Deactivated" option. With this option, you can prevent access via an unused port for security reasons, for example. With this setting, diagnostic events are not generated.

"Monitor" option

With the "Monitor" option, you enable/disable port diagnostics. You can only enable monitoring if the device supports this function.

Below is a list of what is monitored during operation by the device when monitoring is enabled - regardless of whether you have interconnected the port or not:

- The link status is monitored; in other words, diagnostics is generated when the link is down (e.g. with a wire break or a removed connector). With fiber-optic ports (FO), the link power margin is monitored so that a staged maintenance diagnostics is obtained.
- The media settings are monitored; in other words, a diagnostics message is generated when the local port setting (transmission rate/duplex) during operation does not match the setting of the partner port.
- If the port is interconnected: The cable length/signal delay time is monitored; in other words, a diagnostics message is generated if the locally detected cable length/signal delay time does not match the cable length/signal delay time detected by the partner.

Disabling monitoring

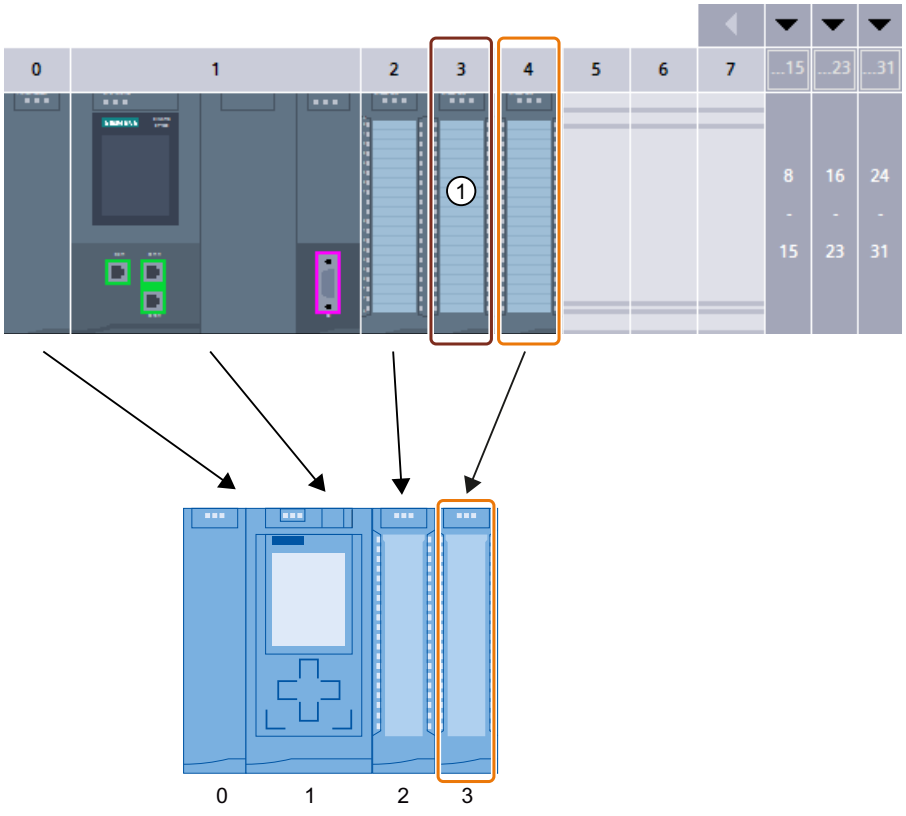
Monitoring does **not** make sense in the following cases:

- The option "Alternative partners" is enabled at the local port, for example, for a docking port of a docking station (tool changer). In this case, a link status change is a normal operational situation.
- The partner port belongs to a non-PROFINET device. Only ports of PROFINET devices transfer their port settings such as the transmission rate, duplex setting etc.
- The local port is connected to a media converter (passive network component with different signal delay times for Ethernet frames).
- At the local port, the option "End of topology discovery" is enabled in the boundaries settings.

Monitoring of the desired topology

With the "Monitoring" option, no monitoring is linked to the adherence to the desired topology. This means that whether the configured interface and port properties such as device name, port name and cable length match the current settings is not influenced by the "Monitoring" option.

Adherence to the desired topology is checked automatically when you interconnect the local port with a partner port; the partner port must not then be disabled. The port properties indicate whether the monitoring of the desired topology is performed or is impossible (port interconnection, partner port refer, to the figure below ①).



ConfDB

		Name	Data type	Start value	...	...	...	...	Comment
1		Static							
2		ConfigControl	Struct			☑	☑		
3		Block_length	USInt	9		☑	☑		4+number of slots
4		Block_ID	USInt	196		☑	☑		
5		Version	USInt	4		☑	☑		
6		Subversion	USInt	0		☑	☑		
7		slot_0	USInt	0		☑	☑		configured slot_0 / assigned "real" slot 0
8		slot_1	USInt	1		☑	☑		configured slot_1 / assigned "real" slot 1
9		slot_2	USInt	2		☑	☑		configured slot_2 / assigned "real" slot 2
10		slot_3	USInt	16#FF		☑	☑		configured slot_3 / not used ①
11		slot_4	USInt	3		☑	☑		configured slot_4 / assigned "real" slot 3

① Display of whether the device can check the actual properties of the partner port against the configured properties of the partner port. This adherence to the desired topology is checked automatically when you interconnect the local port with a partner port.

② Setting the monitoring to link down, media settings and cable length/signal delay time. At runtime, the local port compares its own settings with the settings of the partner port.

Option "Enable autonegotiation "

The autonegotiation setting can only be changed if a concrete medium (for example, TP 100 Mbps full duplex) is selected. Whether or not a concrete medium can be set depends on the properties of the module.

If autonegotiation is disabled, this causes the port to be permanently specified, as for example, is necessary for a prioritized startup of the IO device.

You must make sure the partner port has the same settings because, with this option, the operating parameters of the connected network are not detected and the data transmission rate and transmission mode can therefore not be optimally set.

Note

When a local port is interconnected, STEP 7 makes the setting for the partner port if the partner port supports the setting. If the partner port does not support the setting, an error message is generated.

Port options for using the GBIT PROFINET interface

The PROFINET interface (X3) of the CPU 1518-4 PN/DP supports a maximum transmission rate of 1000 Mbps (GBIT).

In order to achieve this transmission rate, the following requirements must be met:

- CPU Firmware Version V1.7 or higher.
- Devices on the same PROFINET subnet must also support the 1000 Mbps transmission rate.
- The network infrastructure (network cables and outlets) must be category CAT 5e or better.
- The port options of PROFINET interface X3 must be set as follows:
  - “Transmission rate/Duplex”: Automatic
  - “Autonegotiation”: Activated

See also

Wiring rules for disabled autonegotiation (Page 1796)

Boundaries at the port (Page 1797)

Wiring rules for disabled autonegotiation

Requirement

You have made the following settings for the port in question, for example, to accelerate the startup time of the IO device:

- Fixed transmission speed
- Autonegotiation incl. autocrossing disabled

This saves you the time required to negotiate the transmission rate during startup.

If you have disabled autonegotiation, you must observe the wiring rules.

Wiring rules for disabled autonegotiation

PROFINET devices have the following two types of ports:

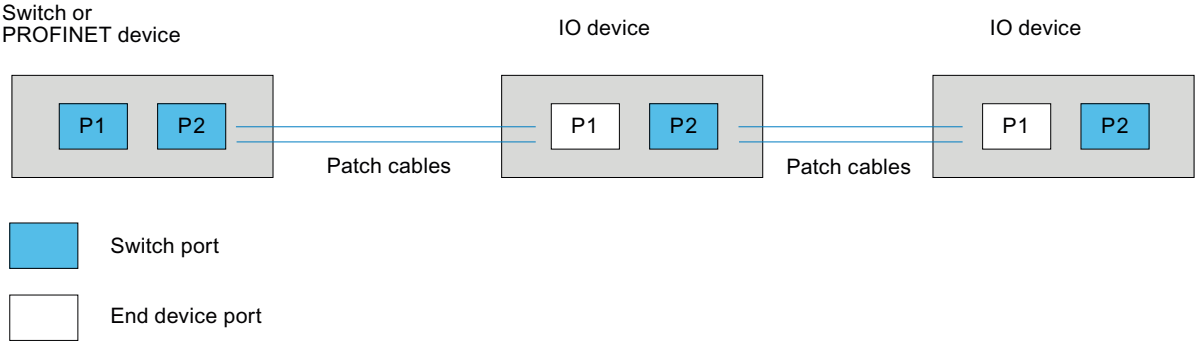
Type of port	PROFINET devices	Note
Switch port with crossed pin assignment	For IO devices: Port 2 For S7 CPUs with 2 ports: Ports 1 and 2	Crossed pin assignment means that the pin assignment for the ports for sending and receiving between the respective PROFINET devices is exchanged internally.
End device port with uncrossed pin assignment	For IO devices: Port 1 For S7 CPUs with one port: Port 1	-

Validity of the wiring rules

The cabling rules described in the following paragraph apply exclusively to the situation in which you have specified a fixed port setting.

Rules for cabling

You can connect several IO devices in a line using a patch cable (one-to-one wiring of both connectors). To do this, you connect port 2 (P2) of the IO device to port 1 (P1) of the next IO device. The following graphic gives an example with two IO devices.



Boundaries at the port

Requirement

The respective device must support boundaries settings in order to work with boundaries. If the device for PROFINET does not support boundary settings, the corresponding parameters are disabled, such as in the CPU 1215C V3, for example.

Enable boundaries

"Boundaries" are limits for transmission of certain Ethernet frames. The following boundaries can be set at a port:

- "End of detection of accessible devices"
  DCP frames for detection of accessible devices are not forwarded. Devices located behind this port are no longer displayed in the project tree under "Accessible devices". Devices located behind this port can no longer be reached by the CPU.
  The device must have more than one port for this function.
- "End of topology discovery"
  LLDP frames (Link Layer Discovery Protocol) for topology detection are not forwarded.
- "End of sync domain"
  Sync frames transmitted for synchronization of devices within a sync domain are not forwarded.
  For example, if you operate a PROFINET device with more than two ports in a ring, you should prevent the sync frames from being fed into the ring by setting a sync boundary (at the ports not inside the ring).
  The device must have more than one port for this function.
  Additional example: If you want to use several sync domains, configure a sync domain boundary for the port connected to a PROFINET device of the other sync domain.

Restrictions

The following restrictions must be observed:

- The individual check boxes can only be used if the port supports the function in question.
- If a partner port has been defined for the port, the following check boxes cannot be used:
  - "End of detection of accessible devices"
  - "End of topology discovery"

Enabling device replacement without exchangeable medium

Replacing an IO device without exchangeable medium

It is often necessary to replace IO devices in automation systems. The IO devices are generally assigned a device name by either inserting an exchangeable medium or via the programming device. The IO controller uses this device name to identify the IO device.

Subject to certain conditions, IO devices can also receive their device names without the insertion of an exchangeable medium (e.g., memory card) or without a programming device. For this purpose, the IO controller uses Ethernet mechanisms (LLDP protocol; Link Layer Discovery Protocol) to analyze the relationships between the individual IO devices and the IO controller. From these relationships, the IO controller detects which IO device was replaced and assigns the configured device name to it.

Requirements

- A port interconnection is already configured.
- The affected IO devices in the automation system must support device replacement without exchangeable medium (LLDP protocol).

Note

Use only new IO devices as replacement devices, or restore IO devices already configured to the delivery state prior to commissioning.

For S7-1500 CPUs with firmware version V1.5 or higher, it is not necessary to reset IO devices with an existing parameter assignment to delivery state. Requirement: The "Permit overwriting of device names of all assigned IO devices" option is selected for the IO controller ("Advanced option > Interface options" area of the properties of the PROFINET interface).

Procedure

In order to enable the replacement of an IO device without exchangeable medium, proceed as follows:

1. In the device or network view, select the PROFINET interface of the corresponding IO controller.
2. In the interface properties under "Advanced settings > Interface options", select the "Support device replacement without exchangeable medium" check box.

The option "Support device replacement without exchangeable medium" also permits automatic commissioning, which means you can commission the IO system with the IO devices without assigning their device names in advance.

IO devices without corresponding functionality

When individual IO devices in the automation system do not support device replacement without exchangeable medium, a corresponding alarm is output for the IO device. It is possible to use the functionality nevertheless. However, all of the following conditions have to be fulfilled:

- The function "Device replacement without exchangeable medium" is activated in the higher-level IO controller.
- The IO device supports the LLDP protocol.
- At least one "neighbor" of the IO device supports
  - The function "Device replacement without exchangeable medium" as an IO controller or
  - The LLDP protocol as an IO device.

When you configure the environment of the IO device so that the conditions described above are met for the "Device replacement without exchangeable medium", the usage of the function is possible even if the IO device does not support the functionality directly.

See also

Assigning the device name and IP address (Page 1769)

Components with the the device replacement without exchangeable medium function (http://support.automation.siemens.com/WW/view/en/36752540)

Connect the DP slave via the IE/PB Link to a PROFINET IO system

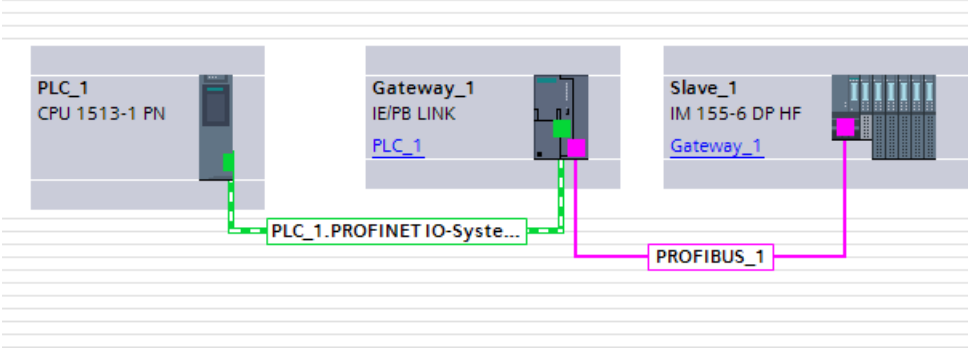
Requirements

- STEP 7 as of V12
- S7-1500 CPU as of firmware version 1.7
- ET 200SP CPU as of firmware version 1.7
- S7-1500 software controller
- S7-300/400 CPU

Procedure for connecting a DP slave via an IE/PB Link

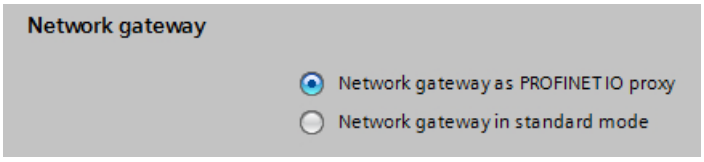
To connect a DP slave to a PROFINET IO system via an IE/PB Link in STEP 7, follow these steps:

1. Drag a PROFINET CPU, e.g. 1513-1 PN, from the hardware catalog to the network view of STEP 7.
2. Drag an IE/PB Link PN IO from the hardware catalog into the network view of STEP 7. The IE/PB Link PN IO is located under Network components > Gateways > IE/PB Link PN IO.
3. Assign the IE/PB Link PN IO to the CPU.
4. Drag a PROFIBUS interface module, e.g. IM155-6 DP HF, from the hardware catalog to the network view.
5. Assign the interface module to the IE/PB Link.

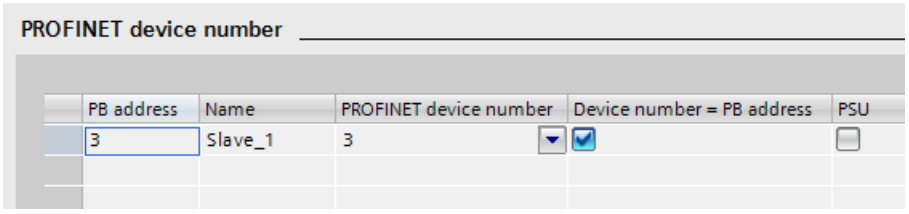


6. Select the IE/PB Link PN IO in the network view of STEP 7.

7. In the Inspector window, go to the "Gateway" area and select the "Network gateway as PROFINET IO proxy" option.



8. In the PROFINET device number area, you can assign a PROFINET device number for the DP slave.
   If you have selected the "Device number = PB address" check box (default), STEP 7 automatically assigns the device number according to the PROFIBUS address of the slave. In addition, you no longer need to update the device number if the PROFIBUS address changes.



See also

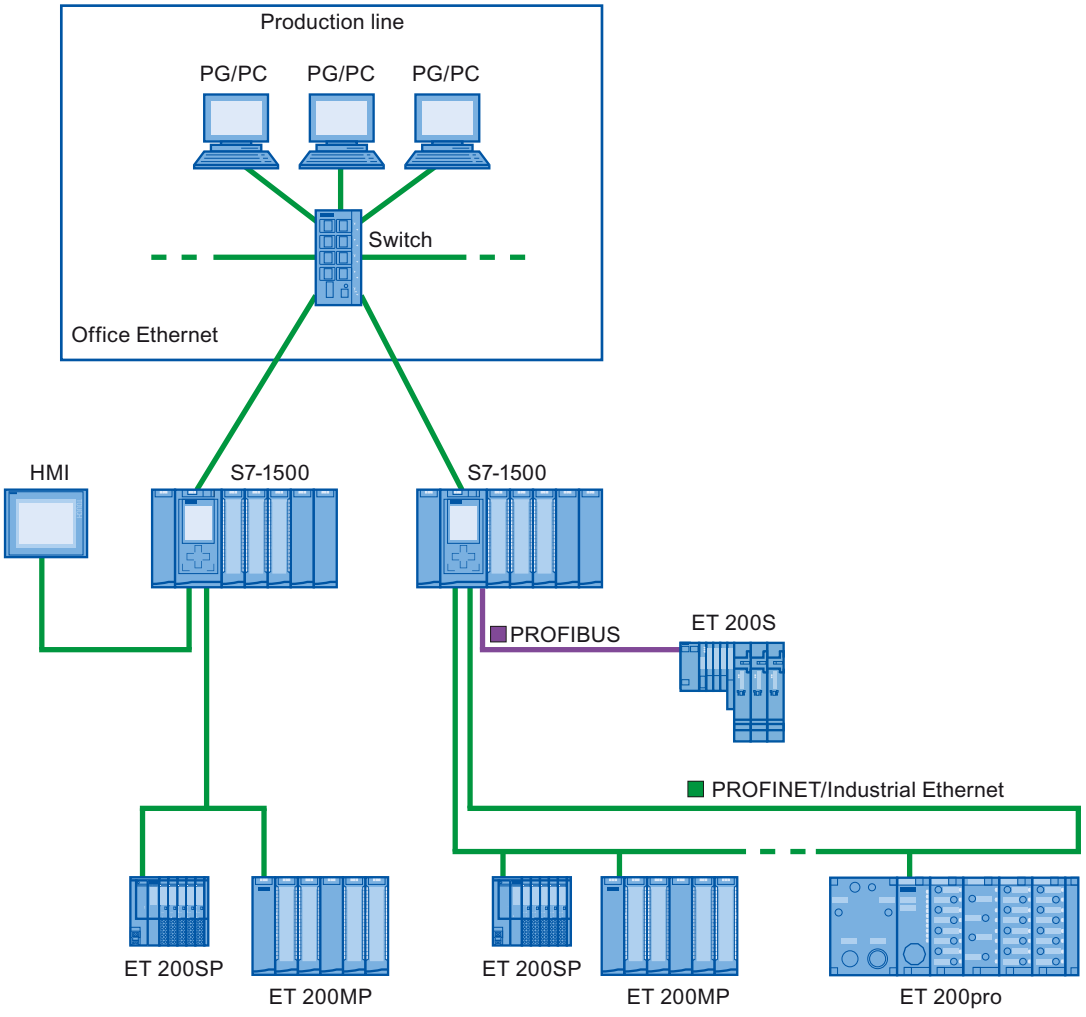
Configuration using IE/PB link PN IO (Page 1765)

Setup recommendations for optimizing PROFINET

Optimizing PROFINET with RT

PROFINET provides you with high-performance communication on all levels.

The figure below shows an example of an optimized PROFINET topology.



When setting up your PN network topology, take care that the various automation applications are distributed among separate network branches so that a sufficient bandwidth reserve will be available for future expansions.

- When you integrate standard Ethernet devices into the network topology or use standard Ethernet communication, take into account the network load caused by standard Ethernet and adapt the network topology as appropriate (max. bandwidth 100 Mbps).
- For communication with higher-level networks with a high data volume, use the most direct paths possible to the higher-level network infrastructure.

Also observe the Installation Guideline of the PROFIBUS User Organization.

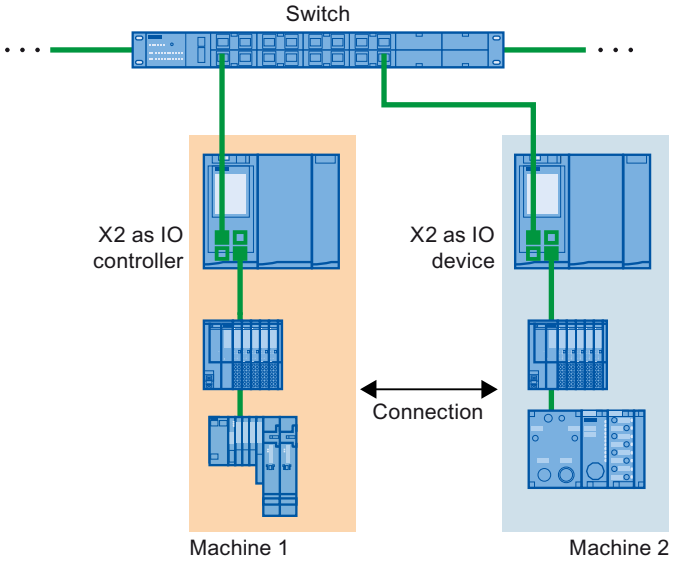
Setting up PROFINET with IRT

Keep in mind the following rules for setting up and operating a PROFINET IO system in IRT mode. These rules will ensure best possible operation of your PROFINET IO system.

- When using IRT, you must configure the topology. This will enable exact calculation of the update time, bandwidth, and other parameters.
- If you would like to use multiple sync domains, configure a sync boundary for the port which is currently connected to the PROFINET device of another sync domain.
- In a sync domain, you can only configure one sync master at a time.
- A PROFINET IO system may only be part of a single sync domain.
- When you configure PROFINET devices in a sync domain and want to synchronize with IRT, the PROFINET devices concerned must support IRT communication.
- If possible, use the same PROFINET device as the PROFINET IO controller and sync master.
- If only some of the PROFINET devices in a PROFINET IO system are synchronized, please keep in mind the following: Assign PROFINET devices that are not participating in the IRT communication to the RT class "RT" and the synchronization role "unsynchronized" in the sync domain.

Applications for CPU with multiple PROFINET IO interfaces

- Connecting machines: Your configuration contains machines located on separate IO lines. You can operate real-time communication between the CPUs over X2 PROFINET IO interfaces. Use the I-device or shared I-device function for this purpose.
  The figure below shows an example configuration for 2 machines connected over the X2 interface with an I-device relationship.



- Distribution by automation tasks:
  - For automation tasks with high performance and deterministic requirements, use PROFINET with IRT over the X1 interface.
  - For other tasks that you can implement with RT, use the X2 interface.

If you use interface X2 as the PROFINET IO interface for one of the following CPUs, this can affect performance:

- CPU 1515(F)-2 PN
- CPU 1515T-2 PN
- CPU 1516(F)-3 PN/DP
- CPU 1516(F)pro-2 PN

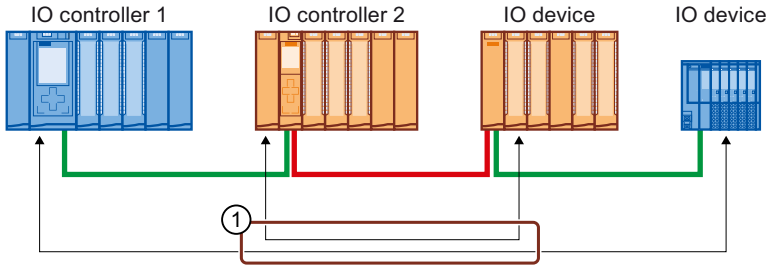
You can find additional information in the Cycle and Response Times function manual.

Topological overlap of IO systems in multi-controller applications

In a configuration with multiple IO controllers, shared paths are subject to the combined network loads of all connected PROFINET IO systems.

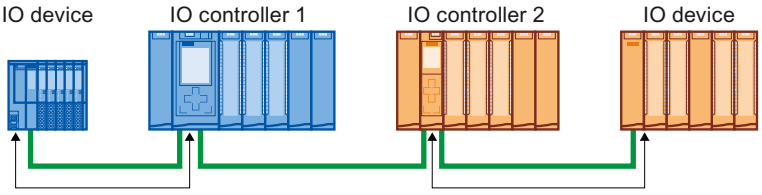
To avoid high communication loads in multi-controller applications, observe the following recommendations:

- Avoid paths that are shared by multiple IO systems.
  The figure below shows a configuration with two PROFINET IO systems that use the same paths.



① Shared path

In the figure below, the two PROFINET IO systems do not use shared paths.



- If separation is not possible: Increase the update time for the affected IO devices.

Installation guidelines of the PROFIBUS user organization

The installation guidelines can be found on the Internet.

Limitation of the data infeed into the network

Limiting the data infeed into the network on PROFINET interfaces

The "Limit data infeed into the network" function limits the network load of standard Ethernet communication which is fed into the network by the interface to a maximum value. This does not apply to cyclic real-time communication (RT/IRT).

In a PROFINET IO system, critical network loads can occur quickly with standard Ethernet communication. All devices in a PROFINET IO system should support the "limitation of the data infeed into the network".

If you use devices that feed a lot of "standard Ethernet communication" into the PROFINET IO system, you should adapt the topology, if necessary.

Depending on the interface, you can activate or deactivate the "Limit data infeed into the network" function. If you use the X1 interface of an S7-1500 CPU as the IO controller or I-device, the "Limit data infeed into the network" function is always enabled. Example: If you are not using the X1 interface of an S7-1500 CPU as IO controller or I-device, you can enable or disable the function.

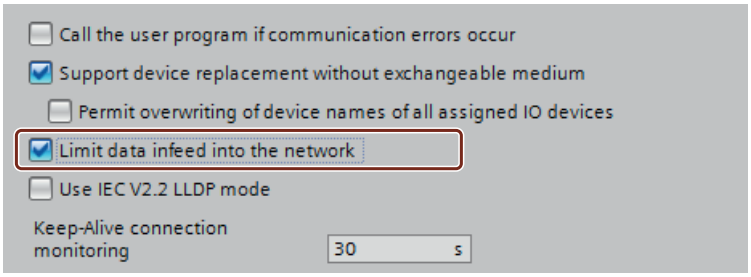
Uses of the limitation of the data infeed to the network

- Division of the bandwidth for standard Ethernet communication between devices:
  In PROFINET networks cyclic real-time communication and standard Ethernet communication share the same network. This means that only a limited bandwidth remains for standard Ethernet communication. The limitation of the data infeed ensures that the remaining bandwidth for standard Ethernet communication is not used just by one device, but is divided between several devices.
- Smoothing peaks in the data infeed:
  The limitation of the data infeed smooths peak loads of standard Ethernet communication (for example, from Open User Communication, access by the Web server).
- Limiting problems in the data infeed:
  If applications in a device generate too much data, this data is not forwarded to the PROFINET network. Negative effects (for example data loss, communication breakdown) remain limited to the device and its communications partner. Other nodes are not disturbed.

Setting limitation of the data infeed into the network for a CPU

To set the limitation of the data infeed into the network ("infeed load limitation"), follow these steps:

1. In the network view of STEP 7, select the interface of the CPU.
2. In the Inspector window, go to "Properties" > "General" > "Advanced options" > "Interface options".
3. Select or clear the "Limit data infeed into the network" check box.



Using GSD files

GSD files for IO devices

Basic information on GSD files of IO devices

The properties of PROFINET IO devices are not stored in a keyword-based text file (as for PROFIBUS DP slaves), but in an XML file whose structure and rules are determined by a GSDML schema.

The language used to describe the GSD files is GSDML (Generic Station Description Markup Language). It is defined by the GSDML schema. A GSDML schema contains validation rules

that allow it, for example, to check the syntax of a GSD file. GSDML schemas (as schema files) are acquired by IO device manufacturers from PROFIBUS International.

Functional enhancements in the area of PROFINET IO will have an effect on the GSDML specification and the corresponding schema. A new version of the specification and of the schema is created by the functional enhancement.

Names of GSD files for IO devices

One possible example of a GSD file name for IO devices is:

"GSDML-V1.0-Siemens-ET200S-20030616.xml"

Name component	Explanation
GSDML	String at the start of each GSD file for IO devices
V1.0	Version of the GSDML schema
Siemens	Manufacturer
ET200S	Name of the device
20030616	Version code (date)
.xml	File extension

Versioning of GSD files for IO devices

The version information for GSD files for PROFINET IO devices consists of two parts:

- Version of the GSDML schema: This determines the language scope used by a GSD file.
- Version number in form of a date: The version number of GSD files is incremented, for example, after elimination of an error or introduction of a functional enhancement.

Functional enhancements may result in a new version of the GSDML schema. A new version of a GSDML schema might only be supported with restrictions.

Installing the GSD file

Introduction

A GSD file (general station description file) contains all properties of an IO device. If you want to configure an IO device that is not available in the hardware catalog, you must install the GSD file provided by the manufacturer. IO devices installed via GSD files are displayed in the hardware catalog and can then be selected and configured.

Requirement

- The hardware and network editor is closed.
- You have access to the required GSD files in a directory on the hard disk.

Procedure

To install a GSD file, follow these steps:

1. In the "Options" menu, select the "Manage general station description files (GSD)" command.
2. In the "Installed GSDs" tab, select the directory in which the GSD files are stored.
3. Choose one or more files from the list of displayed GSD files.
4. Click on the "Install" button. The selected GSD files are now being installed.
5. To create a log file for the installation, click on the "Save log file" button.
   Any problems during the installation can be tracked down using the log file.
6. Click "Close". You will be notified that the IO devices from the installed GSD files are entered into the hardware catalog. This process can take a few seconds.

By installing a GSD file you have added the IO device described in the file to the hardware catalog.

Result

You will find the new IO devices installed by means of GSD files in the hardware catalog under "Additional field devices > PROFINET IO".

GSD files are always saved together with the project, which means all information relevant for display of the device (including symbols) is also available in the saved project.

See also

Overview of hardware and network editor (Page 525)

Deleting GSD file

Introduction

You can delete installed DP slaves using GSD files. These DP slaves are then no longer displayed in the hardware catalog.

Requirement

- The hardware and network editor is closed.
- You will find the new IO devices installed by means of GSD files in the hardware catalog under "Additional field devices > PROFINET".

Procedure

To delete a GSD file, follow these steps:

1. In the "Options" menu, select the "Manage general station description files (GSD)" command.
2. In the "Installed GSDs" tab, select the directory in which the GSD file is stored.
3. Select the file that is to be deleted from the list of displayed GSD files.
4. Click the "Delete" button.

The selected GSD file was deleted and the DP slave is no longer located in the hardware catalog.

Finding unused GSD files

Introduction

When you add a GSD-based hardware component to your hardware configuration, different GSD files are added to the project data. If the GSD-based hardware component is once again deleted from the configuration, the GSD files remain in the data inventory of the project. The GSD files are no longer needed in the project if the corresponding GSD-based hardware component is not used in the hardware configuration again. You can find files that are no longer needed in the project with a search function and delete them.

Requirement

- A project must be open.
- The project must not be online.
- The project must not be write-protected.

Procedure

To find and delete unused GSD files, follow these steps:

1. In the "Options" menu, select the "Manage general station description files (GSD)" command.
2. Click on the "Find unused GSDs" button in the "GSDs in the project" tab.

   Note

   Unused GSD files are only available if at least one GSD-based hardware component was added to the configuration and then removed from the configuration. Unused GSD files in the project libraries are included in the search. GSD-based hardware components that are still being used in the configuration are not considered to be unused.

   Installed GSD-based hardware components that are included in the hardware catalog but have not been used in the hardware configuration of the project yet, could not have stored unused GSD data in the project.

3. If GSD files that are no longer needed were found in the project data, you can now delete them. To do so, click the "Delete" button.

GSD files that are no longer needed are deleted from the project data. The GSD-based hardware component is still available in the hardware catalog and, if necessary, can be added to the configuration once again. The necessary GSD data are once again copied to the project data in this case.

Note

If you delete a GSD-based hardware component with "Options > Manage general station description files (GSD) > Installed GSDs", it is deleted from the hardware catalog. However, unused GSD files of this hardware component can still be a part of the project data and can be deleted with the function for finding the unused GSD files.

NOTICE
Actions that cannot be undone
When you delete GSD files that are not being used as an action, the action stack for actions that can be undone is deleted. This means the action of deleting unused GSD files as well as all previous actions can no longer be undone.

See also

Basics of undoing and redoing actions (Page 316)

Changing the revision of a GSD file

Changing the revision of a GSD file

You can change the revision of a GSD file for an IO device:

- Only for the current IO device
- All suitable IO devices within the IO system
- All suitable IO devices within the complete project

First, all existing GSD files for the current IO device are shown. The only difference between the GSD files shown is their revision status. The currently used GSD file is highlighted.

Requirement

- The I/O data is the same for all IO devices whose revision is to be changed.
- The article number has not changed.
- The number of submodules is identical.
- The configuration data has not changed.
- There must be no module or submodule in a slot that is invalid after the new GSD file has been created.

Procedure

To change the revision of one or more IO devices, proceed as follows:

1. Select the IO device whose GSD file revision is to be changed.
2. Click on the "Change revision" button under "General> Catalog information" in the properties of the IO device.
   The "Change revision" dialog box opens.
3. Select the GSD revision you want to use in the "Available revisions" table.
4. Under "Use selected revision for", select the devices whose version are to be changed:
   - Only for the current IO device
   - For all suitable IO devices in the IO system
   - For all suitable IO devices in the project
5. Click the "Apply" button.

10.1.4.8 Bus coupling with PN/PN coupler

Application and function

Application

The PN/PN coupler is used to link two Ethernet subnets with one another and to exchange data. That way use data about input or output address areas or datasets can be used. The maximum size of the transferable input and output data is 1024 bytes. The division into input and output data is preferable, so that e.g. 800 byte input data and 200 byte output data can be configured.

As a device, the PN/PN coupler has two PROFINET interfaces, each of which is linked to one subnet.

In the configuration, two IO Devices are produced from this one PN/PN coupler which means that there is one IO Device for each station with its own subnet. The other part of PN/PN coupler in each case is known as the bus node. Once configuring is complete, the two parts are joined.

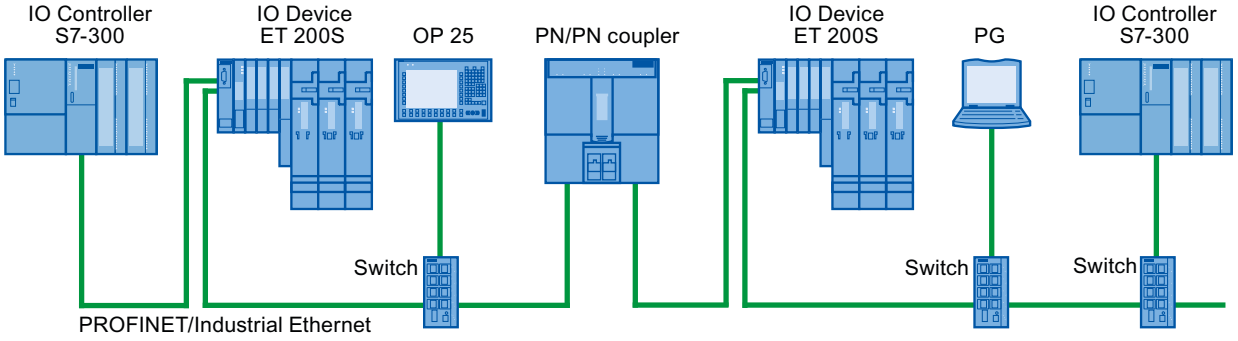


Figure 10-19 Coupling two PROFINET IO subnets with one PN/PN coupler

Additional information

For additional information on "PN/PN couplers", refer to Service & Support on the Internet (http://support.automation.siemens.com/WW/view/en/44319532).

Linking Ethernet subnets

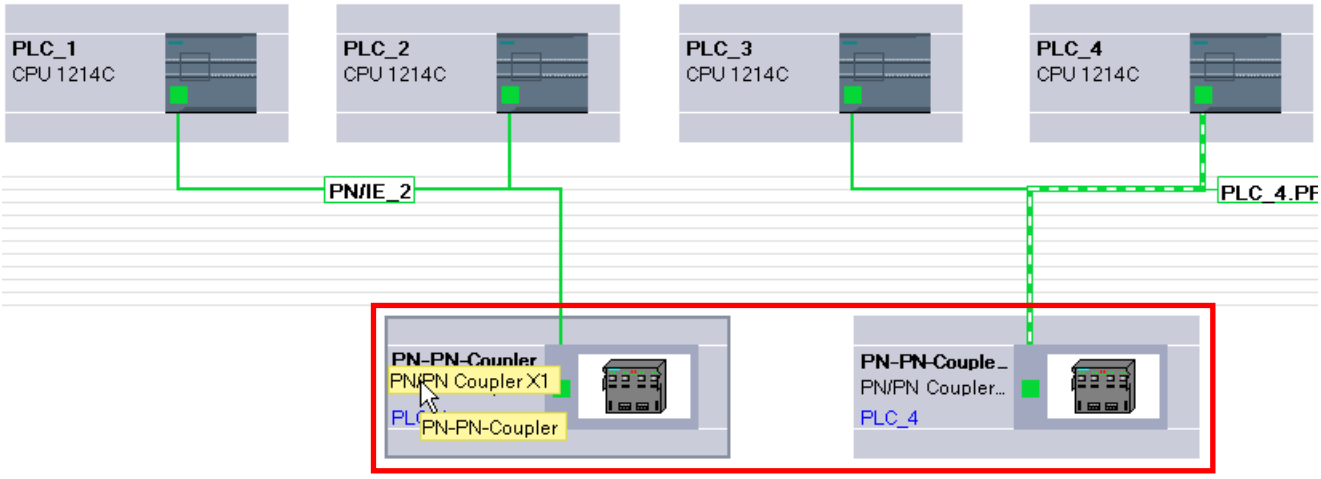
Linking Ethernet subnets with a PN/PN coupler

You can link Ethernet subnets with the standard device PN/PN coupler.

To link Ethernet subnets, follow these steps:

1. Create your Ethernet subnets.
2. Select the standard field devices in the hardware catalog. Find the PN/PN coupler as head module in the "PROFINET IO" folder.

3. In the network view, drag the two components X1 and X2 to the required version of the PN/PN coupler per drag-and-drop operation. The components form a device, but are shown separately to make handling easier.
4. Connect the Ethernet interface of the PN/PN coupler X1 to the first Ethernet subnet.
5. Connect the Ethernet interface of the PN/PN coupler X2 to the second Ethernet subnet. The Ethernet subnets are now linked through the two components of the PN/PN coupler.



10.1.4.9 Integrating external tools

Integrating S7-external tools

Introduction

Tools external to STEP 7 ("Device Tools") with a special call interface (Tool Calling Interface) can be used to configure distributed devices. Such devices are also referred to as "TCI capable".

The performance range of these tools exceeds the possibilities provided within GSD configuration, for example, they can provide expanded graphical input options.

Distributed devices can be as follows:

- PROFIBUS DP slaves
- Modules within a DP slave
- PROFINET IO devices
- Modules within an IO device

Note

Warranty and liability

Siemens accepts no liability for third-party software (device tools) called with the TCI (Tool Calling Interface) or for proper interaction with the associated devices.

Requirement

The call interface of the tool complies with the TCI specification. Parameters and commands are forwarded to the distributed device via this call interface.

Such tools have to be installed using a setup provided by the manufacturer. The "S7-PCT" (Port Configuration Tool) device tool for IO-Link master modules and IO-Link devices is an exception; this is supplied with STEP 7. Special note: After the installation, the tool is not shown in the list of installed software or in the list of software products in the project.

The GSD file of the distributed device that is to be configured with the Device Tool must be installed.

Starting the device tool

The command for starting the device tool is available in the shortcut menu of the TCI-capable device in the shortcut menu of the graphical and tabular device view: "Start device tool".

See also

Starting the SIMATIC S7-PCT (Page 1814)

Starting the SIMATIC S7-PCT

Introduction

The "S7-PCT" (Port Configuration Tool) device tool is installed with STEP 7.

The tool is used to assign parameters to the ports of IO-Link modules such as 4SI IO-Link (S7-1200, ET 200S) or 4IOL+8DI+4DO (ET 200eco PN).

Requirement

You have configured the corresponding CPU, the DP slave or the IO device with an IO-Link module.

Procedure

To start via the graphical device view, follow these steps:

1. Select the IO-Link module in the device view.
2. Select "Start device tool" from the shortcut menu.

OR, to start with the tabular device view, follow these steps:

1. Select the IO-Link module in the device view.
2. Arrange the areas in the work area in such a way that the tabular device overview is visible (it is located between the device view and the inspector window).
3. Select the row with the IO-Link module in the device overview.
4. Select "Start device tool" from the shortcut menu.

Result

The tool starts and you can configure the ports.

See also

Integrating S7-external tools (Page 1813)

10.1.4.10 Loading a configuration

Introduction to loading a configuration

In order to commission a device, identical configurations must be stored on the programming device/PC and on the connected devices. To synchronize the configurations on the programming device/PC and the connected devices, you load a configuration. Configuration data can be loaded in two directions:

- From the programming device/PC to a device
- From the device to a PG/PC

See also

Uploading project data from a device (Page 354)

General information on loading (Page 348)

Downloading a configuration to a device (Page 1815)

Downloading project data to a device (Page 350)

General information on loading to PG/PC (Page 1816)

Special features during startup (Page 962)

Downloading a configuration to a device

Loading the hardware configuration

After you have inserted a new device in the project and configured it or if you have modified an existing hardware configuration, the next step is to load the current configuration to the device. This ensures that the same configuration is set on the programming device/PC and on the module that is physically present. Use the "Online > Download to device (Page 350)" menu command for this.

In the first download, the complete hardware configuration is downloaded. In subsequent downloads, only changes to the hardware configuration are downloaded.

You have the following options for loading the hardware configuration:

- Loading in the device or network view
- Loading in the project tree
- Loading to an accessible device

WARNING
Perform load operation only in STOP mode Following loading, the machine or process may behave unexpectedly if the parameter assignment is incorrect. A CPU must be set to STOP mode for the load operation to rule out possible damage to equipment or personal injury.

Special considerations for loading isochronous applications

Isochronous applications consist of a hardware configuration part and a software part.

Example: If you change the number of an IO system, the delay time, or the assignment of a process image partition of the isochronous I/O in the hardware configuration, this affects the parameters of the isochronous mode interrupt OB and thus also the software part.

With isochronous applications, you should always load the complete project (hardware and software). With partial loading (loading hardware and software separately at different times), inconsistencies can arise that, for example, can prevent CPU startup or isochronous operation of the application.

See also

General information on loading (Page 348)

Loading a configuration to PG/PC

General information on loading to PG/PC

Introduction

If you bring your PG/PC to a plant and the STEP 7 project used to create the configuration of this plant is not available, load the configuration to a new project on your PG/PC, for example. Use the "Online > Loading the device as new station (hardware and software) (Page 354)" menu command for this.

The list of accessible devices in the project tree is always used when loading a device to your programming device. You can select multiple devices and load them to the project at the same time.

Requirements

- The hardware configuration in the device must be created in TIA Portal, V12 or higher. A hardware configuration present in the device that was created with an older version cannot be loaded and must be upgraded.
- Modules present in the device from GSD (ML), HSPs, or service packs must be installed in TIA Portal on the PG/PC.

- A project must be open. This project can be a new (empty) project or an existing project.
- The opened project is in offline mode.

Scope of load operation

The following list shows an overview of the loadable components of a configuration:

- The device (e.g., a CPU) with all I/O modules and all parameter settings
- PROFIBUS master systems and all PROFIBUS-relevant settings
- PROFINET IO systems and all PROFINET-relevant settings
- I devices and I slaves
- Settings for direct data exchange

After loading a CPU, all other modules within the address area of the CPU are also loaded.

The following connections are also loaded when the configuration is loaded:

- S7 connections (including routed connections) in combined PB/IE networks, including via IE-CP or PB-CM interfaces. S7 connections are automatically accepted as configured at one end when a device configuration is loaded, even if the S7 connection was configured at both ends in the original project. When both connection partners are loaded, the connection is joined together again during the next compilation.
- TCP connections via CPU-internal Ethernet interface, UDP/ISOonTCP connections, and TCP, UDP, ISO, and ISOonTCP connections via IE-CP interface
- Connections via the OUC connection parameter assignment for projects from STEP 7 V13 and higher

Note

The hardware configuration loaded to PG/PC is not completely identical to the configuration originally loaded to the device. Note the additional information on loading, in particular with regard to partially loaded configuration data in the case of cross-device communication.

Loading PC systems, such as WinAC or PC-based automation, is not possible.

See also

General information on loading (Page 348)

Loading specific device configurations

Information on loading

When device configurations are loaded to the PG/PC, all assigned parameters are transferred from the device to the project. If the CPU is connected to a subnet, all parameters of the device are loaded and the CPU is displayed in the network view as networked.

Note

CPUs reset to their factory settings do not have any hardware configuration. Therefore, nothing is loaded in this case after "Load to PG/PC" is selected in the "Online" menu.

Loading S7-300/400 configurations

To avoid conflicts when loading a device to an existing project, the following rules must be followed:

- Device names for CPUs, PROFIBUS slaves (DP slaves, I-slaves) and PROFINET devices (IO devices, I devices) must be unique
- The combination of network name, subnet ID, and IP/DP address for modules must be unique

If conflicts occur, the load operation is canceled and an alarm provides information about the problems that occurred. You can then adapt the project correspondingly or install missing components and then repeat the load operation.

Alarm configurations are not loaded to PG/PC.

Loading S7-1200/1500 configurations

Take the following into consideration when loading:

- CPUs of the S7-1200 series with firmware version V1.0 are not supported when loading.
- Device-specific diagnostic interrupts of the S7-1200 are not supported when loading. For device-specific diagnostic interrupts of the S7-1200 to be generated on the PG/PC once again, the hardware configuration must be compiled once again.
- Module comments of the S7-1200/1500 are loaded from the device to the PG/PC if the same project language is set that was used during loading to the device. You can deselect the loading of comments, if desired.

Note

All types of PC systems, such as WinAC, Embedded Controller, CP 1616 or PC CPs, do not support loading to PG/PC.

Loading distributed I/O

The following functionalities and settings of the distributed I/O are loaded:

- DP master systems/IO systems with associated DP masters/IO controllers (CPUs and CPs), DP slaves/IO devices, the utilized modules and their parameter assignment and properties, such as options handling, status bytes, or SYNC/FREEZE
- Connection of process image partitions (PIP) to organization blocks (OB). Applies to the module and OB properties
- Configured hardware interrupts with the associated properties
- DP master systems with I-slave
- CPs as PROFIBUS I-slave or PROFINET I-device
- Direct data exchange

Master-slave relationships between the I-slave/I-device and assigned DP master/IO controller are only established in the project if both the master as well as the I-slave are loaded to the PG. It does not matter whether you load the DP master/IO controller or the I-slave/I-device first. As soon as both devices are loaded, the master-slave/controller-device relationships will also be re-established.

Loading subnets and devices with MPI, PROFIBUS, Ethernet, and PtP

The following special considerations apply to loading subnets and end points of connections for MPI, PROFIBUS, Ethernet, and PtP with their respective connection properties:

- If a device with PROFIBUS interface is loaded, the bus parameters of the device will initially be different from the settings in the original project. The bus parameters will only match those of the original project after all devices involved are loaded and no additional devices are on the same bus.
- Passive communication devices that are not connected as DP slaves or IO devices to a corresponding master system or IO system do not participate in the data exchange. They are therefore not loaded.
- All devices involved should therefore be loaded for cross-device configurations. A warning is output during compiling of the project for missing network stations. Missing routing information due to communication stations that are not loaded is displayed as warning during compilation. If the configuration is loaded from the PG/PC back to the device, different routing information results.

When you compile the project after loading devices to the PG/PC, STEP 7 checks if all devices to which communication relationships have been configured are available. If devices are missing, you receive an alarm with the number of missing communication stations.

NOTICE
Cross-device communication When you load a configuration with cross-device communication to the PG/PC, you must also load all corresponding network stations to the PG/PC. If required network stations are missing and the configuration is loaded back to the device, there is no guarantee that cross-device communication is working again.

Loading configurations with web server

Information on loading

The hardware configuration of a CPU also contains the Web server settings. A number of restrictions apply to loading a Web server configuration to the PG/PC:

- The assignment of the Web server language and project language is not loaded for S7-300/400. The project texts are not loaded and an alarm is output that no project languages are assigned. The languages assigned in STEP 7 are loaded without restriction for S7-1200/1500 CPU.
- User administration data of the S7-1200/1500 can be loaded but not edited. You can use a check box to select whether you want to use the existing data as read-only data or discard these data and enter new data.
- Watch tables of the Web server are not loaded.

The source files of the user-defined web pages (HTML pages, Java scripts, etc.) are not loaded. The program blocks generated during loading can only be edited if you enter the properties and the HTML page yourself.

See also

Information about the web server (Page 1050)

Loading configurations with PROFIBUS

Information on loading

A DP master is loaded to PG/PC. The DP master system and all connected DP slaves are inserted into the project. The respective settings remain unchanged. If a suitable PROFIBUS subnet has already been created, the loaded devices with PROFIBUS interface are connected to the existing subnet.

As a prerequisite for loading DP master systems with standard slaves, the corresponding GSD files must be installed in the TIA Portal and available in the hardware catalog. If a required GSD file is not available in the same version as in the device, differences are identified during the consistency check.

Note

Direct data exchange in a configuration is only loaded if all communication partners involved in the direct data exchange are loaded to the PG/PC.

Isochronous mode

Note the following when loading DP master systems with activated PROFIBUS functionality "Isochronous mode":

- Bus parameters and the settings for isochronous mode are only identical after the device is loaded to the PG/PC if all devices relevant for calculation of isochronous mode are loaded.
- Only mono-master systems with isochronous mode are supported. Therefore, only configurations with just one DP master on the PROFIBUS subnet are loaded.

I-slave

Master-slave relations between DP master and I slave are only established in the project if both the DP master and the I slave are downloaded to the PG. It does not matter whether you load the DP master or the I-slave first. If you load the DP master from a DP master system with connected I-slave, the DP master and its DP slaves are loaded. An I-slave proxy is loaded as dummy module for an involved I-slave.

For I-slave proxy devices:

- DP master can be complied and loaded
- Properties are displayed but cannot be changed
- Diagnostics in the project tree is not carried out

In the network view, proxy devices are represented with a question mark:

It is possible to load a DP master with connected I-slave proxy from the PG/PC to the device.

Replacing the I-slave proxy

In order to edit the I-slave in the project, you must load the I-device from the device to the PG/PC. The I-slave proxy is hereby replaced by the full I-slave with its subordinate DP slaves.

You have the following options for replacing the I-slave proxy with the corresponding I-slave:

- In the menu bar with the command "Online >Upload device as new station (hardware and software)
- With selected I-slave proxy using the shortcut menu command "Upload from device > Hardware and software".
- In the toolbar with the button "Upload from device".

Note

The replacement of I-slave proxy devices is only possible if the required I-device is available in the hardware catalog.

I-slaves protected by a password are not loaded.

Loading configurations with PROFINET

Information on loading

If you have selected a CPU in the list of accessible devices and perform an load operation to PG/PC, all IO controllers and IO devices associated with this device along with their IO systems will be loaded. Settings for the topology are also transferred. If there is already a suitable Ethernet network in the project, the loaded devices are integrated into the existing network.

Relationships between the IO controllers and IO devices are only established within the project if both the IO controller as well as the I-device are loaded to the PG. It does not matter whether you load the IO controller or the I-devices first.

Supported functions

The following functionalities and settings are loaded:

- PROFINET configurations (RT and IRT) in IO systems with the associated IO controllers (CPUs and CPs), IO devices, and the modules used
- Logical addresses and interface properties
- Port interconnections
- Isochronous mode
- Sync domains/MRP domains
- Redundancy role "Client" or "Manager" for MRP configurations

Note

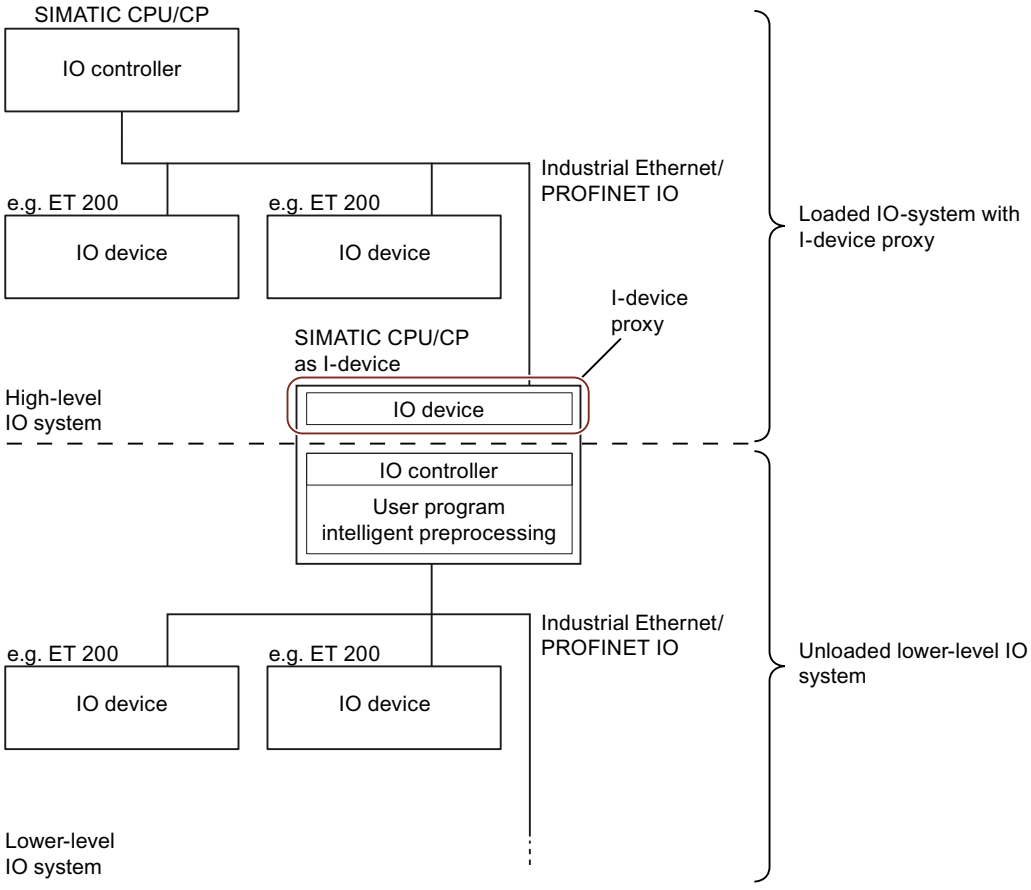
Empty Sync domains and MRP domains are not included in the loading.

GSD-based IO devices

As a prerequisite for loading GSD-based IO devices, the corresponding GSD files must be installed in the TIA Portal and available in the hardware catalog. If a required GSD file is not available in the same version as in the device, differences are identified during the consistency check.

I-device

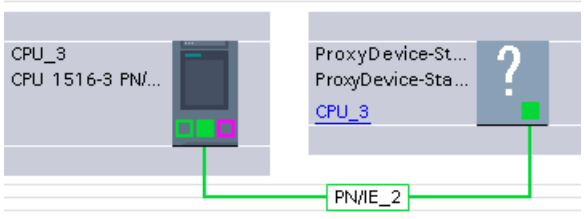
If you load the IO controller from an IO system with connected I-device, the IO controller and its IO devices will be loaded. An I-device proxy is loaded as dummy module for an involved I-device. The CPU parameter assignment, including the parameter assignment of the CPU's "own" (subordinate) IO system, is missing in the I-device proxy. Only the interface to the higher-level IO controller is loaded.



For I-device proxy devices:

- IO controller can be compiled and loaded
- Properties are displayed but cannot be changed
- Diagnostics in the project tree is not carried out

In the network view, proxy devices are represented with a question mark:



It is possible to load an IO controller with connected I-device proxy from the PG/PC to the device.

Replacing the I-device proxy

In order to edit the I-device in the project, you must load the I-device from the device to the PG/PC. This replaces the I-device proxy with the complete I-device along with its subordinate IO devices.

You have the following options for replacing the I-device proxy with the corresponding I-device:

- In the menu bar with the command "Online >Upload device as new station (hardware and software)
- With selected I-device proxy using the shortcut menu command "Upload from device > Hardware and software".
- In the toolbar with the button "Upload from device".

Note

The replacement of I-device proxy devices is only possible if the required I-device is available in the hardware catalog.

I-devices protected by a password are not loaded.

An I-device proxy representing a SIMOTION I-device cannot be loaded and replaced.

Configurations with IE/PB link

If one of the following configurations with IE/PB Link PN IO as PROFINET IO device is present, the complete configuration with all subordinate PROFIBUS devices is loaded:

- CPU/CP of S7-300/400
- PC station and connected PROFIBUS master system

The complete configuration consists of the following:

- CPU
- CP configuration
- PROFINET IO system with connected IE/PB link
- PROFIBUS master system of the IE/PB link with connected DP slaves

An example configuration consists of an S7-300 CPU with a CP as PROFINET IO controller. An IE/PB link as IO device is connected to the IO controller. As the PROFIBUS DP master, the IE/PB link polls a PROFIBUS DP slave, e.g., ET 200L. If you load the CPU from the device to the PG/PC, the complete configuration is loaded.

Note

If the IE/PB link is not operated as a PROFINET IO proxy but rather as a gateway in standard mode, the IE/PB link functions as a CPU and can be loaded separately.

Loading Shared Devices

The following applies to loading comments: If an input/output module with the Module-internal Shared Input (MSI) or Module-internal Shared Output (MSO) function consists of only one submodule, the submodule does not have its own comment. Instead, it uses the comment of the input/output module. The module must be subdivided into several submodules for the input/output module and all submodules to have their own comment fields.

Loading HMI devices

Information on loading

We distinguish between the following cases when loading HMI devices to PG/PC:

- HMI devices connected to a DP master as DP slave or to an IO controller as IO device are loaded as DP slave or IO device respectively (for example, PP 17-I PROFIsafe).
- HMI devices in a master system as I-slave or in an IO system as I-device are loaded as I-slave proxies or I-device proxies (Page 1822) (for example, SIMATIC Comfort Panels). The settings of the device proxy are read-only.
- HMI devices are not loaded if they are connected to a subnet (PROFIBUS or PROFINET) but not to a master system or IO system (for example, KP600 Basic color DP).

Going online with loaded configurations

You can go online with the loaded project or with loaded portions of the project.

Requirement

The hardware configuration loaded from the device to the PG/PC has been compiled. The statuses of the central and distributed modules are only correctly displayed after compilation.

Note

If you go online before compiling, the diagnostics icon “?” is displayed (diagnostics not possible). A corresponding alarm is displayed under "Info > General" in the Inspector window.

Dependencies

Depending on how completely the hardware configuration was loaded to the PG, restrictions apply to going online and to diagnostics:

- Completely loaded device with all associated central and distributed modules, such as DP slaves or IO devices:
  Going online and diagnostics are possible.
- Loaded device with connected I-devices/I-slaves:
  - I device/I slave is not loaded: Going online for the device and its modules is possible. For the dependent components of the configuration that are not loaded, device proxies are handled with only minimum diagnostics support. The online status is represented as an icon. The standard diagnostics is shown in the online and diagnostics view. I&M data are not loaded.
  - I device/I slave is also loaded: Going online is possible for all devices; diagnostics is fully supported.

10.1.5 Additional information on configurations

10.1.5.1 Identification systems

RFID systems

Blocks

Ident profile and Ident blocks

You will find detailed information on the Ident profile in the manual "Ident Profile and Ident Blocks, Standard Function for Ident Systems" on the pages of "Industry Online Support (https://support.industry.siemens.com/cs/ww/en/ps/14971/man)".

Programming Ident blocks

Basic blocks

Read

The "Read" block reads the user data from the transponder and enters this in the "IDENT_DATA" buffer. The physical address and the length of the data are transferred using the "ADDR_TAG" and "LEN_DATA" parameters. With the RF68xR readers, the block reads the data from memory bank 3 (USER area). Specific access to a certain transponder is made with the "EPCID_UID" and "LEN_ID".

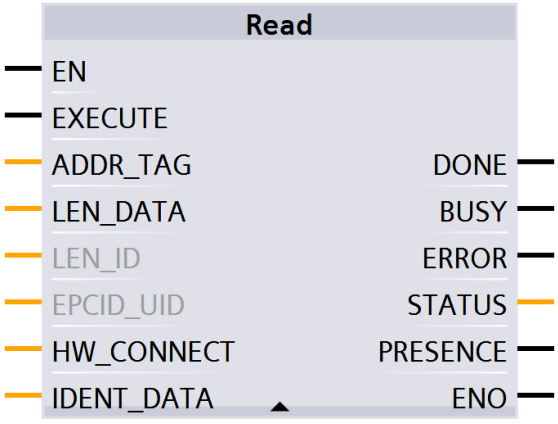


Figure 10-20 "Read" block

Table 10-108 Explanation of the "Read" block

Parameter	Data type	Default values	Description
ADDR_TAG	DWord	DW#16#0	Physical address on the transponder where the read starts. You will find more information on addressing in the section "AUTOHOTSPOT". With MV: The length of the read code is located starting at address "0" (2 bytes). The read code is located starting at address "2". [1)]
LEN_DATA	Word	W#16#0	Length of the data to be read
LEN_ID	Byte	B#16#0	Length of the EPC-ID/UID Default value: 0x00 ≙ unspecified single tag access (RF680R, RF685R)

Parameter	Data type	Default values	Description
EPCID_UID	Array[1...62] of Byte	0	Buffer for up to 62 bytes EPC-ID, 8 bytes UID or 4 bytes handle ID. • 2 - 62-byte EPC-ID is entered at the start of the buffer (length is set by "LEN_ID") • 8-byte UID is entered at the start of the buffer ("LEN_ID = 8") • 4-byte handle ID must be entered in the array element [5]-[8] ("LEN_ID = 8") Default value: 0x00 ≙ unspecified single tag access (RF620R, RF630R)
IDENT_DATA	Any / Variant	0	Data buffer in which the read data is stored. Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/ UDTs can also be created.

[1)] You will find further information on working with optical reader systems in the operating instructions "SIMATIC MV420 / SIMATIC MV440".

Read_MV

The "Read_MV" block reads out the read result of a camera. The length of the data to be read is calculated automatically by the camera based on the length of the created receive buffer. The actual length of the read result is output in the "LEN_DATA" output parameter. The data will be saved in the "IDENT_DATA" data buffer. If the buffer is too small, the error message "0xE7FE0400" appears and the expected length is output at "LEN_DATA".

To achieve an optimum speed, we recommend that you adapt the length of the data type "IDENT_DATA" so that this is as close as possible to the maximum expected length of the read result (2 bytes code length + read code).

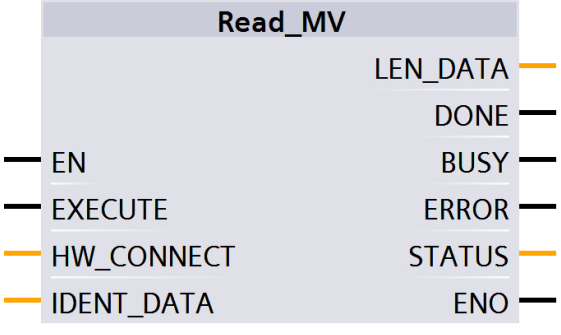


Figure 10-21 "Read_MV" block

Table 10-109 Explanation of the "Read_MV" block

Parameter	Data type	Default values	Description
IDENT_DATA	Any / Variant	0	Read result The length of the read code is located in bytes 0 and 1.
LEN_DATA	Word	W#16#0	Length of the read result ≙ 2 bytes code length + read code Note: For Variant, an "Array_of_Byte" with a variable length can be created. For Any, other data types/UDTs can also be created.

Reset_Reader

The "Reset_Reader" block can currently only be used in conjunction with the RF680R and RF685R readers or the RF120C communications module with a reader connected.

Using the "Reset_Reader" block, you can reset all reader types of the Siemens RFID systems. All the readers are reset to the settings stored in the device configuration of the RF120C or that were configured in the RF68x reader using the WBM. The "Reset_Reader" block does not have any device-specific parameters and is executed using the "EXECUTE" parameter.

You will find descriptions of other Reset blocks for operation with the communications modules RF180C and ASM 456 or optical reader systems in the section "AUTOHOTSPOT".

With the "Reset_Reader" block and the other reset blocks, you can interrupt any active Ident block at any time. These blocks are then ended with "DONE = true" and "ERROR = false".

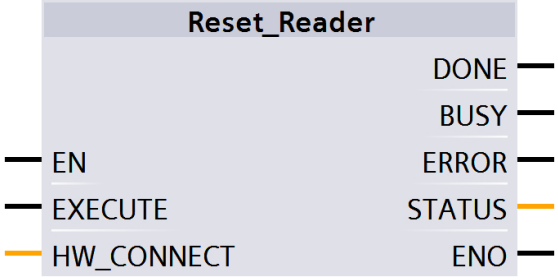


Figure 10-22 "Reset_Reader" block

Set_MV_Program

With the aid of the "Set_MV_Program" block, you can change the program in a camera. The required program number is transferred using the "PROGRAM" parameter.

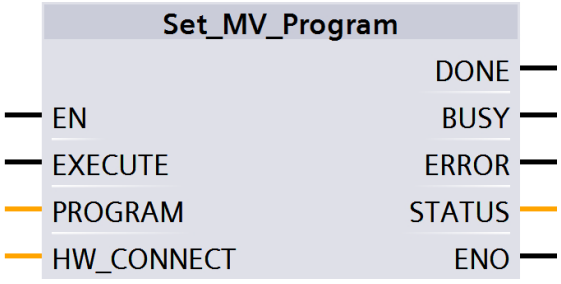


Figure 10-23 "Set_MV_Program" block

Table 10-110 Explanation of the "Set_MV_Program" block

Parameter	Data type	Default values	Description
PROGRAM	Byte	B#16#1	Program number Range of values: 0x01 ... 0x0F

Write

The "Write" block writes the user data from the "IDENT_DATA" buffer to the transponder. The physical address and the length of the data are transferred using the "ADDR_TAG" and "LEN_DATA" parameters. With the RF68xR readers, the block writes the data to memory bank 3 (USER area). Specific access to a certain transponder is made with the "EPCID_UID" and "LEN_ID".

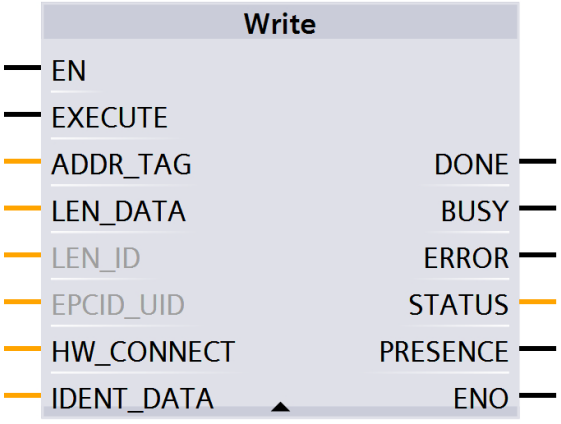


Figure 10-24 "Write" block

Table 10-111 Explanation of the "Write" block

Parameter	Data type	Default values	Description
ADDR_TAG	DWord	DW#16#0	Physical address on the transponder where the write starts. You will find more information on addressing in the section "AUTOHOTSPOT". With MV: Address is always 0. [1]
LEN_DATA	Word	W#16#0	Length of the data to be written
LEN_ID	Byte	B#16#0	Length of the EPC-ID/UID Default value: 0x00 ≙ unspecified single tag access (RF680R, RF685R)
EPCID_UID	Array[1...62] of Byte	0	Buffer for up to 62 bytes EPC-ID, 8 bytes UID or 4 bytes handle ID. • 2 - 62-byte EPC-ID is entered at the start of the buffer (length is set by "LEN_ID") • 8-byte UID is entered at the start of the buffer ("LEN_ID = 8") • 4-byte handle ID must be entered in the array element [5]-[8] (LEN_ID = 8) Default value: 0x00 ≙ unspecified single tag access (RF620R, RF630R)
IDENT_DATA	Any / Variant	0	Data buffer with the data to be written. With MV: The first byte encodes the corresponding MV command.[1] Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/ UDTs can also be created.

[1] You will find further information on working with optical reader systems in the operating instructions "SIMATIC MV420 / SIMATIC MV440".

Extended blocks

Config_Upload/-_Download

Using the "Config_Upload" and "Config_Download" blocks, you can read out ("Config_Upload") or write ("Config_Download") the configuration of the RF680R/RF685R readers via the control program.

The configuration data is not interpretable data. Save the data on the controller so that it can be written to the reader again if a device is replaced. Bytes 6-9 (see table below) contain a configuration ID with a unique version identifier. With the configuration ID, when performing a

"Config_Upload", you can check whether the configuration data read matches the configuration data stored on the controller. The configuration data has the following structure:

Table 10-112 Structure of the configuration data

Byte	Name
0	Structure identifier (2 bytes)
2	Length information (4 bytes) Length of the version identifier and parameter block
6	Version identifier (4 bytes) Based on the identifier, you can uniquely identify the configuration. This is a time stamp in Linux format. The time stamp indicates how many seconds have passed since January 1, 1979, 00:00 (midnight). The identifier is assigned when a configuration is generated.
10 ... end "DATA"	Parameter block

"Config_Upload/Config_Download" can be executed on every channel of the RF680R/ RF685R. It is always the same configuration data that is transferred.

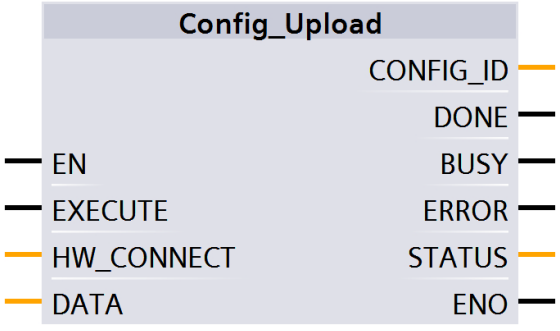


Figure 10-25 "Config_Upload" block

Table 10-113 Explanation of the "Config_Upload" block

Parameter	Data type	Description
DATA	Any / Variant	Data buffer for configuration data. The real length of the data depends on the complexity of the configuration and the firmware version of the reader. With a standard configuration of the RF680R/RF685R reader, we recommend a memory size of 4 KB. If you use advanced reader configurations (filtering) or want to change the configuration in the future without needing to adapt the memory size of "DATA", we recommend a memory size of 8-16 KB. Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/UDTs can also be created.
CONFIG_ID	DWord	Version identifier (4 bytes) Based on the identifier, you can uniquely identify the configuration. This is a time stamp in Linux format. The time stamp indicates how many seconds have passed since January 1, 1979, 00:00 (midnight). The identifier is assigned when a configuration is generated.

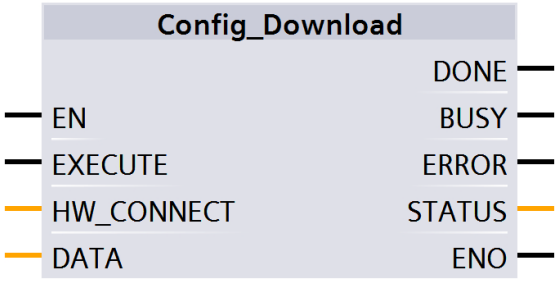


Figure 10-26 "Config_Download" block

Table 10-114 Explanation of the "Config_Download" block

Parameter	Data type	Description
DATA	Any / Variant	Data buffer for configuration data. The real length of the data depends on the complexity of the configuration and the firmware version of the reader. With a standard configuration of the RF680R/RF685R reader, we recommend a memory size of 4 KB. If you use advanced reader configurations (filtering) or want to change the configuration in the future without needing to adapt the memory size of "DATA", we recommend a memory size of 8-16 KB. Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/UDTs can also be created.

Inventory

The "Inventory" block activates the taking of inventories. With the RF620R and RF630R readers, inventories are always taken as soon as the antenna is turned on.

Special feature of the RF680R, RF685R readers

Note that the length of the data buffer ("IDENT_DATA") must correspond to at least the length of the maximum expected data. If more transponders are identified and data read out than have space in the assigned buffer length of "IDENT_DATA", the data of these transponders is lost. This reaction is indicated by the error "E7FE0400h" (buffer overflow).

For the RF680R and RF685R readers, the parameters "DURATION" and "DUR_UNIT" are also available. Using the parameters, you can specify the duration of the inventories.

With the RF680R/RF685R readers. there are four different modes that you can select with the "ATTRIBUTE" parameter.

- At the start, a certain duration/number (period of time, number of inventories, number of "observed" events or identified transponders) is specified. A distinction is made between the following options:
  - Duration
    Take inventories for a specified period of time
  - Number of inventories
    Take a specified number of inventories
  - Number of "observed" events
    Take inventories until a specified number of transponders have been identified at the same time.

  Inventories are then taken by the reader for this time or number of inventories. When the specified time/number is reached, the block is ended and returns all identified transponders in "IDENT_DATA". In other words, other commands can only be executed when all inventories have been taken completely. The unit (time or number) is specified using "DUR_UNIT" and the value (time value or number) using "DURATION". This mode can be executed using the attributes "0x80" and "0x81". Depending on the attribute, more or less data is supplied about the identified transponders.
- With the attributes "0x86" (start "Presence_Mode") and "0x87" (end "Presence_Mode"), inventories can be taken permanently. The presence of a transponder can then always be queried using "PRESENCE" without needing to start the block with "EXECUTE". No information about the identified transponders is returned when the command executes! To obtain information about the identified transponders, use one of the two calls listed above (with time / number of inventories = 0).
  When this mode is active, commands relating to transponders are not executed immediately but only when a transponder is identified. This achieves shorter reaction times since the command is already pending when the transponder enters the antenna field.
  The "Presence_Mode" is practical in the context of the "Repeat command" function.

The "NUMBER_TAGS" output parameter is used to output the number of identified transponders. With the attributes "0x80" and "0x81" on completion of the read operation, the sum of all identified transponders is displayed. With the attribute "0x86" the number of currently identified transponders is shown at the "NUMBER_TAGS" output parameter (max. 15), without needing to start the module with "EXECUTE".

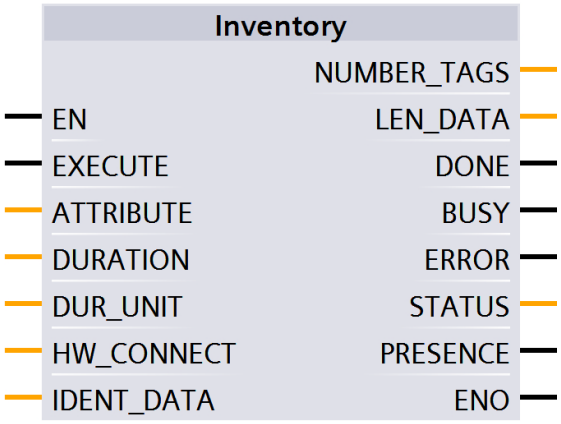


Figure 10-27 "Inventory" block

Table 10-115 Explanation of the "Inventory" block

Parameter	Data type	Default values	Description
ATTRIBUTE	Byte	B#16#0	Selecting the status mode: • RF300, MOBY U: 0x00 • RF620R, RF630R: 0x82 (read out next data record), 0x83, 0x85, 0x90, 0x91, 0x92 • RF680R, RF685R: 0x80, 0x81, 0x86, 0x87
DURATION	Word	W#16#0	RF680R, RF685R: Duration dependent on "DUR_UNIT" Period of time or number of inventories or number of "Observed" events Example: • 0x00 ≙ no inventory • 0x01 ≙ one inventory
DUR_UNIT	Word	W#16#0	RF680R, RF685R: Unit for "DURATION" • 0x00 ≙ time [ms] • 0x01 ≙ inventories • 0x02 ≙ number of "Observed" events
IDENT_DATA	Any / Variant	0	Data buffer for inventory data Note: For Variant, an "Array_of_Byte" with a variable length can be created. For Any, other data types/UDTs can also be created.
NUMBER_TAGS	Int	0	Number of transponders in the antenna field
LEN_DATA	Word	W#16#0	Length of the valid data

Results for MOBY U

Table 10-116 ATTRIBUTE "0x00" (data type "IID_INVENT_00_MOBY_U")

Name	Type	Comment
number_MDS	WORD	Number of MDS
UID_length	WORD	length of UID
UID	ARRAY[1..12] of IID_IN_I_8BYTE	
UID[1]	IID_IN_I_8BYTE	
UID	ARRAY[1..8] of BYTE	
UID[1]	BYTE	
UID[2]	BYTE	
UID[3]	BYTE	
UID[4]	BYTE	
UID[5]	BYTE	
UID[6]	BYTE	
UID[7]	BYTE	
UID[8]	BYTE	
UID[2]	"IID_IN_I_8BYTE"	
UID[3]	"IID_IN_I_8BYTE"	
UID[4]	"IID_IN_I_8BYTE"	
UID[5]	"IID_IN_I_8BYTE"	
UID[6]	"IID_IN_I_8BYTE"	
UID[7]	"IID_IN_I_8BYTE"	
UID[8]	"IID_IN_I_8BYTE"	
UID[9]	"IID_IN_I_8BYTE"	
UID[10]	"IID_IN_I_8BYTE"	
UID[11]	"IID_IN_I_8BYTE"	
UID[12]	"IID_IN_I_8BYTE"	

Results for RF620R, RF630R

Table 10-117 ATTRIBUTE "0x83" (data type "IID_INVENT_82_83_RF600") for RF620R, RF630R with EPC-ID/UID

Name	Type	Comment
reserved0	BYTE	
number_MDS	BYTE	Number of MDS
EPC	ARRAY[1..19] of "IID_IN_I_12BYTE"	
EPC[1]	"IID_IN_I_12BYTE"	
ID	ARRAY[1..12] of BYTE	
ID[1]	BYTE	

Name	Type	Comment
ID[2]	BYTE	
ID[3]	BYTE	
ID[4]	BYTE	
ID[5]	BYTE	
ID[6]	BYTE	
ID[7]	BYTE	
ID[8]	BYTE	
ID[9]	BYTE	
ID[10]	BYT	
ID[11]	BYTE	
ID[12]	BYTE	
EPC[2]	"IID_IN_I_12BYTE"	
EPC[3]	"IID_IN_I_12BYTE"	
EPC[4]	"IID_IN_I_12BYTE"	
EPC[5]	"IID_IN_I_12BYTE"	
EPC[6]	"IID_IN_I_12BYTE"	
EPC[7]	"IID_IN_I_12BYTE"	
EPC[8]	"IID_IN_I_12BYTE"	
EPC[9]	"IID_IN_I_12BYTE"	
EPC[10]	"IID_IN_I_12BYTE"	
EPC[11]	"IID_IN_I_12BYTE"	
EPC[12]	"IID_IN_I_12BYTE"	
EPC[13]	"IID_IN_I_12BYTE"	
EPC[14]	"IID_IN_I_12BYTE"	
EPC[15]	"IID_IN_I_12BYTE"	
EPC[16]	"IID_IN_I_12BYTE"	
EPC[17]	"IID_IN_I_12BYTE"	
EPC[18]	"IID_IN_I_12BYTE"	
EPC[19]	"IID_IN_I_12BYTE"	

Note

Number of EPC-IDs

"number_MDS" specifies the number of EPC-IDs (1 to 19) transferred with the "INVENTORY" block. To receive the handle IDs of all transponders located in the antenna field, it may be necessary to run the "INVENTORY" block again with ATTRIBUTE "0x82".

Table 10-118 ATTRIBUTE "0x83", "0x90", "0x91" und "0x92" (data type "IID_INVENT_8x_9x_RF6_D") for RF620R, RF630R with handle ID

Name	Type	Comment
reserved	BYTE	
number_MDS	BYTE	Number of MDS

Name	Type	Comment
UID	ARRAY[1..29] of "IID_IN_I_8BYTE"	
UID[1]	"IID_IN_I_8BYTE"	
UID	ARRAY[1..8] of BYTE	
UID[1]	BYTE	
UID[2]	BYTE	
UID[3]	BYTE	
UID[4]	BYTE	
UID[5]	BYTE	
UID[6]	BYTE	
UID[7]	BYTE	
UID[8]	BYTE	
UID[2]	"IID_IN_I_8BYTE"	
UID[3]	"IID_IN_I_8BYTE"	
UID[4]	"IID_IN_I_8BYTE"	
UID[5]	"IID_IN_I_8BYTE"	
UID[6]	"IID_IN_I_8BYTE"	
UID[7]	"IID_IN_I_8BYTE"	
UID[8]	"IID_IN_I_8BYTE"	
UID[9]	"IID_IN_I_8BYTE"	
UID[10]	"IID_IN_I_8BYTE"	
UID[11]	"IID_IN_I_8BYTE"	
UID[12]	"IID_IN_I_8BYTE"	
UID[13]	"IID_IN_I_8BYTE"	
UID[14]	"IID_IN_I_8BYTE"	
UID[15]	"IID_IN_I_8BYTE"	
UID[16]	"IID_IN_I_8BYTE"	
UID[17]	"IID_IN_I_8BYTE"	
UID[18]	"IID_IN_I_8BYTE"	
UID[19]	"IID_IN_I_8BYTE"	
UID[20]	"IID_IN_I_8BYTE"	
UID[21]	"IID_IN_I_8BYTE"	
UID[22]	"IID_IN_I_8BYTE"	
UID[23]	"IID_IN_I_8BYTE"	
UID[24]	"IID_IN_I_8BYTE"	
UID[25]	"IID_IN_I_8BYTE"	
UID[26]	"IID_IN_I_8BYTE"	
UID[27]	"IID_IN_I_8BYTE"	
UID[28]	"IID_IN_I_8BYTE"	
UID[29]	"IID_IN_I_8BYTE"	

Name	Type	Comment
reserved1	DWORD	
Data	ARRAY[1..222] of BYTE	

Note

Number of handle IDs

"number_MDS" specifies the number of handle IDs (1 to 29) transferred with the "INVENTORY" block. To receive the handle IDs of all transponders located in the antenna field, it may be necessary to run the "INVENTORY" block again with ATTRIBUTE "0x82".

Table 10-119 ATTRIBUTE "0x85" (data type "IID_INVENT_85_RF600")

Name	Type	Comment
reserved	BYTE	
number_MDS	STRUCT	Number of MDS
ID	BYTE	
ID[1]	BYTE	
Handle	ARRAY[1..8] of BYTE	
Handle[1]	BYTE	
Handle[2]	BYTE	
Handle[3]	BYTE	
Handle[4]	BYTE	
Handle[5]	BYTE	
Handle[6]	BYTE	
Handle[7]	BYTE	
Handle[8]	BYTE	
EPC	ARRAY[1..12] of BYTE	
EPC[1]	BYTE	
EPC[2]	BYTE	
EPC[3]	BYTE	
EPC[4]	BYTE	
EPC[5]	BYTE	
EPC[6]	BYTE	
EPC[7]	BYTE	
EPC[8]	BYTE	
EPC[9]	BYTE	
EPC[10]	BYTE	
EPC[11]	BYTE	
EPC[12]	BYTE	
ID[2]	"IID_IN_I_20Byte"	

Name	Type	Comment
ID[3]	"IID_IN_I_20Byte"	
ID[4]	"IID_IN_I_20Byte"	
ID[5]	"IID_IN_I_20Byte"	
ID[6]	"IID_IN_I_20Byte"	
ID[7]	"IID_IN_I_20Byte"	
ID[8]	"IID_IN_I_20Byte"	
ID[9]	"IID_IN_I_20Byte"	
ID[10]	"IID_IN_I_20Byte"	
ID[11]	"IID_IN_I_20Byte"	

Note

Number of IDs transferred

"number_MDS" specifies the number of IDs (1 to 11 handle IDs and EPC-IDs) transferred with the "INVENTORY" block. To receive the IDs of all transponders located in the antenna field, it may be necessary to run the "INVENTORY" block again with ATTRIBUTE "0x82".

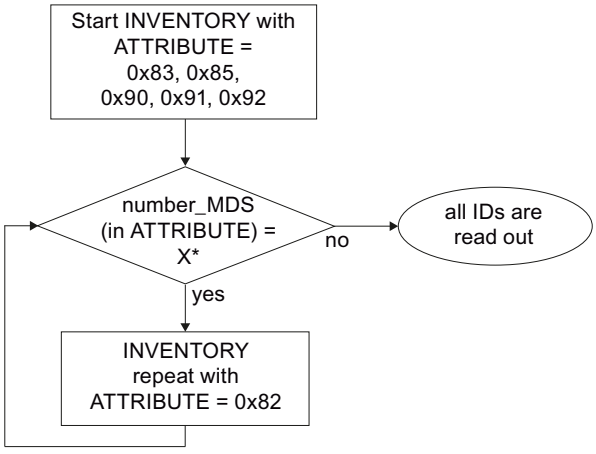
You will find more detailed information on the individual status modes in the manuals matching the modes "FB 45", "FB55" and "SIMATIC RF620R/RF630R".

The identifiers of the status modes correspond to the following identifiers in the other manuals:

0x82 ≙ 0x02
0x83 ≙ 0x03
0x85 ≙ 0x05
0x90 ≙ 0x10
0x91 ≙ 0x11
0x92 ≙ 0x12

Programming ATTRIBUTE "0x82"

If the number of transponders in the antenna field is unknown, repeat the "INVENTORY" block with the ATTRIBUTE = "0x82".



* The number of returned IDs "X" depends on the "ATTRIBUTE" used.

Figure 10-28 Program sequence of ATTRIBUTE "0x82" with unknown transponder populations

Results for RF680R, RF685R

The number of "TAG_DATA[x]" elements of the data types of the ATTRIBUTES "0x80" and "0x81" depends on the number of transponders to be expected. For this reason, you need to assemble the receive buffer yourself. Not the following structure when creating the receive buffer :"IDENT_DATA"/data type:

- The first element "NUM_MDS" is always of the type "WORD".
- The next element "TAG_DATA" is always of the type "ARRAY". The number of transponders to be expected ("n") must be entered in the "ARRAY".

The following tables show an example of the structure of the receive buffer "IDENT_DATA"/ data type for the ATTRIBUTES "0x80" and "0x81".

Table 10-120 ATTRIBUTE "0x80"

Name	Type	Comment
NUM_MDS	WORD	Number of MDS
TAG_DATA	ARRAY[1..n]of IID_IN_I_80	Length of EPC ID
TAG_DATA[1]	IID_IN_I_80	
Reserved	BYTE	
ID_Len	BYTE	Length of EPC ID
EPC_ID	ARRAY[1..62] of BYTE	EPC-ID
tagPC	WORD	
TAG_DATA[2]	IID_IN_I_80	
...	...	
TAG_DATA[n]	IID_IN_I_80	

Table 10-121 ATTRIBUTE "0x81"

Name	Type	Comment
NUM_MDS	WORD	Number of MDS
TAG_DATA	ARRAY[1..n] of IID_IN_I_81	
TAG_DATA[1]	IID_IN_1_81	
reserved	BYTE	
ID_LEN	BYTE	EPC length
EPC_ID	ARRAY[1..62] of BYTE	EPC-ID
tagPC	WORD	
RSSI	BYTE	RSSI value
MaxRSSI	BYTE	highest RSSI value
MinRSSI	BYTE	lowest RSSI value
channel	BYTE	channel; 1..15_ESTI; 1..53:FCC
antenna	BYTE	antenna; bit coded; Bit 0=antenna 1; Bit 1=antenna 2
polarization	BYTE	polarizatuin of antenna; 0=undefined; 1=circular
time	Time_OF_Day	S7 time
power	BYTE	power in dBm
filterDataAvailable	BYTE	0=false; 1=true
Inventoried	WORD	[1)]
TAG_DATA[2]	IID_IN_1_81	
...	...	
TAG_DATA[n]	IID_IN_1_81	

[1)] Indicates how often the transponder was identified via the air interface before it changed to the "Observed" status.

Read_EPC_Mem

The "Read_EPC_Mem" block reads data starting at address 4 from the EPC memory of the RF600 transponder. Access is to memory cell 1 as of the start address. The length of the EPC memory to be read out is specified by the "LEN_DATA" parameter.

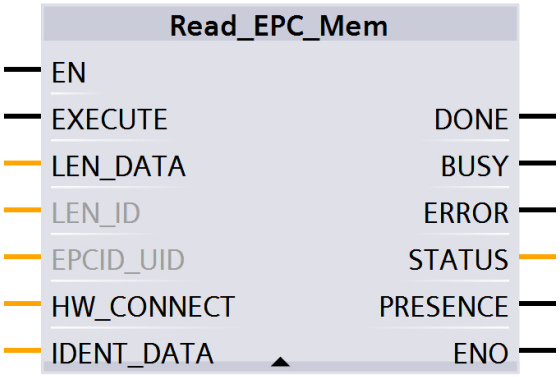


Figure 10-29 "Read_EPC_Mem" block

Table 10-122 Explanation of the "Read_EPC_Mem" block

Parameter	Data type	Default values	Description
LEN_DATA	Word	W#16#0	Length of the EPC memory to be read out (1 ... 62 bytes)
LEN_ID	Byte	B#16#0	Length of the EPC-ID/UID Default value: 0x00 ≙ unspecified single tag access (RF680R, RF685R)
EPCID_UID	Array[1...62] of Byte	0	Buffer for up to 62 bytes EPC-ID, 8 bytes UID or 4 bytes handle ID. • 2 - 62-byte EPC-ID is entered at the start of the buffer (length is set by "LEN_ID") • 8-byte UID is entered at the start of the buffer ("LEN_ID = 8") • 4-byte handle ID must be entered in the array element [5]-[8] ("LEN_ID = 8") Default value: 0x00 ≙ unspecified single tag access (RF620R, RF630R)
IDENT_DATA	Any / Variant	0	Data buffer in which the read EPC memory data is stored. Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/ UDTs can also be created.

Read_TID

The "Read_TID" block reads data from the TID memory area (Tag Identification Memory Bank) of the RF600 transponder. The length of the TID to be read is specified by the "LEN_DATA" parameter. The length of the TID varies depending on the transponder and can be found in the transponder data sheet.

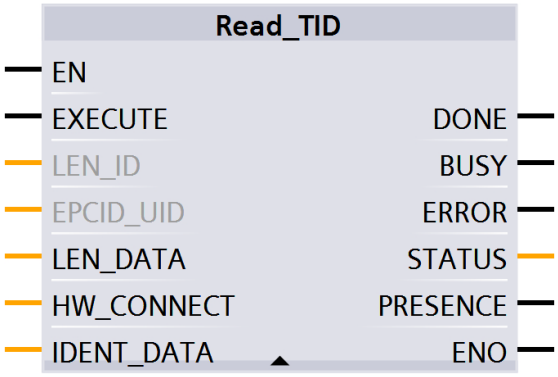


Figure 10-30 "Read_TID" block

Table 10-123 Explanation of the "Read_TID" block

Parameter	Data type	Default values	Description
LEN_ID	Byte	B#16#0	Length of the EPC-ID/UID Default value: 0x00 ≙ unspecified single tag access (RF680R, RF685R)
EPCID_UID	Array[1...62] of Byte	0	Buffer for up to 62 bytes EPC-ID, 8 bytes UID or 4 bytes handle ID. • 2 - 62-byte EPC-ID is entered at the start of the buffer (length is set by "LEN_ID") • 8-byte UID is entered at the start of the buffer ("LEN_ID = 8") • 4-byte handle ID must be entered in the array element [5]-[8] ("LEN_ID = 8") Default value: 0x00 ≙ unspecified single tag access (RF620R, RF630R)
LEN_DATA	Word	W#16#4	Length of the EPC memory to be read out (1 ... 62 bytes)
IDENT_DATA	Any / Variant	0	Read TID Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/ UDTs can also be created.

Read_UID

The "Read_UID" block reads the UID of an HF transponder. The UID always has a fixed length of 8 bytes.

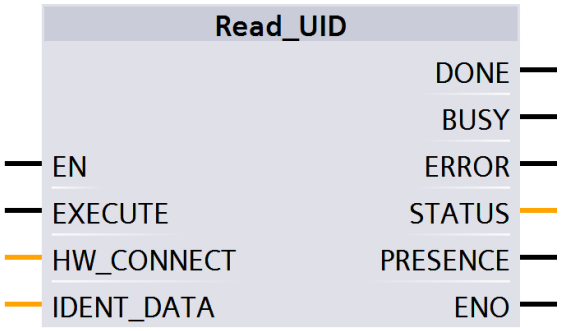


Figure 10-31 "Read_UID" block

Table 10-124 Explanation of the "Read_UID" block

Parameter	Data type	Description
IDENT_DATA	Any / Variant	UID Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/UDTs can also be created.

Set_Ant

With the aid of the "Set_Ant" block, you can turn antennas on or off. There are different blocks for RF300 and RF600. The "Set_Ant_RF300" block can also be used for RF200, MOBY D and MOBY U. The "Set_Ant_RF600" block relates only to the RF620R and RF630R readers.

Set_Ant_RF300

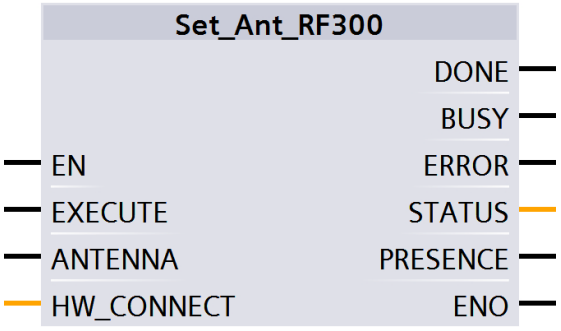


Figure 10-32 "Set_Ant_RF300" block

Table 10-125 Explanation of the "Set_Ant_RF300" block

Parameter	Data type	Description
ANTENNA	Bool	0 = turn antenna off 1 = turn antenna on

Set_Ant_RF600

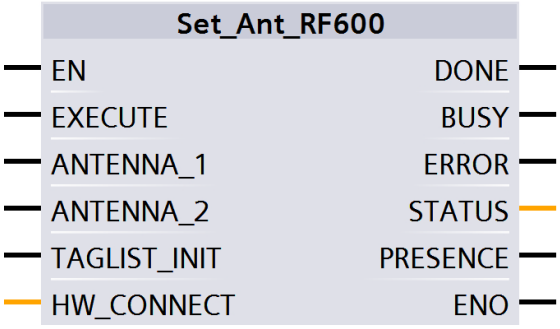


Figure 10-33 "Set_Ant_RF600" block

Table 10-126 Explanation of the "Set_Ant_RF600" block

Parameter	Data type	Description
ANTENNA_1	Bool	0 = turn antenna 1 off 1 = turn antenna 1 on
ANTENNA_2	Bool	0 = turn antenna 2 off 1 = turn antenna 2 on
TAGLIST_INIT	Bool	0 = TagList is reset 1 = the existing TagList continues to be used

Set_Param

With the "Set_Param" block, you can change UHF parameters on an RF680R/RF685R during runtime (e.g. the antenna power).

Note

Settings saved only temporarily

Note that the parameters in the "Set_Param" block are only stored temporarily. If the power for the reader is interrupted, the stored values are lost and must be set again.

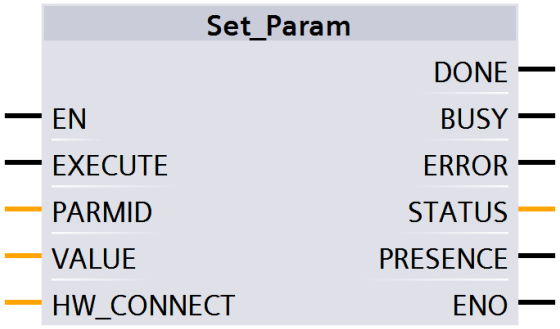


Figure 10-34 "Set_Param" block

Table 10-127 Explanation of the "Set_Param" block

Parameter	Data type	Default values	Description
PARMID	DWORD	0x00	Parameter identifier
VALUE	DWORD	0x00	Parameter value

Table 10-128 Parameter values

PARMID (hex)	PARMID (ASCII)	Parameter	VALUE
0x41315057	A1PW	Antenna 01: Radiated power	Range of values: 0.5 ... 33 Increment: 0.25 Radiated power of the antenna in [dBm]. Bytes 1 and 2 are not used, byte 3 represents the integer and byte 4 the decimal place. Example: A radiated power of 10.25 dBm represents a "VALUE" of "0x0A19".
0x41325057	A2PW	Antenna 02: Radiated power	
0x41335057	A3PW	Antenna 03: Radiated power	
0x41345057	A4PW	Antenna 04: Radiated power	
0x41315452	A1TR	Antenna 01: RSSI threshold	Range of values: 0 ... 255 Threshold value for RSSI. Transponders with lower values are discarded. Value without unit without a direct relationship with the radiated power.
0x41325452	A2TR	Antenna 02: RSSI threshold	
0x41335452	A3TR	Antenna 03: RSSI threshold	
0x41345452	A4TR	Antenna 04: RSSI threshold	
0x5331444C	S1DL	Read point 01: RSSI delta	Range of values: 0 ... 255 Difference for RSSI values. Transponders with lower values relative to the transponder with the highest RSSI value are discarded. Value without unit without a direct relationship with the radiated power.
0x5332444C	S2DL	Read point 02: RSSI delta	
0x5333444C	S3DL	Read point 03: RSSI delta	
0x5334444C	S4DL	Read point 04: RSSI delta	

PARMID (hex)	PARMID (ASCII)	Parameter	VALUE
0x4131504F	A1PO	Antenna 01: Polarization	Range of values: 0, 1, 2, 4 Polarization of the antenna (for intelligent antennas e.g. internal antenna RF685R) • 0: default, undefined • 1: circular • 2: vertical linear • 4: horizontal linear Input is bit coded. Combinations are possible (adding values).
0x4132504F	A2PO	Antenna 02: Polarization	
0x4133504F	A3PO	Antenna 03: Polarization	
0x4134504F	A4PO	Antenna 04: Polarization	
0x52364353	R6CS	Modulation scheme	Range of values: 32, 33, 34, 35, 37, 65 Modulation scheme of the read point Specification of which transponder types are identified (ISO 18000-63/-6B).
0x57544348	WTCH	Date and time	Range of values: 01.01.2000 00:00 a.m. ... 19.01.2038 3:14 a.m. 01.01.2000 01:00 a.m. ≙ 946684800 Date and time (UTC) Time in seconds since 01.01.1970; Setting of the internal reader clock, the date and time are set.
0x57544F44	WTOD	Time	Range of values: 0:00 – 23:59 p.m. S7 time (TOD, UTC) Milliseconds since midnight; Setting of the internal reader clock, only the time is changed but not the date.
0x57444154	WDAT	Date	Range of values: 01.01.2000 ... 18.01.2038 S7 date Days since 01.01.1970; Setting of the internal reader clock, only the date is changed but not the time.

Write_EPC_ID

The "Write_EPC_ID" block overwrites the EPC-ID of the RF600 transponder and adapts the length of the EPC-ID in the memory of the transponder. The new EPC-ID length to be written is specified with the "LEN_ID_NEW" parameter and the previous EPC-ID is specified using the "LEN_ID" and "EPCID_UID" parameters.

Make sure that when you execute the block only one transponder is located in the antenna field. This ensures that the identification when writing the ID is unique. If there is more than one transponder in the antenna field, a negative response is returned.

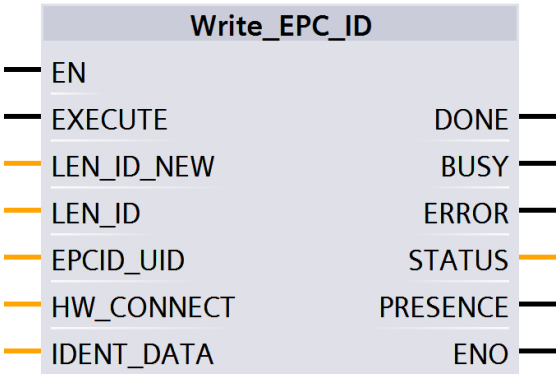


Figure 10-35 "Write_EPC_ID" block

Table 10-129 Explanation of the "Write_EPC_ID" block

Parameter	Data type	Default values	Description
LEN_ID_NEW	Byte	W#16#0C	Length of the current EPC-ID
LEN_ID	Byte	B#16#0	Length of the previous EPC-ID
EPCID_UID	Array[1...62] of Byte	0	Previous EPC ID
IDENT_DATA	Any / Variant	0	Current EPC ID Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/ UDTs can also be created.

Write_EPC_Mem

The "Write_EPC_Mem" block overwrites the EPC memory of the RF600 transponder starting at address 4. The length of the EPC memory to be overwritten is specified by the "LEN_DATA" parameter.

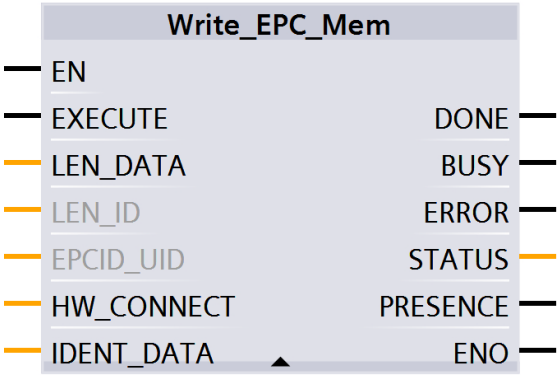


Figure 10-36 "Write_EPC_Mem" block

Table 10-130 Explanation of the "Write_EPC_Mem" block

Parameter	Data type	Default values	Description
LEN_DATA	Word	W#16#0	Length of the EPC memory to be overwritten (1 ... 62 bytes)
LEN_ID	Byte	B#16#0	Length of the EPC-ID/UID Default value: 0x00 ≙ unspecified single tag access (RF680R, RF685R)
EPCID_UID	Array[1...62] of Byte	0	Buffer for up to 62 bytes EPC-ID, 8 bytes UID or 4 bytes handle ID. • 2 - 62-byte EPC-ID is entered at the start of the buffer (length is set by "LEN_ID") • 8-byte UID is entered at the start of the buffer ("LEN_ID = 8") • 4-byte handle ID must be entered in the array element [5]-[8] ("LEN_ID = 8") Default value: 0x00 ≙ unspecified single tag access (RF620R, RF630R)
IDENT_DATA	Any / Variant	0	Data buffer with the EPC memory data to be overwritten. Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/ UDTs can also be created.

AdvancedCMD

With the "AdvancedCmd" block, every command can be executed including commands not represented by other blocks. This general structure can be used for all commands and is intended only for trained users.

This gives you the option of sending the command as a chained command. To allow this, the block provides a CMD buffer for 100 commands. All chained commands must be entered starting at the first position in the buffer. For every chained command, the "chained bit" must also be set in the CMD structure. The "chained bit" is not set in the last command in the chain. You will find further information on the "chained bit" in the section "AUTOHOTSPOT".

The entire command structure must be specified in the "CMD" input parameter. You create the structure for the "CMD" parameter in a data block.

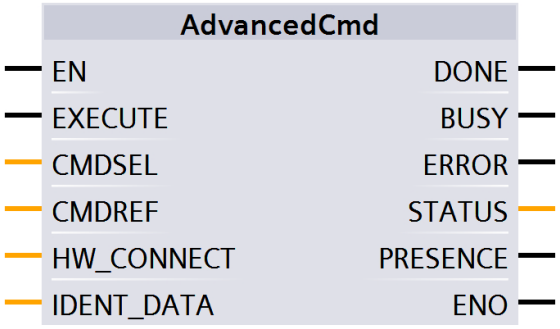


Figure 10-37 "AdvancedCmd" block

Table 10-131 Explanation of the "AdvancedCmd" block

Parameter	Data type	Default values	Description
CMDSEL	Int	1	Selection of the command to be executed "CMDREF"; 1 ⇒ 1. Command, ... The value of the "CMDSEL" parameter can never be > 100.
CMDREF	Any / Variant	--	You will find a detailed description of the parameter in the sections: • "AUTOHOTSPOT" • "AUTOHOTSPOT"
IDENT_DATA	Any / Variant	0	Buffer for data to be written or read. Note: For Variant, currently only an "Array_of_Byte" with a variable length can be created. For Any, other data types/ UDTs can also be created.

Reset blocks

The reset blocks described in this section are required when you want to operate the optical reader systems MV420, MV440 or the communications modules RF180C, ASM 456 with a SIMATIC S7-1200/S7-1500 controller. As an alternative you can also use these blocks for the RF120C if you have selected the appropriate setting in the device configuration.

In the system, these reset blocks have the same function as the "Reset_Reader" block described earlier. However, with the blocks described here, you need to set reader-dependent parameters.

Remember that the default value will be used automatically if you do not select a value manually.

Reset_MOBY_D

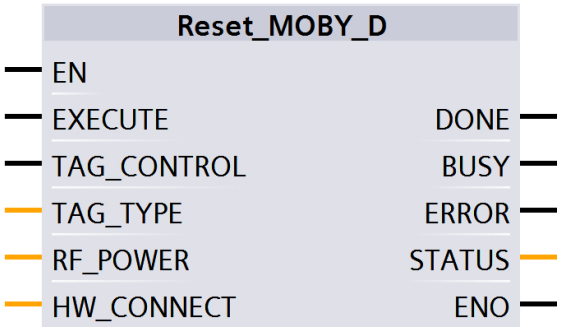


Figure 10-38 "Reset_MOBY_D" block

Table 10-132 Explanation of the "Reset_MOBY_D" block

Parameter	Data type	Default values	Description
TAG_CONTROL	Bool	True	Presence check
TAG_TYPE	Byte	1	Transponder type: • 1 = every ISO transponder
RF_POWER	Byte	0	Output power RF power from 0.5 W to 10 W in increments of 0.25 W (range of values: 0x02 ... 0x28)

Reset_MOBY_U

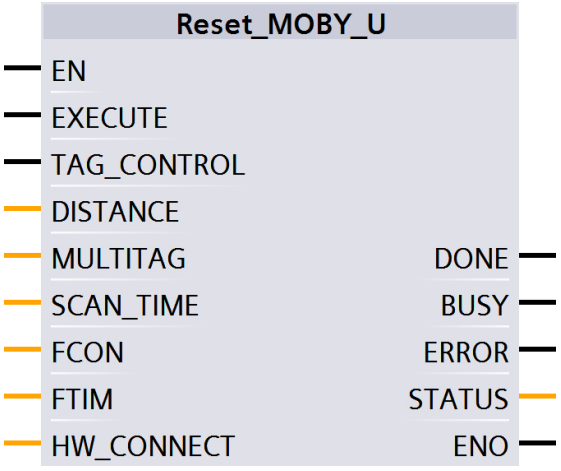


Figure 10-39 "Reset_MOBY_U" block

Table 10-133 Explanation of the "Reset_MOBY_U" block

Parameter	Data type	Default values	Description
TAG_CONTROL	Bool	True	Presence check
DISTANCE	Byte	23h	Range limitation (range of values: 0x02 ... 0x23 or 0x82 ... 0xA3 for reduced transmit power)
MULTITAG	Byte	1	Maximum number of transponders that can be processed at the same time in the antenna field. (Range of values: 0x01 ... 0x12)
SCAN_TIME	Byte	0	Scanning time: Standby time of the transponder (range of values: 0x00 ... 0xC8)
FCON	Byte	0	field_ON_control: BERO mode (range of values: 0x00 ... 0x03)
FTIM	Byte	0	field_ON_time: Time for BERO mode (range of values: 0x00 ... 0xFF)

Reset_MV

To reset cameras of the optical reader systems, call the block and activate the "EXECUTE" parameter.

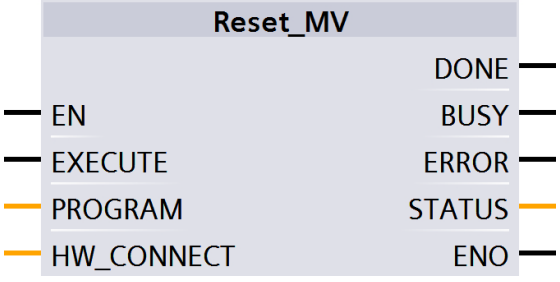


Figure 10-40 "Reset_MV" block

Table 10-134 Explanation of the "Reset_MV" block

Parameter	Data type	Description
PROGRAM	Byte	Program selection • B#16#0: Reset without program selection or in the case of diagnostics, the error code for "IN_OP = 0" is shown at the "STATUS" output parameter. • B#16#1 ... B#16#0F: Number of the program to be started ⇒ Reset with program selection (as of firmware V5.1 of the MV4x0)

Reset_RF200

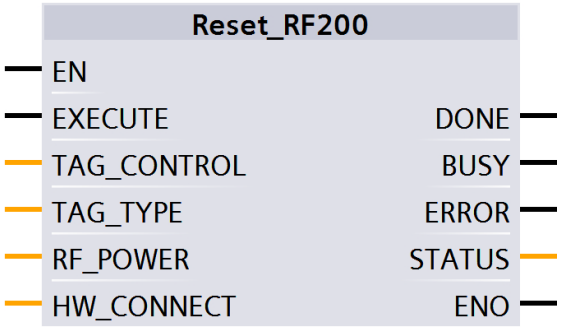


Figure 10-41 "Reset_RF200" block

Table 10-135 Explanation of the "Reset_RF200" block

Parameter	Data type	Default values	Description
TAG_CONTROL	Byte	1	Presence check
TAG_TYPE	Byte	1	Transponder type: • 1 = every ISO transponder • 3 = MDS D3xx - optimization
RF_POWER	Byte	4	Output power; only relevant for RF290R RF power from 0.5 W to 5 W in increments of 0.25 W (range of values: 0x02 ... 0x14). Default value 0x04 ≙ 1 W.

Reset_RF300

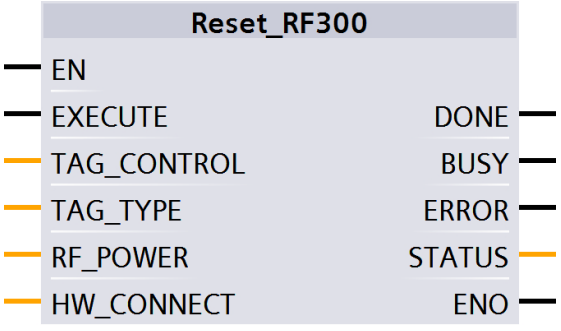


Figure 10-42 "Reset_RF300" block

Table 10-136 Explanation of the "Reset_RF300" block

Parameter	Data type	Default values	Description
TAG_CONTROL	Byte	1	Presence check • 0 = Off • 1 = on • 4 = presence (antenna is off. The antenna is turned on only when a Read or Write command is sent.)
TAG_TYPE	Byte	0	Transponder type: • 1 = every ISO transponder • 0 = RF300 transponder
RF_POWER	Byte	0	Output power; only relevant for RF380R RF power from 0.5 W to 2 W in increments of 0.25 W (range of values: 0x02 ... 0x08). Default value 0x00 ≙ 1.25 W.

Reset_RF600

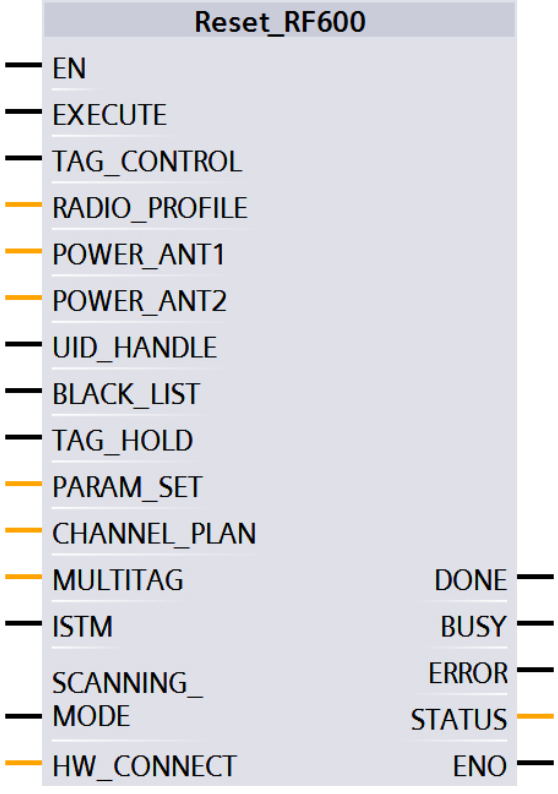


Figure 10-43 "Reset_RF600" block

Table 10-137 Explanation of the "Reset_RF600" block

Parameter	Data type	Default values	Description
TAG_CONTROL	Bool	True	Presence check
RADIO_PROFILE	Byte	1	Scanning time: Wireless profile according to EPC Global (range of values: 0x01 ... 0x09 depending on the reader variant)
POWER_ANT1	Byte	0	Transmit power for antenna 1 or internal antenna (range of values: 0x00 ... 0x0F)
POWER_ANT2	Byte	0	Transmit power for antenna 2 or external antenna (range of values: 0x00 ... 0x0F)
UID_HANDLE	Bool	False	Meaning of the UID in the command: True = Handle ID, only the least significant 4 bytes of the UID are evaluated; False = UID/EPC-ID with a length of 8 bytes
BLACK_LIST	Bool	False	True = activate black list
TAG_HOLD	Bool	False	True = activate Tag Hold
PARAM_SET	Byte	0	Field_ON_Control (0 = fast; range of values: 0x00, 0x02)
CHANNEL_PLAN	Byte	0F	Field_ON_Time (range of values: 0x00 ... 0x0F; ETSI only)
MULTITAG	Byte	1	Maximum number of transponders that can be processed at the same time in the antenna field. (Range of values: 0x01 ... 0x50)
ISTM	Bool	False	True = activate intelligent single tag mode
SCAN-NING_MODE	Bool	False	True = activate scanning mode [1]

[1] Is not currently possible with the Ident blocks.

Reset_Univ

The "Reset_Univ" block is a universal reset block with which all identification systems can be reset. Use this block only after consulting Support.

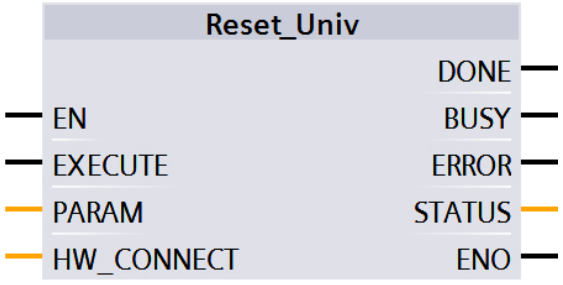


Figure 10-44 "Reset_Univ" block

Table 10-138 Explanation of the "Reset_Univ" block

Parameter	Data type	Description
PARAM	Array [1...16] of Byte	Data for reset frame The data to be set here can be made available by Support when necessary for special settings.

Table 10-139 Structure of the "PARAM" parameter

Byte	1	2...5	6	7...8	9	10	11	12	13...14	15	16
Value	04h	0	0Ah	0	scan-ning_ time	param	op-tion_ 1	dis-tance_ limiting	Num-ber of trans-pond-ers	field_ on_ control	field_ on_ time

Status blocks

Reader_Status

The "Reader_Status" block reads status information from the reader. For the various reader families, there are different status modes that you can select using the "ATTRIBUTE" parameter.

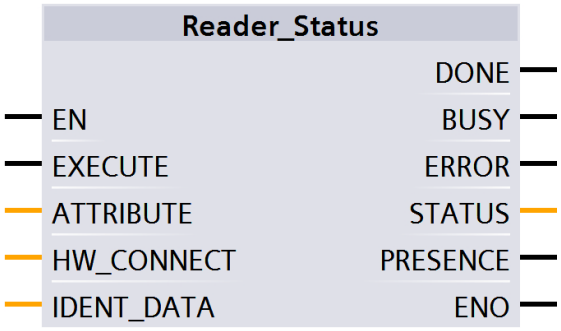


Figure 10-45 "Reader_Status" block

Table 10-140 Explanation of the "Reader_Status" block

Parameter	Data type	Default values	Description
ATTRIBUTE	Byte	B#16#81	Identifier of the status modes / possible entries: • RF200: 0x81 • RF300: 0x81, 0x86, 0xEF • RF620R, RF630R: 0x87, 0x88, 0xA0, 0xA1 • RF680R, RF685R: 0x89 • MOBY U: 0x81, 0x84, 0x85 • MOBY D: 0x81
IDENT_DATA	Any / Variant	0	Event values depending on attributes Note: For Variant, an "Array_of_Byte" with a variable length can be created. For Any, other data types/UDTs can also be created.

Results

Apply the correct data type that is assigned to the ATTRIBUTE value at the "IDENT_DATA" input of the block so that the data can be correctly interpreted.

Note that the UDTs can only be used when the blocks "Reader_Status" or "Tag Status" used.

Table 10-141 ATTRIBUTE "0x81" (data type "IID_READSTAT_81_RF2_3_U")

Name	Type	Comment
status_info	BYTE	SLG status mode
hardware	CHAR	Type of hardware
hardware_version	WORD	Version of hardware
loader_version	WORD	Version of loader
firmware	CHAR	Type of firmware
firmware_version_HB	BYTE	Version of firmware
firmware_version_LB	BYTE	
driver	CHAR	Type of driver
driver_version	WORD	Version of driver
interface	BYTE	Type of interface (RS 232/RS 422)
baud	BYTE	Baudrate
reserved1	BYTE	Reserved
reserved2	BYTE	Reserved
reserved3	BYTE	Reserved
distance_limiting_SLG	BYTE	Distance limiting of SLG
multitag_SLG	BYTE	Multitag SLG
field_ON_control_SLG	BYTE	Field ON control
field_ON_time_SLG	BYTE	Field On time
sync_SLG	BYTE	Synchronization with SLG
status_ant	BYTE	Status of antenne
stand_by	BYTE	Time of standby after command
MDS_control	BYTE	Presence mode

Table 10-142 ATTRIBUTE "0x84" (data type "IID_READSTAT_84_MOBY_U")

Name	Type	Comment
status_info	BYTE	SLG status mode
number_MDS	BYTE	Range 1..24
UID	ARRAY [1..24] of DWord	

Table 10-143 ATTRIBUTE "0x86" (data type "IID_READSTAT_86_RF300")

Name	Type	Comment
status_info	BYTE	SLG status mode
FZP	BYTE	Error counter passive: distortion without communication
ABZ	BYTE	Dropout counter

Name	Type	Comment
CFZ	BYTE	Code error counter
SFZ	BYTE	Signature error counter
CRCFZ	BYTE	CRC-error counter
BSTAT	BYTE	Status of last command
ASMFZ	BYTE	Error counter for host interface (ASM)
reserved0	ARRAY [1..17]	

Table 10-144 ATTRIBUTE "0x87" (data type "IID_READSTAT_87_RF600")

Name	Type	Comment
status_info	BYTE	SLG status mode
hardware	CHAR	Type of hardware
hardware_version	WORD	Version of hardware
reserved0	WORD	
firmware	CHAR	Type of firmware
firmware_version_HB	BYTE	Version of firmware highbyte
firmware_version_LB	BYTE	Version of firmware lowbyte
driver	CHAR	Type of driver
current_time_hour	BYTE	Hours [1)]
current_time_min	BYTE	Minutes
current_time_sec	BYTE	Seconds
reserved1	BYTE	
SLG_version	BYTE	SLG version
baud	BYTE	Baudrate
reserved2	BYTE	
distance_limiting_SLG	BYTE	Selected transmit power
multitag_SLG	BYTE	Multitag SLG
field_ON_control_SLG	BYTE	Selected comunication typ
field_ON_time_SLG	BYTE	Selected channel
expert_mode	BYTE	Expert mode
status_ant	BYTE	Status of antenna [2)]
scanning_time_SLG	BYTE	Radio communication profile (country specific radio standart)
MDS_control	BYTE	Presence mode

1) The internal time stamp of the reader that relates to this event is output. The internal reader time stamp is not synchronized with UTC.

2) The antenna status relates to the "ATTRIBUTE" (bits 0 and 1) of the last executed "SET-ANT" or to the default value set by "init-run". In "init_run" of the RF620R, the default value is "1" (int. antenna on), with the RF630R, it is "3" (antennas 1 and 2 on).

Table 10-145 ATTRIBUTE "0x88" (data type "IID_READSTAT_88_RF600")

Name	Type	Comment
status_info	BYTE	SLG status mode(Subcommand)
hardware	CHAR	Type of hardware
hardware_version	WORD	Version of hardware
reserved_word1	WORD	Reserved
firmware	CHAR	Type of firmware
firmware_version_HB	BYTE	Version of firmware (High-Byte)
firmware_version_LB	BYTE	Version of firmware (Low-Byte)
driver	CHAR	Type of driver
current_time_hour	BYTE	Hours [1]
current_time_minute	BYTE	Minutes [1]
current_time_sec	BYTE	Seconds [1]
current_time_reservByte	BYTE	
SLG_version	BYTE	SLG-Version
baud	BYTE	Baudrate
reserved_byte1	BYTE	Reserved
distance_limiting_SLG	BYTE	Selected transmit power
multitag_SLG	BYTE	Multitag SLG
field_ON_control_SLG	BYTE	Selected communication type
field_ON_time_SLG	BYTE	Selected channel
expert_mode	BYTE	Expert mode
status_ant	BYTE	Status of antenna [2]
scanning_time_SLG	BYTE	Radio communication profile (country specific radio standart)
MDS_control	BYTE	Presence mode
blink_pattern	BYTE	Blink Pattern
act_algor_Single_Tag	Bool	Single_Tag [1]
act_algor_ITF_Phase2	Bool	ITF_Phase2 [2]
act_algor_ITF_Phase1	Bool	ITF_Phase1 [3]
act_algor_Smoothing	Bool	Smoothing [4]
act_algor_Blacklist	Bool	Blacklist [5]
act_algor_RSSI_Threshold	Bool	RSSI_Threshold [6]
act_algor_Power_Ramp	Bool	Power_Ramp [7]
act_algor_Power_Gap	Bool	Power_Gap [8]
Reserved1	Bool	Reserved1 [1]
Reserved2	Bool	Reserved2 [2]
Reserved3	Bool	Reserved3 [3]
Reserved4	Bool	Reserved4 [4]
act_algor_EPC_MemBankFilter	Bool	EPC_MemBankFilteres [5]
act_algor_Tag_Holg	Bool	Tag_Hold [6]
act_algor_Multi_Tag	Bool	Multi_Tag [7]
act_algor_ISTM	Bool	ISTM [8]

Name	Type	Comment
reserved_word2	WORD	Reserved
reserved_word3	WORD	Reserved
reserved_word4	WORD	Reserved
filtered_max_rssi	BYTE	Maximum RSSI value of a tag, of all filtered tags
reserved_byte2	BYTE	Reserved
filtered_tags_rssi	BYTE	Number of tags, filtered out by the RSSI threshold
reserved_byte3	BYTE	Reserved
filtered_tags_black_list	WORD	Number of tags, filtered out via Black-List
filtered_tags_epc_data	WORD	Number of tags, filtered out via EPC Data Filter
filtered_tags_smoothing	WORD	Number of tags in Tag List of status Not-Observed
itf_ph1_max_detect	WORD	Number of reads of a Tag, filtered out via ITF-phase 1
itf_ph1_tags_detect	WORD	Number of tags, filtered out via ITF-phase 1
itf_ph2_max_detect	WORD	Number of reads of a Tag, filtered out via ITF-phase 2
itf_ph2_tags_detect	WORD	Number of tags, filtered out via ITF-phase 2
filtered_istm_min_dist	WORD	Minimum distance of tags according to sorting criterion of ISTM
filtered_istm_tags	WORD	Number of tags, filtered out via ISTM algorithm
last_error	BYTE	error code of the last occuring error (last_command)
reserved_byte4	BYTE	Reserved
error_command1	WORD	Last command (has lead to error code) "last_error"
error_command2	WORD	Last command (has lead to error code) "last_error"
error_command3	WORD	Last command (has lead to error code) "last_error"
reserved_word5	WORD	Reserved
reserved_array_byte	ARRAY[1..30] of Byte	

Table 10-146 ATTRIBUTE "0x89" (data type "IID_READSTAT_89_RF68xR")

Name	Type	Comment
status_info	BYTE	SLG status mode(Subcommand)
hardware_version	BYTE	Version of hardware
firmware_version	ARRAY[1..4] of CHAR	Version of firmware
config ID	DWORD	Unix timestamp
inventory_status	WORD	0=inventory not active; 1=inventory active; 2=presence mode active
sum_of_filtered_tags	WORD	All filtered Tags
filtered_smoothing	WORD	Filtered Tags trough Smoothing
filtered_blacklist	WORD	Filtered Tags trough Blacklist
filtered_data-filter	WORD	Filtered Tags trough Data-Filter

Name	Type	Comment
filtered_RSSI_threshold	WORD	Filtered Tags trough RSSI Threshold
filtered_RSSI_delta	WORD	Filtered Tags trough RSSI Delta

Table 10-147 ATTRIBUTE "0xA0" and "0xA1" (data type "IID_READSTAT_A0_A1_RF600")

Name	Type	Comment
reserved	BYTE	
Status_info	BYTE	Status-Info, SLG-Status SubCommand 20/21
number_tags_frame	BYTE	Number of Tags in this frame
number_tags_next_frames	BYTE	Number of Tags in the next frames
reserved_byte1	BYTE	Reserved
reserved_byte2	BYTE	Reserved
reserved_byte3	BYTE	Reserved
reserved_byte4	BYTE	Reserved
reserved_byte5	BYTE	Reserved
reserved_byte6	BYTE	Reserved
Black_List_ID	ARRAY[1..13] of "IID_IN_Blacklist"	EPC-ID Length
Black_List_ID[1]	"IID_IN_Blacklist"	
EPC_Length	BYTE	EPC-ID Length
Antenna	BYTE	Antenna = Default 3
Filtered_Tag	WORD	Number of times - EPC-ID filtered out via BlackList
EPC	ARRAY[1..12] of Byte	EPC-ID
Black_List_ID[2]	"IID_IN_Blacklist"	
Black_List_ID[3]	"IID_IN_Blacklist"	
Black_List_ID[4]	"IID_IN_Blacklist"	
Black_List_ID[5]	"IID_IN_Blacklist"	
Black_List_ID[6]	"IID_IN_Blacklist"	
Black_List_ID[7]	"IID_IN_Blacklist"	
Black_List_ID[8]	"IID_IN_Blacklist"	
Black_List_ID[9]	"IID_IN_Blacklist"	
Black_List_ID[10]	"IID_IN_Blacklist"	

Name	Type	Comment
Black_List_ID[11]	"IID_IN_Blacklist"	
Black_List_ID[12]	"IID_IN_Blacklist"	
Black_List_ID[13]	"IID_IN_Blacklist"	

You will find more detailed information on the individual status modes in the manuals matching the modes "FB 45", "FB55" and "SIMATIC RF620R/RF630R".

The identifiers of the status modes correspond to the following identifiers in the other manuals:

0x81 ≙ 0x01
0x82 ≙ 0x02
0x83 ≙ 0x03
0x85 ≙ 0x05
0x87 ≙ 0x07
0x88 ≙ 0x08
0x90 ≙ 0x10
0x91 ≙ 0x11
0x92 ≙ 0x12
0xA0 ≙ 0x20
0xA1 ≙ 0x21

Tag_Status

The "Tag_Status" block reads the status information of the transponder. For the various transponder types and reader families, there are different status modes that you can select using the "ATTRIBUTE" parameter.

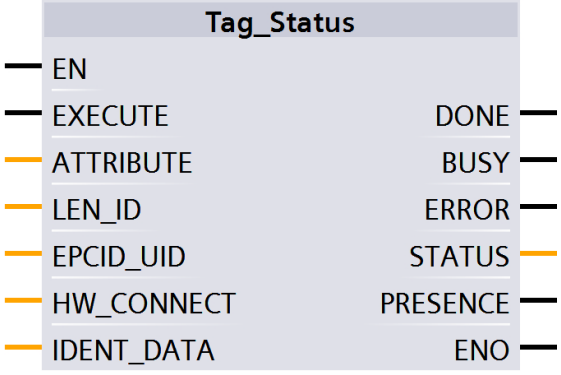


Figure 10-46 "Tag_Status" block

Table 10-148 Explanation of the "Tag_Status" block

Parameter	Data type	Default values	Description
ATTRIBUTE	Byte	B#16#0	Identifier of the status modes / possible entries: • RF200: 0x83 • RF300: 0x04, 0x82, 0x83 (only ISO transponders) • RF620R, RF630R: 0x84, 0x85 • MOBY D: 0x83 [1)] • MOBY U: 0x80
LEN_ID	Byte	B#16#0	Length of the EPC-ID/UID Default value: 0x00 ≙ unspecified single tag access (RF680R, RF685R)
EPCID_UID	Array[1...62] of Byte	0	Buffer for up to 62 bytes EPC-ID, 8 bytes UID or 4 bytes handle ID. • 2 - 62-byte EPC-ID is entered at the start of the buffer (length is set by "LEN_ID") • 8-byte UID is entered at the start of the buffer ("LEN_ID = 8") • 4-byte handle ID must be entered in the array element [5]-[8] ("LEN_ID = 8") Default value: 0x00 ≙ unspecified single tag access (RF620R, RF630R)
IDENT_DATA	Any / Variant	0	Event values depending on attributes Note: For Variant, an "Array_of_Byte" with a variable length can be created. For Any, other data types/UDTs can also be created.

1) SLG D10S only

Results

Note that the UDTs can only be displayed when the blocks "Reader_Status" or "Tag_Status" are used.

Table 10-149 ATTRIBUTE "0x04" ("IID_TAG_STATUS_04_RF300" data type)

Name	Type	Comment
reserved	BYTE	
status_info	BYTE	MDS status mode
UID	ARRAY [1..8] of BYTE	
MDS_type	BYTE	Type of MDS

Name	Type	Comment
Lock_state	BYTE	Write Protection Status EEPROM
Reserved1	ARRAY[1..6] of BYTE	

Table 10-150 ATTRIBUTE "0x80" ("IID_TAG_STATUS_80_MOBY_U" data type)

Name	Type	Comment
UID	ARRAY [1..4] of BYTE	Unique indentifier (MDS-Number)
MDS_type	BYTE	Type of MDS
sum_subframe_access_1	BYTE	Sum of subframe access Byte 1
sum_subframe_access_2	BYTE	Sum of subframe access Byte 2
sum_subframe_access_3	BYTE	Sum of subframe access Byte 3
sum_subframe_access_4	BYTE	Sum of subframe access Byte 4
sum_searchmode_access_1	BYTE	Sum of search mode access Byte 1
sum_searchmode_access_2	BYTE	Sum of search mode access Byte 2
ST_date_Week	BYTE	Date of last sleep-time change (week of year)
ST_date_Year	BYTE	Date of last sleep-time change (year)
battery_left_1	BYTE	Battery power left (percent) Byte 1
battery_left_2	BYTE	Battery power left (percent) Byte 2
ST	BYTE	Actual sleep-time on MDS

Table 10-151 ATTRIBUTE "0x82" ("IID_TAG_STATUS_82_RF300" data type)

Name	Type	Comment
reserved	BYTE	
status_info	BYTE	MDS status mode
UID	ARRAY [1..8] of BYTE	
LFD	BYTE	Magnetic flux density: correlation between limit-value
FZP	BYTE	Error counter passive: distortion without communication
FZA	BYTE	Error counter active: distortion during communication
ANWZ	BYTE	Presence counter: measure value for presence time
reserved1	ARRAY [1..3] of BYTE	

Table 10-152 ATTRIBUTE "0x83" ("IID_TAG_STATUS_83_ISO" data type)

Name	Type	Comment
reserved	BYTE	
status_info	BYTE	MDS status mode
UID	ARRAY [1..8] of BYTE	
MDS_Type	BYTE	Type of MDS
IC_version	BYTE	Chip version
size_HB	BYTE	Size of Memory (high Byte)
size_LB	BYTE	Size of memory (low Byte)
lock_state	BYTE	Write protection status EEPROM
block_size	BYTE	Size of a block in addressable memory
number_of_block	BYTE	Number of blocks in addressable memory

Table 10-153 ATTRIBUTE "0x84" ("IID_TAG_STATUS_84_RF600" data type)

Name	Type	Comment
reserved	BYTE	
status_info	BYTE	MDS status mode
UID	ARRAY [1..8] of BYTE	
antenna	BYTE	Antenna which has observed the MDS
RSSI	BYTE	RSSI value
last_observed_hour	BYTE	Last observed time hour
last_observed_min	BYTE	Last observed time minute
last_observed_sec	BYTE	Last observed time seconds
last_observed_channel	BYTE	Last observed time channel
EPC_length	BYTE	EPC-Length
reserved1	BYTE	

[1] The internal time stamp of the internal reader clock that relates to this event is output. The internal reader clock is not synchronized with UTC.

Table 10-154 ATTRIBUTE "0x85" ("IID_TAG_STATUS_85_RF600" data type)

Name	Type	Comment
status_info	BYTE	MDS status mode
antenna	BYTE	Antenna which has observed the MDS
channel	BYTE	Channel
UID	ARRAY [1..8] of BYTE	

Name	Type	Comment
DT_glimpsed_1	BYTE	Time elasped between acknowledgement and first read in [ms]1 Highbyte
DT_glimpsed_2	BYTE	Time elasped between acknowledgement and first read in [ms]2
DT_glimpsed_3	BYTE	Time elasped between acknowledgement and first read in [ms]3
DT_glimpsed_4	BYTE	Time elasped between acknowledgement and first read in [ms]4 Low-Byte
reserved1	BYTE	
reserved2	BYTE	
reserved3	BYTE	
reserved4	BYTE	
last_observed_hour	BYTE	Last observed time hour
last_observed_min	BYTE	Last observed time minutes [1]
last_observed_sec	BYTE	Last observed time seconds [1]
last_observed_EPC_length	BYTE	Last observed time EPC length
EPC_ID_Byte	ARRAY [1..62] of BYTE	EPC-ID
reads_HB	BYTE	Number of Reads of MDS in Inventory (1 - 65535)
reads_LB	BYTE	Number of Reads of MDS in Inventory (1 - 65535)
RSSI	BYTE	Current RSSI value of MDS [2]
mean_RSSI	BYTE	Mean RSSI value of MDS
max_RSSI	BYTE	Max RSSI value of MDS
min_RSSI	BYTE	Min RSSI value of MDS
min_POWER	BYTE	Min Power value of MDS
current_POWER	BYTE	Current Power value of MDS [3]
reserved5	ARRAY[1..137] of BYTE	

[1] The internal time stamp of the reader that relates to this event is output. The internal reader time stamp is not synchronized with UTC.

[2] The value "Reads" indicates the total transponder recognitions (inventories) regardless of the set smoothing parameters. In this way, in extreme situations, the "Reads" counter can reach extremely high values without the transponder ever reaching the "Observed" status.

[3] The "current_Power" value is specified as transmit power in 0.25 dBm steps (ERP/EIRP). A "current_Power" value of "72" (0x48) therefore corresponds to 18 dBm (ERP/EIRP).

You will find more detailed information on the individual status modes in the manuals matching the modes "FB 45", "FB55" and "SIMATIC RF620R/RF630R".

The identifiers of the status modes correspond to the following identifiers in the other manuals:

0x04 ≙ 0x01
0x82 ≙ 0x02
0x83 ≙ 0x03
0x84 ≙ 0x04
0x85 ≙ 0x05

Programming the Ident profile

Structure of the Ident profile

Note

Parallel operation using Ident blocks and Ident profile is not possible

Note that the CM or reader cannot be operated at the same time using the Ident blocks and the Ident profile.

The blocks described in the section"AUTOHOTSPOT" represent a simplified interface of the Ident profile. If the functionality available with the blocks is not adequate for your application, you can use the Ident profile as an alternative. Using the Ident profile, you can set complex command structures and work with command repetition. The following graphic shows the Ident profile including the commands that can be implemented with it.

Note

Ident profile for trained users

The Ident profile is a complex block containing all the functionality of the Ident blocks. The Ident profile was developed specially for trained block users who want to configure complex functions with their own blocks. For untrained users, we recommend using the Ident blocks.

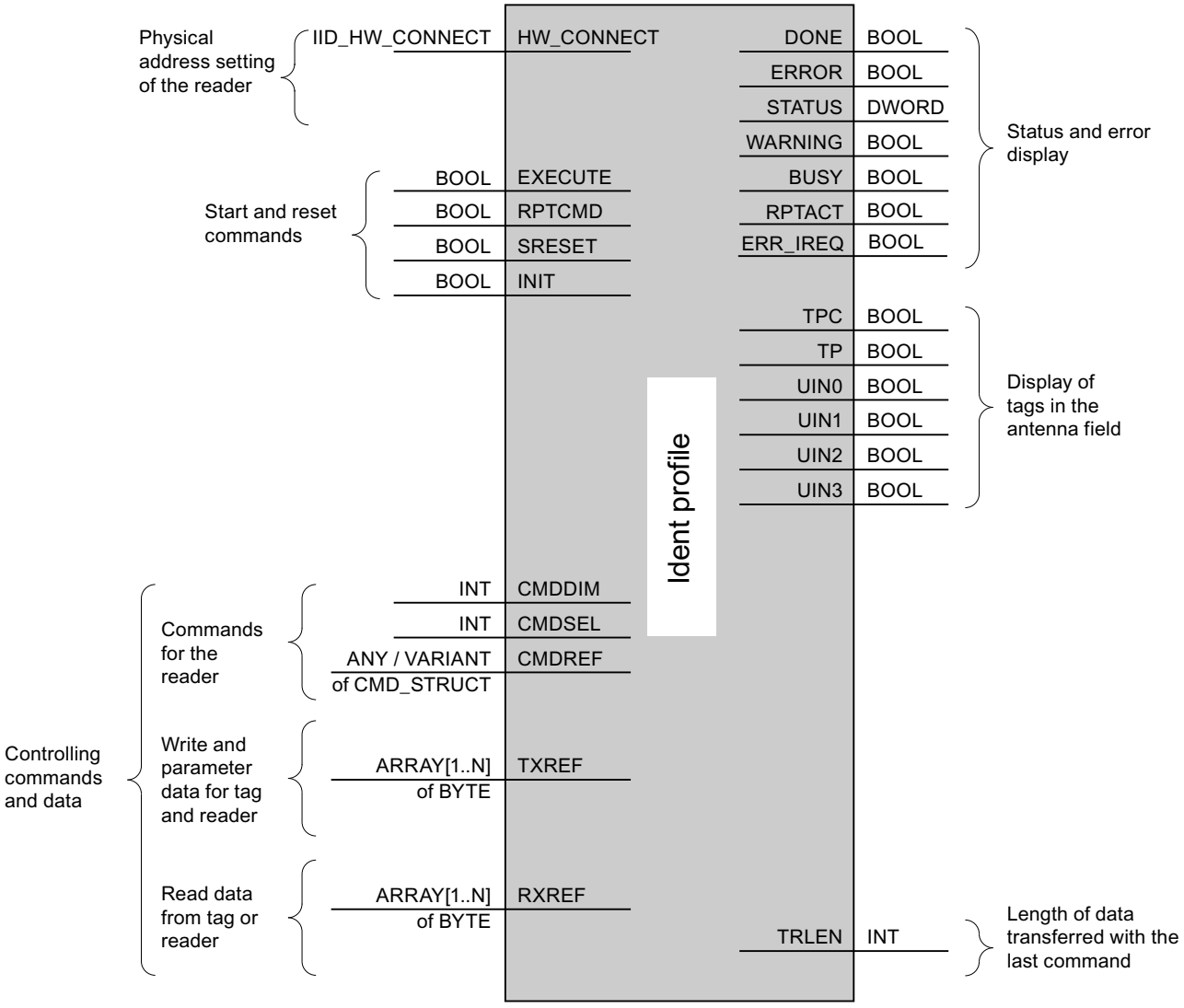


Figure 10-47 The input parameters of the Ident profile

Note

Working with multiple channels

If you work with several channels, you must ensure that for each channel, the block is called with a separate instance DB.

Interface description

Table 10-155 Input parameter

Input parameter	Data type	Default value	Description
HW_CONNECT	HW_CONNECT	--	Own data type for physical addressing of communications modules and readers and for synchronizing the blocks used for each reader. The addressing is as described in the section "AUTOHOTSPOT".
EXECUTE	BOOL	FALSE	TRUE = triggers a new command Before starting you need to set the command and the corresponding parameters in the memory linked to "CMDREF".
RPTCMD	BOOL	FALSE	TRUE = Repeating the command currently being executed or the next command to be executed by communications module
SRESET	BOOL	FALSE	TRUE = Cancellation of the command currently processed on the communications module
INIT	BOOL	FALSE	TRUE = Communications module executes a Reset and is re-assigned parameters
CMDDIM	INT	10	Number of commands in the parameter "CMDREF"
CMDSEL	INT	0	Selection of the command to be executed "CMDREF"; 1 ⇒ 1. command, ... The value of the "CMDSEL" parameter can never be higher than the value of the "CMDDIM" parameter.
CMDREF	ARRAY[1...10] of CMD_STRUCT	--	Command field The field can hold up to 10 commands. The commands are complex variables of the type "CMD_STRUCT". You will find more information on "CMDREF" in the section "AUTOHOTSPOT".
TXREF	ARRAY[1...n] of BYTE	--	Reference to global memory area for send data. The memory area can be shared with other block instances. The value "n" of the individual blocks is variable and can be up to 32 KB in size.
RXREF	ARRAY[1...n] of BYTE	--	Reference to global memory area for receive data. The memory area can be shared with other block instances. The value "n" of the individual blocks is variable and can be up to 32 KB in size.

Table 10-156 Output parameter

Output parameter	Data type	Default value	Description
DONE	BOOL	FALSE	TRUE = Command was executed free of errors.
ERROR	BOOL	FALSE	TRUE = Error was detected. The error is output in the "STATUS" parameter. The bit is reset automatically when a new command is started.

Output parameter	Data type	Default value	Description
STATUS	DWORD	FALSE	Warning and error If "ERROR = TRUE" or "WARNING = TRUE", the error or warning information is contained in the "STATUS" parameter. For more information, refer to the section "AUTOHOTSPOT".
WARNING	BOOL	FALSE	TRUE = Warning was detected. The warning is output in the "STATUS" parameter. If the "ERROR" parameter is not set at the same time, the data was correctly processed. The bit is reset automatically when a new command is started.
BUSY	BOOL	FALSE	TRUE = the block is executing a command. Other commands except for "INIT" and "SRESET" cannot be started.
RPTACT	BOOL	FALSE	TRUE = "RPTCMD" is active The acknowledgement bit shows that the "Repeat mode" of the CM/reader is active.
ERR_IREQ	BOOL	FALSE	TRUE = An error has occurred on the communications module or reader (e.g. at power-up or connection termination)
TPC	BOOL	FALSE	Transponder Presence Changed (only when the Presence_Mode is active) TRUE = A new transponder enters the antenna field of the reader or a transponder has left the antenna field. The parameter is set to "FALSE" after the successful execution of the next "INVENTORY" (0x80, 0x81, 0x87) or "INIT" command.
TP	BOOL	FALSE	Transponder Presence TRUE = There is a transponder in the antenna field of the reader.
UIN0	BOOL	FALSE	With RFID readers, the number of transponders in the antenna field is indicated. With optical reader devices, the various statuses of the reader device are displayed. UIN0: Corresponds to IN_OP bit of the reader device UIN1: Corresponds to RDY bit of the reader UIN2 + UIN3: These two bits are interpreted as an unsigned value (bit 2 is the less significant bit) that represents the number of available decoded codes. If the value is = 3, three or more decoded codes are available.
UIN1	BOOL	FALSE	
UIN2	BOOL	FALSE	
UIN3	BOOL	FALSE	
TRLEN	INT	0	Number of data elements received after successful execution of the command.

Data structure of the Ident profile

Each time the Ident profile is called, you need to supply the parameters ("HW_CONNECT", "CMDREF", "TXREF" and "RXREF") with values as described in section "Structure of the Ident profile (Page 1870)".

The call for the Ident profile is always via the input parameter "HW_CONNECT" and the "IN/OUT" parameters "CMDREF", "TXREF" and "RXREF". All three parameters need to be created

in a data block. The relationship between the three "IN/OUT" parameters is described in greater detail below:

- CMDREF (command buffer):
  Array[1...10] of CMD_STRUCT
- TXREF (send buffer):
  Array[1...n] of Byte
- RXREF (receive buffer):
  Array[1...n] of Byte

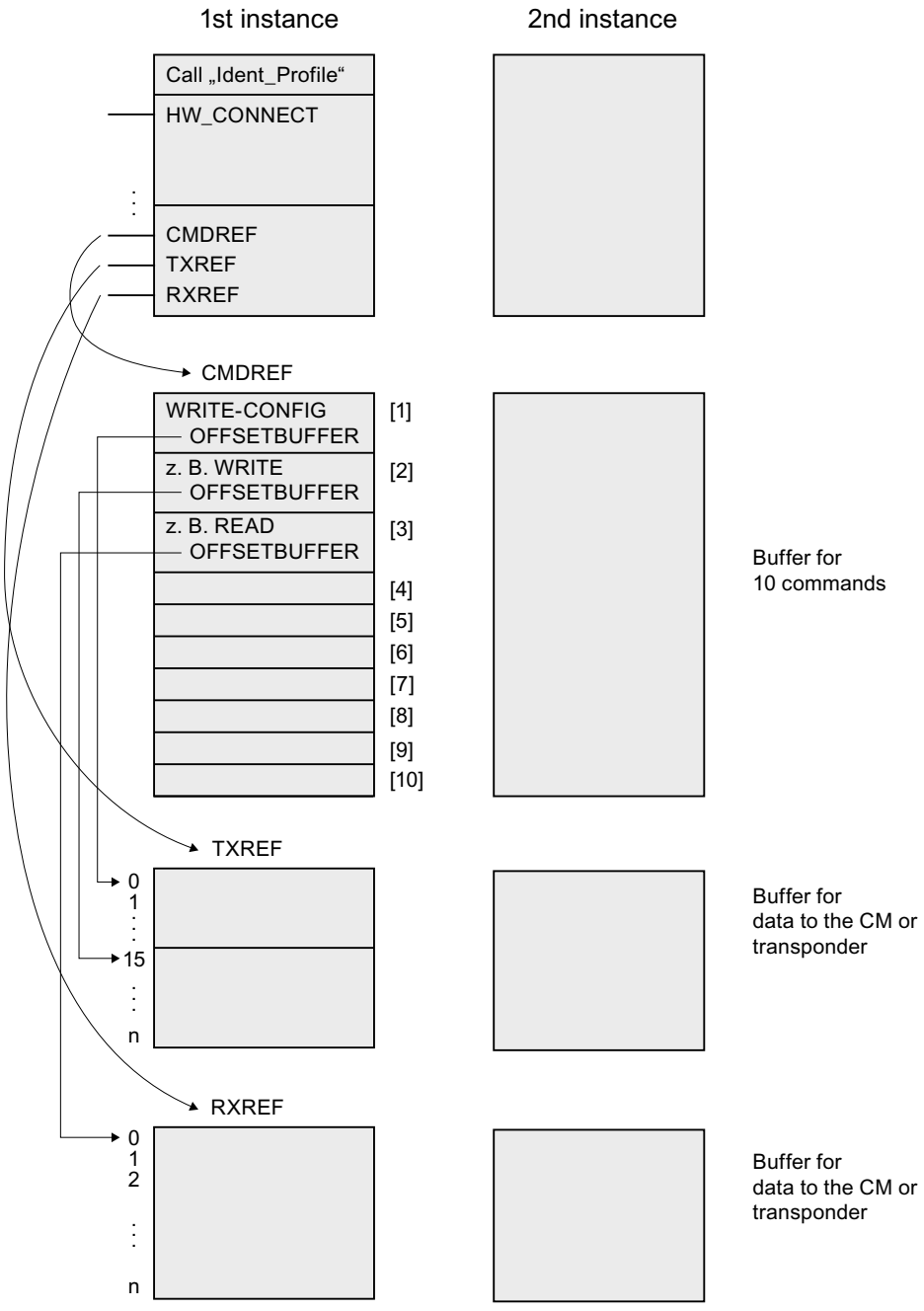


Figure 10-48 Data structure example of the Ident profile

Explanation of the data structure example

- CMDREF[1]:
  Command "WRITE-CONFIG", OFFSETBUFFER = 0
  At the CMDREF[1] point you need to set the "WRITE-CONFIG" command so that the "INIT/ Reset" is correctly executed.
- CMDREF[2]:
  Command "WRITE", OFFSETBUFFER = 15
- CMDREF[3]:
  Command "READ", OFFSETBUFFER = 0

If the "CMDREF[2]" command is selected, a write command is started and the data to be written is fetched starting at byte 15 of the "TXREF" parameter. If the "CMDREF[3]" command is selected, the read data is stored starting at byte 0 in the "RXREF" parameter.

Communications modules

Communications module RF120C

"Reader" parameter group

The "Reader" parameter group contains the following:

- Diagnostics: Setting to decide whether or not hardware diagnostics messages are output.
- Connected device: Selection of the connected Ident device / system. Depending on the selection you make, the "Ident system" parameter group is adapted.

"Diagnostics" parameter

Parameter assignment options:

- None
  Apart from standard diagnostics messages, no other diagnostics messages are generated.
- Hard Errors
  Extended diagnostics messages are generated if the following events occur.
  - Hardware error (memory test)
  - Firmware error (checksum)
  - Connection to reader lost
  - Short-circuit fault/interruption if supported by the hardware

Further information

For further information on diagnostics, refer the documentation for the RF120C communications module with application blocks for S7-1200 and S7-1500.

"Ident system" parameter group

In the "ID system" parameter group the parameters for the selected identification system are set.

The table shows the parameters that exist with all identification systems (RFID and optical reading systems).

Table 10-157 Standard parameters for all identification systems

Parameter	Parameter value	Default value	Description
Transmission speed	19.2 kBd 57.6 kBd 115.2 kBd	115.2 kBd	After changing the transmission speed, the reader must be turned off and on again.
Presence check	On Off (RF field off) Off (RF field on)	On	On = presence is reported as soon as there is a transponder in the antenna field of the reader Off (RF field on) = the presence check in the FB is suppressed. The antenna on the reader is nevertheless turned on as long as it has not been turned off by a command. Off (RF field off) = the antenna is turned on only when a command is sent and it then turns itself off again (RF300 only)
Reset error LED	On Off	Off	On = the flashing of the error LED on the RF120C is reset by each FB reset. Off = the error LED always indicates the last error. The display can only be reset by turning off the RF120C.

The following parameters are system specific according to the selection you made in the "Reader" parameter group.

RF200

Table 10-158 RF200 general

Parameter	Parameter value	Default value	Description
-	-	-	Only the standard parameters are available.

Table 10-159 RF290R

Parameter	Parameter value	Default value	Description
RF power	0.50 - 5.00 W	1.00 W	Setting for the output power of the reader.

RF300

Table 10-160 RF300 general

Parameter	Parameter value	Default value	Description
Transponder type	RF300 ISO 15693	RF300	Selection of the transponders used.

Table 10-161 RF380R

Parameter	Parameter value	Default value	Description
RF power	0.50 - 2.00 W	1.25 W	Setting for the output power of the reader.
Transponder type	RF300 ISO 15693	RF300	Selection of the transponders used.

RF600

Table 10-162 Ident system: RF600

Parameter	Parameter value	Default value	Description
Max. no. of transponders	1 - 80	1	Number of transponders expected in the antenna field. Permitted values: • 1 - 40 for RF620R • 1 - 80 for RF630R with 2 antennas • 1 - 40 for RF630R with 1 antenna The value stored in "multitag" defines the expected number of transponders to be read (EPC-ID) in the inventory. The value does not restrict the number of transponders to be processed in the antenna field. To ensure an efficient inventory of transponders in the antenna field, make sure that the values specified here do not deviate by more than approximately 10% from the maximum number of transponders to be expected in the antenna field.
Reader type	RF620R ETSI RF620R FCC RF620R CMIIT RF630R ETSI RF630R FCC RF630R CMIIT	RF620R ETSI	Selection of the reader used. By selecting a reader, you open the "Reader type" parameter group. The parameters are described in the following table.

Reader type (RF600 only)

In the parameter group "Reader type", further specific parameters are set for the reader type selected in the "ID system" parameter group.

Table 10-163 RF600 menu: Reader type

Parameter	Parameter value	Default value	Description
Wireless profile	-	-	Selection of the relevant wireless profile for ETSI, FCC or CMIIT.
Multitag mode	UID = EPC-ID (8 bytes) UID = Handle ID (4 bytes)	UID = EPC-ID (8 bytes)	• UID = EPC-ID (8 bytes) 8 byte UID of the bytes 5-12 of the 12 byte long EPC-ID • UID = handle ID (4 bytes) 4 byte UID as handle ID for access to transponders with an EPC-ID of any length
Intelligent Single Tag Mode (ISTM)	On Off	Off	Enable/disable the "Intelligent Single Tag Mode ISTM" algorithm [1]
Black List	On Off	Off	Enable/disable the "Black list" [1]
Radiated power internal antenna (RF620R)	0 - B	4	Setting the radiated power for the internal antenna [1] [2]
Internal antenna (RF620R)	-	-	Enable/disable the internal antenna. With the RF620R, either only the internal or only the external antenna can be set.
Transmit power (external antenna RF620R)	0 - 9	4	Setting the transmit power for the external antenna [1] [2]
External antenna (SetAnt block required) (RF620R)	-	-	Enable/disable the external antenna. With the RF620R, either only the internal or only the external antenna can be set.
Transmit power ANT 1 (RF630R)	0 - 9	4	Setting the transmit power for antenna 1 [1] [2]
Transmit power ANT 2 (RF630R)	0 - 9	4	Setting the transmit power for the antenna 2 [1] [2]
Communications speed	Reliable detection Fast detection	Reliable detection	[1]
Tag hold	On Off	Off	Enable/disable "Tag hold" [1]
Scanning mode	On Off	Off	Enable/disable the "Scanning mode" [1]
Channel assignment (only with wireless profile ETSI)	-	-	Selection of the wireless channels to be used [1]

[1] You will find further information in the "Configuration manual RF620R/RF630R".*

[2] The values for the transmit/radiated power of the antennas can be found in the following table.

* You will find the configuration manual RF620R/RF630R here (http://support.automation.siemens.com/WW/view/en/10805817).

Table 10-164 Transmit / radiated power of the antennas

Hex value	RF630R transmit power	RF620R radiated power (internal antenna)			RF620R transmit power
		ETSI	FCC	CMIIT	
	dBm / (mW)	dBm / (mW) ERP	dBm / (mW) EIRP	dBm / (mW) ERP	dBm / (mW)
0	18 / (63)	18 / (63)	20 / (100)	18 / (65)	18 / (63)
1	19 / (79)	19 / (79)	21 / (126)	19 / (79)	19 / (79)
...	...	...	...	...	...
4	22 / (158)	22 / (158)	24 / (251)	22 / (158)	22 / (158)
...	...	...	...	...	...
9	27 / (501)	27 / (501)	29 / (794)	27 / (501)	27 / (501)
A	27 / (501)	28 / (631)	30 / (1000)	28 / (631)	27 / (501)
B (...F)	27 / (501)	29 / (794)	31 / (1259)	29 / (794)	27 / (501)

SLG D10S

Table 10-165 SLG D10S

Parameter	Parameter value	Default value	Description
RF power	0.50 - 10.00 W	1.00 W	Setting for the output power of the reader.
Transponder type	ISO I-Code (e.g. MDS D139)	ISO	Selection of the transponders used.

SLG D11S/D12S

Table 10-166 SLG D11S/D12S

Parameter	Parameter value	Default value	Description
Transponder type	ISO I-Code (e.g. MDS D139)	ISO	Selection of the transponders used.

MOBY U

Table 10-167 MOBY U

Parameter	Parameter value	Default value	Description
Standby time	0 - 1400 ms	0 ms	Standby time (scanning_time) for the transponder. If the transponder receives a further command before the standby time has elapsed, this command can be executed immediately. If the transponder receives a command after standby time has elapsed, command execution is delayed by the "sleep_time" of the transponder.
Distance limitation	0.2 m 0.5 m 1.0 m 1.5 m 2.0 m 2.5 m 3.0 m 3.5 m	1.5 m	-
Max. no. of transponders	1 - 12	1	Maximum number of transponders that can be processed at the same time in the antenna field.

Parameter	Parameter value	Default value	Description
BERO mode	Without BEROs 1 or 2 BEROs 1st BERO on, 2nd BERO off Synchronization by cable connection	Without BEROs	• Without BEROs No reader synchronization • 1 or 2 BEROs The BEROs are logically ORed. The antenna field is turned on during the actuation of a BERO. • 1st BERO on, 2nd BERO off The 1st BERO switches the antenna field on and the 2nd BERO switches the antenna filed off. If there are two BEROs and a "BERO time in s" is set, the antenna field is automatically turned off if the 2nd BERO does not switch within this BERO time. If no "BERO time in s" is set, the antenna field remains turned on until the 2nd BERO is activated. • Synchronization by cable connection Enable reader synchronization via cable connection (see manual on configuration, installation and service for MOBY U).
BERO time in s	0 - 255 s	0 s	Can only be set if the BERO mode is set to "1st BERO on, 2nd BERO off". • 0 The time monitoring is disabled. To switch the field off, the 2nd BERO is required. • 1 ... 255 s Operating time for the reader field

General Reader

Table 10-168 General Reader

Parameter	Parameter value	Default value	Description
Input box for byte sequence (hexadecimal)	00000000	00000000	Expert mode With this function, you can specify the reset parameters directly in hexadecimal notation. This setting may only be selected if you have previously received the hexadecimal string for the setting from a member of the SIEMENS staff.

Parameters via FB / optical readers

Table 10-169 Parameters via FB / optical readers

Parameter	Parameter value	Default value	Description
MOBY mode	RF200/RF300/ RF600; MV4x0; MO-BY U/D	RF200/RF300/ RF600; MV4x0; MO-BY U/D	Only the mode "RF200/RF300/RF600; MV4x0; MOBY U/D" is supported. For further information on resetting a function block, refer to the documentation for the RF120C communications module with application blocks for S7-1200 and S7-1500. *

* You will find information on the documentation here (https://support.industry.siemens.com/cs/ww/en/ps/14971/man).

Communications module RF170C

Communications module for Ident systems - RF170C, RS422/RS232

The module RF170C RS422/RS232 is a communications module for Ident systems for the I/O system ET 200pro. Several RF170C RS422/RS232 communications modules can be mounted and operated on an IM 154-8 CPU and on an interface module IM 154-1 DP, IM 154-2 DP High Feature or IM 154-4 PN High Feature of the ET 200pro.

A maximum of two readers can be connected to the RF170C RS422/RS232. The connected readers are processed at the same time

The function blocks FB 45, FC 45 and the program block for the Ident profile/RFID standard profile allow simple programming using the SIMATIC S7 tools. The function blocks FB 45, FC 45 and the program block for the Ident profile/RFID standard profile can be operated both on the S7-300 and the S7-400. For the S7-1200 and S7-1500 controllers, the Ident profile must be used.

The transponder data is accessed by means of normal addressing or file handler addressing.

Physical addressing of the transponder

Physical addressing of a transponder is also referred to as normal addressing. When using normal addressing you can set up the data structure of a transponder yourself You decide which physical address which data is written to. The addressing of the transponder memory generally begins at the address 0x0000 hex and ends at an end address that corresponds to the size of the transponder memory.

File handler addressing of the transponder

With file handler addressing you specify a file name for access to the data. The file name consists of eight ASCII characters. The file handler manages the user data on the transponder independently. You do not need to configure any data structures on the transponder.

"Module parameters" parameter group

In this parameter group, you can configure all module-specific parameters of the RF170C RS422/RS232.

Table 10-170 Parameters in the "Module parameters" parameter group

Parameter	Parameter value	Default value	Description
User Mode	Ident profile/RFID standard profile FB 45 / FC 45 FB 55 / FC 55	Ident profile/RFID standard profile	Selection depends on the communications module and and Ident system being used. With this parameter you select the block: • Ident profile/RFID standard profile: The program block for the Ident profile/RFID standard profile is used on the controller. • FB 45 / FC 45: Single tag mode FB 45 (PROFIBUS/ PROFINET) or FC 45 (PROFIBUS) is used on the controller. • FB 55 / FC 55: Multitag mode. FB 55 (PROFIBUS/ PROFINET) or FC 55 (PROFIBUS) is used on the controller.
MOBY Mode	RF200/RF300/RF600; MV4x0; MOBY U/D MOBY I/E MV3xx Freeport protocol RF300 filehandler	RF200/RF300/ RF600; MV4x0; MOBY U/D	Selection depends on the communications module and and Ident system being used. With this parameter, you set the mode of the communications module. • RF200/RF300/RF600; MV4x0; MOBY U/D • MOBY I/E • MV3xx • Freeport protocol • RF300 filehandler Normal addressing: The transponder is addressed with physical addresses. Filehandler: Prior to use, the transponder needs to be formatted.
Transmission speed	19.2 kBd 57.6 kBd 115.2 kBd	115.2 kBd	Selection depends on the communications module and and Ident system being used. With this parameter you set the transmission speed between the communications module and reader. When the RFID reader is connected: After changing the transmission speed, the reader must be turned off and on again. With a connected optical reader: The transmission speed selected here must match the transmission speed selected in the firmware of the reader.

Parameter	Parameter value	Default value	Description
Diagnostics messages	None Hard errors	None	Selection depends on the communications module and and Ident system being used. With this parameter you set whether hardware diagnostics messages will be reported. • None: Apart from standard diagnostics, no other alarms are generated. • Hard errors: Critical hardware errors are reported by the S7 diagnostics.
Suppression of Error LED	None Channel 1 Channel 2	None	Disabling the Error LED (ERR) of a channel. The communications module has two channels to which the readers / optical readers can be connected. The Error LED of the other channel flashes permanently when only one of the channels is being used. With the help of the suppression, you can disable the Error LED of the unused channel.
Interface	RS-232 RS-422	--	Selection of the interface type that the connected hardware (reader / optical reader) uses.

"Message Frame" parameter group

The "Frame" parameter group is only displayed when you have selected the parameter value "Freeport protocol" in the "MOBY Mode" parameter of the module parameters. In this parameter group, you can configure all parameters specific to the "Freeport protocol".

Table 10-171 Parameters in the "Frame" parameter group

Parameter	Parameter value	Default value	Description
Data bits	7 8	8	Selection of the number of bits on which a character is represented.
Parity	None Odd Even Fixed value 1 Fixed value 0	None	Parity selection A sequence of data bits can be expanded by a parity bit. With its value "0" or "1", the parity bit is added to the sum of all bits (data bits and parity bits) to form a defined status. This increases data reliability. • None: Data is sent without a parity bit. • Odd: The parity bit is set so that the sum of the data bits (including the parity bit) is odd when the signal state is "1". • Even: The parity bit is set so that the sum of the data bits (including the parity bit) is even when the signal state is "1". • Fixed value 1: The parity bit is set permanently to the value "1". • Fixed value 0: The parity bit is set permanently to the value "0".
Stop bits	1 2	1	Selection of the number of stop bits that indicate the end of a character. The stop bits are appended to every transferred character during transmission.

Parameter	Parameter value	Default value	Description
Specifying end detection	After character delay time elapses On receipt of fixed number of characters On receipt of the end delimiter(s):	After character delay time elapses	Specifies the end detection of a received frame. • After character delay time elapses: The frame has neither a fixed length nor defined end delimiters. The end of a frame is indicated by a gap in the character sequence. The size of this gap is specified by the character delay time. • On receipt of fixed number of characters: The length of the received frame is always the same. When data is received, the end of the frame is recognized when the set number of characters has been received. • On receipt of the end delimiter(s): At the end of the frame there are one or two defined end delimiters. When data is received, the end of the frame is recognized when the set end delimiter is received.
No. of end delimiters	1 2	1	Selection of the number of end delimiters. A maximum of 2 end delimiters can be configured. When data is received, the end of the frame is recognized when the selected end delimiter combination is received.
1st end delimiter	0...7F / 0...FF	3	Input of the 1st end delimiter of maximum two end delimiters for end criteria “On receipt of the end delimiter(s)”. The selected end delimiter or the selected end delimiter combination limits the length of the frame. Parameter value depending on the "Data bits" parameter.
2nd end delimiter	0...7F / 0...FF	0	Input of the 2nd end delimiter of maximum two end delimiters for end criteria “On receipt of the end delimiter(s)”. The selected end delimiter or the selected end delimiter combination limits the length of the frame. Parameter value depending on the "Data bits" parameter.
Frame length	1...233 / 1...229	233	Entry of the frame length in bytes for the end criterion "On receipt of fixed number of characters".
Character delay time	0...65535	15	Entry of the time [ms] that may elapse until a frame end is recognized. Select the character delay time dependent on the send behavior of your communications partner. Depending on the data transmission speed the character delay time is limited to a minimum value. Note that the ASCII driver pauses between two message frames when sending.

"I/O addresses" parameter group

Inputs

1. Assign a start address to the module.
2. If you want to assign the address area to a process image partition, select the desired process image partition in the "Process image" drop-down list.

Outputs

1. Assign a start address to the module.
2. If you want to assign the address area to a process image partition, select the desired process image partition in the "Process image" drop-down list.

Note

Input and output address must be identical.

RF180C communications module

Communications module for Ident systems - RF180C

The module RF180C is a communications module for Ident systems for PROFIBUS IO.

A maximum of two readers can be connected to the RF180C. The connected readers are processed at the same time

The function blocks FB 45, FB 55, and the program block for the RFID standard profile allow simple programming using the SIMATIC S7 tools. The function blocks FB 45, FB 55,and the program block for the RFID standard profile can be operated both on the S7-300 and the S7-400.

The transponder data is accessed by means of normal addressing or file handler addressing.

Physical addressing of the transponder

Physical addressing of a transponder is also referred to as normal addressing. When using normal addressing you can set up the data structure of a transponder yourself You decide which physical address which data is written to. The addressing of the transponder memory generally begins at the address 0x0000 hex and ends at an end address that corresponds to the size of the transponder memory.

File handler addressing of the transponder

With file handler addressing you specify a file name for access to the data. The file name consists of eight ASCII characters. The file handler manages the user data on the transponder independently. You do not need to configure any data structures on the transponder.

"Module parameters" parameter group

In this parameter group, you can configure all module-specific parameters of the RF180C.

Table 10-172 Parameters in the "Module parameters" parameter group

Parameter	Parameter value	Default value	Description
User Mode	FB 45 FB 55 RFID standard profile	FB 45	Selection depends on the communications module and and Ident system being used. With this parameter you select the block: • FB 45: Single tag mode FB 45 (PROFIBUS/ PROFINET) is used on the controller. • FB 55: Multitag mode. FB 55 (PROFIBUS/ PROFINET) is used on the controller. • RFID standard profile The program block for the RFID standard profile is used on the controller.
MOBY Mode	MOBY I/E normal addressing MOBY I file handler RF200/RF300/RF600; MOBY U/D normal adr. MOBY U file handler RF300 filehandler	RF200/RF300/ RF600; MOBY U/D normal addr.	Selection depends on the communications module and and Ident system being used. With this parameter, you set the mode of the communications module. • MOBY I/E normal addressing • MOBY I file handler • RF200/RF300/RF600; MOBY U/D normal adr. • MOBY U file handler • RF300 filehandler Normal addressing: The transponder is addressed with physical addresses. Filehandler: Prior to use, the transponder needs to be formatted.
Transmission speed	19.2 kBd 57.6 kBd 115.2 kBd	115.2 kBd	Selection depends on the communications module and and Ident system being used. With this parameter you set the transmission speed between the communications module and reader. When the RFID reader is connected: After changing the transmission speed, the reader must be turned off and on again. With a connected optical reader: The transmission speed selected here must match the transmission speed selected in the firmware of the reader.

Parameter	Parameter value	Default value	Description
Diagnostics messages	None Hard errors Hard/Soft Errors	None	With this parameter you set whether hardware diagnostics messages will be reported. • None: Apart from standard diagnostics, no other alarms are generated. • Hard errors: Critical hardware errors are reported by the S7 diagnostics. • Hard/Soft Errors: Critical hardware faults and errors occurring when processing commands are reported by the S7 diagnostics.
Suppression of Error LED	None Channel 1 Channel 2	None	Disabling the Error LED (ERR) of a channel. The communications module has two channels to which the readers / optical readers can be connected. The Error LED of the other channel flashes permanently when only one of the channels is being used. With the help of the suppression, you can disable the Error LED of the unused channel.

ASM 456 communications module

Communications module for Ident systems - ASM 456

The ASM 456 module is a communications module for Ident systems for PROFIBUS DP V1.

A maximum of two readers can be connected to the ASM 456. The connected readers are processed at the same time

The function blocks FB 45, FC 45, FB 55, FC 55, FC 56 and the program block for the Ident profile/RFID standard profile allow simple programming using the SIMATIC S7 tools. The function blocks FB 45, FC 45, FB 55, FC 55, FC 56 and the program block for the Ident profile/ RFID standard profile can be operated both on the S7-300 and the S7-400. For the S7-1200 and S7-1500 the Ident profile must be used.

The transponder data is accessed by means of normal addressing or file handler addressing.

Physical addressing of the transponder

Physical addressing of a transponder is also referred to as normal addressing. When using normal addressing you can set up the data structure of a transponder yourself You decide which physical address which data is written to. The addressing of the transponder memory generally begins at the address 0x0000 hex and ends at an end address that corresponds to the size of the transponder memory.

File handler addressing of the transponder

With file handler addressing you specify a file name for access to the data. The file name consists of eight ASCII characters. The file handler manages the user data on the transponder independently. You do not need to configure any data structures on the transponder.

"Device-specific parameters" parameter group.

In this parameter group, you can configure all module-specific parameters of the ASM 456.

Table 10-173 Parameters in the "Module parameters" parameter group

Parameter	Parameter value	Default value	Description
User Mode	Ident profile/RFID standard profile FB 45 / FC 45 FB 55 / FC 55 FC 56	Ident profile/RFID standard profile	Selection depends on the communications module and and Ident system being used. With this parameter you select the block: • Ident profile/RFID standard profile: The program block for the Ident profile/RFID standard profile is used on the controller. • FB 45 / FC 45: Single tag mode FB 45 (PROFIBUS/ PROFINET) or FC 45 (PROFIBUS) is used on the controller. • FB 55 / FC 55: Multitag mode. FB 55 (PROFIBUS/ PROFINET) or FC 55 (PROFIBUS) is used on the controller. • FC 56 File handler for S7-300 and S7-400
MOBY Mode	RF200/RF300/RF600; MV4x0; MOBY U/D RF680R/RF685R MOBY I/E normal addressing RF300 filehandler MOBY U file handler MOBY I File handler	RF200/RF300/ RF600; MV4x0; MOBY U/D	Selection depends on the communications module and and Ident system being used. With this parameter, you set the mode of the communications module. • RF200/RF300/RF600; MV4x0; MOBY U/D • RF680R/RF685R • MOBY I/E normal addressing • RF300 filehandler • MOBY U file handler • MOBY I file handler Normal addressing: The transponder is addressed with physical addresses. Filehandler: Prior to use, the transponder needs to be formatted.

Parameter	Parameter value	Default value	Description
Transmission speed	19.2 kBd 57.6 kBd 115.2 kBd	115.2 kBd	Selection depends on the communications module and and Ident system being used. With this parameter you set the transmission speed between the communications module and reader. When the RFID reader is connected: After changing the transmission speed, the reader must be turned off and on again. With a connected optical reader: The transmission speed selected here must match the transmission speed selected in the firmware of the reader.
Diagnostics messages	None Hard errors Hard/soft errors low priority Hard/soft errors high priority	None	Selection depends on the communications module and and Ident system being used. With this parameter you set whether hardware diagnostics messages will be reported. • None: Apart from standard diagnostics, no other alarms are generated. • Hard errors: Critical hardware errors are reported by the S7 diagnostics. • Hard/soft errors low priority Critical hardware errors and errors that occur during command processing are reported by the S7 diagnostics. The "Ext_Diag" bit is not set. • Hard/soft errors high priority Critical hardware errors and errors that occur during command processing are reported by the S7 diagnostics. The "Ext_Diag" bit is set.
Suppression of Error LED	None Channel 1 Channel 2	None	Disabling the Error LED (ERR) of a channel. The communications module has two channels to which the readers / optical readers can be connected. The Error LED of the other channel flashes permanently when only one of the channels is being used. With the help of the suppression, you can disable the Error LED of the unused channel.

ASM 475 communications module

What you should know about the ASM 475

Introduction

Up to eight ASM 475 interface modules can be plugged in and operated in a rack of the SIMATIC S7-300. If you have a configuration with several racks (max. four racks), the ASM 475 can be plugged in and operated in each rack. This means that with the maximum configuration of a SIMATIC S7-300, 32 ASM 475 modules can be operated.

A maximum of two SLGs (write/read devices) can be connected to the ASM 475. The processing of the connected SLGs is parallel. FC45 allows simple programming using the SIMATIC S7 tools.

FC45 can be used both in the S7-300 and S7-400.

With S7-400, the ASM 475 is connected via an ET 200M.

Note that the

- IM 153-1 must have at least order number 6ES7 153-1AA03-0XB0 or 6ES7 153-1AA83-0XB0 and
- IM 153-2 at least order number 6ES7 153-2AA02-0XB0 or 6ES7 153-2AB01-0XB0.

Access to the MDS data is with normal addressing.

Physical addressing of the MDS (mobile data storage)

The physical addressing of an MDS is also known as **normal addressing**. Users set up the structure of the MDS themselves. They know the physical MDS address to which data is written. The addressing of the MDS storage generally begins at 0000 hex and ends at an end address corresponding to the size of the MDS storage.

File handler addressing of the MDS

For file handler addressing, the user specifies a file name for access to the data. The file name consists of eight ASCII characters. The file handler manages the user data independently on the MDS. The user does not need to configure data structures on the MDS.

Prior to use, the MDS needs to be formatted.

Reader

RF680R/RF685R reader

Readers RF680R/RF685R for the UHF range

The UHF readers RF680R and RF685R are intended for use in logistics and automation, for example on a production line but are equally suitable for applications in logistics. To meet these requirements, the readers were equipped with a high transmit power and degree of protection (IP65). The readers are equipped with extensive diagnostics options and can process ISO 18000-6C and ISO 18000-6B transponders.

The RF685R has one special feature with its internal, adaptive antenna. This significantly increases the reliability of read and write actions even under difficult radio conditions.

The RF680R and RF685R readers are either integrated without problems in SIMATIC S7 automation systems via an integrated PROFINET connector or via the RS-422 interface and the ASM 456 communications module via PROFIBUS. Suitable programming blocks are available. A second Ethernet interface (both M12) can be used for diagnostics during operation so that the connection to the higher-level system does not need to be interrupted.

The WBM (Web Based Management) allows commissioning, configuration and diagnostics of the devices using an Internet browser. This makes additional updates and installation of configuration and diagnostics software unnecessary.

"Web Based Management" parameter group

In this parameter group you can start the Web Based Management.

Table 10-174 Parameters in the "Web-based management" parameter group

Parameter	Description
Web Based Management (WBM)	Starting the Web Based Management of the reader. The Web Based Management provides a wide range of functions for configuring the readers. Note: The WBM can only be started if either the PROFINET connection between the CPU and reader is established or the reader was assigned the IP address stored in the project. This means that the device name has been assigned and the TIA configuration must be loaded on the SIMATIC controller.

"Module parameters" parameter group

In this parameter group, you can configure all module-specific parameters of the RF680R/ RF685R readers.

Table 10-175 Parameters in the "Module parameters" parameter group

Parameter	Parameter value	Default value	Description
RFID read point alarm	Off On	Off	Enabling/disabling read-point related diagnostics messages.

"Configuration management" parameter group

In this parameter group you can load or save the configuration data.

Table 10-176 Parameters in the "Configuration management" parameter group

Parameter	Description
User name	User name of a user created on the reader Note that the user must have the required rights.
Password	Entry of the password for the user
Load configuration to device	Loading configuration data on the reader from the STEP 7 project.
Check the configuration in the project.	Saving configuration data of the reader in the current STEP 7 project.

Requirement

The following requirements must be met to allow configuration data to be loaded or saved:

- The correct IP address of the reader is entered in the "PROFINET interface [X1]".
- The user name and corresponding password have been created in the WBM.
- The entered user has the required rights to run the download/upload.

Note

User name and password only necessary if user management is enabled

The "User name" and "Password" text boxes only need to be completed if the user management of the WBM is enabled.

"RFID communication" parameter group

In this parameter group you can configure the address range of the RFID communication.

Table 10-177 Parameters in the "RFID communication" parameter group

Parameter	Description
General	General settings Input boxes for the name of the communication and for a comment.
I/O addresses	Specify the I/O address of the reader ("LADDR") This parameter is used in the "IID_HW_CONNECT" variable.
Hardware identifier	Specify the hardware identifier of the reader ("HW-ID") This parameter is used in the "IID_HW_CONNECT" variable.

"Digital inputs/outputs" parameter group

In this parameter group you can configure the address range of the digital inputs / outputs.

Table 10-178 Parameters in the "Digital inputs / outputs" parameter group

Parameter	Description
General	General settings Input boxes for the name of the of the digital inputs / outputs and for a comment.
I/O addresses	Specify the I/O address of the digital inputs/outputs Using the set address range (I/O address), the digital inputs/outputs configured in the WBM of the reader can be accessed.
Hardware identifier	Specify the hardware identifier of the digital inputs/outputs.

Technology object "SIMATIC Ident"

Overview of the steps in configuration

Introduction

The technology object for SIMATIC Ident products allows simple parameter assignment of an Ident system in the TIA Portal. It provides an interface with simple parameter assignment between the configured hardware and the function blocks in the "SIMATIC Ident" library.

With the aid of the "Configuration" object you can select a configured device, decide on the channel / the read point and if necessary specify the connected Ident system. The technology object therefore automatically determines all the parameters required for operation.

The "Diagnostics" object displays the last 5 error messages signaled by the Ident device to the function blocks. In addition to this for every error message an error description and detailed information about the executed command are displayed.

The following overview shows the basic procedure for adding and configuring a "SIMATIC Ident" technology object.

Procedure

Follow the steps below to configure the "SIMATIC Ident" technology object:

Step	Description
1	Adding a technology object (Page 1897)
2	AUTOHOTSPOT
3	Calling the Ident instruction in the user program (Page 1904)
4	Load technology object in CPU
5	AUTOHOTSPOT

Adding a technology object

Adding a technology object in the project navigator

When you add a technology object, a data block is generated. The configuration of the technology object is stored in this data block.

Requirements

To use the "SIMATIC Ident" technology object the following requirements must be met:

- A project (V14 SP1 or higher) was created.
- According to the hardware used, the connected devices were added to the hardware configuration ("Devices & Networks"), networked with each other and configured.
- The project contains at least one S7 controller (S7-1200 or S7-1500) and at least one device (e.g. communications module, reader, optical reader).

Procedure

Follow the steps below to add a technology object:

1. Open the CPU folder in the project tree.
2. Open the "Technology objects" folder.
3. Double-click on ""Add new object".
   The "Add new object" dialog box opens.
4. Click the "SIMATIC Ident" button.
5. In the "Version" column select the newest version (as of "V5.0").
6. Select the object "TO_Ident".
7. Type in an individual name for the technological object in the "Name" input field.
8. Click "More information" if you want to store your own information for the technology object.
9. Confirm with "OK".

Result

The new technology object is created and saved to the "Technological objects" folder in the project tree.

Configuring a technology object

Working with the configuration dialog

You configure the properties of the technology object in the configuration window. To open the configuration window of the technology object, follow these steps:

1. In the project tree, open the "Technological objects" folder.
2. Open the technology object the project tree.
3. Double-click on the "Configuration" object.

The configuration is divided into the following categories:

- Basic parameters
- Ident device parameters
- Reader parameters

Symbols of the configuration window

Symbols in the area navigation of the configuration and Inspector window show additional information about the completeness of the configuration:

	The configuration contains default values and is complete The configuration only contains default values. With these preset values you can use the technology object without additional changes.
	The configuration contains values defined by the user or automatically adapted values and is complete All input fields of the configuration contain valid values and at least one default value was modified.
	The configuration is valid but contains unexpected values At least one input field or a drop down list contains an unexpected value. The corresponding field, or the drop-down list is displayed on a yellow background. When you click on it, the roll-out warning message shows you the cause of the warning.
	The configuration is incomplete or incorrect At least one input field or a drop down list contains no or an invalid value. The corresponding field, or the drop-down list, is displayed on a red background. Click the roll-out error message to identify the cause of the error.

Configuring basic parameters

In this parameter group, you can configure the module-specific parameters of your system.

Table 10-179 Parameters in the "Basic parameters" parameter group

Parameter	Description
Ident device	Click "..." and select the Ident device to be configured in the folder "Distributed I/O" or "Local modules". The following devices can be selected: • ASM 456 • RF120C, RF170C, RF180C • RF68xR • MV4x0
Channel	Open the drop down list and select the channel of the communications module to which the device to be configured is connected.
Reader parameter assignment	Open the drop down list and select the device/system connected to the Ident device that needs parameter assignment.
Read point	Requirement: In the "Reader parameter assignment" parameter the readers RF680R or RF685R were selected. Open the drop down list and select the read point to be used.

Configuring Ident device parameters

Depending on the selected Ident device, the following Ident device parameters are displayed in this parameter group. The displayed Ident device parameters are adopted automatically from the device configuration of the connected devices.

Note that not all parameter values adopted and displayed from the device configuration of the connected devices are permitted in conjunction with the technology object. If non-permitted

values are set in the device configuration you can change these either automatically with the "Autoconfiguration" button or manually using the device configuration.

Table 10-180 Parameters in the "Ident device parameters" parameter group

Parameter	Possible displayed parameter value	Permitted parameter values	Description
User Mode	Ident profile/RFID standard profile FB 45 / FC 45 FB 55 / FC 55	Ident profile/RFID standard profile	With this parameter you select the block. Note that only the following parameter values are permitted in conjunction with the technology object. • Ident profile/RFID standard profile: The program block for the Ident profile/RFID standard profile is used on the controller.
MOBY Mode	RF200/RF300/RF600; MV4x0; MOBY U/D RF680R/RF685R MOBY I/E Normal addressing RF300 file handler MOBY U file handler MOBY I file handler	RF200/RF300/RF600; MV4x0; MOBY U/D RF680R/RF685R	With this parameter, you set the mode of the communications module. Note that only the following parameter values are permitted in conjunction with the technology object. • RF200/RF300/RF600; MV4x0; MOBY U/D • RF680R/RF685R
Connected device	Parameters via FB / optical readers	Parameters via FB / optical readers	This parameter is only displayed in conjunction with the communications module RF120C. Note that only the following parameter values are permitted in conjunction with the technology object. • Parameters via FB / optical readers

"Autoconfiguration" button

The displayed HWCN parameters are adopted automatically from the device configuration of the connected devices. Note that not all parameter values are permitted in conjunction with the technology object. Non-permitted parameter values are highlighted in color.

Click the "Autoconfiguration" button to set the permitted parameter values automatically. The changed parameter values are adopted automatically in the device configuration of the connected devices from the technology object.

Note

Behavior with MV4x0 and RF68xR

Note that with the devices MV4x0 and RF68xR the "Auto configuration" button does not work. With these devices you need to enter the HWCN parameters in the device configuration of the connected devices manually.

Configuring reader parameters

Depending on the set basic parameters you can configure the following reader parameters in this parameter group.

Table 10-181 Overview of the "Reader parameters"

Reader parameter assignment in "Basic parameters"	Transmission speed	Presence check	Reset error LED	HF power	Max. number of transponders	Transponder type			Program selection with reset	Byte sequence of the reset parameter [hex]
						ISO Mode	RF300	MOBYE		
RF200 general	✓	✓	✓	-	✓	✓	-	-	-	-
RF290R	✓	✓	✓	✓	✓	✓	-	-	-	-
RF300 general	✓	✓	✓	-	✓	✓	✓	-	-	-
RF380R	✓	✓	✓	✓	✓	✓	✓	-	-	-
RF300 Gen 2 general	✓	✓	✓	-	✓	✓	✓	✓	-	-
RF380R Gen 2	✓	✓	✓	✓	✓	✓	✓	✓	-	-
RF680R [1]	-	-	-	-	-	-	-	-	-	-
RF685R [1]	-	-	-	-	-	-	-	-	-	-
MV4x0 [1]	-	-	-	-	-	-	-	-	✓	-
General readers	✓	-	-	-	-	-	-	-	-	✓

[1] The device-specific parameters of these devices are set in the WBM of the relevant devices.

Table 10-182 Parameters in the "Reader parameters" parameter group

Parameter	Parameter value	Default value	Description
Transmission speed	19.2 kBd 57.6 kBd 115.2 kBd	115.2 kBd	The selection depends on the communications module and Ident system being used. Note that the value specified here is adopted automatically from the device configuration of the connected devices. With this parameter you set the transmission speed between the communications module and reader. When the RFID reader is connected: After changing the transmission speed, the reader must be turned off and on again (cycle power). When an optical reader is connected: The transmission speed selected here must match the transmission speed selected in the firmware of the reader.
Presence check	On Off (RF field on) Off (RF field off)	On	On = As soon as there is a transponder in the antenna field of the reader, its presence is reported. Off (RF field on) = the presence check in the FB is suppressed. The antenna on the reader is nevertheless turned on as long as it has not been turned off by a command. Off (RF field off) = the antenna is turned on only when a command is sent and it then turns itself off again (RF300 only).
Reset error LED	On Off	On	On = the flashing of the error LED on the communications module is reset by each FB reset. Off = the error LED always indicates the last error. The display can only be reset by turning off the communications module.
HF power [1]	0.50 ... 2.00 0.50 ... 5.00	1.00 1.25	Setting for the output power of the reader. The selectable values depend on the connected device.
Max. number of transponders	1 ... 4	1	Number of transponders expected in the antenna field. The selection depends on the connected device.
Transponder type	[1]	[1]	Selection of the transponder types used. The selection depends on the connected device.
Program selection with reset	No program change 1 ... 15	No program change	Selection of the program that will be used after restarting the optical reader. If the parameter "No program change" is set, after restarting the device the last active program is selected.
Byte sequence of the reset parameter [hex]	[1]	[1]	Hexadecimal representation of the byte sequence of the reset parameter of the reader.

[1] You will find a detailed description of this parameter in the following paragraphs:

The "HF power" parameter

The selectable values depend on the value specified in the parameter "Reader parameter assignment".

- RF290R
  - Range of values: 0.50 ... 5.00 W
  - Default value: 1.00 W
- RF380R; RF380R (gen 2)
  - Range of values: 0.50 ... 2.00 W
  - Default value: 1.25 W

The "Transponder type" parameter

Selection of the transponders used. The selection depends on the value specified in the parameter "Reader parameter assignment". The following transponder types can be selected:

- ISO Mode
- RF300
- MOBY E

If you select "ISO mode", the following selection options are displayed depending on the reader parameter assignment:

Reader parameter assignment	Values
RF200 general RF290R	ISO 15693 ISO 15693 (Infineon)
RF300 general RF380R	RF300 ISO 15693 general ISO 15693 (Infineon) ISO 15693 (Fujitsu) ISO 15693 (NXP) ISO 15693 (TI) ISO 15693 (STM)
RF300 general (gen 2) RF380R general (gen 2)	ISO 15693 general ISO 15693 (Infineon) ISO 15693 (Fujitsu) ISO 15693 (NXP) ISO 15693 (TI) ISO 15693 (STM) ISO 15693 (Fujitsu - 8 kB)

The parameter "Byte sequence of the reset parameter [hex]"

With this parameter you can specify the Reset parameter in hexadecimal format. This setting may only be selected if you have previously received the hex string for the setting from a member of the SIEMENS staff.

The following Reset parameters are available:

Byte	1	2...5 [1]	6	7...8	9	10	11	12	13...14	15	16
Value	0x04	0x00	0x0A	0x00	scan-ning_ time	param	op-tion_1	distance_ limiting	multitag	field_on_ control	field_on_ time

[1] In the communications module RF180C as of V2.2 in conjunction with MOBY U byte 4 is preset with the calendar week and 5 with the year.

Calling the Ident instruction in the user program

Procedure

Proceed as follows to call the instruction in the user program:

1. Open the CPU folder in the project tree.
2. Open the "Program blocks" folder.
3. Double-click on the OB for cyclic program execution.
   The block is opened in the working area.
4. In the "Instructions" window open the group "Optional packages" and the "SIMATIC Ident" folder.
   The folder and the subfolders contain the instructions.
5. Select an instruction and drag it to your OB.
   The "Call options" dialog is opened.
6. From the "Name" list select an instance data block or enter the name for a new instance data block.
7. Confirm with "OK".
8. Select the configured technology object and drag it to the parameter "HW_CONNECT".

Reset the readers using the "Ident profile" or the "Reset_Reader" block

There are various methods for resetting readers:

- If you work with the Ident blocks, reset all reader types with the "Reset_Reader" block. This transfers the values set in the technology object to the readers.
- If you work with the Ident profile, when you set the "INIT" bit, the values set in the technology object are transferred to the readers.
- If you want to send your own "WRITE CONFIG" command, instead of the "INIT" bit you need to use the "EXECUTE" bit.

If a Reset parameter of the technology object needs to be changed at runtime, the relevant value can be overwritten in the data block of the technology object using the user program. The Reset parameters are located in the data block of the technology object in "Static.CONFIGURATION.param_buffer[]".

You will find detailed information on the Ident profile in the manual ""Ident Profile and Ident Blocks, Standard Function for Ident Systems" on the pages of "Industry Online Support (https://support.industry.siemens.com/cs/ww/en/ps/14971/man)".

Loading the configuration on the CPU

After configuration you need to load the changes from the technology object or the S7 controller on the connected devices.according to

Load to device

When loading to the devices note that a distinction is made between "Software (changes only)" and "Hardware and software (changes only)". If only changes to the software were made, loading is via the technology object. If changes were made to the hardware and software, loading is via the S7 controller.

NOTICE
Load to device when using the "Autoconfiguration" button If the "Autoconfiguration" button was used during configuration, you need to load the changes on the connected devices via the S7 controller ("Hardware and software (changes only)").

Diagnosing a technology object

After configuration of the technology object is complete, you can monitor the devices configured in the technology object using the diagnostics function.

Requirements

- There is an online connection between STEP 7 (TIA Portal) and the S7 controller.
- The Ident instruction is called in the user program.
- A technology object was created and loaded on the S7 controller-
- The S7 controller is in "RUN" mode.

Procedure

To open the display editor for the diagnostics function, follow these steps:

1. In the project tree, open the "Technological objects" folder.
2. Open the relevant technology object in the project tree.
3. Double-click on the "Diagnostics" object.
4. Click on the symbol."Monitor all" to activate the monitoring mode.

Display

The following values are read by the technology object and displaed:

Table 10-183 Displayed values of "Block diagnostics"

Displayed values	Description
Date and time	Shows the date and time when the error occurred.
Command	Shows the command with which the error occurred.
Error description	Shows a brief description of the error that occurred. Note: The last five error messages are displayed and described in the table. If a new error message is displayed, the previously displayed error messages are moved one line down. A maximum of five error messages are displayed.
Error message	Shows the error message of the error that occurred
Error description	Shows a detailed description of the error that occurred
Date and time	Shows the date and time when the error occurred.
Command code	Show the command code of the command with which the error occurred.
Command	Shows the command with which the error occurred.
Attribute	Shows the attribute of the command with which the error occurred.

Optical reading systems

Optical readers MV420/MV440

Optical readers MV420/MV440

The optical reading systems MV420/MV440 were designed specifically for the recognition and evaluation of numerous machine-readable codes, plaint text and objects in industrial production.

The readers can be operated both in the controllers S7-300, S7-400, S7-1200 and S7-1500 as well as with various third-party controllers. The connection to the controller is via PROFINET IO (FB79/Ident profile), TCP, RS-232 or via communications modules.

The function blocks FB79 and the Ident profile allow simple programming using the SIMATIC S7 tools.

"Operating mode" parameter group

In this parameter group, you can configure the mode of the optical readers MV420/MV440.

Table 10-184 Parameters in the "Operating mode" parameter group

Parameter	Parameter value	Default value	Description
Function block	FB79 Ident profile	FB79	With this parameter you select the block: • FB79 Compatible with the "VS130-2" reader; Compatible controllers: S7-300 and S7-400 • Ident profile Complex Ident block with MV-specific blocks; Compatible controllers: S7-300, S7-400, S7-1200 and S7-1500

10.1.5.2 Distributed I/O

Distributed I/O systems

SIMATIC ET 200 - The right solution for all applications

SIMATIC ET 200 provides the most varied range of distributed I/O systems.

- Solutions for use in the control cabinet
- Solutions without control cabinet directly at the machine

Additionally, there are also components that can be used in explosive areas. SIMATIC ET 200 systems for construction without a control cabinet are contained in robust, glass-fibre reinforced plastic casing and are therefore shock-resistant, resistant to dirt and watertight.

Their modular design allows the ET 200 systems to be easily scaled and expanded in small steps. Fully-integrated auxiliary modules lower costs and also provide a wide range of possible applications. There are several combination possibilities available:

- Digital and analog I/OS
- Intelligent modules with CPU functions,
- Safety technology,
- Pneumatics,
- Frequency converters
- Various technology modules.

Communication via PROFIBUS and PROFINET, uniform engineering, clear diagnostic possibilities as well as optimal connection to SIMATIC controller and HMI devices vouch for the unique consistency provided by Totally Integrated Automation.

The following table provides an overview of I/O devices for use in the control cabinet:

I/O device	Properties
ET 200L	• Cost-effective digital block I/O • Digital electronic blocks up to 32 channels
ET 200M	• Modular design with standard modules from SIMATIC S7-300 • Failsafe I/O modules • Use in hazardous areas up to Zone 2, sensors and actuators up to Zone 1 • High level of plant availability, for example by plugging and unplugging during operation
ET 200MP	• Modular design with standard SIMATIC S7-1500 modules • Isochronous PROFINET IO with send clocks up to 250 µs • IO device with PROFINET IO interface • DP slave with PROFIBUS DP interface • Technology modules for counting, positioning, time-based IO
ET 200S	• Highly modular design with multiple conductor connections • Multifunctional due to a wide range of modules • I-device • Use in explosive areas (Zone 2)
ET 200S COMPACT	• Highly modular design with multiple conductor connections • Multifunctional due to a wide range of modules • Use in explosive areas (Zone 2) • Integrated DE/DA

I/O device	Properties
ET 200SP	• Compact modules • Permanent wiring with one or more wire connection • PROFINET interface with 3 ports • Isochronous PROFINET IO with the profiles PROFIsafe and PROFIenergy • I-device • Modules for counting, positioning, weighing and measuring • Fail-safe CPUs and modules
ET 200iSP	• Modular design, also possible with redundancy • Robust and intrinsically safe design • Use in explosive areas up to Zone 1/21; sensors and actuators even up to Zone 0/20 • High level of plant availability, for example by plugging and unplugging during operation

The following table provides an overview of I/O devices for use without a control cabinet:

I/O device	Properties
ET 200AL	• Simple installation in all locations, in the smallest spaces • Connection to PROFINET IO, PROFIBUS DP or ET 200SP • High degree of protection IP65/IP67 For temperatures from -25 °C to +55 °C, acceleration up to 5 g • Connection of sensors and actuators using M8 and M12 connector system
ET 200eco PN	• Cost-efficient, space-saving block I/OS • Digital modules up to 16 channels (also configurable) • Analog modules, IO-link master and load voltage distributor • PROFINET connection with 2-port switch in each module • Can be flexibly distributed via PROFINET in line or star shape directly within the plant
ET 200eco	• Cost-effective digital block I/O • Flexible connection possibilities • Failsafe modules • High level of plant availability

I/O device	Properties
ET 200pro	• Modular design with compact housing • Easy assembly • Multifunctional due to a wide range of modules • I-device • High level of plant availability due to plugging and unplugging during operation and permanent wiring • Comprehensive diagnostics
ET 200R	• Specially for use on robots • Assembled directly on the chassis • Resistant to weld spatter due to robust metal housing

See also

Documentation on ET 200L (http://support.automation.siemens.com/WW/view/en/1142908/0/en)

Documentation on ET 200S (http://support.automation.siemens.com/WW/view/en/1144348/0/en)

Documentation on ET 200M (http://support.automation.siemens.com/WW/view/en/1142798/0/en)

Documentation on ET 200pro (http://support.automation.siemens.com/WW/view/en/21210852/0/en)

Documentation on ET 200iSP (http://support.automation.siemens.com/WW/view/en/28930789/0/en)

Documentation on ET 200R (http://support.automation.siemens.com/WW/view/en/11966255/0/en)

Documentation on ET 200eco PN (http://support.automation.siemens.com/WW/view/en/29999018/0/en)

Documentation on ET 200eco (http://support.automation.siemens.com/WW/view/en/12403834/0/en)

Documentation on ET 200SP (http://support.automation.siemens.com/WW/view/en/84133942/0/en)

Documentation on ET 200MP (http://support.automation.siemens.com/WW/view/en/86140384/en)

Documentation on ET 200AL (http://support.automation.siemens.com/WW/view/en/95242965/0/en)

Configuring HART variables

Introduction

Numerous HART field devices make available additional measured quantities (e.g. sensor temperature). These can be read if they are set accordingly in the field device configuration. Using the HART variables. it is possible to apply the set measured values directly from the field device into the I/O area of your automation system.

Regardless of the number of configured channels, a maximum of 8 HART variables can be assigned for HART modules and no more than 4 HART variables per channel. You assign the HART variables to a channel in the properties for the module ("HART variable settings" area). To do this, check the manual of the corresponding module.

Address assignment

By default, the HART modules occupy 16 input/output bytes (user data). If you configure HART variables, the modules occupy an additional 5 bytes for each HART variable.

If you use all 8 HART variables, the HART input module occupies a total of 56 input/output bytes (16 bytes + 8 x 5 bytes = 56 bytes).

The "None" configuration occupies no additional input bytes.

Configuration of HART variables

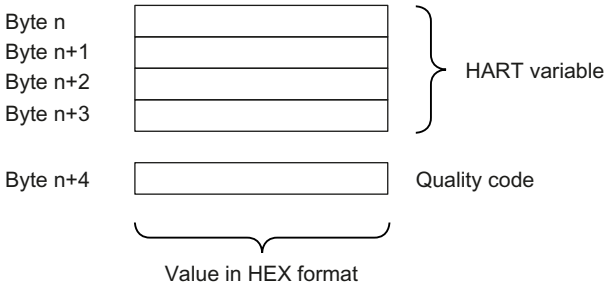
You can configure up to 4 HART variables for a channel

- PV (Primary Variable)
- SV (Secondary Variable)
- TV (Tertiary Variable)
- QV (Quaternary Variable)

CiR is a placeholder that reserves the address space for a HART variable. You must configure the HART variables you are not using with the "None" parameter.

Configuration of HART variables

Each HART variable occupies 5 bytes of input data and is structured as follows:



The quality code can accept different values depending on the module. To do this, check the manual of the corresponding module.

See also

Documentation for HART analog modules (http://support.automation.siemens.com/WW/view/en/22063748)

ET 200iSP

ET 200iSP Distributed I/O Station

Definition

The ET 200iSP distributed I/O station is a highly modular and intrinsically safe DP slave with degree of protection IP 30.

Area of application

The ET 200iSP distributed I/O station can be operated in potentially explosive atmospheres characterized by gas and dust:

Approval	ET 200iSP Station*	Inputs and outputs
ATEX	Zone 1, Zone 21	up to Zone 0, Zone 20 **
IECEx	Zone 2, Zone 22	up to Zone 0, Zone 20 **
* In combination with an appropriate enclosure ** for electronic module 2 DO Relay UC60V/2A: up to Zone 1, Zone 21		

The ET 200iSP distributed I/O station can, of course, also be used in the safety area.

You can insert almost any combination of ET 200iSP I/O modules directly next to the interface module that transfers the data to the DP master. This means you can adapt the configuration to suit your on-site requirements.

Every ET 200iSP consists of a power supply module, an interface module and a maximum of 32 electronic modules (for example digital electronics modules). Remember not to exceed the maximum current consumption.

Terminal modules and electronic modules

In principle, the ET 200iSP distributed I/O station consists of various passive terminal modules onto which you plug the power supply and the electronic modules.

The ET 200iSP is connected to PROFIBUS RS 485-IS by means of a connector on terminal module TM-IM/EM. Every ET 200iSP is a DP slave on the PROFIBUS RS 485-IS.

DP master

All ET 200iSP modules support communication with DP masters that are compliant with *IEC 61784-1:2002 Ed1 CP 3/1* and operate with "DP" transmission protocol (DP stands for distributed peripherals or distributed I/O).

See also

Documentation on ET 200iSP (http://support.automation.siemens.com/WW/view/de/28930789/0/en)

Assigning the channel and IEEE tag

Properties

Analog electronic modules 4 AI I 2WIRE/HART, 4 AI I 4WIRE/ HART and 4 AO I HART support up to four IEEE tags.

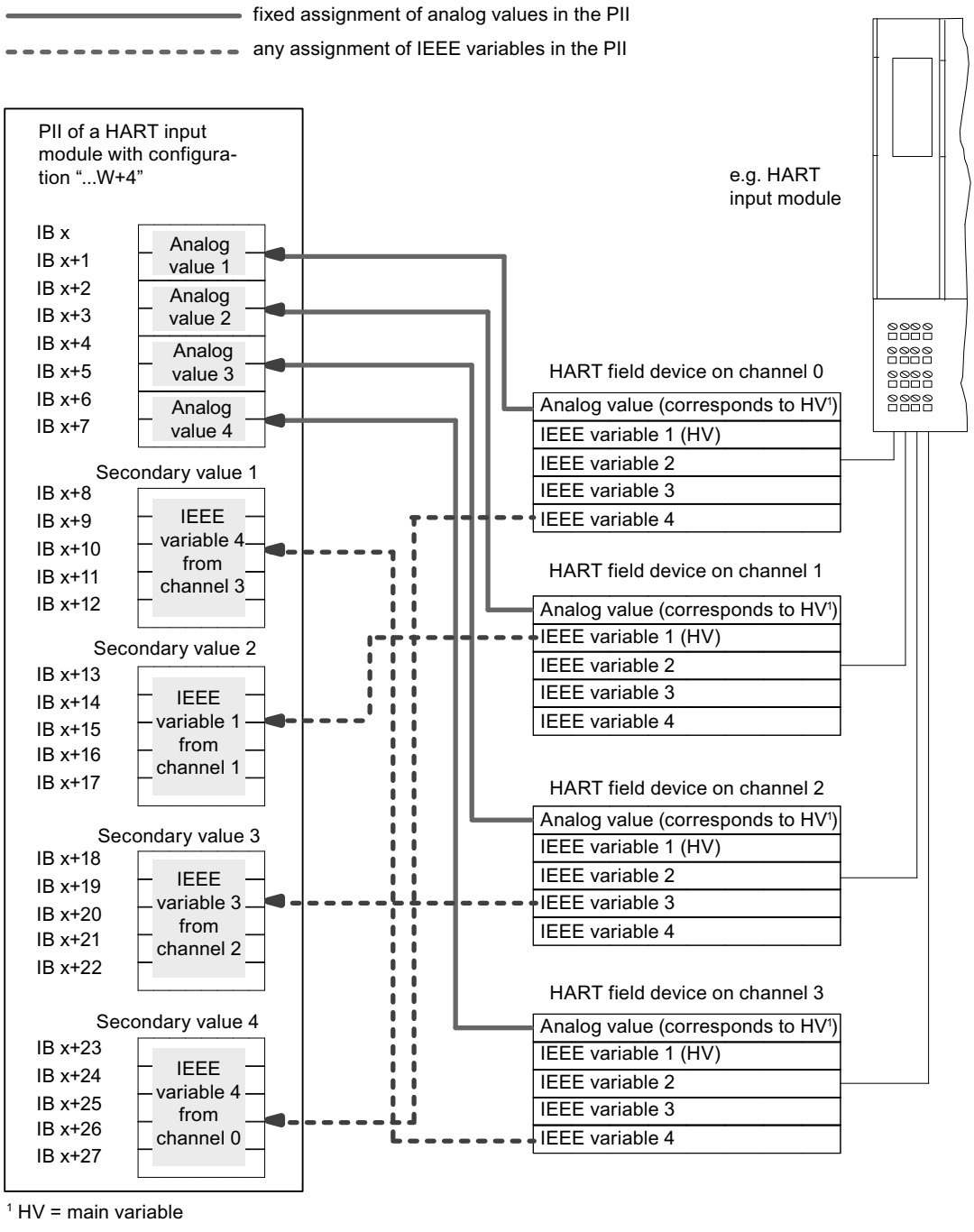
The process input image (PII) provides up to 20 bytes per module for the IEEE tags. Thus, four blocks of 5 bytes each are available for the four IEEE tags within the PII.

Requirements

The HART field device must support the assigned number of IEEE tags.

Assigning IEEE tags

You assign the IEEE tags of the field devices to any one of the four blocks in the PII.



[1] HV = main variable

See also

Documentation on ET 200iSP (http://support.automation.siemens.com/WW/view/de/28930789/0/en)

Assigning parameters to reference junctions for thermocouples

Compensation of the reference junction temperature

There are various ways of obtaining the reference junction temperature in order to get an absolute temperature value from the temperature difference between the reference junction and the measuring point.

Table 10-185 Compensation of the reference junction temperature

Option	Explanation	Reference junction parameters
No compensation	You record not only the temperature of the measurement point. The temperature of the reference junction (transition from Cu line to thermocouple line) also affects the thermo-electromotive force. The measured value then includes an error.	None
Use of a Pt100 Climatic Range resistance thermometer to record the reference junction temperature (best method)	You can record the reference junction temperature using a resistance thermometer (Pt100 Climatic Range). If parameterized accordingly, this temperature value is distributed to the 4 AI TC modules in the ET 200iSP where it is offset against the temperature value obtained at the measuring location. Number of reference junctions: 2	The parameter assignment of the IM 152 and the 4 AI TC must be coordinated: • 4 AI RTD assigned parameters for Pt100 climatic range in correct slot; • 4 AI TC: Reference junction : "yes"; select reference junction number "1" or "2" • IM 152-1:Assignment of the reference junction to a slot with 4 AI RTD; channel selection;
Internal compensation 4 AI TC	The TC sensor module (temperature sensor) is mounted onto the terminals of terminal module EM 4 AI TC. The temperature sensor reports the temperature of the terminals to the 4 AI TC. This value is then calculated together with the measured value from the channel of the electronic module.	• 4 AI TC: Reference junction number "internal"

Compensation by means of a resistance thermometer at the 4 AI RTD

If thermocouples that are connected to the inputs of the 4 AI RTD have the same reference junction, compensate by means of a 4 AI RTD.

For both channels of the 4 AI TC module, you can select "1", "2" or "internal" as the reference junction number. If you select "1" or "2", the same reference junction (RTD channel) is always used for all four channels.

Setting parameters for the reference junction

You set the reference junctions for the 4 AI TC electronic modules by means of the following parameters:

Table 10-186 Reference junction parameters

Parameter	Module	Range of values	Explanation
Slot reference junction 1 to slot 2	IM 152	none, 4 to 35	With this parameter, you can assign up to 2 slots (none, 4 to 35), on which the channels for reference temperature measurement (calculating the compensation value) are located.
Input reference junction 1 to 4 input reference junction	IM 152	RTD on channel 0 RTD on channel 1 RTD on channel 2 RTD on channel 3	This parameter allows you to set the channel (0/1/2/3) for measuring the reference temperature (calculation of the compensation value) for the assigned slot.
Reference junction E0 to reference junction E3	4 AI TC	None yes	This parameter allows you to enable the use of the reference junction.
Reference junction number	4 AI TC	1 2 Internal	This parameter allows you to assign the reference junction (1, 2) that contains the reference temperature (compensation value).

See also

Documentation on ET 200iSP (http://support.automation.siemens.com/WW/view/de/28930789/0/en)

Fundamentals of Time Stamping

Properties

Time stamping is possible with the IM 152 in customer applications using FB 62 (FB TIMESTMP).

Principle of operation

A modified input signal is assigned a time stamp and stored in a buffer (data record). If time stamped signals exists or a data record is full, a hardware interrupt is generated to the DP master. The buffer is evaluated with "Read data record". Special messages are generated for events that influence the time stamping (communication with the DP master interrupted, frame failure of time master, ...).

Parameter Assignment

With the parameter assignment you define which IM 152 user data will be monitored. For the time stamping these are digital inputs that are monitoring for signal changes.

Parameter	Setting	Description
Time stamping	• disabled • enabled	Activate the time staming for the channels of the electronics module 8 DI NAMUR.
Edge evaluation incoming event	• rising edge • falling edge	Determine the type of signal change that will be time-stamped.

Counting

Count properties

Counting functions

The 8 DI NAMUR electronics module has configurable counting functions:

- 2 x 16-bit up counters (standard counting function) or
- 2 x 16-bit down counters (standard counting function) or
- 1 x 32-bit down counter (cascading counter function)
- Setting a setpoint with the PIQ
- GATE function
- You can configure the control signals of the counters:
  - Configuration channel 0..1: "Counter", channel 2..7: "DI": Two counters are configured. The control signals of the counters are stored in the PIQ (process image output).
  - Configuration channel 0..1: "Counter", channel 2..7: "Control": Two counters are configured. The control signals of the counters are stored in the PIQ (process image output). They are also controlled by the digital inputs of the 8 DI NAMUR.

See also

Principle of operation (Page 1917)

Configuring counters (Page 1920)

Assigning parameters to counters (Page 1922)

Principle of operation

16-bit up counters (standard counting function)

The counting range is 0 to 65,535.

With each count pulse at the digital input, the count is incremented by 1. Once the count limit is reached, the counter is reset to 0 and it counts up again from this value.

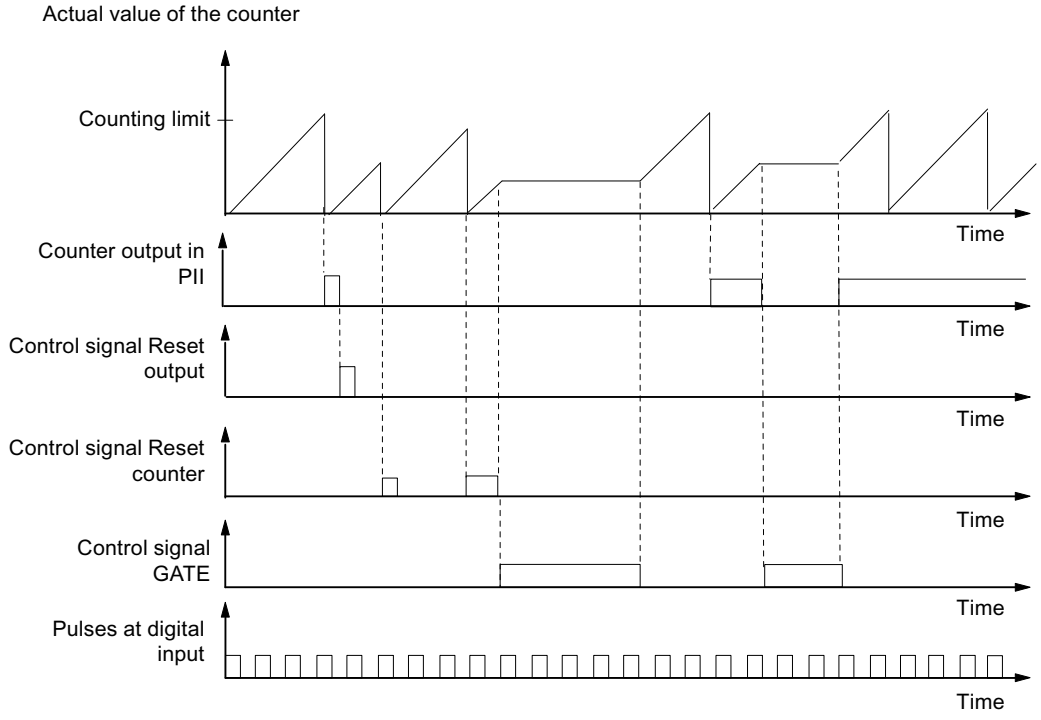
If there is counter overflow, the corresponding output is set in the PII.

A positive edge of the *Reset output* control signal resets the output in the PII. This does not affect the current count value.

In 16-bit up counting operations, the system does not set any outputs in the PIQ. These are always reset.

The positive edge of the *Reset counter* control signal sets the counter to 0 and resets the set counter output.

The *GATE* control signal pauses the counting on a positive edge. Count pulses are processed at the digital input again, but only at the negative edge. The *Reset counter* control signal is also effective when *GATE* is active.



16-bit down counters (periodic counting function)

The maximum counting range is always 65,535 to 0.

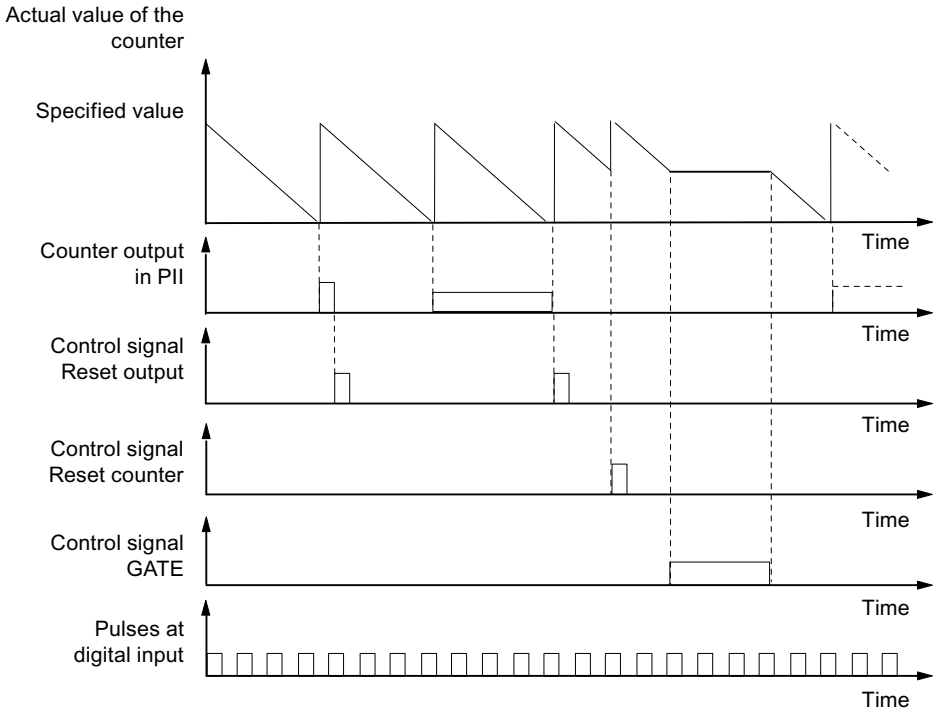
When the counter is started, the actual value is set to the selected setpoint. Each counted pulse reduced the actual value by 1. Once the actual value reaches 0, the corresponding output in the PII is turned on and the actual value is set to the selected setpoint. The counter then counts down from this value.

The positive edge of the *Reset counter* control signal resets the selected setpoint and the corresponding output in the PII.

A positive edge of the *Reset output* control signal resets the output in the PII. This does not affect the current count value.

The *GATE* control signal pauses the counting on a positive edge. At the same time, the assigned output in the PII is reset. Count pulses are processed at the digital input again, but only at the negative edge. The *Reset output* and *Reset counter* control signals are also effective when *GATE* is active.

The setpoint of the counter is set and changed using the PIQ. The setpoint is adopted on a positive edge of the *Reset counter* control signal or when the counter has reached zero.



32-bit down counter (cascading counter function)

The maximum counting range is always 4294967295 to 0.

The principle of operation is identical to that of the 16-bit down counter. Channel 1 has no function.

See also

Count properties (Page 1917)

Configuring counters

Procedure

1. Using the mouse, pull module 8 DI Namur from the hardware catalog into distributed I/O station ET 200iSP.
2. Select the required configuration (channel 0..1: "Counter", channel 2..7: "DI" or "Control"). In the module properties (inspector window), you can find this setting under "Parameters > Inputs > Configuration".

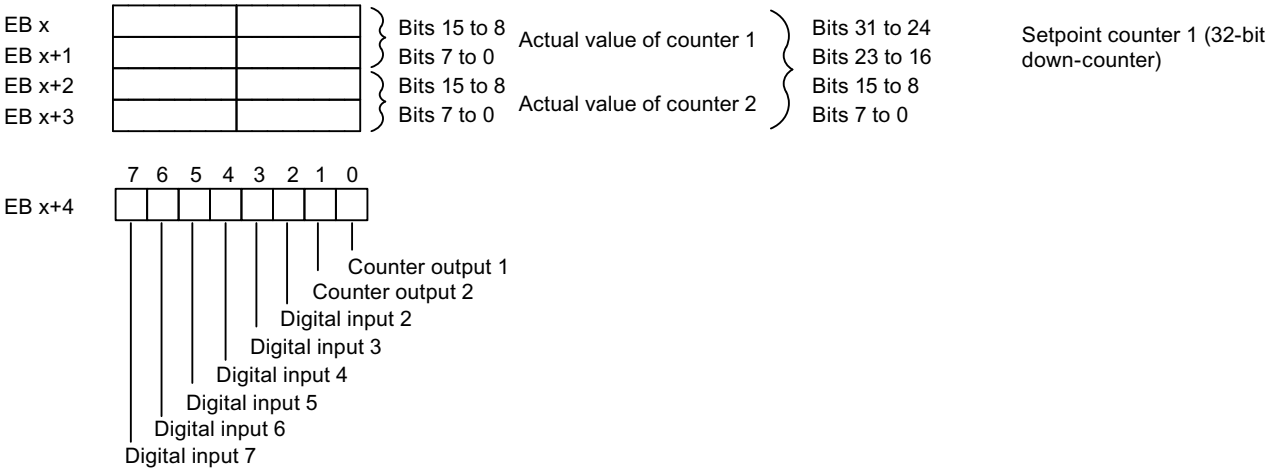
Configuration channel 0..1: "Counter", channel 2..7: "DI"

- Assignment of the digital inputs on the electronic module 8 DI NAMUR

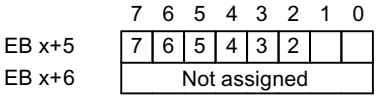
Table 10-187 Assignment of digital inputs for channel 0..1: "Counter", channel 2..7: "DI":

Digital input	Terminal	Assignment
Channel 0	1, 2	Counter 1
Channel 1	5, 6	Counter 2 (does not apply to 32-bit down counters)
Channel 2	9, 10	Digital input 2
Channel 3	13, 14	Digital input 3
Channel 4	3, 4	Digital input 4
Channel 5	7, 8	Digital input 5
Channel 6	11, 12	Digital input 6
Channel 7	15, 16	Digital input 7

- Assignment of the process image input (PII)

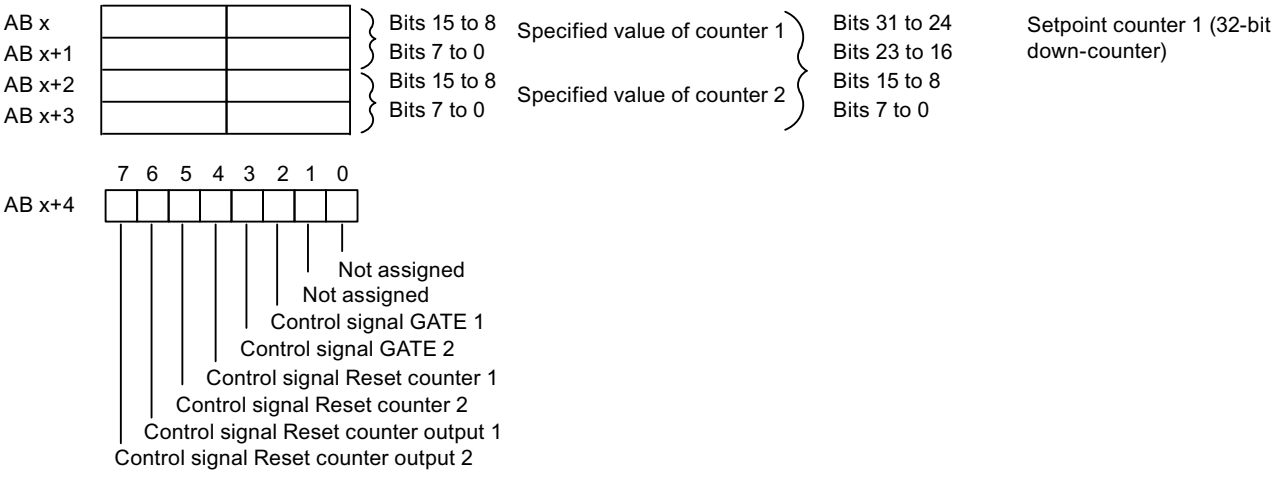


S7 format



Value status for channels 2 to 7:
1_B: Input signal is valid
0_B: Input signal is invalid

- Assignment of the process image output (PIQ)



Configuration channel 0..1: "Counter", channel 2..7: "CONTROL"

With this configuration, you can also control the counters over the digital inputs.

- Assignment of the digital inputs on electronic module 8 DI NAMUR
  For further information on input assignments, refer to the technical data for electronic module 8 DI NAMUR.

Table 10-188 Assignment of the digital inputs for 2 Count/ 6 Control

Digital input	Terminal	Assignment
Channel 0	1, 2	Counter 1
Channel 1	5, 6	Counter 2 (does not apply to 32-bit down counters)
Channel 2	9, 10	control signal *GATE 1*
Channel 3	13, 14	control signal *GATE 2*
Channel 4	3, 4	control signal *Reset counter 1*
Channel 5	7, 8	control signal *Reset counter 2*
Channel 6	11, 12	control signal *Reset counter output 1*
Channel 7	15, 16	control signal *Reset counter output 2*

- Assignment of the process image input (PII)
  Assignment is identical to configuration 0..1: "Counter", channel 2..7: "DI".
- Assignment of the process image output (PIQ)
  Assignment is identical to configuration 0..1: "Counter", channel 2..7: "DI".

See also

Count properties (Page 1917)

Assigning parameters to counters

Parameters for the counting function

Only those parameters that are relevant for the counters are explained below. These belong to the parameters of electronic module 8 DI NAMUR and depend on the selected configuration:

Table 10-189 Parameters for the counters

Parameter	Setting	Description
Sensor type counter inputs	• Channel disabled • NAMUR sensor • Single contact, no load resistance	Select the sensor for the respective counter of channels 0 or 1.
Mode for counter 1	• Standard counting function • Periodic counting function • Cascaded counting function	Select the mode for counter 1.
Mode for counter 2	• Standard counting function • Periodic counting function • Cascaded counting function	Select the mode for counter 2. This parameter is not relevant if you have set the "Mode for counter 1" parameter to "Cascaded counter function".

See also

Count properties (Page 1917)

Frequency measurement

Frequency measurement properties

Properties

The electronic module 8 DI NAMUR allows the frequencies to be measured on channel 0 and 1:

- 2 frequency meters from 1 Hz to 5 kHz
- Configurable metering window (GATE)
- The signals of the frequency meter are read in by means of the digital inputs of the electronic module.

See also

Principle of operation (Page 1923)

Configuring frequency meters (Page 1924)

Assigning parameters for the frequency meters (Page 1926)

Principle of operation

Frequency measurement

The signal frequencies are identified from the input signals of channel 0 or 1 of the electronic module. To calculate the frequency the signals are measured within a configurable gate.

The frequency is displayed as 16-bit value in fixed-point format and transferred to the PII.

The frequency meter calculates the frequency according to the follow formula:

$$\text{Frequency [Hz]} = \frac{\text{Number of rising edges at digital input}}{\text{Measuring window [s]}}$$

Exceeding the input frequency

If the input frequency exceeds 5kHz, $7FFF_H$ is reported as actual value. If the input frequency is above approx. 8 kHz it is no longer possible to display correct actual values.

See also

Frequency measurement properties (Page 1923)

Configuring frequency meters

Procedure

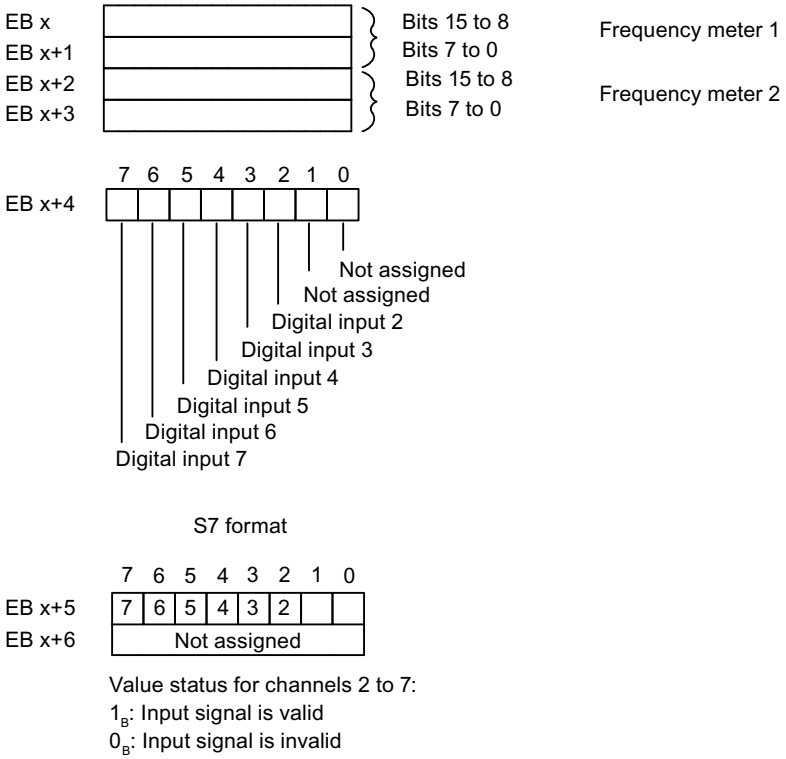
1. Using the mouse, pull module 8 DI Namur from the hardware catalog into distributed I/O station ET 200iSP.
2. Select the required configuration (channel 0..1: "Trace", channel 2..7: "DI"). In the module properties (inspector window), you can find this setting under "Parameters > Inputs > Configuration".

Configuration 0..1: "Trace", channel 2..7: "DI"

Assignment of the digital inputs on electronic module 8 DI NAMUR

Digital input	Terminal	Assignment
Channel 0	1, 2	Frequency counter 1
Channel 1	5, 6	Frequency counter 2
Channel 2	9, 10	Digital input 2
Channel 3	13, 14	Digital input 3
Channel 4	3, 4	Digital input 4
Channel 5	7, 8	Digital input 5
Channel 6	11, 12	Digital input 6
Channel 7	15, 16	Digital input 7

Assignment of process image input (PII) for configuration of channel 0..1: "Trace", channel 2..7: "DI"



Assignment of the process image output (PIQ): The PIQ is not assigned.

See also

Frequency measurement properties (Page 1923)

Assigning parameters for the frequency meters

Parameters for frequency meter

Only those parameters that are relevant for the frequency meters are explained below. These are part of the parameters of electronic module 8 DI NAMUR.

Table 10-190 Parameters for the frequency meters

Parameter	Setting	Description
Sensor type frequency inputs	• Channel disabled • NAMUR sensor • Single contact, no load resistance	Select the sensor for the relevant frequency meter for channel 0 or 1.
Measuring window (GATE)	• 50 ms • 200 ms • 1 s	Select the required measuring window for channel 0 or 1. To achieve the highest possible accuracy when metering frequencies, remember the following rules: • High frequencies (> 4 kHz): Set a low measuring window (50 ms) • Variable/medium frequencies: set medium measuring window (200 ms) • Low frequencies (< 1 kHz): Set a high measuring window (1 s)

See also

Frequency measurement properties (Page 1923)

ET 200eco PN

ET 200eco PN Distributed I/O Device

Definition

The ET 200eco PN distributed I/O device is a compact PROFINET IO device in degree of protection IP 65/66 or IP 67 and UL Enclosure Type 4x, Indoor use only.

Field of application

The fields of application of the ET 200eco PN are derived from its special properties.

- A robust design and degree of protection IP 65/66 or IP 67 make the ET 200eco PN distributed I/O device suitable in particular for use in rugged industrial environments.
- The compact design of the ET 200eco PN is particularly favorable for applications in confined areas.
- The easy handling of ET 200eco PN facilitates efficient commissioning and maintenance.

Properties

The ET 200eco PN has the following properties:

- Integrated switch with 2 ports
- Supported Ethernet services:
  - ping
  - arp
  - Network diagnostics (SNMP)
  - LLDP
- Interrupts
  - Diagnostics interrupts
  - Maintenance interrupts
- Port diagnostics
- Isochronous real-time communication
- Prioritized startup
- Device replacement without programming device
- Media redundancy
- Connection to intelligent sensors/actuators via IO link master interface module.

IO Controller

The ET 200eco PN can communicate with all IO Controllers that conform to IEC 61158.

ET 200eco PN can be configured on a CPU with advanced diagnostics.

See also

Documentation on ET 200eco PN (http://support.automation.siemens.com/WW/view/en/29999018)

Parameter description analog input

Group diagnostics

You can generally enable and disable the diagnostics function of the device with this parameter.

The "Fault" and "Parameter assignment error" diagnostics functions are always independent of the group diagnostics.

Diagnostics, missing 1L+

If you enable this parameter, the check for missing supply voltage is enabled.

Diagnostics, sensor supply short circuit

If you enable this parameter, a diagnostics event is generated if a short-circuit of the sensor supply to ground is detected and the channel is enabled. The sensor supply is monitored for connectors X1, X3, X5 and X7. It is not possible to differentiate which connector has experienced the sensor short circuit.

Interference frequency suppression

With this parameter, you set the integration time of the device, based on the selected interference frequency. Select the frequency of the supply voltage used. Interference frequency suppression **Off** means 500 Hz, which corresponds to an integration time of 2 ms for a measurement channel.

Temperature unit

Specify the unit of the temperature measurement here.

Measurement type (channel-wise)

With this parameter, you set the measurement type, for example, voltage. For any unused channels, you must select the **disabled** setting. For a disabled channel, the conversion time and integration time of the channel = 0 s and the cycle time is optimized.

Measuring range

With this parameter, you set the measuring range of the selected measurement type.

Temperature coefficient (for RTD, thermoresistor)

The correction factor for the temperature coefficients (α-value) indicates by what extent the resistance of specific material changes relatively if the temperature increases by 1 °C.

The α-values conform to EN 60751, GOST 6651, JIS C 1604 and ASTM E-1137.

The temperature coefficient depends on the chemical composition of the material.

Smoothing

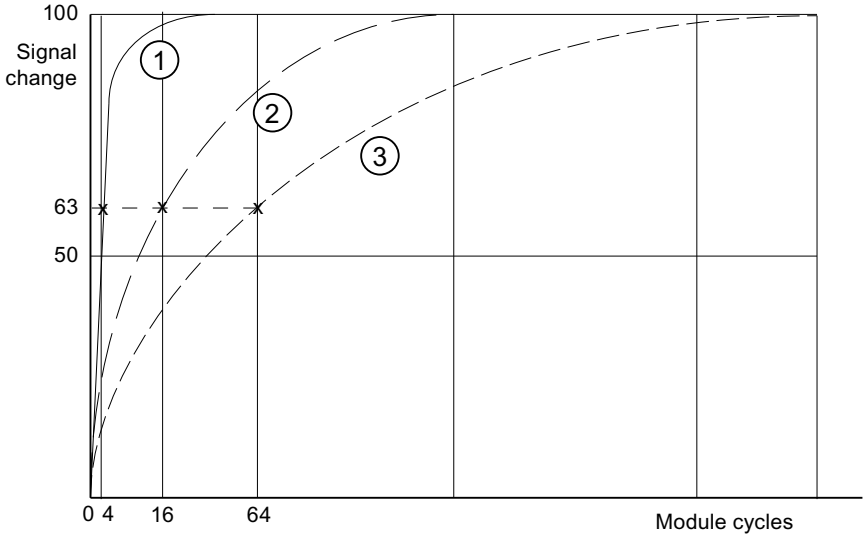
Smoothing of the analog values produces a stable analog signal for further processing. The smoothing of analog values is useful when handling wanted signals (measured values) with a slow rate of change, for example, temperature measurements.

The measured values are smoothed by digital filtering. To achieve smoothing, the device generates a mean value from a specified number of converted (digitized) analog values.

You assign a maximum of four levels for the smoothing (none, weak, medium, strong). The level determines the number of module cycles, from which the mean value is generated.

The stronger the smoothing, the more stable the smoothed analog value and the longer it takes until the smoothed analog value is applied following a signal change (see the example below).

The figure below shows the number of cycles a module requires to apply the smoothed analog value at almost 100% after a step response, based on the smoothing function settings. The figure applies to all signal changes at the analog input. The smoothing value defines the number of cycles a module requires to reach 63% of the end value of the changed signal.



① Smoothing, weak
② Smoothing, medium
③ Smoothing, strong

Diagnostics, wire break

When this parameter is enabled, the **Wire break** diagnostics event is generated when a wire break is detected.

Observe the rules outlined below to handle a wire break in the 1 to 5 V and 4 to 20 mA measuring ranges:

Parameter	Event	Measured value	Explanation
Enable wire break[1]	Wire break	$7FFF_H$	Diagnostics, **wire break**
Wire break disabled[1] Underflow enabled	Wire break	8000_H	Measured value after leaving the underrange Diagnostic message **Lower limit value undershot**
Wire break disabled[1] Underflow disabled	Wire break	8000_H	Measured value after leaving the underrange
[1] Measuring range limits for wire break detection and measuring range undershoot detection: • 1 to 5 V: At 0.296 V • 4 to 20 mA: At 1.185 mA			

Diagnostics, underflow

If you enable this parameter, the **Underflow** diagnostics event is generated when the measured value reaches the underflow range.

Diagnostics, overflow

If you enable this parameter, the **Overflow** diagnostics event is generated when the measured value reaches the overflow range.

Reference junction for thermoresistor (TC)

A difference in temperature between the measuring point and the free ends of the thermocouple (terminal point) generates a voltage between the free ends, namely the thermoelectric voltage. The value of this thermoelectric voltage is determined by the temperature difference between the measuring point and the free ends and by the type of material combination of the thermocouple. Since a thermocouple always measures a temperature difference, the free ends at the reference junction must be maintained at a known temperature in order to determine the temperature of the measuring point.

If you specify **Internal compensation**, the temperature of the measuring point in the housing of the I/O device is measured. With the **External compensation** setting, you can connect a compensation box in series in order to increase the accuracy of the temperature measurement.

Parameter description analog output

Group diagnostics

You can generally enable and disable the diagnostics function of the device with this parameter.

The "Fault" and "Parameter assignment error" diagnostics functions are always independent of the group diagnostics.

Diagnostics, missing 1L+

If you enable this parameter, the check for missing supply voltage is enabled.

Diagnostics, sensor supply short circuit

When this parameter is enabled, the system generates a diagnostics event if it detects a short-circuit of the sensor supply to ground. This diagnostics function is activated when the group diagnostics function is enabled.

Response to CPU/Master STOP

Select how the module's outputs will respond to a CPU STOP:

- Shut down
  The I/O device goes to the safe state. The process image output is deleted (=0).
- Keep last value
  The I/O device retains the last value to be output before STOP.
- Substitute value
  The I/O device outputs the value for the channel set beforehand.

Note

Make sure that the plant is always in a safe state if "Keep last value" is selected.

Type of output

With this parameter, you set the output type, for example, voltage. For any unused channels, select the **disabled** setting. For a disabled channel, the conversion time and integration time of the channel = 0 s, and the cycle time is optimized.

Output range

With this parameter, you set the output range of the selected output type.

Diagnostics, wire break (in current mode)

When this parameter is enabled, the **Wire break** diagnostics event is generated when a wire break is detected. This diagnostics event cannot be detected in the zero range.

Diagnostics, short circuit (in voltage mode)

If you enable this parameter, a diagnostics event is generated in the event of a short circuit in the output line. This diagnostics event cannot be detected in the zero range.

Diagnostics, overload

If you enable this parameter, the diagnostics event is generated in the event of an overload.

Substitute values

With this parameter, you enter a substitute value that the module is to output in CPU-STOP mode. The substitute value must be in the nominal range, overrange, or underrange.

ET 200SP

ET 200SP distributed I/O system

Definition

The ET 200SP distributed I/O system is a scalable, highly flexible distributed I/O system for connection of process signals to a central controller via a field bus.

Application area

The ET 200SP is a multi-functional distributed I/O system for various fields of application. The scalable design allows you to configure the system exactly to the specific requirements on location.

The ET 200SP is approved for degree of protection IP 20 and for installation in a control cabinet.

Structure

The ET 200SP is mounted on a mounting rail and comprises:

- An interface module which can communicate with all IO controllers that conform to the PROFINET standard IEC 61158
- Up to 32 I/O modules which can be inserted on passive BaseUnits in any combination
- A server module that completes the design of the ET 200SP.

ET 200SP with own CPU module

ET 200SP with own CPU

The ET 200SP can be equipped with its own CPU, for example with the CPU 1510SP-1 PN or CPU 1512SP-1 PN.

You can connect up to 65 modules of the S7-1500 automation system to this CPU.

Note the configuration rules of the ET 200SP (http://support.automation.siemens.com/WW/view/en/58649293).

Interface module parameters

Status bytes

Status bytes

If you enable the "Status bytes" option, 4 bytes of input data are reserved for the status of the supply voltage of each I/O module.

	7	6	5	4	3	2	1	0
Byte 0	8	7	6	5	4	3	2	1
Byte 1	16	15	14	13	12	11	10	9
Byte 2	24	23	22	21	20	19	18	17
Byte 3	32	31	30	29	28	27	26	25

I/O module slots

Bit = 0: Load voltage missing or I/O module not present

Bit = 1: Load voltage and I/O module present

Note

An inserted or missing server module always reports for the slot bit = 0.

Group diagnostics, missing supply voltage L+

Group diagnostics, missing supply voltage L+

This diagnostics is a group diagnostics that covers the supply voltage status of all I/O modules of a potential group which are defined by BaseUnits with incoming power supply (light-colored BaseUnit BU...D).

The group diagnostics is formed from the states of the supply voltage of the inserted I/O modules within the potential group.

The group diagnostics does not depend on the "Missing supply voltage L+" parameter of the I/O modules being enabled.

The server module does not influence the missing supply voltage L+ group diagnostics.

Requirements for the correct operation of the group diagnostics for missing supply voltage L+:

- I/O modules or BU covers must be inserted on the light-colored and dark-colored BaseUnits. If no I/O module is inserted on a light-colored BaseUnit, the start of this potential group will not be detected by the interface module; the I/O modules of this potential group will thus belong to the previous potential group. A supply voltage L+ group error will then be assigned to the wrong potential group.
  When an I/O module is inserted on the light-colored BaseUnit, the interface module detects the new potential group, re-evaluates the status, and reports a new group diagnostics in the case of an error.
- The server module must be inserted.
  The server module itself does not influence the missing supply voltage L+ group diagnostics.

Configuration control with ET 200SP

Operating principle

Configuration control allows you to operate various real configurations (options) with a single configuration of the distributed I/O device ET 200SP.

Configuration control provides you with the option of configuring the ET 200SP distributed I/O device with its maximum configuration and nevertheless operating it with missing modules. If missing modules are retrofitted later, no new configuration is required and the hardware configuration does not have to be reloaded either.

Using control data record 196, which is transferred to the interface module in the user program, you define a current preset configuration.

- The configured module is not present on a slot.
  - A BU cover can be inserted in this slot instead of the configured I/O module. As there is no configured module on the slot, the term "Configuration control with empty slots" is also used.
  - The module that is configured to the right of the missing module can be inserted on this slot instead of the configured module. The missing module makes the actual configuration appear pushed together. As the configured module is missing but no gap arises in the configuration, this is also referred to as a "Configuration without empty slots".
- The configuration is extended by an already configured module.
  - In the case of configuration control with empty spaces, you extend the configuration by inserting the configured module on the corresponding empty slot.
  - In the case of configuration control without empty spaces, insert the configured module on the right-hand side next to the last module of the ET 200SP.

Requirement

The CPU startup parameter "Compare preset to actual configuration" is set to Startup even if mismatch (default setting). This setting is also selected for the startup parameters of the individual modules of the ET 200SP.

Enabling configuration control

In the properties of the interface module under Module parameters > General > Configuration control, select "Enable reconfiguration of device via user program". This activates configuration control.

Control data record 196 for ET 200SP

The figure below shows the data block 196 for the configuration control of an ET 200SP with four modules.

The value "12" is in the "block_length" element.

If you configure an ET 200SP in STEP 7 with more modules, the data block will be longer. The data record for the maximum configuration with 65 modules is 134 bytes long (configuration with PN interface module).

There are two bytes in the data record for each module.

The positions of these two bytes in the data record each code a module in the original configuration with STEP 7:

- "slot_1" and "info_slot_1" (bytes 4 and 5 in the data record, see figure below) correspond to the module in slot 1 in configuration with STEP 7.
- "slot_2" and "info_slot_2" (bytes 6 and 7) correspond to the module in slot 2 in configuration with STEP 7.
- "slot_3" and "info_slot_3" (bytes 8 and 9) correspond to the module in slot 3 in configuration with STEP 7.
- etc.

"slot_x" byte

The current slot is coded by the figure that is assigned to "slot_x" (by its value). Examples:

- The value "1" in byte 4 means you are assigning the module originally inserted in slot 1 to slot 1 in the current configuration (slot_1 = 1).
- The value "2" in byte 4 means you are assigning the module originally inserted in slot 1 to slot 2 in the current configuration (slot_1 = 2).
- The value "3" in byte 4 means you are assigning the module originally inserted in slot 1 to slot 3 in the current configuration (slot_1 = 3).
- etc.
- The value "1" in byte 6 means you are assigning the module originally inserted in slot 2 to slot 1 in the current configuration (slot_2 = 1).
- The value "2" in byte 6 means you are assigning the module originally inserted in slot 2 to slot 2 in the current configuration (slot_2 = 2).
- The value "3" in byte 6 means you are assigning the module originally inserted in slot 2 to slot 3 in the current configuration (slot_2 = 3).
- etc.

If a BU cover can be inserted instead of a module, code this by adding 128 to the slot (bit 7 of the "slot_x" byte is set). Examples:

- The value "129" in slot_1 means you are also assigning the module originally inserted in slot 1 to slot 1 in the current configuration. A BU cover can also be used instead of this module. In the real plant configuration, either the module or a BU cover is inserted.
- The value "130" in slot_1 means you are assigning the module originally inserted in slot 1 to slot 2 in the current configuration. A BU cover can also be used instead of this module. In the real plant configuration, either the module or a BU cover is inserted.
- The value "131" in slot_1 means you are assigning the module originally inserted in slot 1 to slot 3 in the current configuration. A BU cover can also be used instead of this module. In the real plant configuration, either the module or a BU cover is inserted.

"info_slot_x" byte

If a new potential group is opened with the module, assign the "info_slot_x" byte the value 1 (bit 0 of the byte is set). Examples:

- The value "1" in the "info_slot_2" byte means that a new potential group is opened with module 2.
- The value "1" in the "info_slot_3" byte means that a new potential group is opened with module 3.
- The value "1" in the "info_slot_4" byte means that a new potential group is opened with module 4.

Exception: A new potential group is automatically assigned to the first module in the original configuration in STEP 7. This is not coded in the data record. You can enter any value in "info_slot_1".

You can choose any name for the components of the control data record (for example "slot_1").

Example of control data record 196 for ET 200SP

The figure below shows control data record 196 for an ET 200SP with four modules.

The module inserted in slot 2 in the configuration in STEP 7 can also be inserted in slot 2 in this configuration. It can also be inserted in slot 2 of a BU cover. Otherwise, nothing has changed compared to the original configuration.

Record_196_ET_200SP

	Name	Data type	Default value
1	block_length	USInt	12
2	block_ID	USInt	196
3	version	USInt	2
4	subversion	USInt	0
5	slot_1	USInt	1
6	info_slot_1	USInt	0
7	slot_2	USInt	130
8	info_slot_2	USInt	0
9	slot_3	USInt	3
10	info_slot_3	USInt	0
11	slot_4	USInt	4
12	info_slot_4	USInt	0

Addressing the interface module using the HW identifier

To transfer data record 196 with the instruction WRREC, you must enter the HW identifier of the IM submodule with the extension "~Head" as the input parameter for the instruction. The system constant of this HW identifier is, for example, "IO-Device_2~Head". The system constants of a selected device are, for example, displayed in the network view in the "System constants" tab. Use the corresponding value for addressing.

Readback data record 197 for ET 200SP

Readback data record 197 is used to read the actual configuration of a station (in this case, of an ET 200SP).

This readback data record allows you to check the real configuration of the ET 200SP (actual configuration). The readback data record for each configured module specifies whether or not it is actually available.

- The value "1" means that the correct module is inserted in the correct slot.
- The value "0" codes all other possibilities (wrong module, empty slot, BU cover).

Configuration details:

The configuration of the data block corresponds to the original configuration of the ET 200SP with STEP 7.

There are two bytes in the data record for each module. The position of these two bytes in the data record corresponds to the position of a module in the original configuration with STEP 7.

Sequence of bytes:

- "status_slot_1" and "reserve_1" (bytes 4 and 5 in the data record) correspond to the module in slot 1 in the configuration,
- "status_slot_2" and "reserve_2" (bytes 6 and 7) correspond to the module in slot 2 in the configuration
- "status_slot_3" and "reserve_3" (bytes 8 and 9) correspond to the module in slot 3 in the configuration,
- etc.

Example

The original configuration in STEP 7 has been changed by a control data record 196 (see example above): In the modified configuration, module 2 can either be inserted in slot 2 or be replaced by a BU cover.

The figure below shows readback data record 197 which ET 200SP outputs to indicate that there is a module in slot 2: The "status_slot_2" byte has the value "1".

The other modules are also available and are inserted in the correct slots.

Record_197_ET_200SP

	Name	Data type	Default value
1	block_length	USInt	12
2	block_ID	USInt	197
3	version	USInt	2
4	subversion	USInt	0
5	status_slot_1	USInt	1
6	reserve_1	USInt	0
7	status_slot_2	USInt	1
8	reserve_2	USInt	0
9	status_slot_3	USInt	1
10	reserve_3	USInt	0
11	status_slot_4	USInt	1
12	reserve_4	USInt	0

The figure below displays the readback data record 197 which ET 200SP outputs to indicate that a BU cover is being used in slot 2: The "status_slot_2" byte has the value "0".

The other modules are available and are inserted in the correct slots.

		Record_197_ET_200SP		
		Name	Data type	Default value
1		block_length	USInt	12
2		block_ID	USInt	197
3		version	USInt	2
4		subversion	USInt	0
5		status_slot_1	USInt	1
6		reserve_1	USInt	0
7		status_slot_2	USInt	0
8		reserve_2	USInt	0
9		status_slot_3	USInt	1
10		reserve_3	USInt	0
11		status_slot_4	USInt	1
12		reserve_4	USInt	0

Reading readback data record 197

You can read readback data record 197 from the ET 200SP with the instruction RDREC. RDREC operates asynchronously. If you call RDREC in the startup OB, you must call the instruction multiple times using a loop until the "BUSY" or "DONE" output parameter indicates that the data record has been read.

To read data record 197 with the instruction RDREC, you must enter the HW identifier of the IM submodule with the extension "~Head" as the input parameter for the instruction. The system constant of this HW identifier is, for example, "IO-Device_2~Head". The system constants of a selected device are, for example, displayed in the network view in the "System constants" tab. Use the corresponding value for addressing.

Additional information and examples

Specific examples of configuration control can be found in this application description (http://support.automation.siemens.com/WW/view/en/29430270).

Further information on ET 200SP can be found in the manuals IM 155-6 PN (http://support.automation.siemens.com/WW/view/en/73184046) and IM 155-6 DP (http://support.automation.siemens.com/WW/view/en/73098660)

You will find a library of the data records and other application examples here:

Templates for the data records (http://support.automation.siemens.com/WW/view/en/29430270)

See also

Configuration control with ET 200AL (Page 1981)

Configuration control for ET 200SP with integrated ET 200AL modules (Page 1939)

Configuration control for ET 200SP with integrated ET 200AL modules

ET 200SP with integrated ET 200AL modules

Configuration control for ET 200SP and ET 200AL is described in separate help topics. See the links under "See also".

The steps outlined also apply to configuration control of an ET 200SP with integrated AL modules. The procedure is different for control data record 196 and readback data record 197.

The following help text describes control data record 196 and the readback data record 197 for an ET 200SP expanded with ET 200AL modules.

Control data record 196

The two figures below show parts of control data record 196 for a configuration of an ET 200SP expanded with ET 200AL modules.

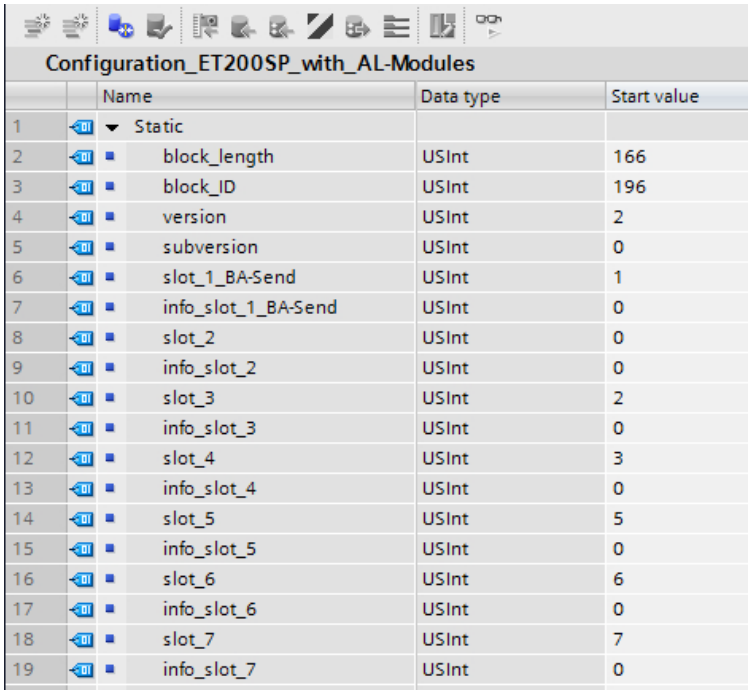
This configuration is given as an example:

- The module "BA Send 1xFC" is in slot 1 "slot_1". This module allows you to integrate ET 200AL modules into an ET 200SP. In our configuration example, 16 AL modules are connected to the BA-Send (maximum configuration). If a BA-Send is being used, this module must be inserted in slot 1.
- All slots from 2 to 64 are assigned ET 200SP modules.
- A server module is inserted in slot 65.
- There are 16 AL modules in slots 66 to 81.

The ET 200SP with integrated AL modules, originally configured with STEP 7, is now to be reconfigured from the user program.

The new configuration has the following properties:

- The "BA Send 1xFC" module is inserted in "slot_1_BA-Send" (fixed setting).
- Module 2 "slot_2" is not available in the modified configuration (value "0").
- Module 3 "slot_3" is in slot 2 in the modified configuration (value "2").
- Module 4 "slot_4" is in slot 3 in the modified configuration (value "3").
- All modules from slot 5 to slot 81 are operated in the original configuration with STEP 7.

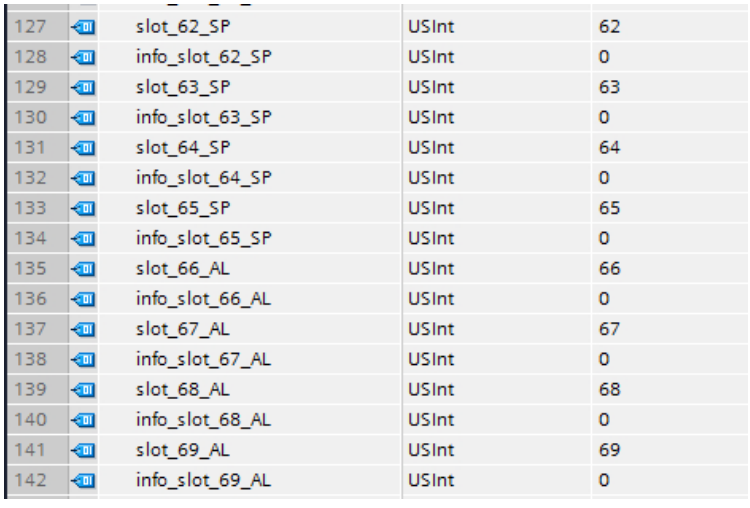


Configuration_ET200SP_with_AL-Modules

	Name	Data type	Start value
1	Static		
2	block_length	USInt	166
3	block_ID	USInt	196
4	version	USInt	2
5	subversion	USInt	0
6	slot_1_BA-Send	USInt	1
7	info_slot_1_BA-Send	USInt	0
8	slot_2	USInt	0
9	info_slot_2	USInt	0
10	slot_3	USInt	2
11	info_slot_3	USInt	0
12	slot_4	USInt	3
13	info_slot_4	USInt	0
14	slot_5	USInt	5
15	info_slot_5	USInt	0
16	slot_6	USInt	6
17	info_slot_6	USInt	0
18	slot_7	USInt	7
19	info_slot_7	USInt	0

The components of control data record 196 (figure above):

- block_length: Note the length of the control data record here; in the example: 166 (bytes). The length of the control data block is calculated using the formula "2 x number of slots + 4".
- block_ID: Enter the figure 196 here.
- version: The ET 200SP uses version 2 of control data record 196.
- subversion: The ET 200SP uses subversion 0 of control data record 196.



127	slot_62_SP	USInt	62
128	info_slot_62_SP	USInt	0
129	slot_63_SP	USInt	63
130	info_slot_63_SP	USInt	0
131	slot_64_SP	USInt	64
132	info_slot_64_SP	USInt	0
133	slot_65_SP	USInt	65
134	info_slot_65_SP	USInt	0
135	slot_66_AL	USInt	66
136	info_slot_66_AL	USInt	0
137	slot_67_AL	USInt	67
138	info_slot_67_AL	USInt	0
139	slot_68_AL	USInt	68
140	info_slot_68_AL	USInt	0
141	slot_69_AL	USInt	69
142	info_slot_69_AL	USInt	0

The components of control data record 196 (figure above):

- slot_65_SP: This byte relates to the server module in the ET 200SP rack. It ends the backplane bus of the ET 200SP.
- From "slot_66_AL" come the 16 configured ET 200AL modules: Our configuration example does not change the configuration with STEP 7. The byte "slot_66_AL" has the value "66", the byte "slot_67_AL" has the value "67", the byte "slot_68_AL" has the value "68", etc.

Definition of control data record 196

A control data record 196 containing a slot assignment is defined for configuration control.

Byte	Component	Value	Description	
0	block_length	e.g. 166 for maximum configuration with 65 ET-200SP modules and 16 ET 200AL modules (with DP interface module, maximum of 33 ET 200SP modules and 16 ET 200AL modules)	The length of the data record is calculated using the formula: 4 + "number of modules" x 2	Header
1	Block ID	196	ID for control data record 196	
2	version	2	Version 2 of control data record 196	
3	subversion	0	Subversion 0 of control data record 196	
4	slot_1_BA-Send	Real slot for SP module 1 Possible value: 1	When AL modules are integrated in ET 200SP, the module "BA Send 1xFC" must always be inserted in slot 1.	1st slot for SP modules Assignment for the configured SP module 1 to a real slot
5	info_slot_1_BA-Send	0 or 1	The value "1" means that a new potential group is opened with this module. (Not evaluated in this byte)	

6	slot_2	Real slot for SP module 2 Possible values: 2 to 65 (not 66 to 81, reserved for AL modules) 0 (if the configured module 2 is not available)	The configured SP module 2 can be inserted in any real slot from 2 to slot 65 (2 to 33 for a DP interface module). Slots 66 to 81 are for AL modules (34 to 49 with DP interface module).	2nd slot for SP modules Assignment for the configured SP module 2 to a real slot
7	info_slot_2	1	The value "1" means that a new potential group is opened with this module. A new potential group must always be opened in this byte as BA-Send cannot open a new potential group.	
8	slot_3	Real slot for SP module 3 Possible values: 2 to 65 (not 66 to 81, reserved for AL modules) 0 (if the configured module 3 is not available)	The configured SP module 3 can be inserted in any real slot from 2 to slot 65 (2 to 33 for a DP interface module). Slots 66 to 81 are for AL modules (34 to 49 with DP interface module).	3rd slot for SP modules Assignment for the configured SP module 3 to a real slot
9	info_slot_3	1	The value "1" means that a new potential group is opened with this module. .	
:	:	:	:	:
132	slot_65	Real slot for SP module 65 Possible values: 2 to 65 (not 66 to 81, reserved for AL modules) 0 (if the configured module 65 is not available)	The configured SP module 65 can be inserted in any real slot from 2 to slot 65 (2 to 33 for a DP interface module). Slots 66 to 81 are for AL modules (34 to 49 with DP interface module)	65th Slot for SP modules Assignment for the configured SP module 65 to a real slot
133	info_slot_65	0 or 1	The value "1" means that a new potential group is opened with this module (the value is not evaluated in this slot).	

134	slot_66	Real slot for AL module 1 Possible values: 66 to 81 (not 1 to 65, reserved for SP modules) 0 (if the configured AL module 1 is not available)	The configured AL module 1 can be inserted in any slot from 66 to slot 81 (34 to 49 for PROFIBUS).	1st Slot for AL modules Assignment for the configured AL module 1 to a real slot
135	info_slot_66	-	Reserve	
:	:	:	:	:
164	slot_81	Real slot for AL module 16 Possible values: 66 to 81 (not 1 to 65, reserved for SP modules) 0 (if the configured AL module 16 is not available)	The configured AL module 16 can be inserted in any slot from 66 to slot 81 (34 to 49 for DP interface module).	16th Slot for AL modules Assignment for the configured AL module 16 to a real slot
165	info_slot_81	-	Reserve	

Rules

- If the "BA Send 1xFC" module is being used, it must be inserted in slot 1.
- ET 200SP modules are inserted in slots 2 to 65 (slots 2 to 33 for DP interface module).
- AL modules are inserted in slots 66 to 81 (slots 34 to 49 for DP interface module).
- If you expand an ET 200SP with ET 200 AL modules, the 1st AL module is always coded in bytes 134 and 135 of the control data record, the 2nd AL module in bytes 136 and 137, etc., even if not all SP slots are to be occupied by SP modules. Non-assigned SP slots are coded with the value "0".

Error messages

The following error messages are returned if an error occurs when writing control data record 196:

Table 10-191 Error messages

Error code	Meaning
16#80A2	DP protocol error on layer 2 Indicates that a data record has not been acknowledged due to the system.
16#80B1	Invalid length; the length information in data record 196 is not correct.
16#80B5	Configuration control parameters not assigned.

Error code	Meaning
16#80B2	Invalid slot: The configured slot is not assigned.
16#80B8	Parameter error; module signals invalid parameters.
16#80C5	DP slave or module not available. Indicates that a data record has not been acknowledged due to the system.

Readback data record 197 for ET 200SP with AL modules

The actual configuration of an ET 200SP with AL modules can be checked with readback data record 197.

Data record 197 largely corresponds to readback data record 197 for ET 200SP without AL modules; however, it is longer as the additional AL modules also need to be coded.

There are two bytes in the data record for each module. The position of these two bytes in the data record codes a module in the original configuration with STEP 7.

In the figure below:

- The components "status_slot_1" and "reserve_1" (bytes 4 and 5 in the data record) correspond to the module in slot 1 in the configuration with STEP 7,
- "status_slot_2" and "reserve_slot_2" (bytes 6 and 7) correspond to the module in slot 2,
- "status_slot_3" and "reserve_slot_3" (bytes 8 and 9) correspond to the module in slot 3,
- etc.

The following data record is structured for configuration with 65 SP modules and 16 AL modules. The value "166" therefore appears in the "block_length" element of the data record.

If you configure an ET 200SP in STEP 7 with fewer AL modules, the data block will be shorter.

If you use fewer SP modules in a configuration, this has no effect on the length of data record 197 (with an expansion of ET 200SP with ET 200AL modules).

The "reserve_x" component of readback data record 197 is reserved for future applications.

You can choose any name for the components of the readback data record (for example "status_slot_1").

The figure below shows the start of readback data record 197 for reading the actual configuration of an ET 200SP with AL modules.

DB_read_back_ET_200SP_with_AL

	Name	Data type	Start value
1	Static		
2	block_length	USInt	166
3	block_ID	USInt	197
4	version	USInt	2
5	subversion	USInt	0
6	status_slot_1	USInt	1
7	reserve_slot_1	USInt	0
8	status_slot_2	USInt	1
9	reserve_slot_2	USInt	0
10	status_slot_3	USInt	1
11	reserve_slot_3	USInt	0
12	status_slot_4	USInt	1
13	reserve_slot_4	USInt	0
14	status_slot_5	USInt	1
15	reserve_slot_5	USInt	0
16	status_slot_6	USInt	1
17	reserve_slot_6	USInt	0

It does not show the components "status_slot_7" to "reserve_81" (maximum configuration of an ET 200SP with PN interface module) or "status_slot_7" to "reserve_slot_49" (maximum configuration of an ET 200SP with DP interface module).

Meaning of "status_slot_x":

- The value "1" in status_slot_x means that module x is inserted in the correct slot
- The value "0" codes all other possibilities (wrong module, empty slot, BU cover).

The figure below shows part of feedback data record 197 for reading the actual configuration of an ET 200SP with AL modules (and PN interface module). ET 200SP modules are inserted up to slot 65, and then the AL modules. For example, a value of "1" in the byte "status_slot_66_AL" means that the 1st AL module actually exists in the plant and is inserted in the correct slot.

128	status_slot_62_SP	USInt	1
129	reserve_slot_62_SP	USInt	0
130	status_slot_63_SP	USInt	1
131	reserve_slot_63_SP	USInt	0
132	status_slot_64_SP	USInt	1
133	reserve_slot_64_SP	USInt	0
134	status_slot_65_SP	USInt	1
135	reserve_slot_65_SP	USInt	0
136	status_slot_66_AL	USInt	1
137	reserve_slot_66_AL	USInt	0
138	status_slot_67_AL	USInt	1
139	reserve_slot_67_AL	USInt	0
140	status_slot_68_AL	USInt	1
141	reserve_slot_68_AL	USInt	0
142	status_slot_69_AL	USInt	1
143	reserve_slot_69_AL	USInt	0

Reading feedback data record 197

You can read the feedback data record 197 from the ET 200SP with the RDREC instruction. RDREC operates asynchronously. If you call RDREC in the startup OB, you must call the instruction multiple times using a loop until the "BUSY" or "DONE" output parameter indicates that the data record has been read.

Further information and examples

Specific examples of configuration control can be found in this application description (http://support.automation.siemens.com/WW/view/en/29430270).

Further information on ET 200SP can be found in the manuals for IM 155-6 PN (http://support.automation.siemens.com/WW/view/en/73184046) and IM 155-6 DP (http://support.automation.siemens.com/WW/view/en/73098660)

Further information on ET 200AL is available here (http://support.automation.siemens.com/WW/view/en/89254863).

See also

Configuration control with ET 200SP (Page 1934)

Configuration control with ET 200AL (Page 1981)

Expanding ET 200SP with ET 200AL modules

Introduction

The ET 200SP is a distributed I/O system for installation in a control cabinet.

The system can be expanded with modules of the ET 200AL series with IP65/67 degree of protection. ET 200AL modules can be mounted on site, for example on a machine.

Slot rules

The following rules apply to the use of ET 200AL modules in an ET 200SP with interface module:

- The "BA-Send 1xFC" module must be inserted in slot 1 of the ET 200SP if the ET 200SP is to be expanded with ET 200AL modules.
- The ET 200AL modules must be configured with no gaps.

The procedure detailed further on covers the use of ET 200AL modules in an ET 200SP with interface module.

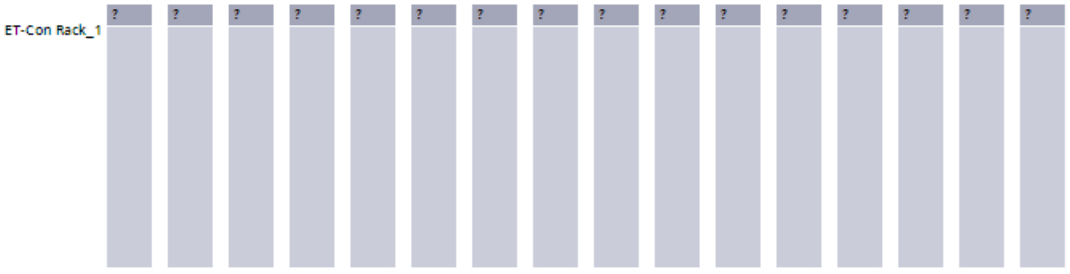
The following, different, rules apply to the use of ET 200AL modules in an ET 200SP with CPU:

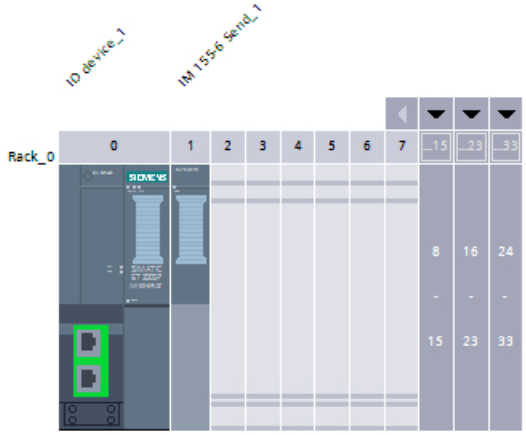
- The "BA-Send 1xFC" module can be located in slots 2 to 4 of the ET 200SP CPU.
- A CM DP communication module can be inserted as CPU submodule, but then occupies slot 2.

Procedure

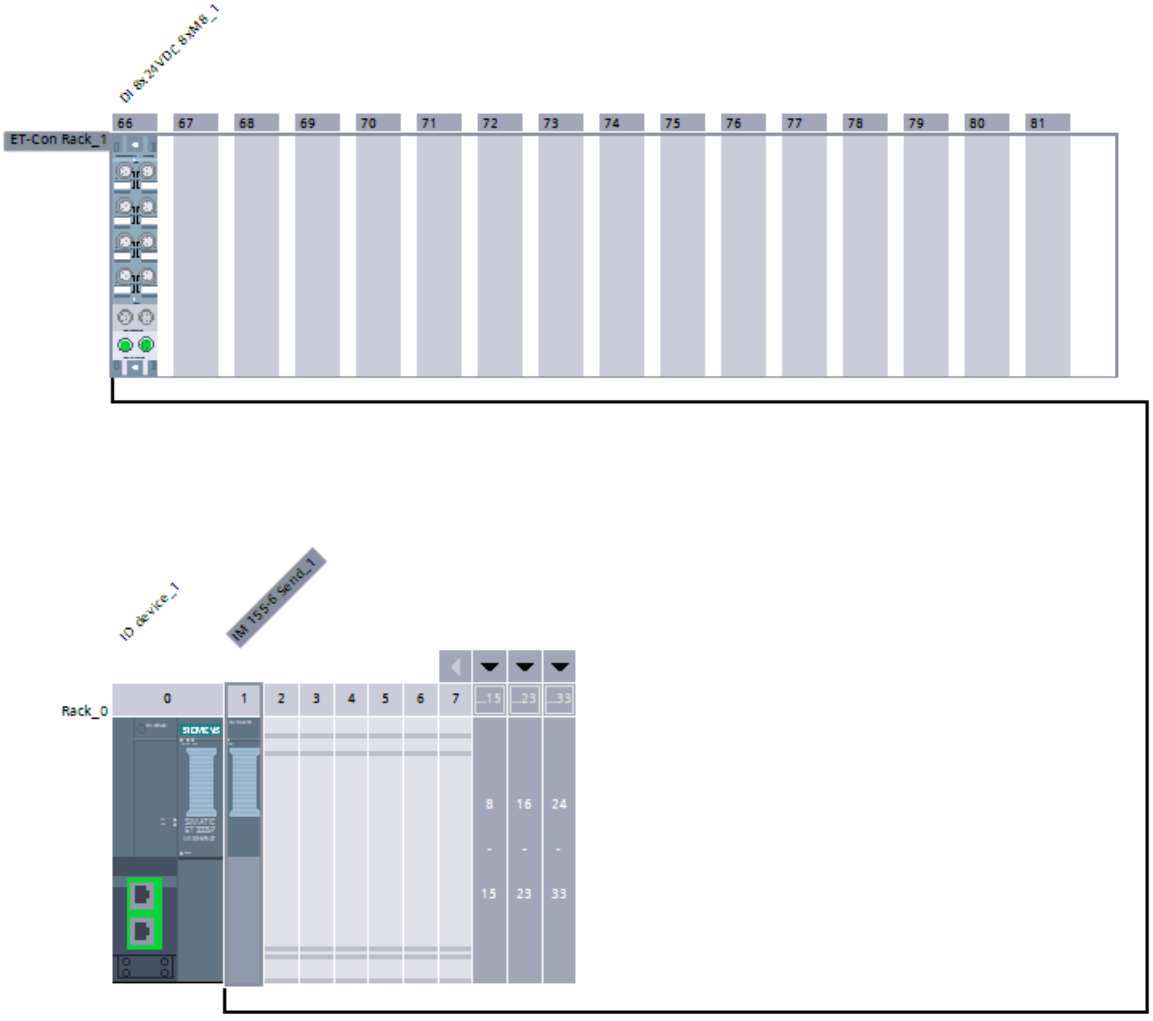
Follow the steps below to configure an ET 200SP with interface module with ET 200AL modules:

1. Drag an interface module (PROFINET or PROFIBUS) from the ET 200SP series to the network view.
2. Go to the device view. You do this by clicking twice on the icon of the module you have just inserted.
3. Insert the module "BA Send 1xFC" into slot 1 of the ET 200SP.
   STEP 7 now generates an ET-Connection rack with 16 slots for ET 200AL modules (figure below).
   An ET-Connection rack is a virtual rack that sets the order of the connected ET 200AL modules.
   Question marks are displayed above the slots as an ET 200AL module has yet to be connected to BA-Send.

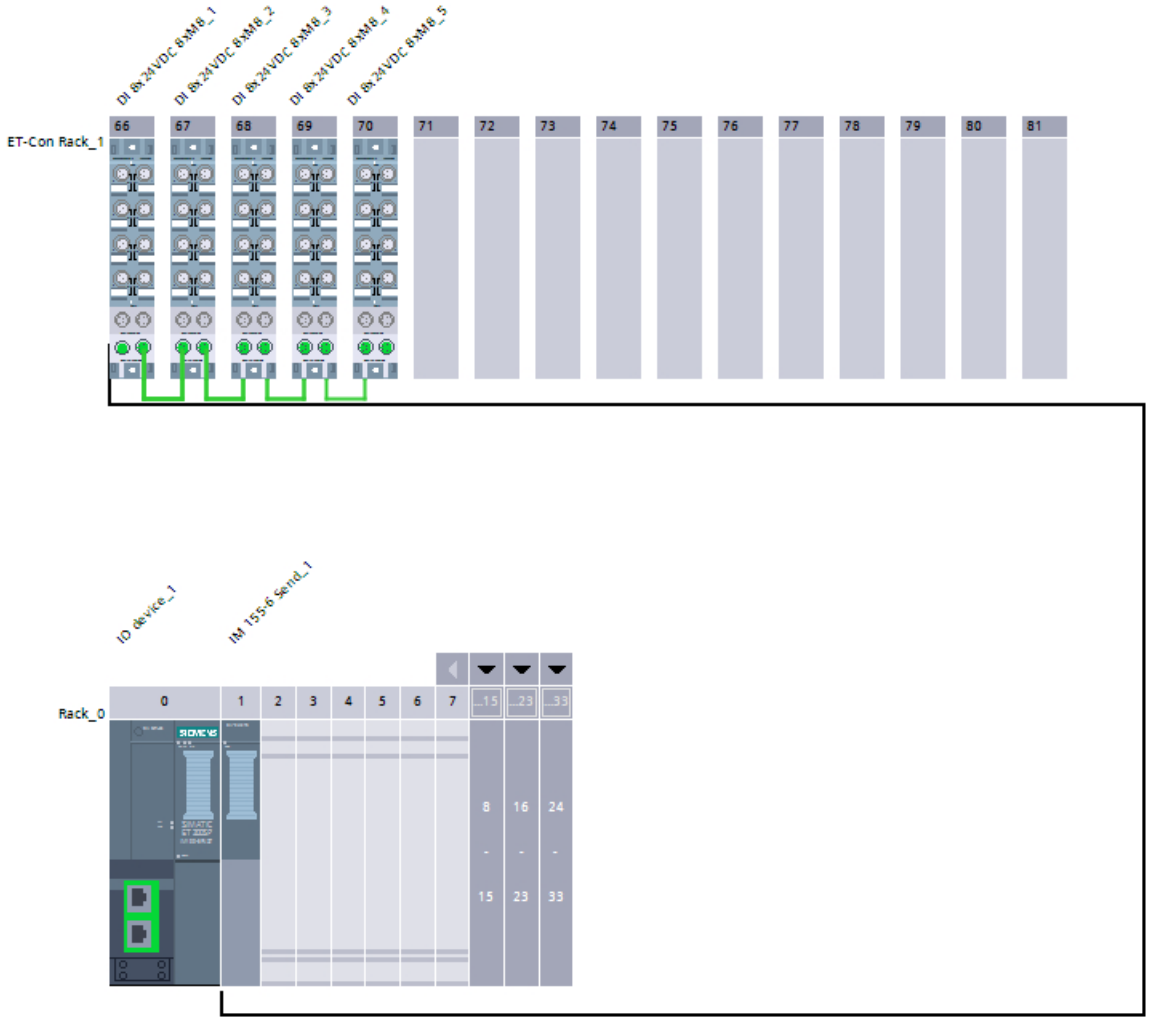




4. From the hardware catalog (subfolder ET 200AL in the ET 200SP folder), select the first ET 200AL module to be connected to the ET 200SP: Drag and drop this module to the 1st slot in the ET connection rack.
   From this module, STEP 7 generates a line to the "BA-Send 1xFC" module and allocates the slot numbers 66 to 81 (figure below).
   If you configure the ET 200SP with a DP interface module, STEP 7 assigns the slot numbers 34 to 49 for the ET 200AL modules.



5. Now drag all other ET 200AL modules into the free slots in the ET-Connection rack.
   STEP 7 automatically inserts the ET-Connection between the individual ET 200AL modules (green loops).
   In the configuration below, five ET 200AL modules are connected in series.



6. Complete the configuration of the ET 200SP: Drag all ET 200SP modules from the hardware catalog to the slots in the ET 200SP.
   Five ET 200SP modules are plugged in the configuration example below: The PN interface module in slot 0, the module "BA-Send 1xFC" in slot 1, one input module each in slots 2 and 3, and a server module in slot 4:



See also

Configuration control with ET 200AL (Page 1981)

ET 200AL distributed I/O system (Page 1974)

Output module parameters

Substitute value reaction

Substitute value reaction

In the ET 200SP, the substitute value reaction is executed by the IO controller per slot.

The respective output reacts according to its set substitute value reaction:

- "Turn off"
- "Output substitute value"
- "Keep last value"

This substitute value reaction is triggered in the following cases:

- IO controller in STOP
- Controller failure (connection interruption)
- Firmware update
- Reset to factory settings
- More than one I/O module withdrawn simultaneously
- Disable the IO device
- Station stop
  - Missing server module
  - More than one I/O module withdrawn simultaneously
  - At least one I/O module is inserted on the wrong BaseUnit

Note

Reducing a configuration

If you reduce the configuration of the ET 200SP and download the configuration to the CPU, the modules which are no longer configured but still present retain their original substitute value reaction. This applies until the supply voltage on the BaseUnit BU...D or on the interface module is turned off.

Input module parameters

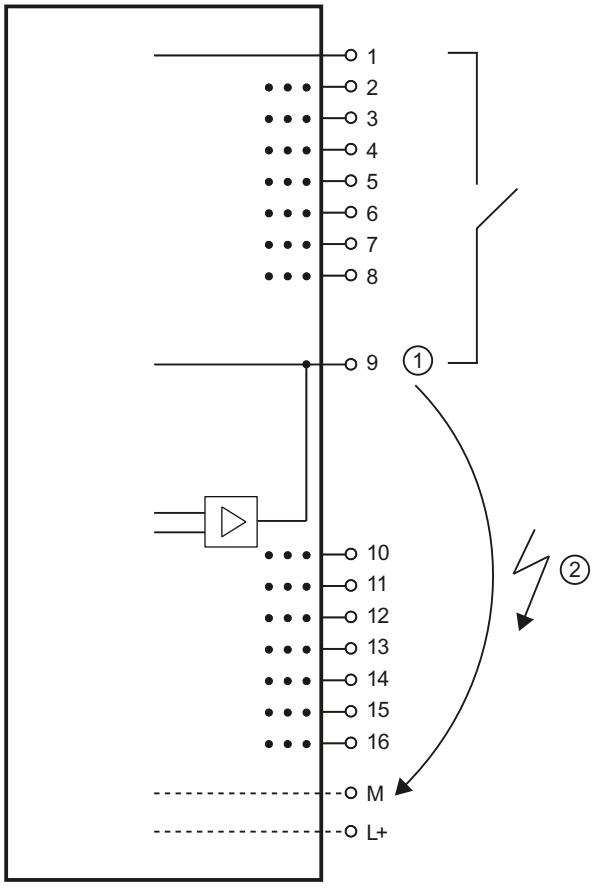
Parameters of the digital input modules

Diagnostics missing supply voltage L+

Enabling of the diagnostics for missing or insufficient supply voltage L+.

Diagnostics short-circuit to ground

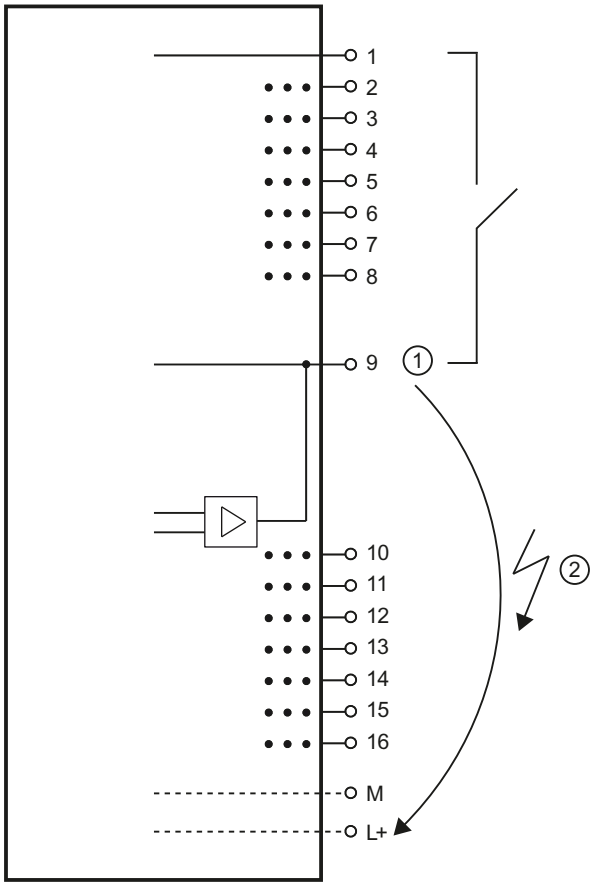
Enabling of the diagnostics if a short-circuit of the actuator supply to ground occurs.



① Encoder supply
② Short-circuit

Diagnostics short-circuit to L+

Enabling of the diagnostics if a short-circuit of the encoder supply to L+ occurs.



① Encoder supply
② Short-circuit

Diagnostics wire break

Enabling diagnostics if the line to the encoder is interrupted.

Operating mode

Determines whether a channel is enabled or disabled.

Pulse extension (only High Feature modules)

Pulse extension is a function for changing a digital input signal. A pulse at a digital input is extended to at least the configured length. If the input pulse is already longer than the configured length, the pulse is not changed.

Pulse extension is started whenever the state of the input signal changes and no pulse extension is active for this channel.

Potential group of the left module / new potential group

Specifies whether the I/O module is located on a base unit with supply voltage feed (new potential group) or whether it is it located on a base unit without supply voltage feed (in which case it belongs to the potential group of the left module).

Parameters of the analog input modules

Missing supply voltage L+

Enabling of the diagnostics, with missing or too little supply voltage L+.

Reference junction (AI 4xRTD/TC 2-/3-/4-wire HF)

A BaseUnit with internal temperature sensor (BU..T) or the channel 0 of the I/O module can be used as reference junction, provided this has been configured as "Thermal resistor Pt100 climatic range Celsius".

A possible parameter assignment is shown below (see also Information about reference channel mode (Page 1957)):

Table 10-192 RTD channel

Setting	Description
No reference channel operation	Temperature value at channel 0 can be used as module-wide reference value if the parameters of the other channels are assigned accordingly.
Reference channel of Group x	The channel acts as sender for the reference junction temperature of Group x. Distribution is performed via the interface module.

Table 10-193 TC channel

Setting	Description
Reference channel of the module	The corresponding TC channel uses the channel 0 of the same module as reference junction temperature. This must be set as "Thermal resistor Pt 100 climatic range Celsius" and "No reference channel operation"; otherwise, reference junction diagnostics is triggered.
Internal reference junction	The reference junction temperature is read by an internal temperature sensor on the BaseUnit. Reference junction diagnostics is triggered if there is a wrong BaseUnit type.
Reference channel of Group x	With the setting "TC" (thermocouple...), the channel acts as receiver for the reference junction temperature of Group x.
Fixed reference temperature	No temperature compensation occurs. The linearization is executed with an assumed reference junction temperature of 0 °C.

Overflow

Enabling of the diagnostics if the measured value exceeds the overflow range.

Underflow

Enabling of the diagnostics if the measured value falls below the underflow range.

Wire break

Enabling of the diagnostics if the module has no current flow or too low a current flow for the measurement on the corresponding configured input.

Smoothing

The individual measurements are smoothed using digital filtering. The smoothing can be set in 4 stages, whereby the smoothing factor k multiplied by the cycle time of the I/O module corresponds to the time constant of the smoothing filter. The larger the smoothing, the larger the time constant of the filter.

The following figure shows the step response for the various smoothing factors depending on the number of module cycles.

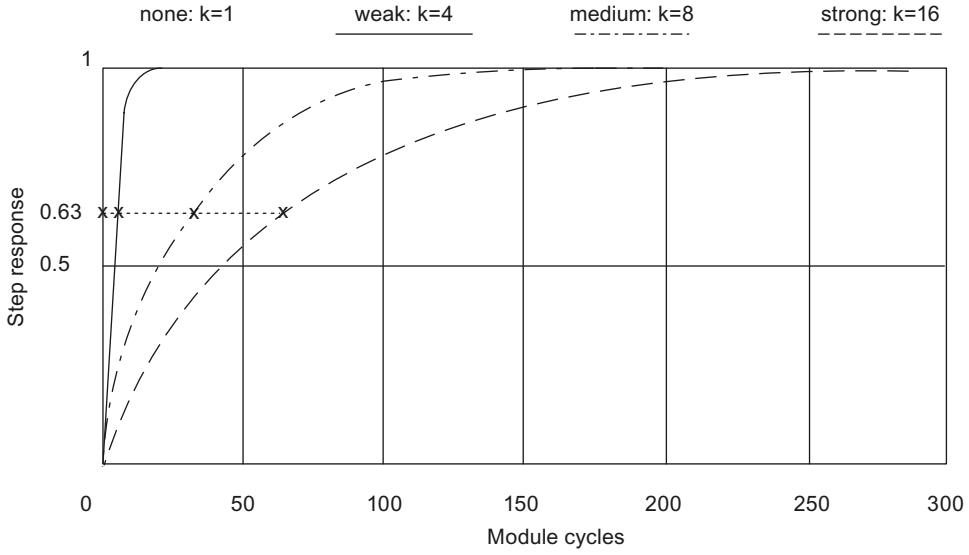


Figure 10-49 Smoothing with AI 4×RTD/TC 2-/3-/4-wire HF

Interference frequency suppression

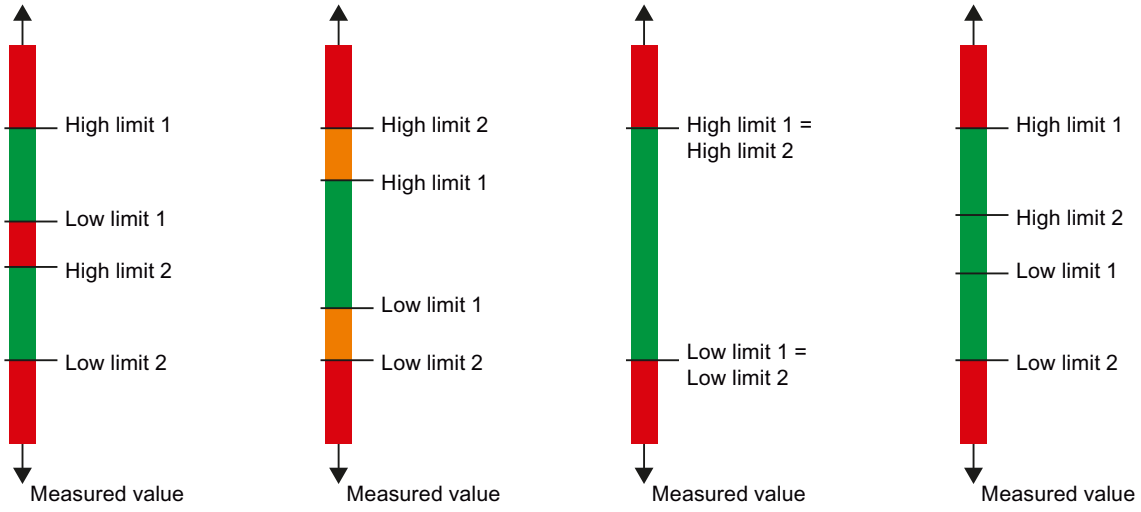
With analog input modules, suppresses the disturbance caused by the frequency of the AC network used.

The frequency of the AC network may interfere with measured values, particularly for measurements within low voltage ranges and when thermocouples are being used. This parameter is used to define the predominant power frequency of the system.

Hardware interrupt limits

If the high limit 1/2 or the low limit 1/2 is violated, the module triggers a hardware interrupt.

Below are some examples for the selection of the limits 1 and 2.



Low limit 1/2

Specify a threshold whose undershoot triggers a hardware interrupt.

High limit 1/2

Specify a threshold whose overrange triggers a hardware interrupt.

Potential group of the left module/New potential group

Specifies whether the I/O module is located on a BaseUnit with supply voltage infeed (new potential group) or on a BaseUnit without supply voltage feed (in which case, it belongs to the potential group of the left module).

Temperature coefficient (measurement type thermoresistor)

The correction factor for the temperature coefficient (α value) defines the relative rate of change of the resistance of a specific material at a temperature rise of 1 °C.

The temperature coefficient depends on the chemical composition of the material. In Europe, only one value is used per sensor type (default value).

The further values facilitate a sensor-specific setting of the temperature coefficient and enhance accuracy.

See also

Special features of AI 4xRTD/TC 2-/3-/4-wire HF (Page 1965)

Information about reference channel mode

An RTD/TC module of the ET 200SP works in reference channel mode when a channel sends the reference temperature to other channels of the station. The receiving channels use the reference temperature for temperature compensation when measuring with thermocouples.

Structure and use of thermocouples

The thermocouple consists of two wires with different metals or alloys that are welded together at the end. The weld is called measuring point.

The other end of the two wires is open. This end is called reference junction.

A thermal voltage, which depends on the temperature at the measuring point, occurs between the two metals/alloys at the measuring point. Other thermal voltages also occur at the reference junction - with transition from thermocouple to copper lines, for example - that falsify the actual measured value and need to be compensated. No compensation is required for a reference junction temperature of 0 °C.

Different methods are used to compensate the reference junction temperature:

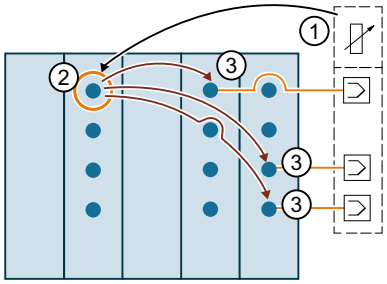
- Fixed reference temperature: The reference junction is permanently set to a specific temperature, for example, to 0 °C by an icewater bath (no compensation required).
- Internal reference junction: The reference junction is the terminal of the BaseUnit on which the analog module is plugged. If you select the "Internal reference junction" compensation type, you have to use BaseUnits with integrated temperature measurement to compensate the reference junction temperature. These BaseUnits have the designation "BU..T". The module records the temperature at the reference junction and uses this to determine the actual temperature at the measuring point.
- Reference channel of Group x: An external thermal resistor records the temperature at the reference junction for the Group x (group of channels within a station). The actual temperature at the respective measuring point can be determined as a result. One external thermal resistor is required for each group. The thermal resistors are each connected to one channel of an analog module. These channels are called senders (for the temperature at the reference junction).
  The mode of operation and settings are described in the section "Station-wide distribution of reference temperature".
- Reference channel of the module: The mode of operation is comparable with "Reference channel of Group x". You connect an external thermal resistor to channel 0 of the module to measure the temperature at the reference junction. Other channels of the same module use this reference temperature for temperature compensation.
  The mode of operation and settings are described in the section "Module-wide distribution of reference temperature".

You can find information about the structure and mode of operation of a thermocouple in the Analog value processing (http://support.automation.siemens.com/WW/view/en/67989094) manual.

Station-wide distribution of the reference temperature

You can record the temperature at the reference junction x using a thermal resistor on a channel (sender of the reference temperature) and send it to other channels within a station (receivers). All channels that receive the temperature at a reference junction x form the Group x.

For each group, you assign parameters to exactly one channel as sender of the reference temperature.



① Thermal resistor at the reference junction

② Channel records temperature at the reference junction and sends it to the other channels within a station (sender of Group x). The temperature value is used for the compensation of the reference junction temperature.

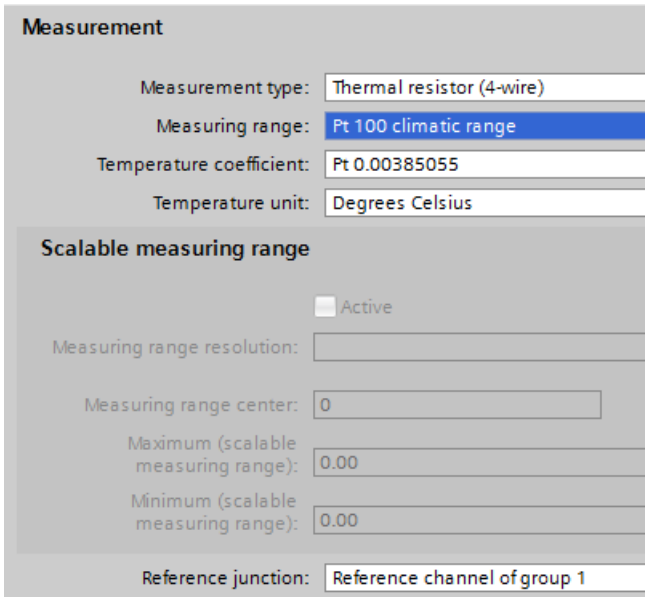
③ Channels of the Group x receive the temperature at the reference junction (receivers)

Parameter assignment of a channel as reference channel (sender for Group 1)

Parameter assignment is described below using Group 1 as example:

1. Open the project in STEP 7
2. Select the required analog module (RTD/TC) in the device view.
3. Then select a channel that is to work as sender of the reference junction temperature.
   The following settings are required:
   "Measurement type": "Thermal resistor", for example, "Thermal resistor (4-wire connection)"
   "Measuring range": "Pt 100 climatic range"
   "Temperature unit": "Degrees Celsius"
   "Reference junction": "Reference channel of Group 1"

The following figure shows the parameter assignment.

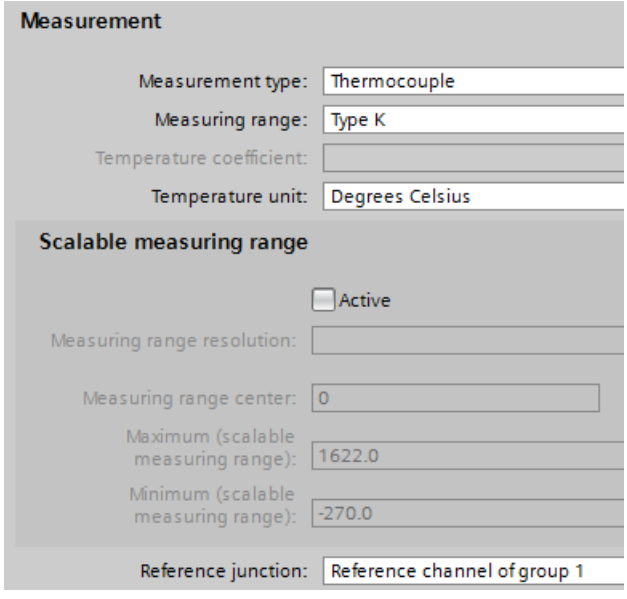


The channel configured in this way (thermal resistor measurement type) works as reference channel of Group 1 and sends the measured temperature to all channels (thermocouple measurement type) that are configured as receivers of Group 1.

In the next section, you will learn how to assign parameters for channels that are receivers of Group 1.

Parameter assignment of a channel as receiver of Group 1

The figure below shows the parameter assignment of a channel that receives the temperature at the reference junction of Group 1.

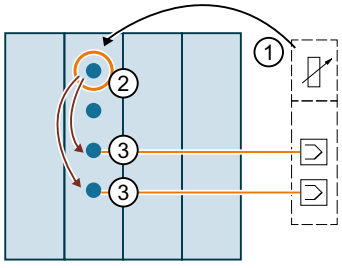


The following settings are required:

- "Measurement type": "Thermocouple"
- "Reference junction": "Reference channel of Group 1"

Module-wide distribution of the reference temperature

You can record the temperature on a reference junction using the channel 0 of the module and use the temperature value for the channels 1, 2, 3... of this module. The recorded temperature value is not sent to the channels of the other modules within the station (no reference channel mode).



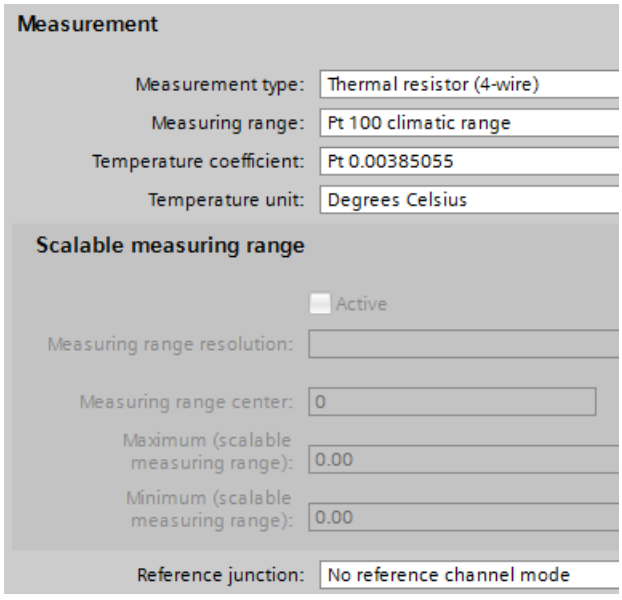
① Thermal resistor at the reference junction

② Channel records temperature at the reference junction, "Reference junction" parameter: "No reference channel mode"

③ Channels of the same module use the temperature value for the compensation of the reference junction temperature for the measurement with thermocouples, "Reference junction" parameter: "Reference channel of the module"

Parameter assignment of channel 0 as reference channel of the module

The figure below shows the parameter assignment of channel 0 of a module that is to be used for recording the temperature at the reference junction:

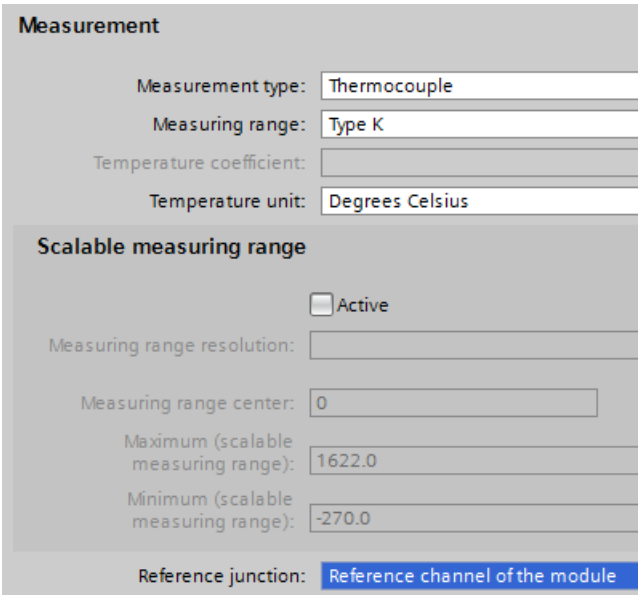


The following settings are required:

- "Measurement type": "Thermal resistor", for example, "Thermal resistor (4-wire connection)"
- "Measuring range": "Pt 100 climatic range"
- "Temperature unit": "Degrees Celsius"
- "Reference junction": "No reference channel mode"

Parameter assignment of a channel that uses channel 0 as reference channel

The figure below shows how parameters have to be assigned for a channel that uses the channel 0 of this module as reference channel for temperature compensation.



The settings below are required for the channels of the module that compensate the temperature of the reference junction using channel 0:

- "Measurement type": "Thermocouple"
- "Reference junction": "Reference channel of the module"

See also

Parameters of the analog input modules (Page 1954)

Information about the Oversampling function

High-speed analog modules (HS) are available to meet high performance and speed requirements. The main characteristics of these HS analog modules compared to Standard analog modules (ST) is their shorter cycle times. To achieve this goal, the input and output modules are equipped with components with extremely short throughput and conversion times. In addition, the entire architecture of the modules is designed for faster signal processing.

HS analog modules convert the output of measured values and output values at the same time. Each channel within the module has its own A/D or D/A converter. This means the cycle time is basically the conversion time and independent of the number of activated channels. This is true for analog input modules as well as analog output modules. This means HS modules can be used in fast isochronous mode.

Apart from isochronous mode, the HS analog modules also provide benefits in non-isochronous (free-running) mode. Due to the fast processing of the process signals, HS analog modules are able to detect changes in the process values more quickly and to respond to

these events with the appropriate program blocks (for example, hardware interrupt or cyclic interrupt organization blocks).

Isochronous mode

Isochronous mode refers to synchronous coupling

- Of signal acquisition and output via the distributed I/O
- Of signal transmission via PROFIBUS or PROFINET
- Of program processing with the constant bus cycle time of the PROFIBUS or PROFINET.

The result is a system that acquires its input signals in constant time intervals, processes them and outputs the output signals. Isochronous mode guarantees reproducible and defined process reaction times as well as equidistant and synchronous signal processing with distributed I/O.

The bus system and the I/O modules work synchronously with configured isochronous mode. The transmitted input and output data are linked to an "isochronous task" in the CPU. As a result, the data of a cycle are always consistent. All data of a process image belong together logically and in time. Jitter in the user program caused by the acquisition of outdated values is therefore almost impossible.

Even fast processes can be perfectly controlled by the exact timely reproducibility of all processes. Isochronous mode thus contributes to high control quality and hence to greater manufacturing precision. While possible fluctuations of the process reaction times are drastically reduced. The time-assured processing can be utilized to improve machine cycle times. Shorter cycle times increase the processing speed and help to lower production costs.

Oversampling

The use of the oversampling function in analog input or analog output modules requires an isochronous configuration.

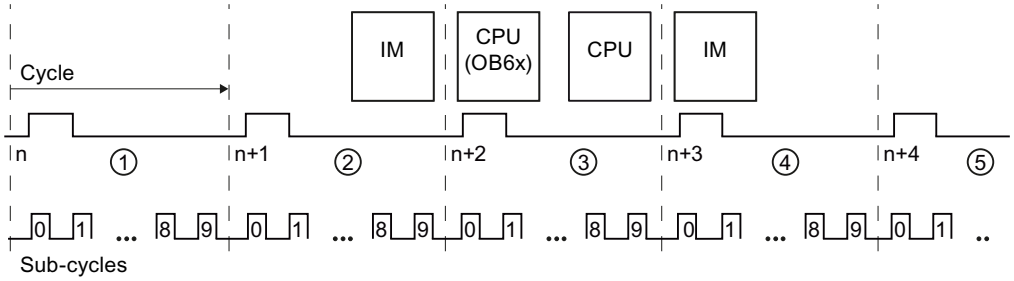
With analog input modules, the set send clock is divided into time-equidistant sub-cycles. The send clocks can be subdivided into 2 to 16 sub-cycles. Each sub-cycle reads in a measured value. The read-in measured values of a data cycle are copied to the interface module (IM) in the next send clock and are then available to the processing CPU one clock later.

With analog output modules, the set send clock is also divided into time-equidistant sub-cycles. The send clocks can be subdivided into 2 to 16 sub-cycles. Each sub-cycle returns an output value. The output values are copied to the interface module by the CPU within the same send clock and are written to the process one send clock later.

The read-in and output values are transmitted in the user data of the analog module. In this way, the address space of the module is extended from 2 bytes of user data per channel to 16 x 2 bytes of user data per channel (with 16 sub-cycles). If you subdivide the send clock into fewer than 16 sub-cycles, the unused addresses are assigned the error value 0x7FFF during input. For output, the values of the unused addresses are ignored.

Because the sub-cycles have to be within a send clock, oversampling needs an additional clock to copy the data to the interface module, unlike the 3-cycle model of isochronous mode. The result is a 5-cycle model.

Isochronous mode



① Send clock is divided into sub-cycles which record the measured value (in this case: 10 sub-cycles)

② Measured values are copied to the IM

③ Measured values are processed and output values determined

④ Output values are copied by the CPU to the IM

⑤ Output values are written to the process

Higher sampling rates through oversampling

Due to the configured modules, the IO device has a minimum possible update time. Within this update period, the IO device/IO controller of the PROFINET IO system is supplied once with new data.

The following applies with respect to the channels of a single standard module in the I/O device: The shortest update time ("sampling rate") is exactly one send clock.

If you want to shorten the update time for the channels of a standard module, you have to shorten the send clock. Because of the properties of the involved components and the structure of the I/O system, this is only possible to a certain extent (e.g., down to 0.25 ms).

Modules with oversampling, however, offer the option of further reducing the update time ("sampling rate") for their channels without having to shorten the send clock for the entire IO device at the same time.

The subdivision of the send clock into time-equidistant sub-cycles enables the processing of faster processes through higher sampling rates.

Example

In practice, the use of oversampling makes sense when the isochronous system works with only one specific send clock (for example, 1 ms) due to the modules used and when faster sampling of the process values is required. By using oversampling and subdividing the send clock into 4 sub-cycles, for example, you can sample the process values in intervals of 250 µs.

Configuring oversampling

Enable the option "Isochronous mode" in the IO device used, and set the corresponding parameters ("Send clock", etc.).

With the distributed analog input modules (e.g. AI 2xU/I 2,4-wire HS), you specify the number of sub-cycles using the "Sampling rate" parameter.

With the distributed analog output modules (e.g. AQ 2xU/I HS), you specify the number of sub-cycles using the "Sampling rate" parameter.

If, for example, you configure a "Sampling rate" of 4 "Values/cycle" for a send clock of 1 ms, the send clock is subdivided into 4 sub-cycles and the process values are sampled at intervals of 250 µs.

Reference

You can find additional information in the manuals on the high-speed analog modules and in the Analog value processing (http://support.automation.siemens.com/WW/view/en/67989094) function manual.

Special features of AI 4xRTD/TC 2-/3-/4-wire HF

Use of Cu10 sensors

- Select "3-wire thermal resistor" and "Cu10" in the parameter assignment.
- Wire the Cu10 sensor using 3-wire connection technology.
- An automatic, internal compensation of the line resistance of the missing measuring line takes place during operation.

Note

To ensure optimum line compensation for Cu10, please note the following:

- The sum of cable resistance and measuring resistance must not exceed 31 Ω.
- Cable resistance should not exceed 8 Ω if you want to use the temperature range up to over 312 °C.
  Example: A 200 m long copper cable with 0.5 mm^2 core cross-section has approx. 7 Ω.
  A lower cross-section reduces the permissible cable length accordingly.

Use of PTC resistors

PTCs are suitable for temperature monitoring of or as thermal protective equipment for complex drives or transformer windings.

- Select "2-wire resistor" and "PTC" in the parameter assignment.
- Connect the PTC using 2-wire technology.
- Use type A PTC resistors (PTC thermistors) in accordance with DIN/VDE 0660, Part 302.
- If the "Over-/underflow" diagnostics is enabled, a "low limit violation" diagnostics which shows a short circuit is generated for resistance values < 18 Ω.
- Sensor data on PTC resistance:

Table 10-194 Use of PTC resistors

Property	Technical specifications	Note
Switching points	**Reaction to rising temperature**	
	< 550 Ω	**Normal range:** • SIMATIC S7: bit 0 = "0", bit 2 = "0" (in the PII)
	550 Ω to 1650 Ω	**Prewarning range:** • SIMATIC S7: bit 0 = "0", bit 2 = "1" (in the PII)
	< 1650 Ω	**Response range:** • SIMATIC S7: bit 0 = "1", bit 2 = "0" (in the PII)
	Reaction to falling temperature	
	< 750 Ω	**Response range:** • SIMATIC S7: bit 0 = "1", bit 2 = "0" (in the PII)
	750 Ω to 540 Ω	**Prewarning range:** • SIMATIC S7: bit 0 = "0", bit 2 = "1" (in the PII)
	< 540 Ω	**Normal range:** • SIMATIC S7: bit 0 = "0", bit 2 = "0" (in the PII)
(TNF-5) °C (TNF+5) °C (TNF+15) °C Measuring voltage Voltage on the PTC	max. 550 Ω min. 1330 Ω min. 4000 Ω max. 7.5 V	RRT= rated response temperature

- Assignment in the process image inputs (PII) with SIMATIC S7

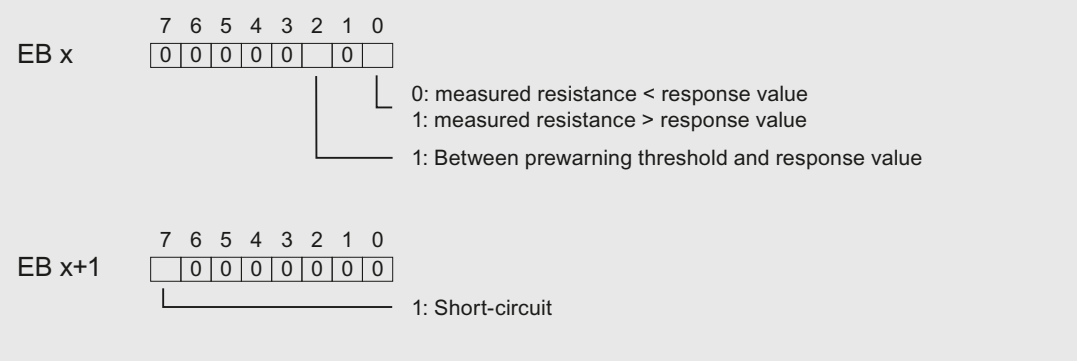


Figure 10-50 Assignment in the process image inputs (PII)

- Notes on programming

Note

Only the bits 0+2 are relevant for the evaluation in the process image inputs. You can use the bits 0+2 to monitor the temperature, for example, of a motor.

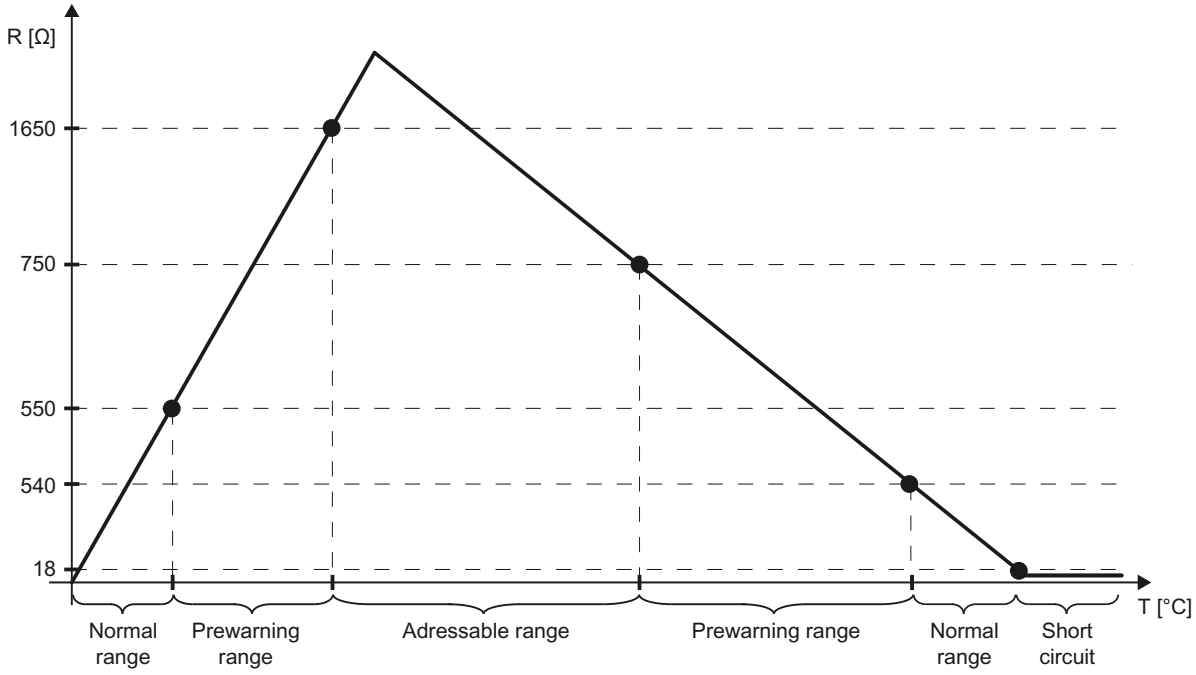
The bits 0+2 in the process image inputs have no latching function. When you are assigning parameters, take into consideration that a motor, for example, starts up in a controlled manner (via an acknowledgment).

Bits 0+2 can never be set simultaneously, but are instead set consecutively.

For safety reasons, always evaluate the diagnostic entries of the AI 4×RTD/TC 2-/3-/4-wire HF, as no measurement is possible if I/O modules are unplugged, if the supply voltage of the I/O module has failed, or if there is a wire break or short circuit of the measuring lines.

Example

The following diagram shows the temperature variation and the associated switching points.



See also

Parameters of the analog input modules (Page 1954)

What you should know about the scalable measuring range (Page 1968)

What you should know about the scalable measuring range

The scalable measuring range

The scalable measuring range is a section from the temperature measuring range of an analog input module (for example, the ET 200SP module "AI 8xRTD/TC 2-wire HF").

In this section, a higher resolution of the measured values is possible, comparable with a magnifier that allows a subsection to be viewed in greater detail.

The scalable measuring range is supported for the following measurement types:

- Thermal resistor (RTD) standard
- Thermocouple

The scalable measuring range is not available with the following measurement types:

- Voltage
- Resistance
- Thermal resistor climatic

Position and resolution of the scalable measuring range

The position and resolution of the scalable measuring range can be set (scaled):

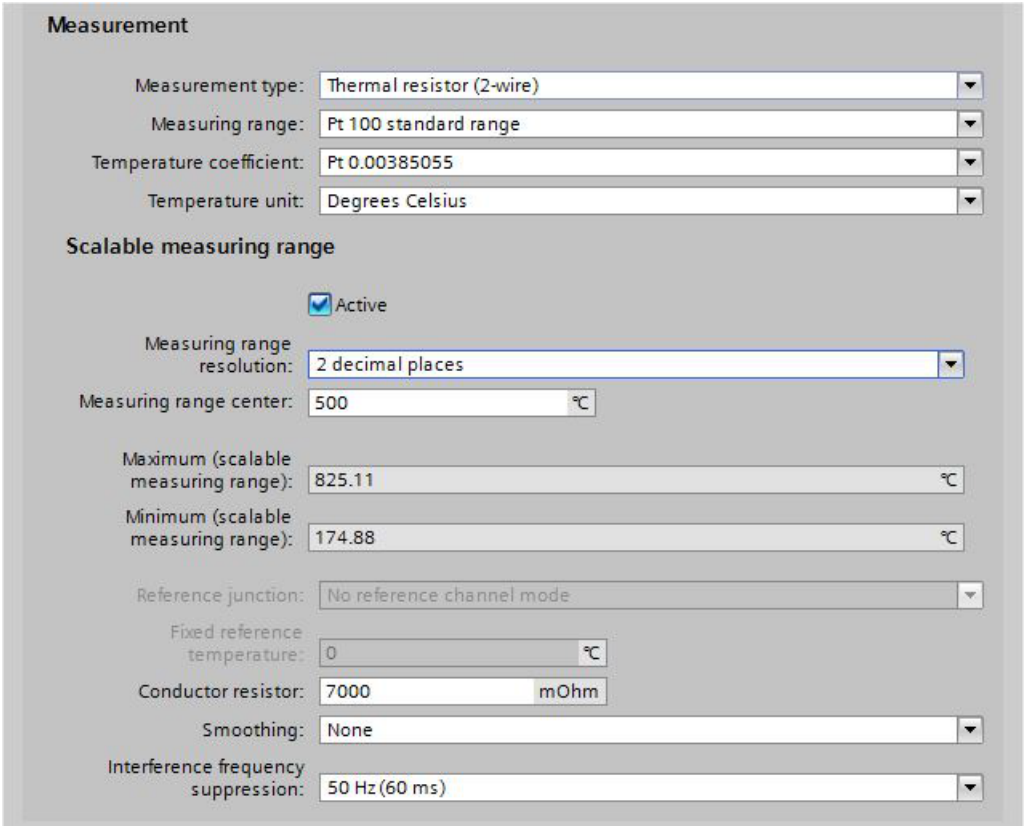
- **Position:** The scalable measuring range can be moved over the entire standard measuring range. This allows you to select the temperature range for which your application requires a higher resolution.
  Exception: The scalable measuring range cannot be moved to the extent that it enters the overflow or underflow of the standard measuring range (Clipping).
  You select the position of the scalable measuring range with the "Measuring range center" parameter (see figure below).
- **Resolution:** The following values can be set:
  - 2 decimal places (0.01 °C)
  - 3 decimal places (0.001 °C)

  You set the resolution with the "Measuring range resolution" parameter (figure below).

Example of a parameter assignment

The following figure shows a parameter assignment for the ET 200SP module "AI 8xRTD/TC 2-,3-,4-wire HF".

In STEP 7, you can find parameters in the properties box via General > AI 4 > Inputs > Channel 0 to channel 3.



Conductor resistor:

The "Conductor resistor" parameter in the parameter assignment above is enabled only if the "Thermal resistor (2-wire terminal)" measurement type was selected.

Here, enter the value for the resistance of the connecting cable of the thermal resistor: A 200 m long copper cable with 0.5 mm^2 wire cross-section, for example, has a resistance value of seven ohms.

Measuring range resolution:

In the parameter assignment above, a resolution of 0.01 °C was selected (measuring range resolution "2 decimal places").

Measuring range center:

The measuring range center is set to 500 °C.

With a resolution of 0.01 °C, this results in a scalable measuring range from 174.88 °C to 825.11 °C.

At a resolution of 0.01 °C, the scalable measuring range covers 650.23 °C.

Maximum (scalable measuring range):

This value represents the high limit of the scalable measuring range. In the example above, 825.11 °C.

The value is calculated by STEP 7 (at a resolution of 0.001 °C, the high limit is at 532.511 °C, see figure below).

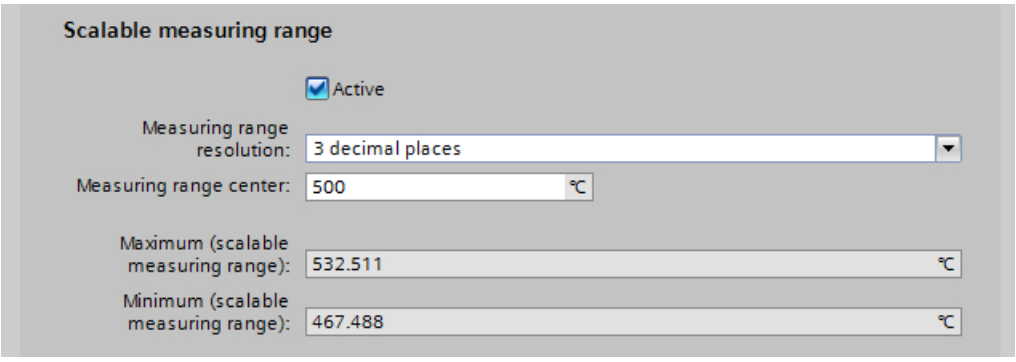
Minimum (scalable measuring range):

This value represents the low limit of the scalable measuring range. In the example above, 174.88 °C.

The value is calculated by STEP 7 (at a resolution of 0.001 °C, the low limit is at 467.488 °C; see figure below).

Higher resolution:

The following figure shows a parameter assignment with a resolution of 0.001 °C (otherwise the same example as in the figure above):



At a resolution of 0.001 °C, the scalable measuring range is between 467.488 and 532.511 °C and covers 65.023 °C (a tenth of the measuring range at a resolution of 0.01 °C).

Standard measuring range with 0.1 °C resolution

The following table shows the standard measuring range for thermal resistors of the type "Pt 100", values in degrees Celsius.

Pt 100 standard in °C (1 digit = 0.1 °C)	Decimal values	Hexadecimal values	Ranges
> 1000.0	32767	7FFF	Overflow
1000.0 : 850.1	10000 : 8501	2710 : 2135	Over range
850.0 : -200.0	8500 : -2000	2134 : F830	Nominal range
-200.1 : -243.0	-2001 : -2430	F82F : F682	Under range
< -243.0	-32768	8000	Underflow

The standard measuring range is the basis for the scalable measuring range.

You can set the measuring range center within the nominal range (-200 °C to 850 °C, see table above).

For temperatures below and above the set measuring range center, you then obtain measured values with a higher resolution.

The width of this range around the measured value center depends on the selected resolution.

Scalable measuring range 0.01 °C and 0.001 °C resolution

The scalable measuring range is identified by the following value ranges:

Scalable measuring range	Measuring range resolution (values in °C)		Hexadecimal values
	2 decimal places	3 decimal places	
Overflow	> 325.11	> 32.511	7FFF
High limit	325.11	32.511	7EFF
Measuring range center	0	0	0
Low limit	325.11	-32.511	8100
Underflow	< -325.11	< -32.511	8000

The maximum and minimum of the scalable measuring range depend on the selected resolution:

- 2 decimal places, resolution of 0.01 °C:
  The high limit is 325.11 °C above the measuring range center you have set.
  The low limit is 325.11 °C below the measuring range center you have set.
  This means that the scalable measuring range is 650.22 °C around the measuring range center.
- 3 decimal places, resolution of 0.001 °C:
  The high limit is 32.511 °C above the measuring range center you have set.
  The low limit is 32.511 °C below the measuring range center you have set.
  This means that the scalable measuring range is 65.022 °C around the measuring range center.

Calculation of the temperature

You calculate the temperature value by adding the value you receive from the module to the measuring range center.

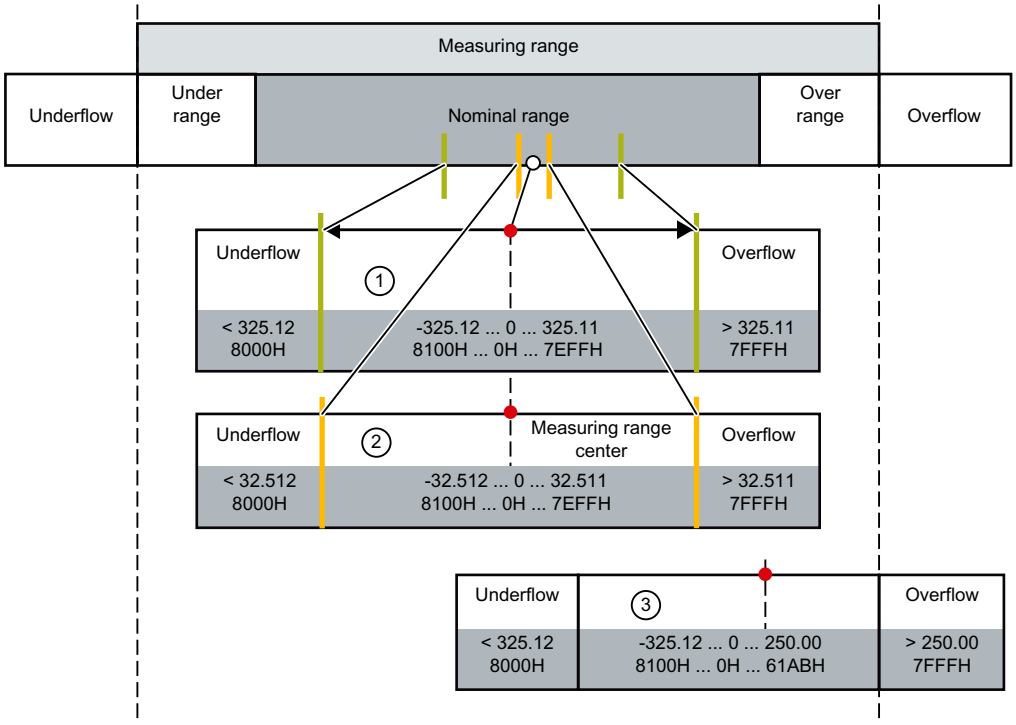
Example:

- You have set the measuring range center 500 °C (see example in the section "Example of a parameter assignment"). For the resolution, you selected "2 decimal places".
- From the module, you receive the hexadecimal value "0100" in S7 format:

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

- The hexadecimal value "0100" corresponds to the decimal value 256.
- Since you have selected a resolution of 0.**01** °C, the number 256 corresponds to the temperature value "2.**56** °C".
- You now add 500 °C and 2.56 °C and obtain the measured value 502.56 °C.

Scalable measuring range in the standard measuring range



① Scalable measuring range with 2 decimal places, temperature values in S7 format.

② Scalable measuring range with 3 decimal places, temperature values in S7 format.

③ Scalable measuring range cut off at the overflow of the standard measuring range ("Clipping").

The sum of the measuring range center (for example 750 °C) and the measured value returned by the module must not extend into the overflow of the standard measuring range. For this reason, in the example above, the maximum value that the module can return is limited to 250 °C

Clipping

STEP 7 limits the maximum of the scalable measuring range so that the sum of the measured value center and the maximum measured value that the module can return is not located in the overflow of the standard measuring range. In the same way, STEP 7 restricts the minimum of the scalable measuring range.

See also

Special features of AI 4xRTD/TC 2-/3-/4-wire HF (Page 1965)

ET 200AL

ET 200AL distributed I/O system

SIMATIC ET 200AL

The SIMATIC ET 200AL distributed I/O system is a scalable and highly flexible, distributed I/O system for connecting process signals to a superordinate control with a field bus.



Properties

- Connection to PROFINET or PROFIBUS or integration into ET 200SP / ET 200SP CPU
- Up to 32 modules on an ET 200AL
- Integration into ET 200SP / ET 200SP CPU: Up to 16 AT modules can be connected to an ET 200AL
- Connection of modules via ET-Connection
- Spatially separated mounting possible
- Module widths of 30 and 45 millimeters
- Degree of protection IP65/IP67
- Suitable for temperatures from -25 to +55 °C and accelerations up to 5 g.
- Installation in all positions
- Color coding of the cables and connections
- CA-compliant labeling of the interfaces
- PROFIenergy integrated
- Configuration control
- Connection of sensors and actuators using M8 and M12 connection system

Area of application

The SIMATIC ET 200AL distributed I/O system is especially well suited for use in tight spaces, moving applications and for assembly and handling technique. Thanks to its scalable construction, you have the option precisely customize its configuration to your on site needs.

The SIMATIC ET 200AL distributed I/O system features protection type IP65/IP67 and is suited for distributed use on a machine or assembly line.

Structure

The SIMATIC ET 200AL distributed I/O system is made up of the following components:

- Interface modules (PROFINET/PROFIBUS)
- Digital and analog I/O modules
- Communications module

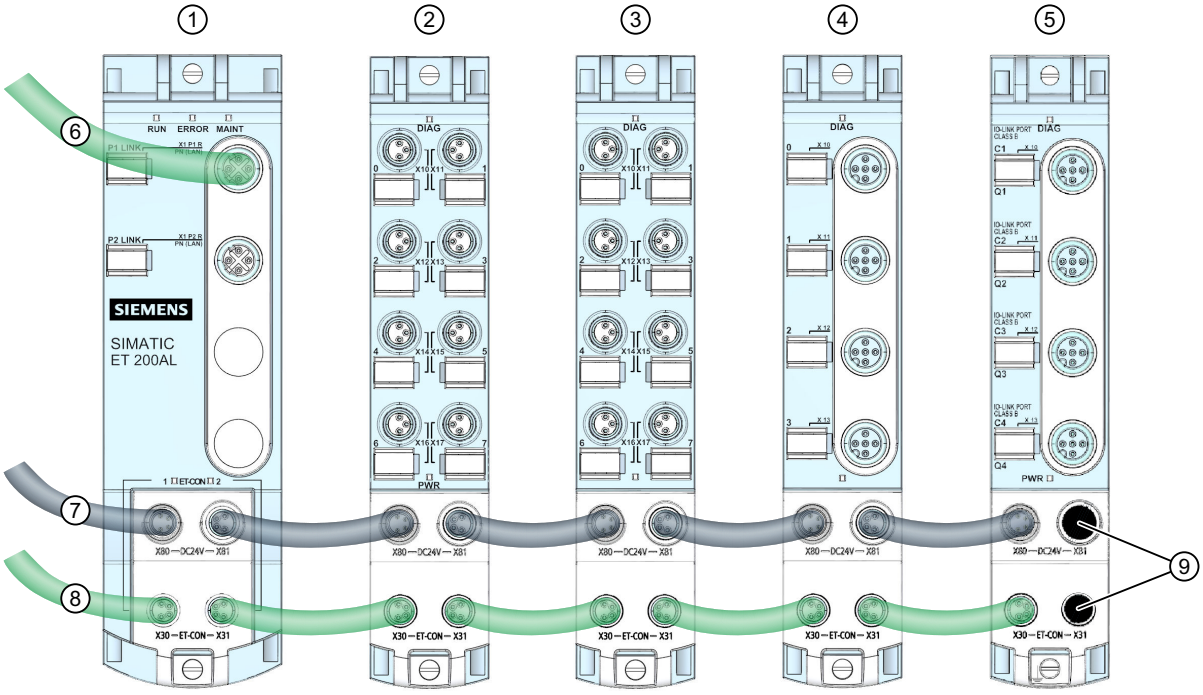
After an interface module you can configure 2 lines (ET-Connection), each with 16 modules.

Alternatively, you can configure a line with 16 I/O modules on the SIMATIC ET 200SP distributed I/O system with BusAdapter BA-Send 1xFC.

The ET-Connection backplane bus is designed as a cable. This allows you to create spatial distances of up to 10 m between the modules.

Configuration example

The figure below shows a configuration example of the SIMATIC ET 200AL distributed I/O system with a PROFINET interface module.



① Interface module (PROFINET)
② Digital input/Digital output module
③ Digital input module
④ Analog input module
⑤ Communications module
⑥ PROFINET cable
⑦ Power supply cable
⑧ ET-Connection line
⑨ Sealing caps

Figure 10-51 Example configuration of the ET 200AL

See also

Expanding ET 200SP with ET 200AL modules (Page 1946)

Configuring ET 200AL

Introduction

ET 200AL is a distributed I/O system with an IP65/67 degree of protection. It is therefore designed for use on site, for example right beside a machine (no control cabinet required).

The system includes interface and communications modules as well as input and output modules.

There are two use cases for the ET 200AL:

1. As I/O device or DP slave: The system interface module is connected to a field bus (PROFINET or PROFIBUS) and connected to the PN or DP interface of a CPU.
2. As an ET 200SP expansion: The ET 200AL modules are connected to the ET 200SP over the "BA Send 1xFC" module ("mixed mode").

The following sections describe how to configure an ET 200AL as I/O device or DP slave (use case 1).

For use case 2, see the link under "See also".

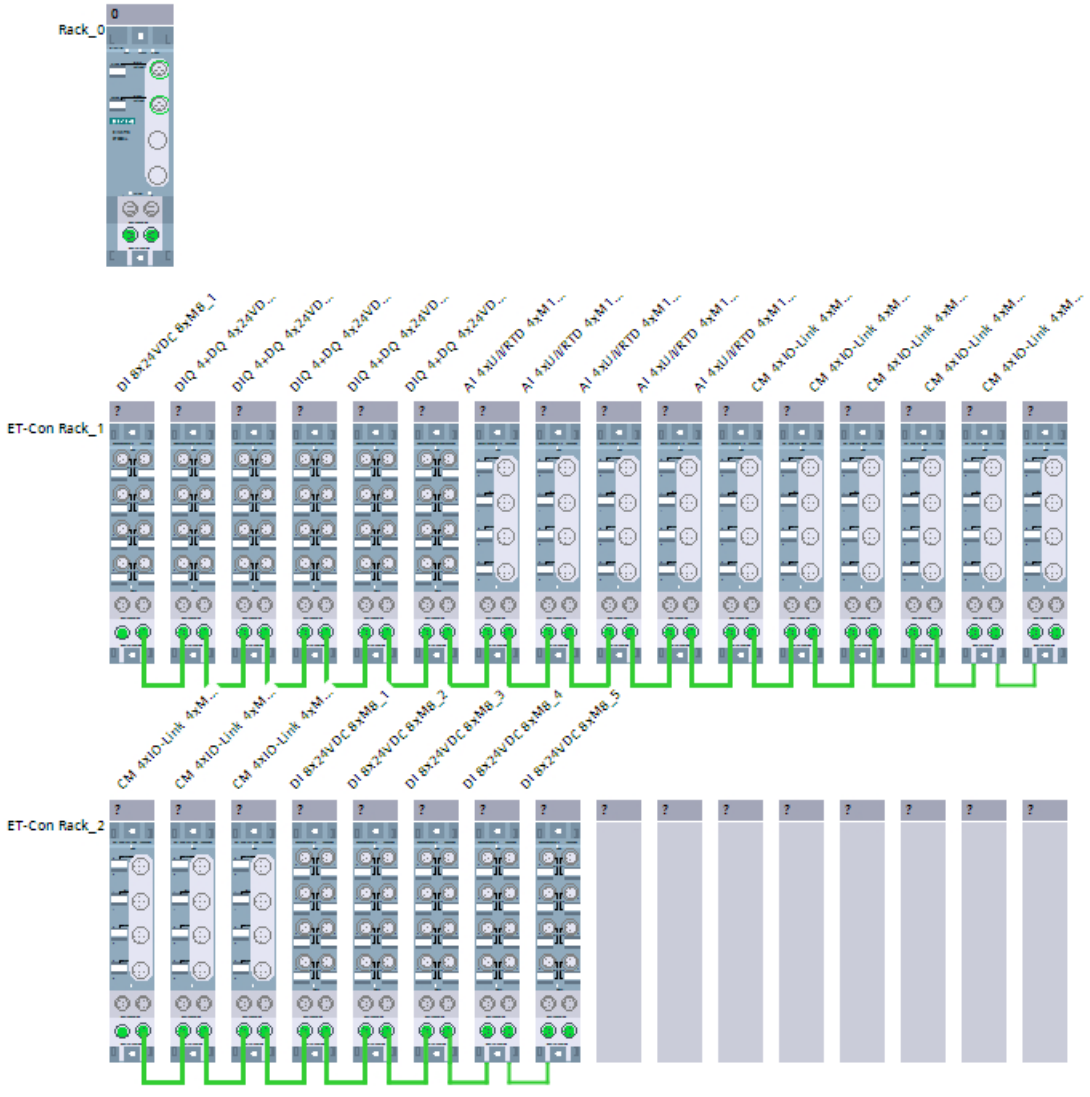
Procedure

Follow the steps below to configure an ET 200AL in STEP 7:

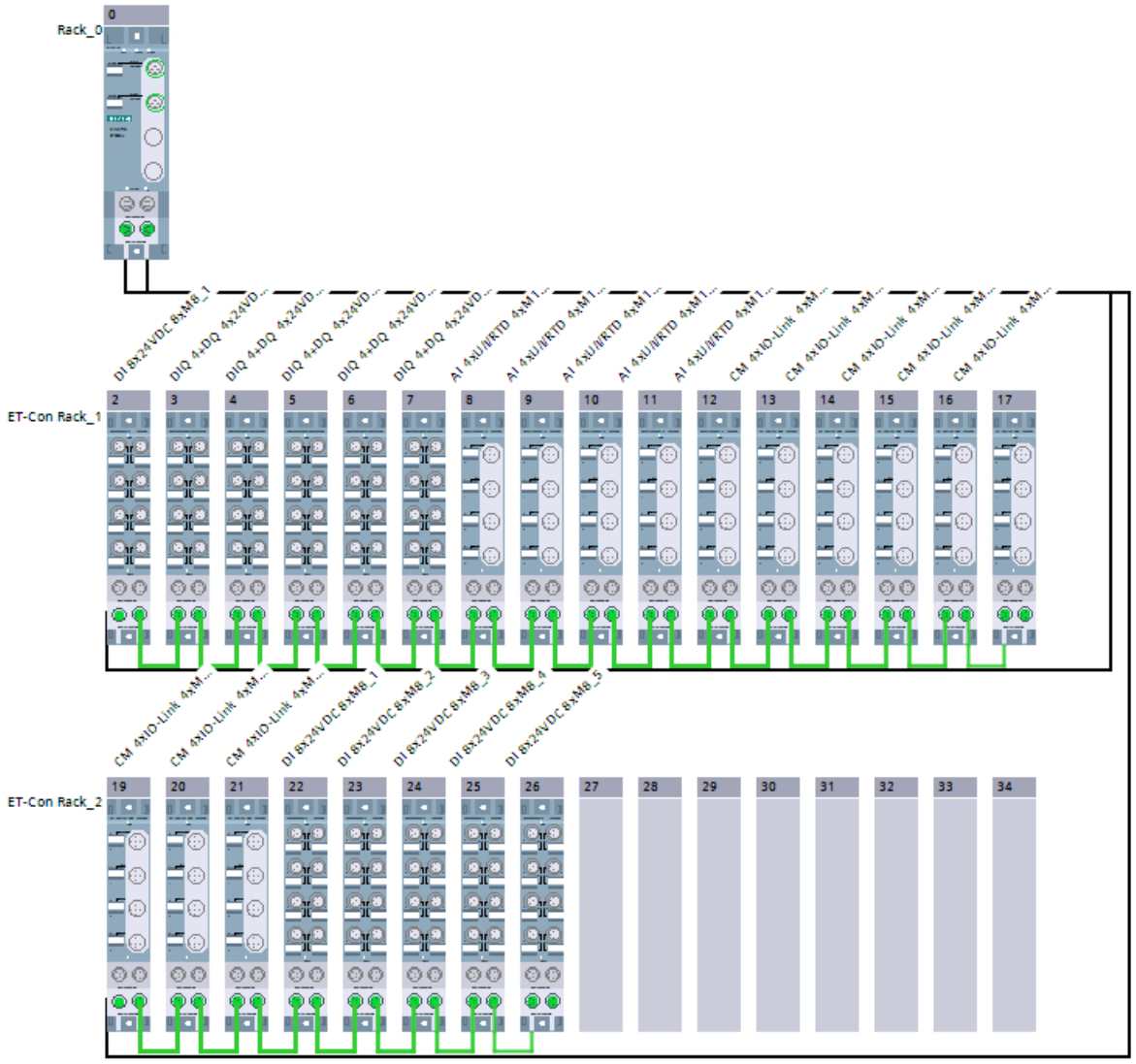
1. Copy an interface module (PROFINET or PROFIBUS) from the ET 200SP series to the network view (drag-and-drop operation from the hardware catalog).
2. Go to the device view. Double-click on the interface module you have just inserted.
   The interface module and two ET-Connection racks are displayed in the device view (figure below).
   There are no slot numbers assigned yet, which is why question marks are displayed above the slots.



3. Now select the modules (input, output and communication modules) from the hardware catalog (ET 200AL folder) and drag them to the free slots (outlined in blue; not shown). You can place up to 16 ET 200AL modules in each ET-Connection rack (figure below). An ET-Connection rack is a virtual rack that sets the order of the connected ET 200AL modules.



4. Now connect the interface module to the two ET-Connection racks.
   First click on an ET-Connection interface of the interface module, then hold and drag a line to the left-hand ET connection of the first module in one of the two ET-Connection racks.
   Repeat this step for the second ET-Connection interface of the interface module and the second ET-Connection rack (if used).



5. Double-click on a module to access the module properties and set the module parameters.

Rules

- The ET 200AL modules must be configured with no gaps.
- The first module in an ET-Connection rack must be connected to the interface module.

See also

ET 200AL distributed I/O system (Page 1974)

Configuration control with ET 200AL

Operating principle

With configuration control, you can change the original configuration of an ET 200AL (created by configuring with STEP 7) with a user program and operate the ET 200AL in this modified configuration. STEP 7 is no longer required for this configuration: You use your user program to signal to the ET 200AL the slot in which a configured module is actually inserted.

In this case, use control data record 196. In this data record, you code which modules are missing or located in different slots in the real configuration compared to the configuration with STEP 7. The configuration control has no effect on the parameter assignment of the modules (for example, the enabling of diagnostic alarms).

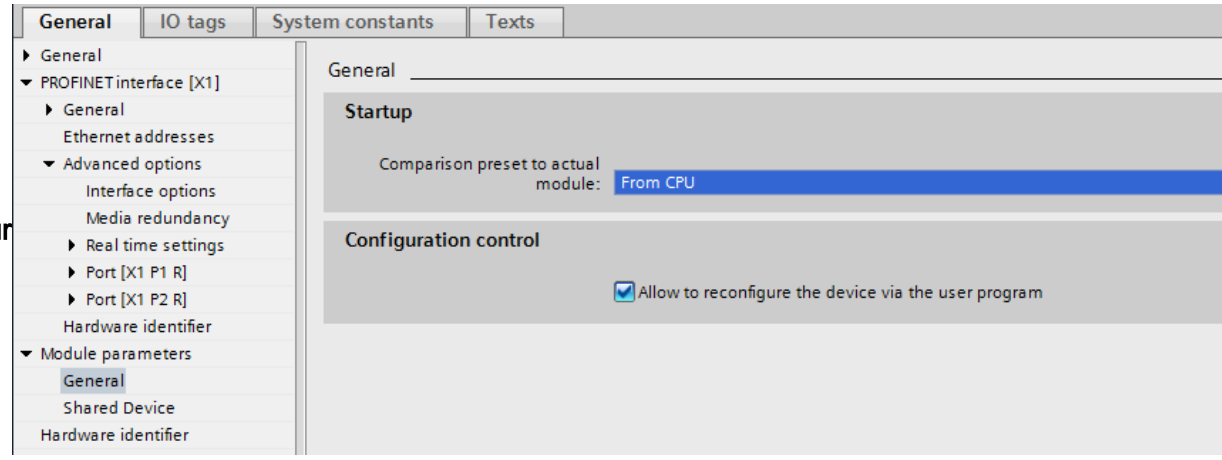
You then call the "WRREC" instruction and use it to write the data record to the interface module of the ET 200AL.

Configuration control gives you the flexibility to vary the configuration of an ET 200AL as long as the real configuration can be derived from a preset maximum configuration (originally created with STEP 7).

The following sections describe how to enable configuration control and how to structure the required data record 196 for the ET 200AL.

Requirement

Enabling configur



Structure of control data record 196

The configuration of the data block corresponds to the original configuration of the ET 200AL with STEP 7.

There are two bytes in the data record for each module. The position of these two bytes in the data record codes a module in the original configuration with STEP 7.

- Bytes 4 and 5 in the data record correspond to the module in slot 1 in the original configuration.
- Bytes 6 and 7 in the data record correspond to the module in slot 2 in the original configuration.
- Bytes 8 and 9 in the data record correspond to the module in slot 3 in the original configuration.
- etc.

The current (real) slot is coded by the number that is assigned to the "Slot_x" byte (by its value): Examples:

- The value "2" in byte 6 means you are assigning the module originally inserted in slot 2 to slot 2 in the current configuration.
- The value "3" in byte 6 means you are assigning the module originally inserted in slot 2 to slot 3 in the current configuration.
- The value "4" in byte 6 means you are assigning the module originally inserted in slot 2 to slot 4 in the current configuration.
- etc.

Creating control data record 196

The figure below shows part of control data record 196 for configuration of an ET 200AL.

This configuration is given as an example:

- ET-Con1 is inserted in "slot_1" (fixed setting). The two ET-Connection submodules "ET-Con1" and "ET-Con2" are submodules of the interface module of the ET 200AL. They are integrated as fixed modules in the IM module. They cannot be inserted individually.
- ET-Con2 is inserted in "slot_18" (fixed setting).
- In this configuration, 16 AL modules are connected to ET-Con1 ("slot_2" to "slot_17" in the following data record). This is the maximum configuration.
- In this configuration, one AL module is connected to ET-Con2 ("slot_19"). However, a total of 16 AL electronic modules could be connected to ET-Con2 (as to ET-Con1).

The ET 200AL originally configured with STEP 7 is now to be reconfigured from the user program.

The new configuration has the following properties:

- ET-Con1 is inserted in "slot_1" (fixed setting).
- Module 2 is also operated in slot 2 in the modified configuration.
- Module 3 is not used.
- Module 4 is now inserted in slot 3.
- Module 5 is now inserted in slot 4.
- No other modules at ET-Con1 are used.

- ET-Con2 is inserted in "slot_18" (fixed setting)
- The module at ET-Con2 is used.

Record_196_ET_200AL

	Name	Data type	Default value
1	block_length	USInt	42
2	block_ID	USInt	196
3	version	USInt	2
4	subversion	USInt	1
5	slot_1_ET_Con1	USInt	1
6	reserve_1_ET_Con1	USInt	0
7	slot_2	USInt	2
8	reserve_2	USInt	0
9	slot_3	USInt	0
10	reserve_3	USInt	0
11	slot_4	USInt	3
12	reserve_4	USInt	0
13	slot_5	USInt	4
14	reserve_5	USInt	0
15	slot_6	USInt	0
16	reserve_6	USInt	0
17	slot_7	USInt	0
18	reserve_7	USInt	0
19	slot_8	USInt	0
20	reserve_8	USInt	0
21	slot_9	USInt	0
22	reserve_9	USInt	0
23	slot_10	USInt	0
24	reserve_10	USInt	0
25	slot_11	USInt	0
26	reserve_11	USInt	0
27	slot_12	USInt	0
28	reserve_12	USInt	0
29	slot_13	USInt	0
30	reserve_13	USInt	0
31	slot_14	USInt	0
32	reserve_14	USInt	0
33	slot_15	USInt	0
34	reserve_15	USInt	0
35	slot_16	USInt	0
36	reserve_16	USInt	0
37	slot_17	USInt	0
38	reserve_17	USInt	0
39	slot_18 ET_Con2	USInt	1
40	reserve_18_ET_Con2	USInt	0
41	slot_19	USInt	1
42	reserve_19	USInt	0

The components of control data record 196 (definition in section "Control data record 196" below):

- block_length: Note the length of the control data record here; in the example: 42 (bytes). The length of the control data block is calculated using the following formula: 2 x "number of modules" + 4.
- block_ID: Enter the figure 196 here. This number identifies the data record as the data record for configuration control.
- version: The ET 200AL uses version 2 of control data record 196.
- subversion: The ET 200AL uses subversion 1 of control data record 196.
- slot_1: The ET-Connection 1 submodule is always inserted in slot 1 of the ET 200AL.
- reserve_1: This byte is not used (value "0").
- slot_2: The configured module 2 is inserted in slot 2 (value "2").
- reserve_2: This byte is not used (value "0").
- slot_3: The configured module 3 is not present in the current configuration (value "0").
- reserve_3: This byte is not used (value "0").
- slot_4: The configured module 4 is inserted in slot 3 in the current configuration (value "3").
- reserve_4: This byte is not used (value "0").
- slot_5: The configured module 5 is inserted in slot 4 in the current configuration (value "4").
- reserve_5: This byte is not used (value "0").
- slot_6: The configured module 6 is not present in the current configuration (value "0").
- reserve_6: This byte is not used (value "0").
- slot_7: The configured module 7 is not present in the current configuration (value "0").
- reserve_7: This byte is not used (value "0").
- etc.
- slot_18: The ET-Connection 2 submodule is always inserted in slot 18 of the ET 200AL (value "18").
- reserve_18: This byte is not used (value "0").
- slot_19: The configured module 19 is inserted in slot 19 in the current configuration (value "19").
- reserve_19: This byte is not used. (value "0")

Control data record 196 for ET 200AL

A control data record 196 containing a slot assignment is defined for configuration control.

Byte	Element	Value	Description	
0	Block length in bytes	e.g. 14 for ET 200AL with 5 modules	The length of the data record is calculated using the formula: 4 + (number of modules x 2) bytes	Header
1	Block ID	196	ID for control data record 196	
2	Version	2	Version 2 of control data record 196	
3	Subversion	1	Subversion 1 of control data record 196	
4	Configured module 1 (ET-Connection 1)	1	ET-Connection 1 is always assigned to slot 1 (fixed). The value "1" must therefore always be entered in byte 4.	Assignment for configured module 1 (ET-Connection 1) to real slot 1
5	Reserve for configured module 1	0	Not used	
6	Configured module 2	Real slot of module 2 Possible values: 2 up to the number of modules (except 18) 0 (if the configured module 2 is not present)	The configured module 2 can be inserted in any slot from 2 to slot 34. Slot 18 is reserved for ET-Con 2. If the configured module is not used, this byte contains the value "0".	Assignment for the configured module 2 to a real slot
7	Reserve for configured module 2	0	Not used	
8	Configured module 3	Real slot of module 3 Possible values: 2 up to the number of modules (except 18) 0 (if the configured module 2 is not present)	The configured module 3 can be inserted in any slot from 2 to slot 34. Slot 18 is reserved for ET-Connection 2. If the configured module is not used, this byte contains the value "0".	Assignment for the configured module 3 to a real slot
9	Reserve for configured module 3	0	Not used	
:	:	:	:	:
39	Configured module 18 (ET-Connection 2)	18	ET-Connection 2 is always assigned to slot 18 if AT modules are connected at this submodule.	Assignment for configured module 18 (ET-Connection 2) to real slot 18
40	Reserve for configured module 18	0	Not used	
:	:	:	:	:

(Bytes 4 to 70, not byte 39)	Configured module x	Real slot of module x Possible values: 2 up to the number of modules (except 18) 0 (if there is no configured module x)	The configured module x can be inserted in any real slot from 2 to slot 34. Slot 18 is reserved for ET-Con 2 (bytes 39 and 40 in the control data record)	Assignment for configured module x to a real slot y
(Bytes 5 to 71, not byte 40)	Reserve for configured module x	0	Not used	

Rules

- The ET-Connection 1 and ET-Connection 2 submodules must be treated like real modules in configuration control. Restriction: ET-Connection 1 is always placed in slot 1 and ET-Connection 2 is always placed in slot 18 (fixed assignment).
- There are no reserve modules for the ET 200AL (unlike for the ET 200S or with BU cover modules for the ET 200SP). For this reason, bit 7 of "slot_x" must not be set (i.e. only the values 0 to 127 may be used).
- The value "0" of "slot_x" indicates that this module is not inserted in the current configuration.
- Gaps must not be left between the AL modules when configuring with STEP 7.
- If no modules are connected to ET-Con2 when STEP 7 is configured, ET-Con2 is not configured: This shortens data record 196.
- If fewer than 16 modules are connected to ET-Con1 during configuration with STEP 7 and there are also modules connected to ET-Con2, control data record 196 must contain all unassigned slots for ET-Con 1. They are assigned zero as the value for the real slot.

Writing a data record

Transfer the control data record to the ET 200AL module.

To do so, call the extended WRREC (Write data record) instruction, and transfer the control data record created.

If you do not transfer a control data record, the interface module uses the original configuration with STEP 7. Here, the following applies: Configured module x is inserted in real slot x.

Addressing the interface module using the HW identifier

To transfer data record 196 with the instruction WRREC, you must enter the HW identifier of the IM submodule with the extension "~Head" as the input parameter for the instruction. The system constant of this HW identifier is, for example, "IO-Device_2~Head". The system constants of a selected device are, for example, displayed in the network view in the "System constants" tab. Use the corresponding value for addressing.

Error messages

The following error messages are returned if an error occurs when writing control data record 196:

Table 10-195 Error messages

Error code	Meaning
16#80A2	DP protocol error on layer 2 Indicates that a data record has not been acknowledged due to the system.
16#80B1	Invalid length; the length information in data record 196 is not correct.
16#80B5	Configuration control parameters not assigned.
16#80B2	Invalid slot: The configured slot is not assigned.
16#80B8	Parameter error; module signals invalid parameters.
16#80C5	DP slave or module not available. Indicates that a data record has not been acknowledged due to the system.

Feedback data record 197 for ET 200AL

The feedback data record 197 is used to read the actual configuration of a station (in this case of an ET 200AL).

This data record allows you to check the real configuration of the ET 200AL (actual configuration). The feedback data record for each configured module specifies whether or not it is actually present.

- The value "1" means that the correct module is inserted in the correct slot.
- The value "0" codes all other possibilities (wrong module, empty slot, BU cover).

Example:

A module has been configured with STEP 7 for slot 4.

This module has then been moved to slot 3 in the current configuration using data record 196.

If this module is also really in slot 3, this is coded by the value "1" (status_slot_4 = 1).

Configuration details:

The configuration of the data block corresponds to the original configuration of the ET 200AL with STEP 7.

There are two bytes in the data record for each module. The position of these two bytes in the data record corresponds to the position of a module in the original configuration with STEP 7.

Sequence of bytes:

- "status_slot_1_ET_Con1" and "reserve_slot_1_ET-Con1" (bytes 4 and 5 in the data record) correspond to the module in slot 1 in the configuration,
- "status_slot_2" and "reserve_slot_2" (bytes 6 and 7) correspond to the module in slot 2 in the configuration
- "status_slot_3" and "reserve_slot_3" (bytes 8 and 9) correspond to the module in slot 3 in the configuration,
- etc.

Example

The following feedback data record 197 is returned by the ET 200AL that was reconfigured with control data record 196 in the example above (section "Creating control data record 196").

Record_197_ET_200AL

	Name	Data type	Default value
1	block_length	USInt	42
2	block_ID	USInt	197
3	version	USInt	2
4	subversion	USInt	1
5	status_slot_1_ET_Con1	USInt	1
6	reserve_1_ET_Con1	USInt	0
7	status_slot_2	USInt	1
8	reserve_2	USInt	0
9	status_slot_3	USInt	0
10	reserve_3	USInt	0
11	status_slot_4	USInt	1
12	reserve_4	USInt	0
13	status_slot_5	USInt	1
14	reserve_5	USInt	0
15	status_slot_6	USInt	0
16	reserve_6	USInt	0
17	status_slot_7	USInt	0
18	reserve_7	USInt	0
19	status_slot_8	USInt	0
20	reserve_8	USInt	0
21	status_slot_9	USInt	0
22	reserve_9	USInt	0
23	status_slot_10	USInt	0
24	reserve_10	USInt	0
25	status_slot_11	USInt	0
26	reserve_11	USInt	0
27	status_slot_12	USInt	0
28	reserve_12	USInt	0
29	status_slot_13	USInt	0
30	reserve_13	USInt	0
31	status_slot_14	USInt	0
32	reserve_14	USInt	0
33	status_slot_15	USInt	0
34	reserve_15	USInt	0
35	status_slot_16	USInt	0
36	reserve_16	USInt	0
37	status_slot_17	USInt	0
38	reserve_17	USInt	0
39	status_slot_18 ET_Con2	USInt	1
40	reserve_18_ET_Con2	USInt	0
41	status_slot_19	USInt	1
42	reserve_19	USInt	0

Modules 2, 4 and 5 are actually connected to ET-Con1.

None of the other modules that were connected to ET-Con1 in the configuration with STEP 7 are present in the current configuration (in line with the control data record 196 settings from the example above).

A module is really connected to ET-Con2 as in the original configuration with STEP 7.

Reading feedback data record 197

You can read the feedback data record 197 from the ET 200AL with the RDREC instruction. RDREC operates asynchronously. If you call RDREC in the startup OB, you must call the instruction multiple times using a loop until the "BUSY" or "DONE" output parameter indicates that the data record has been read.

To read data record 197 with the instruction RDREC, you must enter the HW identifier of the IM submodule with the extension "~Head" as the input parameter for the instruction. The system constant of this HW identifier is, for example, "IO-Device_2~Head". The system constants of a selected device are, for example, displayed in the network view in the "System constants" tab. Use the corresponding value for addressing.

Additional information and examples

Further information on ET 200AL is available here (http://support.automation.siemens.com/WW/view/en/89254863).

Specific examples of configuration control can be found here in this application description (http://support.automation.siemens.com/WW/view/en/29430270).

You will find a library of the data records and other application examples here: Templates for the data records (http://support.automation.siemens.com/WW/view/en/29430270)

See also

ET 200AL distributed I/O system (Page 1974)

Expanding ET 200SP with ET 200AL modules (Page 1946)

Configuration control with ET 200SP (Page 1934)

ET 200MP

ET 200MP distributed I/O system

Definition

The ET 200MP distributed I/O system is a scalable and flexible distributed I/O system for connection of process signals to a central controller via a field bus.

Application area

The ET 200MP is a multi-functional distributed I/O system for various fields of application. The scalable design allows you to configure the system exactly to the specific requirements on location.

The ET 200MP complies with IP 20 degree of protection and is intended for installation in a control cabinet.

Structure

The ET 200MP is installed on a mounting rail and comprises:

- An interface module that communicates with all IO controllers conforming to the PROFINET standard IEC 61158
- Up to 30 modules (power supply modules and I/O modules from the S7-1500 I/O range) can be inserted to the right of the interface module.
- If you insert a power supply module to the left of the interface module, this yields a possible maximum configuration of 32 modules in total.
- The number of insertable I/O modules is limited by their power requirements.

Slot rules

- Slot 0: Power supply module (optional)
- Slot 1: Interface module
- Slot 2 to 31: I/O modules or power supply modules

Interface module parameters

Supply voltage L+ connected

Parameter "Supply voltage L+ connected"

This parameter influences the diagnostics and the checking of the power budget.

- Diagnostics of the ET 200MP:
  If the actual configuration does not match the preset configuration with regard to the supply voltage of the interface module, the interface module generates a diagnostic alarm.
  Example: You have deactivated the "Supply voltage L+ connected" option, but you have connected the supply voltage in the actual configuration.
- Power budget check during configuration:
  The power budget changes in accordance with the parameter setting: Either the interface module feeds power into the backplane bus or it draws power from the backplane bus.

The default ("Supply voltage L+ connected" option is **activated**) means that the front of the interface module is supplied with 24 V DC and the power is stored in the backplane bus.

If the "Supply voltage L+ connected" option is **deactivated**, the interface module may not be supplied with 24 V DC on the front.

In this case, insert a power supply unit (PS) on the left next to the interface module that supplies the interface module and the modules to the right of the interface module.

Note

We recommend that you always supply the interface module on the front side with 24 V DC. If a system power supply unit (PS) is inserted and connected additionally **before** or on the left next to the interface module, both the power from the system power supply unit (PS) as well as the power from the integrated power supply of the interface module are then available to the configuration.

In this case, you do not have to change the default setting of the parameter.

Configuration control (handling options) with ET 200MP

Operating principle

Configuration control allows you to operate various real configurations (options) with a single configuration of the distributed I/O device ET 200MP.

Configuration control provides you with the option of configuring the ET 200MP distributed I/O device with its maximum configuration and still operating it with modules missing. If missing modules are retrofitted later, no new configuration is required and the hardware configuration does not have to be reloaded either.

Using control data record 196, which is transferred to the interface module in the user program, you define a current configuration. You transfer the control data record with the instruction WRREC.

Readback data record 197 is used to read the actual configuration of an ET 200MP.

Requirements

The CPU startup parameter "Compare preset to actual configuration" is set to Startup even if mismatch (default setting). This setting is also selected for the startup parameters of the individual modules of the ET 200MP.

Enabling configuration control

In the properties of the interface module under Module parameters > General > Configuration control, select "Enable reconfiguration of device via user program". This activates configuration control.

Control data record 196 for ET 200MP

The figure below shows the start of control data record 196 for the configuration control of an ET 200MP.

The data block is 36 bytes long (maximum configuration with 32 modules). The value "36" therefore appears in the "block_length" element of the data record.

If you configure an ET 200MP in STEP 7 with fewer modules, the data block will be shorter: If there are only five modules, for example, the data record is reduced to 9 bytes (4 bytes for the header plus one byte for each module).

There is one byte in the data record for each module. The position of this byte in the data record codes a module in the original configuration with STEP 7:

- "slot_0 power supply" (byte 4 in the data record below) corresponds to the power supply module in slot 0 in the configuration with STEP 7.
- "slot_2" (byte 5 in the data record) corresponds to the module in slot 2 in the configuration.
- "slot_3" (byte 6 in the data record) corresponds to the module in slot 3 in the configuration.
- "slot_4" (byte 7) corresponds to the module in slot 4 in the configuration.
- etc.

Note

Interface module in slot 1

Configuration control is controlled by the interface module (slot 1/submodule 1). The interface module in slot 1 is therefore not an element of the configuration control and is not included in the data record.

Value in slot_x

The current slot is coded by the figure that is assigned to "slot_x" (by its value). Examples:

- The value "2" in slot_2 means you are assigning the module originally inserted in slot 2 to slot 2 in the current configuration (slot_2 = 2).
- The value "3" in slot_2 means you are assigning the module originally inserted in slot 2 to slot 3 in the current configuration (slot_2 = 3).
- The value "4" in slot_2 means you are assigning the module originally inserted in slot 2 to slot 4 in the current configuration (slot_2 = 4).
- etc.

Example for data record 196

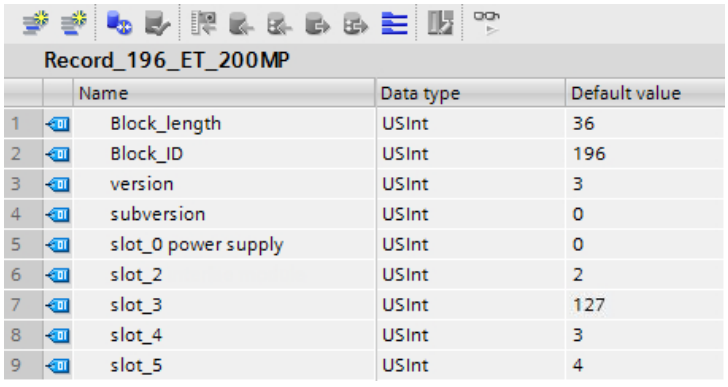
The following data record was created for a configuration that changes the original configuration with STEP 7.

The modified configuration has the following properties:

- The module inserted in slot 0 in the configuration (power supply module) is also inserted in slot 0 in the current configuration (specification).
- The module inserted in slot 2 in the configuration (module 2) is also inserted in slot 2 in the current configuration.
- The module inserted in slot 3 in the configuration (module 3) does not exist in the current configuration.
- The module inserted in slot 4 in the configuration (module 4) is inserted in slot 3 in the current configuration.

- The module inserted in slot 5 in the configuration (module 5) is inserted in slot 4 in the current configuration.
- etc.

The bytes "slot_6" to "slot_31" are not shown in the figure below.



Record_196_ET_200MP

	Name	Data type	Default value
1	Block_length	USInt	36
2	Block_ID	USInt	196
3	version	USInt	3
4	subversion	USInt	0
5	slot_0 power supply	USInt	0
6	slot_2	USInt	2
7	slot_3	USInt	127
8	slot_4	USInt	3
9	slot_5	USInt	4

Rules

- If a module dos not exist in the current configuration, this is indicated by the value 127: "slot_x" = 127.
- The power supply module is always in slot 0 ("slot_0 power supply" = 0).
- The interface module in slot 1 is not included in the control data record.

Addressing the interface module using the HW identifier

To transfer data record 196 with the instruction WRREC, you must enter the HW identifier of the IM submodule with the extension "~Head" as the input parameter for the instruction. The system constant of this HW identifier is, for example, "IO-Device_2~Head". The system constants of a selected device are, for example, displayed in the network view in the "System constants" tab. Use the corresponding value for addressing.

Readback data record 197 for ET 200MP

Readback data record 197 is used to read the actual configuration of a station (in this case, of an ET 200MP).

This data record allows you to check the real configuration of the ET 200MP (actual configuration). The readback data record for each configured module specifies whether or not it is actually available.

- The value "1" means that the correct module is inserted in the correct slot.
- The value "0" codes all other options (wrong module, empty slot, reserve module).

Example:

A module has been configured with STEP 7 for slot 4.

This module has then been moved to slot 3 in the current configuration using data record 196.

If this module is also really in slot 3, this is coded by the value "1" (status_slot_4 = 1).

Configuration details:

The configuration of the data block corresponds to the original configuration of the ET 200MP with STEP 7.

There is a byte in the data record for each module. The position of this bytes in the data record corresponds to the position of a module in the original configuration with STEP 7.

Sequence of bytes:

- "status_slot_0 power supply" (byte 4 in the data record below) corresponds to the power supply module in slot 0 in the configuration with STEP 7.
- "status_slot_2" (byte 5) corresponds to the module in slot 2 in the configuration.
- "status_slot_3" (byte 6) corresponds to the module in slot 3 in the configuration,
- etc.

You can choose any name for the components (for example "status_slot_2").

Meaning of "status_slot_x":

- The value "1" in status_slot_x means that module x is in the correct slot.
- The value "0" in status_slot_x codes all other possibilities (wrong module, module does not exist).

Example:

The figure below shows readback data record 197 for the configuration of an ET 200MP in which there is no module 3 (the module in slot 3 in the configuration).

All other modules are available and correctly plugged.

The bytes "status_slot_6" to "status_slot_31" are not shown in the figure below.

Record_197_ET_200MP

	Name	Data type	Default value
1	Block_length	USInt	36
2	Block_ID	USInt	197
3	version	USInt	3
4	subversion	USInt	0
5	status_slot_0 power supply	USInt	1
6	status_slot_2	USInt	1
7	status_slot_3	USInt	0
8	status_slot_4	USInt	1
9	status_slot_5	USInt	1

Reading readback data record 197

You can read readback data record 197 from the ET 200MP with the instruction RDREC. RDREC operates asynchronously. If you call RDREC in the startup OB, you must call the instruction multiple times using a loop until the "BUSY" or "DONE" output parameter indicates that the data record has been read.

To read data record 197 with the instruction RDREC, you must enter the HW identifier of the IM submodule with the extension "~Head" as the input parameter for the instruction. The system constant of this HW identifier is, for example, "IO-Device_2~Head". The system constants of a selected device are, for example, displayed in the network view in the "System constants" tab. Use the corresponding value for addressing.

Additional information and examples

Further information on ET 200MP can be found in the manual for IM 155-5 PN (http://support.automation.siemens.com/WW/view/en/89261636).

Specific examples of configuration control can be found in this application description (http://support.automation.siemens.com/WW/view/en/29430270).

You will find a library of the data records and other application examples here: Templates for the data records (http://support.automation.siemens.com/WW/view/en/29430270)

See also

Documentation on configuration control (http://support.automation.siemens.com/WW/view/en/67295970)

Input module parameters

Parameters of the analog input modules

Missing supply voltage L+

Enabling of the diagnostics for missing or insufficient supply voltage L+.

Wire break

Enabling of the diagnostics if the module has no current flow or the current is too weak for the measurement at the corresponding configured input or the applied voltage is too low.

Current limit for wire break diagnostics

Threshold at which a wire break is reported. The value can be set to 1.185 mA or 3.6 mA, depending on the sensor used.

Overflow

Enabling of the diagnostics if the measured value exceeds the overrange.

Underflow

Enabling of the diagnostics if the measured value undershoots the underrange.

Common mode error

Enable diagnostics if the valid common mode voltage is exceeded.

Reference channel error (only for AI 8xU/I/RTD/TC ST)

- Enable diagnostics on error at the temperature compensation channel, e.g. wire break.
- Dynamic reference temperature compensation type is configured and no reference temperature has been transferred to the module yet.

Temperature coefficient

The temperature coefficient depends on the chemical composition of the material. In Europe, only one value is used per sensor type (default value).

The correction factor for the temperature coefficient (α value) specifies how much the resistance of a certain material changes when the temperature is raised by 1 °C.

The further values facilitate a sensor-specific setting of the temperature coefficient and enhance accuracy.

Interference frequency suppression

At analog input modules, this suppresses interference caused by the frequency of AC mains.

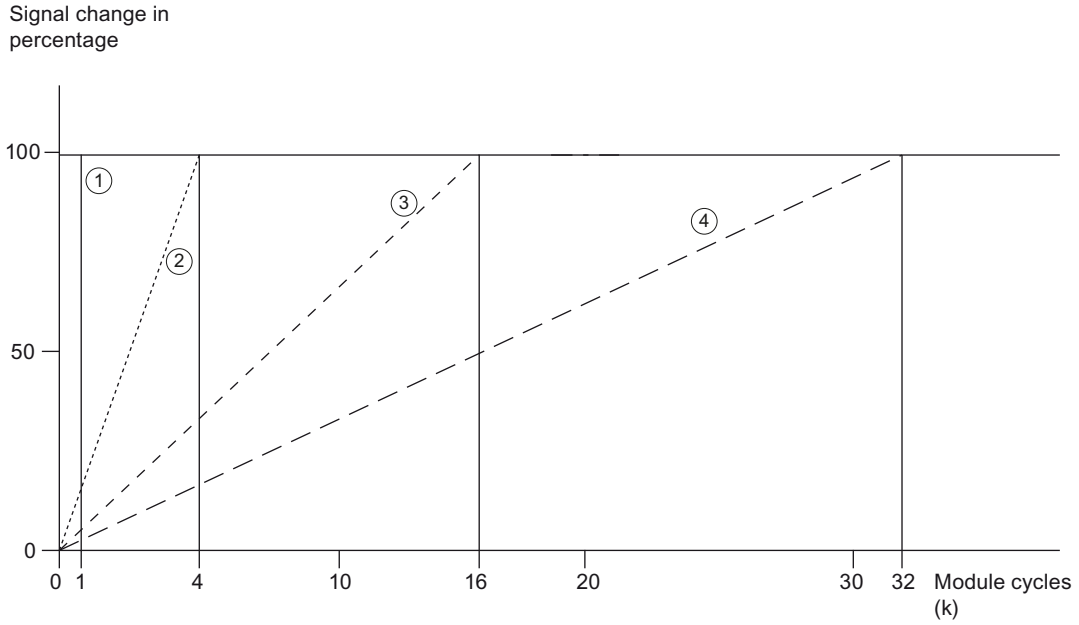
The frequency of the AC network may interfere with measured values, particularly for measurements within low voltage ranges and when thermocouples are being used. With this parameter, you define the mains frequency in your system.

Smoothing

The individual measured values are smoothed using filtering. Smoothing can be set in 4 stages for the analog input modules AI 8xU/I/RTD/TC ST and AI 8xU/I HS.

Smoothing time = number of module cycles (k) x cycle time of the module.

The figure below shows the number of module cycles after which the smoothed analog value is almost 100%, depending on the set smoothing. Is valid for each signal change at the analog input.



① None (k = 1)
② Weak (k = 4)
③ Medium (k = 16)
④ Strong (k = 32)

Reference junction (only for AI 8xU/I/RTD/TC ST)

The following settings can be configured for the reference junction parameter:

Table 10-196 Possible parameter assignments for the reference junction parameter

Setting	Description
Fixed reference temperature	The reference junction temperature is configured and stored in the module as a fixed value.
Dynamic reference temperature	The reference junction temperature is transferred in the user program from the CPU to the module by data records 192 to 199 using the WRREC (SFB 53) instruction.
Internal reference junction	The reference junction temperature is determined using an integrated sensor of the module.
Reference channel of the module	The reference junction temperature is determined using an external resistance thermometer (RTD) at the reference channel (COMP) of the module.

Note

Fixed reference temperature

During parameter assignment of a thermocouple Type B, only the setting "Fixed reference temperature" with a temperature of 0 °C is possible.

Enable hardware interrupt 1 or 2

Enable a hardware interrupt if high limit 1 or 2 is exceeded, or low limit 1 or 2 is violated.

Low limit 1 or 2

Specifies the low limit threshold that triggers hardware interrupt 1 or 2.

High limit 1 or 2

Specifies the high limit threshold that triggers hardware interrupt 1 or 2.

Temperature compensation for thermocouples

Introduction

You have several options of measuring the reference junction temperature in order to obtain an absolute temperature value as a function of the temperature difference between the reference junction and the measuring point.

You can use various compensation options depending on the required location of the reference junction.

Note

During parameter assignment of a thermocouple Type B, only the setting "Fixed reference temperature" with a temperature of 0 °C is possible.

Options of compensating for the reference junction temperature

Compensation options	Explanation	Application case
Internal reference junction	With this compensation, the reference junction temperature is determined using an integrated sensor of the module. **Procedure** Connect the thermocouple to the I/O module directly or with compensating lines.	• For the connection, you use compensating lines matching the thermocouple material. • If the reference junction temperature and the module temperature are identical in your system, you may also use lines made from a different material.
Reference channel of the module	The reference junction temperature is determined using an external resistance thermometer (RTD). **Procedure** Connect the thermocouple to the supply lines at the reference junction, either directly or with compensating lines. You connect the supply lines to the appropriate terminals of the module. Connect the resistance thermometer (RTD) to the reference channel of the module. The resistance thermometer (RTD) must be placed in the area of the reference junction.	• You want to measure the temperature directly at the reference junction. • The measured temperatures of all channels that you have configured for this compensation type is corrected automatically by the temperature value of the reference junction. • You can use inexpensive lines, e.g., copper lines, from the reference junction to the module.
Dynamic reference temperature	The temperature of the reference junction is determined via a module. This temperature value is transferred to other modules via a data record in the user program. **Procedure** Connect the resistance thermometer (RTD) for the reference junction to any channel. The reference junction temperature is communicated from the CPU to the module by data records 192 to 199 using the WRREC instruction.	• You use multiple modules at the reference junction and can therefore compensate all channels using a common temperature value. • You require only one resistance thermometer (RTD) to acquire the temperature value. • You can use inexpensive lines, e.g., copper lines, from the reference junction to the module.
Fixed reference temperature	The reference junction temperature is stored in the module as a fixed value. **Procedure** Connect the thermocouple to the supply lines at the reference junction, either directly or with compensating lines. You connect the supply lines to the appropriate terminals of the module. When configuring the module, specify a fixed temperature value for the reference junction (e.g. 20 °C).	• You keep the reference junction temperature constant and know the temperature value. • You can use inexpensive lines, e.g., copper lines, from the reference junction to the module.

Output module parameters

Parameters of the analog output modules

Missing supply voltage L+

Enabling of the diagnostics, with missing or too little supply voltage L+.

Short-circuit to ground

Enabling of the diagnostics if a short-circuit of the actuator supply to ground occurs.

Wire break

Enabling diagnostics if the line to the encoder is interrupted.

Overflow

Enabling of the diagnostics if the measured value exceeds the overflow range.

Underflow

Enabling of the diagnostics if the measured value falls below the underflow range.

Reaction to CPU STOP

Determines the reaction of the output to the CPU going into STOP state.

Substitute value

The substitute values are values that the outputs (the output) issue in the event of a CPU STOP.

Pulse-width modulation (PWM)

The channels of some modules support the pulse width modulation (PWM) function. You can use the pulse width modulation function to easily generate periodic pulses with a constant rated voltage and a variable pulse duration for the above-mentioned channels.

Configuration

You configure the pulse width modulation with the following parameters:

- Enable PWM function on module
- Set PWM operating mode for the corresponding channel
- Set period

How it works

In the pulse width modulation mode, the correspondingly configured outputs provide a pulse width modulated output signal.

Pulse width modulation is characterized by its time period (frequency) and its duty factor. The duty factor describes the relation between pulse duration and time period.

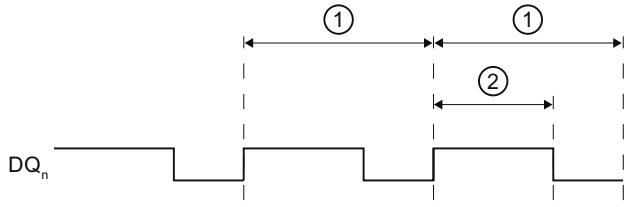
The pulse duration is derived from the time period and the duty factor: Pulse duration = duty factor x time period.

Example for duty factor of 50% and time period of 10 ms:

Pulse duration 0.5 x 10 ms = 5 ms

You define the duty factor of the channels in the user program using the output value (0 ... 1000) in the process image output.

The output signal is a square wave signal (pulse sequence of on and off pulses).



① Time period T (2 to 100 ms); Frequency of the pulse width modulation: f = 1/T (10 to 500 Hz)

② Pulse duration (duty factor x time period)

Figure 10-52 How pulse width modulation works

Minimum pulse duration

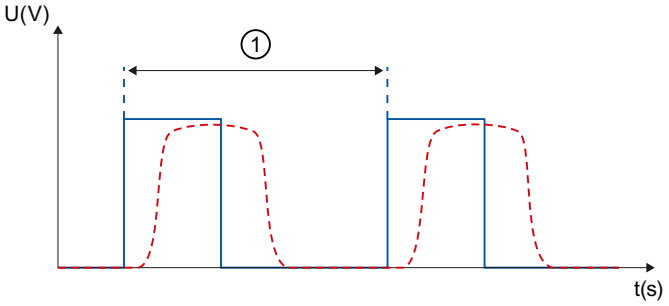
The minimum pulse duration is 300 µs due to the hardware. The duty factor can be adjusted from 0.0 to 100.0%. The time period can be adjusted from 2 to 100 ms.

Example: If you configure a time period of 2 ms and set a duty factor of 0.1% for the output, this would result in a pulse duration of 200 µs. In fact, the output works with a minimum pulse duration of 300 µs.

Pulse waveform

The pulse duration of the actual signal profile is slightly longer than the specified, ideal pulse duration.

The figure below shows the reaction of the output to control by PWM. The blue line shows the specified, ideal signal profile (square wave signal), with which the output is controlled. The red dashed line show the actual signal profile on the output graph, caused by the externally connected load.



① Time period
② Pulse duration (duty factor x time period)

Figure 10-53 Pulse waveform at output terminal

ET 200M

Configuring an ET 200M

Introduction

For the ET 200M series, you can find a wide range of modules in the hardware catalog under "Distributed I/O".

Configuration and parameter assignment

Information on configuration and parameter assignment can be found in the following sections.

ET 200M configuration

Definition

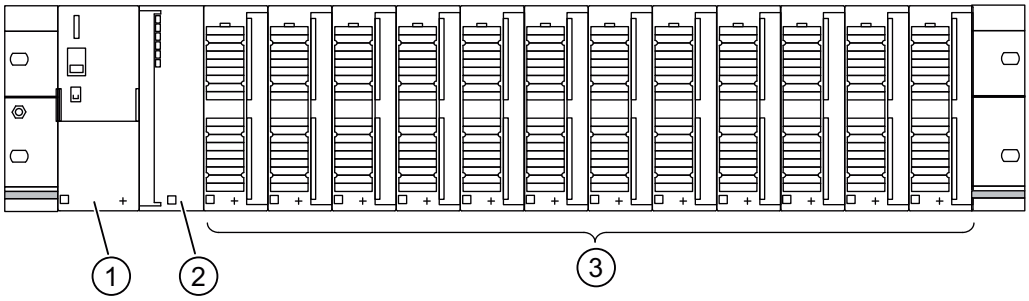
The distributed IO device, ET 200M, is a modular DP slave with an IP 20 degree of protection.

The ET 200M has the configuration technology of the S7-300 automation system and consists of an IM 153-x and I/O modules of the S7-300.

ET 200M supports communication with:

- all DP masters compliant with IEC 61784-1:2002 Ed1 CP 3/1
- all IO controllers compliant with IEC 61158

Configuration of the ET 200M (example)



① Power supply module PS 307
② Interface module IM 153-x
③ up to 12 I/O modules (SM / FM / CP)

Configuration of the 'Module replacement during operation' function

Introduction

The ET 200M supports the "Replace modules during operation" function and the associated pull/plug interrupt.

The "Replace modules during operation" function makes it possible for you to pull modules from or plug modules into the ET 200M rack during operation.

Requirement

You have configured an interface module that supports replacing modules during operation. (as of IM 153-1, article no. 153-1AA02-0XB0).

In addition, the configured CPU must also support the function, e.g. for PROFIBUS an S7-400 with DP interface.

You must use the active backplane bus (bus rail with slots) for the hardware configuration. The conventional profile rail with bus connectors between the modules does not support this function.

Configuring

If the configuration requirements have been met, the "Replace modules during operation" parameter is available for selection in the inspector window's "Module parameters" area. Below this parameter, a table for the configured modules is displayed, which shows the required active bus modules for the hardware configuration.

For a PROFIBUS configuration, the "Startup if preset configuration does not match actual configuration" option is displayed. This option is automatically enabled if "Replace modules during operation" is enabled.

Signal modules for process automation

Fundamentals

Introduction

Signal modules for the process automation are S7-300 models, such as SM 321; DI 16xNAMUR or SM 322; DO 16x24VDC/0.5A.

They are being operated in a DP slave (IM 153-2).

Unlike standard modules, they offer the following additional technical functions, such as pulse extension and chatter monitoring.

See also

Changeover contact (Page 2004)

Technological parameters (Page 2005)

Changeover contact

"Changeover contact" sensor type

If the digital inputs of a channel group are configured as "changeover contacts", the module runs diagnostics for the changeover contact sensor type for this channel group.

Changeover contact

A changeover contact is an auxiliary switch with only one moving switch element with one close setting each for closed and open switching device.

Remember the following rule:

- Always connect a normally open contact to the "even" channel
- Always connect a normally closed contact to the "odd" channel.

The tolerated switchover time between the two channels is fixed at 300 ms.

If the result of the check is negative, then

- the module identifies the value status of the normally open channel as "invalid"
- the module generates a diagnostic entry for the normally open channel
- triggers a diagnostic interrupt (if diagnostic interrupts have been enabled)

The digital input signal and the value status are updated only for the normally open channel. For the normally closed channel, the digital input signal is set permanently to "zero" and the value status to "invalid" since this channel is used only to check the sensor.

Diagnostics depends on the "Selection" parameter (of the sensor). You should also note the special features of diagnostics with the changeover contact sensor type in the "Signal Modules for Process Automation" manual.

See also

Documentation on modules for process automation (http://support.automation.siemens.com/WW/view/de/7215812/0/en)

Technological parameters

Pulse extension and flutter monitoring

Pulse extension is a function for changing a digital input signal. A pulse at a digital input is extended to at least the length set in the parameters. If the input pulse is already longer than the specified length, it is not changed.

If you want the pulse to be extended, click in the box to select the time. If you do not want the pulses to be extended, select the "---" entry.

Flutter monitoring is a process control function for digital input signals. It detects and reports signal changes that are unexpected in process control, for example when an input signal fluctuates too often between "0" and "1".

Flutter monitoring is possible only when group diagnostics has also been enabled for this input.

Monitoring window and number of signal changes

Flutter monitoring works with aid of the two parameters Monitoring window and Number of signal changes.

The first time the signal changes, the time set as the monitoring window is started. If the signal changes more often during this time than allowed by the number of signal changes parameter, this is signaled as a flutter error. If no flutter error is detected during the monitoring window time, the monitoring window can be restarted at the next signal change.

Note

If you set pulse extension for an input channel, this also affects the flutter monitoring enabled for this channel. The "extended pulse" signal is the input signal for the flutter monitoring. You should therefore make sure that the values set for pulse extension and flutter monitoring are compatible with each other.

See also

Documentation on modules for process automation (http://support.automation.siemens.com/WW/view/de/7215812/0/en)

IQ Sense module

Properties of 8 IQ-SENSE

Properties

The 8 IQ-SENSE module has the following properties:

- Connection of sensors with IQ-SENSE®, photoelectric proximity switches: for example, reflex sensors, diffuse sensors, and laser sensors.
- It can be used centrally in an S7-300 or distributed in an ET 200M.
- You can connect up to 8 sensors to every module. Each sensor requires a two-wire cable.
- Function reserve that can be assigned parameters.
- Time functions, switching hysteresis, synchronous mode that can be assigned parameters
- Sensitivity and distance values can be specified (*IntelliTeach* using the "IQ-SENSE Opto" FB)
- Teach-in
- Sensors can be removed and inserted during operation (automatic reassignment of parameters)

Anti-interference group

Only for optical IQ Sense devices (IQ profile ID 1).

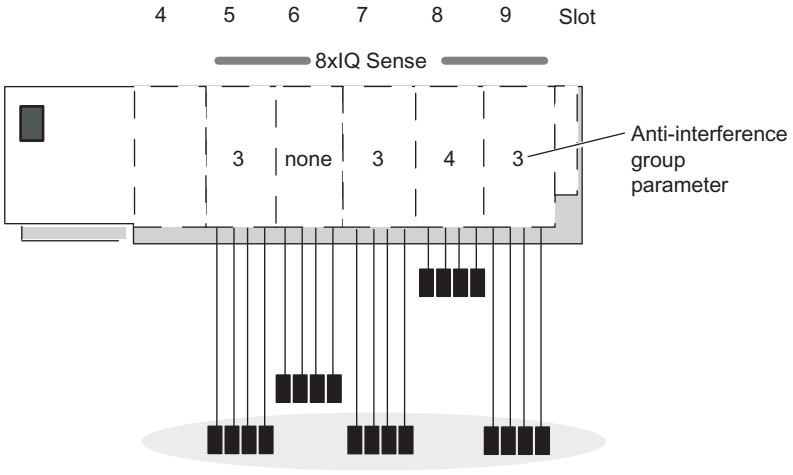
For IQ Sense devices with IQ profile ID 128 (ultrasound), see "Multiplex/synchronous mode" under the channel-specific parameters.

Prevention of interference (e.g., scattered light) by assigning an anti-interference group. This means:

- Anti-interference group: None (= default)
  Optical sensors on one or more modules can mutually influence each other when unfavorably arranged.
- Anti-interference group: 3 or 4
  Optical sensors on the same module with anti-interference group 3 or 4 cannot mutually influence each other. Similarly, optical sensors on different modules with anti-interference group 3 or 4 cannot mutually influence each other. You need not maintain minimum clearance between the IQ Sense devices and can, for example, align two retroreflective sensors on a single reflector.

Operating principle

The diagram below explains the functioning of the anti-interference group parameter:



Mutual interference is only possible between the optical sensors of the modules in slot 5, 6, 7 and 9 because they are in the same anti-interference group 3 or "None" is set.

Note

Sensors in the same anti-interference group must be installed to maintain the minimum clearance (see sensor package insert) and to prevent mutual interference.

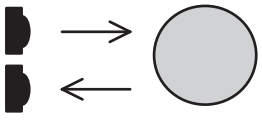
Encoder type

This parameter is used to set the sensor type per channel:

- Reflex sensor or
- Diffuse sensor or
- Disabled

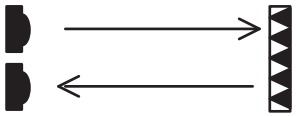
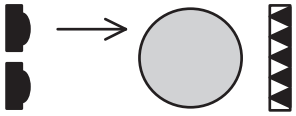
Diffuse sensor

Table 10-197 Diffuse sensor

Diffuse sensor	Object	
Transmitter Receiver		Circuit state 0: No object detected, which means the object is not in the beam. The receiver does not see any light.
Transmitter Receiver		Circuit state 1: Object detected, which means the object is in the beam. The receiver does not see any light.

Reflex sensor

Table 10-198 Reflex sensor

Reflex sensor	Object	
Transmitter Receiver		Circuit state 0: No object detected, which means the object is not in the beam. The receiver sees light.
Transmitter Receiver		Circuit state 1: Object detected, which means the object is in the beam. The receiver does not see any light.

Switching hysteresis

Faults with the diffuse sensor or in the production process can result in signal wobbles. The measured value then changes the switching threshold by 100 % (object detected - object not detected). You can prevent this switching threshold wobble using the switching hysteresis parameter. This will ensure a stable output signal on the sensor.

You can assigned parameters to 5 %/10 %/20 %/50 % for switching hysteresis.

Requirements

You can only set the switching hysteresis parameter for diffuse sensors with background fadeout.

Operating principle

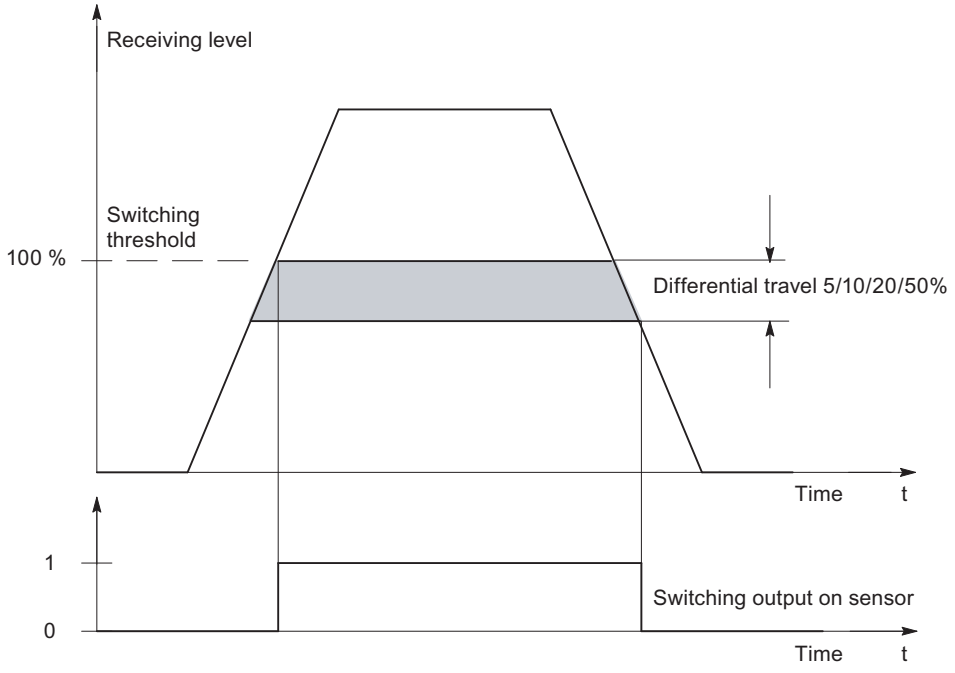


Figure 10-54 Switching hysteresis parameter

Time function,time value

These parameters can be used to set the electronic module for its specific application.

Operating principle

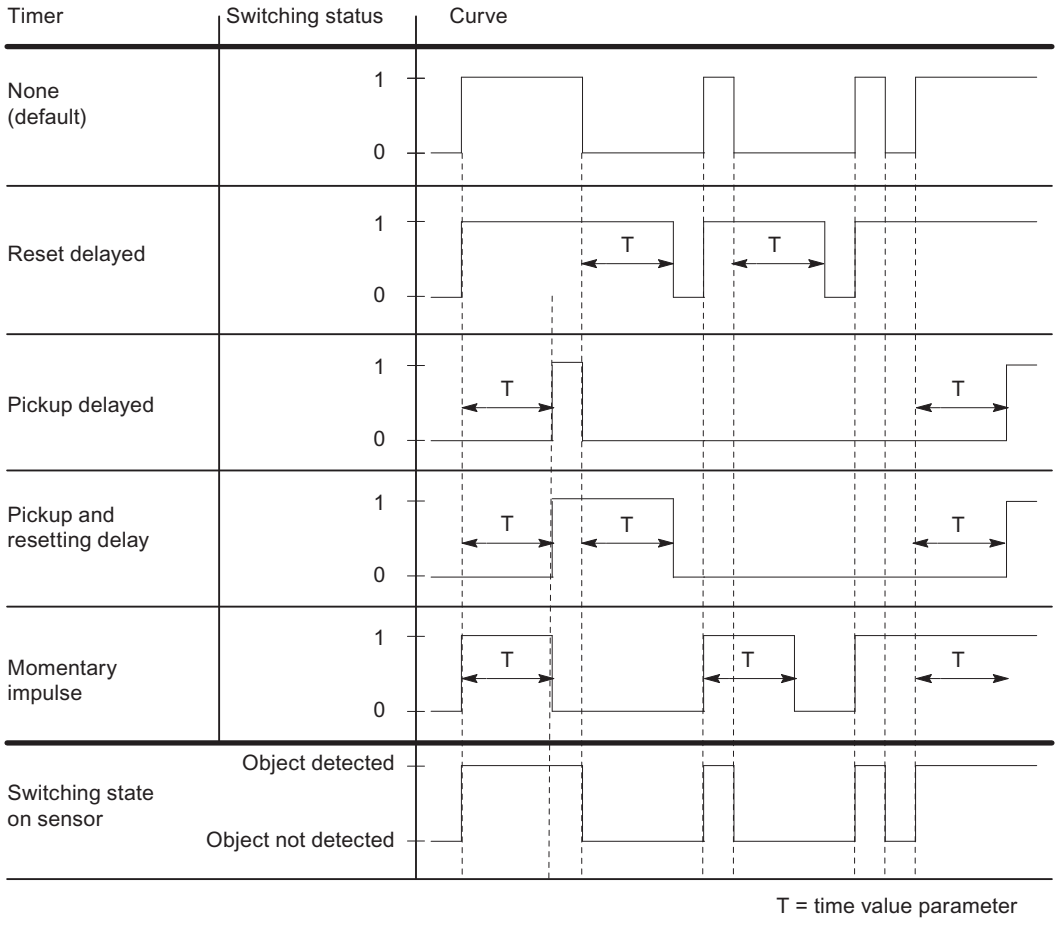


Figure 10-55 Time functions, time values parameters

Multiplex/synchronous mode

For the prevention of mutual influence between IQ Sense ultrasound devices in spatial proximity (devices with IQ profile ID 128), use the "Multiplex/synchronous operation" parameter.

Settings for the multiplex/synchronous mode parameter

Disabled: Mutual influence between IQ Sense ultrasound sensors in spatial proximity is possible (default). The cycle time is determined by the IQ Sense ultrasound sensor.

Multiplex: The IQ Sense ultrasound sensors determine the process value (distance) one after another, preventing them from affecting one another. The cycle time here is the sum of the configured synchronous cycle times of the IQ Sense ultrasound sensors that are to be multiplexed.

Synchronization: The IQ Sense ultrasound sensors determine the process value (distance) at exactly the same time, preventing them from affecting one another. The cycle time here corresponds to the greatest configured synchronous cycle time from among the IQ Sense ultrasound sensors that are to be synchronized.

You can, for example, use synchronous operation for a curtain function in which several IQ Sense ultrasound sensors aligned in parallel share a single extended detection area. The sensors simultaneously emit an ultrasound impulse. When an object enters the detection area, the sensor nearest to the object receives the echo most quickly. The object can therefore not only be detected, it can be located as well.

AFI value

Using the AFI value (application series identifier, as defined in the ISO 15693-3 international standard), transponders can be selected for different applications. Only transponders whose AFI value coincides with the value set on the sensor are processed. If a transponder has the AFI value "0", it can be identified and processed regardless of the AFI value of the sensor.

This parameter is only important if it is supported by the ident system, otherwise any value (normally "0") may be assigned.

Transponder type

Depending on the type of the transponder, you must configure whether it is an ISO transponder or a vendor-specific type.

For transponders in accordance with international standard ISO 15693, the value "1" should be selected; for all other types "0" is set. Based on this setting, one of the two possible air interface drivers is selected in the sensor.

This parameter is only important if it is supported by the ID system, otherwise any value (normally "0") may be assigned.

ET 200S

Configuring an ET 200S

Introduction

For the ET 200S series, you can find a wide range of modules in the hardware catalog under "Distributed I/O".

Assigning parameters

For information on configuration and parameter assignment, refer to "See also".

Frequency converters

Use of the frequency converter

Frequency converters

The frequency converter ICU24 and ICU24F (as fail-safe version) are modular design frequency converters that are completely embedded in the distributed I/O system ET 200S. For parameterization of both modules, please see the following.

Message frame

The message frame number and the operating mode of the module are only displayed and cannot be modified.

Application ID

You indicate the saved parameters in the frequency converter as a whole with the application ID. Enter an application ID from the value range 0 to 65535. During startup (or pull/plug), this ID is compared with the application ID stored on the converter.

Converters that work with identical applications are usually also identically parameterized and should be identified with the same application ID. Converters with the same application ID may be exchanged between each other. Copying of the complete parameterization of a converter to another converter, for example, via an MMC, is only accepted, if both have the same application ID.

Converters that work with different applications and are parameterized differently must be identified by different application IDs. This prevents a converter with unsuitable parameterization from starting on an incorrect slot, i.e. on the wrong application. This also prevents the parameterization that is saved in the converter from being accidently overwritten with any parameteriation that is stored on an MMC.

Enable diagnostic interrupt

You can enable the diagnostic interrupt for the frequency converter. If diagnostic interrupt is enabled, an OB 82 must be available in a CPU to process the diagnostic events.

See also

Documentation for the frequency converter (http://support.automation.siemens.com/WW/view/en/26291825/0/en)

ET 200pro

ET 200pro with own CPU module

ET 200pro with own CPU

The ET 200pro can be equipped with its own CPU, for example with the CPU 1516pro.

In this regard, note the following rules.

Expansion rules with the CPU 1516pro:

- You can expand the CPU 1516pro with up to 16 modules.
- The total width can be a maximum of one meter.

Expansion with wide modules

If the maximum width of one meter is exceeded in a configuration with large modules (for example frequency converters or motor starters), the configuration cannot be compiled. In this case, reduce the configuration.

Configuration control with ET 200pro

Operating principle

Through the configuration control, it is possible to change the original configuration of an ET 200pro (created by configuring with STEP 7) and to operate the ET 200pro in this modified configuration. STEP 7 is no longer required for this configuration: You communicate to the ET 200pro the slot in which a configured module is actually inserted by means of your user program.

In this case, use control data record 196. In this data record, you code which modules are missing or located in different slots in the real configuration compared to the configuration. The configuration control has no effect on the parameter assignment of the modules (for example, the enabling of diagnostic alarms).

You then call the "WRREC" instruction and use it to write the data record to the interface module of the ET 200pro.

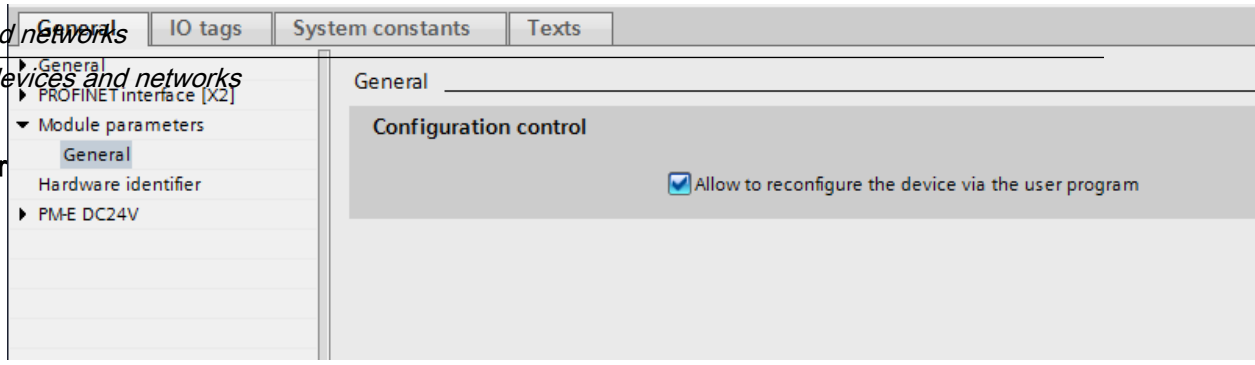
Configuration control gives you the flexibility to vary the configuration of an ET 200pro as long as the real configuration can be derived from a preset maximum configuration (originally created with STEP 7).

The following sections describe how to enable configuration control. They also outline how control data record 196 and readback data record 197 are structured.

Requirements

The CPU startup parameter "Compare preset to actual configuration" is set to Startup even if mismatch (default setting). This setting is also selected for the startup parameters of the individual modules of the ET 200pro.

Enabling configur



Configuration of control data record 196 for the ET 200pro

There is a byte in control data record 196 for each module.

The position of this byte in the data record codes one module in the original configuration with STEP 7:

- "slot_IM" (byte 4 in the data record, figure below) corresponds to the module in slot 1 in the configuration
- "slot_2" (byte 5) corresponds to the module in slot 2 in the configuration.
- "slot_3" (byte 6) corresponds to the module in slot 3 in the configuration.
- etc.

"slot_x" byte

The current slot is coded by the figure that is assigned to "slot_x" (by its value). Examples:

- The value "2" in byte 5 means you are assigning the module originally inserted in slot 2 to slot 2 in the current configuration (slot_2 = 2).
- The value "3" in byte 5 means you are assigning the module originally inserted in slot 2 to slot 3 in the current configuration (slot_2 = 3).
- The value "4" in byte 5 means you are assigning the module originally inserted in slot 2 to slot 4 in the current configuration (slot_2 = 4).
- etc.

There are no reserve modules for the ET 200pro (unlike for the ET 200S or with BU cover modules for the ET 200SP). For this reason, bit 7 of "slot_x" must not be set.

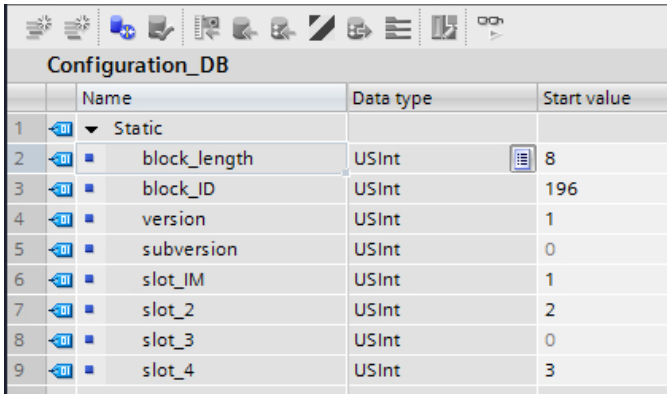
The value "0" in "slot_x" indicates that this module is not inserted in the current configuration.

Example of control data record 196

The figure below shows control data record 196 for a configuration of an ET 200pro with four modules.

This configuration is given as an example:

- The module originally configured with STEP 7 in slot 1 is also inserted in slot 1 in the current configuration.
- The module in slot 2 is in slot 2 in the current configuration.
- The module in slot 3 does not exist in the current configuration.
- The module in slot 4 is actually inserted in slot 3 in the current configuration.



Configuration_DB

	Name	Data type	Start value
1	Static		
2	block_length	USInt	8
3	block_ID	USInt	196
4	version	USInt	1
5	subversion	USInt	0
6	slot_IM	USInt	1
7	slot_2	USInt	2
8	slot_3	USInt	0
9	slot_4	USInt	3

The components of control data record 196 (figure above):

- block_length: Note here the length of the control data record; in the example: 8 (bytes). The length is calculated using the following formula: "Number of assigned slots" + 4.
- block_ID: Enter the figure 196 here.
- version: The ET 200pro uses Version 1 of control data record 196.
- subversion: The ET 200pro uses Subversion 0 of control data record 196.
- slot_IM: The IM module is always inserted in slot 0 of the ET 200pro. Slot 1 always contains the virtual power module that is integrated as a fixed module in the IM module. The "slot_IM" (name can be changed) can contain any values. This byte is not interpreted in the configuration control of the ET 200pro.
- slot_2: The configured module 2 is inserted in slot 2 (value "2").
- slot_3: The configured module 3 is not present in the current configuration (value "0").
- slot_4: The configured module 4 is inserted in slot 3 in the current configuration (value "3").

Definition of control data record 196

A control data record 196 containing a slot assignment is defined for configuration control.

Byte	Element	Value	Description
0	Block length	e.g., 8 for ET 200pro with four modules	The length of the data record is calculated using the formula: 4 + Number of modules in bytes
1	Block ID	196	ID for control data record 196
2	Version	1	Version 1 of control data record 196
3	Subversion	0	Subversion 0 of control data record 196
4	Slot_1	Any values are possible: For example "1"	This byte is not interpreted for the ET 200pro because the power module that is integrated as a fixed module in the IM module of the ET 200pro is always inserted in slot 1.

5	Slot_2	Coding of the actual slot: 2 = Slot 2 3 = Slot 3 4 = Slot 4 etc. Coding for missing module: 0 = No slot, module not present	Byte 5 indicates where the module originally configured with STEP 7 in slot 2 is actually inserted in the current configuration. Example: 2 The module originally configured with STEP 7 in slot 2 is also actually located in slot 2 in the current configuration (value "2").
6	Slot_3	Coding of the actual slot: 2 = Slot 2 3 = Slot 3 4 = Slot 4 etc. Coding for missing module: 0 = No slot, module not present	Byte 6 indicates where the module originally configured with STEP 7 in slot 3 is actually inserted in the current configuration. Example: 0 The module originally configured with STEP 7 in slot 3 is not present in the current configuration (value "0").
7	Slot_4	Coding of the actual slot: 2 = Slot 2 3 = Slot 3 4 = Slot 4 etc. Coding for missing module: 0 = No slot, module not present	Byte 7 indicates where the module originally configured with STEP 7 in slot 4 is actually inserted in the current configuration. Example: 3 The module originally configured with STEP 7 in slot 4 is actually located in slot 3 in the current configuration (value "3").
:	:	:	:

Writing a data record

Transfer the control data record to the ET 200pro module.

To do so, call the extended WRREC (Write data record) instruction, and transfer the control data record created.

If you do not transfer a control data record, the interface module uses the original configuration with STEP 7. Here, the following applies: Configured module x is inserted in real slot x.

Addressing the interface module using the HW identifier

To transfer data record 196 with the instruction WRREC, you must enter the HW identifier of the IM submodule with the extension "~Head" as the input parameter for the instruction. The system constant of this HW identifier is, for example, "IO-Device_2~Head". The system constants of a selected device are, for example, displayed in the network view in the "System constants" tab. Use the corresponding value for addressing.

Error messages

The following error messages are returned if an error occurs when writing control data record 196:

Table 10-199 Error messages

Error code	Meaning
16#80A2	DP protocol error on layer 2 Indicates that a data record has not been acknowledged due to the system.
16#80B1	Invalid length; the length information in data record 196 is not correct.
16#80B5	Configuration control parameters not assigned.
16#80B2	Invalid slot: The configured slot is not assigned.
16#80B8	Parameter error; module signals invalid parameters.
16#80C5	DP slave or module not available. Indicates that a data record has not been acknowledged due to the system.

Readback data record 197 for ET 200pro

Readback data record 197 is used to read the actual configuration of a station (in this case of an ET 200pro).

This data record allows you to check the real configuration of the ET 200pro (actual configuration). The feedback data record for each configured module specifies whether or not it is actually present.

- The value "1" means that the correct module is inserted in the correct slot.
- The value "0" codes all other options (wrong module, empty slot).

Example:

A module has been configured with STEP 7 for slot 4.

This module has then been moved to slot 3 in the current configuration using data record 196.

If this module is also really in slot 3, this is coded by the value "1" (status_slot_4 = 1).

Configuration details:

The configuration of the data block corresponds to the original configuration of the ET 200pro with STEP 7.

There is a byte in the data record for each module. The position of this bytes in the data record corresponds to the position of a module in the original configuration with STEP 7.

Sequence of bytes:

- "status_slot_IM" (byte 4 in the data record) corresponds to the module in slot 1 in the configuration,
- "status_slot_2" (byte 5) corresponds to the module in slot 2 in the configuration
- "status_slot_3" (byte 6) corresponds to the module in slot 3 in the configuration,
- etc.

The following example is for a configuration with 4 modules. The value "8" therefore appears in the "block_length" element of the data record.

If you configure an ET 200pro in STEP 7 with fewer modules, the data block will be shorter.

You can choose any name for the components of the control data record (for example "status_slot_2").

Meaning of "status_slot_x":

- The value "1" in status_slot_x means that module x is inserted in the correct slot
- The value "0" in status_slot_x codes all other options (wrong module, module does not exist).

Example:

The figure below shows readback data record 197 for an ET 200pro with four modules.

- There is no module 3 (this was assigned in control data record 196, see section "Example of control data record 196" above)
- The three other modules are actually inserted in the ET 200pro.

Record_197_ET_200pro

	Name	Datentyp	Defaultwert
1	block_length	USInt	8
2	block_ID	USInt	197
3	version	USInt	1
4	subversion	USInt	0
5	status_slot_IM	USInt	1
6	status_slot_2	USInt	1
7	status_slot_3	USInt	0
8	status_slot_4	USInt	1

Reading readback data record 197

You can read readback data record 197 from the ET 200pro with the instruction RDREC. RDREC operates asynchronously. If you call RDREC in the startup OB, you must call the instruction multiple times using a loop until the "BUSY" or "DONE" output parameter indicates that the data record has been read.

To read data record 197 with the instruction RDREC, you must enter the HW identifier of the IM submodule with the extension "~Head" as the input parameter for the instruction. The system constant of this HW identifier is, for example, "IO-Device_2~Head". The system constants of a selected device are, for example, displayed in the network view in the "System constants" tab. Use the corresponding value for addressing.

Additional information and examples

Further information on the PN interface module of the ET 200pro is available here (http://support.automation.siemens.com/WW/view/en/98099372).

Specific examples of configuration control can be found here in this application description (http://support.automation.siemens.com/WW/view/en/29430270).

You will find a library of the data records and other application examples here:

Templates for the data records (http://support.automation.siemens.com/WW/view/en/29430270)

See also

Configuration control with ET 200SP (Page 1934)

Use of the frequency converter

Frequency converters

The frequency converters ET 200pro FC and ET 200pro F-FC (as fail-safe version) are modularly design frequency converters that are completely embedded in the distributed I/O system ET 200pro. The following section describes how to configure the two modules.

Message frame

The message frame number and the operating mode of the module are only displayed and cannot be modified.

Application ID

You indicate the saved parameters in the frequency converter as a whole with the application ID. Enter an application ID from the value range 0 to 65535. During startup (or pull/plug), this ID is compared with the application ID stored on the converter.

Converters that work with identical applications are usually also identically configured and should be identified with the same application ID. Converters with the same application ID may be exchanged between each other. Copying of the complete configuration of a converter to another converter, for example, via an MMC, is only applied, if both have the same application ID.

Converters that work with different applications and are configured differently must be identified by different application IDs. This prevents a converter with unsuitable configuration from starting on an incorrect slot, in other words on the wrong application. This also prevents the configuration that is saved in the converter from being accidently overwritten with any configuration that is stored on an MMC.

Enable diagnostic interrupt

You can enable the diagnostic interrupt for the frequency converter. If diagnostic interrupt is enabled, an OB 82 must be available in a CPU to process the diagnostic events.

10.1.5.3 IPv4 routing with CP 1543-1 / CM 1542-1

IPv4 routing with CP 1543-1 / CM 1542-1

The following restrictions apply to the "IPv4 routing over the backplane bus" function of the CM/CP.

- IPv4 routing over the CM / CP to the CPU of the station is not possible.
- IP configuration with CM 1542-1 as PROFINET IO controller
  If the CM with enabled IPv4 routing over the backplane bus is used as PROFINET IO controller, please note the following points for IP configuration of the subordinate IO devices:
  If IP address configuration for the subordinate IO devices is fixed ("Set IP address in the project" option), IPv4 routing over the CM to the IO devices is not possible.
  The following settings are required in the IP configuration of the subordinate IO devices to allow IPv4 routing over the CM to those IO devices:
  - First select the "Use router" option for the IO device.
    Copy the IP address of the IO controller (CM) to the router address field.
  - Activate the "IP address is set directly at the device" option for the IO device.

10.1.5.4 IPv6 configuration

IPv6 protocol

The Internet Protocol version 6 - called IPv6 below - extends the Internet Protocol version 4 (IPv4) that is used predominantly at the current time.

Address format IPv6: Notation

IPv6 addresses consist of 8 fields each with four-character hexadecimal numbers (128 bits in total). The fields are separated by a colon.

Example: fd00:0000:0000:0000:0db8:002f:0000:0000

Rules / simplifications:

- Leading zeros within a field can be omitted.
  Example: fd00:0000:0000:0000:0db8:002f:0000:0000, abbreviated fd00:0:0:0:db8:2f:0:0
- If one or more blocks have the value 0, a shortened notation is possible with two successive colons. To ensure uniqueness, this shortened form can only be used once within the entire address. With two or more such abbreviations, it may not be possible to determine how many blocks were shortened.
  Valid example: fd00::db8:2f:0:0 or fd00:0:0:0:db8:2f:: (both clearly recognizable as fd00:0:0:0:db8:2f:0:0)
  Invalid example: fd00::db8:2f:: (not clear, can be interpreted as fd00:0:0:0:db8:2f:0:0 or fd00:0:0:db8:2f:0:0:0 etc.)
- The last 2 fields or 4 bytes can be written in the normal decimal notation with periods.
  Example: The IPv6 address fd00::db8:fd01:2f is equivalent to fd00::db8.253.1.0.47

Entry and appearance

The entry of IPv6 addresses is possible in the notations described above. IPv6 addresses are always shown in the same notation in which they were entered.

See also

IPv6 with the CP 1543-1 (Page 2021)

IPv6 with the CP 1543-1

Use of IPv6 with the CP 1543-1

For all IP services, the CP supports the Internet protocol version 4 (IPv4)

The additional specification of addresses in IPv6 format can be used with the CP for the following services and applications:

- FETCH/WRITE: Direct write/read access by PC stations, SIMATIC S5 or third-party devices
- FTP client: FTP access from the S7-1500 CPU to an FTP server with program block FTP_CMD
- FTP server: FTP access from an FTP client to data areas of the S7-1500 CPU
- SNMP: Data query using MIB objects according to SNMP
- E-mail: Data transfer from the S7-1500 CPU using the program block T_Mail

10.2 Device and network diagnostics

10.2.1 Hardware diagnostics

10.2.1.1 Overview of hardware diagnostics

Principal methods of hardware diagnostics

Principal methods of hardware diagnostics

Hardware diagnostics can be performed as follows:

- Using the Online and Diagnostics view
- Using the "Online Tools" task card
- Using the "Diagnostics > Device Info" area of the Inspector window
- Using diagnostics icons, for example, in the device view and the project tree

Structure of the Online and Diagnostics view

The Online and Diagnostics view consists of two windows alongside each other:

- The left window shows a tree structure with folders and - when you open the folder - groups.
- The right window contains detailed information on the selected folder or selected group.

The "Online access" group and the "Diagnostics" and "Functions" folders are located here:

- "Online access" group: Displays whether or not there is currently an online connection with the associated target. In addition, you can establish or disconnect the online connection.
- "Diagnostics": Contains several diagnostics groups for the selected module.
- "Functions": Contains several groups, in which you can make settings for the selected module or issue commands to the module.

Function and structure of the "Online Tools" task card

For modules with their own operating mode (such as CPUs), the "Online tools" task card allows you to read current diagnostics information and commands to the module.

If you selected a module without its own operating mode or if you selected several modules before activation of the "Online Tools" task card, the task card relates to the relevant CPU.

The "Online Tools" task card consists of the following panes:

- CPU control panel
- Cycle time
- Memory

Note

A pane is filled with content only if the module controls the associated functions and an online connection exists.

If there is no online connection to the respective module, the display "No online connection" appears in blue in each pane. If an existing online connection was disconnected, then "This target is not available" will be displayed.

Structure or the "Diagnostics" tab of the Inspector window

The "Diagnostics" tab of the Inspector window itself consists of several tabs. Of these tabs, the following is relevant for the hardware diagnostics.

- Device information
  This tab relates to all online devices (e.g. CPUs) to which an online connection has been established and to the devices that are assigned to these online devices (such as PROFINET devices and PROFIBUS slaves). Alarms related to the faulty online devices and the faulty devices are output here.

Note

What is displayed when a module is faulty?

If a module within a device is faulty, only the associated device or its proxy (e.g. head module) are shown, but not the module itself.

The faulty devices are displayed "at the top level" and not in a hierarchical view below their online device (as is the case in the project tree).

See also

Basics on task cards (Page 257)

Inspector window (Page 255)

Determination of which of the devices that are connected online are defective

Overview of the defective devices

In the "Diagnostics > Device Info" area of the Inspector window you will obtain an overview of the defective devices that are or were connected online.

The "Diagnostics> Device Info" area of the Inspector window consists of the following elements:

- Header line with the number of defective devices
- Table with detailed information on each defective device

If you originate the establishment of an online connection to a device which is not reachable or reports one or more faults or is not in RUN mode, it will rank as defective.

Structure of the table with detailed information on the defective devices

The table consists of the following columns:

- Online status: Contains the online status as a diagnostic symbol and in words
- Operating mode: Contains the operating mode as a symbol and in words
- Device / module: Name of the affected device or the affected module
- Message: explains the entry of the previous column
- Details: The link opens the online and diagnostics view for the device, and places it in the foreground. If an online connection does not exist any longer, the link will open the connection establishment dialog.
- Help: The link supplies further information on the defect that has occurred.

See also

Displaying diagnostics status and comparison status using icons (Page 2024)

Displaying diagnostics status and comparison status using icons

Determining diagnostics status online and displaying using icons

When the online connection to a device is established, the diagnostics status of the device and, if applicable, its lower-level components is determined. The operating state of the device is also determined, where applicable.

The following is a description of which icons are displayed in specific views.

- Device view
  - The associated diagnostic icon is displayed for every hardware component (except the signal board on the CPU).
    To start the Online and Diagnostics view (if available), double-click the diagnostic icon.
  - For a hardware component with lower-level components, if there is a hardware error in at least one lower-level component, the diagnostic icon appears as follows: The hardware component's diagnostic icon has a pale appearance and the diagnostic icon "Hardware error in lower-level component" is also shown in the lower right corner.
  - For hardware components with their own operating state, the operating state icon is also displayed to the left of or above the diagnostic icon.
  - The following applies to modules or submodules of a shared device with an S7-1500 CPU: No diagnostic icons are displayed (due to the configuration as GSDML device).

  Note

  Diagnostics status of SCALANCE-XM-400 and SCALANCE-XR-500-managed modules

  Incorrect or no diagnostic icons are displayed in case of an existing online connection to the SCALANCE-XM-400 and SCALANCE-XR-500-managed modules.

  The correct diagnostic icons will be displayed when you establish the online connection to an associated CPU.

- Device overview
  - The associated diagnostic icon is displayed for every hardware component.
    To start the Online and Diagnostics view (if available), double-click the diagnostic icon.
  - For a hardware component with lower-level components, if there is a hardware error in at least one lower-level component, the diagnostic icon appears as follows: The hardware component's diagnostic icon has a pale appearance and the diagnostic icon "Hardware error in lower-level component" is also shown in the lower right corner.
  - The following applies to modules or submodules of a shared device with an S7-1500 CPU: For modules assigned to the CPU, the associated diagnostic icon is displayed (modules that are not assigned do not receive a diagnostic icon). The associated diagnostic icon is displayed for plug-in submodules of an assigned module (submodules that are not pluggable are not visible and therefore do not receive a diagnostic icon).
- Network view
  - The associated diagnostic icon is displayed for every device.
    To start the Online and Diagnostics view (if available), double-click the diagnostic icon.
  - For a hardware component with lower-level components, if there is a hardware error in at least one lower-level component, the diagnostic icon appears as follows: The hardware component's diagnostic icon has a pale appearance and the diagnostic icon "Hardware error in lower-level component" is also shown in the lower right corner.
  - The following applies to modules or submodules of a shared device with an S7-1500 CPU: A diagnostic icon is displayed. It belongs to that part of the station that is assigned to the CPU.

- Network overview
  - The associated diagnostic icon is displayed for every hardware component.
    To start the Online and Diagnostics view (if available), double-click the diagnostic icon.
  - For a hardware component with lower-level components, if there is a hardware error in at least one lower-level component, the diagnostic icon appears as follows: The hardware component's diagnostic icon has a pale appearance and the diagnostic icon "Hardware error in lower-level component" is also shown in the lower right corner.
  - The following applies to modules or submodules of a shared device with an S7-1500 CPU: A diagnostic icon is displayed. It belongs to that part of the station that is assigned to the CPU.
- Topology view
  - The associated diagnostic icon is displayed for every device.
    To start the Online and Diagnostics view (if available), double-click the diagnostic icon.
  - For a hardware component with lower-level components, if there is a hardware error in at least one lower-level component, the diagnostic icon appears as follows: The hardware component's diagnostic icon has a pale appearance and the diagnostic icon "Hardware error in lower-level component" is also shown in the lower right corner.
  - The associated diagnostic icon is displayed for every port. The meaning of the individual colors is described further below.
  - Each cable between two online ports is assigned the color associated with its diagnostics status.
    The color of the cable between two ports depends on the status of the individual ports:

Color of the first port	Color of the second port	Color of the connecting cable
light green	light green	light green
light green	dark green	dark green
green	gray	gray
green	red	red
gray	red	red

  - The following applies to modules or submodules of a shared device with an S7-1500 CPU: A diagnostic icon is displayed. It belongs to that part of the station that is assigned to the CPU.
- Topology overview
  - The associated diagnostic icon is displayed for every hardware component.
    To start the Online and Diagnostics view (if available), double-click the diagnostic icon.
  - For a hardware component with lower-level components, if there is a hardware error in at least one lower-level component, the diagnostic icon appears as follows: The hardware component's diagnostic icon has a pale appearance and the diagnostic icon "Hardware error in lower-level component" is also shown in the lower right corner.
  - The following applies to modules or submodules of a shared device with an S7-1500 CPU: A diagnostic icon is displayed. It belongs to that part of the station that is assigned to the CPU.

- Project tree
  - The associated diagnostic icon is displayed behind every hardware component.
  - For a hardware component with lower-level components (e.g., distributed I/O, Slave_1), if there is a hardware error in at least one lower-level component, the diagnostic icon appears as follows: The hardware component's diagnostic icon has a pale appearance and the diagnostic icon "Hardware error in lower-level component" is also shown in the lower right corner.
  - For hardware components with their own operating state, the operating state icon is also displayed in the top right corner of the diagnostic icon.
  - If forcing is active on a CPU, a red F is displayed at the left margin of the diagnostic icon.
  - The diagnostic icon "Hardware error in lower-level component" is displayed behind the "Local modules" folder when there is a hardware error in at least one of the associated modules.
  - The diagnostic icon "Hardware error in lower-level component" is displayed behind the "Distributed I/O" folder when there is a hardware error in at least one of the associated modules.
  - The diagnostic icon "Hardware error in lower-level component" is displayed behind the project folder when the "Hardware error in lower-level component" diagnostic icon is displayed behind at least one of the "Local modules" or "Distributed I/O" folders.
  - The following applies to modules or submodules of a shared device with an S7-1500 CPU: The associated diagnostic icon is displayed for modules assigned to the CPU (modules that are not assigned are grayed out and do not receive a diagnostic icon). The associated diagnostic icon is displayed for plug-in submodules of an assigned module (submodules that are not pluggable are not visible and therefore do not receive a diagnostic icon).

Note

If the diagnostic for a hardware component is "not reachable from the CPU", the diagnostic icon "Hardware error in lower-level component" is not additionally shown.

Diagnostic icons for modules and devices

The following table shows the available icons and their meaning.

Icon	Meaning
	The connection with a CPU is currently being established.
	The CPU is not reachable at the set address.
	The configured CPU and the CPU actually present are of incompatible types.
	On establishment of the online connection to a protected CPU, the password dialog was terminated without specification of the correct password.
	No fault

Icon	Meaning
	Maintenance required
	Maintenance demanded
	Error
	The module or device is deactivated.
	The module or the device cannot be reached from the CPU (valid for modules and devices below a CPU).
	No input or output data are available, because the (sub)module has blocked its input or output channels.
	Diagnostics data are not available because the current online configuration data differ from the offline configuration data.
	The configured module or device and the module or device actually present are incompatible (valid for modules or devices under a CPU).
	The connection is established, but the module status has not yet been determined or is unknown.
	The configured module does not support display of the diagnostics status.
	Hardware error in lower-level component: A hardware fault has occurred in at least one lower-level hardware component (occurs as a separate icon only in the project tree)

Note

Some modules, for example, the FM 450-1, are only indicated as having a problem in the case of an error if you have enabled the diagnostic interrupt when assigning the module property parameters.

Icons for the comparison status

The diagnostic icons can be combined at the bottom right with additional smaller icons that indicate the result of the online/offline comparison. The following table shows the available comparison icons and their meaning.

Icon	Meaning
	Hardware error in lower-level component: The online and offline versions differ (only in the project tree) in at least one lower-level hardware component.
	Software error in lower-level component: The online and offline versions differ (only in the project tree) in at least one lower-level software component.
	Online and offline versions of the object are different
	Object only exists online
	Object only exists offline
	Online and offline versions of the object are the same

Note

If both a comparison icon and the "Error in lower-level component" diagnostic icon are to be displayed at the bottom right in the device view, the following rule applies: The diagnostic icon for the lower-level hardware component has a higher priority than the comparison icon. This means that a comparison icon is only displayed if the lower-level hardware components have no errors.

Display of software errors in the project tree

- The associated comparison icon is shown behind each block.
- Behind each folder, under which exclusively blocks are contained, the diagnostic icon "Software error in lower-level component" is displayed when there is a software error in at least one of the associated blocks.
- For a hardware component with lower-level software components, if there is no hardware error and there is an error in at least one lower-level software component, the diagnostic icon appears as follows: The hardware component's diagnostic icon has a pale appearance and the diagnostic icon "Software error in lower-level component" is also shown in the lower right corner.

Combined diagnostics and comparison icons

The following table shows examples of icons that are displayed in the diagnostics icon.

Icon	Meaning
	Folder contains objects whose online and offline versions differ (only in the project tree)
	Object only exists online

Operating state icons for CPUs and CPs

The following table shows the available icons and their respective operating states.

Icon	Operating state
	RUN
	STOP
	STARTUP
	HOLD
	DEFECTIVE

Icon	Operating state
	Unknown operating state
	The configured module does not support display of the operating state.

Note

If forcing is active on a CPU, a red F is displayed on a pink background at the bottom right of the operating state icon.

Color marking of ports and Ethernet cables

The following table shows the available colors and their respective meaning.

Color	Meaning
	No fault or maintenance required
	Offline
	Maintenance demanded
	Communication error or topological error
	no diagnostic capability

Start online and diagnostics view

Overview of possible ways of starting the Online and Diagnostics view

You can start the Online and Diagnostics view of a module to be diagnosed at the following locations:

- Overview
- Project tree
- Device view
- Device overview
- Network view
- Network overview
- Topology view
- Topological overview

In the following, examples are used to show how to proceed.

Requirement

The project with the module to be diagnosed is open.

Note

This requirement does not apply if you call the online and diagnostics view from the project tree after you have identified the accessible devices.

Procedure

To start the online and diagnostics view of a module, follow these steps:

1. In the project tree, open the respective device folder.
2. Double click on "Online & Diagnostics".

Or:

1. In the project tree, select the respective device folder.
2. Select the "Online & Diagnostics" command in the shortcut menu or the "Online" main menu.

Or:

1. In the project tree, open the "Online access" folder.
2. Open the folder for the interface with which you want to establish the online connection.
3. Double click on "Show/Update accessible devices".
4. Select the module to be diagnosed.
5. Select the "Online & Diagnostics" command in the shortcut menu or the "Online" main menu.

Or:

1. In the project tree, open the "Local modules" folder.
2. Select the respective device or the module that is to be diagnosed.
3. Select the "Online & Diagnostics" command in the shortcut menu or the main menu.

Or:

1. Open the device view in the device configuration.
2. Select the module to be diagnosed.
3. Select the "Online & Diagnostics" command in the shortcut menu or the "Online" main menu.

Or:

1. Open the device view in the device configuration.
2. Establish an online connection to the module to be diagnosed.
3. Double-click on the diagnostics icon above the module.

Or:

1. Open the network view in the device configuration.
2. Select the station with the module to be diagnosed.
3. Select the "Online & Diagnostics" command in the shortcut menu or the "Online" main menu.

Or:

1. Open the topology view in the device configuration.
2. Establish an online connection to the module to be diagnosed.
3. In the topology view, double-click the diagnostic icon associated with the module.

Result

The online and diagnostics view of the module to be diagnosed will be started. If an online connection to the associated CPU had previously been created, the header bar of the Online and Diagnostics view will now have an orange background.

Note

If no online connection exists when the online and diagnostics view is started, no online information is displayed and the display fields remain empty.

Activation of the "Online Tools" task card

Activation of the "Online Tools" task card

You can activate this task card as follows:

1. Start the online and diagnostics view.
2. Click on the "Online Tools" task card.

Or:

1. Start the device view.
2. Click on the "Online Tools" task card.

Or:

1. Start the network view.
2. Click on the "Online Tools" task card.

10.2.1.2 Showing non-editable and current values of configurable module properties

Showing general properties and system-relevant information for a module

Where do I find the information I need?

The general properties and system-relevant information for a module can be found in the "General" group in the "Diagnostics" folder in the online and diagnostics view of the module to be diagnosed.

Structure of the "General" group

The "General" group consists of the following areas:

- Module
- Module information
- Vendor information

"Module" area

This area shows the following data of the module:

- Short designation, for example, CPU 1214C DC/DC/DC
- Article no.
- Hardware
- Firmware
- TIA Portal version

  Note

  TIA Portal version

  The displayed TIA Portal version is the minimum version that is required to open the blocks loaded to the device or to load the blocks to the ES. This means it does not necessarily have to be the version with which you have loaded the blocks to the device.

  Fictitious example: You have installed the TIA Portal in version V14.0 SP1 on your ES and are creating a new project. This project is assigned project version V14.0. You download this project (along with its project version) to the CPU. The project version, which means V14.0, is displayed.

- Racks
- Slot

"Module information" area

This area shows the following data of the module that you configured during hardware configuration:

- Module name
- Installation date (not displayed for all modules)
- Additional information (not displayed for all modules)

"Manufacturer information" area

This area shows the following data of the module:

- Manufacturer
- Serial number
- Profile: Profile ID as hexadecimal number

  Note

  You will find the corresponding profile name in the profile ID table for PROFIBUS International (see "www.profibus.com").

- Profile details: Profile-specific type as hexadecimal number

  Note

  You will find the corresponding profile-specific type name in the profile-specific type table for PROFIBUS International (see "www.profibus.com").

Display configured cycle times

Where do I find the information I need?

The required information can be found in the following places:

- In the "Cycle time" group of the "Diagnostics" folder in the Online and Diagnostics view of the module to be diagnosed.
- In the "Cycle time" pane of the "Online Tools" task card

Structure of the "Cycle time" group in the "Diagnostics" folder of the Online and Diagnostics view

The "Cycle time" group consists of the following areas:

- Cycle time diagram (graphical display of the assigned and measured cycle times)
- Cycle time configured (display of the assigned cycle times as absolute values)
- Cycle times measured (display of the measured cycle times as absolute values)

Structure of the "Cycle time" pane of the "Online Tools" task card

The "Cycle time" pane displays the cycle time diagram and below it the measured cycle times as absolute values.

Assigned cycle times

The following assigned cycle times are displayed in the cycle time diagram and in the "Cycle time configured" area.

- Minimum cycle time
- Maximum cycle time

In the cycle time diagram, the minimum cycle time and the maximum cycle time correspond to the two markings on the time axis.

In the "Cycle time configured" area, the assigned cycle times are displayed as absolute values.

Show interfaces and interface properties of a module

Where do I find the information I need?

The interfaces and interface properties of a module can be found in the "Diagnostics" folder in the online and diagnostics view of the module to be diagnosed in the following group:

- PROFINET interface

"PROFINET interface" group

This group is divided into the following areas:

- "Ethernet address" with the "Network connection" and "IP Parameters" subareas
- "Ports"

"Network connection" subarea of the "Ethernet address" area

This subarea shows the following data for the module:

- MAC Address:
  MAC address of the interface.
  The MAC address consists of two parts. The first part ("Basic MAC address") identifies the manufacturer (Siemens, 3COM, ...). The second part of the MAC address differentiates between the various Ethernet devices. Each Ethernet module is assigned a unique MAC address.

"IP Parameters" subarea of the "Ethernet address" area

This subarea shows the following data for the module:

- IP address:
  Internet protocol address of the device on the bus (TCP/IP)
- Subnet mask:
  The subnet mask shows which part of the IP address determines the membership of a particular sub-network.
- Default router:
  If the subnet is connected via a router to other subnets, the IP address of the default router must be known. This is the only way a datagram can be forwarded with a non-matching subnet address.
- IP settings:
  Identifier for the path by which the device has obtained its IP settings (IP address, subnet mask, default router).

Identifier	Meaning
0	IP address is not initialized
1	By configuration (i.e., by the configuration loaded to the device from the device or network view)
2	Via the "Assign IP address" group of the online and diagnostics view
3	Via the DHCP server (i.e., the IP parameters are obtained by a special service from a DHCP server (Dynamic Host Configuration Protocol) and assigned for a limited time)
4	IP address is set by a user program
5	Source of IP address unknown

- IP setting time:
  Time stamp of the last change to the IP address directly through the Ethernet connection of the module

"Ports" area

This area shows the following data for the module:

- Ethernet ports
  Physical properties of the PROFINET interface

Properties of the PROFINET interface	Meaning
Port no.	Port number The short description of interface (X + interface no.) and port (P + port no.) is specified in parentheses. An "R" in the short description of a port means that it is a ring port.
Status	Displays the status of the port LINK LED. • Status "OK" means another device (such as a switch) is connected to the port and the physical connection is available. • Status "disconnected" means no other device is connected to the port. • Status "deactivated" means that access to the port is blocked.
Settings	• "Automatic" for automatic network settings of the device • Network settings for speed and transmission method for manual network settings of the device
Operating mode	Network settings for speed and transmission method

If you select a line in the port table, additional help information will be provided for the corresponding port.

Displaying IO controllers that access modules of a Shared Device (S7-1500)

Where do I find the information I need?

The display of those IO controllers that access the modules of a Shared Device can be found in the Online and Diagnostics view of the interface module of the Shared Device in the "Diagnostics" folder in the following area of the "PROFINET interface" group:

- IO controller

Displaying sync domain properties of a PROFINET device

Where do I find the information I need?

The sync domain properties of a PROFINET device can be found in the following area of the "PROFINET interface" group in the "Diagnostics" folder of the Online and Diagnostics view of the device to be diagnosed:

- Domain

"Domain" area

This area is divided into the following subareas:

- Sync domain
- MRP domain

What is a sync domain?

A sync domain is a group of PROFINET devices that are synchronized to a common clock. Exactly one device has the role of the sync master (clock generator); all other devices assume the role of a sync slave. The sync master is usually an IO controller or a switch.

Non-synchronized PROFINET devices are not part of a sync domain.

"Sync domain" subarea of the "Domain" area

This subarea shows the following properties of the sync domain:

- Name:
  Name of sync domain
- Role:
  Role of the PROFINET device in the sync domain. The following roles are possible:
  - Sync master
  - Sync slave
- Synchronization interval:
  Interval at which the synchronization is performed
- Send clock
  Smallest possible send interval for the data exchange
- Jitter accuracy of the send clock
- Reserved bandwidth for cyclic communication

Displaying MRP domain properties of a PROFINET device

Where do I find the information I need?

The MRP domain properties of a PROFINET device can be found in the following area of the "PROFINET interface" group in the "Diagnostics" folder of the Online and Diagnostics view of the device to be diagnosed:

- Domain

"Domain" area

This area is divided into the following subareas:

- Sync domain
- MRP domain

What is an MRP domain?

The Media Redundancy Protocol (MRP) enables redundant networks to be structured. Redundant transmission paths (ring topology) ensure that, if one transmission path fails, an alternative communication path is available. The PROFINET devices that are part of this redundant network form an MRP domain.

"MRP domain" subarea of the "Domain" area

This subarea shows the following properties of the MRP domain:

- Name:
  Name of MRP domain
- Role:
  Role of the PROFINET device in the MRP domain. The following roles are possible:
  - Manager
  - Manager (Auto)
  - Client
  - Not a device of the ring
- Ring port 1:
  The port of the PROFINET device that has the "Ring port 1" property
- Ring port 2:
  The port of the PROFINET device that has the "Ring port 2" property
- Status of the MRP ring:
  Indicates whether the ring is interrupted ("open" status) or not ("closed" status).

Displaying current firmware of a module (S7-300, S7-400, S7-1500)

Displaying firmware

You can display the currently installed firmware of a module.

Requirements

- The module supports a firmware update.
- The module is connected online.

Procedure

To display the current firmware, follow these steps:

1. Open the module in the Online and Diagnostics view.
2. Select the "Firmware update" group in the "Functions" folder.
3. You read off the current firmware in the "Online data" area under "Firmware".

10.2.1.3 Showing the current values of dynamic modules properties

Display measured cycle times

Where do I find the information I need?

The measured cycle times can be found at each of the following places:

- In the "Cycle time" group of the "Diagnostics" folder in the Online and Diagnostics view of the module to be diagnosed.
- In the "Cycle time" pane of the "Online Tools" task card

Structure of the "Cycle time" group in the "Diagnostics" folder of the Online and Diagnostics view

The "Cycle time" group consists of the following areas:

- Cycle time diagram (graphical display of the assigned and measured cycle times)
- Cycle time configured (display of the assigned cycle times as absolute values)
- Cycle times measured (display of the measured cycle times as absolute values)

Structure of the "Cycle time" pane of the "Online Tools" task card

The "Cycle time" pane displays the cycle time diagram and below it the measured cycle times as absolute values.

Graphical display of the measured cycle times

The following measured cycle times are displayed in the cycle time diagram:

- Shortest cycle time: Duration of the shortest cycle since the last transition from STOP to RUN
  This corresponds to the dashed gray arrow on the left in the diagram.
- Current / last cycle time: Duration of the last cycle
  This corresponds to the green arrow in the diagram. If the current / last cycle time exceeds the maximum cycle time, the arrow will turn red.

  Note

  If the duration of the last cycle comes close to the maximum cycle time, it may be possible that it will be exceeded. Depending on the CPU type, parameter assignment and your user program, the CPU can switch to STOP mode. If for instance you are monitoring the tags in your program, this will increase the cycle time.

  If the cycle lasts longer than double the maximum cycle time, and you do not restart the maximum cycle time in the user program (by calling the extended RE_TRIGR) instruction, the CPU will switch to STOP mode.

- Longest cycle time: Duration of the longest cycle since the last transition from STOP to RUN.
  This corresponds to the dashed blue arrow on the right in the diagram.

A blue band extends between the two dashed lines; this band corresponds to the entire range of the measured cycle times. If a measured cycle time is greater than the maximum cycle time, the portion of the band that lies outside the assigned limits will be colored red.

Display of the measured cycle times as absolute values

The following measured times are displayed in the "Cycle times measured" area and in the "Cycle time" pane.

- Shortest cycle time since the last transition from STOP to RUN.
- Current/last cycle time:
- Longest cycle time since the last transition from STOP to RUN.

Showing the current status of the LEDs of a CPU

Where do I find the information I need?

The current status of the LEDs of a CPU can be found in the display area of the "CPU control panel" pane of the "Online tools" task card.

Display area of the "CPU control panel" pane of the "Online Tools" task card

This area contains the following displays:

- Station name and CPU type (short designation)
- RUN / STOP (corresponds to the "RUN / STOP" LED of the CPU)
- ERROR (corresponds to the "ERROR" LED on the CPU)
- MAINT (corresponds to the "MAINT" LED on the CPU)

Showing fill levels of all types of memory on a CPU

Where do I find the information I need?

The fill levels of all types of memory on a CPU can be found on the following two pages:

- In the display area of the "Memory" group in the "Diagnostics" folder in the online and diagnostics view of the module to be diagnosed
- In the display area of the "Memory" pane on the "Online Tools" task card

Display area of the "Memory" group in the "Diagnostics" folder of the online and diagnostics view

This area contains the current memory utilization of the associated module and details of the individual memory areas.

The memory utilization is shown both as a bar diagram and as a numerical value (percentage).

The following memory utilizations are shown:

- Load memory
  If no memory card is inserted, the internal load memory is displayed.
  If a memory card is inserted, the operating system only uses the inserted load memory as the load memory. This is displayed here.
- Work memory
- Retentive memory

Display area of the "Memory" pane of the "Online Tools" task card

This area contains the current memory utilization of the associated module. The available memory is shown both as a bar diagram and as a numerical value (percentage). The numerical value is rounded to an integer value.

Note

If less than 1% of a memory area is utilized, the available portion of this memory area is shown as "99%".

The following memory utilizations are shown:

- Load memory
  If no memory card is inserted, the internal load memory is displayed.
  If a memory card is inserted, the operating system only uses the inserted load memory as the load memory. This is displayed here.
- Work memory
- Retentive memory

Displaying fill level of all types of memory of an S7-1500 CPU (S7-1500)

Where do I find the information I need?

The fill levels of all types of memory of an S7-1500 CPU can be found at the following two places:

- In the display area of the "Memory" group in the "Diagnostics" folder in the online and diagnostics view of the module to be diagnosed
- In the display area of the "Memory" pane on the "Online Tools" task card

Display area of the "Memory" group in the "Diagnostics" folder of the online and diagnostics view

This area contains the current memory utilization of the associated module and details of the individual memory areas.

The memory utilization is shown both as a bar diagram and as a numerical value (percentage).

The following memory utilizations are shown:

- Load memory

  Note

  The load memory is located on the SIMATIC memory card.

- Code work memory: work memory for program code
- Data work memory: work memory for data blocks
- Retentive memory

Display area of the "Memory" pane of the "Online Tools" task card

This area contains the current memory utilization of the associated module. The available memory is shown both as a bar diagram and as a numerical value (percentage). The numerical value is rounded to an integer value.

Note

If less than 1% of a memory area is utilized, the available portion of this memory area is shown as "99%".

The following memory utilizations are shown:

- Load memory

  Note

  The load memory is located on the SIMATIC memory card.

- Code work memory: work memory for program code
- Data work memory: work memory for data blocks
- Retentive memory

10.2.1.4 Checking a module for defects

Determining the diagnostic status of a module

Where is the diagnostics status of a module displayed?

The diagnostic status of a module is displayed in the "Diagnostic status" group in the "Diagnostics" folder in the online and diagnostics view of the module to be diagnosed.

The "Diagnostics status" group consists of the following areas:

- Status
- Standard diagnostics (for S7-300 and S7-400 only for non-CPU modules)

"Status" area

The following status information is displayed in this area:

- Status of the module as viewed by the CPU, for example:
  - Module available and OK.
  - Module defective.
    If the module experiences a fault and you have enabled the diagnostic error interrupt during configuration, the "Module defective" status is displayed.
  - Module configured, but not available.
    Example: Diagnostics data is not available because the current online configuration differs from the offline configuration.
- Detected differences between the configured and the inserted module. Provided it can be ascertained, the article number will be displayed for the set and actual type.

The scope of the displayed information depends on the selected module.

"Standard diagnostics" area

The following diagnostics information for non-CPU modules is displayed in this area:

- Internal and external faults that relate to the overall module
- Associated diagnostics events

Examples of such diagnostics information are:

- Entire backup failed
- Module defective

Note

Diagnostic interrupts

A diagnostic interrupt can be reported to the CPU only if the module has diagnostic interrupt capability and the diagnostic interrupt has been enabled.

The display of the diagnostic interrupt is a snapshot. Sporadic module defects can be identified in the diagnostics buffer of the respective CPU.

Performing communication diagnostics of PROFINET ports

Where are port-specific errors of PROFINET ports displayed?

The port-specific errors of PROFINET ports are displayed in the "Communication diagnostics" group of the "Diagnostics" folder in the Online and Diagnostics view of the module to be diagnosed.

Structure of the "Communication diagnostics" group

The "Communication diagnostics" group consists of the following areas:

- Table containing the port-specific errors
- "Details of the event" text field
- "Help on event" text field

Structure of the table containing the port-specific errors

Each line of the error table corresponds to one error.

The table consists of the following columns:

- Name: port no. and port designation, as well as diagnostics icon
- Error: description of the error that occurred on this port

"Details" text field

If you select a line in the error table, the "Detail" text field will contain detailed information (if available) on the corresponding error.

"Help on selected diagnostic line" text field

If you select a line in the error table, the "Help on selected diagnostic line" text field will contain help information (if available) on the corresponding error.

PROFINET IO component diagnostics

Where do I perform PROFINET IO component diagnostics?

You perform the PROFINET IO component diagnostics in the "PROFINET IO diagnostics" group under "PROFINET interface" in the "Diagnostics" folder of the Online and Diagnostics view.

"PROFINET IO diagnostics" group

Manufacturer-related diagnostics texts for the IO device or the selected module of the IO device are displayed in this area.

If the diagnostics can be read by the IO controller, then communication errors and configuration errors (online/offline discrepancies) are displayed here.

Reading out the diagnostics buffer of a CPU

Where do you read out the diagnostics buffer of a CPU?

You read out the diagnostics buffer of a CPU in the "Diagnostics buffer" group in the "Diagnostics" folder in the Online and Diagnostics view.

Structure of the "Diagnostics buffer" group

The "Diagnostics buffer" group consists of the following areas:

- "Events"
- "Settings"

Diagnostics buffer

The diagnostics buffer is used as a log file for the diagnostics events that occurred on the CPU and the modules assigned to it. These are entered in the order of their occurrence, with the latest event shown at the top.

"Events" area

The "Events" area consists of the following elements:

- Check box "CPU time stamp takes into account local PG/PC time"
- Event table
- "Freeze display" or "Cancel freeze" button
- Details of the event: Event no., event ID, module and possibly station or device name, rack/slot, description, help on the event, plant designation, location identifier, coming/going information, event type
- "Open in editor", "Save as" buttons

Check box "CPU time stamp takes into account local PG/PC time"

If you have not activated the check box, the diagnostics buffer entries are shown with the module time.

If you have activated the check box, the diagnostics buffer entries are shown with the time given by the following formula:

Displayed time = module time + time zone offset on your programming device / PC

This requires the module time to be identical to UTC.

You should use this setting if you wish to see the times of the diagnostics buffer entries for the module expressed in the local time of your programming device / PC.

Selecting or clearing the check box immediately changes the times displayed for the diagnostics buffer entries.

Note

If you use the "WR_SYS_T" instruction in your program or if you set the real-time clock of the CPU using an HMI device instead of using UTC, we recommend that you clear the "CPU time stamp takes into account local PG/PC time" check box. In this case, the module time is the sole time of concern.

Event table

The following information is displayed in the table for each diagnostics event:

- Sequential number of the entry
  The first entry contains the latest event.
- Date and time of the diagnostics event
  If no date and time are shown, the module has no integrated clock.
- Short name of the event and, if applicable, the reaction of the CPU

  Note

  If an individual parameter of a text cannot be determined, the character string "###" is shown in its place.

  If no display text is yet available for new modules or new events, the numbers of the events and the individual parameters are stated as hexadecimal values.

- Only for S7-1200 and S7-1500 CPUs: Icon for the event type
  The following table shows the available icons and their meaning.

Icon	Meaning
	OK (no maintenance and/or no fault)
	Maintenance required
	Maintenance demanded
	Error

- Icon for information related to incoming/outgoing status
  The following table shows the available icons and their meaning.

Icon	Meaning
	Incoming event
	Outgoing event
	Incoming event to which there is no independent outgoing event
	User-defined diagnostics event

You can change the order of the columns, adjust the column widths and remove and add individual columns in the event table. In addition, you can sort as follows: by sequential number, by "Date and time" and by "Event".

"Freeze display" or "Cancel freeze" button

The "Freeze display" or "Cancel freeze" button is only enabled when there is an online connection to the CPU.

The default setting is "Freeze display".

The following happens when you click the "Freeze display" button:

- The current display of the diagnostics buffer entries is frozen.
- The labeling of the button changes to "Cancel freeze".

If an error has occurred in your system, diagnostics events can occur very quickly in succession. This produces a high update rate on the display. Freezing the display allows you to calmly examine the situation in more detail.

If the display is frozen and you click the "Cancel freeze" button, the following happens:

- The display of the diagnostics buffer entries is updated again.
- The labeling of the button changes to "Freeze display".

Note

If you freeze the diagnostics buffer display, the CPU continues to enter events in the diagnostics buffer.

Details of the event

If you select a line in the list of events, you obtain detailed information on the respective event:

- Sequential number of the event in the diagnostics buffer
- Event ID
- Module and, if applicable, station or device name
- Rack/slot
- Description of the event with event-specific additional information. Examples of this additional information:
  - Command that caused the event
  - Operating mode switch caused by the diagnostics event
- Help on the event: The selected event is explained in more detail and possible solutions may be specified. With outgoing events, the text "Outgoing event: "No user action required" is displayed.
- Only for S7-1200 and S7-1500 CPUs: Plant designation, location identifier

- Information on whether the event is an incoming or outgoing event
- Type of event. The following types of event are possible:
  - OK (no maintenance and/or no fault)
  - Maintenance required
  - Maintenance demanded
  - Error

"Open in editor" button

The following table shows if the "Open block" button is active and which function it conceals.

When is the "Open in editor" button enabled?	What happens when you click this button?
If the diagnostics event references the relative address of a block. This is the address of the command that caused the event.	The "Open in editor" function opens the referenced block in the offline view at the programming instruction that causes the error. This allows you to check and, if necessary, change the source code of the block at the specified place and then download it again to the CPU.
If the diagnostics event was triggered by a module.	The "Open in editor" function opens the Device view of the module involved.

"Save as ..." button

If you click this button, the content of the diagnostics buffer is saved in a text file. "Diagnostics", depending on the language, with the extension ".txt" is suggested as the file name. You can however change this name.

"Settings" area

The "Settings" area consists of the following elements:

- "Display events" list
- "Apply settings as default" button
- "Output event information in hexadecimal format" check box

List "Display events:"

There is an check box in this list for every event class (default setting: all check boxes are selected). If you clear a check box, the events of that event class is no longer displayed in the "Events" area. Reselecting the check box displays the associated events once again.

"Apply settings as default" button

If you click this button, the settings are also applied to future occasions when the "Events" tab is opened.

"Output event information in hexadecimal format" check box

If you select the check box, the event IDs in the Events list of the "Events" area is displayed in hexadecimal format. If you clear the check box, the event information is given in text form.

See also

Basic information on the diagnostics buffer (Page 2070)

Saving service data (S7-1500)

Purpose

In the event of servicing it may be possible that the SIEMENS Customer Support requires very special information about the state of a module of your system for diagnostic purposes.

If such a case occurs in your system, you will be asked by Customer Support to save the service data of the module and send the resulting file to them.

Where do you carry out the saving of service data of a module?

You carry out the saving of service data of a module in its online and diagnostics view at the following points: In the "Functions" folder in the "Save service data" group"

The "Save service data" group consists of the following areas:

- Online data
- Saving service data

"Online data" area

This area shows the following data of the module:

- Article number
- Firmware version
- Module name (you configured this while configuring the hardware.)
- Rack
- Slot

"Save service data" area

Proceed as follows to create and save a file with special service data:

1. Select the point in the file system at which you want to save the file:
   - You use the path preset in the "Path" field.
   - Click the three-dot (browse) button. In the dialog that opens specify the desired path and enter the file name.
2. Click the "Save data" button.

10.2.1.5 Changing the properties of a module or the programming device/PC

Changing the mode of a CPU

Requirement

There is an online connection to the CPU whose mode you want to change.

Procedure

To change the mode of the CPU, follow these steps:

1. Enable the "Online tools" task card of the CPU.
2. Click the "RUN" button in the "CPU control panel" pane if you want to change the CPU to RUN mode or the "STOP" button if you want to change the CPU to STOP mode.

   Note

   The only button active is the one that can be selected in the current operating mode of the CPU.

3. Acknowledge the confirmation prompt with "OK".

Or:

1. Open the "Online" menu.
2. Choose the "Start CPU" menu command if you want to set the CPU to RUN mode and "Stop CPU" if you want to set the CPU to STOP mode.

   Note

   The only button that is active is the one that can be chosen in the current operating mode of the CPU.

3. Acknowledge the confirmation prompt with "OK".

Or:

1. Click the "Start CPU" button in the toolbar if you want to set the CPU to RUN mode and the "Stop CPU" button if you want to set the CPU to STOP mode.

   Note

   The only button that is active is the one that can be chosen in the current operating mode of the CPU.

2. Acknowledge the confirmation prompt with "OK".

Result

The CPU will be switched to the required operating mode.

Performing a memory reset

Requirement

- There is an online connection to the CPU on which the memory reset is to be performed.
- The CPU is in STOP mode.

Note

If the CPU is still in RUN mode and you start the memory reset, you can place it in STOP mode after acknowledging a confirmation prompt.

Procedure

To perform a memory reset on a CPU, follow these steps:

1. Enable the "Online Tools" task card of the CPU.
2. Click the "MRES" button in the "CPU control panel" pane.
3. Acknowledge the confirmation prompt with "OK".

Result

The CPU is switched to STOP mode, if necessary, and the memory reset is performed on the CPU.

Determining and setting the time of day on a CPU

Where do I find the functions I need?

You determine and change the time of day on a CPU in the "Set time of day" group in the "Functions" folder of the Online and Diagnostics view. This requires an online connection.

Structure of the "Set time of day" group

The "Set time of day" group consists of the following areas:

- Area for reading out and setting the time of day
- Time system (This area does not exist for S7-1200 and will not be examined here.)

Structure of the area for reading out and setting the time of day

This area consists of the following parts:

- Programming device / PC time
  Here the time zone setting, the current date and the current time setting of your programming device / PC are displayed.
- Module time
  Here the date and time values currently read from the module (for example the CPU), are converted to local time and date and displayed.
  If the "Take from PG/PC" check box is selected, when you click the "Apply" button, the date and the PG/PC time converted to UTC are transferred to the module.
  If the "Take from PG/PC" check box is not selected, you can assign the date and time for the integrated clock of the module. After clicking the "Apply" button, the date and the time recalculated to UTC time are transferred to the module.

Resetting an S7-1200 CPU to factory settings (S7-1200)

Requirement

- There is no memory card inserted in the CPU.
- There is an online connection to the CPU that you want to reset to the factory settings.
- The CPU is in STOP mode.

Note

If the CPU is still in RUN mode and you start the reset operation, you can place it in STOP mode by answering the security prompt with yes.

Procedure

To reset an S7-1200 CPU to factory settings, follow these steps:

1. Open the Online and Diagnostics view of the CPU.
2. Select the "Reset to factory settings" group in the "Functions" folder.
3. Select the "Keep IP address" check box if you want to keep the IP address or the "Delete IP address" check box if you want to delete the IP address.

   Note

   The two check boxes mentioned are only available if the module to be reset is able to choose whether to retain or delete the IP address.

4. Click the "Reset" button.
5. Acknowledge the confirmation prompt with "OK".

Result

The module is switched to STOP mode if necessary and the settings are then reset to factory settings. This means:

- The work memory and the internal load memory and all operand areas are cleared.
- All parameters are reset to their defaults.
- The diagnostic buffer is cleared.
- The time is reset.
- The IP address is kept or deleted depending on which setting you made.

Resetting an S7-1500 CPU to factory settings (S7-1500)

Requirement

- If you start a reset to factory settings from the project context, an online connection to the relevant CPU must exist.
- The relevant CPU is in STOP mode.

Note

If the CPU is still in RUN mode and you start the reset operation, you can place it in STOP mode after acknowledging a confirmation prompt.

Procedure

To reset an S7-1500 CPU to factory settings, follow these steps:

1. Open the Online and Diagnostics view of the CPU (either from the project context or via "Accessible devices").
2. Select the "Reset to factory settings" group in the "Functions" folder.
3. Select the "Keep IP address" check box if you want to keep the IP address or the "Delete IP address" check box if you want to delete the IP address.

   Note

   With "Delete IP address", all IP addresses are deleted. This applies regardless of how you created the online connection.

   If a memory card is inserted, selecting the "Delete IP address" option causes the following: The IP addresses are deleted and the CPU is reset to factory settings. Then, the configuration (including IP addresses) that is stored on the memory card is transferred into the CPU (see below). If the memory card was formatted before resetting to factory settings or if it is empty, no IP address is transferred into the CPU.

4. Click the "Reset" button.
5. Acknowledge the confirmation prompt with "OK".

Result

The module is switched to STOP mode if necessary and the settings are then reset to factory settings. This means:

- The work memory and the internal retentive system memory and all operand areas are cleared.
- All parameters are reset to their defaults.
- The diagnostic buffer is cleared.
- The time of day is reset.
- The I&M data are deleted except for I&M0 data.
- The runtime meters are reset.
- The IP address is kept or deleted depending on which setting you made.
- If a memory card was inserted prior to the reset to factory settings, the configuration contained on the memory card (hardware and software) is downloaded to the CPU.

Resetting a PROFINET IO device to factory settings (S7-1500)

Requirements

- There must be an online connection to the PROFINET IO device that is to be reset.

Procedure

To reset a PROFINET IO device to factory settings, follow these steps:

1. Open the Online and Diagnostics view of the device (either from the project context or via "Accessible devices").
2. In the "Functions" folder, select the "Reset to factory settings" group.
3. Select the "Retain identification and maintenance data" option button or the "Delete identification and maintenance data" option button, depending on whether you want to retain or delete the identification and maintenance data IM0 to IM3 on the device.
4. Click the "Reset" button.

5. Click "OK" in response to the confirmation prompt.
   If the PROFINET IO device to be reset supports the Reset-to-Factory command according to PROFINET V2.3, its PROFINET device name, its IP address and the SNMP parameters (SNMP: Simple Network Management Protocol) are reset. Your selection in step 3 determines whether the identification and maintenance data is retained or deleted.
   The reset process is now complete.
6. If the PROFINET IO device to be reset does not support the Reset-to-Factory command according to PROFINET V2.3, however, you will see a message informing you that all data on the module will be lost during a reset to factory settings.
   Confirm the security prompt. The reset process will be conducted.
   If you have selected "Delete identification and maintenance data" in step 3, the identification and maintenance data IM0 to IM3 is deleted during the reset.
   However, if you have selected "Retain identification and maintenance data" in step 3, it can still happen that the identification and maintenance data IM0 to IM3 is deleted during the reset: The reset behavior depends on the module in this case.

Result

The PROFINET IO device is reset. The data deleted in the process depends on the factors described above.

Formatting an S7-1500 memory card (S7-1500)

Requirement

- If you start the formatting of the memory card from the project context, an online connection to the relevant CPU must exist.
- The relevant CPU is in STOP mode.

Note

If the CPU is still in RUN mode and you start a formatting operation, you can place it in STOP mode after acknowledging a confirmation prompt.

Procedure

To format an S7-1500 memory card, follow these steps:

1. Open the Online and Diagnostics view of the CPU (either from the project context or via "Accessible devices").
2. Select the "Format memory card" group in the "Functions" folder.
3. Click the "Format" button.
4. Answer the safety prompt with "Yes".

Result

- The memory card is formatted.
- The CPU is temporarily unavailable.
- The project data on the CPU are deleted with the exception of the IP address.
- If you start the formatting of the memory card from the project context, the Online and Diagnostics view remains open. If formatting is started via "Accessible devices", the Online and Diagnostics view will close.

Updating the firmware of a module

Basic information on the firmware update of a module

Performing a firmware update

Using firmware files, you can update the firmware of a module.

Requirements

- The module is connected online.
- The module supports a firmware update.
- For those modules that require supply voltage to perform the firmware update correctly: The supply voltage of the module is ensured. For details, see the documentation of the module.

Procedure

You can call up the firmware update of a module from the project tree or from the project context.

⚠ WARNING

Impermissible system states possible

An S7-1500 CPU immediately goes to STOP mode when you start the firmware update, which can have an effect on the operation of an online process or a machine.
Unexpected operation of a process or a machine can result in severe or fatal injuries and/or damage to property.

Note

After you have run a firmware update, you will need to replace the affected module with the same module with the current firmware version in the hardware configuration of your project. The engineering configuration then matches the actual physical configuration again.

"Run firmware after update" check box

If you have not selected the "Run firmware after update" check box, the previous firmware remains active until the module is reset (for example by power cycling). The new firmware only becomes active after the module has been reset.

If you have selected the check box, the module is automatically reset after the firmware has been downloaded and it then runs with the new firmware.

Activating the firmware following the update has the following consequences:

- A station executes a restart. All modules in the station become unavailable as a result.
- If the corresponding CPU is in RUN, activating the firmware can lead to access errors or other problems in the user program and might even mean that the CPU remains permanently in STOP.

Note

For some CPUs, the "Run firmware after update" check box is grayed out and cleared. In this case, you must restart the CPU manually.

For S7-1500 CPUs, the "Run firmware after update" check box is grayed out and selected. In this case, the new firmware is activated immediately after the download operation.

Calling the firmware update from the project tree

Requirement

- The module is located in the first hierarchical level behind the head module (IM or CPU of the module family S7-1500 or ET200SP).
- The head module is connected to the engineering system via PROFINET.

Procedure

Follow these steps:

1. Select the relevant head module in the project tree.
2. Open the online and diagnostics view of the head module using the shortcut menu.
3. Select the "Firmware update" group in the "Functions" folder.
4. Select "Local modules".
   In the "Online data" area, a table lists modules of the first hierarchy level behind the head module. With Siemens modules, these are the modules that support a firmware update. With GSDML-based devices, this is all modules.
5. Select the module whose firmware you want to update.
6. Click the "Browse" button in the "Firmware loader" area in order to select the path to the firmware update files.
7. Select one of these files. The table then lists all modules for which an update is possible with the selected firmware file.

8. Optional: Select the "Run firmware after update" check box to reset the module after the load operation and to start the new firmware.
9. Click the "Start update" button. If the selected file can be interpreted by the module, it is downloaded to the module. If the operating mode of the CPU needs to be changed for this purpose, you will be prompted to do this by means of dialogs.

Starting the firmware update from the project context

Procedure

To perform a firmware update, follow these steps:

1. Open the module in the Online and Diagnostics view.
2. Select the "Firmware update" group in the "Functions" folder.

   Note

   For S7-1500-CPUs, this group is subdivided into "PLC" and "Display".

3. Click the "Browse" button in the "Firmware loader" area in order to select the path to the firmware update files.
4. Select one of these files. The table then lists all modules for which an update is possible with the selected firmware file.
5. Optional: Select the "Run firmware after update" check box to reset the module after the load operation and to start the new firmware.
6. Click the "Start update" button. If the selected file can be interpreted by the module, it is downloaded to the module. If the operating mode of the CPU needs to be changed for this purpose, you will be prompted to do this by means of dialogs.

Assigning an IP address to a PROFINET IO device

Basic information on assigning an IP address to a PROFINET IO device

Overview

All PROFINET IO devices work with the TCP/IP protocol and therefore require an IP address for operation on Industrial Ethernet. Once an IP address has been assigned to an IO device, it can be accessed via this address. You can then download configuration data or perform diagnostics, for example.

Requirement

- The Ethernet LAN connection must already be established.
- The Ethernet interface of your programming device or PC must be accessible.
- The IO device that is to be assigned an IP address must be in the same IP band as the programming device or PC.

Starting the address assignment via "Accessible devices"

Requirement

- The devices accessible via the associated interface of the PG/PC are displayed in the project tree (to display these, either double-click "Update accessible devices" in the project tree or select the "Accessible devices..." command in the "Online" menu.).
- You have double-clicked "Online access" -> <Selected interface> -> <PROFINET IO device> -> "Online & Diagnostics" in the project tree to open the Online and Diagnostics view.

Procedure

1. Open the "Functions" folder and the "Assign IP address" group inside this folder. The "MAC address" field displays the MAC address of the PROFINET IO device. The "Accessible devices" button is grayed out.
2. Enter the desired IP address.
3. Enter the subnet mask.
4. If a router is to be used, select the "Use router" check box and enter its IP address.
5. Click the "Assign IP address" button.

Result

The IP address is permanently assigned to the IO device or to the relevant PROFINET interface of the IO device. It is retained even through a startup or a power failure.

Note

For an S7-1500 CPU, you can also use the above-described method to change the IP address of a PROFINET interface, if a project has already been downloaded to the CPU via this interface. This overwrites the IP address downloaded via the project.

Starting the address assignment from the project context

Requirement

- An online connection to the PROFINET IO device exists.
- You have opened the Online and Diagnostics view of the PROFINET IO device from the project context.
- The PROFINET IO device is not assigned to any IO controller.

Procedure

1. Open the "Functions" folder and the "Assign IP address" group inside this folder.
2. Click the "Accessible devices" button in order to identify the devices that can be accessed. Note: For an S7-1500 CPU, there are two entries here because it has two PROFINET interfaces.
3. Select the IO device. The "IP address" field, "Subnet mask" field, "Use router" check box and "Router address" field are grayed out and contain the node properties you used to establish the current online access.
4. Click the "Assign IP address" button.

Result

The IP address is permanently assigned to the IO device or to the relevant PROFINET interface of the IO device. It is retained even through a startup or a power failure.

Assigning a PROFINET device name

Basic information on assigning a name to a PROFINET IO device

Device name

Before an IO device can be addressed by an IO controller, it must have a device name. This procedure was chosen for PROFINET because names are easier to handle than complex IP addresses.

Assigning a device name to a PROFINET IO device is comparable to setting the PROFIBUS address for a DP slave.

An IO device has no device name in its delivery state. For an IO controller to address an IO device, it must first be assigned a device name using the programming device or PC. It is now ready to transfer the configuration information including the IP address during startup or exchange user data in cyclic operation.

Rules for the device name

The device name is subject to the following limitations:

- Restricted to a total of 240 characters (lower case letters, numbers, dash, or dot)
- A name component within the device name, which is a character string between two dots, must not exceed 63 characters.
- No special characters such as umlauts, brackets, underscore, slash, blank space, etc. The only special character permitted is the dash.
- The device name must not begin or end with the "-" character.
- The device name must not begin with a number.
- The device name form n.n.n.n (n = 0, ... 999) is not permitted.
- The device name must not begin with the string "port-xyz" or "port-xyz-abcde" (a, b, c, d, e, x, y, z = 0, ... 9).

Where do I find the function I am seeking?

You can assign the name of a PROFINET IO device at the following places:

- In the online and diagnostics view of the device in the "Assign name" group of the "Functions" folder. The user interface for this group differs depending on how you open the online and diagnostics view:
  - Open via "Accessible devices"
  - Open from the project context
- In the "Assign PROFINET device name" dialog

See also

Assigning a name in the online and diagnostics view opened via "Accessible devices" (Page 2062)

Assigning a name in the online and diagnostics view opened from the project context (Page 2063)

Assigning a name in the "Assign PROFINET device name" dialog (Page 2064)

Assigning a name in the online and diagnostics view opened via "Accessible devices"

Requirement

- You have opened the online and diagnostics view of the PROFINET IO device using "Update accessible devices" (in the project tree) or "Accessible devices..." ("Online" menu).

Procedure

1. Open the "Functions" folder and the "Assign name" group inside this folder. The "Type" field displays the module type of the PROFINET IO device.
2. Enter the required device name in the "PROFINET device name" input box.
3. Optional: Select the "Flash LED" check box in order to run an LED flash test on the PROFINET IO device. In this way, you verify that you are naming the desired IO device.

   Note

   The LED flash test is not supported by all PROFINET IO devices.

   The LED flash test runs until you cancel it. This is done, for example, by clearing the "Flash LED" check box or by closing the online and diagnostics view.
4. Click "Assign name".

Result

The entered name is assigned to the PROFINET IO device.

Assigning a name in the online and diagnostics view opened from the project context

Requirement

- You have opened the online and diagnostics view of the PROFINET IO device from the project context.
- The PROFINET IO device can be accessed using at least one PG/PC interface.

Procedure

1. Open the "Functions" folder and the "Assign name" group inside this folder. The "PROFINET device name" drop-down list displays the current name in the offline project, and the "Type" box shows the module type of the PROFINET IO device.

   Note

   For CPUs with several PROFINET interfaces, the names of all existing PROFINET interfaces are displayed in the offline project.

2. Choose a different name from the drop-down list, if necessary.

   Note

   In steps 3 to 5, you determine the IO devices that are present in the PROFINET subnet.

3. In the "PG/PC interface for assignment" drop-down list, select the PG/PC interface you want to use to establish the online connection.
4. Optional: Use the three check boxes to make a selection from all IO devices available online.

5. Click the icon for determining the IO devices present in the PROFINET subnet. The table is then updated.
6. Select the desired IO device in the table.
7. Optional: Select the "Flash LED" check box in order to run an LED flash test on the PROFINET IO device. In this way, you verify that you are naming the desired IO device.

   Note

   The LED flash test is not supported by all PROFINET IO devices.

   The LED flash test runs until you cancel it. This is done, for example, by clearing the "Flash LED" check box, by selecting another IO device in the table, or by closing the online and diagnostics view.
8. Click "Assign name".

Result

The selected name is assigned to the PROFINET IO device or one of its PROFINET interfaces.

Assigning a name in the "Assign PROFINET device name" dialog

Requirement

- You have opened the "Assign PROFINET device name" dialog from the network view (from the shortcut menu of a PN/IE connection).
- The PROFINET IO device can be accessed using at least one PG/PC interface.

Procedure

1. The following is displayed in the "PROFINET device name" drop-down list:
   – the name of the interface that was used to open the project in the current offline project
   – the names of those IO devices that are connected by means of this interface

   The "Type" field displays the module type of the PROFINET IO device.
   Choose a different name from the drop-down list, if necessary.

   Note

   In steps 2 to 4, you determine the IO devices that are present in the PROFINET subnet.
2. In the "PG/PC interface for assignment" drop-down list, select the PG/PC interface you want to use to establish the online connection.
3. Optional: Use the three check boxes to make a selection from all IO devices available online.
4. Click the icon for determining the IO devices present in the PROFINET subnet. The table is then updated.
5. Select the desired IO device in the table.

6. Optional: Select the "Flash LED" check box in order to run an LED flash test on the PROFINET IO device. In this way, you verify that you are naming the desired IO device.

   Note

   The LED flash test is not supported by all PROFINET IO devices.

   The LED flash test runs until you cancel it. This is done, for example, by clearing the "Flash LED" check box or by selecting another IO device in the table.

7. Click "Assign name".

Result

The selected name is assigned to the PROFINET IO device or the interface that was used to open the dialog.

Calibrating an S7-1500 analog module (S7-1500)

Calibrating an S7-1500 analog module - Overview (S7-1500)

Where do you calibrate an S7-1500 analog module?

You calibrate an S7-1500 analog module in its Online and Diagnostics view in the "Calibrate" group of the "Functions" folder.

Overview of the function scope of the calibrating function

You can perform the following functions for an S7-1500 analog module in the "Calibrate" group:

- Specifying the current calibration of all channels
- Calibrating a channel
- Canceling a running calibration
- Resetting the calibration of a channel to the factory settings

Requirement for the calibrating function described below

The following is required for the calibrating function described below:

- You have opened the Online and Diagnostics view from the project context (thus not from the project tree or via the "Online" menu).
- There is an online connection to the analog module that is to be calibrated.
- Offline and online configuration are identical.

Calibrating an S7-1500 analog module (S7-1500)

Overview of the calibration of a channel of an S7-1500 analog module

The calibration of a channel of an S7-1500 analog module consists of the following steps:

1. Start calibration
2. Perform the second step up to the next to last step of the calibration
3. Complete calibration

These steps are described in more detail in the following section.

Requirement

- You have opened the Online and Diagnostic view of the S7-1500 analog module from the project context and are in the "Calibrate" group of the "Functions" folder.
- The associated CPU is online.
- No calibration is currently running on the analog module (if you want to start the calibration) or the last step initiated has been performed successfully (if you want to resume or complete the calibration).

Procedure for starting the calibration

To start the calibration, follow these steps:

1. In the overview table, select the line that belongs to the channel to be calibrated.
2. Click the "Start manual calibration" button.

The user interface then changes as follows:

- The overview table and the "Start manual calibration" button and "Set to factory settings" buttons become inactive.
- The step display is activated and displays the numbers of the current and last steps.
- The "Command" field becomes active and indicates what the user must do in the next calibration step.
- The "Status" field becomes active and shows the current status of the calibration, e.g., "Calibration successfully started".
- The "Measured value" field becomes active. For an input module a value is displayed here; you must enter a value here for an output module.
- The "Cancel" button becomes active.
- With certain analog modules (AI Energy Meter) the "Skip" button becomes active. This button can be used to skip to the next step of the calibration.
- The "Next" button becomes active. This button can be used to advance to the next step of the calibration.

Procedure for the second to the next to last step of the calibration

Follow these steps:

1. Click the "Next" button.

The fields of the user interface described above are then updated.

With certain analog modules (AI Energy Meter), you can skip the next calibration step as long as the next step is not the last step of the calibration procedure. To do this, follow these steps:

1. Click the "Skip" button.

The fields of the user interface described above are then updated.

Procedure for the last step of the calibration

Follow these steps:

1. Click the "Next" button.

The user interface then changes as follows:

- The overview table becomes active.
- The calibration display of the calibrated channel is updated.
- The "Start manual calibration" button and "Set to factory settings" buttons become active.
- The step display is deactivated and the numbers of the current step and last steps are empty.
- The "Command" field becomes inactive and is empty.
- The "Status" field becomes inactive and shows the last status of the calibration, e.g., "Calibration successfully finished".
- The "Measured value" field becomes inactive and is empty.
- The "Cancel" button becomes inactive.
- The "Skip" button becomes inactive.
- The "Next" button becomes inactive.

Error occurrence

If an error occurs during the calibration, the module cancels the calibration. Afterwards, the channel that was to be calibrated has the same settings as before the start of the calibration.

Except for the "Status" field, the user interface appears the same after the occurrence of an error as before the calibration. The "Status" field displays the error that the module detected during the calibration.

Canceling a running calibration of an S7-1500 analog module (S7-1500)

Requirement

- You have opened the Online and Diagnostic view of the S7-1500 analog module from the project context and are in the "Calibrate" group of the "Functions" folder.
- The associated CPU is online.
- A calibration is currently running on the analog module.

Procedure

To cancel a running calibration, follow these steps:

1. Click the "Cancel" button.

Result

The running calibration is canceled, and afterwards the channel to be calibrated has the same settings as before the calibration.

All operator controls in the user interface are deactivated until the cancelation is complete. Except for the "Status" field, the user interface appears the same afterwards as before the calibration. The "Status" field displays the result of the cancelation.

Resetting an S7-1500 analog module to factory settings (S7-1500)

Requirement

- You have opened the Online and Diagnostic view of the S7-1500 analog module from the project context and are in the "Calibrate" group of the "Functions" folder.
- The associated CPU is online.

Procedure

To reset a channel of an S7-1500 analog module to factory settings, follow these steps:

1. Select the line associated with the channel to be reset in the overview table.
2. Click the "Set to factory settings" button.

Result

All operator controls in the user interface are deactivated until the reset operation is complete. Except for the "Status" field, the user interface appears the same afterwards as before the reset operation. The "Status" field displays the result of the reset operation.

Loading I&M data to PROFINET IO devices and your modules

Which I&M data can be loaded to PROFINET IO devices and your modules?

You can load I&M 1 data (plant designation and location identifier) and/or I&M 2 data (installation date) and/or I&M 3 data (additional information) to the actual hardware.

Requirements

- In the project settings (Options > Settings, Hardware configuration > Compiling and downloading), the option "Download I&M data" must be enabled.
- There is an online connection to the PROFINET IO devices and the modules to which you want to load I&M data.
- You have entered the I&M data you want to download in the properties of the respective PROFINET IO devices and your modules (Inspector window: "Properties" tab > "General" tab, Settings > Identification & Maintenance).

Where do I specify which I&M data is downloaded to which PROFINET IO devices?

You specify which I&M data you want to download to which PROFINET IO devices in the "Load preview" dialog. You will find the following alternatives in the drop-down list of the "Identification and maintenance data (I&M)" row:

- Load nothing
  The check boxes for all PROFINET IO devices as well as the check boxes for the loadable I&M data are cleared.
  No I&M data is transferred to the actual hardware during loading with this setting.
- Load data
  The check boxes for all PROFINET IO devices as well as the check boxes for the loadable I&M data are selected.
  The respective I&M 1, I&M 2 and I&M 3 data is transferred to all PROFINET IO devices during loading with this setting.
- Load selected
  You select the check boxes of those PROFINET IO devices to which you want to load I&M data. You also select the check boxes of the identification data you want to load.
  With this setting, you transfer the selected I&M data to the selected PROFINET IO devices during loading.

Note

Language dependency of the I&M data to be loaded

The I&M data are loaded to the real hardware in the form that you specified in the properties of the relevant PROFINET IO devices and your modules. There is no language dependency.

10.2.1.6 Diagnostics in STOP mode

Basic information on the diagnostics buffer

Function

The operating system of the CPU enters the errors detected by the CPU and the diagnostics-capable modules into the diagnostics buffer in the order in which they occurred. This includes the following events:

- Every mode change of the CPU (POWER UP, change to STOP mode, change to RUN mode)
- Every hardware and diagnostic error interrupt

The top entry contains the most recent event. The entries in the diagnostics buffer are stored permanently. They are retained even if the power supply fails and can only be deleted by resetting the CPU to factory settings.

A diagnostics buffer entry contains the following elements:

- Time stamp
- Error ID
- Additional information specific to the error ID

Advantages of the diagnostics buffer

The diagnostics buffer offers the following advantages:

- After the CPU has changed to STOP mode, you can evaluate the last events prior to the STOP so that you can locate and identify the cause of the STOP.
- You can detect and eliminate the causes of errors more quickly and thus increase the availability of the system.
- You can evaluate and optimize the dynamic system response.

Organization of the diagnostics buffer

The diagnostics buffer is a ring buffer. With the S7-1200 CPUs, the maximum number of entries is 50. When the diagnostics buffer is full and a further entry needs to be made, all existing entries are shifted one position (which means that the oldest entry is deleted) and the new entry is made at the first position that is now free (FIFO principle: first in, first out).

Evaluation of the diagnostics buffer

The contents of the diagnostics buffer can be accessed as follows:

- Using the Online and Diagnostics view

The evaluation of events occurring prior to the error event (e.g., transition to STOP mode) allows you to obtain a picture of the possible causes or to zero in more closely or specify in more detail the possible causes (depending on the error type).

Read the detailed information about the events carefully and use the "Help on event" text box to obtain additional information and possible causes of individual entries.

Note

To make the best use of the time stamp information on the diagnostics buffer entries in time-critical systems, it is advisable to check and correct the date and time of day on the CPU occasionally.

Alternatively, it is possible to perform a time-of-day synchronization using an NTP time server.

See also

Resetting an S7-1200 CPU to factory settings (Page 2053)

Determining the cause of a STOP of a CPU (Page 2071)

Determining and setting the time of day on a CPU (Page 2052)

Determining the cause of a STOP of a CPU

Requirement

The CPU you want to analyze is in STOP mode.

Procedure

To find out the reason why a CPU changed to STOP, follow these steps:

1. Open the online and diagnostics view of the CPU.
2. Select the "Diagnostics buffer" group from the "Diagnostics" folder.
3. Evaluate the events occurring prior to the transition to STOP mode. Use this to obtain a picture of the possible causes or to zero in on or specify in more detail the possible causes (depending on the error type).
   Read the detailed information about the events carefully and use the "Help on event" button to obtain additional information and possible causes of individual entries.

Result

You were able to zero in on or determine in more detail the cause of the CPU STOP.

Note

If the analysis does not enable you to overcome the problem, contact Customer Support. In this case, use the "Save as" button to back up the content of the diagnostics data to a text file and submit it to Customer Support.

See also

Reading out the diagnostics buffer of a CPU (Page 2046)

10.2.1.7 Online accesses in the Online and Diagnostics view

Displaying status of the online connection

Requirement

- The associated device can be accessed using at least one PG/PC interface.

Procedure

1. Open the online and diagnostics view for the device whose online connection status you want to display.
2. Select the "Online access" group.

Note

The "Online access" group exists only for CPUs and some CPs. However, if you have opened the online and diagnostics view using the "Show/update accessible devices" function, it is not displayed.

Result

The status of the online connection is displayed in the "Status" area both graphically and in text form.

Specifying a PG/PC interface, going online

Requirement

- The associated device can be accessed using at least one PG/PC interface.
- There is currently no online connection to the relevant device.

Procedure

1. Open the Online and Diagnostics view of the device to which you want to establish an online connection.
2. Choose the "Online access" group and the "Online access" area within this group.

   Note

   The "Online access" group exists only for CPUs and some CPs. If you have opened the Online and Diagnostics view using the "Show/update accessible devices" function, it is not displayed.

3. If an online connection was established previously for the device, the associated data for this online connection is preset in the drop-down lists. In this case, you can immediately continue with the last step of this operating instruction, provided you have not changed the IP address in the meantime using the Online and Diagnostics view.
4. Choose the interface type in the "Type of PG/PC interface" drop-down list.
   The "PG/PC interface for online access" drop-down list then shows only the interfaces of the programming device or PC that match the selected interface type.
5. In the "PG/PC interface for online access" drop-down list, select the programming device or PC interface via which you want to establish the online connection.
6. Optional: Click the "Properties" button to change the properties of the associated CP.
7. In the "Connection to subnet" drop-down list, select the subnet via which the device is connected to the PG/PC interface.

   Note

   The PG/PC interface is connected to an interface of a device.

   If you only want to access this device, select the setting "Directly at slot <interface name>" in the drop-down list.

   If, however, you want to access another device by means of routing, create a subnet at this interface in the device configuration and then select this subnet in the drop-down list.

8. If the device is accessible via a gateway, select the gateway that connects the two subnets involved in the "1st gateway" drop-down list.
9. In the "Device address" entry field, enter the IP address of the device to which you want to establish an online connection, if necessary.

   Note

   For CPUs with multiple IP addresses, select the IP address of the PROFINET interface you want to use to establish an online connection from the "Device address" drop-down list.

10. Alternatively: Click the "Show accessible devices" button and choose the device from the list of accessible devices to which you want to establish an online connection.
11. Click the "Go online" button.

Result

The online connection to the desired device is established.

Going offline

Requirement

- There is currently an online connection to the relevant device.

Procedure

1. Open the online and diagnostics view of the device for which you want to disconnect the online connection.
2. Choose the "Online access" group and the "Online access" area within this group.

   Note

   The "Online access" group exists only for CPUs and some CPs. However, if you have opened the online and diagnostics view using the "Show/update accessible devices" function, it is not displayed.

3. Click the "Go offline" button.

Result

The online connection to the desired device will be disconnected.

Performing the flash test for a device with an online connection

Requirement

- There is currently an online connection to the relevant device.
- The FORCE function is not active.

Procedure

1. Open the online and diagnostics view of the device for which you want to perform a flash test.
2. Choose the "Online access" group and the "Status" area within this group.

   Note

   The "Online access" group exists only for CPUs and some CPs. However, if you have opened the online and diagnostics view using the "Show/update accessible devices" function, it is not displayed.

3. Select the "LED flash test" check box.

Result

- On an S7-1200 CPU, the RUN/STOP, ERROR and MAINT LEDs flash.
- On an S7-1500 CPU, the RUN/STOP, ERROR, and MAINT LEDs flash.
- On an S7-300 or S7-400 CPU, the FRCE LED flashes.

The LEDs flash until you cancel the flash test. This is done, for example, by clearing the "LED flash test" check box, by changing to another group of the online and diagnostics view, or by changing settings in the "Online access" area.

10.2.1.8 Checking PROFIBUS DP subnets for faults

Basic information on the diagnostic repeater

What is the diagnostic repeater?

The diagnostic repeater is a repeater that can monitor a segment of an RS 485 PROFIBUS subnet (copper cable) during operation and signal line errors to the DP master via a diagnostics message frame.

The diagnostic repeater detects, localizes and visualizes line errors during operation at an early stage. As a result, problems in the system are identified early and production downtimes will be minimized.

Function of the diagnostic repeater

The diagnostic repeater can perform line diagnostics on the DP2 and DP3 segments because it has a measuring circuit for these segments.

The line diagnostics run in two steps:

- Step 1: Topology determination
  You start the topology determination by calling the "DP_TOPOL" instruction in your program. The diagnostic repeater then determines the PROFIBUS addresses and the distance of the devices and creates a topology table.
- Step 2: Error localization
  The diagnostic repeater checks the lines during operation. It determines the distance to the point of the error and the reason for the error; it then issues a diagnostics alarm with relative information on the error location.

Display of detailed information on the determined error location

You receive detailed information on the determined error location in the Online and Diagnostics view of the diagnostic repeater.

- By means of icons
- By means of a display with graphics and text

See also

Displaying the status of the segment diagnostics using icons (Page 2076)

Displaying the status of the segment diagnostics using graphics and text (Page 2076)

Displaying the status of the segment diagnostics using icons

Where do I find the information I need?

The icons for the status of the segment diagnostics are available:

- In the expanded "Segment diagnostics" folder in the navigation pane of the Online and Diagnostics view of the relevant diagnostic repeater.

The diagnostics icon associated with the segment will be displayed behind the segment designation. It must be noted here that line errors will be displayed for the DP2 and DP3 segments only. The DP1 and programming device segments do not display errors in the form of a diagnostics icon; rather, they signal only a few bus errors.

Diagnostics icons

The following table shows the available icons and their meaning.

Icon	Meaning
	Segment is error-free
	Segment contains errors
	Segment is deactivated

Displaying the status of the segment diagnostics using graphics and text

Where is the status of the segment diagnostics displayed with graphics and text?

The status of the segment diagnostics will be displayed using graphics and text in the "DP1", "DP2", "DP3", and "PG" groups of the "Segment diagnostics" folder in the Online and Diagnostics view of the relevant diagnostic repeater.

Structure of the "DP1", "DP2" "DP3", and "PG" groups

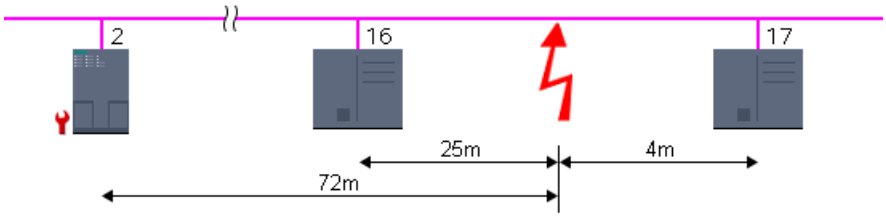
The "DP1", "DP2", "DP3", and "PG" groups consist of the following elements:

- "Error location" field
- "Error" field
- "Resolution" field
- "Help on event" button
- "Freeze display" or "Cancel freeze" button

"Error location" field

This field displays the error location graphically, provided the diagnostic repeater can determine the location.

The following picture shows an example for a line error occurring in the DP2 segment.



In this example, the diagnostic repeater has the PROFIBUS address 2, and a line error has occurred between the devices with PROFIBUS addresses 16 and 17. This line error is located 25 m from device 16, 4 m from device 17, and 72 m from the diagnostic repeater.

"Error" field

The error is explained in plain text in this field.

"Resolution" field

Here, you will find actions for resolving the error.

"Help on event" button

Click this button to obtain a more detailed explanation of the error and additional details on resolving the error, if applicable.

"Freeze display" or "Cancel freeze" button

The "Freeze display" or "Cancel freeze" button is only enabled when there is an online connection to the diagnostic repeater.

The default setting is "Freeze display".

The following happens when you click the "Freeze display" button:

- The current display of the segment diagnostics is frozen.
- The labeling of the button changes to "Cancel freeze".

If the display is frozen and you click the "Cancel freeze" button, the following happens:

- The display of the segment diagnostics is updated again.
- The labeling of the button changes to "Freeze display".

10.2.2 Connection diagnostics

10.2.2.1 Overview of connection diagnostics

Basics

Connection diagnostics, as described below, refers to the diagnostics of communication connections.

The connection diagnostics is started each time an online connection is established to a module (CPU or CP) that participates in one or more communication services. The connection status is updated automatically in the background.

In the case of one-way connections, an online connection must exist to the communication partner that has established the communication connection.

On connections configured at both ends, a distinction between the following two situations must be made:

- If there is an online connection to only one connection endpoint, only the part of the connection belonging to this connection endpoint can be diagnosed.
- If there is an online connection to both connection endpoints, both parts of the connection (and therefore the entire connection) can be diagnosed.

Basic connections diagnostics options

Connection diagnostics can be performed as follows:

- Using icons on the connection status display
  This display is generated in the connection table.
- Through detailed connection diagnostics
  This step is available in the "Diagnostics > Connection information" area of the Inspector window.

Requirement for the connection diagnostics described below

The connection diagnostics is only possible if the devices participating in the connection make the corresponding diagnostics data available.

You can display the details of either all the communication connections created in the project (default) or selected communication connections in the connection table.

The connection diagnostics described in the following assume that you display the details of selected communication connections. To do this, clear the "Show all connections" option in the shortcut menu.

10.2.2.2 Displaying the connection status using icons

Content of connection table without an online connection

- For a CPU or CP, the connection table lists the communication connections (including properties) configured offline, if an online connection is not established.

Content of connection table with an online connection

After the online connection has been established, the properties of the communication connections listed offline will be expanded to include diagnostics icons for the connection status ("Online status" column).

In addition, entries for all communication connections that exist online only (e.g., connections for the instructions for Open User Communication, programming device and OP connections, connections for web server access) will now be added to the connection table.

For connections that exist online or offline only, the diagnostics icon at the bottom right is combined with a smaller additional comparison status icon.

Diagnostics icons for communication connections

The following table shows the diagnostics icons for communication connections.

Icon	Meaning
	Connection setup
	Connection not setup / is being setup
	Connection not available

Diagnostics icons for the comparison status

The diagnostic icons for communication connections can be combined at the bottom right with additional smaller icons that indicate the result of the online/offline comparison. The following table shows the available comparison icons and their meaning.

Icon	Meaning
	Connection exists online only
	Connection exists offline only

10.2.2.3 Detailed connection diagnostics

Detailed connection diagnostics - overview

Where do I perform detailed connection diagnostics?

To perform detailed connection diagnostics, go to the "Diagnostics > Connection" information of the Inspector window.

How do I open the "Diagnostics > Connection information" area of the Inspector window?

The following options are available for opening the "Connection information" tab of the Inspector window.

- Select the line of the relevant connection in the connection table. Click the "Diagnostics" and "Connection information" tabs one after the other in the Inspector window.
- Double-click the diagnostics icon of the relevant connection in the connection table.
- This step takes you to the programming editor for a S7 communication instruction or open user communication instruction. Double-click the diagnostic icon of the instruction (stethoscope).

Structure or the "Diagnostics > Connection information" area of the Inspector window

Requirements: the content of the "Connection information" tab has been filled, and an online connection to at least one end point of the relevant connection has been established.

If a module has been selected (network view), the tab will contain the following group:

- Connection resources (for S7-1200 and S7-1500)

If a connection has been selected (connection table), it will contain the following groups:

- Connection details
- Address details of the connection (for S7-1200 and S7-1500)

Determining online connection resources for S7-1200 (S7-1200)

Where do you determine the online connection resources?

The online connection resources are obtained in the "Connection resources" group. This group is located in the "Diagnostics > Connection information" area of the Inspector window. It is displayed only if you have selected a module in the network view to which an online connection exists.

Number of connection resources

- Maximum number: Specifies the maximum number of available connection resources of the module.
- Not assigned: Indicates how many connection resources are not yet assigned. If connection resources are already reserved for certain types of communication, then the unreserved connection resources cannot always be used for the various connection types.

Reserved and currently assigned connection resources

For the communication types indicated below, the connection resources that are reserved and currently assigned by the module will be displayed.

Communication type	Meaning
PG communication	Resources for connections between the module and programming devices (for example, for the establishment of a connection from the project tree, for online diagnostics, etc.)
HMI communication	Resources for connections between the module and HMI devices
Open User Communication	Resources for connection of open user communication instructions
S7 communication	Resources for configured S7 connections, through which data can be exchanged by calling instructions in the user program.
Other communication	Specifies other assigned connection resources for which connection resources are not reserved.

Determining online connection resources for S7-1500 (S7-1500)

Where do you determine the online connection resources?

The online connection resources are obtained in the "Connection resources" group. This group is located in the "Diagnostics > Connection information" area of the Inspector window. It is displayed only if you have selected a module in the network view to which an online connection exists.

Description of the detailed display of the connection resources

The detailed display of the connection resources includes:

- Number of available connection resources
- Number of configured connection resources
- Number of connection resources still available

For a description of these, go to here .

Determining connection details

Where do I determine the connection details?

The connection details are obtained in the "Connection details" group. This group is located in the "Diagnostics > Connection information" area of the Inspector window.

When is the "Connection details" group filled in?

The following requirements must be met to fill in the "Connection details" group on the "Connection information" tab:

- An online connection to the end point of the relevant connection must exist.
- You have selected a line in the connection table.

Structure of the "Connection details" group

The "Connection details" group consists of the following elements:

- Local ID (hex)
- Connection type (for S7-1200 and S7-1500)
- Protocol
- Connection status: icon and description
- Details
- Last status change (for S7-300 and S7-400 only)

Determining the address details of a connection (S7-1200, S7-1500)

Where do I determine the address details of a connection?

The address details of a connection are obtained in the "Connection address details" group. This group is located in the "Diagnostics > Connection information" area of the Inspector window.

For which CPUs is the "Connection address details" group available?

The "Connection address details" group of the "Connection information" tab is available for S7-1200 and S7-1500 CPUs.

When is the "Connection address details" group filled in?

The following requirements must be met to fill in the "Connection address details" group on the "Connection information" tab:

- An online connection to the end points of the relevant connection must exist.
- You have selected a line in the connection table.

Structure of the "Connection address details" group

The address details relevant to the connection type are specified for the two communication partners.

Programming the PLC 11

11.1 Programming basics

11.1.1 Operating system and user program

11.1.1.1 Operating system

Function

The operating system is contained in every CPU and organizes all CPU functions and sequences that are not associated with a specific control task.

The tasks of the operating system, for example, include the following:

- Processing a warm restart
- Updating the process image of the inputs and outputs
- Calling the user program
- Detecting interrupts and calling interrupt OBs
- Detecting and handling errors
- Managing memory areas

The operating system is a component of the CPU and is already installed there upon delivery.

See also

User program (Page 2085)

11.1.1.2 User program

Function

The user program contains all functions that are necessary for processing your specific automation task.

The tasks of the user program include:

- Checking the requirements for a (warm) restart using startup OBs, for example, limit switch in correct position or safety relay active.
- Processing process data, e.g. linking binary signals, reading in and evaluating analog values, defining binary signals for output, and outputting analog values

- Reaction to interrupts, for example, diagnostic error interrupt if the limit value of an analog expansion module is overshot.
- Error handling in normal program execution

You write the user program and load it into the CPU.

See also

Operating system (Page 2085)

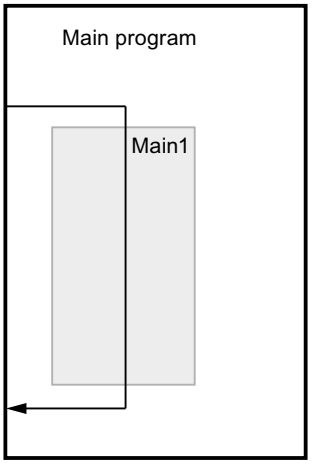
11.1.2 Blocks in the user program

11.1.2.1 Linear and structured programming

Linear programming

Solutions for small automation tasks can be programmed linearly in a program cycle OB. This is only recommended for simple programs.

The following figure shows a linear program schematically: The "Main1" program cycle OB contains the complete user program.



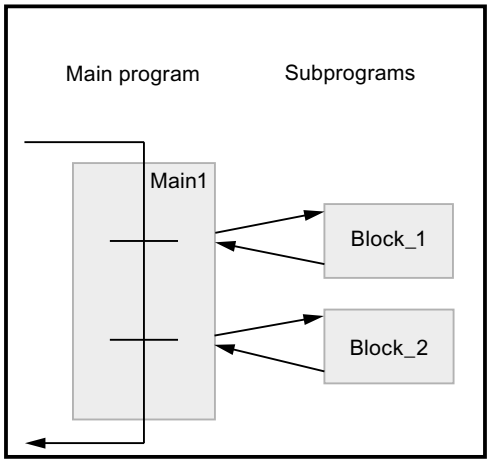
Structured programming

Complex automation tasks can be more easily handled and managed by dividing them into smaller sub-tasks that correspond to the technological functions of the process or that can be reused. These sub-tasks are represented in the user program by blocks. Each block is then an independent section of the user program.

Structuring the program offers the following advantages:

- Extensive programs are easier to program through the structure.
- Individual program sections can be standardized and used repeatedly with changing parameters.
- Program organization is simplified.
- Changes to the program can be made more easily.
- Debugging is simplified since separate sections can be tested.
- Commissioning is simplified.

The following figure shows a structured program schematically: The "Main1" program cycle OB calls subprograms one after the other that execute defined subtasks.



Nesting depth for blocks

The permissible nesting depth for blocks depends on the CPU that is used.

The following table shows the guide values for the maximum nesting depth. For detailed information on the CPU you are using, refer to the technical specifications in the hardware documentation. To call the hardware documentation on the Internet, click the links in the table.

CPU family	Nesting depth (guide value)	Link to hardware documentation
S7-1500	24 blocks per priority class	SIMATIC S7-1500 / ET 200MP Manual Collection (https://support.industry.siemens.com/cs/ww/en/view/86140384)
S7-1200	16 blocks from cycle or startup OB, 6 additional blocks within any interrupt event OB	SIMATIC S7-1200 Automation System (https://support.industry.siemens.com/cs/ww/en/view/91696622)
S7-400	24 blocks per priority class, 1-2 additional blocks within an error OB	SIMATIC S7-400 automation system S7-400 CPU data (https://support.industry.siemens.com/cs/de/en/view/53385241)

CPU family	Nesting depth (guide value)	Link to hardware documentation
S7-300	16 blocks per priority class, 4 additional blocks within an error OB	SIMATIC S7-300 CPU 31xC and CPU 31x: Technical specifications (https://support.industry.siemens.com/cs/de/de/view/12996906/en)
ET 200SP	24 blocks	SIMATIC ET 200SP Manual Collection (https://support.industry.siemens.com/cs/ww/en/view/84133942)

Nesting depth for structures

Structures (STRUCT) and PLC data types (UDT) can be nested to a depth of 8. This nesting depth is independent of the CPU used.

11.1.2.2 Overview of the block types

Block types

Different BLOCK types are available to perform tasks within an automation system. The following table shows the available block types:

Block type	Brief description
Organization blocks (Page 2088) (OB)	Organization blocks define the structure of the user program.
Functions (Page 2089) (FC)	Functions contain program routines for recurring tasks. They have no "memory".
Function blocks (Page 2090) (FB)	Function blocks are code blocks that store their values permanently in instance data blocks, so that they remain available even after the block has been executed.
Instance data blocks (Page 2100)	Instance data blocks are assigned to a function block when it is is called for the purpose of storing program data.
Global data blocks (Page 2091)	Global data blocks are data areas for storing data that can be used by any blocks.

11.1.2.3 Organization blocks (OB)

Definition

Organization blocks (OBs) form the interface between the operating system and the user program. They are called by the operating system and control, for example, the following operations:

- Startup characteristics of the automation system
- Cyclic program processing
- Interrupt-driven program execution
- Error handling

You can program the organization blocks and at the same time determine the behavior of the CPU. Various organization blocks are available to you depending on the CPU used.

For more information on organization blocks, refer to the descriptions of the modes of operation of CPUs in the "Additional information on configurations" chapter in "Configuring Hardware and Networks".

Start information of organization blocks

When certain organization blocks are started, the operating system provides information that can be evaluated in the user program. Refer to the descriptions of the organization blocks to find out which information is provided, if any.

See also

Creating organization blocks (Page 4715)

11.1.2.4 Functions (FCs)

Definition

Functions (FCs) are code blocks without memory. You have no data memory in which values of block parameters can be stored. Therefore, when a function is called, all formal parameters must be assigned actual parameters.
Functions can use global data blocks to store data permanently.

Application

A function contains a program that is executed when the function is called by another code block. Functions can be used, for example, for the following purposes:

- To return function values to the calling block, e.g. for mathematical functions
- To execute technological functions, e.g. individual controls using bit logic operations

A function can also be called several times at different points in a program. As a result, they simplify programming of frequently recurring functions.

Note

Parameter transfer when calling functions

To avoid errors when working with functions, observe the information in chapter "AUTOHOTSPOT".

See also

Creating functions and function blocks (Page 4716)

11.1.2.5 Function blocks (FB)

Definition

Function blocks are code blocks that store their input, output and in-out parameters permanently in instance data blocks, so that they remain available even after the block has been executed. Therefore they are also referred to as blocks "with memory".

Function blocks can also operate with temporary tags. Temporary tags are will not be stored in the instance DB, but are available for one cycle only.

Application

Function blocks contain subroutines that are always executed when a function block is called by another code block. A function block can also be called several times at different points in a program. As a result, they simplify programming of frequently recurring functions.

Instances of function blocks

A call of a function block is referred to as an instance. An instance data block is required for each instance of a function block; it contains instance-specific values for the formal parameters declared in the function block.

The function block can store its instance-specific data in its own instance data block or in the instance data block of the calling block.

Access modes

S7-1200 and S7-1500 offer two different access options for the instance data blocks, which can be assigned to a function block when this is called:

- Data blocks with optimized access
  Data blocks with optimized access have no firmly defined memory structure. The data elements contain only a symbolic name in the declaration, no fixed address within the block.
- Data blocks with standard access (compatible with S7-300/400)
  Data blocks with standard access have a fixed memory structure. The declaration elements contain both a symbolic name in the declaration and a fixed address within the block.

Note

To avoid errors when working with function blocks, refer to the section "AUTOHOTSPOT".

See also

Creating functions and function blocks (Page 4716)

Multi-instances (Page 2112)

Instance data blocks (Page 2100)

Basics of block access (Page 2102)

11.1.2.6 Global data blocks (DB)

Definition

Data blocks are used to store program data. Data blocks thus contain variable data that is used by the user program. Global data blocks store data that can be used by all other blocks.

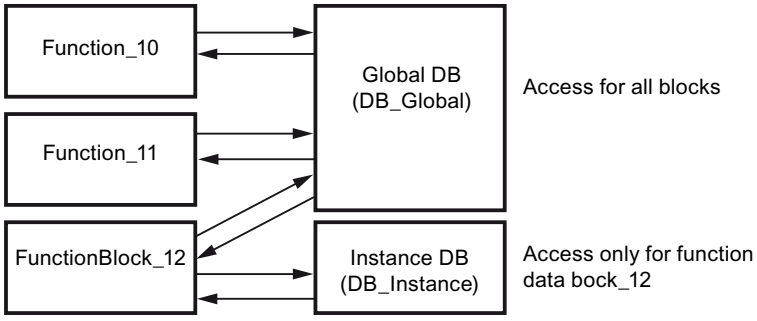
The maximum size of data blocks varies depending on the CPU. You can define the structure of global data blocks anyway you please.

You also have the option of using PLC data types (UDT) as a template for creating global data blocks.

Global data blocks in the user program

Every function block, function, or organization block can read the data from a global data block or can itself write data to a global data block. This data remains in the data block even after the data block is exited. A global data block and an instance data block can be open at the same time.

The following figure shows the different accesses to data blocks:



Access modes

S7-1200 and S7-1500 offer two different access options for global data blocks:

- Data blocks with optimized access
  Data blocks with optimized access have no fixed defined structure. In the declaration, the data elements are assigned only a symbolic name and no fixed address within the block.
- Data blocks with standard access (compatible with S7-300/400)
  Data blocks with standard access have a fixed structure. In the declaration, the data elements are assigned both a symbolic name and a fixed address within the block.

See also

Creating data blocks (Page 4798)

Basics of block access (Page 2102)

11.1.2.7 Global ARRAY data blocks (DB)

ARRAY data blocks (S7-1500)

ARRAY data blocks are a particular type of global data block. These consist of an ARRAY of any data type. For example, an ARRAY of a PLC data type (UDT) is possible. The DB contains no other elements besides the ARRAY. Because of their flat structure, ARRAY data blocks facilitate access to the ARRAY elements and their transfer to called blocks.

The "Optimized block access" attribute is always enabled for ARRAY data blocks. ARRAY data blocks with standard access are not possible.

The "Instructions > Basic instructions" task card in the "Move > ARRAY DB" section offers extended options for addressing ARRAY DBs. These instructions give you the option, for example, to address the DB name indirectly:

- ReadFromArrayDB: Read from array data block
- WriteToArrayDB: Write to ARRAY data block
- ReadFromArrayDBL: Read from ARRAY data block in load memory
- WriteToArrayDBL: Write to array data block in load memory

See also

Example of the use of ARRAY data blocks (Page 2092)

Addressing ARRAY DB (Page 2153)

Creating data blocks (Page 4798)

11.1.2.8 Example of the use of ARRAY data blocks

Using an ARRAY data block (S7-1500)

ARRAY data blocks are global data blocks that consist exclusively of an ARRAY. A data block with a tag of the ARRAY data type is sufficient for most applications because access can be programmed intuitively with a tag of the ARRAY data type (for example, #myArray[#index]) and offer a better runtime performance than ARRAY data blocks. In certain scenarios, however, it is necessary to process ARRAYs with different lengths. The ARRAY data block is particularly suitable for these scenarios.

The following example shows how you can use an ARRAY data block.

Programming example

Individual pieces of material are transported on a conveyor belt. These pieces of material pass a scanner which can read out the information that a piece of material carries with it. The information is read out and transmitted to a panel. Because the scanner and the panel have different clock cycles/speeds, the material information must be cached in each case.

In the following programming example, we show you how to program the program code for passing on material information. For this purpose, you use an ARRAY data block.

The example is set up to be flexible, so that you do not have to know at the time the program code is created which ARRAY data block will be read or written or how large it is. You can thus use ARRAYs of different length. Use of data type DB_ANY gives you this flexibility.

The data type VARIANT is used so that you also have flexibility when specifying the value that is to be read or written.

Not until runtime is the ARRAY data block transferred in order to then access the values in the program block. The data type of the ARRAY elements and the data type of the value that is to be read or written are determined.

You can determine the number of objects and the fill level of the ARRAY data block using special instructions.

Procedure

Create the PLC data type "UDT_Queue". This PLC data type is used by both functions ("FC_Enqueue" and "FC_Dequeue"). This is important, for example, for accessing the tag #Queue.Used, as the function "FC_Enqueue" increments the tag by one and the function "FC_Dequeue" decrements the tag by one.

1. Double-click the "Add new data type" command in the "PLC data types" folder in the project tree.
   A new declaration table for creating a PLC data type will be created and opened.
2. Declare the following lines within the PLC data type:
   DB > Data type: DB_ANY
   Size > Data type: DINT
   Used > Data type: DINT
   ReadPos > Data type: DINT
   WritePos > Data type: DINT

UDT_Queue

	Name	Data type	Default value	Comment
1	DB	DB_ANY	0	
2	Size	DInt	0	
3	Used	DInt	0	
4	ReadPos	DInt	0	Pointer Reading Position
5	WritePos	DInt	0	Pointer Writing Position

Program the "FC_Enqueue" function, which writes the values of the material information to an ARRAY data block. The specific ARRAY data block and the data type of the value do not have

to known at this time, because the interfaces are being programmed with the VARIANT and DB_ANY data types.

1. Create an SCL function and name it "FC_Enqueue".
2. Declare the block interface as follows:

FC_Enqueue

	Name	Data type	Comment
1	Input		
2	Value	Variant	
3	Output		
4	InOut		
5	Queue	"UDT_Queue"	
6	DB	DB_ANY	
7	Size	DInt	
8	Used	DInt	
9	ReadPos	DInt	Pointer Reading Position
10	WritePos	DInt	Pointer Writing Position
11	Temp		
12	Error	Int	
13	<Add new>		
14	Constant		
15	Return		
16	FC_Enqueue	Int	

3. Write the following program code:

```
IF #Queue.Used < #Queue.Size THEN // As long as there is enough space in the DB
                                  // the function keeps on writing.

    #Error := WriteToArrayDB(db := #Queue.DB, index := #Queue.WritePos, value := #Value);

    IF #Error = 0 THEN

        #Queue.Used := #Queue.Used + 1;
        #Queue.WritePos := #Queue.WritePos + 1;

        IF #Queue.WritePos >= #Queue.Size THEN // If this condition is fullfilled
                                               // the end of the DB is reached.
                                               // End of writing pointer.

            #Queue.WritePos := 0;

        END_IF;

    END_IF;

ELSE

    #Error := 4711; // The queue is full.

END_IF;
```

With this function, you check whether there is still space free in the data block. If so, write the value that is specified at the parameter value into the data block at the parameter db. With each new item of material information that is written, the tag #Queue.Used and the pointer tag #Queue.WritePos are incremented by one. As soon as the cursor reaches the end of the data block, it is reset to 0. If the data block is full, the error code #4711 is returned.

Program the "FC_Dequeue" function, which reads the material information from an ARRAY data block and writes it to the panel. The specific ARRAY data block and the data type of the value do not have to known at this time, because the interfaces are being programmed with

the VARIANT and DB_ANY data types. The material information can then be displayed on a panel, for example:

1. Create an SCL function and name it "FC_Dequeue".
2. Declare the block interface as follows:

FC_Dequeue

		Name	Data type	Comment
1		Input		
2		Output		
3		InOut		
4		Value	Variant	
5		Queue	"UDT_Queue"	
6		DB	DB_ANY	
7		Size	DInt	
8		Used	DInt	
9		ReadPos	DInt	Pointer Reading Position
10		WritePos	DInt	Pointer Writing Position
11		Temp		
12		Error	Int	
13		Constant		
14		Return		
15		FC_Dequeue	Int	

3. Write the following program code:

```
IF #Queue.Used > 0 THEN // As long as there is some information in the DB
                        // the function keeps on reading.

    #Error := ReadFromArrayDB(db := #Queue.DB, index := #Queue.ReadPos, value => #Value);

    IF #Error = 0 THEN

        #Queue.Used := #Queue.Used - 1;
        #Queue.ReadPos := #Queue.ReadPos + 1;

        IF #Queue.ReadPos >= #Queue.Size THEN // If this condition is fullfilled
                                              // the end of the DB is reached.
                                              // End of reading pointer.

            #Queue.ReadPos := 0;

        END_IF;

    END_IF;

ELSE

    #Error := 4712; // The queue is empty.

END_IF;
```

With this function, you check whether material information is available in the data block. If so, read the value to which the pointer #Queue.ReadPos points, and write it to the tag #Value. With each item of material information that is read, the tag #Queue.Used is decremented by one and the pointer tag #Queue.ReadPos is incremented by one. As soon as the cursor reaches the end of the data block, it is reset to 0. If the data block is empty, the error code #4712 is returned.

To save the material data, create an ARRAY data block. Use the PLC data type "UDT_Material" as the data type of the ARRAY data block.

First, create the PLC data type "UDT_Material". This PLC data type contains a structure for the material information supplied by the scanner.

1. Double-click the "Add new data type" command in the "PLC data types" folder in the project tree.
   A new declaration table for creating a PLC data type will be created and opened.
2. Declare the following lines within the PLC data type:
   ArticleNumber > Data type: DINT
   ArticleName > Data type: STRING
   Amount > Data type: REAL
   Unit > Data type: STRING

UDT_Material

	Name	Data type
1	ArticleNumber	DInt
2	ArticleName	String
3	Amount	Real
4	Unit	String

Create the ARRAY data block "DB_MaterialBuffer". The ARRAY data block is to contain data records with material information of data type "UDT_Material". The material information is written to the ARRAY data block using the "FC_Enqueue" function.

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Specify the name "DB_MaterialBuffer".
4. Select the type of the data block "ARRAY DB".
5. Select the PLC data type "UDT_Material" as ARRAY data type.
6. Specify "1000" as high ARRAY limit.
7. Click "OK".

DB_MaterialBuffer

	Name	Data type
1	DB_MaterialBuffer	Array[0..1000] of "UDT_Material"
2	DB_MaterialBuffer[0]	"UDT_Material"
3	ArticleNumber	DInt
4	ArticleName	String
5	Amount	Real
6	Unit	String
7	DB_MaterialBuffer[1]	"UDT_Material"
8	DB_MaterialBuffer[2]	"UDT_Material"
9	DB_MaterialBuffer[3]	"UDT_Material"
10	DB_MaterialBuffer[4]	"UDT_Material"
11	DB_MaterialBuffer[5]	"UDT_Material"

Create the startup organization block (OB) "OB_MaterialQueue". In this organization block, you initialize the tags DB and Size.

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Organization block (OB)" button.
3. Specify the name "OB_MaterialQueue".
4. Select the type "Startup".
5. Select SCL as the language of the organization block.
6. Click "OK".
7. Write the following program code:

```
1  "DB_MaterialQueue".DB := "DB_MaterialBuffer";
2  "DB_MaterialQueue".Size := 1000;
3  "DB_MaterialQueue".Used := 0;
4  "DB_MaterialQueue".ReadPos := 0;
5  "DB_MaterialQueue".WritePos := 0;
```

   By assigning the data block, you connect the ARRAY data block "DB_MaterialBuffer" with the SCL functions "FC_Enqueue" and "FC_Dequeue". Enter the size of the ARRAY data block at the Size parameter. The start value of the Used parameter is "0". The first item of material information is thus written to the ARRAY element "0".

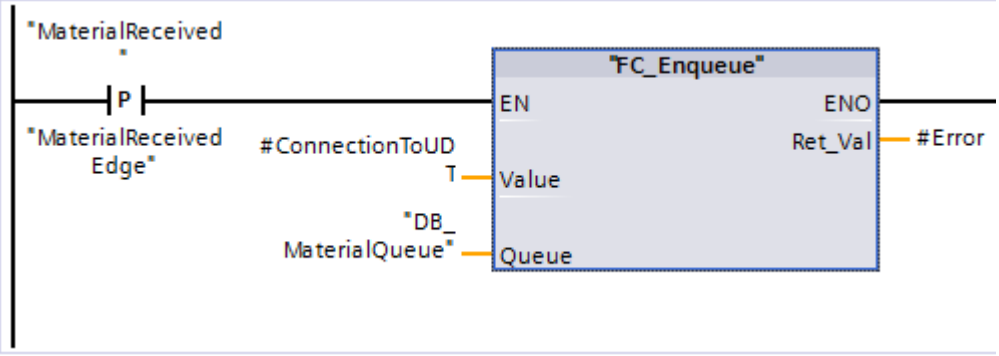
1. Declare the following tags in the "Default tag table":

	Name	Data type	Address
	MaterialReceived	Bool	%I0.0
	MaterialReceivedEdge	Bool	%M0.0
	MaterialDone	Bool	%I1.0
	MaterialDoneEdge	Bool	%M0.1

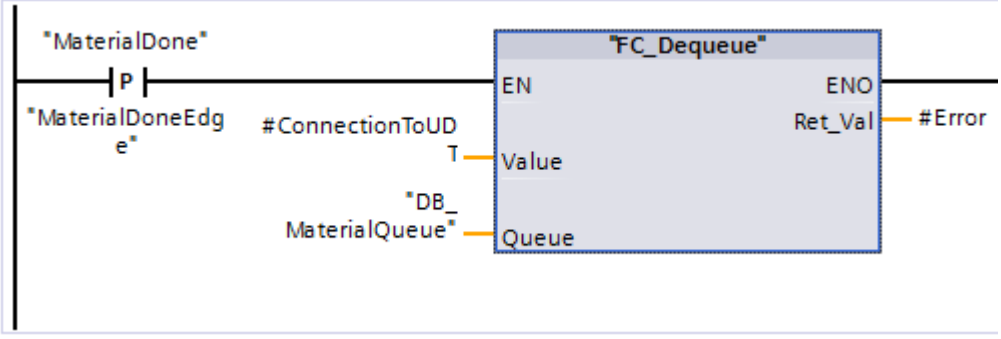
2. Call the SCL function "FC_Enqueue" within the function block in which the scanner reads out the material information.
3. Declare the tag "ConnectionToUDT" in the section "Temp" in the block interface and link this tag with the PLC data type "UDT_Material":

Temp	
ConnectionToUDT	"UDT_Material"
ArticleNumber	DInt
ArticleName	String
Amount	Real
Unit	String

4. Link the function call with the following tags and, at the enable input EN, create the signal edge "P: Scan operand for positive signal edge". Link the signal edge with the global tags from the standard tag table:



5. Call the SCL function "FC_Dequeue"
6. Link the function call with the following tags and, at the enable input EN, create the signal edge "P: Scan operand for positive signal edge". Link the signal edge with the global tags from the standard tag table:



Result

As soon as a positive signal edge occurs, material information is written to an ARRAY data block with the help of the instruction "WriteToArrayDB" and sent to the panel with the help of the instruction "ReadFromArrayDB".

See also

Global ARRAY data blocks (DB) (Page 2092)

Addressing ARRAY DB (Page 2153)

11.1.2.9 Instance data blocks

Definition

The call of a function block is referred to as an instance. The data with which the instance works is stored in an instance data block.

The maximum size of instance data blocks varies depending on the CPU. The tags declared in the function block determine the structure of the instance data block.

Access modes

S7-1200 and S7-1500 offer two different access options for the instance data blocks, which can be assigned to a function block when this is called:

- Data blocks with optimized access
  Data blocks with optimized access have no firmly defined structure. The declaration elements contain only one symbolic name in the declaration, and no fixed address within the block.
- Data blocks with standard access (compatible with S7-300/400)
  Data blocks with standard access have a fixed structure. The declaration elements contain both a symbolic name in the declaration and a fixed address within the block.

See also

Creating data blocks (Page 4716)

Basics of block access (Page 2102)

Insert block calls in LAD (Page 4899)

Inserting block calls in FBD (Page 4956)

11.1.2.10 CPU data blocks

Definition

CPU data blocks are generated by the CPU at runtime. To this purpose, insert the "CREATE_DB" instruction into your user program. You can use the data block that is generated at runtime to save your data.

CPU data blocks are indicated by means of a small CPU icon in the "Program blocks" folder of an available node. You can monitor the values of the variables of a CPU data block in online mode, similar to those of a different data block type.

You cannot create CPU data blocks in your offline project.

Loading CPU data blocks

The CPU data block that the user program has generated by means of the "CREATE_DB" instruction is initially only available on the device in online mode. All CPU data blocks will be included with the other blocks the next time you perform a complete download from the device to the project. The CPU data blocks are marked with a small CPU icon in the process. However, you cannot upload these CPU data blocks to your device again.

Restrictions on CPU data blocks in the project

Once the CPU data blocks have been loaded into your offline project, you can open and view their content. However, note that the CPU data blocks in the project are write-protected. The CPU data blocks in the project are therefore subject to the following restrictions:

- You cannot edit CPU data blocks, or convert these into a different data block type.
- CPU data blocks cannot be assigned a know-how protection.
- You cannot change the programming language of a CPU data block.
- CPU data blocks cannot be compiled or downloaded to a device.

Comparing CPU data blocks

Once the CPU data blocks have been loaded into your offline project, you can run an online/offline comparison for the CPU DBs loaded. The comparison editor provides you with a corresponding overview of the differences. It is possible to synchronize the online and off-line version of CPU data blocks if differences are found, but not by downloading the offline version to the device.

Deleting CPU data blocks

You can delete CPU data blocks both from the project and from the CPU.

See also

Deleting CPU data blocks (Page 4737)

11.1.2.11 Blocks with optimized access

Basics of block access

Introduction

STEP 7 offers data blocks with different access options:

- Data blocks with optimized access (S7-1200/S7-1500)
- Data blocks with standard access (S7-300 / S7-400 / S7-1200 / S7-1500)

Within one program you can combine the two types of blocks.

Data blocks with optimized access

Data blocks with optimized access have no fixed defined structure. In the declaration, the data elements are assigned only a symbolic name and no fixed address within the block. The elements are saved automatically in the available memory area of the block so that there are no gaps in the memory. This makes for optimal use of the memory capacity.

Tags are identified by their symbolic names in these data blocks. To address the tag, enter its symbolic name. For example, you access the "Fill Level" tag in the "Data" DB as follows:
`"Data".Fill Level`

Blocks with optimized access offers the following advantages:

- You can create data blocks with any structure without paying attention to the physical arrangement of the individual data elements.
- Quick access to the optimized data is always available because the data storage is optimized and managed by the system.
- Access errors, as with indirect addressing or from the HMI, for example, are not possible.
- You can define specific individual tags as retentive.
- Optimized blocks are equipped with a memory reserve by default which lets you expand the interfaces of function blocks or data blocks during operation. You can download the modified blocks without setting the CPU to STOP and without affecting the values of already loaded tags.

Note

The "Optimized block access" attribute is always enabled for the following blocks and cannot be deselected.

- GRAPH blocks
- ARRAY data blocks

Data blocks with standard access

Data blocks with standard access have a fixed structure. In the declaration, the data elements are assigned both a symbolic name and a fixed address within the block. The address is shown in the "Offset" column.

Tags in these data blocks can be addressed in both symbolic and absolute form.
`"Data".Fill Level`
`DB1.DBW2`

Setting Retentivity for Optimized Access or Standard Access

If you define data as retentive, its values are retained even after a power failure or a network off. A retentive tag is not initialized after the hot restart but retains the value it had prior to the power failure. If a DB tag is defined as retentive, it is stored in the retentive memory area of the data block.

The options for setting the retentivity depend on the access type of the block.

- In data blocks with standard access, you cannot set the retentive behavior of individual tags. The retentivity setting is valid for all tags of the data block.
- In data blocks with optimized access you can define the retentive behavior of individual tags.
  For structured data type tags, the retentivity setting always applies to the entire structure. You cannot make any individual retentivity setting for separate elements within the data type.

Setting Addressing Options for Optimized Access or Standard Access

Blocks with optimized access permit only "type-safe" access. Type-safe access addresses tags by their symbolic name only. This means even changes to the block or the block interface will not result in inconsistencies in the program or access errors.

The following table shows the permitted addressing options for optimized data:

Addressing	Block with standard access	Block with optimized access
Symbolic addressing	x	x
Indexed addressing of ARRAYs		x
Slice access	x	x
Overlapping with AT	x	-
Absolute addressing	x	-
Indirect addressing via ANY	x	-
Indirect addressing via POINTER and VARIANT	x	with symbolic notation only

See also

Setting up block access (Page 2104)

Creating blocks (Page 4714)

Setting up block access

Introduction

Block access is set up automatically when you create a block:

- Blocks created on CPUs of the S7-1200/1500 product range provide optimized access by means of a default setting.
- New blocks created on CPUs of the S7-300/S7-400 product range provide standard access by means of a default setting.

Access to a block that you copy or migrate to a CPU of a different product range is not converted automatically. However, in certain situations it may be useful to change block access in manual mode, e.g., in order to utilize the full functional scope of the CPU.

In most cases, you will have to recompile and load the program after block access has been converted.

NOTICE
Optimized block access for GRAPH blocks The "Optimized block access" attribute is always enabled for GRAPH blocks in S7-1500 and cannot be deselected.

Procedure

To set the block access, proceed as follows:

1. Open the "Program blocks" folder in the project tree.
2. Right-click on the block whose block access you want to change.
3. Select the "Properties" command in the shortcut menu.
   The properties dialog box of the block opens.
4. Click "Attributes" in the area navigation.
5. Enable or disable the "Optimized block access" option.
6. Confirm your entries with "OK".

Restrictions and special features

As a matter of principle, you can only convert block access on CPUs of the S7-1200/1500 product range, as only these support the "optimized" access mode.

The following restrictions or special features apply in this context:

- Instance data blocks
  The block access of instance data blocks is always determined by the assigned function block and cannot be changed in manual mode. If you change the access mode on a function block, you also need to update the assigned instance data blocks. This update procedure adapts the access mode of the instance data block.
- System blocks and know-how protected blocks
  You cannot manually edit the block access for system blocks and know-how protected blocks.
- Organization blocks
  The start information of an OB with standard access is always stored in the first 20 bytes of the "Temp" section in the block interface. By contrast, the start information of an OB with optimized access is always written to the "Input" section. For this reason, the block interface of OBs will also change whenever you convert block access. Additional information is provided in the following sections.

Converting block access from "standard" to "optimized".

A block copied from the CPU of the S7-300/400 product range to a CPU of the S7-1200/1500 product range will initially retain the "standard" access mode. However, you can significantly increase the performance of program execution by using blocks with optimized access, which is why it may be useful to modify the access mode manually.

The blocks are adapted as follows in the course of conversion:

- Function blocks
  All interface parameters are assigned the "Non-retain" retentivity setting.
- Global data blocks
  The retentivity setting that was assigned centrally to the entire data block is transferred to the individual interface parameters. It is now possible to manipulate the retentivity setting of the various parameters.
  However, the following rule will still apply: For structured data type tags, the retentivity setting always applies to the entire structure. You cannot assign separate retentivity settings to the various elements within a structured data type. It therefore follows that you cannot assign individual retentivity settings to the tags of data blocks that are based on PLC data types.
- Organization blocks
  All interface parameters that are stored in the first 20 bytes of the "Temp" section will be deleted. New CPU-specific start information is entered in the "Input" section. Naming conflicts with user-defined interface parameters occurring in the process are resolved by renaming the user-defined interface parameters.

CAUTION

The conversion of the block access has the following consequences:

- Absolute addressing of the interface parameters of the block is no longer possible after conversion of block access to the "optimized mode.
  Example: `#L0.1` is no longer valid.
- Since conversion to the "optimized" block access mode of organization blocks also modifies the OB interface,

you may possibly have to adapt, recompile and load the program again due to these changes.

Converting block access from "optimized" to "standard".

If you want to copy or move a block from the CPU of the S7-300/400 product range to a CPU of the S7-1200/1500 product range, you first need to set the "standard" access mode.

The blocks are adapted as follows in the course of conversion:

- Function blocks and global data blocks.
  You can no longer set a retentivity in the function block. The corresponding setting is made in the instance data block.
  All interface parameters in the instance DB or global DB are assigned the same retentivity setting. The conversion is subject to the following rule:
  - If all interface parameters in the original block were retentive, the entire block will be retentive after conversion.
  - If all interface parameters in the original block were non-retentive, the entire block will be non-retentive after conversion.
  - If the interface parameters in the original block had different retentivity settings, the entire block will be non-retentive after conversion.
- Organization blocks
  All interface parameters stored in the "Input" section will be deleted. New CPU-specific start information is entered in the "Temp" section. This data is written to the first 20 bytes. Naming conflicts with user-defined interface parameters occurring in the process are resolved by renaming the user-defined interface parameters.

CAUTION

The conversion of the block access has the following consequences:

Since a conversion to "standard" block access mode might change the retentivity settings of the interface parameters, you may possibly have to adapt, recompile and load the program again due to these changes.

See also

Basics of block access (Page 2102)

Specifying block properties (Page 4725)

11.1.3 Block calls

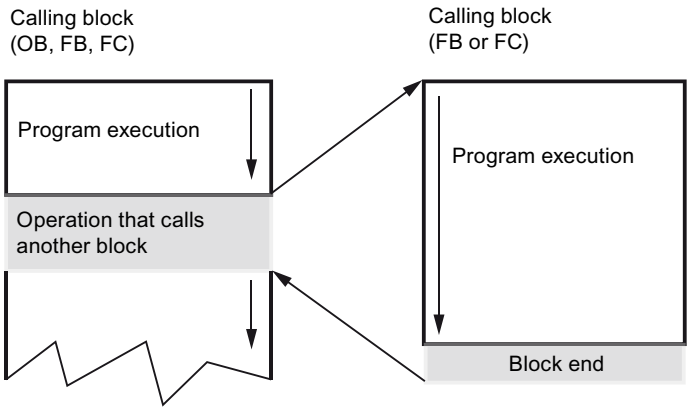
11.1.3.1 Principles of block calls

Function of block calls

For your blocks to be executed in the user program, they need to be called from another block.

When one block calls another block, the instructions of the called block are executed. Only when execution of the called block has been completed does execution of the calling block resume. The execution is continued with the instruction that follows on the block call.

The following figure shows the sequence of a block call within a user program:



Parameter transfer

When a block is called, you must assign values to the parameters in the block interface. By providing input parameters you specify the data with which the block is executed. By providing the output parameters you specify where the execution results are saved.

See also

Call hierarchy (Page 2108)

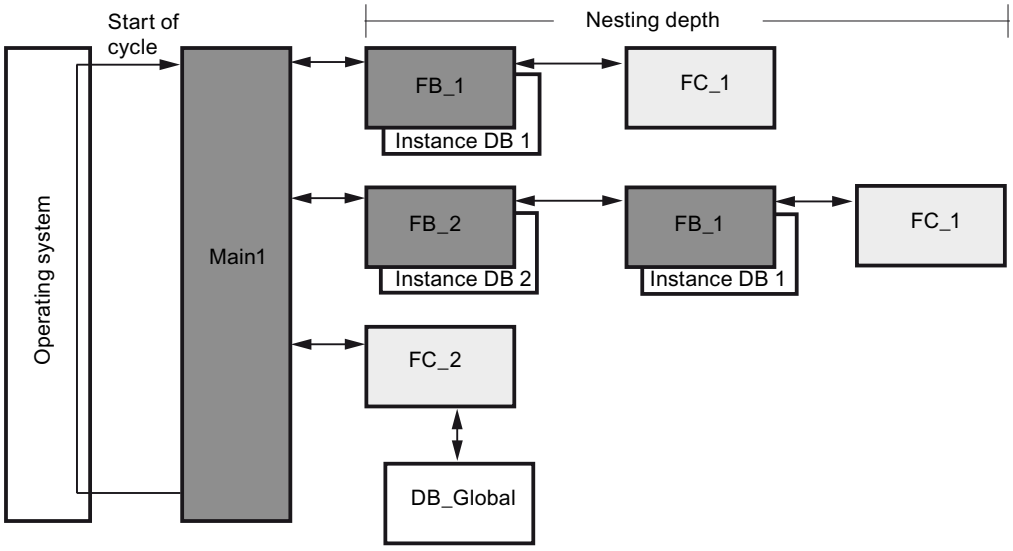
Basics of instances (Page 2110)

11.1.3.2 Call hierarchy

Definition

The order and nesting of block calls is referred to as the call hierarchy.

The following figure shows an example of the order and nesting of block calls within an execution cycle:



Nesting depth for blocks

The permissible nesting depth for blocks depends on the CPU that is used.

The following table shows the guide values for the maximum nesting depth. For detailed information on the CPU you are using, refer to the technical specifications in the hardware documentation. To call the hardware documentation on the Internet, click the links in the table.

CPU family	Nesting depth (guide value)	Link to hardware documentation
S7-1500	24 blocks per priority class	SIMATIC S7-1500 / ET 200MP Manual Collection (https://support.industry.siemens.com/cs/ww/en/view/86140384)
S7-1200	16 blocks from cycle or startup OB, 6 additional blocks within any interrupt event OB	SIMATIC S7-1200 Automation System (https://support.industry.siemens.com/cs/ww/en/view/91696622)
S7-400	24 blocks per priority class, 1-2 additional blocks within an error OB	SIMATIC S7-400 automation system S7-400 CPU data (https://support.industry.siemens.com/cs/de/en/view/53385241)
S7-300	16 blocks per priority class, 4 additional blocks within an error OB	SIMATIC S7-300 CPU 31xC and CPU 31x: Technical specifications (https://support.industry.siemens.com/cs/de/de/view/12996906/en)
ET 200SP	24 blocks	SIMATIC ET 200SP Manual Collection (https://support.industry.siemens.com/cs/ww/en/view/84133942)

Nesting depth for structures

Structures (STRUCT) and PLC data types (UDT) can be nested to a depth of 8. This nesting depth is independent of the CPU used.

See also

Basics of instances (Page 2110)

Principles of block calls (Page 2107)

11.1.3.3 Instances

Basics of instances

Definition

The call of a function block is referred to as an instance. The data with which the instance works is stored in an instance data block.

Instance data blocks have the following properties:

- Instance data blocks are always assigned to an FB.
- The structure of the instance data block is derived from the interface of the corresponding FB and can only be changed there.
- Instance data blocks are created automatically when a function block is called.

Assignment of instances

You have the following options for assigning instance data blocks:

- Single instance:
  The called function block saves its data in an instance data block of its own.
  See also: Single instances (Page 2111)
- Multi-instance:
  The called function block does not save its data in its own instance data block, but in the instance data block of the calling function block.
  See also: Multi-instances (Page 2112)
- Parameter instance:
  You transfer the instance of the block to be called as an in/out parameter to the calling block.
  See also: Parameter instances (Page 2114)

Note

Parameter type "Block_DB" for entry of the instance DB

In LAD and FBD, you cannot enter the instance DB of a function block using an input of the data type "BLOCK_DB". Use a parameter instance instead.

See also

Overview of the block interface (Page 4764)

Addressing instance data (Page 2152)

Principles of block calls (Page 2107)

Call hierarchy (Page 2108)

Single instances

Definition

The call of a function block (FB) that is assigned its own instance data block is referred to as a single instance.

By assigning the instance data block, you specify where the instance data of the FB will be saved.

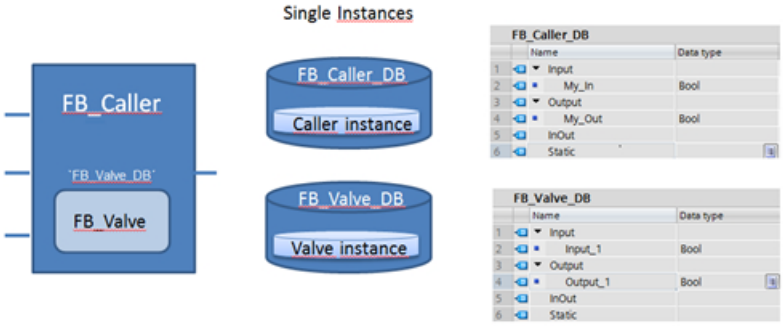
Advantages

The use of single instances provides the following advantages:

- Reusability of the function blocks
- Good structuring options for simple programs

How single instances work

The following figure shows an FB that uses a further FB ("FB_Valve"). "FB_Valve" is called as a single instance; in other words, it saves its data in its own instance DB.



The structure of the instance data block is defined by the interface of the corresponding FB and can only be changed there. The instance DB contains the following data:

- Block parameters
  The block parameters in the sections Input, Output and InOut form the block interface for the call in the program.
- Static local data
  Static local data in the section Static is used to permanently store intermediate results beyond the current program cycle.

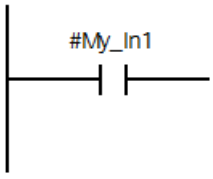
Creating single instances

Instance DBs are created when a function block is called. When you call an FB, a dialog appears in which you can specify whether the FB will be called as a single instance, multi-instance or parameter instance.

Addressing data in single instances

To address a tag from the instance DB of the current block, enter the character # followed by the symbolic tag name:

#<tag name>



See also

Basics of instances (Page 2110)

Multi-instances (Page 2112)

Multi-instances

Definition

When a function block (FB) calls a further FB, you do not need to create a separate instance DB for the called block. The called FB can also save its instance data in the instance DB of the calling FB. This type of block call is referred to as a multi-instance.

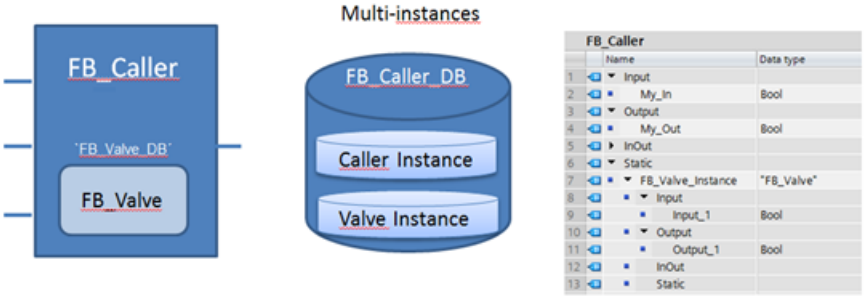
Advantages

The use of multi-instances provides the following advantages:

- Good structuring options for complex blocks
- Lower number of instance DBs
- Simple programming of local sub-programs.

How multi-instances work

The following figure shows an FB that uses a further FB ("FB_Valve"). "FB_Valve" is called as a multi-instance; in other words, it saves its data in the instance DB of the calling FB. Multi-instance data is located in the "Static" section of the calling block.

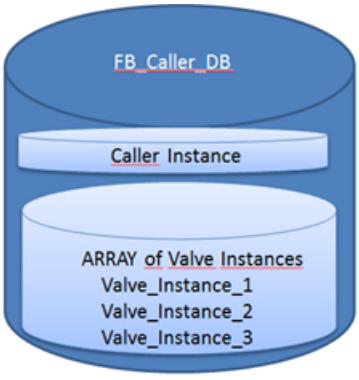


Creating multi-instances

Multi-instances are defined when a function block is called. When you call an FB, a dialog appears in which you can specify whether the FB will be called as a single instance, multi-instance or parameter instance.

ARRAYs of multi-instances

Multi-instances can also be created as an ARRAY. You can address the individual ARRAY elements via a variable index, e.g. when processing program loops.



You do not declare ARRAYS of multi-instances in the dialog that appears when you call the block, but directly in the block interface.

See also: Declaring multi-instances

Addressing data in multi-instances

Both the calling and the called FB can access the multi-instance data.

Within the called FB, you address the data with the # character followed by the tag name. From the calling block, you access the data with #<multi-instance name>.<tag name>.

Syntax	Meaning
`#<tag name>`	`//Access within the called FB`
`#<multi-instance name>.<tag name>`	`//Access from the calling FB`

See also

Basics of instances (Page 2110)

Single instances (Page 2111)

Example of the use of parameter instances (Page 2117)

Parameter instances

Definition

The parameter instance provides a special option for instantiation:

Here, at the time of creating the program, you do not specify a fixed instance for saving the block data. Instead, at runtime, you transfer the instance of the block to be called as an in/out parameter to the calling block.

Advantages

The use of parameter instances provides the following advantages:

- At runtime, you can define which instance is currently being used.
- You can process different instances iteratively in program loops.

Note

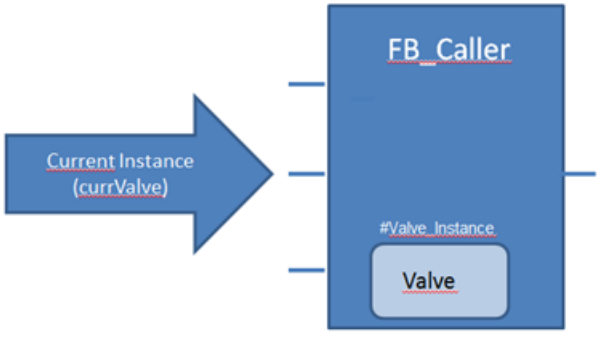
ARRAYs of multi-instances

To simplify the iterative processing of instances in program loops, define an ARRAY of multi-instances.

See also: Multi-instances (Page 2112)

How parameter instances work

The following figure shows the "FB_Caller" that uses a further FB ("Valve"). "Valve" is called as a parameter instance. Therefore, when calling "Valve", do not specify a concrete instance. Instead, define an in/out parameter ("#Valve_Instance") with which the concrete instance is transferred at runtime.

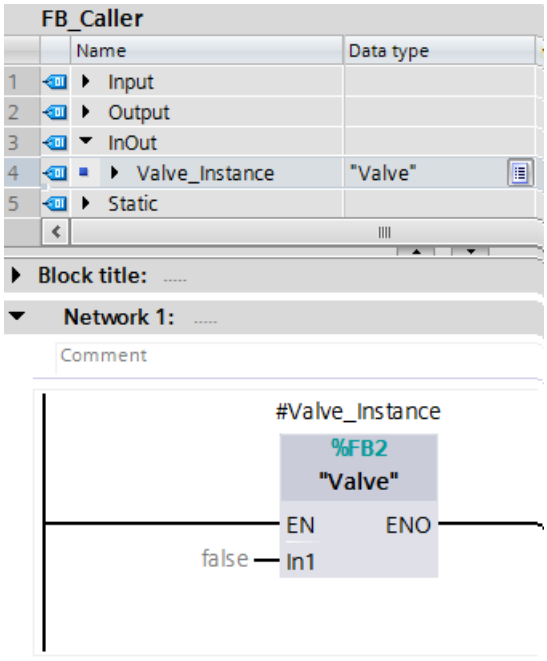


Creating parameter instances

Parameter instances are defined when a function block is called: When you call an FB, a dialog appears in which you can specify whether the FB will be called as a single instance, multi-instance or parameter instance. As an alternative, you also have the option of entering parameter instances manually directly in the block interface.

See also: Declaring parameter instances (Page 4781)

The following figure shows the call of a parameter instance for the FB "Valve":

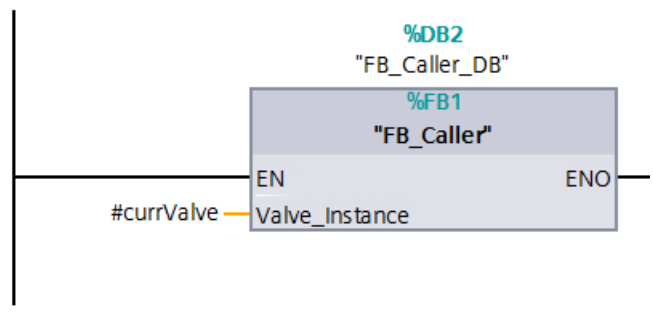


Transferring the instance as a parameter

Each time the higher-level FB is called ("FB_Caller" in the example) transfer an instance for the called FB (in the example "Valve"). You can transfer the following types of instances:

- Single instance
  Transfer an existing instance DB of the called FB (the FB "Valve" in the example).
- Multi-instance
  Transfer an existing multi-instance of the called FB.
- Individual element of an ARRAY of multi-instances
  Transfer an element of an existing ARRAY of multi-instances.

The following figure shows the call of "FB _Caller". With the "Valve_Instance" parameter, the instance "#currValve" is transferred. This instance, which is pending at the #currValve parameter during runtime, is used to call "Valve".



See also

Example of the use of parameter instances (Page 2117)

Example of the use of parameter instances

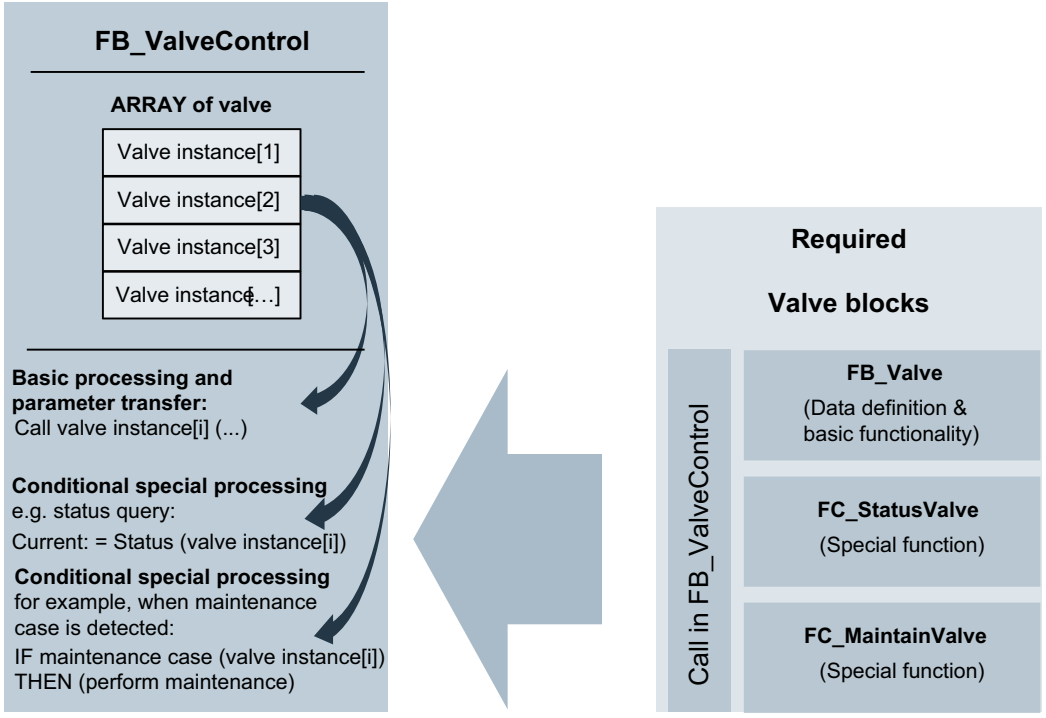
Description

The use of parameter instances allows you to transfer the instance of a function block to another block (FB or FC) for further processing, for example, a data query or an error analysis, or even have the function block executed with the passed instance.

The ARRAY multi-instances can be used to combine objects of the same type and process their instances indexed in a program loop. Indexed ARRAY elements can also be passed to another block as a parameter instance.

To illustrate this, the object "Valve" (FB_Valve) was selected in the following example. On this basis, you are provided with all required information on valve processing in a program block, "FB_ValveControl" in this case.

The figure below illustrates the features needed and how they can be used:



- The two functions "FC_StatusValve" and "FC_MaintainValve" process a parameter instance of the "FB_Valve" program block which they received in the call.
- In the "FB_ValveControl" program block, you use the ARRAY multi-instances on the one hand to declare the number of existing valves and, on the other hand, to manage all of the valves in a loop by means of various functions.

Note

Completeness

This solution is only an example to illustrate how this task can be approached. Keep in mind that you need to adjust the program code to your actual task.

The following program blocks are required to implement this example:

Block	Description	Programming language
FB_ValveControl	This function block is used as a control block, where you can process all the valves using an ARRAY of multi-instances.	SCL
FB_Valve	This function block contains the definition of the valve data and the program code for processing a valve.	SCL (The three program blocks are used by the "FB_ValveControl" program block.)
FC_StatusValve	The function returns the status of the currently processed valve.	
FC_MaintainValve	This function checks if maintenance is required on the valve. If required, the maintenance is performed and the function value TRUE is returned when everything is completed.	

Procedure: Create "FB_Valve"

To create the SCL function block, follow these steps:

1. Double-click the "Add new block" command.
   The "Add new block" dialog opens.
2. Click on the "Function block (FB)" button.
3. Enter the name "FB_Valve".
4. Select "SCL" as the language.
5. Click "OK".

6. The declaration of the block interface is based on a valve which can open and close, and could look like this, for example:

FB_Valve

		Name	Data type	Comment
1		Input		
2		R_max_A_Valve	Real	Maximum valve performance in mA
3		period_duration_factor	Int	Duration = "period_duration_factor" x 0.1 x configured duration
4		MANUAL_DO	Bool	Check output manually
5		SET_DO	Bool	Set output
6		Output		
7		I_actual_A	Real	Actual valve performance in mA
8		Current_Limit	Bool	Maximum valve performance exceeded (< 4A)
9		InOut		
10		SW_ENABLE	Bool	When this tag has the signal state TRUE, maintenance is required.
11		Static		
12		I_actual_mil	Int	The actual value of the program block "ValveControl", that is queried every millisecond.
13		control_byte	Byte	Control Byte: Bit0=SW_ENABLE; Bit1=MANUAL_DO; Bit2=SET_DO

7. The program code for controlling the valves could look like this, for example:

```
1  #I_actual_mil := #period_duration_factor;
2  #SW_ENABLE := #control_byte.%X4;
```

Procedure: Create "FC_StatusValve"

To create the SCL function "FC_StatusValve", follow these steps:

1. Double-click on the "Add new block" command.
   The "Add new block" dialog opens.
2. Click on the "Function (FC)" button.
3. Enter the name "FC_StatusValve".
4. Select "SCL" as the language.
5. Click "OK".
6. The declaration of the block interface with integration of the parameter instance "FB_Valve" could look like this, for example:

FC_StatusValve

		Name	Data type	Comment
1		Input		
2		Output		
3		InOut		
4		CurrentValve	"FB_Valve"	The function block "FB_Valve" is used as a parameter instance to exchange its data with the "FB_Valve".
5		<Add new>		
6		Temp		
7		Constant		
8		Return		
9		FC_StatusValve	Byte	Return value of the function

7. Write the following program code, for example:

```
1  #FC_StatusValve := #CurrentValve.control_byte;
```

Procedure: Create "FC_MaintainValve"

To create the SCL function "FC_MaintainValve", follow these steps:

1. Double-click on the "Add new block" command.
   The "Add new block" dialog opens.
2. Click on the "Function (FC)" button.
3. Enter the name "FC_MaintainValve".
4. Select "SCL" as the language.
5. Click "OK".
6. The declaration of the block interface with integration of the parameter instance "FB_Valve" could look like this, for example:

FC_MaintainValve

	Name	Data type	Comment
1	Input		
2	NewValue	Int	Specifies a new value for the duration of the current valve.
3	Output		
4	InOut		
5	CurrentValve	"FB_Valve"	The function block "FB_Valve" is used as a parameter instance to exchange its data with the "FB_Valve".
6	Temp		
7	Constant		
8	Return		
9	FC_MaintainValve	Bool	Return value of the function

7. Write the following program code, for example:

```
IF #CurrentValve.SW_ENABLE THEN // Valve control enabled
    // Do some tuning and maintenance
    #CurrentValve.period_duration_factor := #NewValue;
    #CurrentValve.control_byte.%X0 := true;
    #FC_MaintainValve := true;
ELSE
    // Tuning currently not allowed
    #FC_MaintainValve := false;
    ;
END_IF;
```

Procedure: Create "FB_ValveControl"

To create the SCL function block, follow these steps:

1. Double-click the "Add new block" command.
   The "Add new block" dialog opens.
2. Click on the "Function block (FB)" button.
3. Enter the name "FB_ValveControl".
4. Select "SCL" as the language.
5. Click "OK".

6. Define the user constant:

		Name	Data type	Value	Comment
1		vmax	Int	99	Number of the valves

The user constant "vmax" is used in this example in order to easily adapt the program to a different number of valves.

7. The declaration of the block interface with integration of the valve instances from "FB_Valve" could look like this, for example:

FB_ValveControl

		Name	Data type	Comment
1		Input		
2		ValueMaintain	Int	Maintenance
3		Duration	Int	Duration of the valve control
4		Output		
5		Bypass	Bool	True or false
6		InOut		
7		Index	Int	Number index
8		Static		
9		ValveInstances	Array[0.."vmax"] of "FB_Valve"	Multiple instances of the valve
10		ValveStatus	Array[0.."vmax"] of Byte	Status of the valve
11		CurrentAmpere	Real	Current number of amperes

8. Write the following program code, for example:

```
// Set #Index, given from outside, to a value that adresses an existing valve.
#Index := (#Index + 1) MOD ("vmax" + 1);

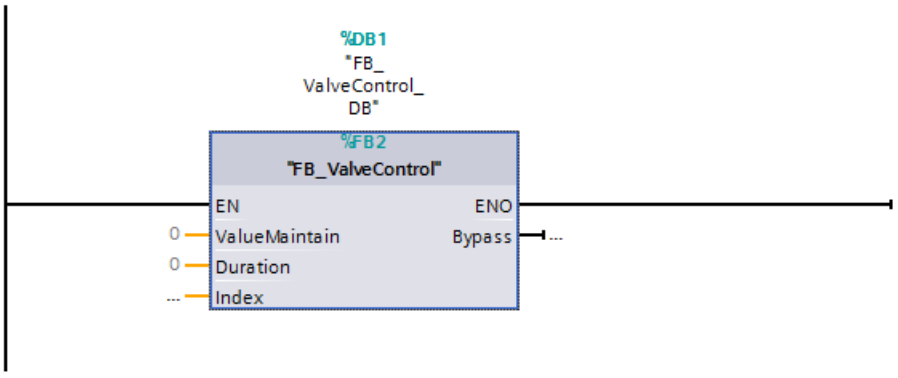
// Indirect call of a valve that is addressed via an ARRAY index.
#ValveInstances[#Index](R_max_A_Valve := 1.0,
                        period_duration_factor := #Duration,
                        MANUAL_DO := 0,
                        SET_DO := 0,
                        I_actual_A => #CurrentAmpere,
                        Current_Limit => #Bypass);

// Do some checks and tuning on the valve.
// FC_MaintainValve sets some values of the passed valve.
// FC_StatusValve returns the current status that is stored as internal value in the Instance.
IF "FC_MaintainValve"(CurrentValve := #ValveInstances[#Index],
                      NewValue := #ValueMaintain) THEN
    #ValveStatus[#Index] := #ValveStatus[#Index] XOR "FC_StatusValve"(CurrentValve := #ValveInstances[#Index]);
END_IF;
```

Procedure: Call of "FB_ValveControl" in OB 1

To call the "FB_ValveControl" function block in OB 1, follow these steps:

1. Open the "Main [OB1]" block with a double-click.
2. Use a drag-and-drop operation to add the "FB_ValveControl" function block to OB1.



Result

One valve is processed in each program cycle. The respective valve selected by index is processed, its status queried and maintenance is performed, if needed.

In this example, one valve per cycle has been processed. Of course, it is also possible to process all valves in one program loop, but then the cycle time increases accordingly. If you need a short program cycle, we recommend that you avoid working with a program loop.

See also

Parameter instances (Page 2114)

Multi-instances (Page 2112)

11.1.3.4 Parameter transfer at block call

Rules for supplying block parameters

Block parameters

The calling block gives the called block the values with which it is to work. These values are referred to as block parameters. The input parameters provide the called block with the values that it has to process. The block returns the results via the output parameters.

Block parameters therefore form the interface between the calling and the called block.

You use input parameters when you want to only query or read values, and output parameters when you want to set or write these values. If block parameters are read and written you have to create these as in-out parameters.

The following rules apply to the use of block parameters within the block:

- Input parameters may only be read.
- Output parameters may only be written.
- In/out parameters may be read and written.

Formal and actual parameters

The block parameters are defined in the interface of the called block. These parameters are referred to as formal parameters. They are placeholders for the parameters that are transferred to the block when it is called. The parameters transferred to the block when it is called are referred to as actual parameters.

The data types of actual and formal parameters must be identical or convertible according to the rules of data type conversion.

Note

S7-1200/1500: Transferring peripheral inputs or outputs as block parameters

If you supply an input parameter with a peripheral input or output, it is possible that I/O access errors will occur during runtime when the block is called, e.g. a read error when an input module is accessed directly.

The system response of the CPUs of the S7-1200/1500 series is as follows:

The block is not called as a result of the I/O access error. Program execution is continued after the block call. If OB 122 exists or local error handling is enabled, these are executed.

To prevent the situation that the block is not called as a result of an I/O access error, first copy the peripheral input or output to a local tag (Temp) and transfer it as a block parameter to the called block.

See also

Using tags within the program (Page 2140)

Parameter assignment to function blocks (Page 2126)

Parameter assignment to functions (Page 2123)

Rules for declaring the block interface (Page 4765)

PLC data types (UDT) (Page 2277)

Parameter assignment to functions

Parameters of functions (FC)

Functions have no data memory in which values of block parameters can be stored. Therefore, when a function is called, all formal parameters must be assigned actual parameters.

Input parameters (Input)

Input parameters are only read once before each block call. Therefore, the rule is that writing an input parameter within the block does not affect the actual parameter. Only the formal parameter is written.

Output parameters (Output)

Output parameters are only read once after each block call. Therefore, the rule is that output parameters should not be read within the block. If you nevertheless read an output parameter, please note that only the value of the formal parameter is read. The value of the actual parameter cannot be read within the block.

If an output parameter of a function is not written in this function, the value that is predefined for the specified data type is used. For example, the value "false" is predefined for BOOL. However, structured output parameters are not pre-assigned with a value.

To prevent unintentional further processing of the predefined value or an undefined value, note the following when programming the block:

- Make sure that the output parameters are written with values for all possible program paths within the block. In doing so, note that jump commands may skip instruction sequences in which outputs are set, for example.
- Note that the set and reset commands are dependent on the result of the logic operation. If the value of an output parameter is determined with these commands and RLO = 0, a value will not be generated.
- If possible, assign a default value for the output parameters of functions.

In/out parameters (InOut)

In/out parameters are read before the block call and written after the block call. If you read or write the parameter within the block, you only access its formal parameter.

An exception is in/out parameters with a structured data type. Structured data types consist of several data elements, for example ARRAY or STRUCT. These are transferred to the called block by a pointer. You therefore always access the actual parameter when you read or write a structured in/out parameter within a block.

When an in/out parameter of a function is not written to this function, the old output value or the input value is used as a value. Nevertheless, you should observe the information provided above for output parameters so that old values are not inadvertently processed further.

Temporary local data (Temp)

Temporary local data is only available during block processing. It is treated differently depending on the optimization setting of the block:

- Standard access
  The following rule applies to code blocks with standard access as well as to all tags with retentivity setting "Set in IDB":
  If you are using temporary local data, you must ensure that the values are initialized prior to use. Otherwise, the values will be random. Temporary data of the STRING of WSTRING data type is an exception: It is automatically pre-assigned the actual length 0.
- Optimized access
  The following rules apply to code blocks with optimized access:
  - If a temporary tag is not written within a function, the value that is predefined for the specified data type is used.
    The following table shows some examples of pre-defined values:

Data type	Pre-defined value
Bool	False
Int	0
REAL	0.0
Char	' '
Wchar	WCHAR#' '
DTL	DTL#1970-01-01-00:00:00
DATE_AND_TIME	DT#1990-01-01-00:00:00
Date	D#1990-01-01

  - Elements of the PLC data types are pre-assigned with the default value that is specified in the declaration of the PLC data type (UDT).
  - STRINGs and WSTRINGs are pre-assigned with the actual value "0", even if they are used within a PLC data type.
  - Elements of the data type ARRAY are pre-assigned with the value "0", even if they are used within a PLC data type.

Function value (Return)

Functions normally calculate a function value. This function value can be returned to the calling block via the RET_VAL output parameter. For this, the RET_VAL output parameter must be declared in the interface of the function. RET_VAL is always the first output parameter of a function. All data types are permitted for the RET_VAL parameter except ARRAY and STRUCT, as well as parameter types TIMER and COUNTER.

In the SCL programming language functions can be call directly in an expression. The result of the expression is then formed with the calculated function value. Therefore, the data type ANY is not permitted in SCL for the function value.

See also

Parameter assignment to function blocks (Page 2126)

Rules for supplying block parameters (Page 2122)

Calling functions (Page 5040)

Examples for calling functions in SCL (Page 5043)

Parameter assignment to function blocks

Supplying parameters of function blocks (FB)

In the case of function blocks the parameter values will be stored in the instance data.

If the input, output, or in-out parameters of a function block were not assigned with values, the stored values are used.

In some cases, it is mandatory to specify an actual parameter.

The following table shows which parameters of a function block must be assigned actual parameters:

Parameter	Elementary data type	Structured data type	Parameter type
Input (Input)	optional	optional	required
Output (Output)	optional	optional	required
In-out (InOut)	optional	required	Permitted with S7-1200 only, parameter assignment required
Temporary (Temp)	required S7-1500: Optional with optimized block access	required S7-1500: Optional with optimized block access	required

See also

Rules for supplying block parameters (Page 2122)

Parameter assignment to functions (Page 2123)

Parameter types (Page 2325)

Transfer parameter as copy or as pointer

Introduction

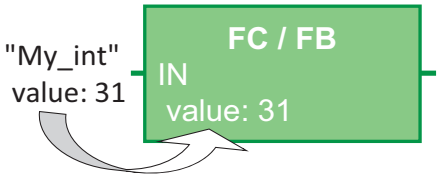
When a block is called, you transfer data to the parameters in the block interface. At the input parameters, you transfer the data with which the block is to work. At the output parameters, you specify where the results of the processing are saved. In/out parameters are used to transfer data to the called block as well as to return results.

Internally, STEP 7 recognizes two different methods of parameter transfer: The data is transferred either as pointer or as copy, depending on the transfer range and data type of the parameter.

Transfer as copy (Call by value)

During a block call, the value of the operand is copied to the input parameter of the called block. With function blocks, the copy is stored in the instance DB; with functions, it is stored in the block stack. Additional storage space is required for the copy.

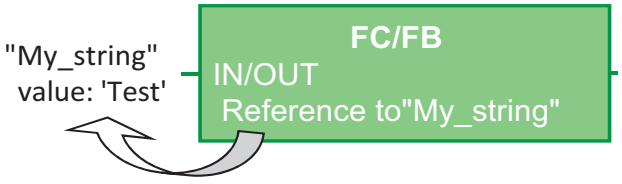
This means that the called block always works with the value that the specified operand had at the time of the block call. It cannot access the operand directly. Write access modifies only the copy, but not the actual value of the specified operand. Read access reads only the copy that was created at the time of the block call.



Transfer as pointer (Call by reference)

The parameters are referenced via a pointer during the block call.

This means that the called block can directly access the memory address of the operand that is specified as parameter: Write access directly results in the changing of the specified operand. Read access reads the value of the operand directly at the time of access. As no copy is created, no additional memory is required.



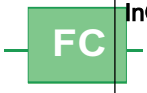
Note

Declare structured data types in the "InOut" area

If possible, use the "InOut" area in the block interface for structured tags (e.g., of data type ARRAY, STRUCT, STRING, …). Because structured in/out parameters by default are transferred as pointers, the required data memory is not increased unnecessarily.

Parameter transfer with S7-1200/1500

The following table shows how block parameters with elementary or structured data type are transferred in S7-1200/1500. Elementary data types are, for example, BOOL, INT or BYTE. Structured data types are, for example, ARRAY, STRUCT or STRING.

		Elementary data types	Structured data types
	**Input**	Copy	Pointer
	Output	Copy	Pointer
	InOut	Copy	Pointer
	**Input**	Copy	Copy
	Output	Copy	Copy
	InOut	Copy	Pointer

Note

Parameter transfer between blocks with optimized access and blocks with standard access

If structures are transferred to the called block as in/out parameters during a block call, they are transferred by default as pointers (Call by reference).

However, this does not apply if the two blocks have different optimization settings: If one of the blocks has the property "Optimized access" and the other block has the property "Standard access", all parameters are always transferred as copies (Call by value).

If the block contains numerous structured parameters, this can quickly lead to the temporary memory area (local data stack) overflowing.

Problems can also occur if the original operands are changed by asynchronous processes, such as by HMI access or interrupt OBs. If the copies are copied back to the original operands after the block processing, the changes carried out asynchronously since the creation of the copy during the block call are hereby overwritten at the original operands.

You can avoid this by setting the same access type for both blocks or by having the asynchronous access written first to a separate memory area and then copying this area synchronously at a suitable time.

See also:

Blocks with optimized access

FAQ 109478253: How can it happen that data of the HMI system or of the Web server is overwritten in the S7-1500? (https://support.industry.siemens.com/cs/de/en/view/109478253)

Parameter transfer with S7-300/400

The following table shows how block parameters with elementary or structured data type are transferred in S7-300/400.

		Elementary data types	Structured data types
FC	Input	Copy*	Pointer
	Output	Copy*	Pointer
	InOut	Copy*	Pointer
FB	Input	Copy	Copy
	Output	Copy	Copy
	InOut	Copy	Pointer

* Exception: Operands from the memory areas I, Q, M, P, L and partly qualified DB addresses (for example, "DW 2") are transferred as pointer.

Note

Special aspects of transfer as pointer in S7-300/400

In cases in which the parameters are transferred via a pointer, it is not possible to forward output parameters or in/out parameters from the calling block to the input parameters of the called block.

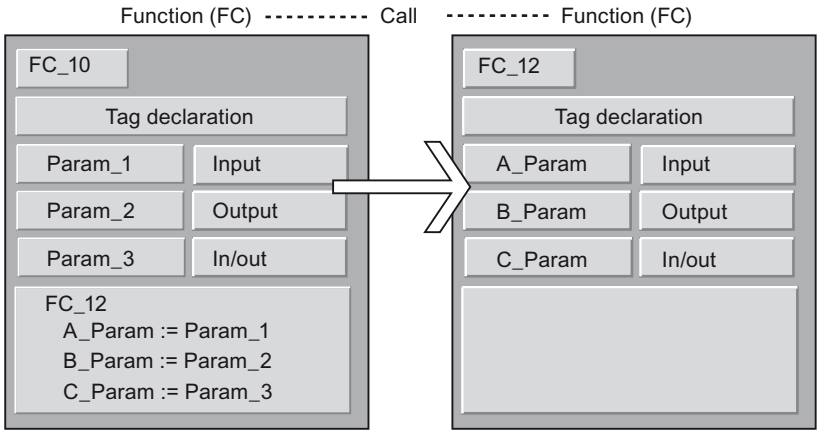
Forwarding of block parameters

Basic information on forwarding block parameters

Definition

The "Forwarding" of block parameters is a special type of parameter use. In this case the block parameters of the calling block are forwarded to the parameters of the called block. The called block uses the values that are currently present at the block parameters of the calling block as the actual parameters.

The following figure shows how the parameters of the function FC_10 are forwarded to the function FC_12:



Rules for LAD/FBD

The following general rules apply in LAD and FBD:

- Input parameters can only be forwarded to input parameters.
- Output parameters can only be forwarded to output parameters.
- In/out parameters can be forwarded to all parameter types.
- In S7-300/400, the two block parameters must have the same data type.
- In S7-1200/1500, the parameters can also be converted according to the rules of implicit conversion.

Rules for STL

The following general rules apply in STL:

- Input parameters can only be forwarded to input parameters.
- Output parameters can only be forwarded to output parameters.
- In/out parameters can be forwarded to all parameter types.
- Both block parameters must have the same data type. In STL, this rule applies to all CPU families.

Rules for SCL

The rules for SCL are less stringent. So that programs from previous SCL versions can be taken over more easily, additional parameter transfer options are permissible. You can, for example, forward a structured in/out parameter to an input parameter. However, in this case you must ensure that the actual parameter is not located in the temporary local data or in a global data block.

Additional rules are described in detail in the following chapters.

See also

Calling a function by another function (Page 2131)

Call of a function by a function block (Page 2132)

Call of a function block by a function (Page 2132)

Call of a function block by another function block (Page 2133)

Calling a function by another function

Permissible data types for the call of a function by another function

Specific rules apply to the forwarding of formal parameters. The following table shows the rules according to which parameters can be forwarded in the various CPU families:

FC calls FC		Data types					
Actual parameter (calling block)	Formal parameters (called block)	Standard data types	ARRAY, STRUCT, STRING, WSTRING, DT	ANY, POINTER	VARIANT	Parameter types (TIMER, COUNTER, BLOCK_XX)	DB_Any
Input	Input	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 as of V2 S7-1500
Output	Output	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	-	S7-1200 S7-1500	-	-
InOut	Input	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	-	-
InOut	Output	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	-	S7-1200 S7-1500	-	-
InOut	InOut	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	-	-

See also

Basic information on forwarding block parameters (Page 2129)

Call of a function by a function block

Permissible data types for the call of a function by a function block

Specific rules apply to the forwarding of formal parameters. The following table shows the rules according to which parameters can be forwarded in the various CPU families:

FB calls FC		Data types					
Actual parameter (calling block)	Formal parameters (called block)	Standard data types	ARRAY, STRUCT, STRING, WSTRING, DT	ANY, POINTER	VARIANT	Parameter types (TIMER, COUNTER, BLOCK_XX)	DB_Any
Input	Input	S7-300/400 S7-1200 S7-1500	S7-300/400 S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 as of V2 S7-1500
Output	Output	S7-300/400 S7-1200 S7-1500	S7-300/400 S7-1200 S7-1500	-	S7-1200 S7-1500	-	-
InOut	Input	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	-	-
InOut	Output	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	-	S7-1200 S7-1500	-	-
InOut	InOut	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	-	-

See also

Basic information on forwarding block parameters (Page 2129)

Call of a function block by a function

Permissible data types for the call of a function block by a function

Specific rules apply to the forwarding of formal parameters. The following table shows the rules according to which parameters can be forwarded in the various CPU families:

FC calls FB		Data types					
Actual parameter (calling block)	Formal parameters (called block)	Standard data types	ARRAY, STRUCT, STRING, WSTRING, DT	ANY, POINTER	VARIANT	Parameter types (TIMER, COUNTER, BLOCK_XX)	DB_Any

Input	**Input**	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	S7-300/400 S7-1500	S7-1200 as of V2 S7-1500
Output	**Output**	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	-	S7-1200 S7-1500	-	-
InOut	**Input**	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	-	-
InOut	**Output**	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	-	S7-1200 S7-1500	-	-
InOut	**InOut**	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	-	-

See also

Basic information on forwarding block parameters (Page 2129)

Call of a function block by another function block

Permissible data types for the call of a function block by another function block

Specific rules apply to the forwarding of formal parameters. The following table shows the rules according to which parameters can be forwarded in the various CPU families:

FB calls FB		**Data types**					
Actual parameter (calling block)	**Formal parameters (called block)**	**Standard data types**	**ARRAY, STRUCT, STRING, WSTRING, DT**	**ANY, POINTER**	**VARIANT**	**Parameter types (TIMER, COUNTER, BLOCK_XX)**	**DB_Any**
Input	**Input**	S7-300/400 S7-1200 S7-1500	S7-300/400 S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	S7-300/400 S7-1500	S7-1200 as of V2 S7-1500
Output	**Output**	S7-300/400 S7-1200 S7-1500	S7-300/400 S7-1200 S7-1500	-	S7-1200 S7-1500	-	-
InOut	**Input**	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	-	-

InOut	**Output**	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	-	S7-1200 S7-1500	-	-
InOut	**InOut**	S7-300/400 S7-1200 S7-1500	S7-1200 S7-1500	S7-1500	S7-1200 S7-1500	-	-

See also

Basic information on forwarding block parameters (Page 2129)

11.1.4 Using and addressing operands

11.1.4.1 Basic information about operands

Introduction

When you program instructions you must specify which data values the instruction should process. These values are referred to as operands. You can, for example, use the following elements as operands:

- PLC tags
- Constants
- Tags in instance data blocks
- Tags in global data blocks

Absolute address and symbolic name

Operands are identified by means of an absolute address and a symbolic name. You define the names and addresses in the PLC tag table or in the tag declaration of the blocks.

Data blocks with optimized access (S7-1200, S7-1500)

Data elements in data blocks with optimized access only receive a symbolic name and no absolute address in the declaration.

See also

Displaying symbolic and absolute addresses (Page 4696)

Basics of block access (Page 2102)

11.1.4.2 Keywords

SIMATIC recognizes a range of key words whose definitions are fixed and which have a certain meaning in the program. You should not use these keywords as names for tags or constants.

Table of keywords

The following table shows all keywords.

Keywords German mnemonics	Keywords English mnemonics	Description
&	&	And logical operation of logical expressions
A	Q	Output, bit
A1	CC1	Condition code bit
A0	CC0	Condition code bit
AB	QB	Output, byte
AD	QD	Output, double word
AND	AND	And logical operation of logical expressions
ANY	ANY	Data type, pointer
AR1	AR1	Address Register 1
AR2	AR2	Address Register 2
ARRAY	ARRAY	Introduces the specification of an array and is followed by the index list between "[" and "]"
AT	AT	Overlaying tag declaration
AUTHOR	AUTHOR	Name of the author, company name, department name, or other name (max. 8 characters, no spaces)
AW	QW	Output, word
B	B	Byte
BEGIN	BEGIN	Introduces the instruction part for code blocks or initialization part for a data block
BIE	BR	Binary result
BLOCK_FB	BLOCK_FB	Parameter type for specification of an FB
BLOCK_FC	BLOCK_FC	Parameter type for specification of an FC
BLOCK_SDB	BLOCK_SDB	Parameter type for specification of an SDB
BOOL	BOOL	Data type
BY	BY	Increment of the FOR loop
BYTE	BYTE	Data type
CALL	CALL	Call

Keywords German mnemonics	Keywords English mnemonics	Description
CASE	CASE	Introduction to the CASE statement
CHAR	CHAR	Elementary data type
CODE_VERSION1	CODE_VERSION1	Label, whether an FB is multiple instance capable or not. If you want to declare multiple instances, the FB must not have this characteristic.
CONST	CONST	Start of the constant declaration
CONTINUE	CONTINUE	Instruction to exit a loop in SCL
COUNTER	COUNTER	Parameter type for specification of a counter
DATA_BLOCK	DATA_BLOCK	Introduces the data block
DATE	DATE	Data type
DATE_AND_TIME	DATE_AND_TIME	Data type
DB	DB	Data block
DB_ANY	DB_ANY	Data type
DBB	DBB	Data block, data byte
DBD	DBD	Data block, data double word
DBLG	DBLG	Data block length
DBNO	DBNO	Data block number
DBW	DBW	Data block, data word
DBX	DBX	Data block, data bit
DI	DI	Instance data block
DIB	DIB	Instance data block, data byte
DID	DID	Instance data block, data double word
DILG	DILG	Instance data block length
DINO	DINO	Instance data block number
DINT	DINT	Data type
DIW	DIW	Instance data block, data word
DIX	DIX	Instance data block, data bit
DO	DO	Introduction of the instruction part in FOR and WHILE instruction
DT	DT	Data type
DTL	DTL	Data type
DWORD	DWORD	Data type
E	I	Input (via process image), bit
EB	IB	Input (via process image), byte
ED	ID	Input (via process image), double word
ELSE	ELSE	Alternative branch in IF and CASE statement

Keywords German mnemonics	Keywords English mnemonics	Description
ELSIF	ELSIF	Alternative condition of the IF instruction
EN	EN	System operand of the EN/ENO mechanism
ENO	ENO	System operand of the EN/ENO mechanism
END_CASE	END_CASE	End of the CASE statement
END_DATA_BLOCK	END_DATA_BLOCK	Ends the data block
END_FOR	END_FOR	End of the FOR statement
END_FUNCTION	END_FUNCTION	Ends the function
END_FUNCTION_BLOCK	END_FUNCTION_BLOCK	Ends the function block
END_IF	END_IF	End of the IF instruction
END_ORGANIZATION_BLOCK	END_ORGANIZATION_BLOCK	Ends the organization block
END_REGION	END_REGION	End of a block
END_REPEAT	END_REPEAT	End of the REPEAT statement
END_STRUCT	END_STRUCT	Ends the specification of a structure
END_SYSTEM_FUNCTION	END_SYSTEM_FUNCTION	Ends the system function
END_SYSTEM_FUNCTION_BLOCK	END_SYSTEM_FUNCTION_BLOCK	Ends the system function block
END_TYPE	END_TYPE	Ends the PLC data type
END_VAR	END_VAR	Ends a declaration block
END_WHILE	END_WHILE	End of the WHILE instruction
EW	IW	Input (via process image), word
EXIT	EXIT	Instruction to exit a loop in SCL
FALSE	FALSE	Predefined Boolean constant: Logical condition false, value equal to 0
FAMILY	FAMILY	Block family name: e.g. controller
FB	FB	Function block
FC	FC	Function
FOR	FOR	Introduction of the FOR statement
FUNCTION	FUNCTION	Introduces the function
FUNCTION_BLOCK	FUNCTION_BLOCK	Introduces the function block
GOTO	GOTO	Introduction of the GOTO statement
IF	IF	Introduction of the IF instruction
INSTANCE	INSTANCE	Data type
INT	INT	Data type
KNOW_HOW_PROTECT	KNOW_HOW_PROTECT	Block protection
L	L	Local data bit
LB	LB	Local data byte
LD	LD	Local data double word

Keywords German mnemonics	Keywords English mnemonics	Description
LDT	LDT	Data type
LINT	LINT	Data type
LTIME	LTIME	Data type
LTOD	LTOD	Data type
LW	LW	Local data word
LWORD	LWORD	Data type
M	M	Memory bit
MB	MB	Memory byte
MD	MD	Memory double word
MOD	MOD	Modulo operator
MW	MW	Memory word
NAME	NAME	Block name
NETWORK	NETWORK	Network
NOT	NOT	Logic inversion
NULL	NULL	Zero pointer
OB	OB	Organization block
OF	OF	Introduction of the data type specification / Introduction of the instruction part of the CASE statement
OR	OR	Or logical operation of logical expressions
ORGANIZATION_BLOCK	ORGANIZATION_BLOCK	Introduces the organization block
OS	OS	Save overflow
OV	OV	Overflow
PA	PQ	Output (direct peripherals), bit
PAB	PQB	Output (direct peripherals), byte
PAD	PQD	Output (direct peripherals), double word
PAW	PQW	Output (direct peripherals), word
PE	PI	Input (direct peripherals), bit
PEB	PIB	Input (direct peripherals), byte
PED	PID	Input (direct peripherals), double word
PEW	PIW	Input (direct peripherals), word
POINTER	POINTER	Data type
PRAGMA_BEGIN	PRAGMA_BEGIN	Introduction of a Pragma
PRAGMA_END	PRAGMA_END	End of a Pragma
READ_ONLY	READ_ONLY	Write protection for data blocks
REAL	REAL	Data type
REGION	REGION	Introduction of a block
REPEAT	REPEAT	Introduction of the REPEAT statement

Keywords German mnemonics	Keywords English mnemonics	Description
RET_VAL	RET_VAL	Return value
RETURN	RETURN	RETURN statement in SCL
S5T	S5T	Syntax for data type S5TIME
S5TIME	S5TIME	Data type
S7_	S7_	Keywords for system attributes
SDB	SDB	System data block
SFB	SFB	System function block
SFC	SFC	System function
SINT	SINT	Data type
STRING	STRING	Data type
STRUCT	STRUCT	Introduces the specification of a structure and is followed by a list of components
STW	STW	Status word
SYSTEM_FUNCTION	SYSTEM_FUNCTION	System function
SYSTEM_FUNCTION_BLOCK	SYSTEM_FUNCTION_BLOCK	System function block
T	T	Time element (timer)
THEN	THEN	Introduction of the instruction part of an IF instruction
THIS	THIS	Syntax for access to an ARRAY data block
TIME	TIME	Elementary data type for time information
TIME_OF_DAY	TIME_OF_DAY	Data type
TIMER	TIMER	Parameter type for specification of a timer
TITLE	TITLE	Optional block title or network title
TO	TO	Definition of the full-scale value of a FOR statement
TOD	TOD	Data type
TRUE	TRUE	Predefined Boolean constant: Logical condition true, value not equal to 0
TYPE	TYPE	Introduction of the PLC data type
UDT	UDT	Global or PLC data type
UDINT	UDINT	Data type
UINT	UINT	Data type
ULINT	ULINT	Data type
UNLINKED	UNLINKED	Marking 'non runtime-related'
UNTIL	UNTIL	End of the instruction part of a REPEAT statement
USINT	USINT	Data type
UO	UO	Query after (Q1=1) AND (Q0=1)

Keywords German mnemonics	Keywords English mnemonics	Description
VAR	VAR	Introduces a declaration block
VAR_IN_OUT	VAR_IN_OUT	Introduces a declaration block
VAR_INPUT	VAR_INPUT	Introduces a declaration block
VAR_OUTPUT	VAR_OUTPUT	Introduces a declaration block
VAR_TEMP	VAR_TEMP	Introduces a declaration block
VARIANT	VARIANT	Data type
VERSION	VERSION	Version number of the block
VOID	VOID	Function has no return value
WCHAR	WCHAR	Data type
WSTRING	WSTRING	Data type
WHILE	WHILE	Introduction of a WHILE instruction
WORD	WORD	Data type
XOR	XOR	Logic operation
Z	C	Counter

11.1.4.3 Using tags within the program

Definition

A variable is a placeholder for a data value that can be changed in the program. The format of the data value is defined. The use of variables makes your program more flexible. For example, you can assign different values to variables that you have declared in the block interface for each block call. As a result, you can reuse a block you have already programmed for various purposes.

A variable consists of the following elements:

- Name
- Data type
- Absolute address
  - PLC tags and DB tags in blocks with standard access have an absolute address.
  - DB variables in blocks with optimized access have no absolute address.
- Value (optional)

Declaring Variables

You can define variables with different scopes for your program:

- PLC tags that apply in all areas of the CPU
- DB variables in global data block that can be used by all blocks throughout the CPU.
- DB tags in instance data blocks that are predominantly used within the block in which they are declared.

The following table shows the difference between the variable types:

	PLC tags	Variables in instance DBs	Variables in global DBs
Range of application	• Are valid throughout the entire CPU. • Can be used by all blocks on the CPU. • The name is unique within the CPU.	• Are predominantly used in the block in which they are defined. • The name is unique within the instance DB.	• Can be used by all blocks on the CPU. • The name is unique within the global DB.
Permissible characters	• Letters, numbers, special characters • Quotation marks are not permitted. • Reserved keywords are not permitted.	• Letters, numbers, special characters • Reserved keywords are not permitted.	• Letters, numbers, special characters • Reserved keywords are not permitted.
Use	• I/O signals (I, IB, IW, ID, Q, QB, QW, QD) • Bit memory (M, MB, MW, MD)	• Block parameters (input, output and in-out parameters), • Static data of a block	• Static data
Location of definition	PLC tag table	Block interface	Declaration table of the global DB

Note

You can find more information on permissible characters in tag names in the Siemens Industry Online Support in the following entries:

When should identifiers and operands be used in "quotation marks" in STEP 7 (TIA Portal)?

https://support.industry.siemens.com/cs/ww/en/view/109477857 (https://support.industry.siemens.com/cs/ww/en/view/109477857)

See also

Keywords (Page 2135)

Basic information about operands (Page 2134)

Displaying symbolic and absolute addresses (Page 4696)

Valid names of PLC tags (Page 4840)

Permissible addresses and data types of PLC tags (Page 4840)

11.1.4.4 Constants

Basics of constants

Definition

Constants are data with a fixed value that you cannot change during program runtime. Constants can be read by various program elements during the execution of the program but cannot be overwritten. There are defined notations for the value of a constant, depending on the data type and data format. A distinction is made between the typed and non-typed notation.

Non-typed constants

In the non-typed notation, you only enter the value of the constant without a data type. Non-typed constants do not receive their data type until the first arithmetic or logical operation in which they are used.

The example below shows the non-typed notation:

```
SCL
#My_Int1 := #My_Int2 + 12345        (*The data type of the constant "12345"
                                    results from the addition with My_Int. 2. "12345"
                                    receives the data type INT.*)
#My_Real1 := #My_Real2 + 12345      (*The data type of the constant "12345" results
                                    from the addition with My_Real2. "12345" re-
                                    ceives the data type REAL.*)
```

Typed constants

In the typed notation, you specify a data type in addition to the value of the constant.

The example below shows the typed notation:

```
SCL
#My_Int1 :=  INT#12345                    (*The data type of the constant is always
                                          INT.*)
```

Note

Constants of BOOL type in LAD/FBD

Constants of the BOOL type may not be used as inputs of instructions in S7-300/400.

Constants of the BOOL type may be used as inputs of instructions which are a system-internal function block (FB) in S7-1200/1500. These instructions are identified by the fact that the "Call options" dialog opens when you insert the instruction into a network. Boolean constants may not be used as inputs for all other instructions.

See also

Declaring the block interface (Page 4771)

Declaring global constants (Page 4852)

Calculating with constants in SCL (Page 5010)

Declaration of symbolic names for constants

Symbolic constants

You have the option of declaring symbolic names for constants and thus making constant values available under a name in the program. This makes a program more readable and easier to maintain when changing constant values.

A symbolic constant consists of the following elements:

- Name
- Data type
  Symbolic constants always have a data type; non-typed notation is not possible for symbolic constants.
- Constant value
  You can select any value from the value range of the specified data type as constant value. For information on the value ranges, refer to the "Data types" chapter.

Declaration of constants

You can define constants with different scopes of validity:

- Global constants that apply to all areas of the CPU
- Local constants that only apply within a block

The table below shows the difference between the constant types:

	Global constants	Local constants
Scope of validity	• Are valid throughout the entire CPU • The name is unique within the CPU.	• Are valid only in the block in which they were declared. • The name is unique within the block.
Permitted characters	• The permitted characters in constant names are letters, digits and special characters.	• The permitted characters in constant names are letters, digits and special characters.
Location of definition	"Constants" tab from the PLC tag table	Block interface
Representation	In quotation marks, for example.: `"Glob_Const"`	Prefixed with a number sign, for example: `#Loc_Const`

Name conflicts can occur if you have declared a local and a global constant with the same symbolic name and have used this doubly specified name as the default value of a tag. In this case, the local constant is automatically used.

Note

Downloading constant declarations (S7-300/400)

Local and global constant declarations are not downloaded into the CPU. If you download a program from a device, the constant declarations may no longer be available.

Note

You can find more information on permissible characters in constant names in the Siemens Industry Online Support in the following entries:

When should identifiers and operands be used in "quotation marks" in STEP 7 (TIA Portal)?

https://support.industry.siemens.com/cs/ww/en/view/109477857 (https://support.industry.siemens.com/cs/ww/en/view/109477857)

See also

Overview of the valid data types (Page 2239)

Calculating with constants in SCL (Page 5010)

Declaring global constants (Page 4852)

Declaring the block interface (Page 4771)

Data types of constants

Permitted data types

For constants, all basic data types as well as all derived data types are permitted:

- Binary numbers
- Bit strings
- Integers
- Floating-point numbers
- Timers
- Date and time
- Character strings

All general rules for explicit and implicit type conversion apply.

Data types of non-typed constants

Non-typed constants do not contain an explicit type specification. They do not receive their data type until the first arithmetic or logical operation in which they are used.

The example below shows how non-typed constants are used:

```
SCL
#My_Int1 := #My_Int2 + 12345        (*The data type of the constant "12345"
                                    results from the addition with My_Int. 2. "12345"
                                    receives the data type INT.*)
#My_Real1 := #My_Real2 + 12345      (*The data type of the constant "12345" results
                                    from the addition with My_Real2. "12345" re-
                                    ceives the data type REAL.*)
```

Note

STEP 7 always uses the data type with the highest possible precision

Unless the data type of a constant can be clearly defined in an expression, the data type with the highest precision available on the current CPU is used.

Example:

```
#My_Real := #My_Int / 3.5
```

In this expression an integer tag is combined with a non-typed floating point constant. In S7-300/400 the right part of the assignment is calculated in the REAL format. In S7-1200/1500, calculation is performed using the highest possible precision, which in this case means LREAL. As a result, the assignment to a REAL tag is invalid or generates a warning.

To precisely define the data type of a constant, use the typed notation.

Example:

```
#My_Real := #My_Int / REAL#3.5
```

See also

Overview of the valid data types (Page 2239)

Calculating with constants in SCL (Page 5010)

Declaring global constants (Page 4852)

Declaring the block interface (Page 4771)

Examples of using constants

Use in instructions, assignments and expressions

Constants can be used in place of tags in instructions or assignments. You can also use constants in expressions in SCL. But because constants cannot be written, they may only be used as inputs.

The example below shows possible uses of constants:

```
SCL
#My_Int :=   3;
#My_Real1 := #My_Real2 * 3;
#My_Real1 := #My_Real2 * #My_local_const;
#My_Real1 := #My_Real2 * "My_global_const";
```

Use as a default value

You can use constants as the default value of a tag. To do so, enter either the value or the symbolic name of the constant in the "Default value" column of the block interface. The data type of the constant must match the data type of the tag or be convertible with it according to the implicit conversion with IEC check.

Use as maximum STRING length

You can use local or global constants of data type UINT, UDINT, ULINT, SINT, INT, DINT, LINT as maximum STRING length.

The example below shows the use of constants as maximum STRING length:

```
SCL
STRING[#My_local_const1]
STRING["My_global_const1"]
```

Use as an ARRAY limit

You can use local or global constants of data type UINT, UDINT, ULINT, SINT, INT, DINT, LINT as ARRAY limits.

The example below shows the use of constants as ARRAY limits:

```
SCL
Array[#My_local_const1..#My_local_const2] of REAL
Array["My_global_const1".."My_global_const1"] of REAL
```

Note

Constants as ARRAY limits or as maximum STRING length

- Constants which are used as ARRAY limits or maximum STRING length cannot be changed if the memory reserve of the block is activated. This applies to both local and global constants. To change these constants, you must first disable the memory reserve.
- Changes to global constants result in inconsistencies in the blocks which use them. The inconsistencies are marked in red in the block used. To remedy these inconsistencies, the data blocks have to be updated.
  See also: Updating data blocks (Page 4800)

See also

Overview of the valid data types (Page 2239)

Calculating with constants in SCL (Page 5010)

Basic information on ARRAY (Page 2286)

Declaring global constants (Page 4852)

Declaring the block interface (Page 4771)

11.1.4.5 Addressing operands

Addressing global variables

Addressing global variables

To address a global PLC variable, you can use the absolute address or the symbolic name.

Note

The LWORD, LINT, ULINT, LREAL, LTIME, LTOD and LDT data types can only be addressed by means of their symbolic name.

Addressing global variables in symbolic form

When you use addressing in symbolic form, you enter the variable name from the PLC tag table. The symbolic name of global variables are automatically enclosed in quotation marks.

You address structured tags that are based on a PLC data type with the symbolic name of the PLC tag. You can also indicate the names of the individual components separated by a dot.

Addressing global variables in absolute form

When you use addressing in absolute form, you enter the address of the variables from the PLC tag table. The absolute address uses numerical addresses starting with zero for each operand range. The address identifier % is set automatically as prefix for the absolute address of global tags.

Examples

The following examples show applications of symbolic and absolute addressing:

Addressing	Explanation
%Q1.0	Absolute address: Output 1.0
%I16.4	Absolute address: Input 16.4
%IW4	Absolute address: Input word 4
"Motor"	Symbolic address "Motor"
"Value"	Symbolic address "Value"
"Structured_Tag"	Symbolic address of a tag that is based on a PLC data type
"Structured_Tag".Component	Symbolic address of the component of a structured tag.

See also

Displaying symbolic and absolute addresses (Page 4696)

Permissible addresses and data types of PLC tags (Page 4840)

Accessing I/O devices (Page 2148)

Accessing I/O devices

Description

The process image of the CPU is updated once in a cycle. In time-critical applications, however, it can be that the current state of a digital input or output has to be read or transferred more often than once per cycle. For this purpose you can use a suffix for I/O access identifiers on the operand to directly access the I/O.

If you want to read the input directly from the peripherals, use the peripheral inputs memory area (PI) instead of the process image input (I). The peripherals memory area can be read as a bit, byte, word, or double word.

If you want to write directly to the output, use the peripheral output (PQ) memory area instead of the process image output (Q). The peripheral output memory area can be written as a bit, byte, word, or double word.

To read or write a signal directly from a peripheral input, you can add the suffix for I/O access ":P", to the operand.

Components of structured PLC tags can also be addressed with ":P". However, access to the higher-level tag with ":P" is not possible.

WARNING

Direct writing of the I/O

Immediate writing to the I/O can lead to hazardous states, for example when writing multiple times to an output in one program cycle.

Note

S7-1200/1500: Transferring peripheral inputs or outputs as block parameters

If you supply an input parameter with a peripheral input or output, it is possible that I/O access errors will occur during runtime when the block is called, e.g. a read error when data of an input module is accessed directly.

The system response of the CPUs of the S7-1200/1500 series is as follows:

The block is not called as a result of the I/O access error. Program execution is continued after the block call. If OB 122 exists or local error handling is enabled, these are executed.

To prevent the situation that the block is not called as a result of an I/O access error, first copy the peripheral input or output to a local tag (Temp) and transfer it as a block parameter to the called block.

Syntax

```
<Operand>:P
```

Example

The following example shows applications of I/O access identifiers:

Addressing	Description
"Motor"	Addresses the "Motor" tag in the process image.
"Motor":P	Addresses the "Motor" tag in the I/O memory area (PI or PQ).
"Structured_Tag".Component	Addresses the component of a structured PLC tag in the process image.
"Structured_Tag".Component:P	Addresses the component of a structured PLC tag in the I/O memory area (PI or PQ).

See also

Addressing global variables (Page 2147)

Addressing variables in data blocks

Addressing variables in global data blocks

Description

Tags in global data blocks can be addressed in symbolic or absolute form. For symbolic addressing, you use the name of the data block and the name of the tag, separated by a dot. The name of the data block is enclosed in quotation marks.

For absolute addressing, you use the number of the data block and the absolute address of the tags in the data block, separated by a dot. The address identifier % is set automatically as prefix for the absolute address.

The S7-1200/1500 provides you with an option of accessing a data block that is not yet known during programming. For this purpose, create a block parameter of data type DB_ANY in the block interface of the accessing block. The data block name or data block number is transferred to this parameter during runtime. In order to access the internal tags of the data block, use the name of the block parameter of data type DB_ANY and the absolute address of the tag, separated by a dot.

Note

Transfer DB with memory reserve to the parameter DB_ANY.

It is not possible to transfer a DB with memory reserve to a block parameter of data type "DB_ANY".

Note

Addressing DB tags in absolute form

Absolute addressing is not possible for the following tags:

- Tags in blocks with optimized access.
- Tags of data type LWORD, LINT, ULINT, LREAL, LTIME, LTOD and LDT.

Best practice is to use the more convenient symbolic addressing for these tags.

ARRAY data blocks

ARRAY data blocks are a particular type of global data block. These consist of an ARRAY of any data type. For example, an ARRAY of a PLC data type (UDT) is possible.

You address elements in ARRAY data blocks with the help of the keyword "THIS". The index is then specified in square brackets. The index can be a constant as well as a tag. Integers with a width of up to 32 bits are permitted as tags for the index.
Extended options for addressing ARRAY DBs are available in the "Move" section of the "Instructions" task card. These instructions give you the option, for example, to also address the DB name indirectly.

Syntax

```
"<DBname>".TagName
%<DBnumber>.absoluteAddress
#<DBAny_name>.%absoluteAddress
"<ArrayDBname".THIS[#i].<Component>.<ComponentElement>
```

SCL:

```
"<ArrayDBname"."THIS"[#i].<Component>.<ComponentElement>
```

The following table show the possible absolute addresses of tags in data blocks:

Data type	Absolute address	Example	Description
BOOL	%DBn.DBXx.y	%DB1.DBX1.0	Data bit 1.0 in DB1
BYTE, CHAR, SINT, USINT	%DBn.DBBy	%DB1.DBB1	Data bit 1 in DB1
WORD, INT, UINT	%DBn. DBWy	%DB1.DBW1	Data word 1 in DB1
DWORD, DINT, UDINT, REAL, TIME	%DBn.DBDy	%DB1.DBD1	Data double word 1 in DB1

Example

The following examples show the addressing of tags in global data blocks:

Addressing	Description
"Motor".Value	Symbolic addressing of the "Value" tag in the "Motor" global data block.
%DB1.DBX1.0	Absolute addressing of the "DBX1.0" tags in the "DB1" global data block.
#MyDBAny.%DBX30.0	Absolute addressing of the "DBX30.0" tag in the global data block that is transferred at runtime at the "MyDBany" parameter.
"MyARRAY_DB".THIS[#MyIndex].MyComponent.MyComponentElement	Addressing an ARRAY data block. The ARRAY index is specified with the "MyIndex" tag. The ARRAY element has two additional substructures: "MyComponent" and "MyComponentElement".

See also

Using the DB_ANY data type (Page 2218)

Addressing individual characters of a STRING or WSTRING (Page 2276)

Addressing PLC data types (UDT) (Page 2281)

Addressing STRUCT components (Page 2285)

Addressing ARRAY components (Page 2292)

Addressing areas of a tag with slice access (Page 2154)

Basics of indirect addressing (Page 2158)

Addressing instance data (Page 2152)

Addressing instance data

Description

You can address data elements from the interface of the current block. These tags are stored in the instance data block.

Note

Tags in blocks with optimized access can only be addressed in symbolic form.

To address a tag from the interface of the current block, enter the character # followed by the symbolic tag name.

You can also access the tags of a multi-instance block. Within the multi-instance block, also use the character # followed by the tag name to address the data. You access the data of the multi-instance block from the calling block using #<Multi-instanceName.TagName>.

Syntax

Use the following syntax for addressing tags in instance data blocks:

```
#<tag name>
#<multi-instance name>.<tag name>
```

Examples

The following examples show the addressing of tags in instance data blocks:

Addressing	Description
#Value	Addressing the "Value" tag in the instance data block.
#On	Addressing the "On" tag within the multi-instance block
#Multi.On	Addressing the "On" tag of the multi-instance block from the calling block

See also

Addressing variables in global data blocks (Page 2150)

Addressing ARRAY components (Page 2292)

Addressing areas of a tag with slice access (Page 2154)

Basics of indirect addressing (Page 2158)

Addressing ARRAY DB (S7-1500)

Addressing ARRAY data blocks

ARRAY data blocks are a particular form of the ARRAY. ARRAY data blocks are global data blocks that consist of exactly one ARRAY. You address elements in ARRAY data blocks using the following syntax:

```
 "<GlobArrayDBname>".THIS[#i].<elementname>
```

SCL:

```
"<GlobArrayDBname>"."THIS"[#i].<elementname>
```

Examples

The following examples show the addressing of elements in an ARRAY DB:

Addressing in SCL	Explanation
"My_ARRAY_DB"."THIS"[0]	"My_ARRAY_DB" is an ARRAY DB, which contains components of the REAL data type. The first component within "My_ARRAY_DB" is addressed here.
"MY_UDT_ARRAY_DB"."THIS"[3].MyREAL	"MY_UDT_ARRAY_DB" is an ARRAY DB, which contains components of a PLC data type (UDT). The PLC data type contains elements of the BYTE data type. In the "MY_UDT_ARRAY_DB", the fourth ARRAY component and the "MyREAL" element in it are addressed here.

Instructions for addressing ARRAY DBs

The "Instructions > Basic instructions" task card in the "Move > ARRAY DB" section offers extended options for addressing ARRAY DBs. These instructions give you the option, for example, to address the DB name indirectly:

- ReadFromArrayDB: Read from array data block
- WriteToArrayDB: Write to array data block
- ReadFromArrayDBL: Read from array data block in load memory
- WriteToArrayDBL: Write to array data block in load memory

See also

Basics of indirect addressing (Page 2158)

Indirect indexing of ARRAY components (Page 2164)

Example of the use of ARRAY data blocks (Page 2092)

Global ARRAY data blocks (DB) (Page 2092)

ARRAY (Page 2286)

Addressing areas of a tag with slice access (S7-1200, S7-1500)

Description

You have the option to specifically address areas within declared tags. You can access areas of the 1-bit, 8-bit, 16-bit, or 32-bit width. The division of a memory area (e.g. BYTE or WORD) into a smaller memory area (e.g. BOOL) is also referred to as a "slice".

Structures, constants and tags overlaying AT cannot be addressed with slice access.

Syntax

The following syntax is used for addressing:

```
<Tag>.X<Bit number>
<Tag>.B<BYTE number>
<Tag>.W<WORD number>
<Tag>.D<DWORD number>
```

The syntax has the following components:

Part	Description
<Tag>	Tag that you access. The tag must be of the "Bit string" or "Integer" data type. With SCL, you can only program slice access to tags of the "Integer" data type when the IEC check is disabled.
X B W D	ID for the access width "Bit (1Bit)" ID for the access width "Byte (8 Bit)" ID for the access width "Word (16 Bit)" ID for access width "DWord (32-bit)"
<BIT number>	Bit number within <tag> that is accessed. Number 0 accesses the least significant BIT.
<BYTE number>	Byte number within <tag> that is accessed. The number 0 accesses the least significant BYTE.
<WORD number>	Word number within <tag> that is accessed. The number 0 accesses the least significant WORD.
<DWORD number>	DWord number within <tag> that is accessed. The number 0 accesses the least significant DWORD.

Examples of slice access

The following examples show the addressing of slices in bit, byte, word and double word modes:

Addressing	Description
"Engine".Motor.X0 "Engine".Motor.X7	"Motor" is a tag of the BYTE, WORD, DWORD or LWORD data type in the global data block "Engine". X0 addresses the bit address 0, X7 the bit address 7 within "Motor".
"Engine".Speed.B0 "Engine".Speed.B1	"Speed" is a tag of the WORD, DWORD or LWORD data type in the global data block "Engine". B0 addresses the byte address 0, B1 the byte address 1 within "Speed".
"Engine".Fuel.W0 "Engine".Fuel.W1	"FUEL" is a tag of the DWORD or LWORD data type in the global data block "Engine". W0 addresses the word address 0, W1 the word address 1 within "Fuel".
"Engine".Data.D0 "Engine".Data.D1	"Data" is a tag of the LWORD data type in the global data block "Engine". D0 addresses the double word address 0, D1 the double word address 1 within "Data".

Programming example

You can find a detailed example in the Siemens Industry Online Support:

https://support.industry.siemens.com/cs/ww/de/view/57374718 (https://support.industry.siemens.com/cs/ww/en/view/57374718)

Overlaying tags with AT

Description

To access data areas within a declared tag, you can overlay the declared tags with an additional declaration. This provides you with the option of addressing an already declared tag with a different data type. You can, for example, address the individual bits of a tag of WORD data type with an ARRAY of BOOL.

Alternatively, you can also use the instructions "SCATTER" and "GATHER" to break down bit sequences into an ARRAY of BOOL or group together individual bits to form a bit sequence. You can find these instructions in the "Basic instructions" task card in the "Move" area.

Rules

The following general rules are valid for tag overlaying:

- Overlaying is possible in S7-1200 and S7-1500 in STL, LAD, FBD and GRAPH.
- SCL supports overlaying in all CPU families.

- Overlaying of tags is possible in the following blocks:
  - In code blocks with standard access
  - In code blocks with optimized access for tags with the retain setting "Set in IDB"
- The data width of the overlaying tag must be equal to or less than that of the overlaid tag.
- It is not possible to overlay tags of the VARIANT and INSTANCE data types.
- Blocks from libraries which are declared as parameters in the interface cannot be overlaid.
- Structured PLC tags that are declared as parameters in the interface cannot be overlaid.
- You cannot address overlaying tags with slice access.

Note

S7-1200/1500: Using AT in FCs

The data widths of the overlaying tag and the overlaid tag must be identical for FCs in S7-1200/1500. If this is not possible in your program, check to see if you can use slice access instead of the AT construct, or use the instructions "SCATTER" or "GATHER".

See also: Addressing areas of a tag with slice access (Page 2154)

The following combination rules are also valid:

		Overlaying tag	Overlaid tag			
Elementary	Structured *	Any/Pointer	DB_ANY			
FB	Input	Elementary	x	x		x
		Structured *	x	x	x	x
		Any/Pointer		x		
	Temp	Elementary	x	x		
		Structured	x	x	x	
		Any/Pointer		x		
	Static, Output	Elementary	x	x		x
		Structured	x	x		x
		Any/Pointer				
	InOut	Elementary	x			x
		Structured		x		
		Any/Pointer				
FC	Temp	Elementary	x	x		
		Structured	x	x	x	
		Any/Pointer		x		
	Input, Output, InOut	Elementary (both tags must have the same bit width)	x			x
		Structured		x		
		Any/Pointer				

		Overlaying tag	Overlaid tag			
OB	Temp	Elementary	x	x		
		Structured	x	x	x	
		Any/Pointer		x		

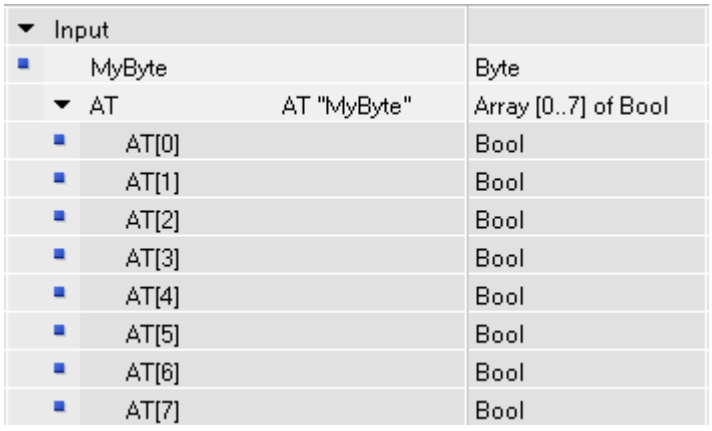
* Structured data types consist of several data elements, e.g. ARRAY or STRUCT.

Declaration

To overlay a tag, declare an additional tag directly after the tag that is to be overlaid and identify it with the keyword "AT".

Example

The following figure shows the declaration of an overlaid tag in the interface of a FB:



When a block is called with the shown tag declaration, the "MyByte" tag is assigned. Within the block there are now two options for interpreting the data:

- as a byte
- As one-dimensional ARRAY of BOOL

See also

Setting retentivity (Page 4809)

Addressing operands indirectly (S7-1200, S7-1500)

Indirect addressing (S7-1200, S7-1500)

Basics of indirect addressing (S7-1200, S7-1500)

Introduction

Indirect addressing offers you the option of addressing operands whose address is not calculated until runtime. With indirect addressing, you can also have program sections run several times and use a different operand during each run. For example, you can use a different index in each run in program loops.

This gives you the following advantages:

- You have more flexibility within your program.
- Indirect addressing is available to you in all STEP 7 programming languages.
- The program code is more straightforward.
- The existing names of data blocks and tags are used (symbolic addressing). This improves readability of the program code.

⚠ WARNING

Risk of access errors

Since operands are only calculated at runtime with indirect addressing, there is a risk that access errors may occur and that the program will operate with incorrect values. In addition, memory areas may inadvertently be overwritten with incorrect values. The automation system can then react in an unexpected manner.

Therefore, indirect addressing should be used with caution.

General indirect addressing options in S7-1200 and S7-1500

The following indirect addressing options are available in all programming languages:

- Indirect addressing via pointer
- Indirect indexing of ARRAY components
- Indirect addressing of a data block via DB_ANY data type.

Language-specific options of indirect addressing

The following specific addressing options are also available in the various programming languages:

- In STL, you can address operands indirectly via the address register.
- In SCL, you can read or write a variable memory area with the following instructions:
  - POKE - Write memory address
  - POKE_BOOL - Write memory bit
  - PEEK - Read memory address
  - PEEK_BOOL - Read memory bit
  - POKE_BLK - Write memory area

See also

Addressing variables in global data blocks (Page 2150)

POKE: Write memory address (Page 3335)

POKE_BOOL: Write memory bit (Page 3337)

PEEK: Read memory address (Page 3332)

PEEK_BOOL: Read memory bit (Page 3334)

POKE_BLK: Write memory area (Page 3339)

Indirect addressing via pointer (Page 2164)

Indirect indexing of ARRAY components (Page 2164)

Indirect addressing in STL (Page 2169)

Examples of indirect addressing (S7-1200, S7-1500)

1. Program example

In the example below, you use the index to access three tags from different memory areas.

Overview of the three tags that are assigned to one index each:

Index	Accessing tag	Memory area
1	Input_WORD_0	IW 0
2	"Processdata".Temperature	DB 1
3	Output_WORD_4	QW 4

Declare the following two tags in the "Default tag table":

Default tag table

		Name	Data type	Address
1		Input_WORD_0	Word	%IW0
2		Output_WORD_4	Word	%QW4

Create a global data block:

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Specify the name "DB_Processdata".
4. Select "Global DB" as the type of the data block.
5. Click "OK".
6. Declare the data block element "Temperature":

DB_Processdata

		Name	Data type	Start value
1		Static		
2		Temperature	DInt	50

Declare indirect access using an index within a function.

1. Create an SCL function and name it "FB_AccessGroupInt".
2. Declare the block interface as follows:

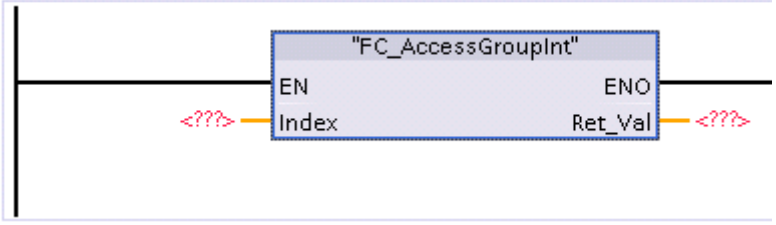
FC_AccessGroupInt

		Name	Data type
1		Input	
2		Index	Int
3		Output	
4		InOut	
5		Temp	
6		Constant	
7		Return	
8		FC_AccessGroupInt	DInt

3. Write the following program code:

```
1
2  CASE #Index OF
3
4      1: // First case: "Input_WORD_0"
5          #FC_AccessGroupInt := "Input_WORD_0";
6      2: // Second case: "DB_Processdata".Temperature
7          #FC_AccessGroupInt := "DB_Processdata".Temperature;
8      3: // Third case: "Output_WORD_4"
9          #FC_AccessGroupInt := "Output_WORD_4";
10
11 END_CASE;
```

4. Call the "FC_AccessGroupInt" function in OB1:



Depending on which number (1, 2 or 3) you specify at the Index parameter, the first, second or third case of the "FC_AccessGroupInt" instruction is executed.

2. Program example

In the example below, you use the index to access three different optimized data blocks.

Since all data blocks should contain the same tags, you can use a PLC data type (UDT) in this case.

1. To create a PLC data type, double-click the "Add new data type" command in the "PLC data types" folder in the project tree.
   A new declaration table for creating a PLC data type will be created and opened.
2. Rename the PLC data type to "UDT_SiloContents".
3. Declare the following lines within the PLC data type:
   MyBool > Data type: BOOL
   MyInt > Data type: INT
   MyWord > Data type: WORD

UDT_SiloContents			
		Name	Data type
1	...	MyBool	Bool
2	...	MyInt	Int
3	...	MyWord	Word

Create three global data blocks.

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Specify the names "DB_SiloWater", "DB_SiloSugar" and "DB_SiloMilk".
4. Select the data blocks "UDT_SiloContents" as the type of the data blocks.
5. Click "OK".

DB_SiloWater

	Name	Data type
1	Static	
2	MyBool	Bool
3	MyInt	Int
4	MyWord	Word

DB_SiloSugar

	Name	Data type
1	Static	
2	MyBool	Bool
3	MyInt	Int
4	MyWord	Word

DB_SiloMilk

	Name	Data type
1	Static	
2	MyBool	Bool
3	MyInt	Int
4	MyWord	Word

Create a function to read the values of the data block tags and to write these to a PLC data type.

1. Create an SCL function and name it "FC_AccessGroupSiloRead".
2. Declare the block interface as follows:

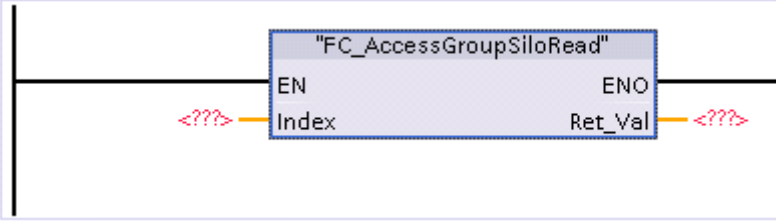
FC_AccessGroupSiloRead

		Name	Data type
1		Input	
2		Index	DInt
3		Output	
4		InOut	
5		Temp	
6		Constant	
7		Return	
8		FC_AccessGroupSiloRead	"UDT_SiloContents"

3. Write the following program code:

```
1
2  CASE #Index OF
3
4      1: // First case: "DB_SiloWater"
5          #FC_AccessGroupSiloRead := "DB_SiloWater";
6      2: // Second case: "DB_SiloSugar"
7          #FC_AccessGroupSiloRead := "DB_SiloSugar";
8      3: // Third case: "DB_SiloMilk"
9          #FC_AccessGroupSiloRead := "DB_SiloMilk";
10
11     ELSE // Optional case
12         #FC_AccessGroupSiloRead := "DB_SiloWater";
13
14 END_CASE;
```

4. Call the "FC_AccessGroupSiloRead" function in OB1:



Depending on which number (1, 2 or 3) you specify at the Index parameter, the first, second or third case of the "FC_AccessGroupSiloRead" instruction is executed.

Indirect addressing via pointer (S7-1200, S7-1500)

Description

For indirect addressing, a special data format is required that contains the address and possibly also the range and the data type of an operand. This data format is referred to as pointer. The following types of pointers are available to you:

- POINTER (S7-1500)
- ANY (S7-1500, only for blocks with standard access)
- VARIANT (S7-1200/1500)

For more information on the pointer data types, refer to "See also".

Note

In SCL the use of the POINTER is restricted. The only option available is to forward it to the called block.

Example

The following example shows an indirect addressing with an area-internal pointer:

Addressing in STL	Explanation
L P#10.0	// Load pointer (P#10.0) in accumulator 1
T MD20	// Transfer pointer to the operand MD20
L MW [MD20]	// Load MW10 in accumulator 1
....	// Any program
L MD [MD20]	// Load MD10 in accumulator 1
....	// Any program
= M [MD20]	// If RLO=1, set the memory bit M10.0

The pointer P#10.0 is transferred to the operand MD20. If the operand MD20 in square brackets is programmed, this will be replaced in runtime by the address that is contained in the pointer.

See also

Basics of indirect addressing (Page 2158)

Pointer (Page 2302)

Indirect indexing of ARRAY components (S7-1200, S7-1500)

ARRAY access with variable index

For addressing ARRAY elements, you can specify either constants or tags of the integer data type as index. Integers with a length of up to 32 bits are allowed here.

With indirect addressing using a tag, the index is only calculated during program runtime. You can, for example, use a different index for each cycle in program loops.

Syntax

The following syntax is used for the indirect indexing of a ARRAY:

```
"MyDB".MyArray[#i]   // One-dimensional ARRAY
"MyDB".MyArray[#i].a // One-dimensional ARRAY of STRUCT
"MyDB".MyArray[#i,#j]// Multi-dimensional ARRAY
"MyDB".MyArray[#i].a // Multi-dimensional ARRAY of STRUCT
```

The syntax has the following components:

Part	Description
MyDB	Name of the data block in which the ARRAY is located
MyArray	Tag of the ARRAY data type
i, j	PLC tags of the integer data type that are used as pointers
a	Additional partial tag of the structure

Note

When you call a block and transfer an indirectly indexed ARRAY component ("MyDB".MyArray[#i]) to it as in/out parameter (InOut), you cannot change the value of the index tag [i] while the block is being executed. The value is therefore always written back to the same ARRAY component from which it was read.

Indexing ARRAY components using the "FieldRead" and "FieldWrite" instructions

You may also use the following instructions for indirect indexing of ARRAY components in LAD and FBD:

- FieldWrite - Write field
- FieldRead - Read field

See also

Examples of indirect indexing of ARRAYs (Page 2166)

Indirect addressing of ARRAY components (Page 2215)

Basics of indirect addressing (Page 2158)

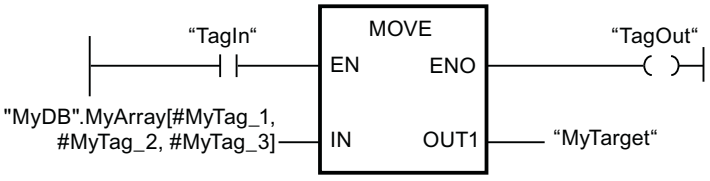
Addressing ARRAY components (Page 2292)

ARRAY (Page 2286)

Examples of indirect indexing of ARRAYs (S7-1200, S7-1500)

Example: ARRAY access with variable index in LAD

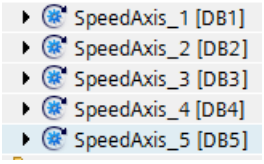
The following example shows the indirect indexing of an ARRAY component using LAD as example. "MyArray" is a three-dimensional ARRAY. #Tag_1, #Tag_2 and #Tag_3 are input parameters of the data type "Integer". Depending on their values, one of the MyArray components is copied to the "MyTarget" tag.



Example: ARRAY access with variable index in SCL

The following example shows you how to address multiple data blocks using a variable index. Five speed-controlled axes are used in the example. These axes are to be processed with SCL iteratively in a FOR loop.

1. First you create five speed-controlled axes. All five axes must be of the same type.
   A data block is created for each axis in the project navigator:



2. Next, generate a global data block and declare an ARRAY of the data type ANY with five components.

1		Static	
2		MyArray	Array[0..4] of DB_ANY
3		MyArray[0]	DB_ANY
4		MyArray[1]	DB_ANY
5		MyArray[2]	DB_ANY
6		MyArray[3]	DB_ANY
7		MyArray[4]	DB_ANY

3. For initialization, you assign the symbolic names of the five axes to the individual ARRAY elements in the startup OB.

```
"MyGlobalDB".MyArray[0] := "SpeedAxis_1";
"MyGlobalDB".MyArray[1] := "SpeedAxis_2";
"MyGlobalDB".MyArray[2] := "SpeedAxis_3";
"MyGlobalDB".MyArray[3] := "SpeedAxis_4";
"MyGlobalDB".MyArray[4] := "SpeedAxis_5";

```

4. In SCL, access to the individual axes takes place iteratively with the help of a FOR loop. The axes are transferred to the function "MaximumVelocity" for execution one after the other.

1	Input			☐	☐	☐	☐	
2	Output			☐	☐	☐	☐	
3	InOut			☐	☐	☐	☐	
4	CurrentMaximum	LReal	0.0	☑	☐	☑	☐	
5	Static			☐	☐	☐	☐	
6	Temp			☐	☐	☐	☐	
7	Index	DInt		☐	☐	☐	☐	
8	Constant			☐	☐	☐	☐	

IF... CASE... OF... FOR... TO DO... WHILE... DO... (*...*) REGION

```
FOR #Index := 0 TO 9 BY 1 DO
    #CurrentMaximum := "MaximumVelocity" (CurrentAxis:="MyGlobalDB".MyArray[#Index]);
END_FOR;

```

Example: ARRAY access with variable index in STL

An application example for ARRAY access with variable index in STL is available under "Addressing ARRAY elements indirectly". (Page 2215)

See also

Indirect indexing of ARRAY components (Page 2164)

Basics of indirect addressing (Page 2158)

Addressing ARRAY components (Page 2292)

ARRAY (Page 2286)

Indirect addressing of individual characters of a STRING (S7-1200, S7-1500)

Description

For addressing the individual characters of a STRING or WSTRING, you can specify constants and also tags as the index. The tags must be of the Integer data type. When tags are used, the index is calculated during runtime. You can, for example, use a different index for each cycle in program loops.

If you transfer a STRING or WSTRING with variable index to an in/out parameter during a block call, please note that: The index tag [i] is read once at the start of the block call and cannot be changed by the called block while it is being executed.

Note

Monitoring STRING access in runtime

When a STRING or WSTRING that exceeds the defined length is written in runtime, unwanted reactions may occur in the program. Violation of the STRING or WSTRING length is monitored in S7-1200/1500. On read access to the STRING, you receive the character '$00' or '$0000'; write access to the STRING is not executed. If the instruction has the enable output ENO, ENO is set to the signal state FALSE. The CPU does not change to STOP.

Syntax

The following syntax is used for the indirect indexing of a STRING or WSTRING:

```
"<Data block>".<STRING>["i"]
"<Data block>".<WSTRING>["i"]
```

Example

The example below shows indirect indexing of a STRING using SCL as an example. "STRING", "WSTRING", "CHAR" and "WCHAR" are tags. "Tag_1" is a PLC tag of the "Integer" data type.

Addressing in SCL	Description
`STRING["Tag_1"] := CHAR;`	`(*Indirect addressing: Assignment of "CHAR" to the character of the STRING*) specified by "Tag_1"`
`WSTRING["Tag_1"] := WCHAR;`	`(*Indirect addressing: Assignment of "WCHAR" to the character of the WSTRING*) specified by "Tag_1"`
`WCHAR := WSTRING["Tag_1"];`	`(*Indirect addressing: Assignment of the WSTRING character specified by "Tag_1" to WCHAR*)`

See also

STRING (Page 2270)

WSTRING (Page 2273)

Indirect addressing in STL (S7-1500)

Basic information about address registers (S7-1500)

Introduction

Two address registers are available for the indirect addressing of operands: address register 1 (AR1), and address register 2 (AR2). The address registers are equal and are 32 bits in length. You can store area-internal and cross-area pointers in the address registers. To define the address of an operand, you can call the stored data in the program.

Data is exchanged between the registers and the other available memory areas with the assistance of load and transfer instructions.

Note

In S7-1500, special rules apply to data exchange via address register and data block register:

- The values in the registers do not remain in existence beyond the block limits.
- The registers are reset when the language is changed within a block.
- You can only reference data in blocks with optimized access if these have the retain setting "Set in IDB".
- It is not possible to reference local data in blocks with optimized access with the help of the address registers (across areas).

See also

Indirect addressing in STL (Page 2169)

Addressing areas of a tag with slice access (Page 2154)

Setting retentivity (Page 4809)

Indirect addressing in STL (S7-1500)

In STL, the following options are available for indirect addressing:

- Memory-indirect addressing
- Register-indirect area-internal addressing
- Register-indirect cross-area addressing

Memory-indirect addressing

In the case of memory-indirect addressing, you store the address in a tag. The tag can be of WORD or DWORD data type. The tag can be located in the memory areas "Data" (DB or DI), "Bit memory" (M) or "Temporary local data" (L). In S7-1500, FB parameters can also be used to store the address. If the tag is located in a data block, it must be a data black with standard access.

The following example shows applications of memory-indirect addressing:

Addressing in STL	Explanation
`U E [MD 2]`	`// Execute an AND logic operation with a variable input bit. The address of the input bit is located in the memory double word MD2.`
`= DIX [DBD 2]`	`// Assign the RLO to a variable data bit. The address of the data bit is located in the data double word DBD2.`
`L EB [DID 4]`	`// Load a variable input byte to ACCU 1. The address of the input byte is located in the instance double word DID4.`
`AUF DB [LW 2]`	`// Open a variable data block. The number of the data block is located in the local data word LW2.`

Register-indirect area-internal addressing

Register-indirect addressing uses one of the address registers (AR1 or AR2) to pick up the address of the operand.

In the case of register-indirect, area-internal addressing, you index only the bit address and the byte address via the address register (e.g. `P#10.0`). You do not enter the memory area for which the address in the address register is to apply until during programming of the instruction. The address in the address register then moves to the memory area specified in the instruction.

Possible memory areas are "Inputs" (I), "Outputs" (Q), "I/O" (PI or PQ), "Bit memory" (M), "Temporary local data" (L) and "Data" (DB or DI). If the operand is located in a data block, it must be a data black with standard access.

When you enter register-indirect, area-internal addressing, specify an offset after the specification of the address register. This offset is added to the contents of the address register without changing the address register. This offset also has the format of a pointer. The specification of a pointer is mandatory and must be entered as constant (e.g. `P#0.0` or `P#2.0`).

The following example shows an application of register-indirect area-internal addressing:

STL	Explanation
`LAR1 P#10.0`	`// Load pointer (P#10.0) to address register 1`
`L IW [AR1, P#2.0]`	`// Increase contents of address register 1 (P#10.0) by offset P#2.0.`
	`// Load contents of input word IW12 into accumulator 1`
`L IW [AR1, P#0.0]`	`// Increase contents of address register 1 (P#10.0) by offset P#0.0.`
	`// Load contents of input word IW10 into accumulator 1`

Register-indirect cross-area addressing

In the case of register-indirect, cross-area addressing, use the address register to index the entire address of the operand, in other words, the bit address and byte address, as well as the memory area. Possible memory areas are "Inputs" (I), "Outputs" (Q), "I/O" (P), "Bit memory" (M), "Temporary local data" (L) and "Data" (DB or DI). If the operand is located in a data block, it must be a data black with standard access or the operand must have the retain setting "Set in IDB".

In the instruction, program only the operand width. Possible operand widths are bit, byte, word, and double word.

The following example shows an application of register-indirect cross-area addressing:

```
LAR1 P#M10.0          // Load cross-area pointer (P#M10.0) to address register 1
L W [AR1, P#2.0]      // Increase contents of address register 1 (P#M10.0) by offset
                      P#2.0.
                      // Load contents of memory word "MW12" into accumulator 1
LAR1 P#A10.0          // Load cross-area pointer (P#A10.0) to address register 1
L W [AR1, P#2.0]      // Add contents of address register 1 (P#A10.0) by offset P#2.0
                      // Load contents of output word QW12.0 into accumulator 1
```

Note

Special features in S7-1500

In S7-1500, special rules apply to data exchange via address register and data block register:

- The values in the registers do not remain in existence beyond the block limits. The registers are also reset when the language is changed within a block.
- If you access an operand of the BYTE, WORD or DWORD type using register-indirect addressing, the address must begin at a byte limit.
  Examples:
  LAR1 P#0.0
  L MW [AR1, P#0.0] // P#0.0 + P#0.0 = P#0.0 - The addressing is allowed, because P#0.0 points to a byte limit.
  L MW [AR1, P#2.1] // P#0.0 + P#2.1 = P#2.1 - The addressing is not allowed, because P#2.1 does not point to a byte limit.

See also

Basics of indirect addressing (Page 2158)

Addressing ARRAY components (Page 2292)

Basic information about address registers (Page 2169)

11.1.5 Handling program execution errors

11.1.5.1 Causes of error

Introduction

There are different causes for various error qualities in TIA Portal to which you can respond with different mechanisms. These mechanisms with which you can react to an error are independent of the programming languages. Only the display of the individual mechanisms depends on the respective programming language.

The situation determines what counts as an error. The overflow in the case of an addition, for example, can be an error because it does not return the desired result. In some cases, however, the addition overflow is acceptable and defined behavior and therefore not an error.

When creating a program code, you must be aware that certain situations may arise. When you program communication, for example, you must be aware that this connection can be canceled at any time. To be prepared for this situation, you must install a corresponding fault reaction because any connection abort prevents the T_SEND program block from transmitting a message. This is why it is important that the connection abort is signaled to inform the operator that the message can no longer be transmitted. Because the instruction T_SEND cannot prevent the connection abort, the error output is the correct behavior for T_SEND. Ignoring this output can be interpreted as an error made by the programmer.

Below, we will always refer to an error even if it is defined behavior.

Different causes for an error

We can distinguish between the following causes for an error:

	Faulty parameter values	Faulty programming	Failure of resources
Description	Errors that are dealt with directly in the instruction.	Programming or access errors that result in the execution of an instruction being aborted.	Errors that are handled by the operating system and for which you can program a reaction in the program code.
Error quality	Handled error	Synchronous error	Asynchronous error
Example	Overflow of an arithmetic instruction	Faulty programming: • Querying a non-existent peripheral input • When accessing an ARRAY by means of a variable index, the value of the index is located outside the valid ARRAY limits	Occurrence of a specific event outside the program code.

	Faulty parameter values	Faulty programming	Failure of resources
Reaction of the program or operating system	Instructions that are linked with the enable output ENO are not executed.	If you have not programmed an error OB, the operating system response depends on the CPU.	1. If an organization block (OB) is not assigned to the event, the operating system executes the default system response when the event occurs. 2. If an organization block (OB) is assigned to the event, it is called.
Mechanisms for error handling in the program	Depending on the instruction, you have different local error handling options [2]: • EN/ENO mechanism • The output parameters: – RET_VAL – STATUS – ERROR	Global error handling [1]: • Program execution error OB • Programming error OB • I/O access error OB Local error handling [2] with the instructions: • GET_ERROR • GET_ERR_ID	Error organization blocks (OB): • Time error (OB 80) • Diagnostic interrupt (OB 82) • Insert/remove module interrupt (OB 83) • Rack error (OB 86) Possible system responses without assigned error OB: • The event is ignored by the operating system. • The CPU changes to STOP mode. • If possible, the error handling takes place locally. If an error OB has been assigned, it will be called in case of a corresponding event. An application example for diagnostics in the user program With evaluation of the errors in the error OBs (https://support.industry.siemens.com/cs/document/98210758?lc=en-WW)

[1] You implement global error handling with the help of organization blocks.

[2] You program local error handling within your program code.

Note

Asynchronous error handling

For a CPU of the S7-1500 series, the error OBs are called asynchronously. This means I/O access errors or programming error OBs may not be processed immediately when the error occurs, but delayed depending on the set priority. If additional errors occur before the I/O access error or the programming error OB was processed completely, no additional I/O access error or programming error OB is called. If you want to prevent I/O access or programming error OBs from being discarded, set the priority correspondingly high.

See also

Example of how to handle program execution errors (Page 2203)

11.1.5.2 Overview of mechanisms for error handling

Overview

There are different mechanisms by which you can track parameter, programming or access errors:

Mechanism	Task	Error handling
Enable input EN or IF instruction	Prevent execution of a program code	Local
Enable output ENO or binary result bit	Indicate an error	
Parameter outputs RET_VAL, STATUS and ERROR		
GET_ERROR and GET_ERR_ID instructions	Respond to an error	
Organization blocks		Global

Local error handling in the case of faulty parameter values

Not only does local error handling let you react to an error immediately after it has occurred, you can also get a specific reaction within your program code. You program the local error handling directly in a program block (OB, FB or FC). It only handles errors that occur within this individual program block.

Advantages of local error handling:

- You can use the error information to program a reaction to the error that occurred in the program block.
- Programmed error evaluation and error reactions do not interrupt the program cycle.
- System performance is not unnecessarily affected by local error handling. If no errors occur, programmed error analyses and reactions are not executed.

The following local error handling options are available:

Error handling method	Validity	Explanation
EN/ENO mechanism [1]	S7-300 / S7-400 / S7-1200 / S7-1500	You can detect and handle specific runtime errors with the help of the enable output ENO. Execution of subsequent instructions depends on the signal state of the enable output. You can avoid program crashes by using the EN/ENO mechanism. The block status is passed on in the form of a Boolean tag. For additional information on the EN/ENO mechanism, refer to: Basics of the EN/ENO mechanism (Page 2177)
Output parameters STATUS and ERROR	S7-300 / S7-400 / S7-1200 / S7-1500	Using the STATUS and ERROR parameters as return values of system function blocks (SFBs), you can query block-specific error information. The error information is output in a pre-defined structure. For further information on the output parameters, refer to the descriptions of the individual instructions in the information system.
Output parameter RET_VAL	S7-300 / S7-400 / S7-1200 / S7-1500	You can display general error codes or specific error codes with the help of the output parameter RET_VAL as return value of sequential functions (SFCs). The general error codes refer to any instruction and the specific error codes refer only to a specific instruction. A maximum of one tag of the data type INT or WORD can be output. Additional information on the output parameter RET_VAL is available here: Evaluating errors with output parameter RET_VAL (Page 2198)
[1] If the parameters of an instruction did not cause any memory access errors, the associated enable output ENO returns the signal state "1" and the outputs return valid values which you can query.		

Global and local error handling in the case of faulty programming

With the global and local error handling, you can react immediately to errors that have occurred without the CPU changing to "STOP" mode. The following options are available for handling programming and access errors:

Type of global error handling	Validity	Explanation
Program execution error OB (OB 85)	S7-300 / S7-400	If you are not using OB 85, the CPU changes from RUN mode to STOP mode in case of a program execution error and generates an entry in the diagnostics buffer. Additional information on OB 85 can be found here: AUTOHOTSPOT
CPU internal error handling in case of programming and access errors	S7-1200	In the event of an error, the CPU generates an entry in the diagnostics buffer and remains in RUN mode without additional programming effort.
Programming error OB (OB 121)	S7-300/ S7-400 / S7-1500	If you are not using OB 121, the CPU changes from RUN mode to STOP mode in the case of a programming error and generates an entry in the diagnostics buffer. Additional information on OB 121 can be found here: S7-300 / S7-400: AUTOHOTSPOT S7-1500: AUTOHOTSPOT

Type of global error handling	Validity	Explanation
I/O access error OB (OB 122)	S7-300/ S7-400 / S7-1500	S7-300 / S7-400: If you are not using OB 122, the CPU changes from RUN mode to STOP mode in the case of an access error. AUTOHOTSPOT S7-1500: In the case of an I/O access error, the CPU always remains in RUN mode and generates an entry in the diagnostics buffer, even if you have not used OB 122. Additional information on OB 122 can be found here: AUTOHOTSPOT
Additional information on reading out the diagnostics buffer can be found here:AUTOHOTSPOT		

You can integrate local error handling directly in your program code with the help of the GET_ERROR and GET_ERR_ID instructions. You also have the option of receiving detailed information on the error and evaluating it in your program close to the error location. You program the local error handling directly in a program block (OB, FB or FC). It only handles errors that occur within this individual program block.

Type of local error handling	Validity	Explanation
GET_ERROR and GET_ERR_ID instructions	S7-1200/ S7-1500	The instructions give you the opportunity to obtain an error ID or detailed error information and to program a direct response in the program code. When information about the first error is queried, the memory space of the error in the system memory is enabled again. Then, when additional errors occur, information about the next error is output. Additional information on the GET_ERROR and GET_ERR_ID instructions and an overview for prioritizing the errors that occur is available here: Use of the instructions GET_ERROR and GET_ERR_ID (Page 2202)

When using local error handling, query via the GET_ERROR instruction, the following default reactions exist:

- In the case of a write error: The error is ignored and processing of the program continues.
- In the case of a read error: Processing of the program continues with the substitute value "0" for arithmetic instructions.
- In the case of an execution error: Processing of the instruction is canceled and processing of the program continues with the next instruction.

Advantages of local error handling:

- Error information that you can query and evaluate is stored in the system memory (e. g. with the help of the GET_ERROR and GET_ERR_ID instructions).
- You can use the error information to program a reaction to the error that occurred in the program block.
- Programmed error evaluation and error reactions do not interrupt the program cycle.

- Local error handling has a lower effect than global error handling on the system performance. If no errors occur, programmed error analyses and reactions are not executed.
- If local error handling has been set for a program block, no global error handling is conducted in the case of errors.

Note

To prevent the CPU from switching to STOP mode in the case of an error, all programming and I/O access errors must be captured either by global or local error handling.

Example

A detailed example of local error handling that contains several of the above described options is available here: AUTOHOTSPOT

See also

Example of how to handle program execution errors (Page 2203)

11.1.5.3 EN/ENO mechanism

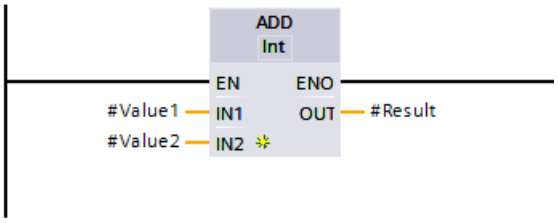
Basics of the EN/ENO mechanism

Introduction

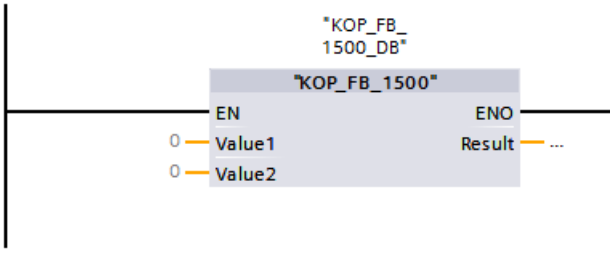
You can detect and handle certain runtime errors using the enable output ENO. Execution of subsequent instructions depends on the signal state of the enable output. You can avoid program crashes by using the EN/ENO mechanism. The block status is passed on in the form of a Boolean tag.

The EN/ENO mechanism can be used at two levels:

- For individual instructions (instruction ENO)



- For program block calls (block ENO)



The EN/ENO mechanism is already available in LAD and FBD for calling the simple and advanced instructions in the program code.

With the EN/ENO mechanism you can influence the calling of the following instructions and customize the jumping out from a program block using the instruction "RET: Return". At the same time you can also assign an individual value (0 or 1) to the enable output ENO of the program block. This behavior is generally used for LAD and FBD program blocks. You can also use this procedure for SCL program blocks. You do not necessarily have to use the instruction "RET" to influence the enable output ENO of the SCL program block.

At the network borders the enable output is repeatedly set to the signal state "1". In a LAD program block you can recognize this, for example, by the fact that the left busbar always supplies current, even when the enable output ENO of the last instruction of the previous network returns the signal state "0".

Function of the instruction "RET: Return" (LAD/FBD)

When the result of logic operation = 0 the instruction is not executed and the next network is processed.

When the result of logic operation = 1 the instruction is executed and a return to the calling program block takes place.

The signal state of the block ENO can be determined with four options:

- RLO: RLO = 1 is applied, i.e. the block ENO is set to TRUE.
- TRUE: The block ENO is set to TRUE.
- FALSE: The block ENO is set to FALSE.
- Operand: The signal state of the specified operand determines the block ENO.

Programming and I/O access errors

You cannot evaluate programming and I/O access errors with the EN/ENO mechanism. To do this, use either the global error handling via OBs or the local error handling using the "GET_ERROR" or "GET_ERR_ID" instructions (S7-1200/1500 only). If none of the errors occurred for an instruction, you can evaluate the associated enable output ENO.

Program block calls in all programming languages (S7-300/400)

The following applies to a CPU of the S7-300/400 series: When you call program blocks that do not include any instructions, neither the BR bit nor the ENO enable output are influenced. The signal state of the BR bit remains constant as a result. No statement can be made about the success of the program block call.

EN/ENO mechanism is LAD

Overview of the EN/ENO mechanism in LAD

To increase the performance, the EN/ENO mechanism is disabled by default for the instructions. It can be enabled at any time also for individual instructions. For additional information on the enabling and disabling of the EN/ENO mechanism, refer to: Enabling and disabling the EN/ENO mechanism in LAD and FBD (Page 2180)

For LAD/FBD program block the EN/ENO mechanism only works with an activated enable output ENO at the respective instructions.

EN/ENO mechanism within an instruction.

You can use the enable input EN to make the execution of the instruction dependent on conditions. The instruction is only executed if the signal state at the enable input EN is "1".

You can use the enable output ENO to query runtime errors in an instruction and react to these:

- The enable output ENO has the signal state "1" if no error occurred during the execution.
- Enable output ENO has the signal state "0" if one of the following conditions applies:
  - Enable input EN has the signal state "0".
  - An error occurred during processing.

EN/ENO mechanism with program block calls

All program block calls are provided with an enable input EN and an enable output ENO. This applies to all calling program blocks, regardless of the programming language in which they were created. This means that when you call an STL or SCL program block in which there is no default and pre-configured EN/ENO mechanism, the LAD or FBD program block call still has an EN/ENO mechanism.

You can use the enable input EN to call a program block depending on conditions. The program block is only executed if the signal state at the enable input EN is "1".

You can query the error status of the program block using the enable output ENO:

- The enable output ENO returns the signal state "1" when the program block is processed.
- If, in the case of error, you do not explicitly set the enable output ENO within the called program block to the signal state "0", it retains the signal state "1". Using the instruction "RET: Return" you can set the block ENO to the signal state "0".
  Additional information on using the RET instruction is available here: Influencing the block ENO of an LAD/FBD program block (Page 2181)

Enabling and disabling the EN/ENO mechanism in LAD and FBD

In LAD and FBD, certain instructions have an enable output ENO and thus use the EN/ENO mechanism. This allows you to query runtime errors in instructions and react to them. In order to increase the performance of the CPU, the EN/ENO mechanism is disabled in the default setting. This means that you are not initially able to react to runtime errors of the instruction using the ENO value. However, you can enable the EN/ENO mechanism again at any time, if required.

You can enable the EN/ENO mechanism individually for each instruction in order to generate the ENO. If you enable the EN/ENO mechanism for an instruction, other instructions that you subsequently add to your program are also inserted with the EN/ENO mechanism enabled. You can disable the EN/ENO mechanism again at any time if you do not want to use the evaluation of ENO for an instruction. Further instructions that you subsequently add to your program will then be inserted without the EN/ENO mechanism.

Runtime errors do not cause the CPU to go to STOP when the enable output ENO is enabled.

Activating the EN/ENO mechanism

Proceed as follows to activate the EN/ENO mechanism of an instruction:

1. In your program, right-click the instruction at which you want to activate the EN/ENO mechanism.
2. Select the "Generate ENO" command from the shortcut menu.
   The ENO value is again generated for the instruction. Additional instructions are then also inserted with the activated enable output.

Deactivating the EN/ENO mechanism

Proceed as follows to deactivate the EN/ENO mechanism of an instruction:

1. In your program, right-click the instruction at which you want to deactivate the EN/ENO mechanism.
2. Select the "Do not generate ENO" command from the shortcut menu.
   The ENO value is no longer generated for the instruction. Additional instructions are then also inserted without the activated enable output.

See also

Basics of the EN/ENO mechanism (Page 2177)

Influencing the block ENO of an LAD/FBD program block

Description

Using the instruction "RET: Return" you jump out from a program block and hereby influence the signal state of the ENO block.

Procedure

To influence the signal state of the block ENO, follow these steps:

1. Enable the EN/ENO mechanism of an instruction (for example, the addition "ADD").
2. Then use a negation to program the instruction "RET: Return" with signal state FALSE at the enable output ENO.

Result

In case of error (for example, an overflow in the result), the signal state "0" is first returned at the enable output ENO. This signal state "0" becomes signal state "1" using the negation and thus forms the RLO = 1 for executing the instruction "RET", which returns the value FALSE. The block ENO of the program block is hereby FALSE and a return takes place from the program block to the next instruction after the previous program block call. This procedure can be programmed in each network (for example, with multiple mathematical functions etc.).

However, it is not compulsory to program a jump out of the program block. Within a network you can also use the signal state "0" of the enable output ENO of a single instruction to determine that the subsequent instruction is not executed. The block ENO is not hereby influenced.

Note

Influencing of the block ENO

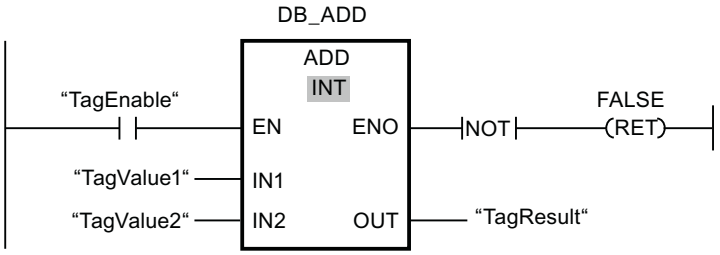
You can only influence the signal state of the block ENO by jumping out of the program block.

Even if the enable output ENO of the last instruction in the last network within a program block has the signal state "0", the block ENO is not influenced.

Example of the use of the EN/ENO mechanism in LAD

Example of an instruction with EN/ENO mechanism

The following example shows the "Add" instruction with EN/ENO circuit and a RET coil (Ret False):



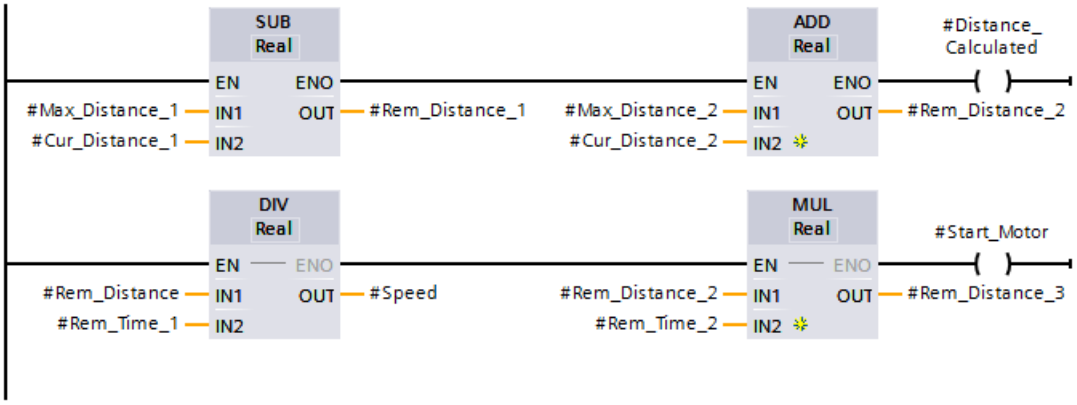
After the NO contact "TagEnable", the enable input EN contains the result of the preceding logic operation:

- If the operand "TagEnable" has the signal state "0", the "Add" instruction is not executed. The enable output ENO is set to signal state "0" and the called program block is exited. The enable output ENO of the calling program block then also has the signal state "0".
- If the operand "TagEnable" has the signal state "1", the enable input "EN has the signal state "1" and the instruction "Add" is executed. The instruction adds two values of the data type INT. If the expected result is outside the value range of INT (16 bits: -32768 to +32767), the instruction still returns a result but it is within the value range of INT. The reason for this is that the 16th bit for INT is the sign bit. This result does not indicate that there was an overflow. Which is why the following applies in case of an addition => ENO:= NOT(OV). If an error occurs during processing of the program block, the enable output ENO is set to signal state "0" and the called program block is exited. The enable output ENO of the calling program block then also has the signal state "0".
- If the operand "TagEnable" has the signal state "1", the enable input "EN has the signal state "1" and the instruction "Add" is executed. If the instruction is executed without errors, the enable output ENO also has the signal state "1" and the output "TagResult" the result.

A detailed description of the instruction "RET: Return" is available here: --(RET): Return (Page 2827)

Example of the effect of the enable output ENO

The following example shows you how to use instructions with the ENO enable output enabled and disabled:

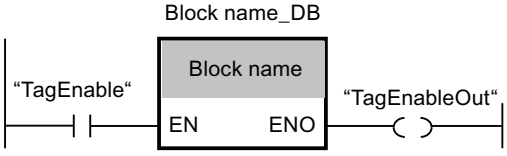


If you have activated the ENO enable output, for example with the SUB instruction, all subsequent instructions are also applied with an activated ENO enable output. If an arithmetic error occurs during the execution of the SUB instruction, the ADD instruction is not executed.

The ENO enable output is disabled for the DIV instruction in the second branch. If a runtime error occurs during execution, the MUL instruction is executed anyway.

Example of a program block call with EN/ENO mechanism

The following example shows the call of the program block with EN/ENO circuit:



The program block is executed only if the operand "TagEnable" has the signal state "1":

- The enable output ENO has a signal state that depends on what was programmed within the program block.
- If the operand "TagEnable" has the signal state "0", the program block call is not executed. The enable input EN and the enable output ENO both have the signal state "0".

See also

Basics of the EN/ENO mechanism (Page 2177)

EN-/ENO mechanism in LAD

Overview of the EN/ENO mechanism in LAD

To increase the performance, the EN/ENO mechanism is disabled by default for the instructions. It can be enabled at any time also for individual instructions. For additional information on the enabling and disabling of the EN/ENO mechanism, refer to: Enabling and disabling the EN/ENO mechanism in LAD and FBD (Page 2185)

For LAD/FBD program block the EN/ENO mechanism only works with an activated enable output ENO at the respective instructions.

EN/ENO mechanism within an instruction.

You can use the enable input EN to make the execution of the instruction dependent on conditions. The instruction is only executed if the signal state at the enable input EN is "1".

You can use the enable output ENO to query runtime errors in an instruction and react to these:

- The enable output ENO has the signal state "1" if no error occurred during the execution.
- Enable output ENO has the signal state "0" if one of the following conditions applies:
  - Enable input EN has the signal state "0".
  - An error occurred during processing.

EN/ENO mechanism with program block calls

All program block calls are provided with an enable input EN and an enable output ENO. This applies to all calling program blocks, regardless of the programming language in which they were created. This means that when you call an STL or SCL program block in which there is no default and pre-configured EN/ENO mechanism, the LAD or FBD program block call still has an EN/ENO mechanism.

You can use the enable input EN to call a program block depending on conditions. The program block is only executed if the signal state at the enable input EN is "1".

You can query the error status of the program block using the enable output ENO:

- The enable output ENO returns the signal state "1" when the program block is processed.
- If, in the case of error, you do not explicitly set the enable output ENO within the called program block to the signal state "0", it retains the signal state "1". Using the instruction "RET: Return" you can set the block ENO to the signal state "0".
  Additional information on using the RET instruction is available here: Influencing the block ENO of an LAD/FBD program block (Page 2185)

Enabling and disabling the EN/ENO mechanism in LAD and FBD

In LAD and FBD, certain instructions have an enable output ENO and thus use the EN/ENO mechanism. This allows you to query runtime errors in instructions and react to them. In order to increase the performance of the CPU, the EN/ENO mechanism is disabled in the default setting. This means that you are not initially able to react to runtime errors of the instruction using the ENO value. However, you can enable the EN/ENO mechanism again at any time, if required.

You can enable the EN/ENO mechanism individually for each instruction in order to generate the ENO. If you enable the EN/ENO mechanism for an instruction, other instructions that you subsequently add to your program are also inserted with the EN/ENO mechanism enabled. You can disable the EN/ENO mechanism again at any time if you do not want to use the evaluation of ENO for an instruction. Further instructions that you subsequently add to your program will then be inserted without the EN/ENO mechanism.

Runtime errors do not cause the CPU to go to STOP when the enable output ENO is enabled.

Activating the EN/ENO mechanism

Proceed as follows to activate the EN/ENO mechanism of an instruction:

1. In your program, right-click the instruction at which you want to activate the EN/ENO mechanism.
2. Select the "Generate ENO" command from the shortcut menu.
   The ENO value is again generated for the instruction. Additional instructions are then also inserted with the activated enable output.

Deactivating the EN/ENO mechanism

Proceed as follows to deactivate the EN/ENO mechanism of an instruction:

1. In your program, right-click the instruction at which you want to deactivate the EN/ENO mechanism.
2. Select the "Do not generate ENO" command from the shortcut menu.
   The ENO value is no longer generated for the instruction. Additional instructions are then also inserted without the activated enable output.

See also

Basics of the EN/ENO mechanism (Page 2177)

Influencing the block ENO of an LAD/FBD program block

Description

Using the instruction "RET: Return" you jump out from a program block and hereby influence the signal state of the ENO block.

Procedure

To influence the signal state of the block ENO, follow these steps:

1. Enable the EN/ENO mechanism of an instruction (for example, the addition "ADD").
2. Then use a negation to program the instruction "RET: Return" with signal state FALSE at the enable output ENO.

Result

In case of error (for example, an overflow in the result), the signal state "0" is first returned at the enable output ENO. This signal state "0" becomes signal state "1" using the negation and thus forms the RLO = 1 for executing the instruction "RET", which returns the value FALSE. The block ENO of the program block is hereby FALSE and a return takes place from the program block to the next instruction after the previous program block call. This procedure can be programmed in each network (for example, with multiple mathematical functions etc.).

However, it is not compulsory to program a jump out of the program block. Within a network you can also use the signal state "0" of the enable output ENO of a single instruction to determine that the subsequent instruction is not executed. The block ENO is not hereby influenced.

Note

Influencing of the block ENO

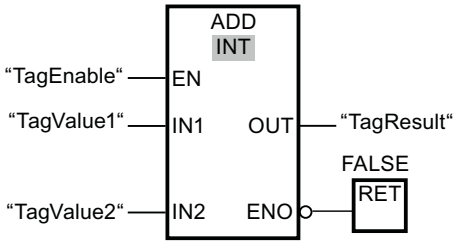
You can only influence the signal state of the block ENO by jumping out of the program block.

Even if the enable output ENO of the last instruction in the last network within a program block has the signal state "0", the block ENO is not influenced.

Example of the use of the EN/ENO mechanism in FBD

Example of an instruction with EN/ENO mechanism

The following example shows the "Add" instruction with EN/ENO circuit and a RET coil (Ret False):



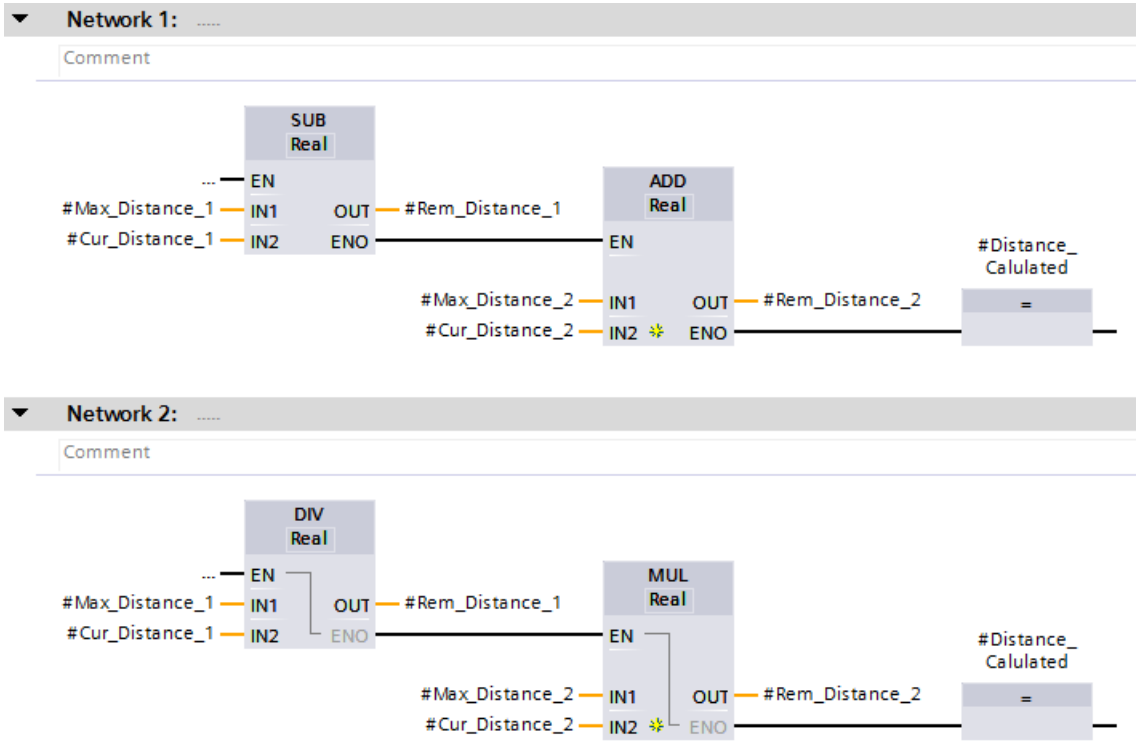
After the NO contact "TagEnable", the enable input EN contains the result of the preceding logic operation:

- If the operand "TagEnable" has the signal state "0", the "Add" instruction is not executed. The enable output ENO is set to signal state "0" and the called program block is exited. The enable output ENO of the calling program block then also has the signal state "0".
- If the operand "TagEnable" has the signal state "1", the enable input "EN has the signal state "1" and the instruction "Add" is executed. The instruction adds two values of the data type INT. If the expected result is outside the value range of INT (16 bits: -32768 to +32767), the instruction still returns a result but it is within the value range of INT. The reason for this is that the 16th bit for INT is the sign bit. This result does not indicate that there was an overflow. Which is why the following applies in case of an addition => ENO:= NOT(OV). If an error occurs during processing of the program block, the enable output ENO is set to signal state "0" and the called program block is exited. The enable output ENO of the calling program block then also has the signal state "0".
- If the operand "TagEnable" has the signal state "1", the enable input "EN has the signal state "1" and the instruction "Add" is executed. If the instruction is executed without errors, the enable output ENO also has the signal state "1" and the output "TagResult" the result.

A detailed description of the instruction "RET: Return" is available here: RET: Return (Page 3152)

Example of the effect of the enable output ENO

The following example shows you how to use instructions with the ENO enable output enabled and disabled:

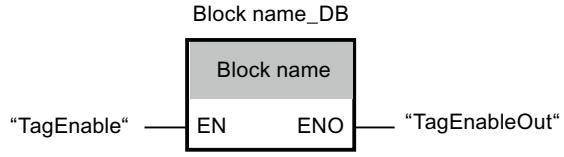


If you have activated the ENO enable output, for example with the SUB instruction, all subsequent instructions are also applied with an activated ENO enable output. If an arithmetic error occurs during the execution of the SUB instruction, the ADD instruction is not executed.

The enable output ENO is disabled for the instruction DIV. If a runtime error occurs during execution, the MUL instruction is executed anyway.

Example of a program block call with EN/ENO mechanism

The following example shows the call of the program block with EN/ENO circuit:



The program block is executed only if the operand "TagEnable" has the signal state "1":

- The enable output ENO has a signal state that depends on what was programmed within the program block.
- If the operand "TagEnable" has the signal state "0", the program block call is not executed. The enable input EN and the enable output ENO both have the signal state "0".

See also

Basics of the EN/ENO mechanism (Page 2177)

EN/ENO mechanism in STL (S7-1500)

Overview of the EN/ENO mechanism in STL (S7-1500)

EN/ENO mechanism within an instruction.

The EN/ENO mechanism is not available for individual instructions. It is mapped by language-specific instruction sequences and use of the status word (BR bit).

For additional information on the status word, refer to: AUTOHOTSPOT

Additional information is available here: Example of the simulation of the EN/ENO mechanism in STL (Page 2189)

EN/ENO mechanism with program block calls

A program block that you call from an STL program block is not provided with the EN and ENO parameters. Regardless of the programming language in which the program block was created, you can get an error statement to the calling STL program block by using the BR bit of the status word.

You can evaluate the error status of the called program block by linking the BR bit of the status word with the result of logic operation (RLO). The RLO has the signal state "1" as soon as the

program block execution of the called program block starts. If you do not explicitly reset the RLO to "0" after the execution of the actions, it retains the signal state "1". To signal an error statement back to the calling program block, you have to explicitly reset the RLO to the signal state "0". The error statement is set with the instructions "SAVE: Save RLO in BR bit" and "JNB: Jump if RLO = 0 and save RLO".

An error analysis of the BR bit is available in STL. The BR bit is the basis for generating the enable output ENO in the programming languages LAD, FBD and SCL.

The value "0" in the BR bit of the status word indicates that an error occurred during an STL program block call (calling program block is an STL program block).

CPU execution of the instruction	BR bit	Return value	Sign of the integer
With error(s)	0	Less than "0"	Negative (sign bit is "1")
Without error	1	Greater than or equal to "0"	Positive (sign bit is "0")

For additional information on the BR bit, refer to: Evaluating errors with output parameter RET_VAL (Page 2198)

Example of the simulation of the EN/ENO mechanism in STL (S7-1500)

Example of a program sequence with EN/ENO mechanism

The following example shows a program section for adding values with EN/ENO mechanism:

STL	Description
A "TagEnable"	// Query the signal state of the "TagEnable" operand for "1" and AND with the current RLO.
JNB MyLABEL	// Evaluation of the enable input EN
	// If RLO = "0", jump to jump label "MyLABEL" and save the current RLO in the BR bit. The following actions are not executed.
	// If the RLO = "1", execute the following actions.
L "Tag_Input_1"	// Load first value for addition.
L "Tag_Input_2"	// Load second value for addition.
+I	// Add values
T "Tag_Result"	// Transfer sum to the operand "Tag_Result".
AN OV	// Query whether an overflow has occurred.
SAVE	// Transfer signal state of the RLO to the BR bit.
CLR	/// Reset RLO to "0" and end logic sequence.
MyLABEL: A BR	// Jump label "MyLABEL"
	// Query BR bit.

STL	Description
= "Tag_Output"	// Assign signal state of the RLO to the "Tag_Output" operand.

The query of the operand "TagEnable" provides the result of the preceding logic operation (RLO). The instruction "JNB: Jump if RLO = 0 and save RLO" saves the RLO in the BR bit. Additionally, the instruction evaluates the RLO and executes the following actions depending on the signal state:

- If the RLO is "0", the processing of the program is continued at the jump label "MyLABEL" with the query of the BR bit. The addition is not executed. The current RLO is assigned with the operand "Tag_Output".
- If the RLO is "1", the addition is executed. A query of the overflow bit (OV) shows whether an error occurred during the addition. The query result is saved in the BR. The instruction "CLR: Reset RLO to 0" resets the RLO to "0" and ends the logic sequence. The BR bit is then queried and assigned to the operand "Tag_Output". The signal state of the BR bit, and the operand "Tag_Output" shows whether the addition was executed without any error.

Example of a program block call with EN/ENO mechanism

The following example shows the call of the program block with EN/ENO circuit:

STL	Description
A "TagEnable"	// Query the signal state of the "TagEnable" operand for "1" and AND with the current RLO.
JNB MyLABEL	// Evaluation of the enable input EN
	// If RLO = "0", jump to jump label "MyLABEL" and save the current RLO in the BR bit. The following actions are not executed.
	// If the RLO = "1", execute the following actions.
CALL "Block name", "Block name_DB"	// The program block is called.
MyLABEL: A BR	// Jump label "MyLABEL"
	// Query BR bit and link with the RLO after AND.
= "Tag_Output"	// Assign signal state of the RLO to the "Tag_Output" operand.

See also

Basics of the EN/ENO mechanism (Page 2177)

EN/ENO mechanism in SCL

Overview of the EN/ENO mechanism in SCL

To increase performance, the EN/ENO mechanism is disabled by default. However, it can be enabled at any time using the block attribute "Set ENO automatically". For additional information on the enabling and disabling of the EN/ENO mechanism, refer to: Enabling and disabling the ENO mechanism in SCL (Page 2191)

In addition to enabling the block attribute "Set ENO automatically", you can also influence the enable output EN using an IF instruction. Additional information is available here: Influencing enable output ENO using an IF instruction (Page 2193)

EN/ENO mechanism within an instruction.

The EN/ENO mechanism is optional for instructions.

Additional information is available here: Using enable output ENO with SCL instructions (Page 2192)

EN/ENO mechanism with program block calls

The call of an SCL program block is also executed unconditionally, regardless of whether the EN/ENO mechanism is enabled or not. However, you can link the call to an IF instruction and then make the call dependent on various conditions. One possible condition can, for example, be the block ENO of the SCL program block. However, the EN/ENO mechanism then has to be enabled for this.

If the block ENO returns the signal state "0" because of a runtime error in a previous instruction, the block call is not executed.

Additional information is available here: Influencing enable output ENO of an SCL program block (Page 2193)

Enabling and disabling the ENO mechanism in SCL

Description

The EN/ENO mechanism checks during runtime whether errors occur in the processing of certain instructions. If a runtime error occurs the block ENO is set to "0".

Enabling EN/ENO mechanism in the block properties

Proceed as follows to activate the EN/ENO mechanism in the block properties:

1. Open the "Program blocks" folder in the project navigator.
2. Right-click the SCL block whose properties you want to display.
3. Select the "Properties" command in the shortcut menu.
   The properties dialog box of the block opens.
4. Click the "Attributes" group in the area navigation.

5. Activate the property "Set ENO automatically".
6. Confirm your entries with "OK".

Result: The EN/ENO mechanism is activated only for the selected SCL program block.

Activate EN/ENO mechanism in the program properties

Proceed as follows to activate the EN/ENO mechanism in the program properties:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the working area.
2. In the area navigation, select the "PLC programming" group.
3. Select the group "SCL (Structured Control Language)".
4. Activate the property "Set ENO automatically".

Result: The EN/ENO mechanism is activated by default for all new SCL program blocks.

Using enable output ENO with SCL instructions

Description

With each SCL instruction, such as a mathematical function, you must query the enable output ENO in order to be able to work with it.

Procedure

To query the enable output ENO, proceed as follows:

1. Activate the EN/ENO mechanism.
2. After the SCL instruction, query the enable output ENO (for example, #MyOutputBool := ENO;)
   If the error case occurs, for example an overflow in the result, returns the enable output ENO returns the signal state "0". With this signal state you can continue to work and, for example, program the instruction "RETURN", a fault display or a substitute value.
3. Reset the signal state of the enable output ENO to "1" (for example, ENO :=1;) before processing the next instruction.
   The signal state for the next error case can hereby be reset to "0". If you do not do this, the signal state of the enable output ENO remains "0".

Note

Jump to an SCL block.

With a jump to an SCL block, the signal state of the enable output ENO is automatically set to "1".

Influencing enable output ENO of an SCL program block

Description

The last instruction ENO within an SCL program block always influences the block ENO.

In addition, you can jump out of the program block and at the same time assign the value "1" or "0" to the block ENO using the instruction "RETURN".

Procedure

To influence the signal state of the block ENO, follow these steps:

1. Activate the EN/ENO mechanism.
2. Then program:

```
IF ENO := 0 THEN
RETURN;
END_IF;
```

Result

In case of error (for example, an overflow in the result), the signal state "0" is initially returned at the enable output ENO, such as an addition instruction. The block ENO is hereby FALSE and a jump takes place from the program block to the next instruction after the previous program block call. This procedure can be programmed multiple times in an SCL program block (for example, with multiple mathematical functions, etc.).

However, it is not compulsory to program a jump out of the program block. Within the SCL program block you can also use the signal state "0" of the enable output ENO of a single instruction to determine that the subsequent instruction is not executed. For this purpose, the instruction ENO is linked with the following instruction in a subsequent IF instruction.

Note

Influencing of the block ENO

With an SCL program block the signal state of the last instruction ENO is forwarded at the end of the block to the block ENO

The instruction ENO thus directly influence the block ENO.

Influencing enable output ENO using an IF instruction

Description

If you influence the enable output EN using an IF instruction, you do not need to activate the block attribute "Set ENO automatically".

Procedure

To influence the ENO enable output using an IF instruction, proceed as follows:

1. Program an IF instruction.
2. Use the enable output ENO either within the IF instruction or with ELSE.

Result

Depending on a Boolean value or a value comparison, the enable output ENO within the IF instruction is set to signal state "1" (ENO := 1;) or with ELSE to the signal state "0" (ENO := 0;).

Example of the use of the EN/ENO mechanism in SCL

Example of an instruction with EN/ENO mechanism

To use the EN/ENO mechanism for instructions, you have to activate the block property "Set ENO automatically". The following example shows the use of the enable output ENO for the operation "a + b":

```
"MyOutputREAL":= #a + #b;
"MyOutputBOOL":= ENO;
```

If the operation "a/b" is executed error-free, the operand "MyOutputBool" has the signal state "1". In case of error, the enable output ENO has the signal state "0".

Example of a program block call with EN/ENO mechanism

The following example shows the call of the program block. For this purpose the EN/ENO mechanism must be activated and an instruction that influences the block ENO must be programmed beforehand:

```
IF ENO := 1 THEN
"Block name_DB"();
END_IF;
```

The program block is only called and executed if the enable output ENO has the signal state "1". The called program block also returns a block ENO if its EN/ENO mechanism is activated. This block Eno is then further used in the called program block.

See also

Basics of the EN/ENO mechanism (Page 2177)

EN/ENO mechanism in GRAPH (S7-1500)

Overview of the EN/ENO mechanism in GRAPH (S7-1500)

EN/ENO mechanism within an instruction.

You have no access to the enable output ENO of the instructions, which means you cannot influence the status of the enable output ENO at the GRAPH function block. However, with conversions, mathematical functions or LAD/FBD instructions, for example, you can have the enable output ENO displayed in the program status.

EN/ENO mechanism with program block calls

You can use the enable input EN to call a program block depending on conditions. The program block is only executed if the signal state at the enable input EN is "1".

You can query the error status of the program block using the enable output ENO:

- The enable output ENO has the signal state "1", as soon as the called program block has been executed error-free.
- The enable output ENO has the signal state "0" if an error occurred during the execution of the called program block.
- The enable output ENO cannot be explicitly set or reset.
- The enable output ENO is not influenced by the ENO of the instructions.

Activating and deactivating EN/ENO mechanism in GRAPH (S7-1500)

Description

The status of the enable output ENO is displayed during testing with program status. This has the value TRUE if the action was successful and FALSE if the action failed.

The display option of the status of the enable output ENO is available at the following positions in the GRAPH program block:

- Permanent pre-instructions
- Sequence view > Actions
- Permanent post-instructions

Enabling EN/ENO mechanism in the block properties

Proceed as follows to activate the EN/ENO mechanism in the block properties:

1. Open the "Program blocks" folder in the project navigator.
2. Right-click the GRAPH block whose properties you want to display.
3. Select the "Properties" command in the shortcut menu.
   The properties dialog box of the block opens.

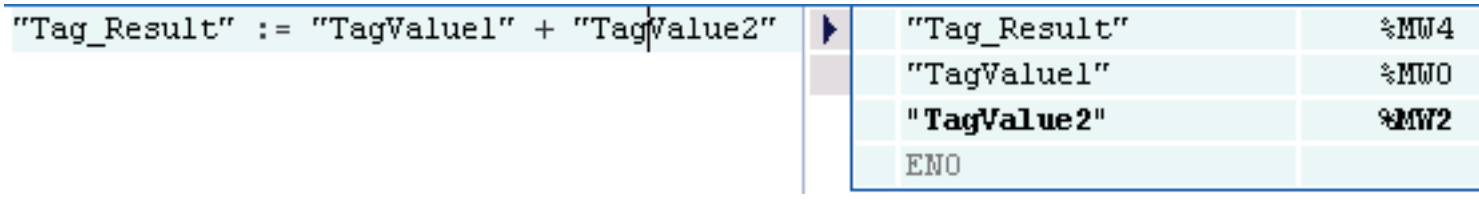
4. Click the "Attributes" group in the area navigation.
5. Activate the property "Set ENO automatically".
6. Confirm your entries with "OK".

Result: The EN/ENO mechanism is activated only for the selected GRAPH block.

Example of the EN/ENO mechanism in GRAPH (S7-1500)

Example of a program status with enable output ENO

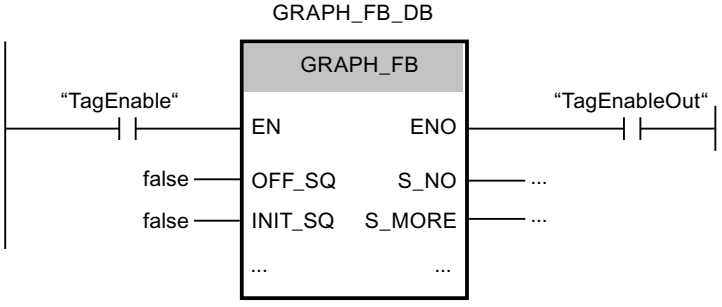
The following example shows the program status of the enable output ENO in the sequence view under actions:



If an error occurs during the execution, the enable output ENO has the signal state FALSE.

Example of a program block call

The following example shows the call of the GRAPH program block in a LAD program block with EN/ENO connected:



The program block is executed only if the operand "TagEnable" has the signal state "1":

- If the operand "TagEnable" has the signal state "0", the called program block call is not executed. Both the enable input EN and the enable output ENO have the signal state "0".
- If the operand "TagEnable" has the signal state "1", the enable input "EN has the signal state "1" and the called program block is executed. The enable output ENO has a signal state that depends on what was programmed within the program block.

See also

Basics of the EN/ENO mechanism (Page 2177)

EN/ENO mechanism in blocks with different network languages (S7-1200, S7-1500)

Description

You can also use the EN/ENO mechanism in blocks with different network languages. Each programming language displays the error status differently:

- SCL has an ENO tag that stores the error status and can be queried. Direct access to this tag is only possible with SCL.
- LAD/FBD/STL have no special tag for ENO. However, you can read the error status for STL from the BR bit, and query it via the RET coil for LAD/FBD.

The following rules apply to reading out the error status for the entire block:

- The last network in the block is a LAD/FBD network:
  If you do not use RET coils, the error status is TRUE by default.
- The last network in the block is an STL network:
  The BR bit determines the error status. The BR bit can be edited in STL networks using the BR tab.
- The last network in the block is an SCL network:
  The ENO tag determines the error status of the block.

Influencing of the block ENO

For LAD/FBD program blocks with multiple networks that could consist of different programming languages (such as LAD, FBD, STL or SCL), the last processed network determines the signal state of the block ENO.

If the last processed network of such a program block is a LAD or FBD network, the signal state of the block ENO is not necessarily influenced. With an instruction in which the EN/ENO mechanism is activated, it is only influenced if the enable output ENO has the signal state "0" and it is possible to return for the program block using the instruction "RET: Return". If this instruction was not programmed, the block ENO always has the signal state "1". The signal state "0", if any, of an instruction ENO has no influence on the block ENO, as the signal state "1" is always set at the start of a network.

This applies also to a LAD or FBD network that is before another network (LAD, FBD or SCL). The signal state of the block ENO is influenced only using the instruction "RET: Return". Otherwise, the signal state of the block ENO is reset to "1" at the transition to the network.

If the last network of a mixed program block is an SCL network, the block ENO is always influenced by this.

See also

Basics of the EN/ENO mechanism (Page 2177)

11.1.5.4 Evaluating errors with output parameter RET_VAL

Basics on error analysis of library blocks (SFB and SFC)

Two additional error evaluation options are available in addition to the output parameter RET_VAL:

- With the EN/ENO mechanism (LAD, FBD, and SCL)
  For additional information on the EN/ENO mechanism, refer to: Basics of the EN/ENO mechanism (Page 2177)
- With the BR bit (binary result bit) of the status word (STL)
- With the output parameter RET_VAL (return value)

The enable output ENO only informs you that an error has occurred. If you want to know which error occurred exactly, the output parameter RET_VAL will provide more information. You can use it to determine whether or not the CPU was able to execute the instruction successfully. In the case of an error, you will also learn why the instruction could not be executed successfully.

The section below includes detailed information on the two error evaluation options RET_VAL and BR bit.

Recommendation for sequence of error analyses

Before evaluating the instruction-specific output parameters (for example, OUT), you should always follow the steps below:

1. First evaluate the enable output ENO or the BR bit of the status word in STL.
2. Then check the output parameter RET_VAL.

If the enable output ENO or the BR bit indicates that an error has occurred during the execution of the instruction or if output parameter RET_VAL contains a general error code, the instruction-specific output parameters return an invalid value.

If a general error occurs with the output parameter RET_VAL, this is only indicated by the value "0" in the BR bit of the status word. The return value is of the data type integer (INT). The relationship of the return value to the value "0" indicates whether or not an error occurred during execution of the instruction.

General and specific error codes (RET_VAL)

The output parameter RET_VAL has two types of error codes:

- A general error code that all instructions can output and
- A specific error code that an instruction can output depending on its specific functions.

The data type of the output parameter RET_VAL is an integer (INT). The error codes of the instruction are grouped according to hexadecimal values. If you want to examine a return value and compare the value with the error codes listed in this documentation, then display the error code in hexadecimal format.

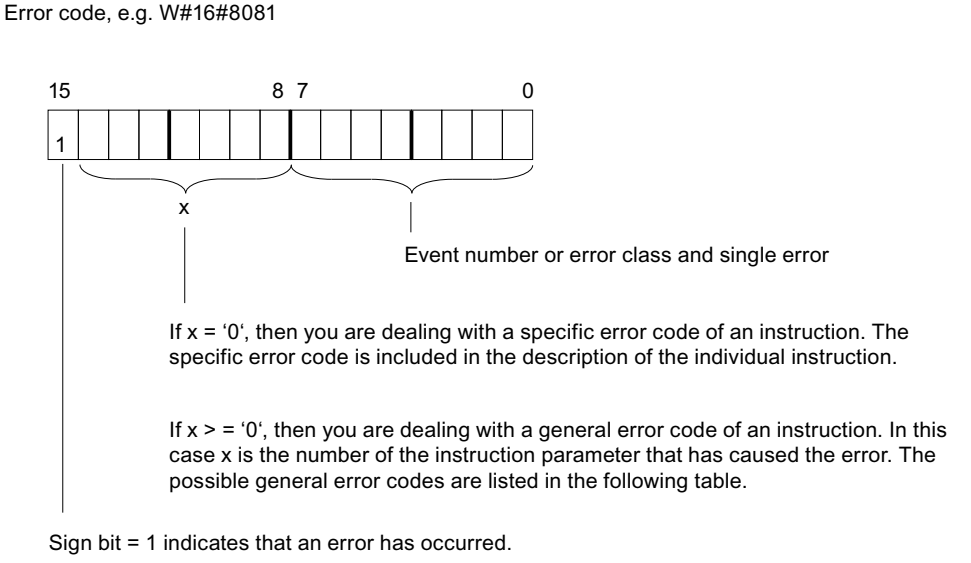
You can write your program so that it reacts to the errors that occur during execution of an instruction. This way you prevent further errors occurring as a result of the first error.

Note

Error during supply of the input parameters

When the instruction containing the RET_VAL parameter is executed and errors during supply of the input parameters occur, the RET_VAL parameter outputs an invalid error code and the output parameters of the instruction must not be evaluated.

The figure below shows the structure of a system function error codes in hexadecimal format:



It is easy for you to determine whether an error has occurred because the output parameter RET_VAL is of the type INT:

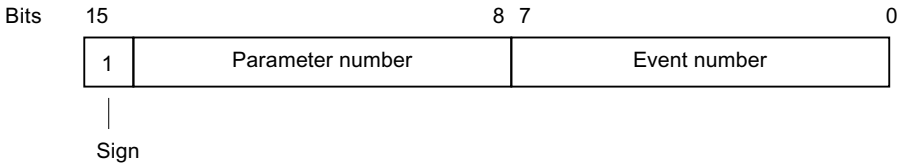
- If the value is < 0, there is an error.
- If the value is = 0, no error has occurred.
- If the value is > 0, no error has occurred but the instruction was not executed successfully. This is used in asynchronous instructions, for example, to indicate that processing of the instruction was started but is not yet complete. See the return values of T_SEND or WRIT_DBL.

General error codes

The general error code indicates errors that can occur in all instructions. A general error code consists of the following two numbers:

- A parameter number from 1 to 111, where 1 indicates the first parameter of the called instruction, 2 the second parameter, and so forth.
- An event number between 0 and 127. The event number indicates an error.

The figure below shows the structure of a general error code:



Note

Reaction to a general error code

If a general error code was entered in the output parameter RET_VAL, the following situations are possible:

- The action associated with the instruction was started or has already been completed.
- A specific instruction error occurred when the action was performed. As a result of a general error that occurred later, the specific error could, however, no longer be indicated.

The following table explains the general error codes of a return value. The error code is shown in hexadecimal format. The letter x in each code number is simply a placeholder and represents the number of the system function parameter that caused the error:

Error code (W#16#...)	Description
8x01	Illegal syntax ID at an VARIANT parameter
8x22	Range length error when reading a parameter.
8x23	Range length error when writing a parameter. This error code indicates that the parameter x is located either entirely or partly outside the range of an address, or that the length of a bit range is not a multiple of 8 with an VARIANT parameter.
8x24	Range error when reading a parameter.
8x25	Range error when writing a parameter. This error code indicates that the parameter x is located in a range that is illegal for the system function. Refer to the descriptions of the individual functions for information about the illegal ranges.
8x26	The parameter contains a timer cell number that is too high. This error code indicates that the timer cell specified in parameter x does not exist.
8x27	The parameter contains a counter cell number that is too high (counter number error). This error code indicates that the counter cell specified in parameter x does not exist.
8x28	Alignment error when reading a parameter.
8x29	Alignment error when writing a parameter. This error code indicates that the reference to parameter x is an operand with bit address that is not equal to 0.
8x30	The parameter is located in a read-only global DB.
8x31	The parameter is located in a read-only instance DB. This error code indicates that parameter x is located in a read-only data block. If the data block was opened by the system function itself, the system function always returns the value W#16#8x30.
8x32	The parameter contains a DB number that is too high (DB number error).

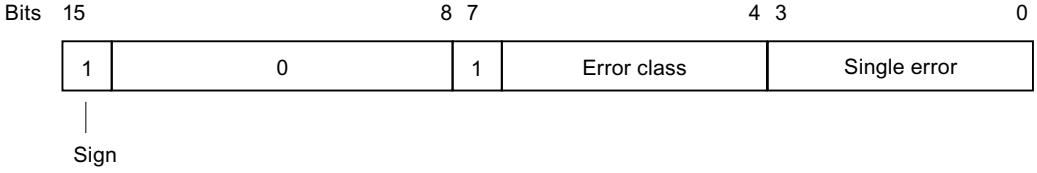
Error code (W#16#...)	Description
8x34	The parameter contains an FC number that is too high (FC number error).
8x35	The parameter contains an FB number that is too high (FB number error). This error code indicates that parameter x contains a block number higher than the highest permitted number.
8x3A	The parameter contains the number of a DB that is not loaded.
8x3C	The parameter contains the number of an FC that is not loaded.
8x3E	The parameter contains the number of an FB that is not loaded.
8x42	An access error occurred while the system was attempting to read a parameter from the peripheral input area.
8x43	An access error occurred while the system was attempting to write a parameter to the peripheral output area.
8x44	Error in the nth (n > 1) read access after an error occurred.
8x45	Error in the nth (n > 1) write access after an error occurred. This error code indicates that access to the required parameter is denied.
8x7F	Internal error This error code indicates an internal error at parameter x.

Specific error codes

Some instructions have a return value that provides an error code specific for the instruction. A specific error code indicates errors that can occur only in specific instructions.

A specific error code consists of the following two numbers:

- An error class from 0 to 7.
- An error number from 0 to 15.



For further information on specific error codes, refer to the descriptions of the individual instructions in the information system.

Additional information in the Siemens Industry Online Support

You can find more information on the error codes in the Siemens Industry Online Support under the following FAQ ID: 770453 (https://support.industry.siemens.com/cs/ww/en/view/770453)

See also

Difference between synchronous and asynchronous instructions (Page 2571)

11.1.5.5 Use of the instructions GET_ERROR and GET_ERR_ID

Introduction

Local error handling makes it possible to query the occurrence of errors within a program block and evaluate the associated error information. You can set local error handling for organization blocks (OBs), function blocks (FBs), and functions (FCs). If local error handling is enabled, the system reaction is ignored.

You can read out the error number from the error information of the instruction GET_ERR_ID. You can find out, for example, which parameter caused an access error from the error information of the instruction GET_ERROR. In order for the instructions to output the required error information, they must be programmed in the user program for each individual program block from which potential errors are to be evaluated. If you work with the instructions, no error OB is called and there is no entry in the diagnostics buffer. With this method of error handling, you actively intervene in the program sequence by programming a reaction to the errors that occurred. Because errors can occur at any location within the program block, we recommend that you integrate the instruction at the end of the program block.

The instructions GET_ERROR and GET_ERR_ID differ in the amount of error information that is output with each one.

As soon as you have integrated one of the two instructions in your program code, the "Handle errors within block" check box is selected in the inspector window under "Properties > Attributes". This setting cannot be edited in the Inspector window. Local error handling can be deactivated by deleting the inserted instructions for local error handling.

Note

Block property "Handle errors within block"

This setting is not applied by a calling block, nor is it transferred to called program blocks. The system settings still apply to higher-level and lower-level program blocks, provided dedicated local error handling has not been programmed for these blocks.

Error output priorities

In local error handling, information about the first error that occurred is displayed using the instructions GET_ERROR or GET_ERR_ID. If multiple errors occur at the same time while an instruction is being executed, these errors are displayed according to their priority. The following table shows the priority of different types of errors:

Priority	Error type
1	Error in the program code
2	Missing reference
3	Invalid range
4	DB does not exist
5	Operands are not compatible
6	Width of specified area is not sufficient
7	Timers or counters do not exist
8	No write access to a DB

Priority	Error type
9	I/O error
10	Instruction does not exist
11	Block does not exist
12	Invalid nesting depth

The highest priority is 1 and the lowest priority is 12.

Additional information

For additional information on the instructions, refer to the information system under PLC programming > Instructions > Instructions (S7-1200, S7-1500) > Basic instructions. The instructions are available in the programming languages LAD/FBD/STL/SCL and GRAPH.

See also

GET_ERROR: Get error locally (Page 3161)

GET_ERR_ID: Get error ID locally (Page 2840)

GET_ERR_ID: Get error ID locally (Page 3164)

GET_ERROR: Get error locally (Page 2837)

11.1.5.6 Example of how to handle program execution errors

Introduction

The local error handling options can be programmed individually or in combination with one another. To ensure that every error scenario that may occur within your program is recognized, we recommend a combination of local error handling options, as shown in the example below.

For a more precise error analysis, in addition to the RET_VAL output parameter you can also use the "GET_ERROR" or "GET_ERR_ID" instruction. These options provide you with error codes whose detailed explanations are available in the descriptions of the respective instructions.

There are also error scenarios in which the RET_VAL output parameter does not output a valid error code. If an access error occurs while reading an input parameter, for example, the outputs of the instruction are no longer written, because the execution of the instruction is interrupted. In this case, we recommend that you integrate the two instructions "GET_ERROR" and "GET_ERR_ID" in your program because they provide reliable error information even when this type of error occurs.

⚠ WARNING

Access error during reading of an input parameter

The RET_VAL parameter does not return a valid error code and no detailed error information is output in the diagnostics buffer.

A first indicator for an error can be either the BR bit of the status word or the ENO enable output. If these return signal state "0", there is an error in the execution of the instruction. Signal state "1" indicates that there is no error and no further error analysis is necessary, except when a memory access error is involved. In this case, signal state "1" can also indicate an error.

Procedure

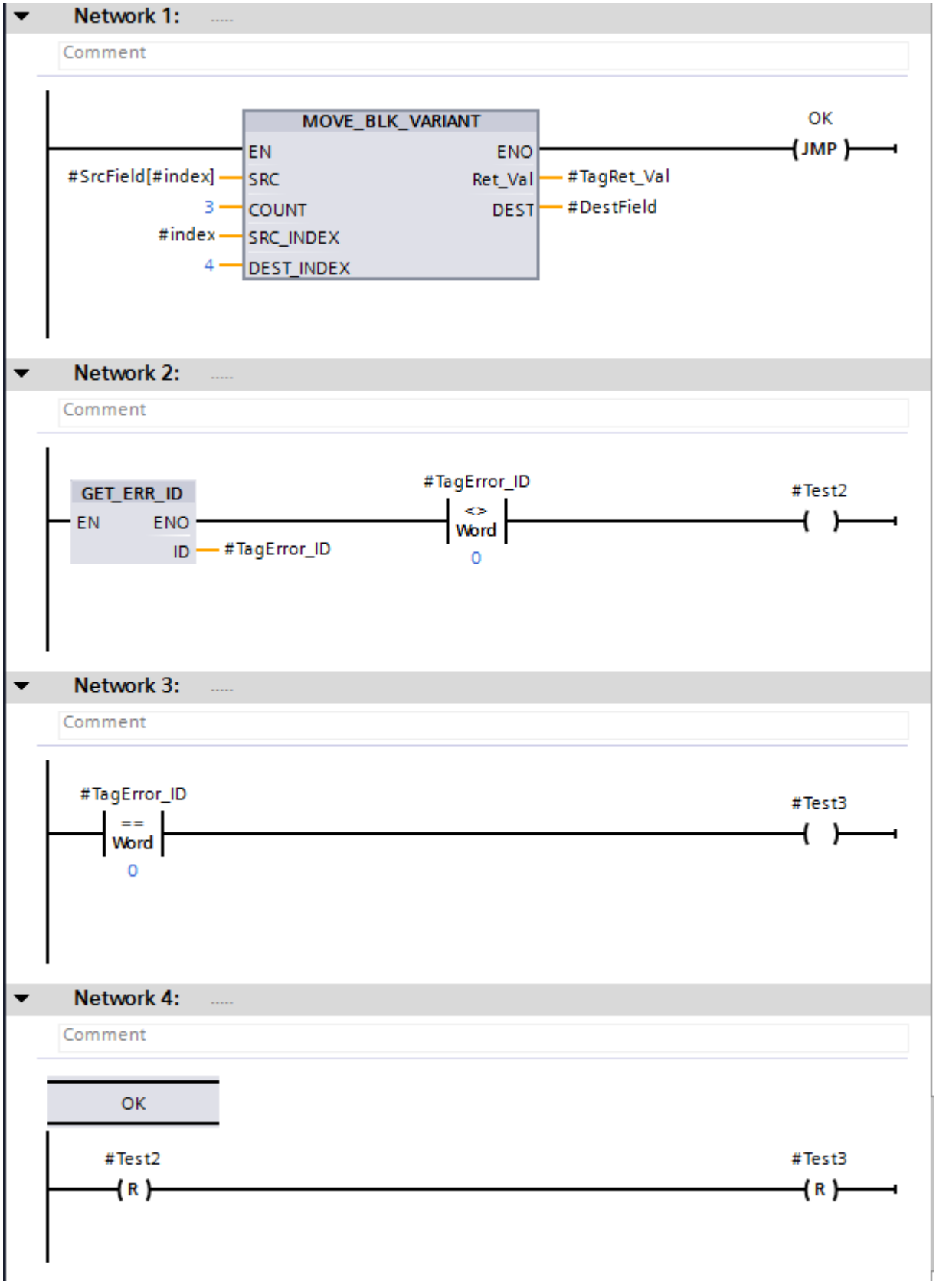
The following example shows you how to recognize an access error during reading of an input parameter:

1. Declare the block interface of your program block as follows:

Interface

		Name	Data type	Default value
1		Input		
2		SrcField	Array [1..100] of Real	
3		index	DInt	0
4		<Add new>		
5		Output		
6		TagOutput	Real	0.0
7		Test2	Bool	false
8		TagRet_Val	Int	0
9		DestField	Array [0..20] of Real	
10		Test3	Bool	false
11		<Add new>		
12		InOut		
13		Static		
14		Temp		
15		TagError_ID	Word	

2. Write the following program code:


Network 1:
Comment
MOVE_BLK_VARIANT
EN
ENO
#SrcField[#index]
SRC
Ret_Val
#TagRet_Val
3
COUNT
DEST
#DestField
#index
SRC_INDEX
4
DEST_INDEX
OK
JMP
Network 2:
Comment
GET_ERR_ID
EN
ENO
ID
#TagError_ID
#TagError_ID
<>
Word
0
#Test2
Network 3:
Comment
#TagError_ID
==
Word
0
#Test3
Network 4:
Comment
OK
#Test2
R
#Test3
R

In network 1, the instruction "MOVE_BLK_VARIANT": Move block" is called. The "SrcField" source area is accessed using a variable index in the SRC parameter. If no errors occur during execution of the instruction, the ENO enable output returns signal state "1", and program execution jumps to network 4 and continues there.

If an access error occurs during execution of the instruction, for example, due to the variable index, the instruction "GET_ERR_ID: Get error ID locally" in network 2 returns an error ID. The comparison of the error ID for "UNEQUAL" with the value "0" in network 2 returns the result #Test2 = TRUE. The comparison of the error ID for "EQUAL" with the value "0" in network 3 returns the result #Test3 = TRUE.

The #TagRet_Val operand at the RET_VAL output parameter returns no valid error code in this case.

Exceptions

There are some instructions, however, for which you cannot program the error handling as described in the example above. These includes the following instructions:

- Instructions that generally have no EN/ENO mechanism
- Instructions in which the ENO was disabled
- S_COMP
- PEEK, PEEK_BOOL, POKE, POKE_BOOL and POKE_BLK

The BR bit or the ENO enable output is set to TRUE in these instructions even if it may result in an access error.

The following example shows how you can program reliable error handling in the STL programming language:

		Name	Data type
1		Input	
2		StringArray	Array[0..10] of String
3		index	DInt
4		Output	
5		TagResult	Bool
6		InOut	
7		Static	
8		Temp	
9		TagError_ID	Word
10		Constant	

```
STL                                 Explanation
SET                                 // The #Tag_ErrorID operand is initial-
L 0                                 ized with "0".
T #Tag_ErrorID

CALL S_COMP                         // The instruction is called.
```

STL	Explanation
src_type := String	// Data type of parameters IN1 and IN2
relation := EQ	// The comparison type of the instruction
IN1 := #StringArray.THIS[#index]	// Variable access to the ARRAY component.
IN2 := 'STRING'	// The two values are compared with each other.
OUT := #TagResult	// If both values are identical, the #TagResult operand receives the signal state "1".
A BR	// The BR bit is queried.
CALL GET_ERR_ID	// The instruction is called.
RET_VAL := #Tag_ErrorID	// The instruction outputs an error code in the case of an access error.

Even if the BR bit has signal state "1", the access error is detected. You can query the error code by evaluating the #Tag_ErrorID operand of instruction "GET_ERR_ID: Get error ID locally".

11.2 Programming recommendations (S7-1200, S7-1500)

11.2.1 The new S7-1200/1500 CPU functions and programming recommendations at a glance (S7-1200, S7-1500)

Higher performance

The S7-1200/1500 CPUs offer much higher performance than the S7-300/400 CPUs. When programming with STEP 7 V5.x, you were probably used to methods such as absolute addressing to achieve higher performance from the CPU and leaner program code. Due to the high performance provided by CPUs of the S7-1200/1500 series, this is now no longer necessary. If you wish to reuse the programming code created on an S7-300/400CPU on an S7-1200/1500 CPU, it will probably be necessary to modify the program code here and there in order to fully exploit the high performance.

In the paragraphs below, we would like to introduce some new programming options of the S7-1200/1500 CPUs.

You can find more tips on improving the performance here: Increasing performance in STEP 7 (TIA Portal) (https://support.industry.siemens.com/cs/ww/en/view/37571372)

Universal symbolism

The S7-1500 allows you to use the symbolism throughout the entire project. Using the auto-complete function, you are given context-dependent support for programming with symbols within the programming editors. The data elements, for example those in a data block, are assigned only a symbolic name in the declaration but no fixed address within the data block. This allows you to fully exploit the high performance of the S7-1500 when accessing these data elements. The absolute addresses of operands need not be known and access errors are avoided.

Your program code will be clearer due to the symbolism and you have to comment less. All points of use are automatically updated when a correction is made to the symbolism.

Note

You can find additional information on the use of universal symbols at:

How can I address symbolically?

Symbolic addressing (Page 2213)

Optimized block access

With optimized block access, the declared data elements are arranged automatically in the available memory area of the block in a way that makes optimal use of its capacity. The data is structured and saved in way that is optimal for the CPU used. The storage is left to the system. The data elements are assigned only a symbolic name in the declaration by which the tag within the block can be addressed. This allows you to increase the performance of the CPU. Access errors, from the HMI, for example, are not possible.

Comparison of block accesses
Standard < > Optimized

SIEMENS

	Standard block access (S7-1200/1500 compatible with S7-300/400)	Optimized block access (S7-1200/1500 only)
Data management	You can address tags using both symbolic (memory-optimized) and absolute (user-defined) addresses.	The system manages and optimizes the data storage. This results in optimal use of the memory capacity.
Performance	The access to a CPU of the S7-1200/1500 series does not always occur as fast as possible because the data storage can be inefficient due to absolute addressing.	The access always occurs as fast as possible because the data storage is optimized by the system and no fixed addresses are assigned.
Susceptibility to errors	Absolute addressing (e.g., from HMI or when indirect addressing is used) can lead to inconsistencies after a change to the fixed address.	Access errors, e.g., from HMI or when indirect addressing is used, are not possible because the access is symbolic.
Retentivity	Valid for all tags of a data block	Valid for individual tags

Recommendation:
To achieve the best-possible performance, the combining of different block access types within your program is not recommended.

Note

You can find additional information on blocks with optimized access here:

AUTOHOTSPOT

New data types

The new data types LWORD, LINT, ULINT, LTIME, LTOD, LDT and ARRAY (32-bit limit) offer much higher calculation accuracy when using mathematical functions. In the area of implicit and explicit data type conversion, you have more options in comparison to the CPUs of the S7-300/400 series.

Note

You can find additional information on the new data types here:

AUTOHOTSPOT

PLC data types

PLC data types (UDT) are data structures that you define and that can be used multiple times within the program. The structure of a PLC data type is made up of several components, each of which can contain different data types. You define the type of components in the declaration of the PLC data type.

You can use the PLC data type as a base data type for defining tags and as a template for creating global data blocks. If you later make changes to the PLC data type, the changes are automatically updated at all points of use.

You can also symbolically access individual elements of an ARRAY within a PLC data type.

Note

You can find additional information on the PLC data types here:

What does a PLC data type look like and how can I declare it in my program?

- PLC data types (UDT) (Page 2277)

You can find programming recommendations for the PLC data type here:

- Using PLC data types (UDT) (Page 2229)
- Using the DB_ANY data type (Page 2218)

Uniform instructions in all programming languages

You are provided with a uniform set of instructions in all programming languages (LAD, FBD, STL, SCL, and GRAPH).

Slice access

Slice access gives you the option of specifically addressing areas within declared tags. You can implement symbolic access to a single bit up to the level of the tag. The single bit is then addressed absolutely.

Note

You can find additional information on slice access here:

How can I address with slice access?

- AUTOHOTSPOT
- AUTOHOTSPOT

Indirect addressing

Indirect addressing offers you the option of addressing operands whose address is not calculated until runtime. All programming languages provide general methods for indirect addressing, for example, via POINTER. In the STL and SCL programming languages, you can also use PEEK and POKE instructions.

Note

You can find additional information on indirect addressing here:

AUTOHOTSPOT

"Sample Library for Instructions"

You can use the "Sample Library for Instructions" to assist in programming. You will find practical and easy-to-understand programming examples for SIMATIC S7-1500 and S7-1200 in the LAD language here. You can simply insert the examples into your program and reuse them there.

Note

You can find additional information on the program examples here:

Sample Library for Instructions (Page 2568)

Additional programming recommendations

You can find additional information about programming recommendations and a programming guide in the Siemens Industry online support under:

- FAQs: Programming recommendations (http://support.automation.siemens.com/WW/llisapi.dll?aktprim=0&lang=en&referer=%2fWW%2f&func=cslib.csinfo&siteid=csius&Datakey=47071380&extranet=standard&groupid=40000002&viewreg=WW&nodeid0=29156492&objaction=csopen)
- Background and system descriptions in the Programming Guideline and Programming Styleguide (https://support.industry.siemens.com/cs/document/81318674/programming-guideline-and-programming-styleguide-for-s7-1200-and-s7-1500?dti=0&lc=en-WW)

11.2.2 Symbolic addressing (S7-1200, S7-1500)

Advantages of symbolic addressing

The use of universally applied and meaningful symbols in the overall project makes the program code easier to read and understand.

This gives you the following advantages:

- You do not have to write detailed comments.
- Data access is faster.
- No errors occur when accessing data.
- You no longer have to work with absolute addresses.
- The assignment of the symbol to the memory address is monitored by STEP 7, which means all points of use are automatically updated when the name or the address of a tag changes.

Programming in STEP 7 V5.x

STEP 7 V5.x already gave you the option to write a program that is easier to read by using descriptive names for operands and blocks. You did this by assigning the symbolic operands to memory addresses and blocks in the symbol table. In order for a change in the symbolism to also have an affect on the program code in the program editor, you had to use the "Operand priority" property to specify whether the symbol or the absolute value had priority.

The use of symbolic addressing allowed you to create a program with greater clarity. However, in some cases, such as when programming with user-defined data types (UDT), this could impair performance.

You could increase the performance by ignoring the symbolism in the UDT and using absolute addressing. This made it necessary, however, to know the data storage. Changes to the UDT were not automatically updated. Using absolute addressing, you could also access parts of a tag and edit these. The disadvantage of exclusively absolute addressing, however, was that the program code became cluttered after a certain level and you had to insert additional comments for better orientation.

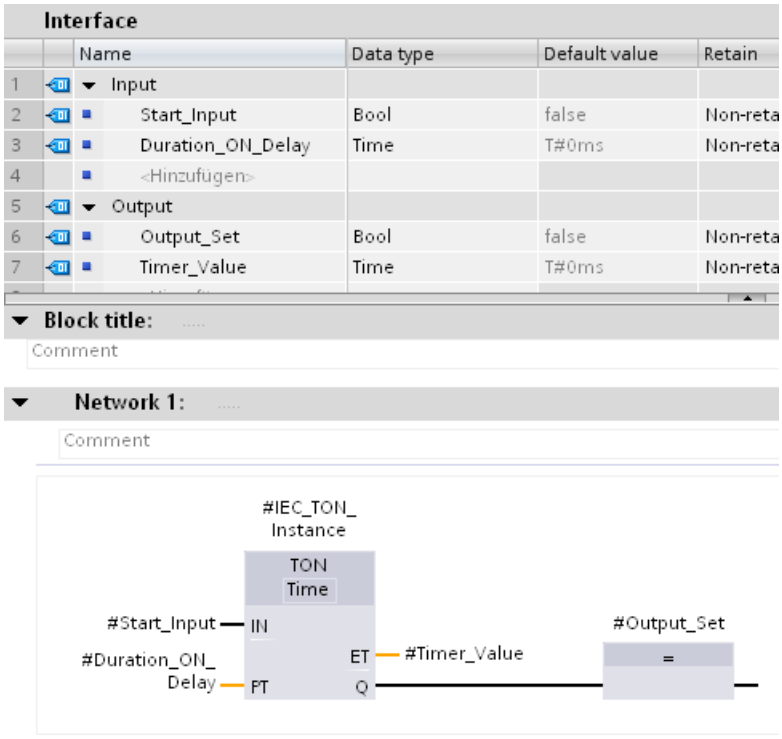
Procedure in STEP 7 TIA Portal

The S7-1500 CPU offers much higher performance than the S7-300/400 CPUs. To make full use of this high performance, we recommend that you enable optimized block access for all blocks and use symbolic addressing in the program code.

The program editor helps you work with symbols through context-sensitive input help, for example, auto-completion. Using it, you can easily access existing tags or instructions during programming.

Programming example

The following example shows you how to symbolically access individual elements:



You can use the tag names that you have defined in the block interface directly at the parameters of the TON instruction without knowing the absolute address of the tags.

See also

Using autocompletion (Page 4706)

11.2.3 Indirect addressing of ARRAY components (S7-1200, S7-1500)

Implementing ARRAY access in TIA Portal with a variable index

It is advisable to use an ARRAY when you want to process assembled data of the same data type. For addressing ARRAY elements, you can specify either constants or tags of the integer data type as index. Integers with lengths of up to 32 bits are allowed here.

With indirect addressing using a tag, the index is only calculated during program runtime. You can, for example, use a different index for each cycle in program loops. You can also access an ARRAY within a PLC data type (UDT).

This gives you the following advantages:

- No addressing is necessary with address registers or with self-assembled pointers, for example, an ANY pointer.
- More flexibility within your program.
- The variable index is available in all STEP 7 programming languages.
- It uses the existing names of data blocks and ARRAY tags (symbolic addressing). This improves readability of the program code.
- The start address of the ARRAY does not have to be known.
- The program code is easier to create and the compiler generates optimized program code.

Procedure in STEP 7 V5.x

In STEP 7 V5.x, you had to use an address register with the help of a self-configured POINTER for indirect addressing of ARRAY elements. The following aspects had to be taken into consideration for this:

- The name of the ARRAY was not used. This reduced the readability of the program code and make additional comments necessary.
- The start address of the ARRAY had to be known to perform the addressing.

The SCL programming language already supported indirect addressing with variable index.

Programming example in STEP 7 V5.x

The "Data_classic" data block is required for the following STL example. To address an element of the "Quantities" ARRAY, the following commands must be used:

STL	Explanation
OPN "Data_classic"	// The "Data_classic" data block is called.
L #index	// The value of the local tag #index is loaded into accumulator 1.
SLD 3	// Move bits 0 to 31 of accumulator 1 by 3 positions to the left.
	// Fill the now empty bit places with zeros.
LAR1	// Load address register 1 with contents of accumulator 1.

STL	Explanation
`L DBW [AR1, P#10.0]`	`// Load the ARRAY element addressed with #index into accumulator 1.` `// P#10.0 = Start address of the array`

Programming example in STEP 7 TIA Portal

In the example below, you see the indirect addressing of an ARRAY element in STL in the TIA Portal.

Create a global data block for this purpose:

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Specify the name "DB_Quantities".
4. Select the type of the data block "ARRAY DB".
5. Select the "DINT" data type as ARRAY data type.
6. Specify "10" as high ARRAY limit.
7. Click "OK".

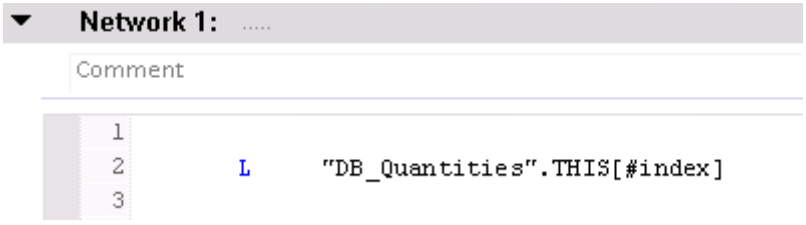
DB_Quantities

	Name	Data type
1	DB_Quantities	Array[0..10] of DInt
2	DB_Quantities[0]	DInt
3	DB_Quantities[1]	DInt
4	DB_Quantities[2]	DInt
5	DB_Quantities[3]	DInt
6	DB_Quantities[4]	DInt
7	DB_Quantities[5]	DInt
8	DB_Quantities[6]	DInt
9	DB_Quantities[7]	DInt
10	DB_Quantities[8]	DInt
11	DB_Quantities[9]	DInt
12	DB_Quantities[10]	DInt

1. Create a function block and name it "FB_Quantities".
2. Declare the block interface as follows:

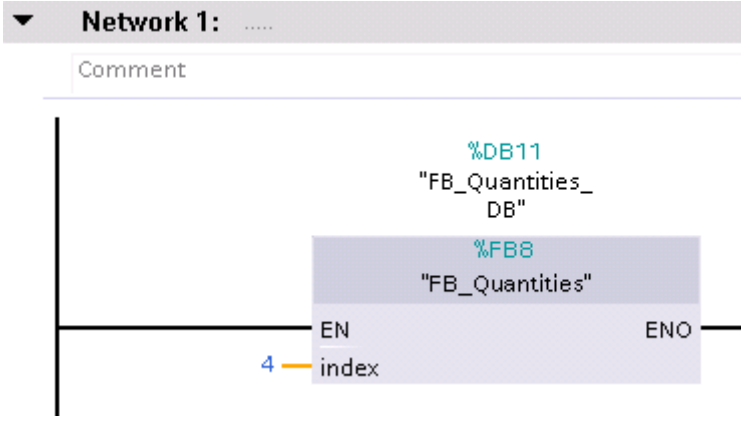
FB_Quantities		
	Name	Data type
1	Input	
2	Output	
3	InOut	
4	Static	
5	Temp	
6	Index	Int
7	Constant	

3. Write the following program code:



You need just one more program line for addressing an ARRAY element in the TIA Portal. The value of the ARRAY element #index is loaded directly from the data block into accumulator 1.

4. Call the "FB_Quantities" function block in OB1 and assign it an index between 0 - 10:



Note the following to get the best performance:

- Declare tags that are used as an ARRAY index as an integer of less than or equal to 32 bits.
- Store intermediate results and ARRAY indexes in the temporary local data area.

See also

Basic information on ARRAY (Page 2286)

ARRAY (Page 2286)

11.2.4 Using the DB_ANY data type (S7-1200, S7-1500)

Using the DB_ANY data type (S7-1200/1500)

The DB_ANY data type is used to identify any data block. For CPUs of the S7-1200/1500 series, you have the option of accessing a data block that is not yet available during programming. For this purpose, create a block parameter of the DB_ANY data type in the block interface of the accessing block. The data block name or a tag of data type DB_ANY which was previously assigned to the data block name is transferred to this parameter during runtime. You can symbolically process the content of a data block by means of the following instructions:

- VARIANT_TO_DB_ANY: Convert VARIANT to DB_ANY
- DB_ANY_TO_VARIANT: Convert DB_ANY to VARIANT

For more information about the instructions, refer to "Basic instructions > STL/SCL > Conversion operations > VARIANT".

The advantage of this procedure: You can create the program code before you know which data block will be processed.

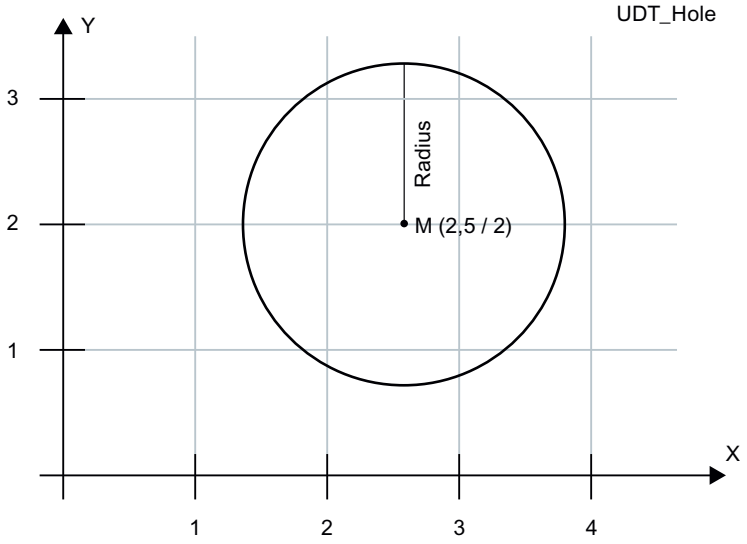
Programming example

The following example shows you how to use the DB_ANY data type:

A punching machine can punch out a variety of geometric shapes. The punching jobs are transferred to the machine and there are specific job data for each individual job. The job data differ in their job type as well as in their values.

Procedure - Creating the PLC data types

In the first job, a round hole is to be punched out of a piece of sheet metal. For the punching machine to be able to execute this job, it needs the center point coordinates and the radius of the hole. You can transfer these job data to the punching machine collectively in a PLC data type (UDT).



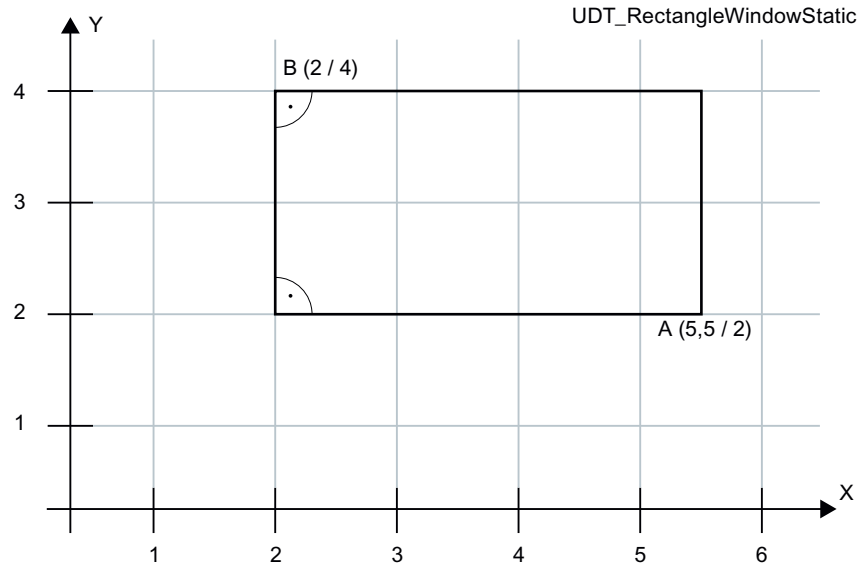
Create the PLC data type "UDT_Hole" to transfer the job data:

1. Double-click the "Add new data type" command in the "PLC data types" folder in the project tree.
   A new declaration table for creating a PLC data type will be created and opened.
2. Declare the following lines within the PLC data type:
   X-coordinate > REAL
   Y-coordinate > REAL
   Radius > REAL

UDT_Hole

	Name	Data type
1	X-coordinate	Real
2	Y-coordinate	Real
3	Radius	Real

A rectangle is to be punched out of a piece of sheet metal in the second job. For this job, you need two coordinates which define the upper left-hand point and the bottom right-hand point of the rectangle. You can transfer these job data to the punching machine collectively in the PLC data type "UDT_RectangleWindowStatic".



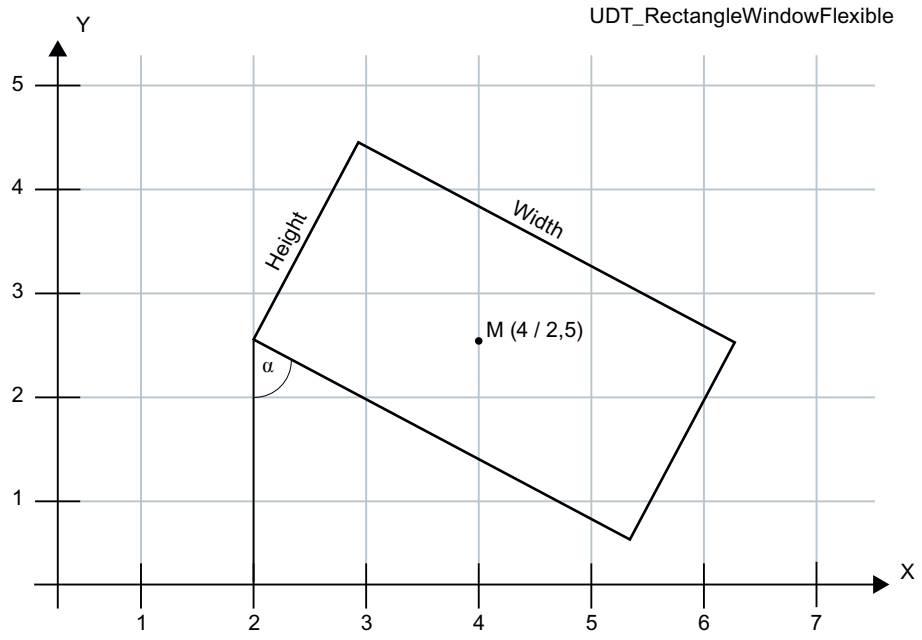
Create the PLC data type "UDT_RectangleWindowStatic":

1. Double-click the "Add new data type" command in the "PLC data types" folder in the project tree.
   A new declaration table for creating a PLC data type will be created and opened.
2. Declare the following lines within the PLC data type:
   X1-coordinate > REAL
   Y1-coordinate > REAL
   X2-coordinate > REAL
   Y2-coordinate > REAL

UDT_RectangleWindowStatic			
		Name	Data type
1		X1-coordinate	Real
2		Y1-coordinate	Real
3		X2-coordinate	Real
4		Y2-coordinate	Real

The job data of the "UDT_RectangleWindowStatic" can only be used to punch out rectangles whose edges are aligned parallel to the x and y axis.

If you want to punch out a rectangle with different alignment, i.e., not parallel to the x and y axis, you need an additional PLC data type. In it, you can specify the height and width, for example, as well as the alignment of the rectangle to the x axis by means of an angle.



Create the PLC data type "UDT_RectangleWindowFlexible":

1. Double-click the "Add new data type" command in the "PLC data types" folder in the project tree.
   A new declaration table for creating a PLC data type will be created and opened.
2. Declare the following lines within the PLC data type:
   X-coordinate > REAL
   Y-coordinate > REAL
   Height > REAL
   Width > REAL
   Angle > REAL

UDT_RectangleWindowFlexible

	Name	Data type
1	X-coordinate	Real
2	Y-coordinate	Real
3	Height	Real
4	Width	Real
5	Angle	Real

The x and y coordinates specify the center of the rectangle.

Procedure - Creating the data blocks

In the next section, you will learn how to transfer simple geometric shapes whose job data you have defined in the PLC data types to the punching machine. The punch jobs are broken down into individual punches in the program code; these punches are then executed consecutively by the punching machine. The punching machine has a cross table on which a sheet metal has been firmly clamped. You can move a cross table along the x and/or y axis just as in a coordinate system. The cross table is moved by two motors. The tool has different dies to punch out shapes from the sheet metal, such as circles and rectangles of different sizes. The tool can also be rotated by up to 90 degrees to cut out rectangles at different alignments.

You now use the PLC data types to create a data block. The data block then includes the specific values, for example, for a hole.

Create the data block "DB_OrderHole":

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Specify the name "DB_OrderHole".
4. Select the data block "UDT_Hole" as the type.
5. Click "OK".

Enter the appropriate start values:

DB_OrderHole

	Name	Data type	Start value
1	Static		
2	Hole	"UDT_Hole"	
3	X-coordinate	Real	10.0
4	Y-coordinate	Real	900.0
5	Diameter	Real	4.0

To manufacture a specific sheet metal part such as the side panel of a control cabinet, for example, the necessary geometric shapes are loaded to the punching machine. For this purpose, you need to create another data block which includes a list of data blocks.

Create the data block "DB_OrderList":

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Specify the name "DB_OrderList".
4. Select "Global DB" as the type of the data block.
5. Click "OK".

Create the following job list in the data block:

DB_OrderList

		Name	Data type
1		Static	
2		Order	Array[0..100] of DB_ANY
3		Order[0]	DB_ANY
4		Order[1]	DB_ANY
5		Order[2]	DB_ANY
6		Order[3]	DB_ANY
7		Order[4]	DB_ANY
8		Order[5]	DB_ANY
9		Order[6]	DB_ANY
10		Order[7]	DB_ANY
11		Order[8]	DB_ANY

Procedure - Creating the program code

The punching machine should now start processing the jobs. If it is already processing the jobs, it should get the next job from the job list and prepare it.

1. Create an SCL function block.
2. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
3. Click the "Function block (FB)" button.
4. Specify the name "FB_PickNextOrder".

5. Declare the block interface as follows:

FB_PickNextOrder		
	Name	Data type
1	Input	
2	Output	
3	InOut	
4	Static	
5	OrderReady	Bool
6	OrderNo	DInt
7	Order	DB_ANY
8	Temp	
9	Error	Int

6. Write the following program code:

```
1
2 IF #OrderReady THEN
3     #Order := "DB_OrderList".Order[#OrderNo];
4     #OrderNo := #OrderNo + 1;
5     #OrderReady := false;
6 END_IF;
```

The next job on the list is prepared by breaking down the current punch job into individual punches. The punching machine must be able to recognize which punch job it is processing.

1. Create an SCL function.
2. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
3. Click the "Function (FC)" button.
4. Specify the name "FC_PrepareOrder".

5. Declare the block interface as follows:

FC_PrepareOrder

	Name	Data type
1	Input	
2	Order	DB_ANY
3	Output	
4	Error	Int
5	InOut	
6	Temp	
7	WindowFlexible	"UDT_RectangleWindowFlexible"
8	WindowStatic	"UDT_RectangleWindowStatic"
9	Hole	"UDT_Hole"
10	a	Variant
11	Constant	
12	Return	
13	FC_PrepareOrder	Void

6. Write the following program code. The calls of the three functions are not yet given a red underline because they still have to be created.

```
1
2   #a := DB_ANY_TO_VARIANT(in := #Order, err => #Error);
3
4   CASE TypeOf(#a) OF
5       UDT_Hole:
6           VariantGet(SRC := #a,
7                      DST => #Hole);
8           "FC_PrepareHole"(#Hole);
9           ;
10
11      UDT_RectangleWindowStatic:
12          VariantGet(SRC := #a,
13                     DST => #WindowStatic);
14          "FC_PrepareWindowStatic"(#WindowStatic);
15          ;
16
17      UDT_RectangleWindowFlexible:
18          VariantGet(SRC := #a,
19                     DST => #WindowFlexible);
20          "FC_PrepareRectangleWindow"(#WindowFlexible);
21          ;
22      ELSE
23          ;
24  END_CASE;
```

The "VariantGet" instruction copies the information from the #a tag to the #Hole tag.

In the following section, you create the functions "FC_PrepareHole" and "FC_PrepareWindowStatic".

A separate function is created for each job type. The punch jobs are broken down into individual punches here and collected in an ARRAY.

1. Create a PLC data type.
2. Double-click the "Add new data type" command under "PLC data types".
   A new PLC data type with the name "UserDataType_x" is created.
3. Rename the PLC data type to "UDT_Punch".
4. Declare the following lines within the PLC data type:
   Tool > DINT
   x > REAL
   y > REAL
   w > BOOL

UDT_Punch

	Name	Data type
1	Tool	DInt
2	x	Real
3	y	Real
4	w	Bool

5. Create the global data block "DB_PunchList":

DB_PunchList

	Name	Data type
1	Static	
2	Punch	Array[0..100] of "UDT_Punch"
3	Punch[0]	"UDT_Punch"
4	Punch[1]	"UDT_Punch"
5	Punch[2]	"UDT_Punch"
6	Punch[3]	"UDT_Punch"
7	Punch[4]	"UDT_Punch"
8	Punch[5]	"UDT_Punch"
9	Punch[6]	"UDT_Punch"
10	Punch[7]	"UDT_Punch"
11	Punch[8]	"UDT_Punch"

To prepare the punch job for a hole, create an SCL function and name it "FC_PrepareHole".

1. Declare the block interface as follows:

FC_PrepareHole

		Name	Data type
1		Input	
2		a	"UDT_Hole"
3		Output	
4		InOut	
5		Temp	
6		Constant	
7		Return	
8		FC_PrepareHole	Void

2. Write the following program code:

```
1
2  IF #a.Radius = 4.0 THEN
3      "DB_PunchList".Punch[0].Tool := 1;
4
5  ELSIF #a.Radius = 6.0 THEN
6      "DB_PunchList".Punch[0].Tool := 2;
7
8  ELSE
9      ;
10     "DB_PunchList".Punch[0].x := #a."X-coordinate";
11     "DB_PunchList".Punch[0].y := #a."Y-coordinate";
12
13 END_IF;
```

To prepare the punch job for a window, you need a function which combines four punch sequences into the punch job. Create an SCL function and name it "FC_PrepareWindowStatic".

1. Declare the block interface as follows:

FC_PrepareWindowStatic		
	Name	Data type
1	Input	
2	a	"UDT_RectangleWindowStatic"
3	Output	
4	InOut	
5	Temp	
6	delta	Real
7	x	Real
8	x2	Real
9	y	Real
10	i	DInt
11	Constant	
12	Return	
13	FC_PrepareWindowStatic	Void

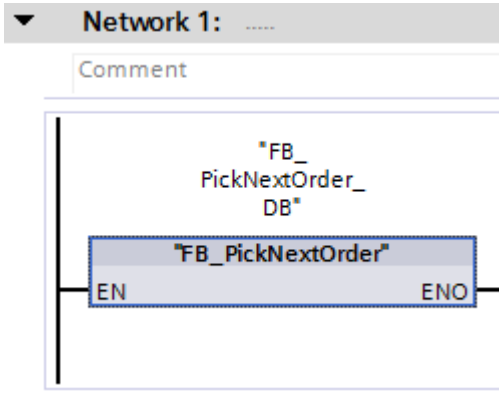
2. Write the following program code:

```
1
2  #delta := 6.0;  // Defined by tool 7
3  #i := 0;
4  #x := #a."X1-coordinate" + #delta;
5  WHILE #x < #a."X2-coordinate" - #delta  DO
6      "DB_PunchList".Punch[#i].Tool := 7;
7      "DB_PunchList".Punch[#i].x := #x;
8      "DB_PunchList".Punch[#i].y := #y + #delta;
9      "DB_PunchList".Punch[#i].w := 0; // Don't rotate
10     #x := #x + 2 * #delta;
11 END_WHILE;
12
13 #y := #a."Y1-coordinate" + #delta;
14 WHILE #y < #a."Y2-coordinate" - #delta DO
15     "DB_PunchList".Punch[#i].Tool := 7;
16     "DB_PunchList".Punch[#i].x := #x2 + #delta;
17     "DB_PunchList".Punch[#i].y := #y;
18     "DB_PunchList".Punch[#i].w := 0; // Don't rotate
19     #y := #y + 2 * #delta;
20 END_WHILE;
```

Call the SCL function block "FC_PrepareOrder" in the SCL function block "FB_PickNextOrder":

```
1
2  IF #OrderReady THEN
3      #Order := "DB_OrderList".Order[#OrderNo];
4      #OrderNo := #OrderNo + 1;
5      #OrderReady := false;
6      "FC_PrepareOrder"(Order:=#Order,
7                        Error=>#Error);
8
9  END_IF;
```

Then, call the "FB_PickNextOrder" in OB1:



Result

This example shows you how to use the instruction "DB_ANY_TO_VARIANT: Convert DB_ANY to VARIANT" to determine the PLC data type of a data block and how to select and execute a suitable function based on this.

11.2.5 Using PLC data types (UDT) (S7-1200, S7-1500)

Using a PLC data type

PLC data types (UDT) are data structures that you define yourself and that can be used multiple times in the program. The structure can be composed of multiple elements of different data types. You define the data types of the individual elements during the declaration of a PLC data type.

PLC data types are frequently used if an assembled data record with various data types is required and these are to be processed from different points in the program. For example, these could be:

- Data records for material tracking
- Parameter sets for a motor setting
- Recipes

The use of PLC data types provides you with the following benefits:

- The elements of a PLC data type can also be addressed indirectly, which means the address is variable and is not calculated until during runtime.
- Tags based on a PLC data type inherit all properties of the PLC data type. When there is a change to the PLC data type, all tags based on this PLC data type are therefore adapted automatically.
- Using universal symbolism makes the program easier to read, because the names of the individual elements of a PLC data type are displayed in the program.
- Optimum utilization of the high performance of an S7-1500 CPU.
- The PLC data type can be passed as a complete structure for a block call.
- A simplified call interface due to a lower number of parameters to be supplied.

Procedure in STEP 7 V5.x

STEP 7 V5.x already gave you the option to create a data record as a structured tag using the STRUCT data type or a PLC data type (UDT). However, the performance was adversely affected by the use of symbolic addressing.

The declaration in the data blocks was mostly implemented as an anonymous structure. The blocks themselves were then programmed to pass the values of the structure as actual parameters and the calculated values were copied back into the structure. This enabled you to also pass the data block number and use absolute addressing in the block. The number of parameters that you had to supply was often very large. The actual data was stored in the data blocks and the calculated values were passed to other blocks. However, no symbolism was available when passing the data block tags.

Programming example in STEP 7 TIA Portal

You can assign both a formula parameter and an actual parameter to a PLC data type. This means that you no longer have to declare individual parameters. If a block has an input parameter that is based on a PLC data type, you must transfer a tag of the same PLC data type as actual parameter.

The following example shows the call and the parameter assignment of a function block (FB) with two formal parameters:

1. To create a PLC data type, double-click the "Add new data type" command in the "PLC data types" folder in the project tree.
   A new declaration table for creating a PLC data type will be created and opened.
2. Rename the PLC data type to "UDT_Material".
3. Declare the following lines within the PLC data type:
   ArticleNumber > Data type: DINT
   ArticleName > Data type: STRING
   Amount > Data type: REAL
   Unit > Data type: STRING

UDT_Material

	Name	Data type
1	ArticleNumber	DInt
2	ArticleName	String
3	Amount	Real
4	Unit	String

Use the PLC data type within a global data block. You can specify the PLC data type either directly as data type of the data block or within the data block as data type of a tag.

Create a global data block for this purpose:

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Specify the name "DB_MaterialBuffer".
4. Select the type of the data block "ARRAY DB".
5. Select the PLC data type "UDT_Material" as ARRAY data type.

6. Specify "1000" as high ARRAY limit.
7. Click "OK".

DB_MaterialBuffer

	Name	Data type
1	DB_MaterialBuffer	Array[0..1000] of "UDT_Material"
2	DB_MaterialBuffer[0]	"UDT_Material"
3	ArticleNumber	DInt
4	ArticleName	String
5	Amount	Real
6	Unit	String
7	DB_MaterialBuffer[1]	"UDT_Material"
8	DB_MaterialBuffer[2]	"UDT_Material"
9	DB_MaterialBuffer[3]	"UDT_Material"
10	DB_MaterialBuffer[4]	"UDT_Material"
11	DB_MaterialBuffer[5]	"UDT_Material"

At the function block call, interconnect the formal parameters with tags from the global data block "DB_MaterialBuffer".

1. Create an SCL function block and name it "FB_Material".
2. Declare the block interface as follows:

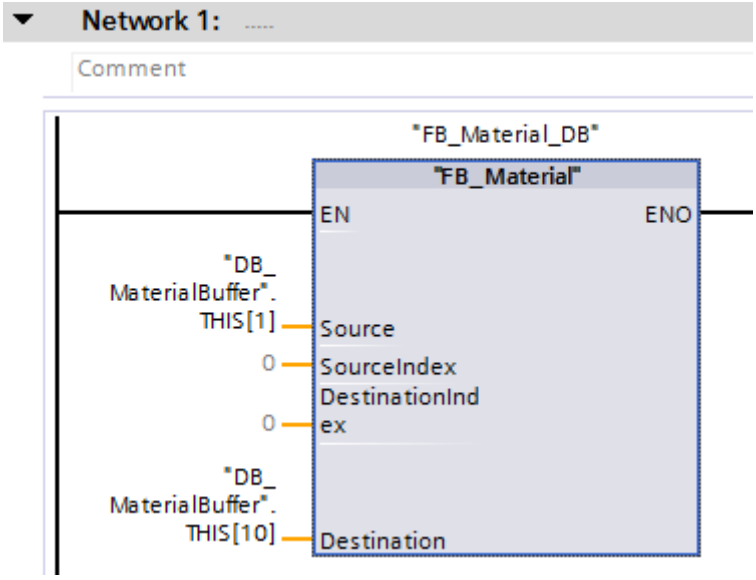
FB_Material

	Name	Data type
1	Input	
2	Source	Variant
3	SourceIndex	DInt
4	DestinationIndex	DInt
5	Output	
6	InOut	
7	Destination	Variant
8	Static	
9	Temp	
10	Result	Int
11	Constant	

3. Write the following program code:

```
1
2 #Result := MOVE_BLK_VARIANT(SRC:=#Source,
3                              COUNT:=3,
4                              SRC_INDEX:=#SourceIndex,
5                              DEST_INDEX:=#DestinationIndex,
6                              DEST=>#Destination);
```

4. Call the "FB_Material" function block in OB1 and interconnect the formal parameters with tags of the global data block "DB_MaterialBuffer":



The material data are moved within the global data block "DB_MaterialBuffer".

See also

Other useful information regarding structured PLC tags (Page 4849)

Creating structured PLC tags (Page 4851)

Basics of PLC data types (UDT) (Page 2277)

PLC data types (UDT) (Page 2277)

11.2.6 Using MOVE instructions in STL (S7-1500)

Possible applications

You can now program with MOVE instructions in STL on an S7-1500 CPU.

This gives you the following advantages:

- The program structure is easier to create.
- The performance of the CPU increases.

Programming in STEP 7 V5.x

In STEP 7 V5.x, you used the "BLKMOV: Move block" and "UBLKMOV: Move block uninterruptible" system functions to implement the MOVE functionality.

Procedure in STEP 7 TIA Portal

The following new MOVE instructions are available in STEP 7 TIA Portal:

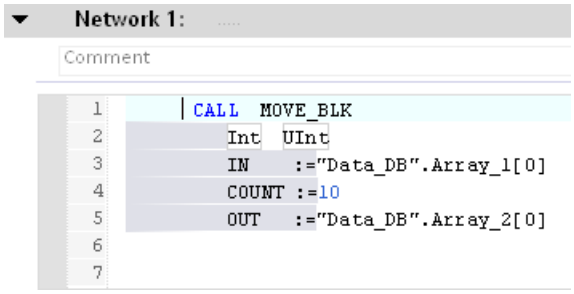
- MOVE: Move value
- MOVE_BLK: Move block
- MOVE_BLK_VARIANT: Move block
- UMOVE_BLK: Move block uninterruptible

Programming example

The following example shows you how the "MOVE_BLK instruction works: Move block". An ARRAY block is copied into another ARRAY block:

Data_DB

	Name	Data type
1	Static	
2	Array_1	Array [0..10] of Int
3	Array_1[0]	Int
4	Array_1[1]	Int
5	Array_1[2]	Int
6	Array_1[3]	Int
7	Array_1[4]	Int
8	Array_1[5]	Int
9	Array_1[6]	Int
10	Array_1[7]	Int
11	Array_1[8]	Int
12	Array_1[9]	Int
13	Array_1[10]	Int
14	Array_2	Array [0..10] of Int
15	Array_2[0]	Int
16	Array_2[1]	Int
17	Array_2[2]	Int
18	Array_2[3]	Int
19	Array_2[4]	Int
20	Array_2[5]	Int
21	Array_2[6]	Int
22	Array_2[7]	Int
23	Array_2[8]	Int
24	Array_2[9]	Int
25	Array_2[10]	Int



Using the MOVE_BLK instruction, ten elements from "Array_1" of the "Data_DB" data block are copied into "Array_2" of the same data block.

11.2.7 Using IEC timers and counters (S7-1200, S7-1500)

Advantages of IEC timers and counters

The universal use of IEC timers and counters makes your program code more efficient.

This gives you the following advantages:

- The blocks can be called multiple times with newly generated instance data blocks.
- The IEC counters have a large counting range.
- Compared to S5 timers, IEC timers have better performance and greater time accuracy.

Programming in STEP 7 V5.x

The S5 timers and counters in STEP 7 V5.x were addressed absolutely via a number. Due to this dependency on numbers, program blocks were not reusable with S5 timers and counters.

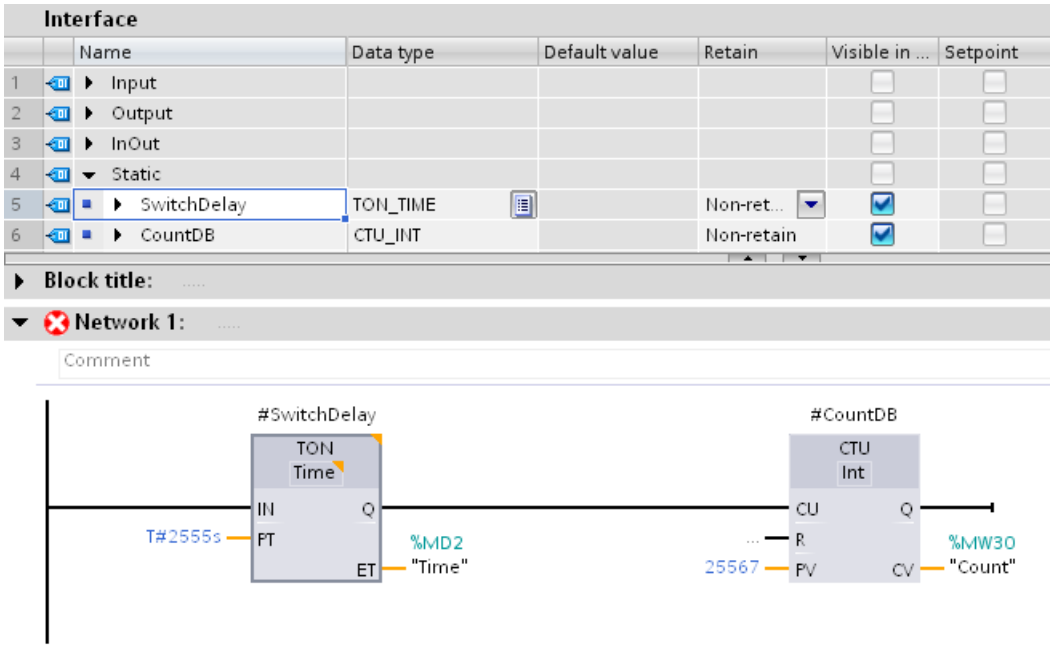
The value range of a timer was limited to 9990 seconds and that of a counter limited to a maximum of 999.

Procedure in STEP 7 TIA Portal

You declare IEC timers and counters in the program block where they are called or needed. The IEC timer is a structure of data type IEC_TIMER, IEC_LTIMER, or TON_TIME and TON_LTIME, for example, which you can also declare as a local tag in a block. The IEC counter is a structure of data type IEC_SCOUNTER, IEC_USCOUNTER, etc.

Programming examples in the TIA Portal

The following example shows you how to declare an IEC timer and IEC counter as a local tag:



The data of the TON IEC timer and the CTU IEC counter is stored as a local tag (multi-instance) in the block interface.

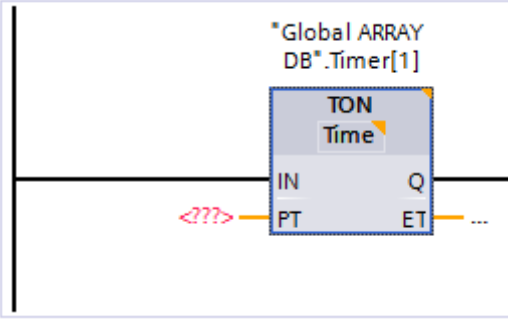
You also have the option to create IEC timers and IEC counters within structures as multiple instances and to use them in your program code.

1. To do this, for example, create a global data block with an ARRAY of TON. The data type TON does not appear in the drop-down list, but can be entered manually:

Global ARRAY DB

	Name	Data type
1	Static	
2	Timer	Array[0..10] of TON
3	Timer[0]	TON
4	Timer[1]	TON
5	Timer[2]	TON
6	Timer[3]	TON
7	Timer[4]	TON
8	Timer[5]	TON
9	Timer[6]	TON
10	Timer[7]	TON
11	Timer[8]	TON
12	Timer[9]	TON
13	Timer[10]	TON

2. Create a function block and drag the instruction "TON: Generate on-delay" to a network. Call the instance of IEC timer TON as follows:



Call of a timer as multi-instance

If you want to use the IN parameter to start a timer as multi-instance, you must not initialize it beforehand in your program code. In this case, the timer called at the IN parameter no longer recognizes a positive signal edge and the timer is not started:

1. Create the timer "Time_1" as a multi-instance of the data type "TP_TIME" in the "Static" section of the block interface.

2. Write the following program code:

STL	Explanation
`A "Tag_Output"`	`// When the "Tag_Output" output receives the signal state 1,`

STL	Explanation
`= #Time_1.IN`	`// the IN parameter of the multi-instance timer #Timer_1 is initialized with a positive signal edge.`

`CALL #Time_1`
`time_type := Time`
`IN := "Tag_Output"`
`PT := T#30s`
`Q := "Tag_4"`
`ET := "Tag_ElapsedTime"`
	`// When the multi-instance timer is now called and the IN parameter queried again, the timer is not started because there is no new positive signal edge.`
	
	`// Enter TIME as data type of the instruction.`

This is why you must program the initialization of the multi-instance timer within the call.

STL	Explanation

`CALL #Time_1`
`time_type := Time`
`IN := "Tag_Output"`
`PT := T#30s`
`Q := "Tag_4"`
`ET := "Tag_ElapsedTime"`
	`// The timer is called and started.`
	
	`// Enter TIME as data type of the instruction.`

See also

Calling IEC timers (Page 3221)
Calling IEC counters (Page 3252)
Timers (Page 2261)

11.3 Data types

11.3.1 Overview of the valid data types

Validity of data type groups

The data type groups define the properties of the data, for example, the representation of the contents and the valid memory areas.

In the user program, you have the option of using pre-defined data types, which you add to the data types you have defined yourself. The following type categories are available for this:

- Elementary data types (binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD CHAR, WCHAR)
- Complex data types (DT, LDT, DTL, STRING, WSTRING, ARRAY, STRUCT)
- User-defined data types (PLC data type (UDT))
- Pointer
- Parameter types
- System data types
- Hardware data types

The following tables show the availability of data types in the various S7-CPUs:

Table 11-1 Binary numbers

Binary numbers	S7-300/400	S7-1200	S7-1500
BOOL (Page 2243)	X	X	X
Bit strings			
BYTE (Page 2244)	X	X	X
WORD (Page 2245)	X	X	X
DWORD (Page 2245)	X	X	X
LWORD (Page 2246)	-	-	X

Table 11-2 Integers

Integers	S7-300/400	S7-1200	S7-1500
SINT (Page 2248)	-	X	X
INT (Page 2250)	X	X	X
DINT (Page 2252)	X	X	X
USINT (Page 2249)	-	X	X
UINT (Page 2251)	-	X	X
UDINT (Page 2253)	-	X	X
LINT (Page 2254)	-	-	X
ULINT (Page 2255)	-	-	X

Table 11-3 Floating-point numbers

Floating-point numbers	S7-300/400	S7-1200	S7-1500
REAL (Page 2257)	X	X	X
LREAL (Page 2258)	-	X	X

Table 11-4 Timers

Timers	S7-300/400	S7-1200	S7-1500
S5TIME (Page 2261)	X	-	X
TIME (Page 2262)	X	X	X
LTIME (Page 2263)	-	-	X

Table 11-5 Date and time

Date and time	S7-300/400	S7-1200	S7-1500
DATE (Page 2264)	X	X	X
TIME_OF_DAY (TOD) (Page 2264)	X	X	X
LTOD (LTIME_OF_DAY) (Page 2265)	-	-	X
DT (DATE_AND_TIME) (Page 2265)	X	-	X
LDT (Page 2266)	-	-	X
DTL (Page 2267)	-	X	X

Table 11-6 Character strings

Character strings	S7-300/400	S7-1200	S7-1500
CHAR (Page 2269)	X	X	X
WCHAR (Page 2269)	-	X	X
STRING (Page 2270)	X	X	X
WSTRING (Page 2273)	-	X	X

Table 11-7 PLC data types (UDT)

PLC data types (UDT)	S7-300/400	S7-1200	S7-1500
PLC data type (Page 2277) (UDT)	X	X	X

Table 11-8 Anonymous structures

Anonymous structures	S7-300/400	S7-1200	S7-1500
STRUCT (Page 2282)	X	X	X

Table 11-9 ARRAY

ARRAY	S7-300/400	S7-1200	S7-1500
ARRAY [....] of <data type> (Page 2286)	X	X	X

Table 11-10 Pointer

Pointer	S7-300/400	S7-1200	S7-1500
POINTER (Page 2302)	X	-	X
ANY (Page 2305)	X	-	X
VARIANT (Page 2308)	-	X	X

Table 11-11 Parameter types

Parameter types	S7-300/400	S7-1200	S7-1500
TIMER (Page 2325)	X	-	X
COUNTER (Page 2325)	X	-	X
BLOCK_FC (Page 2325)	X	-	X
BLOCK_FB (Page 2325)	X	-	X
BLOCK_DB (Page 2325)	X	-	-
BLOCK_SDB (Page 2325)	X	-	-
BLOCK_SFB (Page 2325)	X	-	-
BLOCK_SFC (Page 2325)	X	-	-
BLOCK_OB (Page 2325)	X	X	X
VOID (Page 2325)	X	X	X
PARAMETER (Page 2325)	-	X	X

Table 11-12 System data types

System data types	S7-300/400	S7-1200	S7-1500
IEC_TIMER (Page 2326)	X[1)]	X	X
IEC_LTIMER (Page 2326)	-	-	X
IEC_SCOUNTER (Page 2326)	-	X	X
IEC_USCOUNTER (Page 2326)	-	X	X
IEC_COUNTER (Page 2326)	X[2)]	X	X
IEC_UCOUNTER (Page 2326)	-	X	X
IEC_DCOUNTER (Page 2326)	-	X	X
IEC_UDCOUNTER (Page 2326)	-	X	X
IEC_LCOUNTER (Page 2326)	-	-	X
IEC_ULCOUNTER (Page 2326)	-	-	X
ERROR_STRUCT (Page 2326)	-	X	X
NREF (Page 2326)	-	X	X
CREF (Page 2326)	-	X	X

System data types	S7-300/400	S7-1200	S7-1500
VREF (Page 2326)	-	X	X
SSL_HEADER (Page 2326)	X	-	-
CONDITIONS (Page 2326)	-	X	-
TADDR_Param (Page 2326)	-	X	X
TCON_Param (Page 2326)	-	X	X
HSC_Period (Page 2326)	-	X	-
[1] For S7-300/400 CPUs, the data type is represented by TP, TON and TOF.			
[2] For S7-300/400 CPUs, the data type is represented by CTU, CTD and CTUD.			

Table 11-13 Hardware data types

Hardware data types	S7-300/400	S7-1200	S7-1500
REMOTE (Page 2328)	-	X	X
HW_ANY (Page 2328)	-	X	X
HW_DEVICE (Page 2328)	-	X	X
HW_DPMASTER (Page 2328)	-	-	X
HW_DPSLAVE (Page 2328)	-	X	X
HW_IO (Page 2328)	-	X	X
HW_IOSYSTEM (Page 2328)	-	X	X
HW_SUBMODULE (Page 2328)	-	X	X
HW_MODULE (Page 2328)	-	-	X
HW_INTERFACE (Page 2328)	-	X	X
HW_IEPORT (Page 2328)	-	X	X
HW_HSC (Page 2328)	-	X	X
HW_PWM (Page 2328)	-	X	X
HW_PTO (Page 2328)	-	X	X
EVENT_ANY (Page 2328)	-	X	X
EVENT_ATT (Page 2328)	-	X	X
EVENT_HWINT (Page 2328)	-	X	X
OB_ANY (Page 2328)	-	X	X
OB_DELAY (Page 2328)	-	X	X
OB_TOD (Page 2328)	-	X	X
OB_CYCLIC (Page 2328)	-	X	X
OB_ATT (Page 2328)	-	X	X
OB_PCYCLE (Page 2328)	-	X	X
OB_HWINT (Page 2328)	-	X	X
OB_DIAG (Page 2328)	-	X	X
OB_TIMEERROR (Page 2328)	-	X	X
OB_STARTUP (Page 2328)	-	X	X
PORT (Page 2328)	-	X	X
RTM (Page 2328)	-	X	X
PIP (Page 2328)	-	-	X

Hardware data types	S7-300/400	S7-1200	S7-1500
CONN_ANY (Page 2328)	-	X	X
CONN_PRG (Page 2328)	-	X	X
CONN_OUC (Page 2328)	-	X	X
CONN_R_ID (Page 2328)	-	-	X
DB_ANY (Page 2328)	-	X	X
DB_WWW (Page 2328)	-	X	X
DB_DYN (Page 2328)	-	X	X

Note

Depending on the CPU version, the actually valid data types can deviate slightly from the table.

11.3.2 Binary numbers

11.3.2.1 BOOL (bit)

Description

An operand of data type BOOL represents a bit value and contains one of the following values:

- TRUE
- FALSE

The following table shows the properties of data type BOOL:

Length (bits)	Format	Value range	Examples of value input
1	Boolean	FALSE or TRUE BOOL#0 or BOOL#1 BOOL#FALSE or BOOL#TRUE	TRUE BOOL#1 BOOL#TRUE
	Unsigned integers (decimal system)	0 or 1	1
	Binary numbers	2#0 or 2#1	2#0
	Octal numbers	8#0 or 8#1	8#1
	Hexadecimal numbers	16#0 or 16#1	16#1

Note

Applies to CPUs of the S7-1500 series

For a block with the block property "Optimized block access", the bit has a length of 1 byte.

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.2.2 Bit strings

BYTE

Description

An operand of data type BYTE is a bit string of 8 bits.

The following table shows the properties of data type BYTE:

Length (bits)	Format	Value range	Examples of value input	
			Constants	Absolute and symbolic addresses
8	Integers[1] (decimal system)	Signed integers: -128 to +127 Unsigned integers: 0 to 255	• 15 • BYTE#15 • BYTE#10#15 • B#15	• IB2 • MB10 • DB1.DBB4 • Tag_Name
	Binary numbers	2#0 to 2#1111_1111	• 2#0000_1111 • BYTE#2#0000_1111 • B#2#0000_1111	
	Octal numbers	8#0 to 8#377	• 8#17 • BYTE#8#17 • B#8#17	
	Hexadecimal numbers	16#0 to 16#FF	• 16#0F • BYTE#16#0F • B#16#0F	
[1] The value range depends on the relevant interpretation or conversion.				

Note

The BYTE data type cannot be compared for more than or less than. It can only be supplied with the same decimal data that can be processed by the SINT and USINT data types.

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

WORD

Description

An operand of data type WORD is a bit string of 16 bits.

The following table shows the properties of data type WORD:

Length (bits)	Format	Value range	Examples of value input	
			Constants	Absolute and symbolic addresses
16	Integers (decimal system)	Signed integers: -32_768 to +32_767 Unsigned integers: 0 to 65_535	• 61_680 • WORD#61_680 • WORD#10#61_680 • W#61_680	• MW10 • DB1.DBW2 • Tag_Name
	Binary numbers	2#0 to 2#1111_1111_1111_1111	• 2#1111_0000_1111_0000 • WORD#2#1111_0000_1111_0000 • W#2#1111_0000_1111_0000	
	Octal numbers	8#0 to 8#177_777	• 8#170_360 • WORD#8#170_360 • W#8#170_360	
	Hexadecimal numbers	16#0 to 16#FFFF	• 16#F0F0 • WORD#16#F0F0 • W#16#F0F0	
	BCD	C#0 to C#999	C#55	
	Decimal sequence	B#(0, 0) to B#(255, 255)	B#(127, 200)	

Note

The WORD data type cannot be compared for more than or less than. It can only be supplied with the same decimal data that can be processed by the INT and UINT data types.

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

DWORD

Description

An operand of data type DWORD is a bit string of 32 bits.

The following table shows the properties of data type DWORD:

Length (bits)	Format	Value range	Examples of value input	
			Constants	Absolute and symbolic addresses
32	Integers (decimal system)	Signed integers: -2_147_483_648 to +2_147_483_648 Unsigned integers: 0 to 4_294_967_295	• +15_793_935 • DWORD#+15_793_935 • DWORD#10# +15_793_935 • DW#+15_793_935	• MD10 • DB1.DBD8 • Tag_Name
	Binary numbers	2#0 to 2#1111_1111_1111_1111_1111_1111_1111_1111	• 2#0000_0000_1111_0000_1111_1111_0000_1111 • DWORD#2#0000_0000_1111_0000_1111_1111_0000_1111 • DW#2#0000_0000_1111_0000_1111_1111_0000_1111	
	Octal numbers	8#0 to 8#37_777_777_777	• 8#74_177_417 • DWORD#8#74_177_417 • DW#8#74_177_417	
	Hexadecimal numbers	16#0000_0000 to 16#FFFF_FFFF	• 16#00F0_FF0F • DWORD#16#00F0_FF0F • DW#16#00F0_FF0F	
	Decimal sequence	B#(0, 0, 0, 0) to B#(255, 255, 255, 255)	B#(127, 200, 127, 200)	

Note

The DWORD data type cannot be compared for more than or less than. It can only be supplied with the same decimal data that can be processed by the DINT and UDINT data types.

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

LWORD (S7-1500)

Description

An operand of data type LWORD is a bit string of 64 bits.

The following table shows the properties of data type LWORD:

Length (bits)	Format	Value range	Examples of value input
64	Integers (decimal system)	Signed integers: -9_223_372_036_854_775_808 to +9_223_372_036_854_775_807 Unsigned integers: 0 to 18_446_744_073_709_551_615	• +26_123_590_360_715 • LWORD#+26_123_590_360_715 • LWORD#10#+26_123_590_360_715 • LW#+26_123_590_360_715
	Binary numbers	2#0 to 2#1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111	• 2#0000_0000_0000_0000_0000_1011_1110_0001_0010_1111_0101_0010_1101_1110_1000_1011 • LWORD#2#0000_0000_0000_0000_0000_1011_1110_0001_0010_1111_0101_0010_1101_1110_1000_1011 • LW#2#0000_0000_0000_0000_0000_1011_1110_0001_0010_1111_0101_0010_1101_1110_1000_1011
	Octal numbers	8#0 to 8#1_777_777_777_777_777_777_777	• 8#13_724_557_213 • LWORD#8#13_724_557_213 • LW#8#13_724_557_213
	Hexadecimal numbers	16#0000_0000 to 16#FFFF_FFFF_FFFF_FFFF	• 16#0000_0000_5F52_DE8B • LWORD#16#0000_0000_5F52_DE8B • LW#16#0000_0000_5F52_DE8B
	Decimal sequence	B#(0, 0, 0, 0, 0, 0, 0, 0) to B#(255, 255, 255, 255, 255, 255, 255, 255)	B#(127, 200, 127, 200, 127, 200, 127, 200)

Note

The LWORD data type cannot be compared for more than or less than. It can only be supplied with the same decimal data that can be processed by the LINT and ULINT data types.

See also

Overview of the valid data types (Page 2239)

Overview of data type conversion (Page 2330)

Implicit conversions (Page 2333)

Explicit conversions (Page 2382)

Data type conversion for S7-1200 (Page 2465)

11.3.3 Integers

11.3.3.1 SINT (8-bit integers) (S7-1200, S7-1500)

Description

An operand of data type SINT (Short INT) has a length of 8 bits and consists of two components: a sign and a numerical value in the two's complement. The signal states of bits 0 to 6 represent the number value. The signal state of bit 7 represents the sign. The sign may assume "0" for the positive, or "1" for the negative signal state.

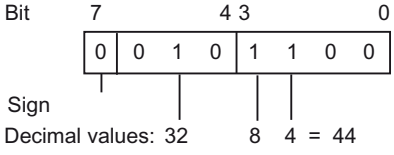
An operand of data type SINT occupies one BYTE in the memory.

The following table shows the properties of data type SINT:

Length (bits)	Format	Value range	Examples of value input
8	Signed integers (decimal system)	-128 to +127	• +44 • SINT#+44 • SINT#10#+44 The value range extends to a maximum of SINT#255 when using the type SINT#. This value is interpreted as an integer with -1.
	Binary numbers (only positive)	2#0 to 2#0111_1111	• 2#0010_1100 • SINT#2#0010_1100 • SINT#2#10
	Octal numbers (only positive)	8#0 to 8#177	• 8#54 • SINT#8#54
	Hexadecimal numbers (only positive)	16#0 to 16#7F	• 16#2C • SINT#16#2C The value range extends to a maximum of SINT#16#FF when using the type SINT#. This value is interpreted as an integer with -1.

Example

The following figure shows the integer +44 as a binary number:



See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.3.2 USINT (8-bit integers) (S7-1200, S7-1500)

Description

An operand of data type USINT (Unsigned Short INT) has a length of 8 bits and contains unsigned numerical values:

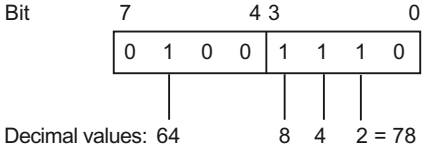
An operand of data type USINT occupies one BYTE in the memory.

The following table shows the properties of data type USINT:

Length (bits)	Format	Value range	Examples of value input
8	Unsigned integers (decimal system)	0 to 255	• 78 • USINT#78 • USINT#10#78
	Binary numbers	2#0 to 2#1111_1111	• 2#0100_1110 • USINT#2#0100_1110 • USINT#2#10
	Octal numbers	8#0 to 8#377	• 8#116 • USINT#8#116
	Hexadecimal numbers	16#0 to 16#FF	• 16#4E • USINT#16#4E

Example

The following figure shows the integer 78 as a binary number:



See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.3.3 INT (16-bit integers)

Description

An operand of data type INT has a length of 16 bits and consists of two components: a sign and a numerical value in the two's complement. The signal states of bits 0 to 14 represent the number value. The signal state of bit 15 represents the sign. The sign may assume "0" for the positive, or "1" for the negative signal state.

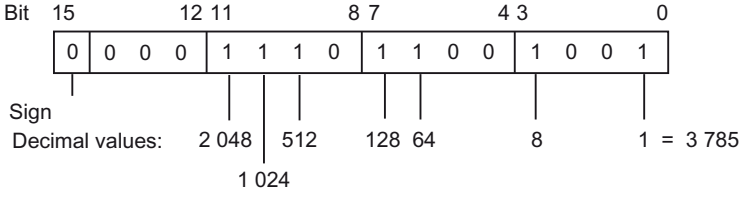
An operand of data type INT occupies two BYTE in the memory.

The following table shows the properties of data type INT:

Length (bits)	Format	Value range	Examples of value input
16	Signed integers (decimal system)	-32_768 to +32_767	• +3_785 • INT#+3_785 • INT#10#+3_785
	Binary numbers (only positive)	2#0 to 2#0111_1111_1111_1111	• 2#0000_1110_1100_1001 • INT#2#0000_1110_1100_1001 • INT#2#10
	Octal numbers	8#0 to 8#7_7777	• 8#7311 • INT#8#7311
	Hexadecimal numbers (only positive)	16#0 to 16#7FFF	• 16#0EC9 • INT#16#0EC9

Example

The following figure shows the integer +3785 as a binary number:



See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.3.4 UINT (16-bit integers) (S7-1200, S7-1500)

Description

An operand of data type UINT (Unsigned INT) has a length of 16 bits and contains unsigned numerical values.

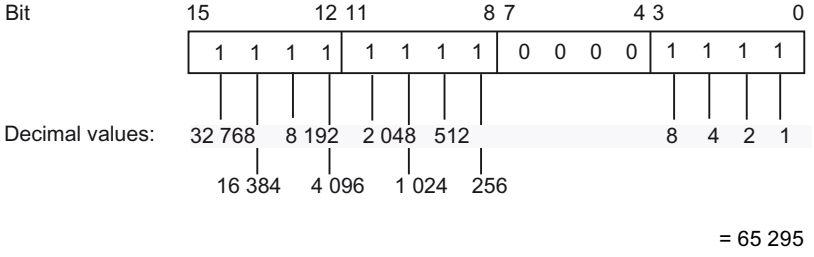
An operand of data type UINT occupies two BYTE in the memory.

The following table shows the properties of data type UINT:

Length (bits)	Format	Value range	Examples of value input
16	Unsigned integers (decimal system)	0 to 65_535	• 65_295 • UINT#65_295 • UINT#10#65_295
	Binary numbers	2#0 to 2#1111_1111_1111_1111	• 2#1111_1111_0000_1111 • UINT#2#1111_1111_0000_1111 • UINT#2#10
	Octal numbers	8#0 to 8#17_7777	• 8#17_7417 • UINT#8#17_7417
	Hexadecimal numbers	16#0 to 16#FFFF	• 16#FF0F • UINT#16#FF0F

Example

The following figure shows the integer 65295 as a binary number:



See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.3.5 DINT (32-bit integers)

Description

An operand of data type DINT (Double INT) has a length of 32 bits and consists of two components: a sign and a numerical value in the two's complement. The signal states of bits 0 to 30 represent the number value. The signal state of bit 31 represents the sign. The sign may assume "0" for the positive, or "1" for the negative signal state.

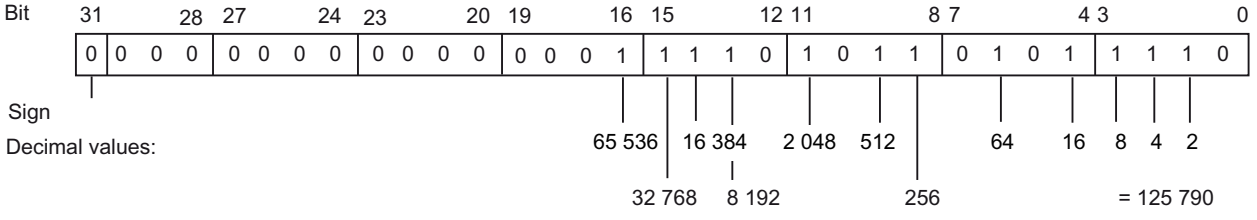
An operand of data type DINT occupies four BYTE in the memory.

The following table shows the properties of data type DINT:

Length (bits)	Format	Value range	Examples of value input
32	Signed integers (decimal system)	-2_147_483_648 to +2_147_483_647	• +125_790 • DINT#+125_790 • DINT#10#+125_790 • L#275
	Binary numbers (only positive)	2#0 to 2#0111_1111_1111_1111_1111_1111_1111_1111	• 2#0000_0000_0000_0001_1110_1011_0101_1110 • DINT#2#0000_0000_0000_0001_1110_1011_0101_1110 • DINT#2#10
	Octal numbers (only positive)	8#0 to 8#177_7777_7777	• 8#36_5536 • DINT#8#36_5536
	Hexadecimal numbers	16#0 to 16#7FFF_FFFF	• 16#0001_EB5E • DINT#16#0001_EB5E

Example

The following figure shows the integer +125790 as a binary number:



See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.3.6 UDINT (32-bit integers) (S7-1200, S7-1500)

Description

An operand of data type UDINT (Unsigned Double INT) has a length of 32 bits and contains unsigned numerical values.

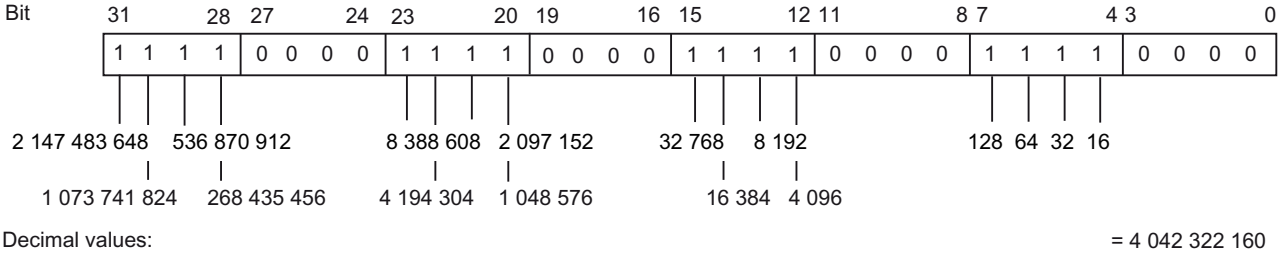
An operand of data type UDINT occupies four BYTE in the memory.

The following table shows the properties of data type UDINT:

Length (bits)	Format	Value range	Examples of value input
32	Unsigned integers (decimal system)	0 to 4_294_967_295	• 4_042_322_160 • UDINT#4_042_322_160 • UDINT#10#4_042_322_160
	Binary numbers	2#0 to 2#1111_1111_1111_1111_1111_1111_1111_1111	• 2#1111_0000_1111_0000_1111_0000_1111_0000 • UDINT#2#1111_0000_1111_0000_1111_0000_1111_0000 • UDINT#2#10
	Octal numbers	8#0 to 8#377_7777_7777	• 8#360_7417_0360 • UDINT#8#360_7417_0360
	Hexadecimal numbers	16#0 to 16#FFFF_FFFF	• 16#F0F0_F0F0 • UDINT#16#F0F0_F0F0

Example

The following figure shows the integer 4042322160 as a binary number:



See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.3.7 LINT (64-bit integers) (S7-1500)

Description

An operand of data type LINT (Long INT) has a length of 64 bits and consists of two components: a sign and a numerical value in the two's complement. The signal states of bits 0 to 62 represent the number value. The signal state of bit 63 represents the sign. The sign may assume "0" for the positive, or "1" for the negative signal state.

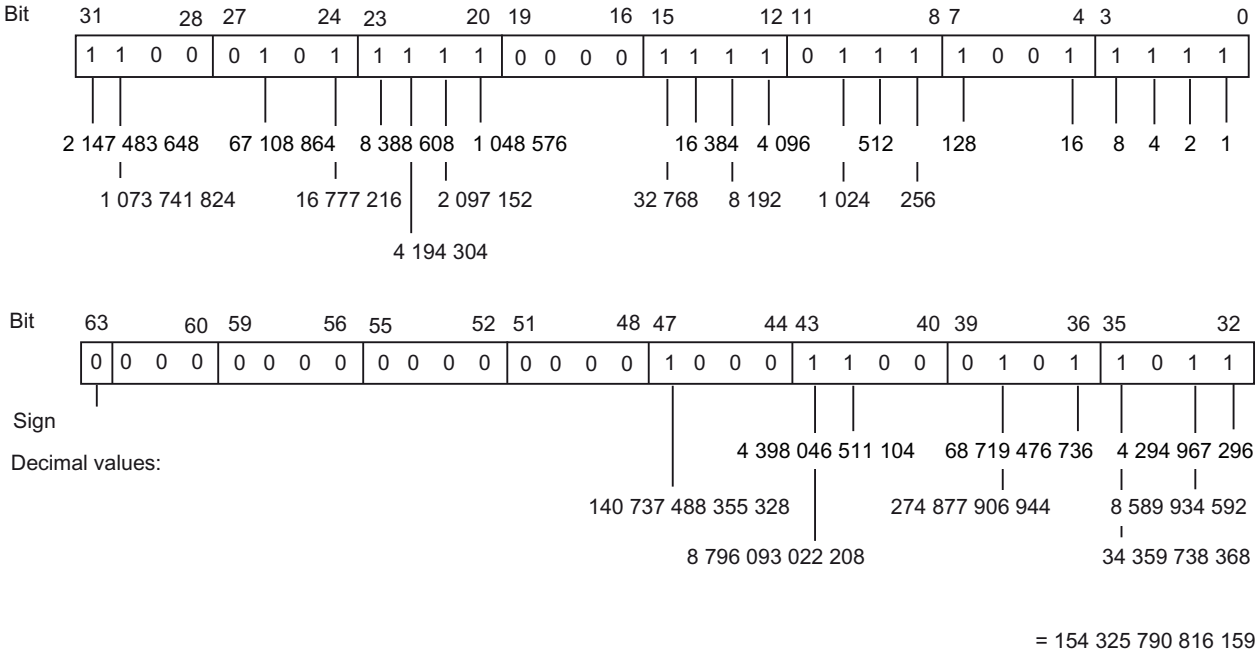
An operand of data type LINT occupies eight BYTE in the memory.

The following table shows the properties of data type LINT:

Length (bits)	Format	Value range	Examples of value input
64	Signed integers (decimal system)	-9_223_372_036_854_775_808 to +9_223_372_036_854_775_807	• +154_325_790_816_159 • LINT#+154_325_790_816_159 • LINT#10#+154_325_790_816_159
	Binary numbers (only positive)	2#0 to 2#0111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111	• 2#0000_0000_0000_0000_1000_1100_0101_1011_1100_0101_1111_0000_1111_0111_1001_1111 • LINT#2#0000_0000_0000_0000_1000_1100_0101_1011_1100_0101_1111_0000_1111_0111_1001_1111 • LINT#2#10
	Octal numbers	8#0 to 8#7_7777_7777_7777_7777_7777	• 8#4305_5705_7417_3637 • LINT#8#4305_5705_7417_3637
	Hexadecimal numbers (only positive)	16#0 to 16#7FFF_FFFF_FFFF_FFFF	• 16#0000_8C5B_C5F0_F79F • LINT#16#0000_8C5B_C5F0_F79F

Example

The following figure shows the integer +154325790816159 as a binary number:



See also

Overview of the valid data types (Page 2239)

Overview of data type conversion (Page 2330)

Implicit conversions (Page 2333)

Explicit conversions (Page 2382)

Data type conversion for S7-1200 (Page 2465)

11.3.3.8 ULINT (64-bit integers) (S7-1500)

Description

An operand of data type ULINT (Unsigned Long INT) has a length of 64 bits and contains unsigned numerical values.

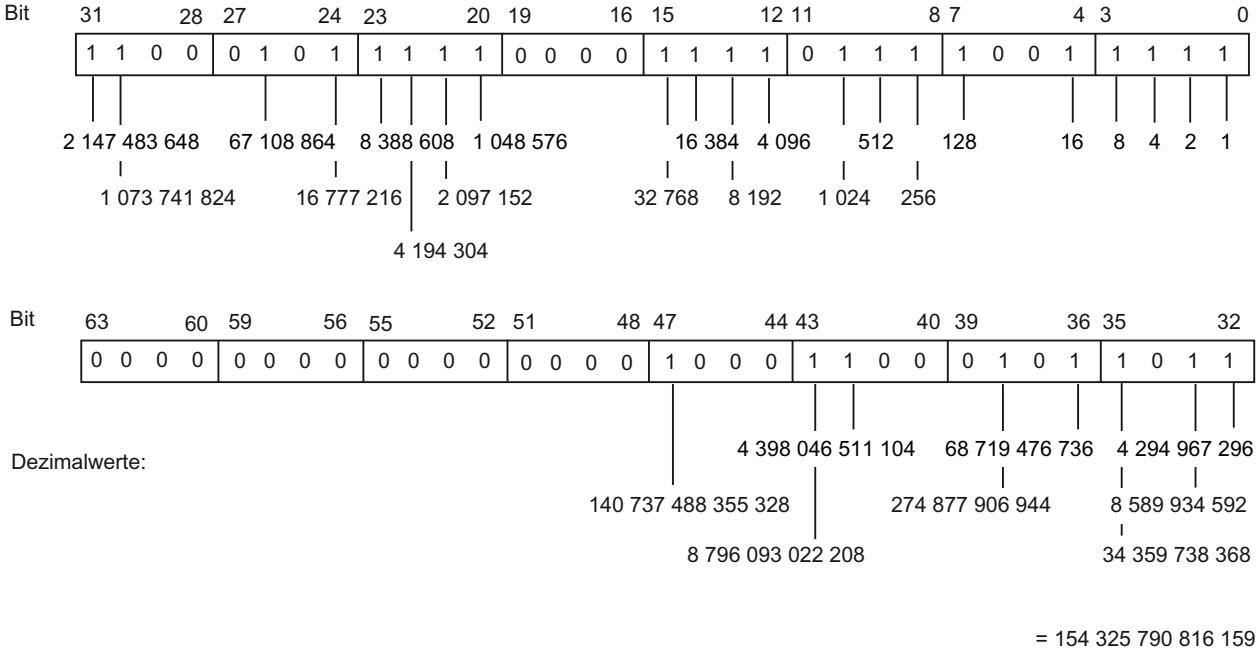
An operand of data type ULINT occupies eight BYTE in the memory.

The following table shows the properties of data type ULINT:

Length (bits)	Format	Value range	Examples of value input
64	Unsigned integers (decimal system)	0 to 18_446_744_073_709_551_615	• 154_325_790_816_159 • ULINT#154_325_790_816_159 • ULINT#10#154_325_790_816_159
	Binary numbers	2#0 to 2#1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111	• 2#0000_0000_0000_0000_1000_1100_0101_1011_1100_0101_1111_0000_1111_0111_1001_1111 • ULINT#2#0000_0000_0000_0000_1000_1100_0101_1011_1100_0101_1111_0000_1111_0111_1001_1111 • ULINT#2#10
	Octal numbers	8#0 to 8#17_7777_7777_7777_7777_7777	• 8#4305_5705_7417_3637 • ULINT#8#4305_5705_7417_3637
	Hexadecimal numbers	16#0 to 16#FFFF_FFFF_FFFF_FFFF	• 16#0000_8C5B_C5F0_F79F • ULINT#16#0000_8C5B_C5F0_F79F

Example

The following figure shows the integer 154325790816159 as a binary number:



See also

Overview of the valid data types (Page 2239)

Overview of data type conversion (Page 2330)

Implicit conversions (Page 2333)

Explicit conversions (Page 2382)

Data type conversion for S7-1200 (Page 2465)

11.3.4 Floating-point numbers

11.3.4.1 REAL

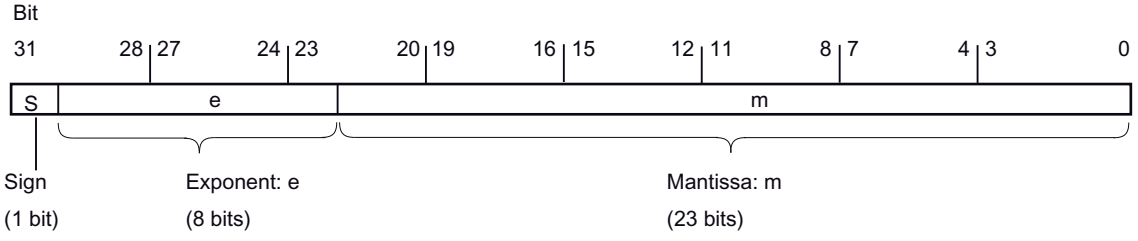
Description

Operands of the data type REAL have a length of 32 bits and are used to represent floating-point numbers. An operand of the REAL data type consists of the following three components:

- Sign: The sign is determined by the signal state of bit 31. The bit 31 assume the value "0" (positive) or "1" (negative).
- 8-bit exponents to basis 2: The exponent is increased by a constant (base, +127), so that it has a value range of 0 to 255.
- 23-bit mantissa: Only the fraction part of the mantissa is shown. The integer part of the mantissa is always 1 with normalized floating-point numbers and is not stored.

The REAL data type is processed with a precision of 6 digits.

The following figure shows the structure of the REAL data type:



Note

With floating-point numbers, only the precision defined by the IEEE754 standard is stored. Additionally specified decimals are rounded off according to IEEE754.

The number of decimal places may decrease for frequently nested arithmetic calculations.

If more decimal places are specified than can be stored by the data type, the number is rounded to the value corresponding to the precision allowed by this value range.

The following table shows the properties of data type REAL:

Length (bits)	Format	Value range	Examples of value input
32	Floating-point numbers according to IEEE754	-3.402823e+38 to -1.175495e-38 ±0.0 +1.175495e-38 to +3.402823e+38	1.0e-5; REAL#1.0e-5
	Floating-point numbers		1.0; REAL#1.0

See also

Overview of the valid data types (Page 2239)

Calculating with floating-point numbers (REAL and LREAL) in SCL (Page 5005)

Data type conversion for S7-1200 (Page 2465)

11.3.4.2 LREAL (S7-1200, S7-1500)

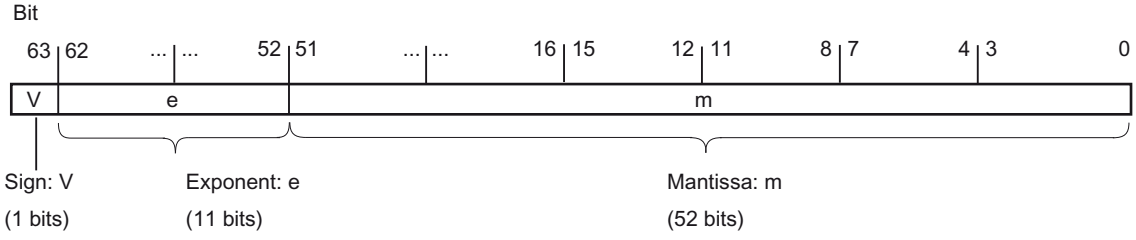
Description

Operands of the data type LREAL have a length of 64 bits and are used to represent floating-point numbers. An operand of the LREAL data type consists of the following three components:

- Sign: The sign is determined by the signal state of bit 63. The bit 63 assumes the value "0" (positive) or "1" (negative).
- 11-bit exponents to base 2: The exponent is increased by a constant (base, +1023), so that it has a value range of 0 to 2047.
- 52-bit mantissa: Only the fraction part of the mantissa is shown. The integer part of the mantissa is always 1 with normalized floating-point numbers and is not stored.

The LREAL data type is processed with a precision of 15 digits.

The following figure shows the structure of the LREAL data type:



The following table shows the properties of data type LREAL:

Length (bits)	Format	Value range	Examples of value input
64	Floating-point numbers according to IEEE754	-1.7976931348623158e+308 to -2.2250738585072014e-308 ±0.0	1.0e-5; LREAL#1.0e-5
	Floating-point numbers	+2.2250738585072014e-308 to +1.7976931348623158e+308	1.0; LREAL#1.0

Note

With floating-point numbers, only the precision defined by the IEEE754 standard is stored. Additionally specified decimals are rounded off according to IEEE754.

The number of decimal places may decrease for frequently nested arithmetic calculations.

If more decimal places are specified than can be stored by the data type, the number is rounded to the corresponding value of the precision allowed by this value range .

See also

Overview of the valid data types (Page 2239)

Calculating with floating-point numbers (REAL and LREAL) in SCL (Page 5005)

Data type conversion for S7-1200 (Page 2465)

11.3.4.3 Invalid floating-point numbers

Description

We distinguish between four number ranges for data types REAL and LREAL:

- normalized numbers stored with full accuracy
- denormalized numbers not stored with full accuracy
- Infinite numbers: +Inf/-Inf (infinity)
- Invalid numbers: NaN (Not a Number)

Note

Floating-point numbers are stored as specified by the IEEE754 standard. Results of conversion or arithmetic functions with a denormalized, infinite or NaN (Not a Number) floating point depend on the CPU.

If you are not working with normalized floating-point numbers in mathematical functions, the result will show significant differences depending on the series of the CPU which you are using.

A CPU cannot calculate with denormalized floating-point numbers, with the exception of older CPU versions of the S7-300 and S7-400 series. The bit pattern of a denormalized number is interpreted as a zero. If the result of calculation falls into this range, it is continued with zero; the status bits OV and OS are set (number range undershoot).

Even though the values of invalid floating-point numbers can only be displayed with limited accuracy for mathematical functions, numbers with an exponent of -39 (e.g. 2.4408e-039) can be monitored in the TIA Portal and therefore do not necessarily represent a faulty result. This means that floating-point values may be located outside the range of valid numerical values.

Note

The following applies to CPUs of the series S7-1200 V1, V2 and V3:

The comparison operation "Equal" uses the bit pattern of the invalid floating-point number. If two "NaN numbers" with the same bit pattern are compared, the output of the "Equal" comparison operation returns the result TRUE.

Note

The following applies to CPUs of the S7-1200 V4 and S7-1500 series:

If two invalid numbers (NaN) are compared with each other, the result is always FALSE, regardless of the bit pattern of the invalid number or the relation (>, >, ...).

Note

Comparison of denormalized floating-point numbers

For the comparison operation "Equal" with two denormalized floating-point numbers, the output for CPUs of the S7-300/400 series is set to the signal state "0" and for CPUs of the S7-1200/1500 series to the signal state "1".

If the input variables of a mathematical function represent an invalid floating-point number, an invalid floating-point number will also be output as result.

You have the following options to evaluate possible errors caused by invalid floating-point numbers:

- In LAD/FBD and SCL, you can query the enable output ENO for FALSE
- In STL, you can evaluate the status bit OV

See also

Overview of the valid data types (Page 2239)

Calculating with floating-point numbers (REAL and LREAL) in SCL (Page 5005)

Data type conversion for S7-1200 (Page 2465)

11.3.5 Timers

11.3.5.1 S5TIME (duration) (S7-300, S7-400)

Format

Data type S5TIME stores the duration in BCD format. The duration is the product from a time in the range 0 to 999 and a time basis. The time basis indicates the interval at which a timer decrements the time value by one unit until it reaches "0". The resolution of the times can be controlled via the time basis.

The following table shows the range of values for data type S5TIME:

Length (bits)	Format	Range of values	Examples of value input
16	S7 time in increments of 10 ms (default value)	S5T#0MS to S5T#2H_46M_30S_0MS	S5T#10s, S5TIME#10s
	Hexadecimal numbers	16#0 to 16#3999	16#2

The following table shows the time base coding for S5TIME:

Time basis	Binary code for time basis
10 ms	00
100 ms	01
1 s	10
10 s	11

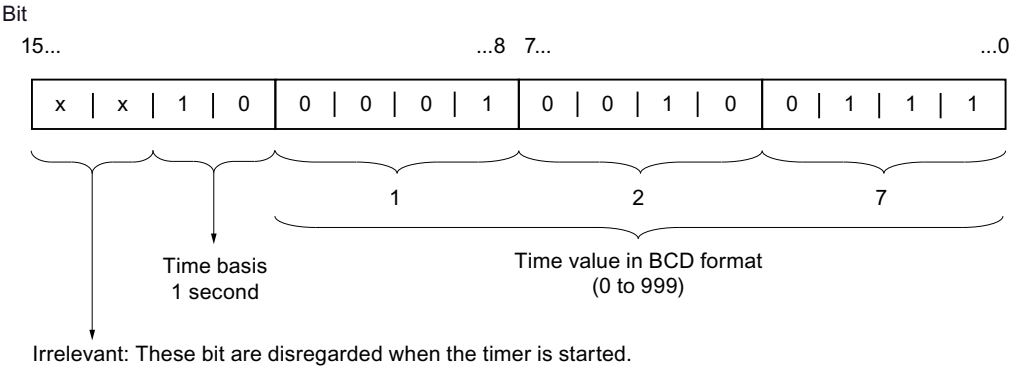
Always observe range limits and the resolution of time values when using data type S5TIME with timers. The table below specifies the range associated with each resolution:

Resolution	Range
0.01 s	10 ms to 9 s 990 ms
0.1 s	100 ms to 1 min 39 s 900 ms
1 s	1 s to 16 min 39 s
10 s	10 s to 2 h 46 min 30 s

Values that exceed 2h46m30s are not accepted.

Example

The following figure shows the content of the time operand for a time value of 127 and a time base of 1 s:



See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.5.2 TIME (IEC time)

Description

The contents of an operand of the data type TIME is interpreted as milliseconds. The representation contains information for days (d), hours (h), minutes (m), seconds (s) and milliseconds (ms).

The following table shows the properties of data type TIME:

Length (bits)	Format	Value range	Examples of value input
32	Signed duration	T#-24d_20h_31m_23s_648ms to T# +24d_20h_31m_23s_647ms	T#10d_20h_30m_20s_630ms, TIME#10d_20h_30m_20s_630ms
	Hexadecimal numbers	16#0000_0000 to 16#FFFF_FFFF	16#0001_EB5E

It is not necessary to specify all time units. T#5h10s is a valid entry, for example. If only one unit is specified, the absolute value of days, hours, and minutes must not exceed the high or low limits. When more than one time unit is specified, the value must not exceed 24 days, 23 hours, 59 minutes, 59 seconds or 999 milliseconds.

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.5.3 LTIME (IEC time) (S7-1500)

Description

The contents of an operand of data type LTIME is interpreted as nanoseconds. The representation contains information for days (d), hours (h), minutes (m), seconds (s) and milliseconds (ms), microseconds (us) and nanoseconds (ns).

The following table shows the properties of data type LTIME:

Length (bits)	Format	Value range	Examples of value input
64	Signed duration	LT#-106751d_23h_47m_16s_854ms_775us_808ns to LT#+106751d_23h_47m_16s_854ms_775us_807ns	LT#11350d_20h_25m_14s_830ms_652us_315ns, LTIME#11350d_20h_25m_14s_830ms_652us_315ns
	Hexadecimal numbers	16#0 to 16#7FFF_FFFF_FFFF_FFFF	16#2

It is not necessary to specify all time units. LT#5h10s is therefore a valid entry, for example. If only one unit is specified, the absolute value of days, hours, and minutes must not exceed the high or low limits. When more than one time unit is specified, the value must not exceed 106751 days, 23 hours, 59 minutes, 59 seconds, 999 milliseconds, 999 microseconds or 999 nanoseconds.

See also

Overview of the valid data types (Page 2239)

Overview of data type conversion (Page 2330)

Implicit conversions (Page 2333)

Explicit conversions (Page 2382)

Data type conversion for S7-1200 (Page 2465)

11.3.6 Date and time

11.3.6.1 DATE

Format

The DATE data type saves the date as an unsigned integer. The representation contains the year, the month, and the day.

The contents of an operand of DATE data type correspond in hexadecimal format to the number of days since 01-01-1990 (16#0000).

The following table shows the properties of data type DATE:

Length (bytes)	Format	Value range	Examples of value input
2	IEC date (Year-Month-Day)	D#1990-01-01 to D#2169-06-06	D#2009-12-31, DATE#2009-12-31
	Hexadecimal numbers	16#0000 to 16#FFFF	16#00F2

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.6.2 TIME_OF_DAY (TOD)

Format

Data type TOD (TIME_OF_DAY) occupies a double word and stores the number of milliseconds since the beginning of the day (0:00 h) as unsigned integer.

The following table shows the properties of data type TOD:

Length (bytes)	Format	Value range	Examples of value input
4	Time-of-day (hours:minutes:seconds.milliseconds)	TOD#00:00:00.000 to TOD#23:59:59.999	TOD#10:20:30.400, TIME_OF_DAY#10:20:30.400

You always need to specify the hours, minutes and seconds. The specification of milliseconds is optional.

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.6.3 LTOD (LTIME_OF_DAY) (S7-1500)

Format

Data type LTOD (LTIME_OF_DAY) occupies two double words and stores the number of nanoseconds since the beginning of the day (0:00 h) as unsigned integer.

The following table shows the properties of data type LTOD:

Length (bytes)	Format	Value range	Examples of value input
8	Time-of-day (hours:minutes:seconds.nanoseconds)	LTOD#00:00:00.000000000 to LTOD#23:59:59.999999999	LTOD#10:20:30.400_365_215, LTIME_OF_DAY#10:20:30.400_365_215

You always need to specify the hours, minutes and seconds. The specification of nanoseconds is optional.

See also

Overview of the valid data types (Page 2239)

Overview of data type conversion (Page 2330)

Implicit conversions (Page 2333)

Explicit conversions (Page 2382)

Data type conversion for S7-1200 (Page 2465)

11.3.6.4 DATE_AND_TIME (date and time of day)

Format

The DT (DATE_AND_TIME) data type saves the information on date and time of day in BCD format.

The following table shows the properties of data type DT:

Length (bytes)	Format	Value range	Example of value input
8	Date and time (year-month-day-hour:minute:second:millisecond [3])	Min.: DT#1990-01-01-00:00:00.000 Max.: DT#2089-12-31-23:59:59.999	DT#2008-10-25-08:12:34.567, DATE_AND_TIME#2008-10-25-08:12:34.567

The following table shows the structure of the DT data type:

Byte	Contents	Value range
0	Year	0 to 99 (Years 1990 to 2089) BCD#90 = 1990 ... BCD#0 = 2000 ... BCD#89 = 2089
1	Month	BCD#1 to BCD#12
2	Day	BCD#1 to BCD# 31
3	Hour	BCD#0 to BCD#23
4	Minute	BCD#0 to BCD#59
5	Second	BCD#0 to BCD#59
6	The two most significant digits of MSEC	BCD#0 to BCD#999
7 (4MSB) [1]	The least significant digit of MSEC	BCD#0 to BCD#9
7 (4LSB) [2]	Weekday	BCD#1 to BCD#7 BCD#1 = Sunday ... BCD#7 = Saturday
[1] MSB: Most significant bit [2] LSB: Least significant bit [3] Fixed point number		

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.6.5 LDT (DATE_AND_LTIME) (S7-1500)

Format

Data type LDT (DATE_AND_LTIME) stores the date and time-of-day information in nanoseconds since 01/01/1970 0:0.

The following table shows the properties of data type LDT:

Length (bytes)	Format	Value range	Example of value input
8	Date and time (Year-Month-Day-Hour:Minute:Second.Nanoseconds)	Min.: LDT#1970-01-01-00:00:00.000000000 Max.: LDT#2262-04-11-23:47:16.854775807	LDT#2008-10-25-08:12:34.567
	Hexadecimal numbers	16#0 to 16#7FFF_FFFF_FFFF_FFFF	16#7FFF

See also

Overview of the valid data types (Page 2239)

Overview of data type conversion (Page 2330)

Implicit conversions (Page 2333)

Explicit conversions (Page 2382)

Data type conversion for S7-1200 (Page 2465)

11.3.6.6 DTL (S7-1200, S7-1500)

Description

An operand of data type DTL has a length of 12 bytes and stores date and time information in a predefined structure.

The following table shows the properties of data type DTL:

Length (bytes)	Format	Value range	Example of value input
12	Date and time (Year-Month-Day-Hour:Minute:Second.Nanoseconds)	Min.: DTL#1970-01-01-00:00:00.0 Max.: DTL#2262-04-11-23:47:16.854775807	DTL#2008-12-16-20:30:20.250

The structure of data type DTL consists of several components each of which can contain a different data type and range of values. The data type of a specified value must match the data type of the corresponding components.

Note

Invalid monitor value of DTL tags in hexadecimal format

If the monitor value of the DTL tags is represented in hexadecimal format, this can be because one of the values (YEAR, MONTH, DAY, etc.) is invalid. For example, this is the case if a value > 24 was specified at the HOUR tag.

The following table shows the structure components of data type DTL and their properties:

Byte	Component	Data type	Value range
0	Year	UINT	1970 to 2262
1			
2	Month	USINT	1 to 12
3	Day	USINT	1 to 31
4	Weekday	USINT	1(Sunday) to 7(Saturday) The weekday is not considered in the value entry.
5	Hour	USINT	0 to 23
6	Minute	USINT	0 to 59
7	Second	USINT	0 to 59
8	Nanosecond	UDINT	0 to 999999999
9			
10			
11			

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

11.3.7 Character strings

11.3.7.1 Character

CHAR

Description

A tag of the CHAR (Character) data type has a length of 8 bits and occupies one BYTE of memory.

The CHAR data type stores a single character in ASCII coding. You can find information on the encoding of special characters under "See also > STRING".

The following table shows the value range of the CHAR data type:

Length (bits)	Format	Value range	Example of value inputs
8	ASCII characters	ASCII character set	'A', CHAR#'A'

See also

Overview of the valid data types (Page 2239)

STRING (Page 2270)

WCHAR (S7-1200, S7-1500)

Description

A tag of the data type WCHAR (Wide Characters) has a length of 16 bits and occupies two BYTE of memory.

The WCHAR data type saves a single character of an expanded character set in UFT-16 coding. However, only a subset of the entire Unicode range is covered. Characters that cannot be displayed are made displayable using an escape character.

The following table shows the value range of the WCHAR data type:

Length (bits)	Format	Value range	Example of value input
16	Unicode	$0000 - $D7FF	WCHAR#'a'

See also

Overview of the valid data types (Page 2239)

11.3.7.2 Character strings

STRING

Description

An operand of the STRING data type saves several characters in a character string that can consist of up to 254 characters. In a character string, all characters of the codepage created on the system are permitted. The characters are specified in single quotation marks.

A character string can also contain special characters. The escape character $ is used to identify control characters, dollar signs and single quotation marks.

Note

Different code pages

Please note that the special characters are coded using the code page currently set in Windows. This means that a string that contains special characters can be displayed differently on a different operating system with a different code page.

The dependency of the codepage on the created system makes an international use of the user program more difficult. Only the characters from the 7-bit ASCII coding are internationally valid.

The following table shows the properties of a STRING tag:

Length (bytes)	Format	Value range	Example of value input
n + 2 [1]	ASCII character string incl. special characters	0 to 254 characters	• 'Name' • STRING#'NAME' • STRING#'Na... (The actual length of the string is longer than the space available on the screen.) • STRING#'' (The string is empty.)
[1] An operand of the STRING data type occupies two bytes more than the specified maximum length in the memory.			

The table below shows examples for the notation of special characters:

Character	Hex	Meaning	Example
$L or $l	0A	Line feed	'$LText', '$0AText'
$N	0A and 0D	Line break The line break occupies 2 characters in the character string and is converted to RL in the display in the editor.	'$NText', '$0A$0DText'
$P or $p	0C	Page feed	'$PText', '$0CText'
$R or $r	0D	Carriage return (CR)	'$RText','$0DText'

Character	Hex	Meaning	Example
$T or $t	09	Tab	'$TText', '$09Text'
$$	24	Dollar sign	'100$$', '100$24'
$'	27	Single quotation mark	'$'Text$'','$27Text$27'

If the escape character $ is followed by a letter from the table, the character indicated in the table is entered in the string. If the escape character $ is followed by a letter that is not in the table, this letter is entered in the string. If the escape character $ is followed by two hexadecimal digits or only one digit, this code is entered in the string.

Use in the watch table

The following applies to a CPU of the S7-300/400 series: If a tag of the STRING data type is being monitored, only the first 30 characters will be displayed. If the actual length is greater than 30 characters, an ellipsis (…) is displayed instead of the final apostrophe ('). STRING values with more than 30 characters cannot be used for modifying.

Maximum length of a character string

The maximum length of the character string can be specified during the declaration of an operand using square brackets after the keyword STRING (for example, STRING[4]). You can also use local or global constants to declare the maximum length (for example, STRING[#loc_const] or STRING["glob_const"]). If the specification of the maximum length is omitted, the standard length of 254 characters is set for the respective operand.

You can find additional information about using local or global constants to declare the maximum length here:

- Declaring the block interface: Declare variables of the STRING and WSTRING data types (Page 4778)
- Programming data blocks: Declaring tags of the STRING data type (Page 4806)
- AUTOHOTSPOT

If the actual length of a specified character string is shorter than the declared maximum length, the characters are written to the character string left-justified and the remaining character spaces remain undefined. Only occupied character spaces that define the actual length of the string are considered for the value processing and all displays.

Note

For S7-300/400 CPUs, please note: If a temporary tag of the STRING data type was defined, you must describe the BYTE "Max. length of string" with the defined length before you use the tags in the user program.

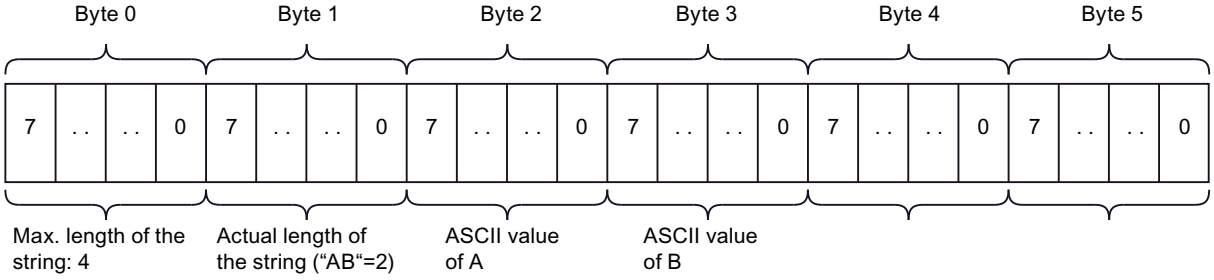
Transferring a STRING for parameter supply

You can transfer the STRING data type as a parameter. You can find additional information on parameter supply with STRING here:

- Transferring a tag of the STRING or WSTRING data type (Page 2277)
- For S7-1200/1500: Valid data types in the block interface (Page 4768)
- For S7-300/400: Valid data types in the block interface (Page 4767)

Example

The example below shows the byte sequence if the STRING[4] data type is specified with output value 'QB':



See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

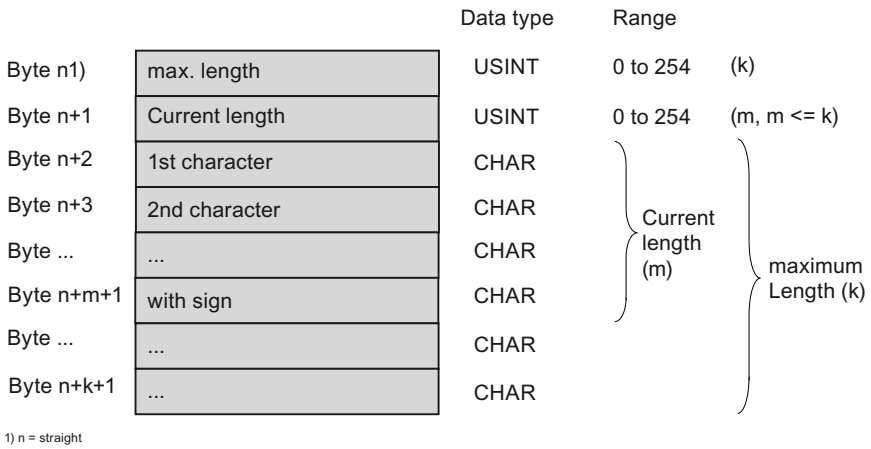
Structure of a STRING tag

Introduction

A tag with STRING data type (character string) is maximum 256 characters long with 254 bytes user data. It starts in a non-optimized block at a word limit (at a byte with even address). In an optimized block it can start at any byte limit.

When tags are created their maximum length is specified. In the case of the preassignment or editing of the character string, the current length (the actually used length of the character string = number of valid characters) is entered. The maximum length is located in the first byte of the character string. The current length is in the second byte. This is followed by the characters, coded according to the codepage set in Windows.

Structure of a STRING tag



WSTRING (S7-1200, S7-1500)

Description

An operand of data type WSTRING (Wide Character String) stores several Unicode characters of data type WCHAR in one character string. If you do not specify a length, the character string has a preset length of 254 characters. In a character string, all characters that are supported by the operating system are permitted. This means you can also use Chinese characters in a character string. Windows supports only some (but sufficient number) of the characters defined in Unicode.

When declaring an operand of data type WSTRING you can define its length using square brackets (for example WSTRING[10]). If you do not specify a length, the length of the WSTRING is set to 254 characters by default. You can declare a length of up to 16382 characters (WSTRING[16382]).

The specification of the characters occurs in single quotes and always with the qualifier WSTRING#.

The following table shows the properties of a WSTRING tag:

Length (WORD)	Format	Value range	Example of value input
n + 2 [1]	Unicode character string; n specifies the length of the character string.	Preset value: 0 to 254 characters Max. possible value: 0 to 16382	• WSTRING#'Hello World' • WSTRING#'Hello Wo... (The actual length of the string is longer than the space available on the screen.) • WSTRING#'' (The string is empty.)
[1] An operand of the WSTRING data type occupies two WORDs more in the memory than the specified maximum length.			

A character string can also contain special characters. The escape character $ is used to identify control characters, dollar signs and single quotation marks.

The table below shows examples for the notation of special characters:

Character	Hex	Meaning	Example
$L or $l	000A	Line feed	'$LText', '$000AText'
$N	000A and 000D	Line break The line break occupies 2 characters in the character string and is converted to RL in the display in the editor.	'$NText', '$000A $000DText'
$P or $p	000C	Page feed	'$PText', '$000CText'
$R or $r	000D	Carriage return (CR)	'$RText','$000DText'
$T or $t	0009	Tab	'$TText', '$0009Text'
$$	0024	Dollar sign	'100$$', '100$0024'
$'	0027	Single quotation mark	'$'Text$'','$0027Text $0027'

If the escape character $ is followed by a letter from the table, the character indicated in the table is entered in the string. If the escape character $ is followed by a letter that is not in the table, this letter is entered in the string. If the escape character $ is followed by four hexadecimal digits, this code is entered in the string.

Note

Conversion of WSTRING tags

Implicit conversion of the WSTRING data type is not possible. Explicit conversion of the WSTRING data type to STRING is generally possible. However, as standard, it is only possible to convert characters in the code range from 0 - 127 in all Windows code pages. For all characters outside this range, the code page character and the lower byte of the Unicode character must be in exactly the same position for the conversion to work without errors.

Use in the watch table

If a tag of the WSTRING data type is being monitored, only the first 254 characters are displayed. If the actual length is greater than 254 characters, an ellipsis (…) is displayed instead of the closing apostrophe ('). WSTRING values with more than 254 characters cannot be used for modifying.

Maximum length of a character string

The maximum length of the character string can be specified during the declaration of an operand using square brackets after the keyword WSTRING (for example, WSTRING[4]). You can also use local or global constants to declare the maximum length (for example, WSTRING[#loc_const] or WSTRING["glob_const"]). If the specification of the maximum length is omitted, the standard length of 254 characters is set for the respective operand.

You can find additional information about using local or global constants to declare the maximum length here:

- Declaring the block interface: Declare variables of the STRING and WSTRING data types (Page 4778)
- Programming data blocks: Declaring tags of the STRING data type (Page 4806)
- AUTOHOTSPOT

If the actual length of a specified character string is shorter than the declared maximum length, the characters are written to the character string left-justified and the remaining character spaces remain undefined. Only occupied character spaces are considered in the value processing.

Transferring a WSTRING for parameter supply

Operands of the data type WSTRING can be transferred as parameters up to the maximum length for blocks with "optimized" access.

For function blocks (FB) with "standard" access, operands of the data type WSTRING can be declared as parameters in all sections of the block interface except in the section "InOut". For a function (FC) with "standard" access, only operands of the WSTRING data type can be transferred as parameters.

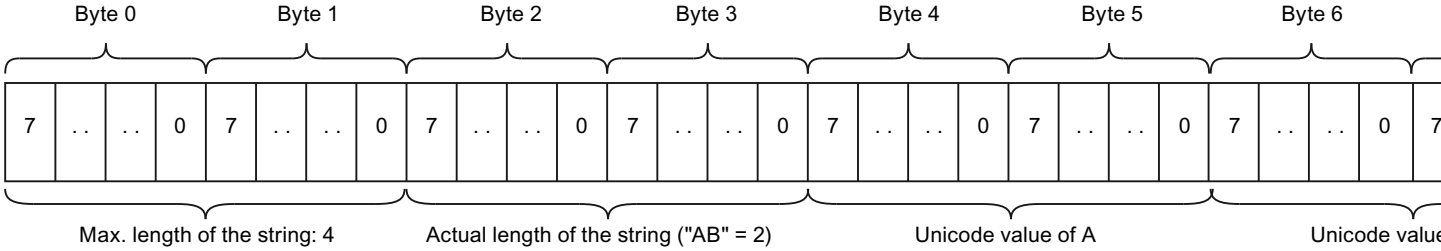
The function value of an FC in the "Return" section and expressions in the SCL programming language are another exception to this rule. In these cases, the WSTRING tag must not be longer than 1022 characters.

The declared lengths of formal and actual parameters may be different. You can find additional information on parameter supply with WSTRING here:

- Transferring a tag of the STRING or WSTRING data type (Page 2277)
- Valid data types in the block interface (Page 4768)

Example

The example below shows the byte sequence if the WSTRING[4] data type is specified with output value 'QB':



See also

Overview of the valid data types (Page 2239)

Addressing individual characters of a STRING or WSTRING (S7-1200, S7-1500)

Addressing individual characters of a STRING or WSTRING

Use the syntax `StringName[i]` to access an individual character of a STRING or WSTRING tag. The counting index "i" begins with "1". Thus, you access the first character of the string with StringName[1].

You cannot access individual characters of a STRING or WSTRING constant.

Examples

The following examples show the addressing of (W)STRINGs:

Addressing	Explanation
"myDB".mySTRING[3]	Addresses the third character of the STRING in the data block.
#myWSTRING[3]	Addresses the third character of the WSTRING.

Troubleshooting W(STRING) access

Access errors result during runtime when you access a character that is located outside the STRING length. On read access to the character string, you receive the character '$00' or '$0000'; write access to the character string is not executed. If the instruction has the enable output ENO, ENO is set to the signal state FALSE. The CPU does not change to STOP.

The only exception is when the character is written directly after the actual length of the character string.

The following example shows the string 'Hello' with the actual length of 5. The 27th character of the STRING lies outside the actual length and can therefore not be written. The STRING remains unchanged; the result of the assignment is 'hello'.

SCL

```
MyDB.mystring := 'hello';
MyDB.mystring[27] := CHAR_TO_BYTE('!');
```

The following example shows the specified exception: The character is written directly after the STRING in the 6th character. The result of the assignment is 'hello!'.

SCL

```
MyDB.mystring := 'hello';
MyDB.mystring[6] := CHAR_TO_BYTE('!');
```

Where possible, use instructions from the "Extended instructions > String + Char" pane to handle STRINGs.

```
CONCAT(IN1 := 'hello', IN2 := '!');
```

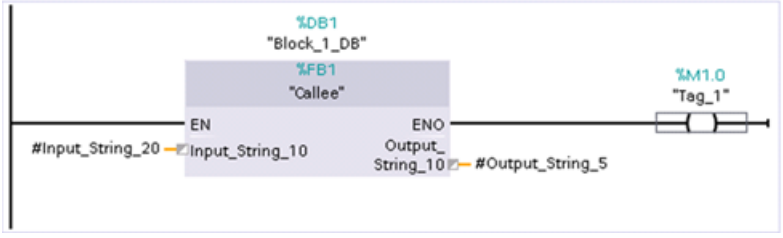
Transferring a tag of the STRING or WSTRING data type

Description

You can transfer tags of the STRING or WSTRING data type as parameters. The following table shows the rules that apply to the (W)STRING transfer in the various CPU families:

CPU family	Data type	Rules for transfer at the block call
S7-300/400	STRING	The declared lengths of formal and actual parameters must be identical.
S7-1200/1500	STRING WSTRING	The declared lengths of formal and actual parameters may be different. If the declared length of the target parameter during runtime is insufficient to accept the (W)STRING, the (W)STRING is truncated and the enable output ENO is set to FALSE. In the programming editor, a gray rectangle at the parameter indicates that (W)STRING might be truncated during runtime. Exception: When calling STL blocks, the declared lengths of formal and actual parameters must always be identical.

The following figure shows a block call in which the lengths of the declared formal and actual parameters differ. Due to the different declared lengths, both "Input_String_20" and "Output_String_10" could be truncated during runtime.



11.3.8 PLC data types (UDT)

11.3.8.1 Basics of PLC data types (UDT)

Description

A PLC data type (UDT) is a complex, user-defined data type that is used for the declaration of a tag. It represents a data structure made up of several components of different data types. The components can also be derived from another PLC data type, from an ARRAY or directly using the STRUCT keyword as a structure. The nesting depth is hereby limited to 8 levels.

You can change a PLC data type (UDT) centrally and use it repeatedly in the program code. All locations of use are updated automatically.

Advantages of the PLC data types:

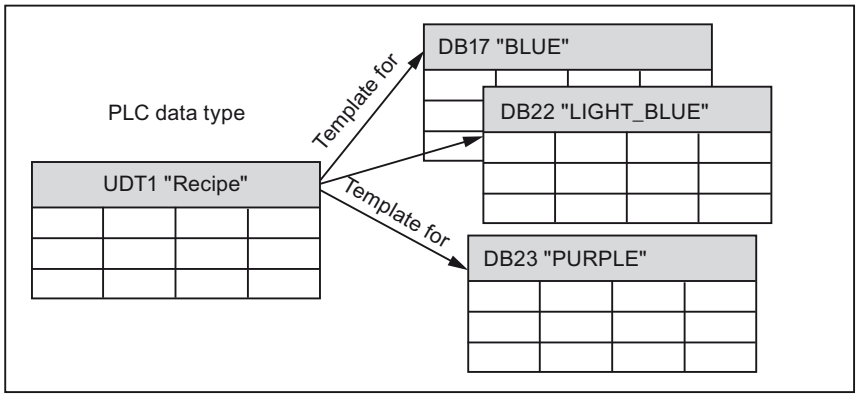
- Simple data exchange via block interfaces among several blocks
- Group data according to the process control
- Transferring a parameter as a data unit

Using a PLC data type

You can declare PLC data types as a type when creating data blocks. Based on this type, you can create a number of data blocks, all of which have the same data structure. These data blocks can be adapted by entering different actual values for the corresponding task.

For instance, create a PLC data type for a recipe for blending paints. You can then assign this data type to several data blocks, each of which contains different quantity information.

The following figure shows this application:



PLC data types can be used for the following applications:

- PLC data types can be used as data types for tags in the tag declaration of logic blocks or in data blocks.
- PLC data types can be used as templates for the creation of global data blocks with identical data structures.
- PLC data types can be used in S7-1200 and S7-1500 as a template for the creation of structured PLC tags.

Nesting depth and potential number of PLC data types

A maximum of 64K objects of FCs, FBs, DBs and PLC data types are defined (the total number is restricted accordingly for CPUs, however).

A maximum of 16K components can be defined per hierarchy level for each structure description (for example, a DB or a PLC data type (UDT)). These components can also be an ARRAY or a STRUCT data type. When a structured component is hereby derived from an explicitly declared PLC data type, there is no restriction. In other words, 16K components of PLC data type (UDT)/SDT are supported for each hierarchy level. However, a maximum of 252 of the 16K components can have "anonymous structures".

PLC data types can be nested to a depth of 8 hierarchy levels. This nesting depth is independent of the CPU used.

Generating external sources from blocks

When generating external sources from blocks, the changes made directly in the block interface to the default values of PLC data types are not exported to the sources. This means that these values are not available when the sources are imported once again. The default values are applied instead. To prevent this loss of data for the modified default values, the changes must be made directly in the PLC data type and not in the block interface. In this case, the changes are also exported when external sources are generated.

Disadvantages of anonymous structures

The components of the structured tag can be identically addressed. This is regardless of whether the declaration has been made using a PLC data type or an anonymous structure.

The following disadvantages apply when using anonymous structures:

- Reusability of the same structure by copying. This makes it difficult to change the structure
- Anonymous structures are not compatible with PLC data types (UDT) of the same structure
- Performance, since the matching types for structural components are checked during runtime
- The high limit is more easily reached when using anonymous structures, since all components must be evaluated individually.

Example

The following example shows the data type definition of the "MyUDT" PLC data type:

myUDT

	Name	Datentyp
1	FirstEntry	Struct
2	Value1	Int
3	Value2	Real
4	Value3	"myOtherUDT"
5	Station1	Int
6	Station2	UInt
7	Station3	DInt

Additional information on the PLC data types (UDTs)

Note

Declaring PLC data types (UDTs):

Structure of the declaration table for PLC data types (Page 4867)

Programming recommendations for PLC data types:

Using PLC data types (UDT) (Page 2229)

Using the DB_ANY data type (Page 2218)

Comparison of PLC data types (UDTs) in the program:

Comparing PLC data types (Page 5117)

Note

You can find more information on the PLC data types in the Siemens Industry Online Support in the following contributions:

How can you initialize structures in optimized memory areas at the S7-1500 in STEP 7 (TIA Portal)?

https://support.industry.siemens.com/cs/ww/de/view/78678760 (https://support.industry.siemens.com/cs/ww/en/view/78678760)

How do you create a PLC data type (UDT) at an S7-1500 control system?

https://support.industry.siemens.com/cs/ww/de/view/67599090 (https://support.industry.siemens.com/cs/ww/en/view/67599090)

How is the specific application of own PLC data types (UDT) effected in STEP 7 (TIA Portal)?

https://support.industry.siemens.com/cs/de/de/view/67582844 (https://support.industry.siemens.com/cs/de/en/view/67582844)

Why should complete structures be transferred at a block call for S7-1500 instead of many individual components?

https://support.industry.siemens.com/cs/ww/de/view/67585079 (https://support.industry.siemens.com/cs/ww/en/view/67585079)

See also

Addressing PLC data types (UDT) (Page 2281)

Declaring tags based on a PLC data type (Page 4778)

Declaring tags based on a PLC data type (Page 4806)

Comparing PLC tags (Page 5086)

Creating structured PLC tags (Page 4851)

Permissible addresses and data types of PLC tags (Page 4840)

CMP ==: Equal (Page 2662)

Transferring a tag of the PLC data type (UDT) (Page 2281)

11.3.8.2 Transferring a tag of the PLC data type (UDT)

Description

You can also transfer tags that are declared as PLC data type (UDT) as actual parameters. If the formal parameter is declared as a PLC data type (UDT) in the tag declaration, you must transfer a tag that has the same PLC data type (UDT) as an actual parameter.

An element of a tag declared using the PLC data type (UDT) can also be transferred as an actual parameter for the block call, provided the data type of the element of the tag matches the data type of the formal parameter.

Structures and user-defined PLC data types can be assigned to one another if they have identical structures. This means that the data types and the order of all structure components must be identical. In nested structures, the data types and the order of the lower-level structures and their components must also match.

Two different PLC data types can also be assigned to one another if the data types and the order of all structure components including the lower-level structures are identical. The names of the PLC data types do not have to match.

User-defined PLC data types and structures cannot be assigned to system data types.

11.3.8.3 Addressing PLC data types (UDT)

Addressing data elements of an PLC data type

The following syntax is used to access elements of a PLC data type:

```
<PLCDataTypeName>.<ElementName>
```

Examples

The following examples show the addressing of structured data type tags:

Addressing	Description
Values.Temperature	Addressing of the "Temperature" element in the "Values" tag, which is based on a PLC data type.

11.3.9 STRUCT data structure (anonymous structures)

11.3.9.1 Basic information on STRUCT

Description

The STRUCT data type represents a data structure that consists of a fixed number of components of different data types. Components of STRUCT or ARRAY data type can also be nested in a structure. Structures can be used to group data according to the process control system and to transfer parameters as one data unit.

Structure declarations that are used directly at the tag are called anonymous structures. An anonymous structure can have the following form:

Global_DB

	Name	Data type
1	Static	
2	Porsche	Struct
3	Motor	Struct
4	Valves	Array[0..5] of Bool
5	Speed_Min	Byte
6	Speed_Max	Byte
7	Temperature	Word
8	Axes	Array[0..4] of Bool
9	Oil	Byte
10	Carriage	Struct
11	Tuning	Struct
12	BMW	Struct
13	Audi	Struct

All of the subsequent explanations refer to this structure image.

Nesting depth and potential number of structures

A nested structure contains at least on additional structure as component. In each case 65535 FCs, FBs, DBs and PLC data types can be defined. For each CPU there is a special high limit relating to the block types and the total of all blocks.

A maximum of 16K components can be defined per hierarchy level for each structure description (for example, a DB or a PLC data type (UDT)). These components can also be an ARRAY or a STRUCT data type. When a structured component is hereby derived from an explicitly declared PLC data type, there is no restriction. For each hierarchy level, 16K components of PLC data type (UDT)/SDT are supported. However, a maximum of 252 "anonymous structures" may exist within a block.

Structures (STRUCT) can be nested to a depth of 8 hierarchy levels. This nesting depth is independent of the CPU used.

Transferring a STRUCT for parameter supply

You can transfer the STRUCT data type as a parameter. You can find additional information on parameter supply with STRUCT here:

Transferring tags of the STRUCT data type (Page 2285)

Disadvantages of anonymous structures

The components of the structured tag can be identically addressed. This is regardless of whether the declaration has been made using a PLC data type or an anonymous structure.

The following disadvantages apply when using anonymous structures:

- Increased maintenance costs: If an anonymous structure was copied multiple times, then it must also have been changed multiple times during a change.
- Anonymous structures are not compatible with PLC data types (UDT) of the same structure
- Performance, since the matching types of all structural components are checked
- Increased memory requirement: Each anonymous structure is a separate object, whose description is loaded to the AS.

Example

If you declare the tag of the data type STRUCT in a PLC data type (UDT), more usage options are available to you (see figure on left). However, you can also declare the tag directly with the data type STRUCT (see figure on right).

Declaration of the structured tag "Motor" with or without PLC data type (UDT):

Structured tag with PLC data type (UDT)

Global_DB

	Name	Data type
1	Static	
2	Porsche	"Vehicle"
3	Motor	Struct
4	Valves	Array[0..5] of Bool
5	Speed_Min	Byte
6	Speed_Max	Byte
7	Temperature	Word
8	Axes	Array[0..4] of Bool
9	Oil	Byte
10	Carriage	Struct
11	Tuning	Struct
12	BMW	"Vehicle"
13	Audi	"Vehicle"

Structured tag without PLC data type (UDT)

Global_DB

	Name	Data type
1	Static	
2	Porsche	Struct
3	Motor	Struct
4	Valves	Array[0..5] of Bool
5	Speed_Min	Byte
6	Speed_Max	Byte
7	Temperature	Word
8	Axes	Array[0..4] of Bool
9	Oil	Byte
10	Carriage	Struct
11	Tuning	Struct
12	BMW	Struct
13	Audi	Struct

See also

Overview of the valid data types (Page 2239)

11.3.9.2 Structure of a STRUCT tag

Introduction

A STRUCT tag always starts in a non-optimized block at a word limit, i.e. at a byte with even address. The individual components are then located in the order of their declaration in the memory. STRUCT tags occupy the memory up to the next word limit.

Components with BOOL data type start in the least significant bit. Components with BYTE and CHAR data type start in the next byte. Components with different data type start at a word limit. In optimized blocks the structures are stored so that all components are aligned according to their size. The order in the memory does not correspond to the display in the editor.

A nested structure is a structure that is a component of another structure. A nesting depth of maximum 8 structures is possible. All components can be separately addressed. The individual names of components are separated by a dot.

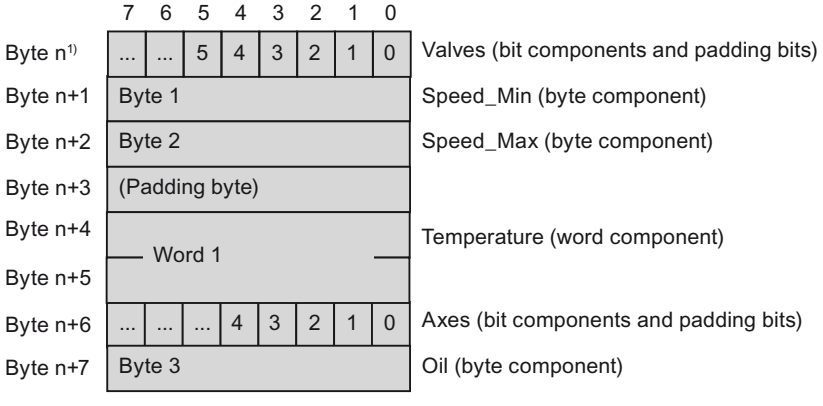
Structure of a STRUCT tag with elementary and complex data types in the non-optimized memory area:

- Declaration of the "Motor" STRUCT tag

Global_DB

	Name	Data type
1	Static	
2	Porsche	Struct
3	Motor	Struct
4	Valves	Array[0..5] of Bool
5	Speed_Min	Byte
6	Speed_Max	Byte
7	Temperature	Word
8	Axes	Array[0..4] of Bool
9	Oil	Byte
10	Carriage	Struct
11	Tuning	Struct
12	BMW	Struct
13	Audi	Struct

- Required memory of the "Motor" STRUCT tag



1) n = even

11.3.9.3 Addressing STRUCT components

Addressing components in structures

You access individual components of a structure using the following syntax:

```
"DBname".name1.name2.name3
```

Example

The following example shows the addressing of structured data type tags:

Addressing	Explanation
"Global_DB".Motor.Valves[2]	Addressing of the 3rd valve of the "Valves" element in the "Motor" structure of the "Global_DB" global data block.

11.3.9.4 Transferring tags of the STRUCT data type

Description

You can also transfer individual components of a STRUCT tag as actual parameters if the components correspond to the data type of the formal parameter.

You can also transfer complete structures as parameters. If a block has an input parameter of the STRUCT data type, you must transfer a STRUCT with identical structure as the actual parameter. This means that the names and data types of all structure components have to be identical.

Structures and user-defined PLC data types can be assigned to one another if they have identical structures. This means that the data types and the order of all structure components must be identical. In nested structures, the data types and the order of the lower-level structures and their components must also match.

Two different PLC data types can also be assigned to one another if the data types and the order of all structure components including the lower-level structures are identical. The names of the PLC data types do not have to match.

User-defined PLC data types and structures cannot be assigned to system data types.

Note

We recommend that you program structures as PLC data types (UDT). PLC data types (UDT) make programming easier, since they can be used repeatedly and changed centrally.

11.3.10 ARRAY

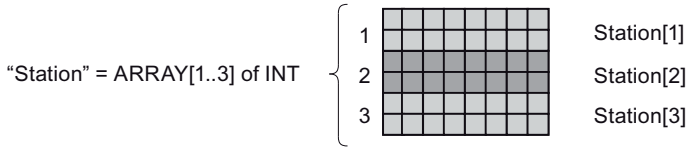
11.3.10.1 Basic information on ARRAY

Description

A tag of the ARRAY data type represents a data structure that consists of a fixed number of components of the same data type. All data types except ARRAY are permitted for the components.

When you create an ARRAY tag the index limits are defined in square brackets and the data type is defined after the keyword "of". The ARRAY limits can be either defined fixed using integers or global and local constants or as formal parameters of a block, or defined variably using ARRAY[*]. The low limit must be smaller than or equal to the high limit. An ARRAY may contain up to six dimensions, the limits of which can be specified separated by a comma.

The structure of a one-dimensional tag of the ARRAY data type has, for example, the following form:



The maximum ARRAY limits depend on the following factors:

- Data type of the ARRAY component
- Memory reserve (only in blocks with optimized access)
  You can find additional information under: AUTOHOTSPOT.
- Maximum size of the data block
- Maximum memory capacity of the CPU (you can find more information in the manual for the device)

The following table shows the properties of the ARRAY data type:

Block property	Format	ARRAY limits	Data type
Standard	ARRAY[low limit..high limit] of <DataType>	[-32 768..32 767] of <DataType>	All data types except ARRAY
Optimized		[-2 147 483 648..2 147 483 647] of <DataType>	Multi-instances

Various options are available for defining the ARRAY limits depending on the ARRAY that you declared.

ARRAY limits	Global data block / tag table	Block interface
Using integers as fixed ARRAY limits	x	x
Using global constants as fixed ARRAY limits	x	x

ARRAY limits	Global data block / tag table	Block interface
Using local constants as fixed ARRAY limits	-	x
Using variable ARRAY limits ARRAY[*]	-	x
ARRAY of multi-instances	-	x

For more information on the section of the block interface in which you can define an ARRAY or an ARRAY[*], refer to: Valid data types in the block interface (Page 4768)

Note

Applies to CPUs of the S7-1500 series

For a block with the "Optimized block access" block property, an element of the BOOL data type requires 1 byte of memory. This is also true when you use an ARRAY of <data type>. An ARRAY[0..1] of BOOL, for example, thus requires 2 bytes in an optimized program block.

Using integers as fixed ARRAY limits

You can use integers as fixed ARRAY limits:

- Example of a one-dimensional ARRAY

Static	
ValueArray	Array[0..2] of Int
ValueArray[0]	Int
ValueArray[1]	Int
ValueArray[2]	Int

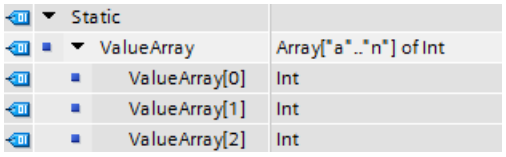
- Example of a three-dimensional ARRAY:

Static	
ValueArray	Array[0..2, 0..2] of Int
ValueArray[0,0]	Int
ValueArray[0,1]	Int
ValueArray[0,2]	Int
ValueArray[1,0]	Int
ValueArray[1,1]	Int
ValueArray[1,2]	Int
ValueArray[2,0]	Int
ValueArray[2,1]	Int
ValueArray[2,2]	Int

Using constants as fixed ARRAY limits

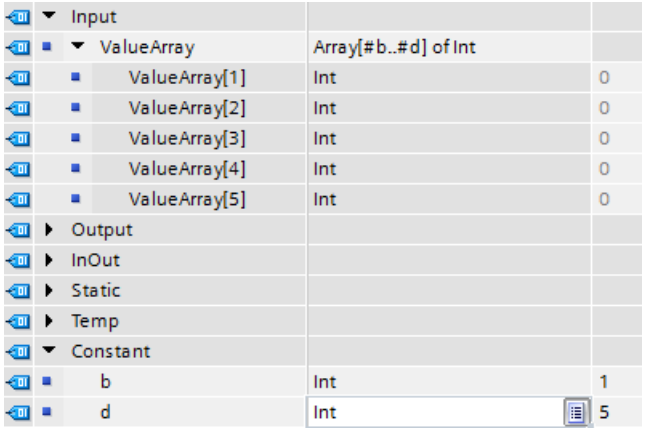
You can use local or global constants as fixed ARRAY limits:

- Fixed ARRAY limits for a one-dimensional ARRAY, consisting of two global user constants



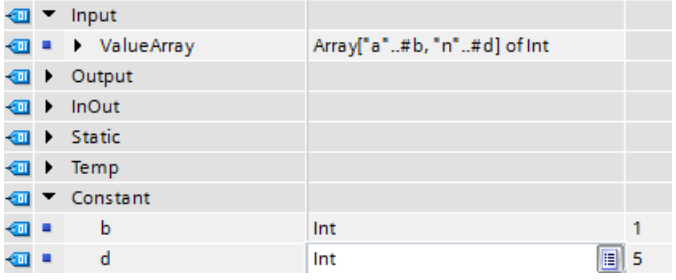
Static		
ValueArray	Array["a".."n"] of Int	
ValueArray[0]	Int	
ValueArray[1]	Int	
ValueArray[2]	Int	

- Fixed ARRAY limits for a one-dimensional ARRAY, consisting of two global user constants



Input		
ValueArray	Array[#b..#d] of Int	
ValueArray[1]	Int	0
ValueArray[2]	Int	0
ValueArray[3]	Int	0
ValueArray[4]	Int	0
ValueArray[5]	Int	0
Output		
InOut		
Static		
Temp		
Constant		
b	Int	1
d	Int	5

- Fixed ARRAY limits for a two-dimensional ARRAY, consisting of global and local user constants



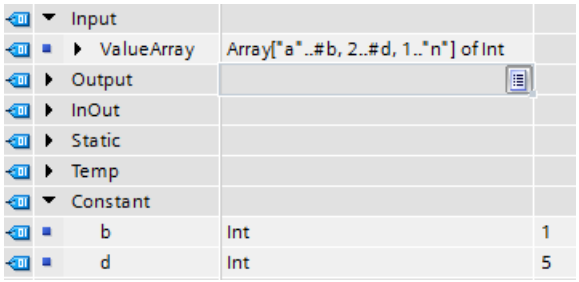
Input		
ValueArray	Array["a"..#b, "n"..#d] of Int	
Output		
InOut		
Static		
Temp		
Constant		
b	Int	1
d	Int	5

For more information on creating and using constants as ARRAY limits, refer to:

- Declaring local tags and constants in the block interface (Page 4772)
- AUTOHOTSPOT

Using integers and global and local constants as fixed ARRAY limits

You can also combine all three above-named options with each other and also used integers and global and local constants mixed within an ARRAY as ARRAY limits.



Using variable ARRAY limits ARRAY[*] (only in blocks with optimized access)

You can use an ARRAY[*] to program blocks that can process an ARRAY with flexible length. To do this, you declare variable ARRAY limits for the parameters of a function or function block. ARRAY[*] is available in all programming languages.

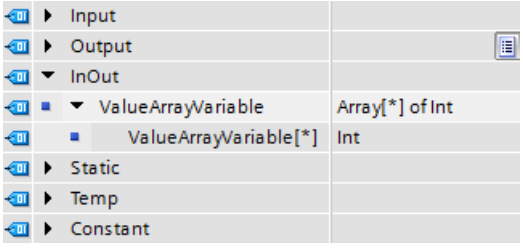
Using the "LOWER_BOUND" and "UPPER_BOUND" instructions, you can read the low and high limits of an ARRAY with variable limits during runtime.

Note

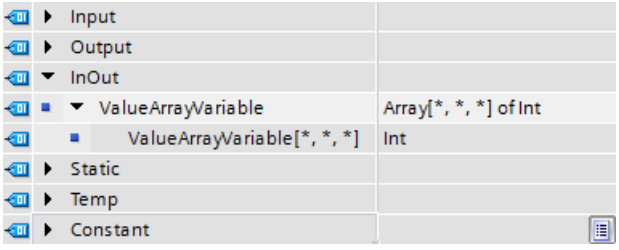
Availability of ARRAY[*]

ARRAY[*] is available in optimized blocks for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0. In functions (FC) you can use ARRAY[*] in all declaration subsections. In function blocks (FB) you can declare ARRAY[*] only as an in-out parameter in the "InOut" section.

- Variable ARRAY limits for a one-dimensional ARRAY:



- Variable ARRAY limits for a one-dimensional ARRAY:

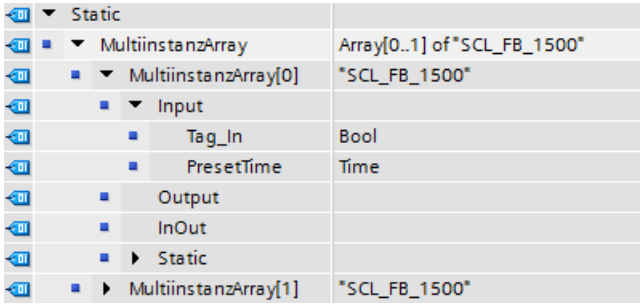


You can find additional information on the use of ARRAY[*] here:

- Transferring tags of the ARRAY / ARRAY[*] data type (Page 2294)
- Declaring tags of the ARRAY data type (Page 4775) in the block interface

Using multiple instances (only in blocks with optimized access)

You can also create multi-instances as ARRAY, in order to be able, for example, to use a variable index to address the individual multi-instances during the processing of program loops:



For more information on creating and using multi-instances, refer to:

- AUTOHOTSPOT
- Declaring multi-instances (Page 4780)

See also

Overview of the valid data types (Page 2239)

Indirect addressing of ARRAY components (Page 2215)

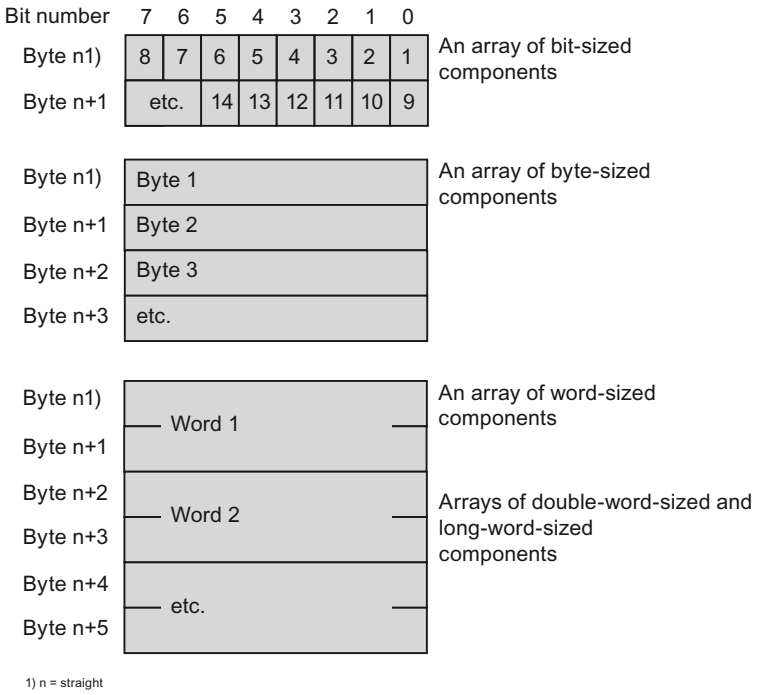
11.3.10.2 Structure of an ARRAY tag

Introduction

In non-optimized blocks an ARRAY tag never starts at a word limit, i.e. at a byte with even address. ARRAY tags occupy the memory up to the next word limit.

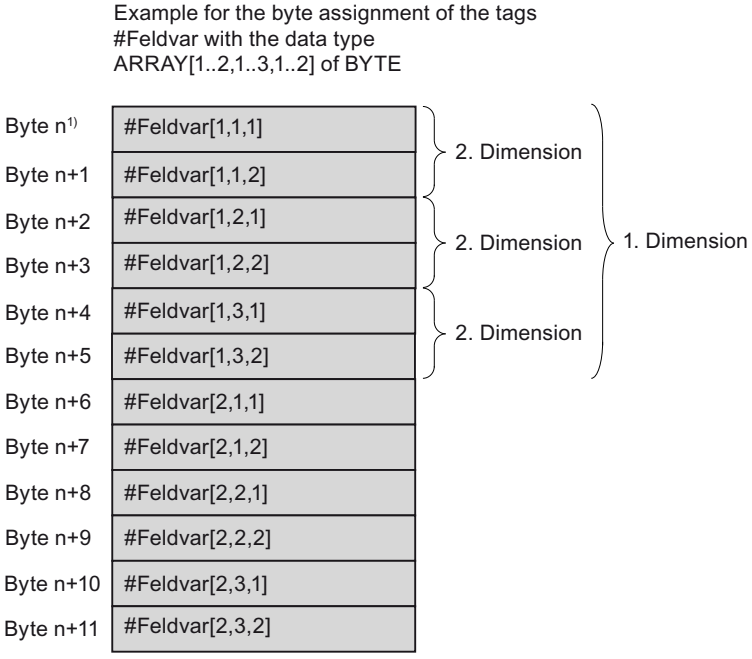
In optimized block the ARRAY requires as much space as the widest element of the structure. This means, for example, that an ARRAY of BYTE is located at a Byte limit and an ARRAY of LREAL is located at an 8-byte limit.

Structure of an ARRAY tag in the one-dimensional array:



Components with BOOL data type start in the least significant bit. Components with BYTE and CHAR data type start in the right byte. The individual components are listed in order.

Structure of an ARRAY tag in the multidimensional array:



In multidimensional arrays, the components are stored line-by-line (dimension-by-dimension), starting with the first dimension. In bit and byte components a new dimension always starts at

the next byte. In components of other data types a new dimension always starts in the next word (in the next even byte) in non-optimized blocks.

In optimized blocks each dimension requires as much space as the first dimension.

11.3.10.3 Addressing ARRAY components

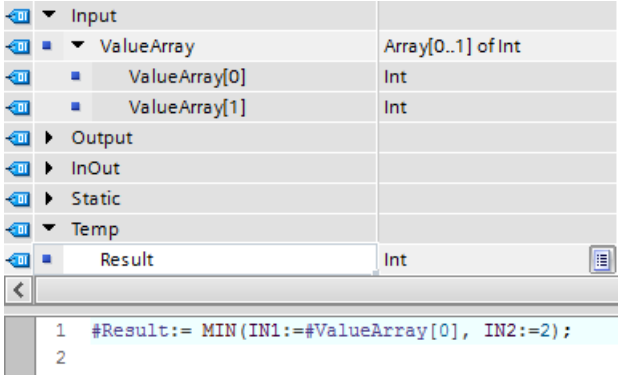
Introduction

ARRAY components are addressed using fixed of variable index. The components of an ARRAY tag can be treated like tags of the same data type.

The index of the component is specified in square brackets. The index includes an integer value (fixed index) or a tag (variable index) for each ARRAY dimension.

Addressing of ARRAY components using fixed index

The addressing of an ARRAY component using fixed index has the following form:

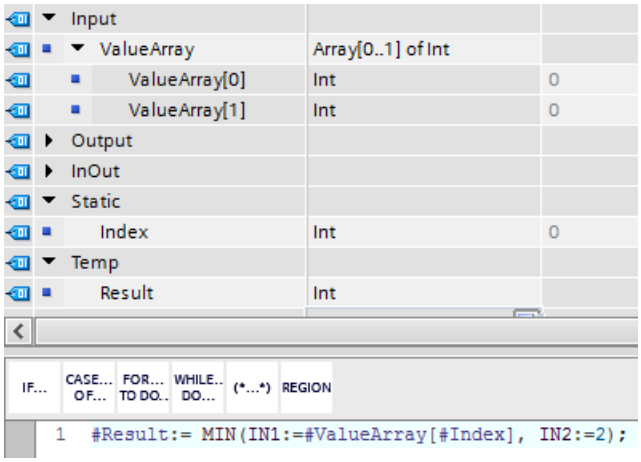


Addressing ARRAY components using variable index

An ARRAY component can also be addressed with a tag the value of which is only calculated during runtime. The tag can be an absolute or symbolic addressed global or local tag of integer data type. This addressing is also possible with multidimensional ARRAYs and with the addressing of sub-arrays. (<ArrayName>[i,j,k...])

A change of the tags in the called block has no effect on an ARRAY component that is created as actual parameter at an in/out parameter and addressed using a variable index. The value is written back to the same ARRAY component, transferred during the call, from which it was read.

The addressing of an ARRAY component using an index tag has the following form:



For more information on the indirect addressing of ARRAY components, refer to:

- AUTOHOTSPOT
- Programming recommendations:Indirect addressing of ARRAY components (Page 2215)

Error handling for ARRAY access

Access errors result when you access an element during runtime which is located outside the declared ARRAY limits. The various CPU families react differently to violations of the ARRAY limits:

- S7-300/400
  - The CPU changes to "STOP" mode.
  - You can program the program execution error OB (OB 85) to prevent this.
  - In SCL, you also have the option of enabling the attribute "Check ARRAY limits" in the block properties. This causes the enable output ENO to be set to FALSE in the case of ARRAY access errors.
- S7-1200
  - The CPU generates a diagnostic buffer entry and remains in "RUN" mode.
- S7-1500
  - The CPU changes to "STOP" mode.
  - You can program the programming error OB (OB 121) to prevent this.
  - You also have the option of programming the local error handling with the instructions "GET_ERROR: Get error locally" or "GET_ERROR_ID: Get error ID locally".

Note

Monitoring ARRAY access errors with ENO

The enable output ENO is not set to the signal state FALSE if the ARRAY limits are violated during execution of an instruction. The only exception is SCL blocks on CPUs of the S7-300/400 series for which the block property "Check ARRAY limits" is set.

11.3.10.4 Transferring tags of the ARRAY / ARRAY[*] data type

Transferring tags of the ARRAY data type

You can also transfer individual elements of an ARRAY as actual parameter if the element corresponds to the data type of the formal parameter.

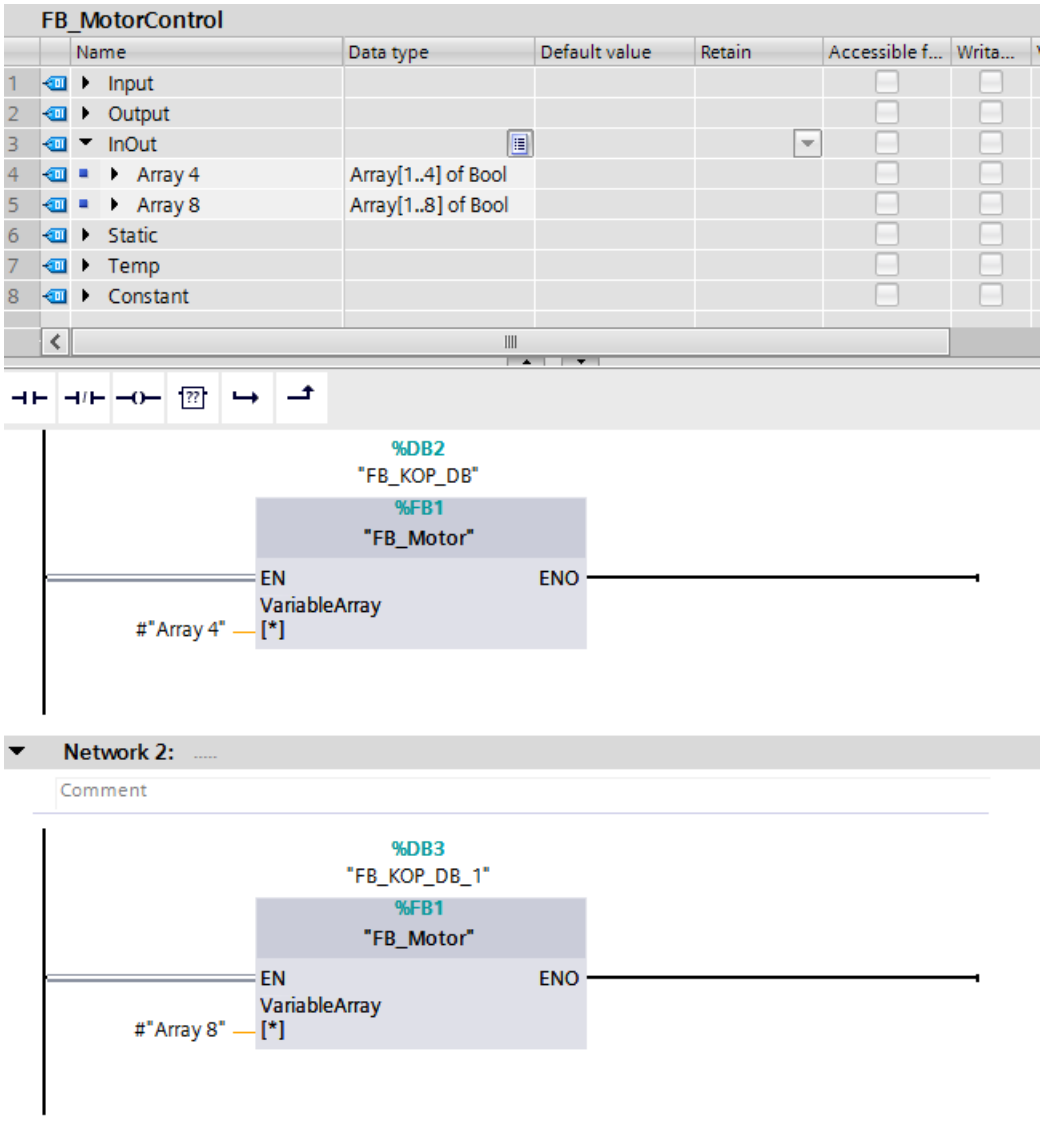
You can transfer tags of the ARRAY data type as parameters. If a block has an input parameter of ARRAY data type, you must pass an ARRAY with identical structure as the actual parameter. This means that the data type, the number of dimensions and the number of field components must be identical.

ARRAYs can be assigned to one another if they have an identical structure. This means that the data type, the number of dimensions and the number of array components must be identical. The names of the ARRAYs do not have to match.

Transferring tags of the ARRAY data type[*]

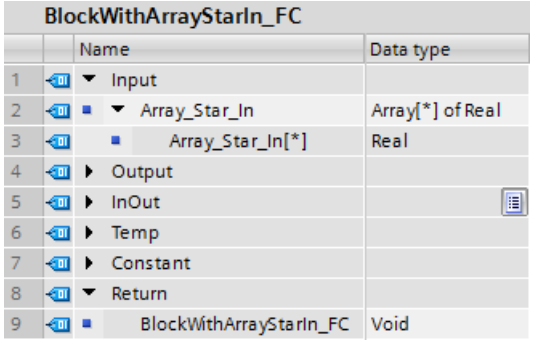
You can use ARRAY[*] to declare ARRAYs with variable limits at the parameters of a function or function block. When you create the block you can define the limits of the ARRAY that will be transferred later during runtime after the call during runtime.

The following figure shows two calls of a block with an input parameter of the ARRAY[*] data type. ARRAYs of different length are transferred with each call.



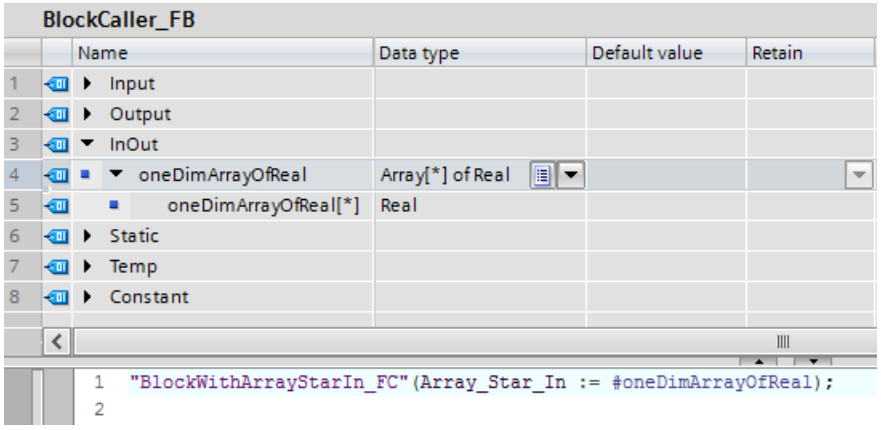
Applicable rules when transferring ARRAYs with flexible limits

Create the function "BlockWithArrayStarIn_FC" for the example so that it can be called later:



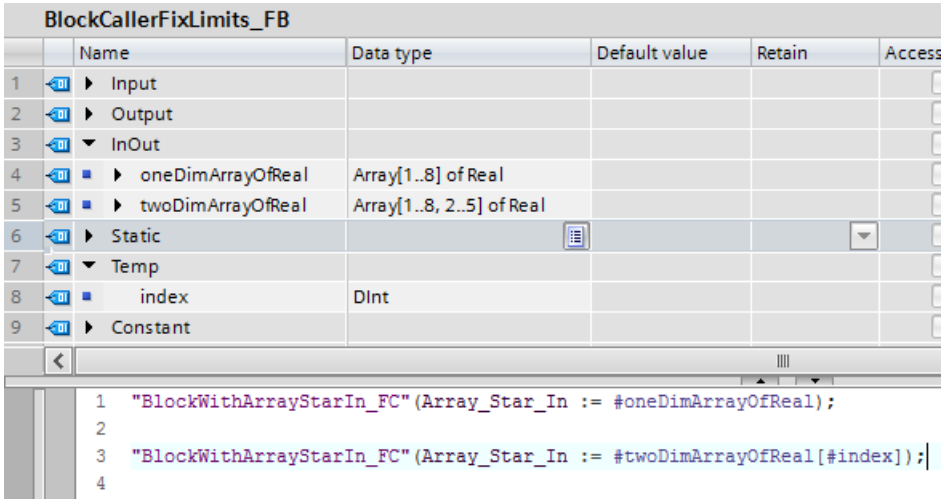
The assignment of ARRAY[*] to ARRAY[*] is permitted if the number of dimensions and the data type match. However, you cannot transfer individual ARRAY components in this process:

1. Declare the function block "BlockCaller_FB" and call the function "BlockWithArrayStarIn_FC":



The assignment of an ARRAY with known limits to an ARRAY[*] is permitted if the number of dimensions and the data type match. Here, you can also assign individual ARRAY components.

1. Declare the function block "BlockCallerFixLimits_FB" and call the function "BlockWithArrayStarIn_FC" twice:



The assignment of ARRAY[*] to VARIANT is permitted. Here, you can also assign individual ARRAY components.

1. Create the function "BlockWithVariantIn_FC" so that it can be called:

BlockWithVariantIn_FC

	Name	Data type
1	Input	
2	Variant_In	Variant
3	Output	
4	InOut	
5	Temp	
6	Constant	
7	Return	
8	BlockWithVariantIn_FC	Void

2. Declare the function "BlockWithArrayStarInVariant_FC" and call the function "BlockWithVariantIn_FC":

BlockWithArrayStarInVariant_FC

	Name	Data type	Default value	Supervision
1	Input			
2	array_star_in	Array[*] of Real		
3	Output			
4	InOut			
5	Temp			
6	index	DInt		
7	Constant			
8	Return			
9	BlockWithArrayStarInVariant_FC	Void		

IF... CASE... OF... FOR... TO DO.. WHILE... DO... (*...*) REGION R_TRIG

```
1  "BlockWithVariantIn_FC"(Variant_In := #array_star_in);
2
3  "BlockWithVariantIn_FC"(Variant_In := #array_star_in[#index]);
4
```

The assignment of ARRAY[*] to ARRAYs with fixed limits is not permitted.

Note

Block parameters of the ARRAY[*] type must be supplied with an actual parameter.

11.3.10.5 Examples of how to use ARRAYs

Example of a multi-dimensional ARRAY

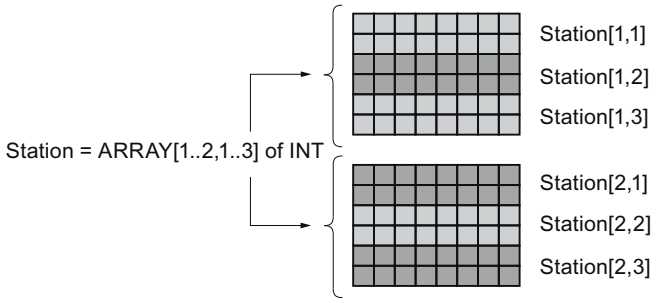
Description

Tags of the ARRAY data type can contain up to 6 dimensions. The same rules apply as for one-dimensional ARRAYs. The dimension areas are written in the declaration in square brackets, each separated by a comma. In multidimensional arrays, the components are stored starting with the first dimension.

The following table shows the declaration of a two-dimensional tag of the ARRAY data type:

Name	Data type	Value	Comment
Betr_Temp	ARRAY[1..2, 1..3] of INT	1,1,4(0)	Two-dimensional tag of the ARRAY data type with 6 components The first two components are assigned the value "1". The remaining four components are assigned the value "0".

The following figure shows the structure of the declared tag of the ARRAY data type:



Access to the components

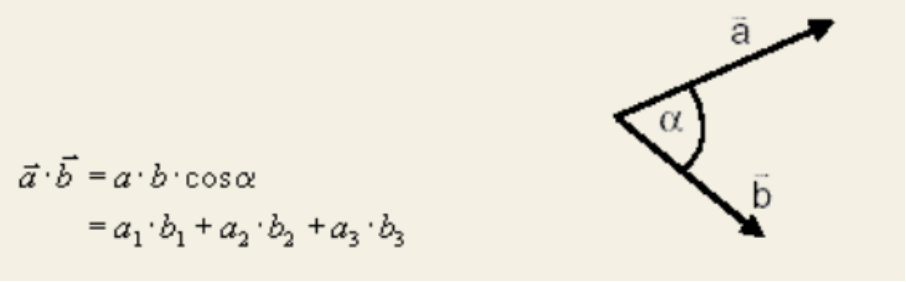
The values of the individual components are accessed via an index. You can use a constant or a tag for the index. The index of the first component is, for example, [1,1] and the index of the fourth array component is [2,1]. For example, you need to declare "Station[2,1]" in the program to enable access to the value of the fourth component.

Example for calculating the scalar product of two vectors with ARRAY[*]

Description

The scalar product is a mathematical link that assigns a number to two vectors (scalar).

The scalar product of two vectors results in a scalar variable and is defined by:



$$\vec{a} \cdot \vec{b} = a \cdot b \cdot \cos\alpha$$
$$= a_1 \cdot b_1 + a_2 \cdot b_2 + a_3 \cdot b_3$$

Here ∝ is the angle between the two vectors -$\vec{a}$ and $\vec{b}$.

An example for calculating the scalar product is:

$$\vec{a} = \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}, \vec{b} = \begin{pmatrix} 2 \\ 0 \\ 4 \end{pmatrix}, \vec{a} * \vec{b} = (1*2+3*0+5*4) = 22$$

In this case, the result is the number 22.

In the following programming example, you see how you can calculate the scalar product of two vectors with flexible ARRAY bounds using ARRAY[*]. Here, the function "ScalarProduct_FC" serves as a template for calculating the individual scalar products.

You need the following objects for the programming example:

- Two data blocks and a PLC data type (UDT) for managing the vector data.
- Function that contains the program code for calculating the scalar product.
- The two instructions "LOWER_BOUND" and "UPPER_BOUND" to read out the ARRAY bounds.
- Organization block for calculating the scalar products

Procedure

In this example vectors, that is to say ARRAYs with 1 dimension, are used. Other calculations, such as matrix multiplications, can of course work with multi-dimensional ARRAYs.

To calculate the scalar products, follow these steps:

1. Create the PLC data type (UDT) "VectorArrays_UDT":

VectorArrays_UDT

	Name	Data type
1	VectorD5Coordinates	Array[1..5] of Real
2	VectorD5Coordinates[1]	Real
3	VectorD5Coordinates[2]	Real
4	VectorD5Coordinates[3]	Real
5	VectorD5Coordinates[4]	Real
6	VectorD5Coordinates[5]	Real
7	VectorE13Coordinates	Array[1..13] of Real
8	VectorE13Coordinates[1]	Real
9	VectorE13Coordinates[2]	Real
10	VectorE13Coordinates[3]	Real
11	VectorE13Coordinates[4]	Real
12	VectorE13Coordinates[5]	Real
13	VectorE13Coordinates[6]	Real
14	VectorE13Coordinates[7]	Real
15	VectorE13Coordinates[8]	Real
16	VectorE13Coordinates[9]	Real
17	VectorE13Coordinates[10]	Real
18	VectorE13Coordinates[11]	Real
19	VectorE13Coordinates[12]	Real
20	VectorE13Coordinates[13]	Real

   The two ARRAYs "VectorD5Coordinates" and "VectorE13Coordinates" provide a data basis with which you can then calculate scalar products.

2. Create the "VectorArrays1_DB" data block on the basis of the "VectorArrays_UDT" PLC data type:

VectorArrays1_DB

	Name	Data type
1	Static	
2	VectorD5Coordinates	Array[1..5] of Real
3	VectorE13Coordinates	Array[1..13] of Real

3. Create the second "VectorArrays2_DB" data block. Besides the vectors from the "VectorArrays_UDT" PLC data type, it also contains two other vectors:

VectorArrays2_DB

	Name	Data type
1	Static	
2	VectorA10Coordinates	Array[1..10] of Real
3	VectorB13Coordinates	Array[1..13] of Real
4	VectorsC	"VectorArrays_UDT"
5	VectorD5Coordinates	Array[1..5] of Real
6	VectorE13Coordinates	Array[1..13] of Real

   You have now created the data basis for calculating the scalar products.

4. Create the "ScalarProduct_FC" function as a template for creating the calculation procedure for calculating the scalar product:
   The block interface:

ScalarProduct_FC

	Name	Data type	Comment
1	Input		
2	Vector1	Array[*] of Real	Interface for data input for Vector 1
3	Vector1[*]	Real	Interface for data input for Vector 1
4	Vector2	Array[*] of Real	Interface for data input for Vector 1
5	Vector2[*]	Real	Interface for data input for Vector 1
6	Output		
7	InOut		
8	Temp		
9	Sum	Real	
10	i	Int	
11	LowerBound1	DInt	
12	LowerBound2	DInt	
13	Upperbound1	DInt	
14	UpperBound2	DInt	
15	Constant		
16	Return		
17	ScalarProduct_FC	Real	

   Program code:

```
1  #LowerBound1 := LOWER_BOUND(ARR := #Vector1, DIM := 1);
2  #LowerBound2 := LOWER_BOUND(ARR := #Vector2, DIM := 1);
3  #Upperbound1 := UPPER_BOUND(ARR := #Vector1, DIM := 1);
4  #Upperbound1 := UPPER_BOUND(ARR := #Vector1, DIM := 1);
5
6  IF #LowerBound1 <> #LowerBound2 OR #Upperbound1 <> #UpperBound2 THEN
7      #ScalarProduct_FC := -1;
8      RETURN;
9  END_IF;
10 #Sum := 0;
11 FOR #i := #LowerBound1 TO #Upperbound1 DO
12     #Sum += #Vector1[#i] * #Vector2[#i];
13 END_FOR;
14
15 #ScalarProduct_FC := #Sum;
16
```

   The upper and lower ARRAY bounds of vector 1 and vector 2 are queried in rows 1–4. It is then known how many coordinates the two ARRAYs of the vectors have. As the scalar product cannot be established until the number of coordinates of the two vectors that you wish to multiply is equal, rows 6 to 9 are required
   If the two lower or upper bounds of the ARRAYs are not the same, the "ScalarProduct_FC" function produces the function value "-1" and the program block is exited (RETURN).
   If the two lower or upper bounds of the ARRAY are the same, the #Sum tag in initialized (row 10) with the value "0" and the calculation of the scalar product is executed (rows 11 - 13).

5. Create the "Main_OB" organization block with the event class "Program cycle". The "ScalarProduct_FC" function created in step 4 now serves as a template for calculating scalar products in "Main_OB":
   The block interface:

Main_OB

		Name	Data type
1		Input	
2		Initial_Call	Bool
3		Remanence	Bool
4		Temp	
5		Result1	Real
6		Result2	Real
7		Result3	Real
8		Result4	Real
9		Result5	Real
10		Result6	Real
11		Result7	Real
12		Constant	

Program code:

```
#Result1 := "ScalarProduct_FC"(Vector1 := "VectorArrays2_DB".VectorA10Coordinates,
                               Vector2 := "VectorArrays1_DB".VectorE13Coordinates);
#Result2 := "ScalarProduct_FC"(Vector1 := "VectorArrays2_DB".VectorsC.VectorD5Coordinates,
                               Vector2 := "VectorArrays1_DB".VectorD5Coordinates);
#Result3 := "ScalarProduct_FC"(Vector1 := "VectorArrays2_DB".VectorB13Coordinates,
                               Vector2 := "VectorArrays1_DB".VectorE13Coordinates);
#Result4 := "ScalarProduct_FC"(Vector1 := "VectorArrays2_DB".VectorA10Coordinates,
                               Vector2 := "VectorArrays2_DB".VectorA10Coordinates);
#Result5 := "ScalarProduct_FC"(Vector1 := "VectorArrays2_DB".VectorA10Coordinates,
                               Vector2 := "VectorArrays1_DB".VectorE13Coordinates);

```

Result

The results of the individual scalar product calculations are written in the tags #Result1 -5.

11.3.11 Pointer

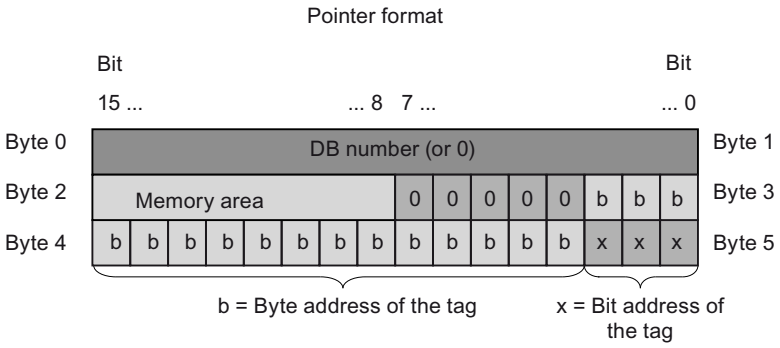
11.3.11.1 POINTER (S7-300, S7-400, S7-1500)

Description

A parameter of the type POINTER is a pointer that can point to a specific tag. It occupies 6 bytes (48 bits) in memory and may contain the following tag information:

- DB number, or 0 if the data is not stored in a DB
- Memory area in the CPU
- Tag address

The following figure shows the structure of parameter type POINTER:



Types of pointer

Depending on the information, you can use parameter type POINTER to declare the following four types of pointer:

- Area-internal pointer:
  An area-internal pointer contains information on the address of a tag.
- Cross-area pointer:
  A cross-area pointer contains information on the memory area and the address of a tag.
- DB pointer:
  You can use a DB pointer to point to a data block tag. A DB pointer also contains a data block number in addition to the memory area and the address of a tag.
- Zero pointer:
  Use the zero pointer to indicate a missing value. A missing value may indicate that no value exists, or that the value is not yet known. A zero value represents the absence of a value, but is also a value.

The following table shows the formats for the declaration of various pointer types:

P#ByteRepresentation	Format	Example of value input	Description
Symbolic	P#Byte.Bit	"MyTag"	Area-internal pointer
	P#OperandAreaByte.Bit	"MyTag"	Cross-area pointer
	P#Data_block.Data_operand	"MyDB"."MyTag"	DB pointer
	P#Zero value	-	Zero pointer
Absolute	P#Byte.Bit	P#20.0	Area-internal pointer
	P#OperandAreaByte.Bit	P#M20.0	Cross-area pointer
	P#Data_block.Data_operand	P#DB10.DBX20.0	DB pointer
	P#Zero value	P#0.0, ZERO	Zero pointer

The prefix P#

Enter the actual value without prefix P# in the block call to supply a formal parameter of the POINTER data type. The entry is then automatically converted to the POINTER format.

If you use the prefix P#, you can only point to memory areas with "Standard" access mode. You can find additional information and a direct comparison of the two access options here:

- AUTOHOTSPOT
- The new S7-1200/1500 CPU functions and programming recommendations at a glance (Page 2209)

The following should be noted when using the prefix P# in the STL programming language:

Block type	"Optimized" access mode	"Standard" access mode
Function (FC)	A tag cannot be used in the program code with the prefix P#.	The following tags can be used in the program code with the prefix P#. • Structured tags that are declared in the InOut section of the block interface. Tags with elementary data types cannot be used with the prefix P#.
Function block (FB)	A tag cannot be used in the program code with the prefix P#.	The following tags can be used in the program code with the prefix P#. • Tags that have been declared in the sections Static, Input and Output of the block interface. • Tags with the PLC data type (UDT) that are declared in the InOut section of the block interface.

You can find additional information about structured and elementary data types here: AUTOHOTSPOT

Memory areas

The following table shows the hexadecimal codes of the memory areas for parameter type POINTER:

Hexadecimal code	Memory area	Description
B#16#80[1)]	P	Peripherals on a CPU S7-300/400
16#1	P	Peripheral inputs on a CPU S7-1500
16#2	P	Peripheral outputs on a CPU S7-1500
B#16#81	I	Memory area of inputs
B#16#82	Q	Memory area of outputs
B#16#83	M	Memory area of bit memory
B#16#84	DBX	Data block
B#16#85	DIX	Instance data block
B#16#86	L	Local data
B#16#87	V	Previous local data
[1)] These data types can only be used for the POINTER pointer on a CPU S7-300/400.		

See also

Overview of the valid data types (Page 2239)

11.3.11.2 ANY (S7-300, S7-400, S7-1500)

Description

An ANY type parameter points to the start of a data area and specifies its length. An ANY pointer occupies 10 bytes of memory and may contain the following information:

- Data type:
  Data type of the elements of the data area
- Repetition factor:
  Number of elements of the data area
- DB number:
  Data block that contains the declaration of data area elements.
- Memory area:
  Memory area of the CPU that stores the data area elements.
- Start address of the data in the format "byte.bit":
  Data area start identified by the ANY pointer.
- Zero pointer:
  Use the zero pointer to indicate a missing value. A missing value may indicate that no value exists, or that the value is not yet known. A zero value represents the absence of a value, but is also a value.

Note

Access options

For a CPU of the S7-1500 series, the ANY pointer can also only point to memory areas with "Standard" access mode. You can find additional information and a direct comparison of the two access options here:

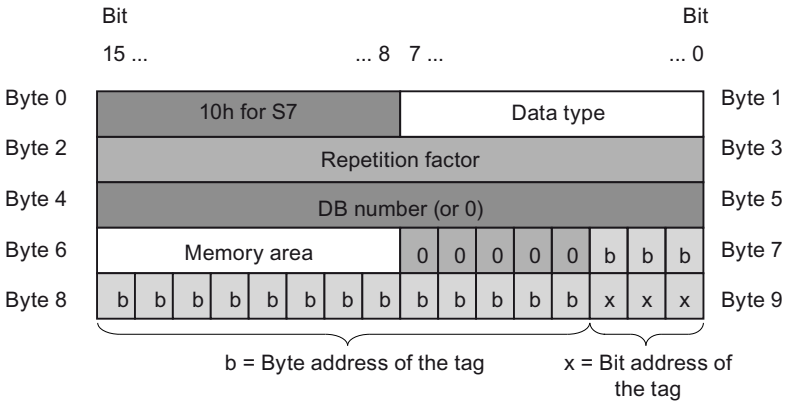
- AUTOHOTSPOT
- The new S7-1200/1500 CPU functions and programming recommendations at a glance (Page 2209)

In the programming languages SCL and STL, memory of any kind can be transferred upon a block call if an ANY pointer has been programmed at a block parameter.

The ANY pointer cannot, however, store any information on the structure of the memory. For example, the ANY pointer does not save the information that it points to a tag of the PLC data type. The ANY pointer sees this as an ARRAY of BYTE.

Parameters of the ANY data type can be passed to system function blocks (SFBs) or system functions (SFCs).

The following figure shows the structure of the ANY pointer:



An ANY pointer cannot identify structures. It can only be assigned to local tags.

The following table shows the formats for the declaration of an ANY pointer:

Representation	Format	Example of value input	Description
Symbolic	P#DataBlock.MemoryArea DataAddress Type Number	"MyDB".StructTag.InitialComponents	Area with 10 words in global DB11 starting with DBB20.0
	P#MemoryArea DataAddress Type Number	"MyMarkerTag"	Area with 4 bytes starting with MB 20.0
		"MyTag"	Input I1.0
	P#Zero value	-	Zero value
Absolute	P#DataBlock.MemoryArea DataAddress Type Number	P#DB11.DBX20.0 INT 10	Area with 10 words in global DB11 starting with DBB20.0
	P#MemoryArea DataAddress Type Number	P#M20.0 BYTE 10	Area with 10 bytes starting with MB 20.0
		P#I1.0 BOOL 8	Range with 8 bits from input I1.0 (the range length specified must be divisible by 8.)
	P#Zero value	P#P0.0 VOID 0, NULL [1]	Zero value
[1] In the programming languages LAD and FBD, only NULL is a value entry for the zero value.			

Coding of data types

The following table lists the coding of data types for the ANY pointer:

Hexadecimal code	Data type	Description
B#16#00	NIL	Null pointer
B#16#01[1]	BOOL	Bits
B#16#02	BYTE	bytes, 8 bits
B#16#03	CHAR	8-bit characters
B#16#04	WORD	16-bit words
B#16#05	INT	16-bit integers
B#16#06	DWORD	32-bit words

Hexadecimal code	Data type	Description
B#16#07	DINT	32-bit integers
B#16#08	REAL	32-bit floating-point numbers
B#16#0B	TIME	Time duration
B#16#0C	S5TIME	Time duration
B#16#09	DATE	Date
B#16#0A	TOD	Date and time
B#16#0E	DT	Date and time
B#16#13	STRING	Character string
B#16#17[1)]	BLOCK_FB	Function block
B#16#18[1)]	BLOCK_FC	Function
B#16#19[1)]	BLOCK_DB	Data block
B#16#1A[1)]	BLOCK_SDB	System data block
B#16#1C[1)]	COUNTER	Counter
B#16#1D[1)]	TIMER	Timer
[1)] These data types can only be used for the ANY pointer on a CPU S7-300/400.		

Coding of the memory area

The following table lists the coding of the memory areas for the ANY pointer:

Hexadecimal code	Area	Description
B#16#80[1)]	P	I/O
B#16#81	I	Memory area of inputs
B#16#82	Q	Memory area of outputs
B#16#83	M	Memory area of bit memory
B#16#84	DBX	Data block
B#16#85[1)]	DIX	Instance data block
B#16#86	L	Local data
B#16#87	V	Previous local data
[1)] These memory areas can only be used for the ANY pointer on an S7-300/400 CPU.		

See also

Overview of the valid data types (Page 2239)

11.3.11.3 VARIANT (S7-1200, S7-1500)

Basic information on VARIANT (S7-1200, S7-1500)

Description

A parameter of VARIANT data type is a pointer or a reference It can point to tags of various data types. The VARIANT pointer cannot point to an instance and therefore cannot point to a multiple instance or ARRAY of multiple instances. The VARIANT pointer can be an object of an elementary data type, such as INT or REAL. It can also be a STRING, DTL, ARRAY of STRUCT, UDT, or ARRAY of UDT. The VARIANT pointer can recognize structures and point to individual structure components. An operand of data type VARIANT occupies no space in the instance data block or work memory. However, it will occupy memory space on the CPU.

A tag of the VARIANT type is not an object but rather a reference to another object. Individual elements of the VARIANT type can only be declared on formal parameters within the block interface of a function in the VAR_IN, VAR_IN_OUT and VAR_TEMP sections. For this reason, it cannot be declared in a data block or in the static section of the block interface of a function block, for example, because its size is unknown. The size of the referenced objects can change.

Using the VARIANT data type, you can in particular create generic, standardized function blocks (FB) or functions (FC) for various data types. Various instructions in all programming languages are available to you for this purpose. During the creation of the program, you can specify which data types the block should be able to process. The VARIANT data type supports you here by allowing the interconnection of any tags. You can then react accordingly to their data types in the block. When a block is called, you can connect the parameters of the block to tags of any data type. When a block is called, the type information of the tag is transferred in addition to a pointer to the tag. The code of the block can then be executed according to its type in line with the tag transferred during runtime.

If, for example, a block parameter of a function has the VARIANT data type, then a tag of the integer data type can be transferred at one point in the program, and a tag of the PLC data type can be transferred at another point in the program. With the help of the VARIANT instructions, the function is then in a position to react to the situation without errors.

Note

You can only point to a complete data block if it was originally derived from a user-defined data type (UDT).

Note

Address I/O

Direct reading or writing of a signal from an I/O input or output is only possible on a CPU of an S7-1500 module. (<Operand>: P)

The following table shows the properties of the VARIANT pointer:

Length (bytes)	Repre-sentation	Format	Example of value input
0	Symbolic	Operand	"TagResult"
		NameDataBlock.NameOper-and.Component	"Data_TIA_Portal".StructVariable.First-Component
	Absolute	Operand	%MW10
		DataBlockNumber.Operand Type Length (valid only for blocks with standard access)	P#DB10.DBX10.0 INT 12 [1]
		P#Zero value [1]	P#0.0 VOID 0, ZERO [1]
[1] If you use the prefix P#, you can only point to memory areas with "standard" access mode.			

Coding of data types

If you use absolute addressing with P#, the following data types are permitted:

Hexadecimal code	Data type	Description
B#16#00	NIL	Null pointer
B#16#01	BOOL	Bits
B#16#02	BYTE	bytes, 8 bits
B#16#03	CHAR	8-bit characters
B#16#04	WORD	16-bit words
B#16#05	INT	16-bit integers
B#16#06	DWORD	32-bit words
B#16#07	DINT	32-bit integers
B#16#08	REAL	32-bit floating-point numbers
B#16#0B	TIME	Time duration
B#16#0C	S5TIME	Time duration
B#16#09	DATE	Date
B#16#0A	TOD	Date and time
B#16#0E	DT	Date and time
B#16#13	STRING	Character string
B#16#17	BLOCK_FB	Function block
B#16#18	BLOCK_FC	Function
B#16#19	BLOCK_DB	Data block
B#16#1A	BLOCK_SDB	System data block
B#16#1C	COUNTER	Counter
B#16#1D	TIMER	Timer

Example

The following example shows how VARIANT works using the "MOVE: Move value" instruction of STL:

STL	Explanation
`CALL MOVE`	`// The instruction is called.`
`value_type := VARIANT`	`// Data type of the parameters IN and OUT`
`IN := "Data_TIA_Portal".StructVaria-ble.FirstComponent`	`// The contents of the "FirstComponent" operand are moved from the "Data_TIA_Portal" DB.`
`OUT := "MotorDB".StructResult.TagResult`	`// And transferred to the "TagResult" operand from the "MotorDB" DB.`

See also

Overview of the valid data types (Page 2239)

VARIANT instructions (Page 2311)

Data type conversion for S7-1200 (Page 2465)

Applications for pointers in the comparison (S7-1200, S7-1500)

Application cases for pointers with an S7-1200/1500 CPU as compared to S7-300/400

The following table provides you with an overview of the various applications for pointers on a CPU of the S7-300/400 series (ANY pointer) and their solution with a CPU of the S7-1200/1500 series.

In most applications, the use of a pointer is no longer required on a CPU of the S7-1200/1500 series. Instead, much simpler language resources are available.

It is only advisable to use the VARIANT data type for indirect addressing, when the data types are only determined during the program runtime.

What is the purpose of the ANY pointer?	Recommendations in the TIA Portal (S7-1200/ S7-1500)
Moving data of any source and destination data type in the program using the instruction "BLKMOV: Move block".	Definition of tags within a PLC data type. You can use the instructions "Serialize" and "Deserialize" to move the tags.
Initializing an ARRAY structure	Using the instruction "FILL_BLK: Fill block", you initialize or fill an ARRAY structure.
Moving ARRAY elements	Using the instruction "MOVE_BLK: Move block". you move the content of multiple elements of an ARRAY structure to another ARRAY structure.

What is the purpose of the ANY pointer?	Recommendations in the TIA Portal (S7-1200/ S7-1500)
Memory and performance optimization using structured data	Use the InOut section in the block interface to optimize memory and performance. You can find additional information in the "Programming Guideline for S7-1200/1500" under the following link Programming Guideline for S7-1200/1500 (http://support.automation.siemens.com/WW/llisapi.dll?aktprim=4&lang=en&referer=%2fWW%2f&func=cslib.csinfo&siteid=csius&groupid=4000002&extranet=standard&viewreg=WW&nodeid4=20229695&objaction=csopen)
Access to individual bits/bytes of a WORD	Use the "slice access" You can find additional information here: Example of a slice access (http://support.automation.siemens.com/WW/llisapi.dll?aktprim=0&lang=en&referer=%2fWW%2f&func=cslib.csinfo&siteid=csius&groupid=4000002&extranet=standard&viewreg=WW&nodeid0=29156492&objaction=csopen)
Determination of the length of structures or data blocks	Use an ARRAY and read its length using the instruction "CountofElements: Get number of ARRAY elements". The instruction only works in conjunction with the data type VARIANT.
Indirect addressing	You can use the VARIANT pointer for the indirect addressing of data types that will only be known during runtime. You can use the data type DB_ANY for indirect access to a data block.

VARIANT instructions (S7-1200, S7-1500)

VARIANT instructions

The following instructions for working with VARIANT are available to you in the TIA Portal:

Basic instructions		
Category	**Instruction**	**Description**
Comparator operations	EQ_Type	Compare data type for EQUAL with the data type of a tag
	NE_Type	Compare data type for UNEQUAL with the data type of a tag
	EQ_ElemType	Compare data type of an ARRAY element for EQUAL with the data type of a tag
	NE_ElemType	Compare data type of an ARRAY element for UNEQUAL with the data type of a tag
	IS_NULL	Query for EQUALS ZERO pointer
	NOT_NULL	Query for UNEQUALS ZERO pointer
	IS_ARRAY	Check for ARRAY
	TypeOf	Check data type of a VARIANT tag
	TypeOfElements	Check element data type of a VARIANT tag

Basic instructions		
Category	**Instruction**	**Description**
Move operations	MOVE_BLK_VARIANT	Move block
	VariantGet	Read out VARIANT tag value
	VariantPut	Write VARIANT tag value
	CountOfElements	Get number of ARRAY elements
Conversion operations	VAR-IANT_TO_DB_ANY	Convert VARIANT to DB_ANY
	DB_ANY_TO_VARIANT	Convert DB_ANY to VARIANT

Note

Differences between MOVE, MOVE_BLK and MOVE_BLK_VARIANT

- You can use the "MOVE" instruction to copy complete structures.
- You can use the "MOVE_BLK" instruction to move parts of ARRAYs with known data type.
- The MOVE_BLK_VARIANT instruction is only required if you want to move parts of ARRAYs with a data type that is only known during program runtime.

You can find additional information on the individual instructions in the information system under "Basic instructions > Respective programming language".

You can also find additional instructions which also work with the VARIANT data type under the "Extended instructions".

See also

Basic information on VARIANT (Page 2308)

Indirect addressing with the VARIANT data type (http://support.automation.siemens.com/WW/llisapi.dll?aktprim=0&lang=en&referer=%2fWW%2f&func=cslib.csinfo&siteid=csius&groupid=4000002&extranet=standard&viewreg=WW&nodeid0=29156492&objaction=csopen)

Application options for VARIANT instructions (S7-1200, S7-1500)

Introduction

In the following chapter, you will learn which application options you have with VARIANT instructions.

Evaluating data types of tags to which a VARIANT points

In the table below, you see which instructions are available to you to evaluate data types of tags to which a VARIANT points:

Function	Instruction	Description
To determine the data type	TypeOf(): Check data type of a VARIANT tag (This instruction is available only in SCL and only in conjunction with an IF instruction.)	You use this instruction to compare the data type to which a VARIANT tag points with the data type of any other tag. You can also make the comparison with a PLC data type.
	TypeOfElements(): Check element data type of a VARIANT tag (This instruction is available only in SCL and only in conjunction with an IF instruction.)	You use this instruction to compare the data type to which a VARIANT tag points with the data type of any other tag. You can also make the comparison with a PLC data type. If the data type of the VARIANT tag is an ARRAY, the data type of the ARRAY elements is compared.
	EQ_Type: Compare data type for EQUAL with the data type of a tag NE_Type: Compare data type for UNEQUAL with the data type of a tag	You use this instruction to compare the data type to which a VARIANT tag points with the data type of any other tag. You can also make the comparison with a PLC data type.
	EQ_ElemType: Compare element data type for EQUAL with the data type of a tag NE_ElemType: Compare element data type for UNEQUAL with the data type of a tag	You use this instruction to compare the data type to which a VARIANT tag points with the data type of any other tag. You can also make the comparison with a PLC data type. If the data type of the VARIANT tag is an ARRAY, the data type of the ARRAY elements is compared.
To evaluate ARRAY elements	IS_ARRAY: Check for ARRAY	You use this instruction to check whether the data type to which a VARIANT tag points is an ARRAY.
	CountOfElements: Get number of ARRAY elements	You use this instruction to read out how many ARRAY elements the tag has to which the VARIANT tag points.

You can find additional information on the individual instructions in the information system under "Basic instructions > Respective programming language".

Reading data to which a VARIANT points

To be able to use the data you must move this data to a tag as an intermediate step, as it cannot be processed directly.

Instruction	Description	Example		Result
		VARIANT points to	Destination data type	
VariantGet: Read out VARIANT tag value	You use this instruction to move the value of a single tag to another tag. The data types of both tags must match.	UDT_1	UDT_1	The instruction is executed
		REAL	REAL	
		DINT	DWORD	The instruction is not executed.

Assigning data to a VARIANT tag

You cannot use the instruction to initialize VARIANT tags. The VARIANT tags must therefore already be initialized when the data is returned to the tag. Do not use uninitialized temporary VARIANT tags.

Instruction	Description	Example		Result
		Source data type	VARIANT points to:	
VariantPut: Write VARIANT tag value	You use this instruction to move the value of a single tag to another tag. The data types of both tags must match.	UDT_1	UDT_1	The instruction is executed
		REAL	REAL	
		DINT	DWORD	The instruction will not be executed, as the data types are different.

Processing dynamic ARRAY structures

To evaluate ARRAY elements	TypeOfElements(): Check data type of an ARRAY element of a VARIANT tag (This instruction is available only in SCL and only in conjunction with an IF instruction.)	You use this instruction to compare the data type to which a VARIANT tag points with the data type of any other tag. You can also make the comparison with a PLC data type. If the data type of the VARIANT tag is an ARRAY, the data type of the ARRAY elements is compared.
	IS_ARRAY: Check for ARRAY	You use this instruction to check whether the data type to which a VARIANT tag points is an ARRAY.
	CountOfElements: Get number of ARRAY elements	You use this instruction to read out how many ARRAY elements the tag has to which the VARIANT tag points.
	MOVE_BLK_VARIANT: Move block	This instruction is used to move dynamic and type-safe (integrated type test) ARRAYs. You can freely select the limit values for the source and destination ARRAYs. The data types of ARRAY elements must match.

Note

Differences between MOVE, MOVE_BLK and MOVE_BLK_VARIANT

- You can use the "MOVE" instruction to copy complete structures.
- You can use the "MOVE_BLK" instruction to move parts of ARRAYs with known data type.
- The MOVE_BLK_VARIANT instruction is only required if you want to move parts of ARRAYs with a data type that is only known during program runtime.

Additional information on the use of the MOVE_BLK_VARIANT instruction can be found in the "Moving data" programming example.

See also

Basic information on VARIANT (Page 2308)

Example of moving data (Page 2316)

Initializing VARIANT (S7-1200, S7-1500)

Description

Initialize the VARIANT data type by assigning a specific tag to the VARIANT block parameter at the block call. This forms a reference to the address of the passed tag. To do this, create a block parameter of the VARIANT data type in the block interface. In the following example, these are the two block parameters SourceArray and DestinationArray in the section InOut.

FC_PartialArrayCopy

		Name	Data type
1		Input	
2		Count	UDInt
3		SourceIndex	DInt
4		DestinationIndex	DInt
5		Output	
6		InOut	
7		SourceArray	Variant
8		DestinationArray	Variant
9		Temp	
10		Error	Int
11		Constant	
12		<Add new>	
13		Return	
14		FC_PartialArrayCopy	Int

The example shows a section from the programming example "Example of moving data". You can find the detailed program code under "See also".

Note

Direct passing of a tag to a VARIANT tag is not possible, such as myVARIANT := #Variable

See also

Example of moving data (Page 2316)

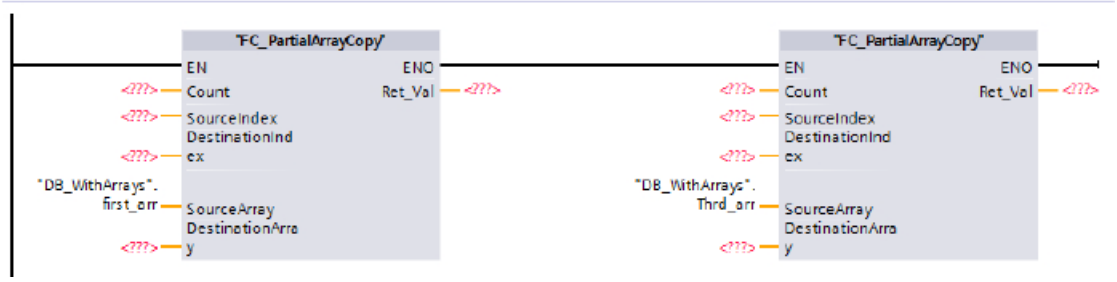
Transferring and reading out various data types with VARIANT (S7-1200, S7-1500)

The example shows a section from the programming example "Example of moving data". You can find the detailed program code under "See also".

Passing various data types

In the following example, you see how the VARIANT block parameter can be initialized with different tags when a generic, standardized function is called multiple times:

The "FC_PartialArrayCopy" function is called twice. The VARIANT parameter SourceArray is interconnected with an ARRAY of "my_struct" with the left call. The VARIANT parameter SourceArray is interconnected with an ARRAY of REAL with the right call.



Reading out and checking data types

Various comparison instructions are available to you to read out the data type of a tag or an element and compare it with data types of other tags or elements.

In the figure below, you see the usage of multiple comparison instructions to check whether the elements of the ARRAYs have the same data type:

```
IF IS_ARRAY(#SourceArray) AND IS_ARRAY(#DestinationArray)
    AND TypeOfElements(#SourceArray) = TypeOfElements(#DestinationArray) THEN
    #Error := MOVE_BLK_VARIANT(COUNT := #Count, SRC := #SourceArray, SRC_INDEX := #SourceIndex,
                               DEST => #DestinationArray, DEST_INDEX := #DestinationIndex);
END_IF;
```

The MOVE_BLK_VARIANT instruction is only executed if the data types of the ARRAY elements are the same.

See also

Example of moving data (Page 2316)

Examples of using the VARIANT data type (S7-1200, S7-1500)

Example of moving data (S7-1200, S7-1500)

Programming example

In this programming example, data values, which are collected for example during a production shift, are moved for further processing. The data is collected in an ARRAY. Using the "MOVE_BLK_VARIANT instruction: Move block", you can move either the entire ARRAY or only individual ARRAY elements dynamically and type-safe. You can freely select the ARRAY limits for the respective source and destination ARRAYs and these do not have to match. However, the data type of the data values that are to be moved must match. This instruction is available in all programming languages.

By using the VARIANT data type, you can use the created program code to move data for other production shifts as well, by specifying a different source and destination area at the block call.

Procedure

1. Create a function using the SCL programming language and give it the name "FC_PartialArrayCopy".
2. Declare the block interface as follows:

FC_PartialArrayCopy

	Name	Data type
1	Input	
2	Count	UDInt
3	SourceIndex	DInt
4	DestinationIndex	DInt
5	Output	
6	InOut	
7	SourceArray	Variant
8	DestinationArray	Variant
9	Temp	
10	Error	Int
11	Constant	
12	<Add new>	
13	Return	
14	FC_PartialArrayCopy	Int

3. Create the SCL program code as follows:
   You can find the program code below as template.

```
IF IS_ARRAY(#SourceArray) AND IS_ARRAY(#DestinationArray)
    AND TypeOfElements(#SourceArray) = TypeOfElements(#DestinationArray) THEN
    #Error := MOVE_BLK_VARIANT(COUNT := #Count, SRC := #SourceArray, SRC_INDEX := #SourceIndex,
                               DEST => #DestinationArray, DEST_INDEX := #DestinationIndex);
END_IF;
#FC_PartialArrayCopy := #Error;
```

4. Create the PLC data type "UDT_MyStruct":

UDT_MyStruct

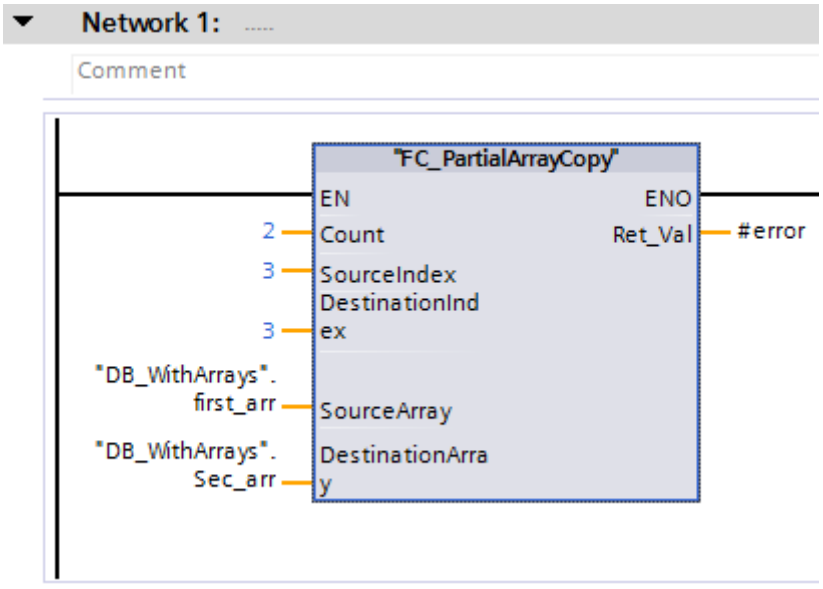
	Name	Data type	Default value
1	a	Bool	1
2	b	Bool	false
3	c	Bool	1
4	d	Bool	false
5	e	Bool	1
6	arr_int	Array[1..100] of Int	
7	f	Int	0
8	g	Real	0.0
9	h	LReal	0.0

5. Create the global data block "DB_WithArrays":

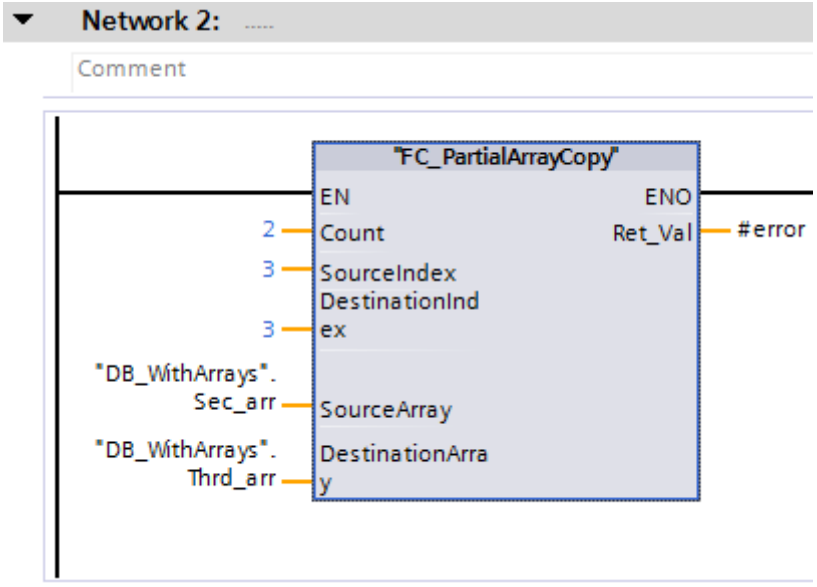
DB_WithArrays

	Name	Data type
1	Static	
2	first_arr	Array[0..100] of "UDT_MyStruct"
3	Sec_arr	Array[1..200] of "UDT_MyStruct"
4	Thrd_arr	Array[1..200] of Real
5	For_arr	Array[1..500] of Real

6. Call the "FC_PartialArrayCopy" function in an organization block, for example OB1, and initialize the parameters with the DB_WithArrays data block. Enter the named constants:



7. Instead of using the first two ARRAYs, which have the data type UDT_MyStruct, you can also use the third and fourth ARRAY, which have the data type REAL:



Result

As soon as the "FC_PartialArrayCopy" block is called in the program cycle, two data values, starting with the fourth element, are copied from the first ARRAY of the "DB_WithArrays" global data block to the second ARRAY of the data block. The copied data values are inserted in the second ARRAY, starting from the fourth element.

SCL program code for copying:

SCL

```
IF IS_ARRAY(#SourceArray) AND TypeOfElements(#SourceArray) =
TypeOfElements(#DestinationArray) THEN
#Error := MOVE_BLK_VARIANT(COUNT := #Count, SRC := #SourceArray, SRC_INDEX := #SourceIndex,
                           DEST => #DestinationArray, DEST_INDEX := #DestinationIndex);
END_IF;
#FC_PartialArrayCopy := #Error;
```

See also

Basic information on VARIANT (Page 2308)

Example of programming a queue (FIFO) (S7-1200, S7-1500)

Programming example

In the following example, you program a ring buffer which consists of an ARRAY and is written and read according to the FIFO principle. The program code has a read-VARIANT pointer and a write-VARIANT pointer. Using the VARIANT instructions, you can program the program code robustly and ensure reliable copying or deleting.

Using the VARIANT data type, program sections can be influenced during runtime. The VARIANT pointer is a type-safe pointer, i.e. a type test is performed during runtime. For blocks that have been created with the block property "optimized", sub-functions that were previously programmed with an ANY pointer can now be resolved with a VARIANT pointer. The VARIANT data type is used to transfer structures to system function blocks.

Procedure

1. Create an SCL function block and name it "FIFOQueue".
2. Declare the block interface as follows:

Declaration	Parameters	Data type	Comment
Input	request	BOOL	The instruction is executed when a positive signal edge is detected at the "request" parameter.
	mode	BOOL	0 = The first entry in the ring buffer is returned. 1 = An entry is written at the last position in the ring buffer.
	initialValue	VARIANT	Value with which the ARRAY of the ring buffer is initialized.
Output	error	INT	Error information
InOut	item	VARIANT	The entry that is either returned from the ring buffer or written to the ring buffer.
	buffer	VARIANT	An ARRAY that is used as a ring buffer.
Static	edgeupm	BOOL	Edge memory bit in which the RLO of the previous query is saved.
	firstItemIndex	INT	Index of the oldest entry in the ring buffer
	nextEmptyItemIndex	INT	Index of the next free entry in the ring buffer
Temp	edgeup	BOOL	Result of edge evaluation
	internalError	INT	Error information
	newFirstItemIndex	INT	Variable index
	newNextEmptyItemIndex	INT	Variable index
	bufferSize	UDINT	Number of ARRAY elements in the ring buffer

3. Create the following program code in the "FIFOQueue" function block:

```
(* This program code section is only executed once after a positive signal
edge. If there is no change in the signal state of the result of logic
operation, the program processing of the "FIFOQueue" FB is terminated. *)
#edgeup := #request & NOT #edgeupm;
#edgeupm := #request;
IF NOT (#edgeup) THEN
   RETURN;
END_IF;
```

```
// ------Validation of whether all parameter inputs are valid.----
(* This program code section checks whether the ring buffer is an ARRAY. If
so, the number of the ARRAY elements is read out. If it is not an ARRAY,
the program execution is terminated at this point and the error code "-10"
is output. *)
IF NOT (IS_ARRAY(#buffer)) THEN
   #error := -10;
   RETURN;
ELSE
   #bufferSize := CountofElements(#buffer);
END_IF;

(* This program code section checks whether the data type of the ARRAY
elements matches the data type of the entry (tag #item). If the data types
do not match, the program execution is terminated at this point and the
error code "-11" is output. *)
IF NOT (TypeOf(#item) = TypeOfElements(#buffer)) THEN
   #error := -11;
   RETURN;
END_IF;

(* This program code section checks whether the initial value of the ring
buffer matches the entry (tag #item). If the data types do not match, the
program execution is terminated at this point and the error code "-12" is
output. *)
IF NOT (TypeOf(#item) = TypeOf(#initialValue)) THEN
   #error := -12;
   RETURN;
END_IF;

(* This program code section checks whether the variable indices are within
the ARRAY limits. If this is not the case, the program execution is
terminated at this point and either error code "-20" or "-21" is output
depending on the index. *)
IF (#nextEmptyItemIndex >= #bufferSize) THEN
   #error := -20;
   RETURN;
END_IF;
IF (#firstItemIndex >= #bufferSize) THEN
   #error := -21;
   RETURN;
END_IF;
```

```
//-----------Program code execution, depending on the Mode
parameter-------------
// The execution of the instructions depends on the signal state of the Mode
parameter.
IF #mode = 0 THEN

// If the Mode parameter has the signal state "0", the first entry from the
passed ring buffer is returned.
(* This program code section checks whether the ring buffer is empty. If
this is the case, program execution is terminated at this point and the
error code "-40" is output. *)
   IF (#firstItemIndex = -1) THEN
      #error := -40;
      RETURN;
   END_IF;

// This program code section returns the first entry of the ring buffer.
   #internalError := MOVE_BLK_VARIANT(SRC := #buffer,
                                      COUNT := 1,
                                      SRC_INDEX := #firstItemIndex,
                                      DEST_INDEX := 0,
                                      DEST => #item);

   IF (#internalError = 0) THEN
(* This program code section checks whether the ring buffer contains ARRAY
elements. If it does, the first entry is passed on and the index is
incremented by 1. *)
      #internalError := MOVE_BLK_VARIANT(SRC := #initialValue,
                                         COUNT := 1,
                                         SRC_INDEX := 0,
                                         DEST_INDEX := #firstItemIndex,
                                         DEST => #buffer);

// This program code section calculates the new index of the first entry.
      #newFirstItemIndex := #firstItemIndex +1;
      #newFirstItemIndex := #newFirstItemIndex MOD
UDINT_TO_INT(#bufferSize);

// This program section checks whether the ring buffer is empty.
      IF (#nextEmptyItemIndex = #newFirstItemIndex) THEN
// If the ring buffer is empty, the index is set to 0.
         #firstItemIndex := -1;
         #nextEmptyItemIndex := 0;
      ELSE
// The index of the first entry is changed.
         #firstItemIndex := #newFirstItemIndex;
```

```
      END_IF;
   END_IF;
ELSE


// If the Mode parameter has the signal state "1", the entry is written to
the passed ring buffer.
(* This program code section checks whether the ring buffer is full. If this
is the case, program execution is terminated at this point and the error
code "-50" is output. *)
   IF (#nextEmptyItemIndex = #firstItemIndex) THEN
      #error := -50;
      RETURN;
   END_IF;

// This program code section writes the entry to the ring buffer.
   #internalError := MOVE_BLK_VARIANT(SRC := #item,
                                      COUNT := 1,
                                      SRC_INDEX := 0,
                                      DEST_INDEX := #nextEmptyItemIndex,
                                      DEST => #buffer);


   IF (#internalError = 0) THEN
// This program code section increments the index by 1 and calculates the
new empty entry index.
      #newNextEmptyItemIndex := #nextEmptyItemIndex +1;
      #newNextEmptyItemIndex := #newNextEmptyItemIndex MOD #bufferSize;
      #nextEmptyItemIndex := #newNextEmptyItemIndex;

(* This program code section checks which index the "#firstItemIndex" tag
has. If the number = -1, the ring buffer is initialized and the entry is
written to the ring buffer. This is why the tag value "0" must be assigned.
*)
      IF (#firstItemIndex = -1) THEN
         #firstItemIndex := 0;
      END_IF;
   END_IF;
END_IF;

//-------------------------Local error handling-----------------------------
(* This program code section checks whether a local error has occurred. If
this is the case, program execution is terminated at this point and the
error code "-100" is output. *)
IF (#internalError > 0) THEN
   #error := -100;
   RETURN;
```

```
END_IF;

// If no error occurred during program execution, the error code "0" is
output.
#error := 0;
```

Result

Call the SCL function block at the position in your program at which the FIFO queue is to run.

11.3.12 Parameter types

11.3.12.1 Parameter types

Description

The parameter types are data types for formal parameters that are transferred to called blocks. A parameter type can also be a PLC data type.

The following table shows the available parameter data types and their purpose:

Parameter type	Length (bits)	Description
TIMER	16	Is used to specify a timer that is used in the called code block. If you supply a formal parameter of the TIMER parameter type, the associated actual parameter must be a timer. Example: T1
COUNTER	16	Is used to specify a counter that is used in the called code block. If you supply a formal parameter of the COUNTER parameter type, the associated actual parameter must be a counter. Example: C10
BLOCK_FC	16	Is used to specify a block that is used as input in the called code block. The declaration of the parameter determines the block type (for example FB, FC, DB) that is to be used. If you supply a formal parameter of the BLOCK parameter type, specify a block address as the actual parameter. Example: DB3
BLOCK_FB	16	
BLOCK_DB	16	
BLOCK_SDB	16	
BLOCK_SFB	16	
BLOCK_SFC	16	
BLOCK_OB	16	

Parameter type	Length (bits)	Description
VOID	-	The VOID parameter type does not save any values. This parameter type is used if the return values of an output are not required. The VOID parameter type can be specified at the STATUS output, for example, if no error information is required.
PARAMETER	-	The PARAMETER data type enables the symbol of a local tag of the program block containing instructions "GetSymbolName: Read out name of a tag at input parameter" and "GetSymbolPath: Query composite global name of the input parameter assignment" to be used with the corresponding input of these instructions.

See also

Overview of the valid data types (Page 2239)

11.3.13 System data types

11.3.13.1 System data types

Description

The system data types (SDT) are made available by the system and have a predefined structure. The structure of a system data type consists of a fixed number of components that can have various data types. It is not possible to change the structure of a system data type.

System data types can only be assigned to one another if they are of the same type and their name matches. This also applies to PLC data types generated by the system, such as IEC_Timer, etc.

The system data types can only be used for specific instructions. The following table shows the available system data types and their purpose:

System data type	Length (bytes)	Description
IEC_TIMER	16	Structure of a timer in which the PT, ET, IN and Q parameters are declared. The time values are of TIME data type. This data type is used for the "TP", "TOF", "TON", "TONR", "RT" and "PT" instructions, for example.
IEC_LTIMER	32	Structure of a timer in which the PT, ET, IN and Q parameters are declared. The time values are of LTIME data type. This data type is used for the "TP", "TOF", "TON", "TONR", "RT" and "PT" instructions, for example.
IEC_SCOUNTER	3	Structure of a counter whose count values are of SINT data type. This data type is used for the "CTU", "CTD" and "CTUD" instructions, for example.

System data type	Length (bytes)	Description
IEC_USCOUNTER	3	Structure of a counter whose count values are of USINT data type. This data type is used for the "CTU", "CTD" and "CTUD" instructions, for example.
IEC_COUNTER	6	Structure of a counter whose count values are of INT data type. This data type is used for the "CTU", "CTD" and "CTUD" instructions, for example.
IEC_UCOUNTER	6	Structure of a counter whose count values are of UINT data type. This data type is used for the "CTU", "CTD" and "CTUD" instructions, for example.
IEC_DCOUNTER	12	Structure of a counter whose count values are of DINT data type. This data type is used for the "CTU", "CTD" and "CTUD" instructions, for example.
IEC_UDCOUNTER	12	Structure of a counter whose count values are of UDINT data type. This data type is used for the "CTU", "CTD" and "CTUD" instructions, for example.
IEC_LCOUNTER	24	Structure of a counter with count values of data type UDINT. This data type is used for the "CTU", "CTD" and "CTUD" instructions, for example.
IEC_ULCOUNTER	24	Structure of a counter with count values of data type UINT. This data type is used for the "CTU", "CTD" and "CTUD" instructions, for example.
ERROR_STRUCT	28	Structure of an error information to a programming or I/O access error. This data type is used, for example, for the "GET_ERROR" instruction.
CREF	8	Components of the ERROR_STRUCT data type, in which information about the address of a block is saved.
NREF	8	Components of the ERROR_STRUCT data type, in which information about the address of an operand is saved.
VREF	12	Is used for storage of a VARIANT pointer. This data type is, for example, used for instructions from S7-1200 Motion Control.
SSL_HEADER	4	Specifies the data structure in which information about the data records are saved during the reading of the system status lists. This data type is used, for example, for the "RDSYSST" instruction.
CONDITIONS	52	User-defined data structure defining the conditions for start and end of a data reception. This data type is used, for example, for the "RCV_CFG" instruction.

System data type	Length (bytes)	Description
TADDR_Param	8	Specifies the structure of a data block which stores descriptions of connections for Open User Communication via UDP. This data type is used for the "TUSEND" and "TURSV" instructions, for example.
TCON_Param	64	Specifies the structure of a data block which stores descriptions of connections for Open User Communication via Industrial Ethernet (PROFINET). This data type is used for the "TSEND" and "TRSV" instructions, for example.
HSC_Period	12	Specifies the structure of the data block for the time period measurement with the extended high-speed counter. This data type is used, for example, for the "CTRL_HSC_EXT" instruction.

See also

Overview of the valid data types (Page 2239)

11.3.14 Hardware data types

11.3.14.1 Hardware data types

Description

The hardware data types are made available by the CPU. The number of available hardware data types depends on the CPU.

Constants of a specific hardware data type are stored based on the modules set in the hardware configuration. When an instruction for controlling or activating a configured module is inserted in the user program, the available constants can be used for the parameters.

The following table shows the available hardware data types and their purpose:

Data type	Basic data type	Description
REMOTE	ANY	Serves to specify the address of a remote CPU. This data type is used, for example, for the "PUT" and "GET" instructions.
HW_ANY	UINT	Identification of any hardware component, e.g. a module.
HW_DEVICE	HW_ANY	Identification of a DP slave/PROFINET IO device
HW_DPMASTER	HW_INTERFACE	Identification of a DP master
HW_DPSLAVE	HW_DEVICE	Identification of a DP slave
HW_IO	HW_ANY	Identification number of the CPU or the interface The number is automatically allocated and is stored in the properties of the CPU or of the interface in the hardware configuration.

Data type	Basic data type	Description
HW_IOSYSTEM	HW_ANY	Identification of a PN/IO system or DP master system
HW_SUBMODULE	HW_IO	Identification of a central hardware component
HW_MODULE	HW_IO	Identification of a module
HW_INTERFACE	HW_SUBMODULE	Identification of an interface component
HW_IEPORT	HW_SUBMODULE	Identification of a port (PN/IO)
HW_HSC	HW_SUBMODULE	Identification of a high-speed counter This data type is used, for example, for the "CTRL_HSC" and "CTRL_HSC_EXT" instructions.
HW_PWM	HW_SUBMODULE	Identification of a pulse width modulation This data type is used, for example, for the "CTRL_PWM" instruction.
HW_PTO	HW_SUBMODULE	Identification of a pulse encoder This data type is used for Motion Control.
EVENT_ANY	AOM_IDENT	Used to identify any event
EVENT_ATT	EVENT_ANY	Is used to specify an event that can be assigned dynamically to an OB This data type is used, for example, for the "ATTACH" and "DETACH" instructions.
EVENT_HWINT	EVENT_ATT	Is used to specify a hardware interrupt event
OB_ANY	INT	Serves to specify any organization block.
OB_DELAY	OB_ANY	Used to specify an organization block that is called when a time-delay interrupt occurs. This data type is used, for example, for the "SRT_DINT" and "CAN_DINT" instructions.
OB_TOD	OB_ANY	Specifies the number of a time-of-day interrupt OB. This data type is used, for example, for the "SET_TINT", "CAN_TINT", "ACT_TINT" and "QRY_TINT" instructions.
OB_CYCLIC	OB_ANY	Is used to specify an organization block that is called when a watchdog interrupt occurs.
OB_ATT	OB_ANY	Is used to specify an organization block that can be assigned dynamically to an event. This data type is used, for example, for the "ATTACH" and "DETACH" instructions.
OB_PCYCLE	OB_ANY	Is used to specify an organization block that can be assigned to an event of the "Cyclic program" event class.
OB_HWINT	OB_ATT	Is used to specify an organization block that is called when a hardware interrupt occurs.
OB_DIAG	OB_ANY	Is used to specify an organization block that is called when a diagnostic interrupt occurs.
OB_TIMEERROR	OB_ANY	Is used to specify an organization block that is called when a time error occurs.
OB_STARTUP	OB_ANY	Is used to specify an organization block that is called when a startup event occurs.

Data type	Basic data type	Description
PORT	HW_SUBMODULE	Serves to specify a communication port. This data type is used for point-to-point communication.
RTM	UINT	Serves to specify the number of an operating hours counter. This data type is used, for example, for the "RTM" instruction.
PIP	UINT	Is used to create and connect a "Synchronous Cycle" OB. This data type is used for the SFCs 26, 27, 126 and 127.
CONN_ANY	WORD	Serves to specify any connection.
CONN_PRG	CONN_ANY	Serves to specify a connection for open communication over UDP.
CONN_OUC	CONN_ANY	Used to specify a connection for open communication over Industrial Ethernet (PROFINET).
CONN_R_ID	DWORD	Data type for the R_ID parameter on the S7 communication blocks.
DB_ANY	UINT	Identification (name or number) of a DB The data type "DB_ANY" has the length 0 in the section "Temp".
DB_WWW	DB_ANY	Number of a DB generated via the Web application (for example, "WWW" instruction) The data type "DB_WWW" has the length 0 in the section "Temp".
DB_DYN	DB_ANY	Number of a DB generated by the user program

See also

Overview of the valid data types (Page 2239)

11.3.15 Data type conversion for S7-1500: (S7-1500)

11.3.15.1 Overview of data type conversion (S7-1500)

Introduction

If you combine several operands in an instruction, you must make sure that the data types are compatible. This also applies when you assign or supply block parameters. If the operands are not of the same data type, a conversion has to be carried out.

There are two options for conversion:

- Implicit conversion
  Implicit conversion is supported by the programming languages LAD, FBD, SCL and GRAPH. Implicit conversion is not possible in the STL programming language.
- Explicit conversion
  You use an explicit conversion instruction before the actual instruction is executed.

Note

Converting bit strings to SCL

All bit strings (BYTE, WORD, DWORD and LWORD) are handled like the corresponding unsigned integers (USINT, UINT, UDINT and ULINT) in expressions. Therefore, implicit conversion from DWORD to REAL is carried out like a conversion from UDINT to REAL, for example.

Implicit conversion

Implicit conversion is executed automatically if the data types of the operands are compatible. This compatibility test can be carried out according to strict or less strict criteria:

- With IEC check (default setting)
  If IEC check is set, the following rules are applied:
  - Implicit conversion of BOOL into other data types is not possible.
  - Only the data types REAL, BYTE, WORD, DWORD, DINT, INT, SINT, UDINT, UINT, USINT, TIME, LDT, DTL, DT, TOD, WCHAR and CHAR can be converted implicitly.
  - The bit length of the source data type may not exceed the bit length of the target data type. A WORD data type operand, for example, cannot be specified at a parameter at which the BYTE data type is expected.
- Without IEC check
  If IEC check is not set, the following rules are applied:
  - Implicit conversion of BOOL into other data types is not possible.
  - Only the data types REAL, LREAL, BYTE, WORD, DWORD, LWORD, SINT, INT, DINT, LINT, USINT, UINT, UDINT, ULINT, TIME, LTIME, S5TIME, LDT, DTL, TOD, LTOD, DATE, STRING, WSTRING, WCHAR and CHAR can be converted implicitly.
  - The bit length of the source data type may not exceed the bit length of the target data type. A DWORD data type operand, for example, cannot be specified at a parameter at which the WORD data type is expected.
  - The bit length of an operand entered at in-out parameters (InOut) must be the same as the programmed bit length for the parameter in question.

  Note

  Implicit conversion without IEC check

  The programming editor uses a gray rectangle to mark operands that are implicitly converted. The dark gray rectangle signals that an implicit conversion is possible without any accuracy loss, for example, if you convert the data type INT to DINT. A light gray rectangle signals that implicit conversion is possible, but errors could occur during runtime. If, for example, you are converting the data type LINT to DINT and an overflow occurs, the enable output ENO is set to "0".

For more information about the setting of the IEC check and the implicit conversion, refer to "See also".

Explicit conversion

If the operands are not compatible and implicit conversion is therefore not possible, you can perform an explicit conversion. You can do this by using the conversion instructions in the "Instructions" Task Card or you can insert the conversion into the program manually. You can find the format for explicit conversion functions under "See also".

Any overflow will be displayed on the ENO enable output. An overflow occurs, for example, if the value of the source data type is greater than the value of the target data type.

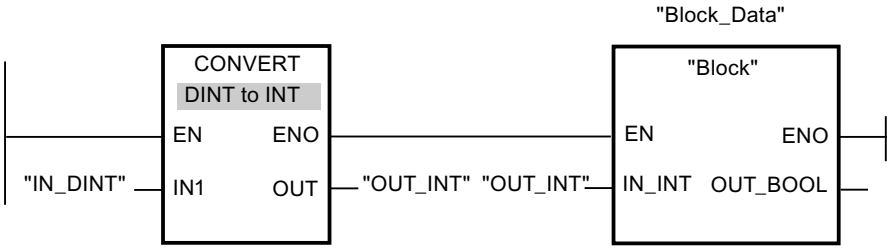
Note

Shifting of bit patterns

If the explicit conversion involves shifting a bit pattern, the enable output ENO is not set.

You can find additional information on explicit conversion under "See also".

The following figure shows an example in which explicit data type conversion must be carried out:



The "Block" function block expects a tag of the INT data type at the "IN_INT" input parameter. The value of the "IN_DINT" tag has therefore to the converted first from DINT to INT. Conversion is performed if the value of the "IN_DINT" tag is within the admissible value range for the INT data type. Otherwise an overflow is reported. However, a conversion takes place even in the event of an overflow, but the values are truncated and the ENO enable output is set to "0".

See also

Implicit conversions (Page 2333)

Explicit conversions (Page 2382)

11.3.15.2 Implicit conversions (S7-1500)

Setting and disabling the IEC check (S7-1500)

The data types of the operands used are checked for compatibility. This compatibility test can be carried out according to criteria that are more or less strict. If "IEC check" is activated, stricter criteria are applied.

You can set the IEC check centrally for all new blocks of the project or for individual blocks.

Setting IEC check for new blocks

To set the IEC check for all new blocks in the project, proceed as follows:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "PLC programming > General" group in the area navigation.
3. Select or clear the "IEC check" check box in the "Default settings for new blocks" group.
   The IEC check is enabled or disabled for all new blocks in the program.

Setting IEC check for a block

To set the IEC check for a block, proceed as follows:

1. Open the block.
2. Open the "Properties" tab in the inspector window.
3. Select the "Attributes" group in the area navigation.
4. Select or clear the "IEC check" check box.
   The IEC check is enabled or disabled for this block. The setting is stored together with the project.

Binary numbers (S7-1500)

Implicit conversion of BOOL (S7-1500)

Implicit conversion options

The BOOL data type cannot be implicitly converted.

See also

BOOL (bit) (Page 2243)

Overview of data type conversion (Page 2330)

Setting and disabling the IEC check (Page 2333)

Explicit conversion of BOOL (Page 2382)

Bit strings (S7-1500)

Implicit conversion of BYTE (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of BYTE data type:

Source	Destination	With IEC check	Without IEC check	Explanation
BYTE	BOOL	-	-	No implicit conversion
	WORD	X	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	DWORD	X	X	
	LWORD	X	X	
	SINT	-	X	
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LINT	-	X	
	ULINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

BYTE (Page 2244)

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)
Explicit conversion of BYTE (Page 2384)

Implicit conversion of WORD (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of WORD data type:

Source	Destination	With IEC check	Without IEC check	Explanation
WORD	BOOL	-	-	No implicit conversion
	BYTE	-	X	The low byte is transferred to the destination data type; the high byte is ignored.
	DWORD	X	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	LWORD	X	X	
	SINT	-	X	The low byte is transferred to the destination data type; the high byte is ignored.
	USINT	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LINT	-	X	
	ULINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	LDT	-	-	No implicit conversion
	DTL	-	-	
	DT	-	-	
	DATE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	TOD	-	-	No implicit conversion
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

WORD (Page 2245)
Setting and disabling the IEC check (Page 2333)
Overview of data type conversion (Page 2330)
Explicit conversion of WORD (Page 2387)

Implicit conversion of DWORD (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of DWORD data type:

Source	Destination	With IEC check	Without IEC check	Explanation
DWORD	BOOL	-	-	No implicit conversion
	BYTE	-	X	The right byte is transferred to the destination data type; the left byte is ignored.
	WORD	-	X	
	LWORD	X	X	
	SINT	-	X	
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	UDINT	-	X	
	LINT	-	X	
	ULINT	-	X	
	REAL	-	X	The value is converted to the destination data type format. (The value "-1", for example, is converted to the value "-1.0".)
	LREAL	-	-	No implicit conversion
	TIME	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	LTIME	-	-	No implicit conversion
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	LTOD	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

DWORD (Page 2245)

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)
Explicit conversion of DWORD (Page 2390)

Implicit conversion of LWORD (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of LWORD data type:

Source	Destination	With IEC check	Without IEC check	Explanation
LWORD	BOOL	-	-	No implicit conversion
	BYTE	-	X	The right byte is transferred to the destination data type; the left byte is ignored.
	WORD	-	X	
	DWORD	-	X	
	SINT	-	X	
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LINT	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	ULINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	X	The value is converted to the destination data type format. (The value "-1", for example, is converted to the value "-1.0".)
	TIME	-	-	No implicit conversion
	LTIME	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	S5TIME	-	-	No implicit conversion
	LDT	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	DTL	-	-	No implicit conversion
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

LWORD (Page 2246)

Explicit conversion of LWORD (Page 2394)

Integers (S7-1500)

Implicit conversion of SINT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of SINT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
SINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. The remaining bits are filled with "0".
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	USINT	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value transfer from SINT #-1 -> INT #-1, not filled with "0".)
	INT	X	X	
	UINT	-	X	
	DINT	X	X	
	UDINT	-	X	
	LINT	X	X	
	ULINT	-	X	
	REAL	X	X	The value is converted to the destination data type format. (The value "-1", for example, is converted to the value "-1.0".)
	LREAL	X	X	
	TIME	-	-	No implicit conversion
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

SINT (8-bit integers) (Page 2248)

Explicit conversion of SINT (Page 2398)

Implicit conversion of USINT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of USINT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
USINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. The remaining bits are filled with "0".
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from USINT #10 -> DINT #10 or USINT #128 -> SINT #-128)
	INT	X	X	
	UINT	X	X	
	DINT	X	X	
	UDINT	X	X	
	LINT	X	X	
	ULINT	X	X	
	REAL	X	X	The value is converted to the destination data type format. (The value "-1", for example, is converted to the value "-1.0".)
	LREAL	X	X	
	TIME	-	-	No implicit conversion
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	

x: Conversion possible

-: Conversion not possible

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

USINT (8-bit integers) (Page 2249)

Explicit conversion of USINT (Page 2402)

Implicit conversion of INT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of INT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
INT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from INT #-1 -> SINT #-1, or INT #-32 767 -> UINT #32 769)
	USINT	-	X	
	UINT	-	X	
	DINT	X	X	
	UDINT	-	X	
	LINT	X	X	
	ULINT	-	X	
	REAL	X	X	The value is converted to the destination data type format. (The value "-1", for example, is converted to the value "-1.0".)
	LREAL	X	X	
	TIME	-	-	No implicit conversion
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	TOD	-	-	No implicit conversion
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

INT (16-bit integers) (Page 2250)

Explicit conversion of INT (Page 2405)

Implicit conversion of UINT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of UINT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
UINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from UINT #100 -> DINT #100 or UINT #60 000 -> INT #-5536)
	USINT	-	X	
	INT	-	X	
	DINT	X	X	
	UDINT	X	X	
	LINT	X	X	
	ULINT	X	X	
	REAL	X	X	The value is converted to the destination data type format. (The value "-1", for example, is converted to the value "-1.0".)
	LREAL	X	X	
	TIME	-	-	No implicit conversion
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	TOD	-	-	No implicit conversion
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

UINT (16-bit integers) (Page 2251)

Explicit conversion of UINT (Page 2409)

Implicit conversion of DINT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of DINT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
DINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from DINT #-1 -> SINT #-1 or DINT #-1 -> USINT #255)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	UDINT	-	X	
	LINT	X	X	
	ULINT	-	X	
	REAL	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from DINT #-1 -> REAL #-1.0, but there is a loss in accuracy for numbers with an absolute value greater than 8 388 608)
	LREAL	X	X	The value is converted to the destination data type format. (The value "-1", for example, is converted to the value "-1.0".)
	TIME	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	LTIME	-	-	No implicit conversion
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	LTOD	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

DINT (32-bit integers) (Page 2252)

Explicit conversion of DINT (Page 2412)

Implicit conversion of UDINT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of UDINT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
UDINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from DINT #-1 -> SINT #-1 or DINT #-1 -> USINT #255)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	LINT	X	X	
	ULINT	X	X	
	REAL	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from DINT #-1 -> REAL #-1.0, but there is a loss in accuracy for numbers with an absolute value greater than 8 388 608)
	LREAL	X	X	The value is converted to the destination data type format. (The value "-1", for example, is converted to the value "-1.0".)
	TIME	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	LTIME	-	-	No implicit conversion
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	LTOD	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

UDINT (32-bit integers) (Page 2253)

Explicit conversion of UDINT (Page 2416)

Implicit conversion of LINT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of LINT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
LINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from LINT #-1 -> SINT #-1 or LINT #-1 -> USINT #255)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	ULINT	-	X	
	REAL	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from LINT #-1 -> REAL #-1.0, but there is a loss in accuracy for numbers with an absolute value greater than 8 388 608)
	LREAL	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from LINT #-1 -> REAL #-1.0, but there is a loss in accuracy for numbers with an absolute value greater than 9 007 199 254 740 992)
	TIME	-	-	No implicit conversion
	LTIME	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. (duration in nanoseconds)
	S5TIME	-	-	No implicit conversion
	LDT	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. (nanoseconds since 1/1/1970)
	DTL	-	-	No implicit conversion
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

LINT (64-bit integers) (Page 2254)

Explicit conversion of LINT (Page 2419)

Implicit conversion of ULINT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of ULINT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
ULINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from ULINT #-1 -> SINT #-1 or ULINT #-1 -> USINT #255)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LINT	-	X	
	REAL	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from LINT #-1 -> REAL #-1.0, but there is a loss in accuracy for numbers with an absolute value greater than 8 388 608)
	LREAL	-	X	The bit pattern of the source value is converted and transferred to the destination data type. (for example, value conversion from LINT #-1 -> REAL #-1.0, but there is a loss in accuracy for numbers with an absolute value greater than 9 007 199 254 740 992)
	TIME	-	-	No implicit conversion
	LTIME	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. (duration in nanoseconds)
	S5TIME	-	-	No implicit conversion
	LDT	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. (nanoseconds since 1/1/1970)
	DTL	-	-	No implicit conversion
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

ULINT (64-bit integers) (Page 2255)

Explicit conversion of ULINT (Page 2423)

Floating-point numbers (S7-1500)

Implicit conversion of REAL (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of REAL data type:

Source	Destination	With IEC check	Without IEC check	Explanation
REAL	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	LWORD	-	-	No implicit conversion
	SINT	-	X	The bit pattern of the source value is rounded off and converted and transferred to the destination data type. (For example, rounding off and value conversion of REAL #2.5 -> INT #2, or negative numbers REAL #-2.5 -> INT #-2 -> USINT #254. With an overflow, the remainder is determined REAL #305.5 -> INT #306 -> USINT #50)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LINT	-	X	
	ULINT	-	X	
	LREAL	X	X	The value is transferred to the destination data type.
	TIME	-	-	No implicit conversion
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

REAL (Page 2257)

Explicit conversion of REAL (Page 2426)

Implicit conversion of LREAL (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of LREAL data type:

Source	Destination	With IEC check	Without IEC check	Explanation
LREAL	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	LWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	SINT	-	X	The bit pattern of the source value is rounded off and converted and transferred to the destination data type. (For example, rounding off and value conversion of LREAL #2.5 -> INT #2, or negative numbers LREAL #-2.5 -> INT #-2 -> USINT #254. With an overflow, the remainder is determined LREAL #305.5 -> INT #306 -> USINT #50)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LINT	-	X	
	ULINT	-	X	
	REAL	-	X	The value is transferred to the destination data type.
	TIME	-	-	No implicit conversion
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

LREAL (Page 2258)

Explicit conversion of LREAL (Page 2429)

Timers (S7-1500)

Implicit conversion of S5TIME (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of data type S5TIME:

Source	Destination	With IEC check	Without IEC check	Explanation
S5TIME	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion shows the duration in milliseconds.
	DWORD	-	-	No implicit conversion
	LWORD	-	-	
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	LINT	-	-	
	ULINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	

x: Conversion possible

-: Conversion not possible

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

S5TIME (duration) (Page 2261)

Explicit conversion of S5TIME (Page 2432)

Implicit conversion of TIME (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of TIME data type:

Source	Destination	With IEC check	Without IEC check	Explanation
TIME	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion shows the duration in milliseconds.
	LWORD	-	-	No implicit conversion
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion shows the duration in milliseconds.
	UDINT	-	X	
	LINT	-	-	No implicit conversion
	ULINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	S5TIME	-	-	
	LTIME	X	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion shows the duration in nanoseconds. (1 ms = 1 000 000 ns)
	LDT	-	-	No implicit conversion
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	X	If the source value is between 0 s and 84599.999 s, the bit pattern of the source value is transferred unchanged to the destination data type. (Display in nanoseconds) The target value remains unchanged otherwise. The result of the conversion shows the time that has passed since midnight.
	LTOD	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

TIME (IEC time) (Page 2262)

Explicit conversion of TIME (Page 2434)

Implicit conversion of LTIME (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of LTIME data type:

Source	Destination	With IEC check	Without IEC check	Explanation
LTIME	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	LWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion shows the duration in nanoseconds.
	SINT	-	-	No implicit conversion
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	LINT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion shows the duration in nanoseconds.
	ULINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	S5TIME	-	-	
	TIME	-	X	If the source value is outside the value range of the destination data type, the target value remains unchanged. (0.123456789 s becomes 0.123 s)
	LDT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion shows the duration in nanoseconds since 1/1/1970.
	DTL	-	-	No implicit conversion
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion shows the duration in nanoseconds since 1/1/1970.
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

LTIME (IEC time) (Page 2263)

Explicit conversion of LTIME (Page 2437)

Date and time-of-day (S7-1500)

Implicit conversion of DT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of DT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
DT	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	LWORD	-	-	
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	LINT	-	-	
	ULINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	-	
	LDT	X	X	The source value is transferred unchanged with the same value to the destination data type. (12/24/2012 14:30 remains 12/24/2012 14:30)
	DTL	X	X	
	DATE	-	-	No implicit conversion
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

DATE_AND_TIME (date and time of day) (Page 2265)

Explicit conversion of DT (Page 2440)

Implicit conversion of LDT (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of LDT data type:

Source	Destination	With IEC check	Without IEC check	Explanation
LDT	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	LWORD	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. (nanoseconds since 1/1/1970)
	SINT	-	-	No implicit conversion
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	LINT	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. (nanoseconds since 1/1/1970)
	ULINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. (nanoseconds since 1/1/1970)
	S5TIME	-	-	No implicit conversion
	DT	-	X	The bit pattern of the source value is converted with a loss in accuracy to the destination data type. (12/24/2012 12:34:56.123456789 becomes 12/24/2012 12:34:56.123)
	DTL	X	X	The source value is transferred unchanged with the same value to the destination data type. (12/24/2012 14:30 remains 12/24/2012 14:30)
	DATE	-	-	No implicit conversion
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

Explicit conversion of LDT (Page 2443)

LDT (DATE_AND_LTIME) (Page 2266)

Implicit conversion of DTL (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of DTL data type:

Source	Destination	With IEC check	Without IEC check	Explanation
DTL	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	LWORD	-	-	
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	LINT	-	-	
	ULINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	-	
	LDT	X	X	The source value is transferred unchanged with the same value to the destination data type. (12/24/2012 14:30 remains 12/24/2012 14:30)
	DT	-	X	The bit pattern of the source value is converted with a loss in accuracy to the destination data type. (12/24/2012 12:34:56.123456789 becomes 12/24/2012 12:34:56.123)
	DATE	-	-	No implicit conversion
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

DTL (Page 2267)

Explicit conversion of DTL (Page 2446)

Implicit conversion of DATE (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of data type DATE:

Source	Destination	With IEC check	Without IEC check	Explanation
DATE	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion is equal to the number of days since 01/01/1990.
	DWORD	-	-	No implicit conversion
	LWORD	-	-	
	SINT	-	-	
	USINT	-	-	
	INT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion is equal to the number of days since 01/01/1990.
	UINT	-	X	
	DINT	-	-	No implicit conversion
	UDINT	-	-	
	LINT	-	-	
	ULINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DT	-	-	
	DTL	-	-	
	TOD	-	-	
	LTOD	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	

x: Conversion possible
-: Conversion not possible

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

DATE (Page 2264)

Explicit conversion of DATE (Page 2449)

Implicit conversion of TOD (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of TOD data type:

Source	Destination	With IEC check	Without IEC check	Explanation
TOD	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion corresponds to the number of milliseconds since the start of day (0:00).
	LWORD	-	-	No implicit conversion
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion corresponds to the number of milliseconds since the start of day (0:00).
	UDINT	-	X	
	LINT	-	-	No implicit conversion
	ULINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion corresponds to the number of milliseconds since the start of day (0:00).
	LTIME	-	-	No implicit conversion
	S5TIME	-	-	
	LDT	-	-	
	DT	-	-	
	DTL	-	-	
	DATE	-	-	
	LTOD	X	X	The source value is transferred unchanged with the same value to the destination data type. (12:34:56.123 becomes 12:34:56.123000000)
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	

x: Conversion possible

-: Conversion not possible

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

TIME_OF_DAY (TOD) (Page 2264)

Explicit conversion of TOD (Page 2451)

Implicit conversion of LTOD (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of LTOD data type:

Source	Destination	With IEC check	Without IEC check	Explanation
LTOD	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	LWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion corresponds to the number of nanoseconds since the start of day (0:00 hrs).
	SINT	-	-	No implicit conversion
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	LINT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion corresponds to the number of nanoseconds since the start of day (0:00 hrs).
	ULINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion corresponds to the number of nanoseconds since the start of day (0:00 hrs).
	S5TIME	-	-	No implicit conversion
	LDT	-	-	
	DT	-	-	
	DTL	-	-	
	DATE	-	-	
	TOD	-	X	The source value is transferred unchanged with the same value rounded off to the destination data type. (12:34:56.123456789 becomes 12:34:56.123)
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	

x: Conversion possible

-: Conversion not possible

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

Explicit conversion of LTOD (Page 2453)

LTOD (LTIME_OF_DAY) (Page 2265)

String (S7-1500)

Implicit conversion of CHAR (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of CHAR data type:

Source	Destination	With IEC check	Without IEC check	Explanation
CHAR	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. The remaining bits are filled from the left with "0".
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LINT	-	X	
	ULINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	WCHAR	-	-	
	STRING	X	X	The STRING is shortened to length 1 and includes the character.
	WSTRING	-	-	No implicit conversion
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

CHAR (Page 2269)

Explicit conversion of CHAR (Page 2456)

Implicit conversion of WCHAR (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of WCHAR data type:

Source	Destination	With IEC check	Without IEC check	Explanation
WCHAR	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. The remaining bits are filled from the left with "0".
	WORD	-	X	
	DWORD	-	X	
	LWORD	-	X	
	SINT	-	X	
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LINT	-	X	
	ULINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	CHAR	-	-	
	STRING	-	-	
	WSTRING	X	X	The WSTRING is shortened to length 1 and includes the character.
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

WCHAR (Page 2269)

Explicit conversion of WCHAR (Page 2458)

Implicit conversion of STRING (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of STRING data type:

Source	Destination	With IEC check	Without IEC check	Explanation
STRING	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	LWORD	-	-	
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	LINT	-	-	
	ULINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	CHAR	-	X	The first character of the STRING is returned if the STRING includes one or more characters. Otherwise, the character is output with coding $00.
	WCHAR	-	-	No implicit conversion
	WSTRING	-	-	
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

STRING (Page 2270)

Explicit conversion of STRING (Page 2460)

Implicit conversion of WSTRING (S7-1500)

Implicit conversion options

The following table shows the options for the implicit conversion of WSTRING data type:

Source	Destination	With IEC check	Without IEC check	Explanation
WSTRING	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	LWORD	-	-	
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	LINT	-	-	
	ULINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	LTIME	-	-	
	S5TIME	-	-	
	LDT	-	-	
	DTL	-	-	
	DT	-	-	
	DATE	-	-	
	TOD	-	-	
	LTOD	-	-	
	CHAR	-	-	
	WCHAR	-	X	The first character of the WSTRING is returned if the WSTRING includes one or more characters. Otherwise, the character is output with coding $0000.
	STRING	-	-	No implicit conversion
x: Conversion possible -: Conversion not possible				

See also

Setting and disabling the IEC check (Page 2333)

Overview of data type conversion (Page 2330)

WSTRING (Page 2273)

Explicit conversion of WSTRING (Page 2463)

11.3.15.3 Explicit conversions (S7-1500)

Binary numbers (S7-1500)

Explicit conversion of BOOL (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the BOOL data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
BOOL	BYTE	X	Only the LSB (Least Significant Bit) is set in the destination data type. The enable output ENO is always "1".	BOOL_TO_BYTE
	WORD	X		BOOL_TO_WORD
	DWORD	X		BOOL_TO_DWORD
	LWORD	X		BOOL_TO_LWORD
	SINT	X		BOOL_TO_SINT
	USINT	X		BOOL_TO_USINT
	INT	X		BOOL_TO_INT
	UINT	X		BOOL_TO_UINT
	DINT	X		BOOL_TO_DINT
	UDINT	X		BOOL_TO_UDINT
	LINT	X		BOOL_TO_LINT
	ULINT	X		BOOL_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	-		-
	LTIME	-		-
	S5TIME	-		-
	LDT	-		-
	DT	-		-
	DTL	-		-
	TOD	-		-
	LTOD	-		-
	DATE	-		-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-
x: Conversion possible - : Conversion not possible				

See also

Overview of data type conversion (Page 2330)

Implicit conversion of BOOL (Page 2334)

BOOL (bit) (Page 2243)

Bit strings (S7-1500)

Explicit conversion of BYTE (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the BYTE data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
BYTE[1)]	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	BYTE_TO_BOOL
	WORD[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	BYTE_TO_WORD
	DWORD[1)]	X		BYTE_TO_DWORD
	LWORD[1)]	X		BYTE_TO_LWORD
	SINT	X		BYTE_TO_SINT
	USINT	X		BYTE_TO_USINT
	INT	X		BYTE_TO_INT
	UINT	X		BYTE_TO_UINT
	DINT	X		BYTE_TO_DINT
	UDINT	X		BYTE_TO_UDINT
	LINT	X		BYTE_TO_LINT
	ULINT	X		BYTE_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	BYTE_TO_TIME
	LTIME	X		BYTE_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	BYTE_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-
	TOD	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	BYTE_TO_TOD
	LTOD	X		BYTE_TO_LTOD
	DATE	X		BYTE_TO_DATE
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	BYTE_TO_CHAR
	WCHAR	X		BYTE_TO_WCHAR

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD, LWORD) are interpreted as an unsigned integer with the same bit length. The data type BYTE is interpreted as USINT, WORD as UINT, DWORD as UDINT and LWORD as ULINT.				

See also

Overview of data type conversion (Page 2330)

Implicit conversion of BYTE (Page 2335)

BYTE (Page 2244)

Explicit conversion of WORD (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the WORD data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
WORD[1)]	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	WORD_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	WORD_TO_BYTE
	DWORD[1)]	X		WORD_TO_DWORD
	LWORD[1)]	X		WORD_TO_LWORD

	SINT	X	ENO = TRUE
			#sint1 := WORD_TO_SINT(16#FFFF); // -1 to
			#sint1 := WORD_TO_SINT(16#FF80); // -128
			#sint1 := WORD_TO_SINT(16#0); // 0 to
			#sint1 := WORD_TO_SINT(16#007F); // 127
			
			ENO = FALSE
			#sint1 := WORD_TO_SINT(16#FF7F); // -129 to
			#sint1 := WORD_TO_SINT(16#8000); // -32768
			#sint1 := WORD_TO_SINT(16#0080); // 128 to
			#sint1 := WORD_TO_SINT(16#7FFF); // 32767
				WORD_TO_SINT

	USINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	WORD_TO_USINT
	INT	X		WORD_TO_INT
	UINT	X		WORD_TO_UINT
	DINT	X		WORD_TO_DINT
	UDINT	X		WORD_TO_UDINT
	LINT	X		WORD_TO_LINT
	ULINT	X		WORD_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	WORD_TO_TIME
	LTIME	X		WORD_TO_LTIME
	S5TIME	X		WORD_TO_S5TIME
	LDT	X		WORD_TO_LDT
	DT	-	No explicit conversion	-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
	DTL	-		-
	TOD	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	WORD_TO_TOD
	LTOD	X		WORD_TO_LTOD
	DATE	X		WORD_TO_DATE
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	WORD_TO_CHAR
	WCHAR	X		WORD_TO_WCHAR
WORD_ BCD16	INT	X	The value to be converted has data type WORD and is accepted as a BCD-coded value between -999 and +999. The result is available after conversion as an integer (in binary notation) of the type INT. A real conversion takes place. If the bit pattern includes an invalid tetrad, a synchronous error is not triggered but only the status bit OV is set instead.	WORD_BCD16_TO_INT
BCD16	INT	X		BCD16_TO_INT

x: Conversion possible

- : Conversion not possible

[1] Bit strings (BYTE, WORD, DWORD, LWORD) are interpreted as an unsigned integer with the same bit length. The data type BYTE is interpreted as USINT, WORD as UINT, DWORD as UDINT and LWORD as ULINT.

See also

Implicit conversion of WORD (Page 2336)

Overview of data type conversion (Page 2330)

WORD (Page 2245)

Explicit conversion of DWORD (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DWORD data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
DWORD[1)]	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	DWORD_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	DWORD_TO_BYTE
	WORD[1)]	X		DWORD_TO_WORD
	LWORD[1)]	X		DWORD_TO_LWORD

	SINT	X	ENO = TRUE
			#sint1 := DWORD_TO_SINT(16#FFFF_FFFF); // -1 to
			#sint1 := DWORD_TO_SINT(16#FFFF_FF80); // -128
			#sint1 := DWORD_TO_SINT(16#0); // 0 to
			#sint1 := DWORD_TO_SINT(16#0000_007F); // 127
			
			ENO = FALSE
			#sint1 := DWORD_TO_SINT(16#FFFF_FF7F); // -129
			#sint1 := DWORD_TO_SINT(16#8000_0000); // -2147483648
			#sint1 := DWORD_TO_SINT(16#0000_0080); // 128 to
			#sint1 := DWORD_TO_SINT(16#7FFF_FFFF); // 2147483647
				DWORD_TO_SINT

	USINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	DWORD_TO_USINT

	INT	X	ENO = TRUE
			#int1 := DWORD_TO_INT(16#FFFF_FFFF); // -1 to
			#int1 := DWORD_TO_INT(16#FFFF_8000); // -32768
			#int1 := DWORD_TO_INT(16#0); // 0 to
			#int1 := DWORD_TO_INT(16#0000_7FFF); // 32767
			
			ENO = FALSE
			#int1 := DWORD_TO_INT(16#FFFF_7FFF); // -32769
			#int1 := DWORD_TO_INT(16#8000_0000); //
				DWORD_TO_INT

Source	Destination	Conversion	Explanation	Mnemonics of the instruction

			-2147483648
			#int1 := DWORD_TO_INT(16#8000); // 32768 to
			#int1 := DWORD_TO_INT(16#7FFF_FFFF); // 2147483647	

	UINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If no errors occur during the conversion, the signal state of ENO = 1; if an error occurs during processing, the signal state of ENO = 0.	DWORD_TO_UINT
	DINT	X		DWORD_TO_DINT
	UDINT	X		DWORD_TO_UDINT
	LINT	X		DWORD_TO_LINT
	ULINT	X		DWORD_TO_ULINT
	REAL	X		DWORD_TO_REAL
	LREAL	-	No explicit conversion	-
	TIME	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	DWORD_TO_TIME
	LTIME	X		DWORD_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	DWORD_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-
	TOD	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	DWORD_TO_TOD
	LTOD	X		DWORD_TO_LTOD
	DATE	X		DWORD_TO_DATE
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	DWORD_TO_CHAR
	WCHAR	X		DWORD_TO_WCHAR

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
DWORD_BCD32	DINT	X	The value to be converted has data type DWORD and is accepted as a BCD-coded value between -9999999 and +9999999. The result is available after conversion as an integer (in binary notation) of the type DINT. A real conversion takes place. If the bit pattern includes an invalid tetrad, a synchronous error is not triggered but only the status bit OV is set instead.	DWORD_BCD32_TO_DINT
BCD32	DINT	X		BCD32_TO_DINT

x: Conversion possible

- : Conversion not possible

[1] Bit strings (BYTE, WORD, DWORD, LWORD) are interpreted as an unsigned integer with the same bit length. The data type BYTE is interpreted as USINT, WORD as UINT, DWORD as UDINT and LWORD as ULINT.

See also

Implicit conversion of DWORD (Page 2338)

Overview of data type conversion (Page 2330)

DWORD (Page 2245)

Explicit conversion of LWORD (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the LWORD data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
LWORD[1)]	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bit is not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	LWORD_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LWORD_TO_BYTE
	WORD[1)]	X		LWORD_TO_WORD
	DWORD[1)]	X		LWORD_TO_DWORD

	SINT	X	ENO = TRUE
			#sint1 := LWORD_TO_SINT(16#FFFF_FFFF_FFFF_FFFF); // -1 to
			#sint1 := LWORD_TO_SINT(16#FFFF_FFFF_FFFF_FF80); // -128
			#sint1 := LWORD_TO_SINT(16#0); // 0 to
			#sint1 := LWORD_TO_SINT(16#0000_0000_0000_007F); // 127
			
			ENO = FALSE
			#sint1 := LWORD_TO_SINT(16#FFFF_FFFF_FFFF_FF7F); // -129
			#sint1 := LWORD_TO_SINT(16#8000_0000_0000_0000); // -9223372036854775808
			#sint1 := LWORD_TO_SINT(16#0000_0000_0000_0080); // 128
			#sint1 := LWORD_TO_SINT(16#7FFF_FFFF_FFFF_FFFF); // 9223372036854775807
				LWORD_TO_SINT

	USINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LWORD_TO_USINT

	INT	X	ENO = TRUE
			#int1 := LWORD_TO_INT(16#FFFF_FFFF_FFFF_FFFF); // -1 to
			#int1 := LWORD_TO_INT(16#FFFF_FFFF_FFFF_8000); // -32768
			#int1 := LWORD_TO_INT(16#0); // 0 to
			#int1 := LWORD_TO_INT(16#0000_0000_0000_7FFF); // 32767
			
			ENO = FALSE
			#int1 := LWORD_TO_INT(16#FFFF_FFFF_FFFF_7FFF); // -32769
			#int1 := LWORD_TO_INT(16#8000_0000_0000_0000); // -2147483648
				LWORD_TO_INT

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
			#int1 := LWORD_TO_INT(16#0000_0000_0000_8000); // 32768 to #int1 := LWORD_TO_INT(16#7FFF_FFFF_FFFF_FFFF); // 2147483647	
	UINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LWORD_TO_UINT

	DINT	X	ENO = TRUE
			#dint1 := LWORD_TO_DINT(16#FFFF_FFFF_FFFF_FFFF); // -1 to
			#dint1 := LWORD_TO_DINT(16#FFFF_FFFF_8000_0000); // -2147483648
			#dint1 := LWORD_TO_DINT(16#0); // 0 to
			#dint1 := LWORD_TO_DINT(16#0000_0000_7FFF_FFFF); // 2147483647
			
			ENO = FALSE
			#dint1 := LWORD_TO_DINT(16#FFFF_FFFF_7FFF_FFFF); // -2147483649 to
			#dint1 := LWORD_TO_DINT(16#8000_0000_0000_0000); // -9223372036854775808
			#dint1 := LWORD_TO_DINT(16#0000_0000_8000_0000); // 2147483648 to
			#dint1 := LWORD_TO_DINT(16#7FFF_FFFF_FFFF_FFFF); // 9223372036854775807
				LWORD_TO_DINT

	UDINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LWORD_TO_UDINT
	LINT	X		LWORD_TO_LINT
	ULINT	X		LWORD_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If no errors occur during the conversion, the signal state of ENO = 1; if an error occurs during processing, the signal state of ENO = 0.	LWORD_TO_LREAL
	TIME	X		LWORD_TO_TIME
	LTIME	X		LWORD_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LWORD_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-
	TOD	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LWORD_TO_TOD
	LTOD	X		LWORD_TO_LTOD
	DATE	X		LWORD_TO_DATE
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LWORD_TO_CHAR
	WCHAR	X		LWORD_TO_WCHAR

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD, LWORD) are interpreted as an unsigned integer with the same bit length. The data type BYTE is interpreted as USINT, WORD as UINT, DWORD as UDINT and LWORD as ULINT.				

See also

Implicit conversion of LWORD (Page 2339)

Overview of data type conversion (Page 2330)

LWORD (Page 2246)

Integers (S7-1500)

Explicit conversion of SINT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the SINT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
SINT	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bit is not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	SINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	SINT_TO_BYTE
	WORD[1)]	X		SINT_TO_WORD
	DWORD[1)]	X		SINT_TO_DWORD
	LWORD[1)]	X		SINT_TO_LWORD
	USINT	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	SINT_TO_USINT
	INT	X		SINT_TO_INT
	UINT	X		SINT_TO_UINT
	DINT	X		SINT_TO_DINT
	UDINT	X		SINT_TO_UDINT
	LINT	X		SINT_TO_LINT
	ULINT	X		SINT_TO_ULINT
	REAL	X	The value is converted to the destination data type format. (The value "-1" is changed to the value "-1.0", for example, with the instruction "Convert value" (CONVERT).	SINT_TO_REAL, NORM_X
	LREAL	X		SINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the destination data type and interpreted as milliseconds.	SINT_TO_TIME
	LTIME	X	The value is transferred to the destination data type and interpreted as nanoseconds.	SINT_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The result is returned in nanoseconds since 1970-1-1-0:0:0.0.	SINT_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-
	TOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value	SINT_TO_TOD

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
			"-1" (16#FFFFFFFF). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in milliseconds since 0:0)	
	LTOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in nanoseconds since 0:0)	SINT_TO_LTOD
	DATE	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in days since 1990-1-1)	SINT_TO_DATE
	STRING	X	The value is converted into a character string.	SINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X	• LAD/FBD: The first characters of the string are padded with spaces. The number of spaces depends on the length of the numerical value. Positive numeric values are output without a sign. • STL/SCL: No spaces are inserted. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	SINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	SINT_TO_CHAR
	WCHAR[1)]	X		SINT_TO_WCHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width including sign, and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of SINT (Page 2341)

Overview of data type conversion (Page 2330)

SINT (8-bit integers) (Page 2248)

Explicit conversion of USINT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the USINT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
USINT	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bit is not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	USINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	USINT_TO_BYTE
	WORD[1)]	X		USINT_TO_WORD
	DWORD[1)]	X		USINT_TO_DWORD
	LWORD[1)]	X		USINT_TO_LWORD
	SINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If the sign is changed during the conversion, the enable output ENO is set to "0".	USINT_TO_SINT
	INT	X	The bit pattern of the source value is converted and transferred to the destination data type.	USINT_TO_INT
	UINT	X		USINT_TO_UINT
	DINT	X		USINT_TO_DINT
	UDINT	X		USINT_TO_UDINT
	LINT	X		USINT_TO_LINT
	ULINT	X		USINT_TO_ULINT
	REAL	X	The value is converted to the destination data type format. (The value "1" is changed to the value "1.0", for example, with the instruction "Convert value" (CONVERT).	USINT_TO_REAL, NORM_X
	LREAL	X		USINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the destination data type and interpreted as milliseconds.	USINT_TO_TIME
	LTIME	X	The value is transferred to the destination data type and interpreted as nanoseconds.	USINT_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The result is returned in nanoseconds since 1970-1-1-0:0:0.0.	USINT_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
	TOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in milliseconds since 0:0)	USINT_TO_TOD
	LTOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in nanoseconds since 0:0)	USINT_TO_LTOD
	DATE	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in days since 1990-1-1)	USINT_TO_DATE
	STRING	X	The value is converted into a character string. If the permitted length of the character string is violated, the enable output ENO is set to "0".	USINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		USINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the destination data type.	USINT_TO_CHAR
	WCHAR[1)]	X		USINT_TO_WCHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width (the non-existing sign is replaced with zeros), and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of USINT (Page 2342)

Overview of data type conversion (Page 2330)

USINT (8-bit integers) (Page 2249)

Explicit conversion of INT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the INT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
INT	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bit is not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	INT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	INT_TO_BYTE
	WORD[1)]	X		INT_TO_WORD
	DWORD[1)]	X		INT_TO_DWORD
	LWORD[1)]	X		INT_TO_LWORD
	SINT	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	INT_TO_SINT
	USINT	X		INT_TO_USINT
	UINT	X		INT_TO_UINT
	DINT	X		INT_TO_DINT
	UDINT	X		INT_TO_UDINT
	LINT	X		INT_TO_LINT
	ULINT	X		INT_TO_ULINT
	REAL	X	The value is converted to the destination data type format. (The value "-1" is changed to the value "-1.0", for example, with the instruction "Convert value" (CONVERT).	INT_TO_REAL, NORM_X
	LREAL	X		INT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the destination data type and interpreted as milliseconds.	INT_TO_TIME
	LTIME	X	The value is transferred to the destination data type and interpreted as nanoseconds.	INT_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The result is returned in nanoseconds since 1970-1-1-0:0:0.0.	INT_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-
	TOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO	INT_TO_TOD

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
			is set to "0". (interpretation in milliseconds since 0:0; check for 24h limit)	
	LTOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in nanoseconds since 0:0; check for 24h limit)	INT_TO_LTOD
	DATE	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in days since 1990-1-1; check for negative value)	INT_TO_DATE
	STRING	X	The value is converted into a character string. • LAD/FBD: The first characters of the string are padded with spaces. The number of spaces depends on the length of the numerical value. Positive numeric values are output without a sign. • STL/SCL: No spaces are inserted. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	INT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		INT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	INT_TO_CHAR
	WCHAR[1)]	X		INT_TO_WCHAR
	BCD16	X	The value to be converted has type INT and is accepted as an integer with a value between -999 and +999. The result is available after conversion as a BCD-coded number of the type WORD. A real conversion takes place. If the value is outside the target	INT_TO_BCD16
	BCD16_WORD	X		INT_TO_BCD16_WORD

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
			area, a synchronous error is not triggered, but rather only the status bit OV is set.	
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width including sign, and then the bits are copied. The source type determines the interpretation.				

See also

Implicit conversion of INT (Page 2344)

Overview of data type conversion (Page 2330)

INT (16-bit integers) (Page 2250)

Explicit conversion of UINT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the UINT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
UINT	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	UINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If bits are lost in the process, the enable output ENO is set to "0".	UINT_TO_BYTE
	WORD[1)]	X		UINT_TO_WORD
	DWORD[1)]	X		UINT_TO_DWORD
	LWORD[1)]	X		UINT_TO_LWORD
	SINT	X		UINT_TO_SINT
	USINT	X		UINT_TO_USINT
	INT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If the sign bit is changed during the conversion, the enable output ENO is set to "0".	UINT_TO_INT
	DINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	UINT_TO_DINT
	UDINT	X		UINT_TO_UDINT
	LINT	X		UINT_TO_LINT
	ULINT	X		UINT_TO_ULINT
	REAL	X	The value is converted to the destination data type format. (The value "1" is changed to the value "1.0", for example, with the instruction "Convert value" (CONVERT).	UINT_TO_REAL, NORM_X
	LREAL	X		UINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the destination data type and interpreted as milliseconds.	UINT_TO_TIME
	LTIME	X	The value is transferred to the destination data type and interpreted as nanoseconds.	UINT_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The result is returned in nanoseconds since 1970-1-1-0:0:0.0.	UINT_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
	TOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in milliseconds since 0:0; check for 24h limit)	UINT_TO_TOD
	LTOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in nanoseconds since 0:0; check for 24h limit)	UINT_TO_LTOD
	DATE	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in days since 1990-1-1; check for negative value)	UINT_TO_DATE, T_CONV
	STRING	X	The value is converted into a character string. If the permitted length of the character string is violated, the enable output ENO is set to "0".	UINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		UINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is transferred unchanged to the destination data type. The enable output ENO is set to "0" in the event of overflow.	UINT_TO_CHAR
	WCHAR[1)]	X		UINT_TO_WCHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width (the non-existing sign is replaced with zeros), and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of UINT (Page 2345)

Overview of data type conversion (Page 2330)

UINT (16-bit integers) (Page 2251)

Explicit conversion of DINT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DINT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
DINT	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	DINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	DINT_TO_BYTE
	WORD[1)]	X		DINT_TO_WORD
	DWORD[1)]	X		DINT_TO_DWORD
	LWORD[1)]	X		DINT_TO_LWORD
	SINT	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	DINT_TO_SINT
	USINT	X		DINT_TO_USINT
	INT	X		DINT_TO_INT
	UINT	X		DINT_TO_UINT
	UDINT	X		DINT_TO_UDINT
	LINT	X		DINT_TO_LINT
	ULINT	X		DINT_TO_ULINT
	REAL	X	The value is converted to the destination data type format. (The value "-1" is changed to the value "-1.0", for example, with the instruction "Convert value" (CONVERT).	DINT_TO_REAL, NORM_X
	LREAL	X		DINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the destination data type and interpreted as milliseconds.	DINT_TO_TIME, T_CONV
	LTIME	X	The value is transferred to the destination data type and interpreted as nanoseconds.	DINT_TO_LTIME, T_CONV
	S5TIME	-	No explicit conversion	-
	LDT	X	The result is returned in nanoseconds since 1970-1-1-0:0:0.0.	DINT_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-
	TOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1"	DINT_TO_TOD

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
			(16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in milliseconds since 0:0)	
	LTOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in nanoseconds since 0:0)	DINT_TO_LTOD
	DATE	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in days since 1990-1-1)	DINT_TO_DATE
	STRING	X	The value is converted into a character string. • LAD/FBD: The first characters of the string are padded with spaces. The number of spaces depends on the length of the numerical value. Positive numeric values are output without a sign. • STL/SCL: No spaces are inserted. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	DINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		DINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	DINT_TO_CHAR
	WCHAR[1)]	X		DINT_TO_WCHAR
	BCD32	X	The value to be converted has type DINT and is accepted as an integer with a value between –999999 and +9999999. The result is available	DINT_TO_BCD32
	BCD32_DWORD	X		DINT_TO_BCD32_DWORD

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
			after conversion as a BCD-coded number of the type DWORD. The enable output is set to "0" in the event of overflow. A real conversion takes place. If the value is outside the target area, a synchronous error is not triggered, but rather only the status bit OV is set.	
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width including sign, and then the bits are copied. The source type determines the interpretation.				

See also

Implicit conversion of DINT (Page 2347)

Overview of data type conversion (Page 2330)

DINT (32-bit integers) (Page 2252)

Explicit conversion of UDINT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the UDINT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
UDINT	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	UDINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If bits are lost in the process, the enable output ENO is set to "0".	UDINT_TO_BYTE
	WORD[1)]	X		UDINT_TO_WORD
	DWORD[1)]	X		UDINT_TO_DWORD
	LWORD[1)]	X		UDINT_TO_LWORD
	SINT	X		UDINT_TO_SINT
	USINT	X		UDINT_TO_USINT
	INT	X		UDINT_TO_INT
	UINT	X		UDINT_TO_UINT
	DINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If the sign bit is changed during the conversion, the enable output ENO is set to "0".	UDINT_TO_DINT
	LINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	UDINT_TO_LINT
	ULINT	X		UDINT_TO_ULINT
	REAL	X	The value is converted to the destination data type format. (The value "1" is changed to the value "1.0", for example, with the instruction "Convert value" (CONVERT).	UDINT_TO_REAL, NORM_X
	LREAL	X		UDINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the destination data type and interpreted as milliseconds.	UDINT_TO_TIME
	LTIME	X	The value is transferred to the destination data type and interpreted as nanoseconds.	UDINT_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The result is returned in nanoseconds since 1970-1-1-0:0:0.0.	UDINT_TO_LDT
	DT	-	No explicit conversion	-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
	DTL	-		-
	TOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in milliseconds since 0:0; check for 24h limit)	UDINT_TO_TOD, T_CONV
	LTOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in nanoseconds since 0:0; check for 24h limit)	UDNT_TO_LTOD, T_CONV
	DATE	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in days since 1990-1-1; check for negative value)	UDINT_TO_DATE
	STRING	X	The value is converted into a character string. If the permitted length of the character string is violated, the enable output ENO is set to "0".	UDINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		UDINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is transferred unchanged to the destination data type. The enable output ENO is set to "0" in the event of overflow.	UDINT_TO_CHAR
	WCHAR[1)]	X		UDINT_TO_WCHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width (the non-existing sign is replaced with zeros), and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of UDINT (Page 2349)

Overview of data type conversion (Page 2330)

UDINT (32-bit integers) (Page 2253)

Explicit conversion of LINT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the LINT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
LINT	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	LINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	LINT_TO_BYTE
	WORD[1)]	X		LINT_TO_WORD
	DWORD[1)]	X		LINT_TO_DWORD
	LWORD[1)]	X		LINT_TO_LWORD
	SINT	X		LINT_TO_SINT
	USINT	X		LINT_TO_USINT
	INT	X		LINT_TO_INT
	UINT	X		LINT_TO_UINT
	DINT	X		LINT_TO_DINT
	UDINT	X		LINT_TO_UDINT
	ULINT	X		LINT_TO_ULINT
	REAL	X	The value is converted to the destination data type format. (The value "-1" is changed to the value "-1.0", for example, with the instruction "Convert value" (CONVERT).	LINT_TO_REAL, NORM_X
	LREAL	X		LINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the destination data type and interpreted as milliseconds.	LINT_TO_TIME, T_CONV
	LTIME	X	The value is transferred to the destination data type and interpreted as nanoseconds.	LINT_TO_LTIME, T_CONV
	S5TIME	-	No explicit conversion	-
	LDT	X	The result is returned in nanoseconds since 1970-1-1-0:0:0.0.	LINT_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-
	TOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative	LINT_TO_TOD

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
			value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in milliseconds since 0:0)	
	LTOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in nanoseconds since 0:0)	LINT_TO_LTOD
	DATE	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0". (interpretation in days since 1990-1-1)	LINT_TO_DATE
	STRING	X	The value is converted into a character string. • LAD/FBD: The first characters of the string are padded with spaces. The number of spaces depends on the length of the numerical value. Positive numeric values are output without a sign. • STL/SCL: No spaces are inserted. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	LINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		LINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the destination data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted into an unsigned destination data type, the enable output ENO is set to "0".	LINT_TO_CHAR
	WCHAR[1)]	X		LINT_TO_WCHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width including sign, and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of LINT (Page 2351)

Overview of data type conversion (Page 2330)

LINT (64-bit integers) (Page 2254)

Explicit conversion of ULINT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the ULINT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
ULINT	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	ULINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	ULINT_TO_BYTE
	WORD[1)]	X		ULINT_TO_WORD
	DWORD[1)]	X		ULINT_TO_DWORD
	LWORD[1)]	X		ULINT_TO_LWORD
	SINT	X		ULINT_TO_SINT
	USINT	X		ULINT_TO_USINT
	INT	X		ULINT_TO_INT
	UINT	X		ULINT_TO_UINT
	DINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If the sign bit is overwritten during the conversion, the enable output ENO is set to "0".	ULINT_TO_DINT
	UDINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	ULINT_TO_UDINT
	LINT	X		ULINT_TO_LINT
	REAL	X	The value is converted to the destination data type format. (The value "1" is changed to the value "1.0", for example, with the instruction "Convert value" (CONVERT).	ULINT_TO_REAL, NORM_X
	LREAL	X		ULINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the destination data type and interpreted as milliseconds.	ULINT_TO_TIME
	LTIME	X	The value is transferred to the destination data type and interpreted as nanoseconds.	ULINT_TO_LTIME
	S5TIME	-	No explicit conversion	-
	LDT	X	The result is returned in nanoseconds since 1970-1-1-0:0:0.0.	ULINT_TO_LDT
	DT	-	No explicit conversion	-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
	DTL	-		-
	TOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in milliseconds since 0:0)	ULINT_TO_TOD, T_CONV
	LTOD	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in nanoseconds since 0:0)	ULINT_TO_LTOD, T_CONV
	DATE	X	The bit pattern of the source value is converted and transferred to the destination data type. (interpretation in days since 1990-1-1)	ULINT_TO_DATE
	STRING	X	The value is converted into a character string. If the permitted length of the character string is violated, the enable output ENO is set to "0".	ULINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		ULINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the destination data type.	ULINT_TO_CHAR
	WCHAR[1)]	X		ULINT_TO_WCHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width (the non-existing sign is replaced with zeros), and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of ULINT (Page 2353)

Overview of data type conversion (Page 2330)

ULINT (64-bit integers) (Page 2255)

Floating-point numbers (S7-1500)

Explicit conversion of REAL (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the REAL data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
REAL	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	REAL_TO_DWORD
	LWORD	-	No explicit conversion	-
	SINT	X	The value is converted into the destination data type. The result of conversion depends on the instruction employed. The enable output ENO is set to "0" if the admissible range of values of the destination data type is exceeded during conversion or the value to be converted has an invalid floating-point number.	REAL_TO_SINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	USINT	X		REAL_TO_USINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	INT	X		REAL_TO_INT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	UINT	X		REAL_TO_UINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	DINT	X		REAL_TO_DINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	UDINT	X		REAL_TO_UDINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	LINT	X		REAL_TO_LINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	ULINT	X		REAL_TO_ULINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	LREAL	X	The value is converted into the destination data type. The result of the conversion depends on the instruction used, e.g. TRUNC(2.5) = 2.0; CEIL(2.5) = 3.0	REAL_TO_LREAL, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	TIME	-	No explicit conversion	-
	LTIME	-		-
	S5TIME	-		-
	LDT	-		-
	DT	-		-
	DTL	-		-
	TOD	-		-
	LTOD	-		-
	DATE	-		-
	STRING	X	The value is converted into a character string. The enable output ENO is set to "0" if the character string length is exceeded or the value to be converted has an invalid floating-point number. The min. length of the character string is 14 characters.	REAL_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		REAL_TO_WSTRING
	CHAR	-	No explicit conversion	-
	WCHAR	-		

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible				

See also

Implicit conversion of REAL (Page 2355)

Overview of data type conversion (Page 2330)

REAL (Page 2257)

Explicit conversion of LREAL (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the LREAL data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
LREAL	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	-		-
	LWORD	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LREAL_TO_LWORD
	SINT	X	The value is converted into the destination data type. The result of conversion depends on the instruction employed. The enable output ENO is set to "0" if the admissible range of values of the destination data type is exceeded during conversion or the value to be converted has an invalid floating-point number.	LREAL_TO_SINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	USINT	X		LREAL_TO_USINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	INT	X		LREAL_TO_INT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	UINT	X		LREAL_TO_UINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	DINT	X		LREAL_TO_DINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	UDINT	X		LREAL_TO_UDINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	LINT	X		LREAL_TO_LINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	ULINT	X		LREAL_TO_ULINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	REAL	X	The value is converted into the destination data type. If the permitted value range of the destination data type is violated during the conversion or if the value to be converted is an invalid floating-point number, the enable output ENO is set to "0". A loss in accuracy is tolerated.	LREAL_TO_LREAL, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	TIME	-	No explicit conversion	-
	LTIME	-		-
	S5TIME	-		-
	LDT	-		-
	DT	-		-
	DTL	-		-
	TOD	-		-
	LTOD	-		-
	DATE	-		-
	STRING	X	The value is converted into a character string. The enable output ENO is set to "0" if the character string length is exceeded or the value to be converted has an invalid floating-point number. The min.	LREAL_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		LREAL_TO_WSTRING

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
			length of the character string is 21 characters.	
	CHAR	-	No explicit conversion	-
	WCHAR	-		-
x: Conversion possible - : Conversion not possible				

See also

Implicit conversion of LREAL (Page 2357)

Overview of data type conversion (Page 2330)

LREAL (Page 2258)

Timers (S7-1500)

Explicit conversion of S5TIME (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the S5TIME data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
S5TIME	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	S5TIME_TO_WORD
	DWORD	-	No explicit conversion	-
	LWORD	-		-
	SINT	-		-
	USINT	-		-
	INT	-		-
	UINT	-		-
	DINT	-		-
	UDINT	-		-
	LINT	-		-
	ULINT	-		-
	REAL	-		-
	LREAL	-		-
	TIME	X	A conversion of the source value into the destination data type takes place here (for example, s5t#10ms becomes T#10ms)	S5TIME_TO_TIME
	LTIME	X	A conversion of the source value into the destination data type takes place here (for example, s5t#10ms becomes LTIME#10ms)	S5TIME_TO_LTIME
	LDT	-	No explicit conversion	-
	DT	-		-
	DTL	-		-
	TOD	-		-
	LTOD	-		-
	DATE	-		-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.				

See also

Implicit conversion of S5TIME (Page 2358)

Overview of data type conversion (Page 2330)

S5TIME (duration) (Page 2261)

Explicit conversion of TIME (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the TIME data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
TIME	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	TIME_TO_BYTE
	WORD[1)]	X		TIME_TO_WORD
	DWORD[1)]	X		TIME_TO_DWORD
	LWORD[1)]	X		TIME_TO_LWORD
	SINT	X	The bit pattern of the source value is transferred unchanged right-justified and interpreted as milliseconds to the destination data type.	TIME_TO_SINT
	USINT	X		TIME_TO_USINT
	INT	X		TIME_TO_INT
	UINT	X		TIME_TO_UINT
	DINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. The result of conversion shows the duration in milliseconds.	TIME_TO_DINT, T_CONV
	UDINT	X	The bit pattern of the source value is transferred unchanged right-justified and interpreted as milliseconds to the destination data type. A change in sign has the result that the enable output ENO is set to "0".	TIME_TO_UDINT
	LINT	X		TIME_TO_LINT
	ULINT	X		TIME_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	S5TIME	X	A conversion of the source value into the destination data type takes place here (for example, T#10ms becomes s5t#10ms)	TIME_TO_TIME
	LTIME	X	A conversion of the source value into the destination data type takes place here (for example, T#10ms becomes LTIME#10ms)	TIME_TO_LTIME
	LDT	-	No explicit conversion	-
	DT	-		-
	DTL	-		-
	TOD	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If the source value is outside the value range of TOD, the destination data type remains unchanged.	TIME_TO_TOD
	LTOD	-	No explicit conversion	-
	DATE	-		-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.				

See also

Implicit conversion of TIME (Page 2360)

Overview of data type conversion (Page 2330)

TIME (IEC time) (Page 2262)

Explicit conversion of LTIME (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the LTIME data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
LTIME	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LTIME_TO_BYTE
	WORD[1)]	X		LTIME_TO_WORD
	DWORD[1)]	X		LTIME_TO_DWORD
	LWORD[1)]	X		LTIME_TO_LWORD
	SINT	X	The bit pattern of the source value is transferred unchanged right-justified and interpreted as nanoseconds to the destination data type. A change in sign has the result that the enable output ENO is set to "0".	LTIME_TO_SINT
	USINT	X		LTIME_TO_USINT
	INT	X		LTIME_TO_INT
	UINT	X		LTIME_TO_UINT
	DINT	X		LTIME_TO_DINT
	UDINT	X		LTIME_TO_UDINT
	LINT	X		LTIME_TO_LINT, T_CONV
	ULINT	X		LTIME_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	S5TIME	X	A conversion of the source value into the destination data type takes place here (for example, LTIME#10ms becomes s5t#10ms)	LTIME_TO_S5TIME
	TIME	X	A conversion of the source value into the destination data type takes place here (for example, LTIMET#10ms becomes T#10ms)	LTIME_TO_TIME
	LDT	X	A conversion of the source value into the destination data type takes place here (for example, LTIMET#10ms becomes LDT#1970-1-1-0:0:0.010000000)	LTIME_TO_LDT
	DT	-	No explicit conversion	-
	DTL	-		-
	TOD	-		-
	LTOD	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type. If the source value is outside the value range of LTOD, the destination data type remains unchanged.	LTIME_TO_LTOD
	DATE	-	No explicit conversion	-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of LTIME (Page 2362)

Overview of data type conversion (Page 2330)

LTIME (IEC time) (Page 2263)

Date and time-of-day (S7-1500)

Explicit conversion of DT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
DT	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	-		-
	LWORD	-		-
	SINT	-		-
	USINT	-		-
	INT	-		-
	UINT	-		-
	DINT	-		-
	UDINT	-		-
	LINT	-		-
	ULINT	-		-
	REAL	-		-
	LREAL	-		-
	S5TIME	-		-
	TIME	-		-
	LTIME	-		-
	LDT	X	A conversion of the source value into the destination data type takes place here (for example, DT#1990-1-2-0:0:1.0 becomes LDT#1990-1-2-0:0:1.0)	DT_TO_LDT
	DTL	X	A conversion of the source value into the destination data type takes place here (for example, DT#1990-1-2-0:0:1.0 becomes DTL#1990-1-2-0:0:1.0)	DT_TO_DTL
	TOD	X	A conversion of the source value into the destination data type takes place here (for example, DT#1990-1-2-0:0:1.0 becomes TOD#1990-1-2-0:0:1.0)	DT_TO_TOD
	LTOD	X	A conversion of the source value into the destination data type takes place here (for example, DT#1990-1-2-0:0:1.0 becomes LTOD#1990-1-2-0:0:1.0)	DT_TO_LTOD
	DATE	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	DT_TO_DATE
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible				

See also

Implicit conversion of DT (Page 2364)

Overview of data type conversion (Page 2330)

DATE_AND_TIME (date and time of day) (Page 2265)

Explicit conversion of LDT (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the LDT data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
LDT	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LDT_TO_BYTE
	WORD[1)]	X		LDT_TO_WORD
	DWORD[1)]	X		LDT_TO_DWORD
	LWORD[1)]	X		LDT_TO_LWORD
	SINT	X		LDT_TO_SINT
	USINT	X		LDT_TO_USINT
	INT	X		LDT_TO_INT
	UINT	X		LDT_TO_UINT
	DINT	X		LDT_TO_DINT
	UDINT	X		LDT_TO_UDINT
	LINT	X		LDT_TO_LINT
	ULINT	X		LDT_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	S5TIME	-		-
	TIME	-		-
	LTIME	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LDT_TO_LTIME
	DT	X	A conversion of the source value into the destination data type takes place here (for example, LDT#1990-1-2-0:0:1.0 becomes DT#1990-1-2-0:0:1.0)	LDT_TO_DT
	DTL	X	A conversion of the source value into the destination data type takes place here (for example, LDT#1990-1-2-0:0:1.0 becomes DTL#1990-1-2-0:0:1.0)	LDT_TO_DTL
	TOD	X	A conversion of the source value into the destination data type takes place here (for example, LDT#1990-1-2-0:0:1.0 becomes TOD#1990-1-2-0:0:1.0)	LDT_TO_TOD
	LTOD	X	A conversion of the source value into the destination data type takes place here (for example, LDT#1990-1-2-0:0:1.0 becomes LTOD#1990-1-2-0:0:1.0)	LDT_TO_LTOD
	DATE	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LDT_TO_DATE
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	-		-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
	WCHAR	-		-

x: Conversion possible

- : Conversion not possible

[1] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of LDT (Page 2366)

Overview of data type conversion (Page 2330)

LDT (DATE_AND_LTIME) (Page 2266)

Explicit conversion of DTL (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DTL data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
DTL	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	-		-
	LWORD	-		-
	SINT	-		-
	USINT	-		-
	INT	-		-
	UINT	-		-
	DINT	-		-
	UDINT	-		-
	LINT	-		-
	ULINT	-		-
	REAL	-		-
	LREAL	-		-
	S5TIME	-		-
	TIME	-		-
	LTIME	-		-
	LDT	X	A conversion of the source value into the destination data type takes place here (for example, DTL#1990-1-2-0:0:1.0 becomes LDT#1990-1-2-0:0:1.0)	DTL_TO_LDT
	DT	X	A conversion of the source value into the destination data type takes place here (for example, DTL#1990-1-2-0:0:1.0 becomes DT#1990-1-2-0:0:1.0)	DTL_TO_DT
	TOD	X	During the conversion, the time of day is extracted from the DTL format and written to the destination data type.	DTL_TO_TOD, T_CONV
	LTOD	X		DTL_TO_LTOD
	DATE	X	During the conversion, the date is extracted from the DTL format and written to the destination data type. The enable output ENO is set to "0" in the event of overflow.	DTL_TO_DATE, T_CONV
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-
x: Conversion possible - : Conversion not possible				

See also

Implicit conversion of DTL (Page 2368)

Overview of data type conversion (Page 2330)

DTL (Page 2267)

Explicit conversion of DATE (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DATE data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DATE	BOOL	-	No explicit conversion	-
	BYTE[1]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DATE_TO_BYTE
	WORD[1]	X		DATE_TO_WORD
	DWORD[1]	X		DATE_TO_DWORD
	LWORD[1]	X		DATE_TO_LWORD
	SINT	X	The number of days since 1/1/1990 is returned as result.	DATE_TO_SINT
	USINT	X		DATE_TO_USINT
	INT	X		DATE_TO_INT
	UINT	X		DATE_TO_UINT
	DINT	X		DATE_TO_DINT
	UDINT	X		DATE_TO_UDINT
	LINT	X		DATE_TO_LINT
	ULINT	X		DATE_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	S5TIME	-		-
	TIME	-		-
	LTIME	-		-
	DT	X	The conversion sets the date in the destination type.	DATE_TO_DT
	LDT	X		DATE_TO_LDT
	DTL [2]	X		DATE_TO_DTL
	TOD	-	No explicit conversion	-
	LTOD	-		-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

x: Conversion possible

- : Conversion not possible

[1] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.

[2] The data type DTL is initialized as follows: 1970-1-1-0:0:0. Conversion from DATE_TO_DTL only sets the corresponding part of the DTL. The rest of the DTL remains as it was initialized. This results in the following differences for the instructions "T_COMBINE" and "T_CONV":

- T_COMBINE(D#2015-24-12, LTOD#12:13:14) returns DTL#2015-24-12-12:13:14
- myDTL := T_CONV(D#2015-24-12); myDTL := T_CONV(LTOD#12:13:14) results in myDTL = DTL#1970-1-1-12:13:14

See also

Implicit conversion of DATE (Page 2370)

Overview of data type conversion (Page 2330)

DATE (Page 2264)

Explicit conversion of TOD (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the TOD data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
TOD	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	TOD_TO_BYTE
	WORD[1)]	X		TOD_TO_WORD
	DWORD[1)]	X		TOD_TO_DWORD
	LWORD[1)]	X		TOD_TO_LWORD
	SINT	X	The number of milliseconds since midnight is returned as result.	TOD_TO_SINT
	USINT	X		TOD_TO_USINT
	INT	X		TOD_TO_INT
	UINT	X		TOD_TO_UINT
	DINT	X		TOD_TO_DINT
	UDINT	X	The result of the conversion corresponds to the number of milliseconds since the start of day (0:00).	TOD_TO_UDINT, T_CONV
	LINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	TOD_TO_LINT
	ULINT	X		TOD_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	S5TIME	-		-
	TIME	X	The duration since midnight is returned as result.	TOD_TO_TIME
	LTIME	-	No explicit conversion	-
	DT	X	The conversion sets the time in the destination data type.	TOD_TO_DT
	LDT	X		TOD_TO_LDT
	DTL	X		TOD_TO_DTL
	DATE	-	No explicit conversion	-
	LTOD	X	A conversion of the source value into the destination data type takes place here (for example, TOD#1:0:0.0 becomes LTOD#1:0:0.0)	TOD_TO_LTOD
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.

See also

Implicit conversion of TOD (Page 2372)

Overview of data type conversion (Page 2330)

TIME_OF_DAY (TOD) (Page 2264)

Explicit conversion of LTOD (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the LTOD data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
LTOD	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LTOD_TO_BYTE
	WORD[1)]	X		LTOD_TO_WORD
	DWORD[1)]	X		LTOD_TO_DWORD
	LWORD[1)]	X		LTOD_TO_LWORD
	SINT	X	The number of nanoseconds since midnight is returned as result.	LTOD_TO_SINT
	USINT	X		LTOD_TO_USINT
	INT	X		LTOD_TO_INT
	UINT	X		LTOD_TO_UINT
	DINT	X		LTOD_TO_DINT
	UDINT	X		LTOD_TO_UDINT
	LINT	X		LTOD_TO_LINT
	ULINT	X		LTOD_TO_ULINT, T_CONV
	REAL	-	No explicit conversion	-
	LREAL	-		-
	S5TIME	-		-
	TIME	-		-
	LTIME	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	LTOD_TO_LTIME
	DT	X	A conversion of the source value into the destination data type takes place here (for example, LTOD#1:0:0.0 becomes DT#1:0:0.0)	LTOD_TO_DT
	LDT	X	A conversion of the source value into the destination data type takes place here (for example, LTOD#1:0:0.0 becomes LDT#1:0:0.0)	LTOD_TO_LDT
	DTL	X	A conversion of the source value into the destination data type takes place here (for example, LTOD#1:0:0.0 becomes DTL#1:0:0.0)	LTOD_TO_DTL
	DATE	-	No explicit conversion	-
	TOD	X	A conversion of the source value into the destination data type takes place here (for example, LTOD#1:0:0.0 becomes TOD#1:0:0.0)	LTOD_TO_TOD
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD, LWORD) and the data types CHAR and WCHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.				

See also

Implicit conversion of LTOD (Page 2374)

Overview of data type conversion (Page 2330)

LTOD (LTIME_OF_DAY) (Page 2265)

String (S7-1500)

Explicit conversion of CHAR (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the CHAR data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
CHAR	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	CHAR_TO_BYTE
	WORD[1)]	X		CHAR_TO_WORD
	DWORD[1)]	X		CHAR_TO_DWORD
	LWORD[1)]	X		CHAR_TO_LWORD
	SINT	X		CHAR_TO_SINT
	USINT	X		CHAR_TO_USINT
	INT	X		CHAR_TO_INT
	UINT	X		CHAR_TO_UINT
	DINT	X		CHAR_TO_DINT
	UDINT	X		CHAR_TO_UDINT
	LINT	X		CHAR_TO_LINT
	ULINT	X		CHAR_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	S5TIME	-		-
	TIME	-		-
	LTIME	-		-
	DT	-		-
	LDT	-		-
	DTL	-		-
	TOD	-		-
	LTOD	-		-
	DATE	-		-
	STRING	X	The value is converted into the first character in the character string (STRING). The length "1" is set after conversion if the character string length is not defined. If the character string length is defined, it remains unchanged following conversion.	CHAR_TO_STRING, S_CONV
	WSTRING	-	No explicit conversion	-
	WCHAR	X		CHAR_TO_WCHAR

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD, LWORD) and the data type WCHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.				

See also

Implicit conversion of CHAR (Page 2376)

Overview of data type conversion (Page 2330)

CHAR (Page 2269)

Explicit conversion of WCHAR (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the WCHAR data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
WCHAR	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	WCHAR_TO_BYTE
	WORD[1)]	X		WCHAR_TO_WORD
	DWORD[1)]	X		WCHAR_TO_DWORD
	LWORD[1)]	X		WCHAR_TO_LWORD
	SINT	X		WCHAR_TO_SINT
	USINT	X		WCHAR_TO_USINT
	INT	X		WCHAR_TO_INT
	UINT	X		WCHAR_TO_UINT
	DINT	X		WCHAR_TO_DINT
	UDINT	X		WCHAR_TO_UDINT
	LINT	X		WCHAR_TO_LINT
	ULINT	X		WCHAR_TO_ULINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	S5TIME	-		-
	TIME	-		-
	LTIME	-		-
	DT	-		-
	LDT	-		-
	DTL	-		-
	TOD	-		-
	LTOD	-		-
	DATE	-		-
	STRING	-		-
	WSTRING	X	The value is converted into the first character in the character string (WSTRING). The length "1" is set after conversion if the character string length is not defined. If the character string length is defined, it remains unchanged following conversion.	WCHAR_TO_WSTRING
	CHAR	X		WCHAR_TO_CHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data type CHAR are initially extended to the necessary width, and then the bits are copied. The source type determines the interpretation.

See also

Overview of data type conversion (Page 2330)

WCHAR (Page 2269)

Implicit conversion of WCHAR (Page 2377)

Explicit conversion of STRING (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the STRING data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
STRING	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	-		-
	LWORD	-		-
	SINT	X	Conversion begins with the first character in the character string (STRING) and stops at the end of the string or at the first inadmissible character. The following characters are admissible for conversion: • Digit • Sign • Dot The first character in the character string may be a sign (+, -) or a digit. Leading spaces are ignored. The dot is used as separation for the conversion of floating-point numbers. The exponential notation "e" or "E" is not permitted. The comma as thousand separator to the left of the decimal point is permitted but is ignored. If the layout of the character string is invalid for the conversion or an overflow occurs, then the enable output ENO will be set to "0".	STRING_TO_SINT, S_CONV, STRG_VAL
	USINT	X		STRING_TO_USINT, S_CONV, STRG_VAL
	INT	X		STRING_TO_INT, S_CONV, STRG_VAL
	UINT	X		STRING_TO_UINT, S_CONV, STRG_VAL
	DINT	X		STRING_TO_DINT, S_CONV, STRG_VAL
	UDINT	X		STRING_TO_UDINT, S_CONV, STRG_VAL
	LINT	X		STRING_TO_LINT, S_CONV, STRG_VAL
	ULINT	X		STRING_TO_ULINT, S_CONV, STRG_VAL
	REAL	X		STRING_TO_REAL, S_CONV, STRG_VAL
	LREAL	X		STRING_TO_LREAL, S_CONV, STRG_VAL
	S5TIME	-	No explicit conversion	-
	TIME	-		-
	LTIME	-		-
	DT	-		-
	LDT	-		-
	DTL	-		-
	TOD	-		-
	LTOD	-		-
	DATE	-		-
	CHAR	X	The first character in the character string (STRING) is transferred to the destination data type. The value "0" is written to the destination data type if the character string is empty.	STRING_TO_CHAR, S_CONV
	WCHAR	-	No explicit conversion	-
	WSTRING	X		STRING_TO_WSTRING
x: Conversion possible - : Conversion not possible				

See also

Implicit conversion of STRING (Page 2379)
Overview of data type conversion (Page 2330)
STRING (Page 2270)

Explicit conversion of WSTRING (S7-1500)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the WSTRING data type:

Source	Destination	Conversion	Explanation	Mnemonics of the instruction
WSTRING	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	-		-
	LWORD	-		-
	SINT	X	Conversion begins with the first character in the character string (WSTRING) and stops at the end of the string or at the first inadmissible character. The following characters are admissible for conversion: • Digit • Sign • Dot The first character in the character string may be a sign (+, -) or a digit. Leading spaces are ignored. The dot is used as separation for the conversion of floating-point numbers. The exponential notation "e" or "E" is not permitted. The comma as thousand separator to the left of the decimal point is permitted but is ignored. If the layout of the character string is invalid for the conversion or an overflow occurs, then the enable output ENO will be set to "0".	WSTRING_TO_SINT, S_CONV, STRG_VAL
	USINT	X		WSTRING_TO_USINT, S_CONV, STRG_VAL
	INT	X		WSTRING_TO_INT, S_CONV, STRG_VAL
	UINT	X		WSTRING_TO_UINT, S_CONV, STRG_VAL
	DINT	X		WSTRING_TO_DINT, S_CONV, STRG_VAL
	UDINT	X		WSTRING_TO_UDINT, S_CONV, STRG_VAL
	LINT	X		WSTRING_TO_LINT, S_CONV, STRG_VAL
	ULINT	X		WSTRING_TO_ULINT, S_CONV, STRG_VAL
	REAL	X		WSTRING_TO_REAL, S_CONV, STRG_VAL
	LREAL	X		WSTRING_TO_LREAL, S_CONV, STRG_VAL
	S5TIME	-	No explicit conversion	-
	TIME	-		-
	LTIME	-		-
	DT	-		-
	LDT	-		-
	DTL	-		-
	TOD	-		-
	LTOD	-		-
	DATE	-		-
	CHAR	-		-
	WCHAR	X	The first character in the character string (WSTRING) is transferred to the destination data type. The value "0" is written to the destination data type if the character string is empty.	WSTRING_TO_WCHAR
	STRING	X		WSTRING_TO_STRING
x: Conversion possible - : Conversion not possible				

See also

Overview of data type conversion (Page 2330)

WSTRING (Page 2273)

Implicit conversion of WSTRING (Page 2380)

11.3.16 Data type conversion for S7-1200 (S7-1200)

11.3.16.1 Overview of data type conversion (S7-1200)

Introduction

If you link several operands in an instruction, you must make sure that the data types are compatible. This applies also for assignments or for supplying block parameters. If the operands are not the same data type, a conversion has to be carried out.

There are two options for the conversion:

- Implicit conversion
  The conversion take place automatically when the instruction is executed.
- Explicit conversion
  You use an explicit conversion instruction before the actual instruction is executed.

Note

The data type conversion options described always refer to the latest version of the CPU (V. 4) Conversions marked as possible may not be available in CPU versions 1 - 3.

Note

Converting bit strings in SCL

All bit strings (BYTE, WORD, and DWORD) are handled like the corresponding unsigned integers (USINT, UINT, and UDINT) in expressions. Therefore, implicit conversion from DWORD to REAL is carried out like a conversion from UDINT to REAL, for example.

Implicit conversion

An implicit conversion is executed automatically if the data types of the operands are compatible. This compatibility test can be carried out according to criteria that are more or less strict:

- With IEC check (default)
  If IEC check is set, the following rules are applied:
  - Implicit conversion of BOOL to other data types is not possible.
  - Only the REAL, BYTE, WORD, DINT, INT, SINT, UDINT, UINT, USINT, TIME, DT, STRING, CHAR and WCHAR data types can be converted implicitly.
  - The bit length of the source data type must not exceed the bit length of the target data type. An operand of data type WORD, for example, cannot be declared at a parameter at which data type BYTE is expected.
- Without IEC check
  If IEC check is not set, the following rules are applied:
  - Implicit conversion of BOOL to other data types is not possible.
  - Only the REAL, LREAL, BYTE, WORD, DWORD, SINT, INT, DINT, USINT, UINT, UDINT, TIME, DTL, TOD, DATE, STRING, CHAR and WCHAR data types can be converted implicitly.
  - The bit length of the source data type must not exceed the bit length of the target data type. An operand of data type DWORD, for example, cannot be declared at a parameter at which data type WORD is expected.
  - The bit length of an operand entered in-out parameters InOut) must be the same as the programmed bit length for the parameter in question.

Note

Implicit conversion without IEC check

The programming editor uses a gray rectangle to mark operands that are implicitly converted. The dark gray rectangle signals that an implicit conversion is possible without any accuracy loss, for example, if you convert the data type SINT to INT. A light gray rectangle signals that implicit conversion is possible, but errors could occur during runtime. If, for example, you are converting the data type DINT to INT and an overflow occurs, the enable output ENO is set to "0".

For more information about the setting of the IEC check and the implicit conversion, refer to "See also".

Explicit conversion

If the operands are not compatible and an implicit conversion is therefore not possible, you can use an explicit conversion instruction. You can find the conversion instructions in the "Instructions" task card.

A possible overflow is displayed at the ENO enable output. An overflow is created, for example, if the value of the source data type is greater than the value of the target data type.

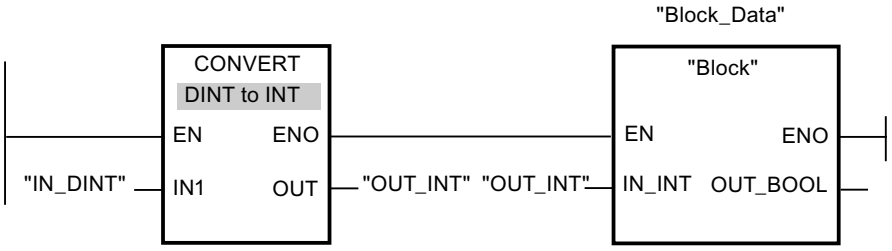
Note

Shifting of bit patterns

If the explicit conversion involves shifting a bit pattern, the enable output ENO is not set.

For more information about explicit conversion, refer to "See also".

The following figure shows an example in which an explicit data type conversion must be carried out:



The "Block" function block expects a tag of the INT data type at the "IN_INT" input parameter. Therefore, the value of the "IN_DINT" tag must first be converted from DINT to INT. If the value of the "IN_DINT" tag is within the permitted value range of the INT data type, the conversion takes place. Otherwise, an overflow is signaled. A conversion still takes place even in case of an overflow, but the values are cut off and the enable output ENO is set to "0".

See also

Setting and canceling the IEC check (Page 2467)

Implicit conversions (Page 2467)

Explicit conversions (Page 2487)

11.3.16.2 Implicit conversions (S7-1200)

Setting and canceling the IEC check (S7-1200)

The data types of the operands used are check for compatibility. This compatibility test can be carried out according to criteria that are more or less strict. If "IEC check" is activated, stricter criteria are applied.

You can set the IEC check centrally for all new blocks of the project or for individual blocks.

Setting IEC check for new blocks

To set the IEC check for all new blocks in the project, proceed as follows:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "PLC programming > General" group in the area navigation.
3. Select or clear the "IEC check" check box in the "Default settings for new blocks" group.
   The IEC check is enabled or disabled for all new blocks in the program.

Setting IEC check for a block

To set the IEC check for a block, proceed as follows:

1. Open the block.
2. Open the "Properties" tab in the Inspector window.
3. Select the "Attributes" group in the area navigation.
4. Select or clear the "IEC check" check box.
   The IEC check is enabled or disabled for this block. The setting is stored together with the project.

Binary numbers (S7-1200)

Implicit conversion of BOOL (S7-1200)

Options for implicit conversion

The implicit conversion of the BOOL data type is not possible.

See also

BOOL (bit) (Page 2243)

Bit strings (S7-1200)

Implicit conversion of BYTE (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the BYTE data type:

Source	Target	With IEC check	Without IEC check	Explanation
BYTE	BOOL	-	-	No implicit conversion
	WORD	x	x	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	DWORD	x	x	
	SINT	-	x	
	USINT	-	x	
	INT	-	x	
	UINT	-	x	
	DINT	-	x	
	UDINT	-	x	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	DTL	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	x	The bit pattern of the source value is transferred unchanged to the target data type.
	WCHAR	-	x	
x: Conversion possible -: Conversion not possible				

See also

BYTE (Page 2244)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of BYTE (Page 2488)

Implicit conversion of WORD (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the WORD data type:

Source	Target	With IEC check	Without IEC check	Description
WORD	BOOL	-	-	No implicit conversion
	BYTE	-	X	The least significant byte is transferred to the target data type, while the most significant byte is ignored.
	DWORD	X	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	SINT	-	X	The least significant byte is transferred to the target data type, while the most significant byte is ignored.
	USINT	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	DTL	-	-	
	TOD	-	-	
	DATE	-	X	The bit pattern of the source value is transferred unchanged to the target data type.
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged to the target data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

WORD (Page 2245)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of WORD (Page 2490)

Implicit conversion of DWORD (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the DWORD data type:

Source	Target	With IEC check	Without IEC check	Explanation
DWORD	BOOL	-	-	No implicit conversion
	BYTE	-	X	The right bytes are transferred to the target data type; the left bytes are ignored.
	WORD	-	X	
	SINT	-	X	
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	UDINT	-	X	
	REAL	-	X	The value is converted to the format of the target data type. (The value "-1", for example, is converted to the value "-1.0".)
	LREAL	-	-	No implicit conversion
	TIME	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	DTL	-	-	No implicit conversion
	TOD	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	DATE	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

DWORD (Page 2245)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of DWORD (Page 2493)

Integers (S7-1200)

Implicit conversion of SINT (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the SINT data type:

Source	Target	With IEC check	Without IEC check	Explanation
SINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. The remaining bits are filled with "0".
	WORD	-	X	
	DWORD	-	X	
	USINT	-	X	The bit pattern of the source value is converted and transferred to the target data type. (for example, value transfer from SINT #-1 -> INT #-1, not filled with "0".)
	INT	X	X	
	UINT	-	X	
	DINT	X	X	
	UDINT	-	X	
	REAL	X	X	The value is converted to the format of the target data type. (The value "-1", for example, is converted to the value "-1.0".)
	LREAL	X	X	
	TIME	-	-	No implicit conversion
	DTL	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

SINT (8-bit integers) (Page 2248)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of SINT (Page 2497)

Implicit conversion of USINT (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the USINT data type:

Source	Target	With IEC check	Without IEC check	Explanation
USINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. The remaining bits are filled with "0".
	WORD	-	X	
	DWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the target data type. (for example, value conversion from USINT #10 -> DINT #10 or USINT #128 -> SINT #-128)
	INT	X	X	
	UINT	X	X	
	DINT	X	X	
	UDINT	X	X	
	REAL	X	X	The value is converted to the format of the target data type. (The value "1", for example, is converted to the value "1.0".)
	LREAL	X	X	
	TIME	-	-	No implicit conversion
	DTL	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

USINT (8-bit integers) (Page 2249)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of USINT (Page 2500)

Implicit conversion of INT (S7-1200)

Options for implicit conversion

The following table shows the options for the implicit conversion of the INT data type:

Source	Target	With IEC check	Without IEC check	Explanation
INT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WORD	-	X	
	DWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the target data type. (for example, value conversion from INT #-1 -> SINT #-1, or INT #-32 767 -> UINT #32 769)
	USINT	-	X	
	UINT	-	X	
	DINT	X	X	
	UDINT	-	X	
	REAL	X	X	The value is converted to the format of the target data type. (The value "-1", for example, is converted to the value "-1.0".)
	LREAL	X	X	
	TIME	-	-	No implicit conversion
	DTL	-	-	
	TOD	-	-	
	DATE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

INT (16-bit integers) (Page 2250)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of INT (Page 2503)

Implicit conversion of UINT (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the UINT data type:

Source	Target	With IEC check	Without IEC check	Explanation
UINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WORD	-	X	
	DWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the target data type. (for example, value conversion from UINT #100 -> DINT #100 or UINT #60 000 -> INT #-5536)
	USINT	-	X	
	INT	-	X	
	DINT	X	X	
	UDINT	X	X	
	REAL	X	X	The value is converted to the format of the target data type. (The value "1", for example, is converted to the value "1.0".)
	LREAL	X	X	
	TIME	-	-	No implicit conversion
	DTL	-	-	
	TOD	-	-	
	DATE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	STRING	-	-	No implicit conversion
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

UINT (16-bit integers) (Page 2251)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of UINT (Page 2506)

Implicit conversion of DINT (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the DINT data type:

Source	Target	With IEC check	Without IEC check	Explanation
DINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WORD	-	X	
	DWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the target data type. (for example, value conversion from DINT #-1 -> SINT #-1 or DINT #-1 -> USINT #255)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	UDINT	-	X	
	REAL	-	X	The bit pattern of the source value is converted and transferred to the target data type. (for example, value conversion from DINT #-1 -> REAL #-1.0, but there is a loss in accuracy for numbers with an absolute value greater than 8 388 608)
	LREAL	X	X	The value is converted to the format of the target data type. (The value "-1", for example, is converted to the value "-1.0".)
	TIME	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	DTL	-	-	No implicit conversion
	TOD	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	DATE	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WCHAR	-	X	
x: Conversion possible -: Conversion not possible				

See also

DINT (32-bit integers) (Page 2252)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of DINT (Page 2509)

Implicit conversion of UDINT (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the UDINT data type:

Source	Target	With IEC check	Without IEC check	Explanation
UDINT	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WORD	-	X	
	DWORD	-	X	
	SINT	-	X	The bit pattern of the source value is converted and transferred to the target data type. (for example, value conversion from DINT #-1 -> SINT #-1 or DINT #-1 -> USINT #255)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	REAL	-	X	The bit pattern of the source value is converted and transferred to the target data type. (for example, value conversion from DINT #-1 -> REAL #-1.0, but there is a loss in accuracy for numbers with an absolute value greater than 8 388 608)
	LREAL	X	X	The value is converted to the format of the target data type. (The value "1", for example, is converted to the value "1.0".)
	TIME	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	DTL	-	-	No implicit conversion
	TOD	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	DATE	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	WCHAR	-	X	

x: Conversion possible
-: Conversion not possible

See also

UDINT (32-bit integers) (Page 2253)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of UDINT (Page 2512)

Floating-point numbers (S7-1200)

Implicit conversion of REAL (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the REAL data type:

Source	Target	With IEC check	Without IEC check	Explanation
REAL	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.
	SINT	-	X	The bit pattern of the source value is rounded off and converted and transferred to the target data type. (for example, rounding off and value conversion of REAL #2.5 -> INT #2 or negative number REAL #-2.5 -> INT #-2 -> USINT #254. With an overflow, the remainder is determined REAL #305.5 -> INT #306 -> USINT #50)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	LREAL	X	X	The value is transferred to the target data type.
	TIME	-	-	No implicit conversion
	DTL	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

REAL (Page 2257)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of REAL (Page 2515)

Implicit conversion of LREAL (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the LREAL data type:

Source	Target	With IEC check	Without IEC check	Explanation
LREAL	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	SINT	-	X	The bit pattern of the source value is rounded off and converted and transferred to the target data type. (for example, rounding off and value conversion of REAL #2.5 -> INT #2 or negative number REAL #-2.5 -> INT #-2 -> USINT #254. With an overflow, the remainder is determined REAL #305.5 -> INT #306 -> USINT #50)
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	REAL	-	-	No implicit conversion
	TIME	-	-	
	DTL	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

Explicit conversion of LREAL (Page 2518)

Timers (S7-1200)

Implicit conversion of TIME (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the TIME data type:

Source	Target	With IEC check	Without IEC check	Explanation
TIME	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion shows the duration in milliseconds.
	SINT	-	-	No implicit conversion
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion shows the duration in milliseconds.
	UDINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	DTL	-	-	
	TOD	-	X	The bit pattern of a source value that is less than 24 h (86 400 00 ms) is transferred without changes to the target data type. No further changes are made to the target value. The result of the conversion shows the time that has passed since midnight.
	DATE	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	

x: Conversion possible
-: Conversion not possible

See also

TIME (IEC time) (Page 2262)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of TIME (Page 2521)

Date and time (S7-1200)

Implicit conversion of DATE (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the DATE data type:

Source	Target	With IEC check	Without IEC check	Explanation
DATE	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion corresponds to the number of days since 01-01-1990.
	DWORD	-	-	No implicit conversion
	SINT	-	-	
	USINT	-	-	
	INT	-	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion corresponds to the number of days since 01-01-1990.
	UINT	-	X	
	DINT	-	-	No implicit conversion
	UDINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	DTL	-	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion corresponds to the number of days since 01-01-1990.
	TOD	-	-	No implicit conversion
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	

x: Conversion possible
-: Conversion not possible

See also

DATE (Page 2264)
Setting and canceling the IEC check (Page 2467)
Overview of data type conversion (Page 2465)
Explicit conversion of DATE (Page 2523)

Implicit conversion of TOD (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the TOD data type:

Source	Target	With IEC check	Without IEC check	Explanation
TOD	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion corresponds to the number of milliseconds since the start of day (0:00 hrs).
	SINT	-	-	No implicit conversion
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion corresponds to the number of milliseconds since the start of day (0:00 hrs).
	UDINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion corresponds to the number of milliseconds since the start of day (0:00 hrs).
	DTL	-	-	No implicit conversion
	DATE	-	-	
	STRING	-	-	
	WSTRING	-	-	
	CHAR	-	-	
	WCHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

TIME_OF_DAY (TOD) (Page 2264)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of TOD (Page 2524)

Implicit conversion of DTL (S7-1200)

Options for implicit conversion

The DTL data type cannot be implicitly converted.

See also

Explicit conversion of DTL (Page 2525)

Character strings (S7-1200)

Implicit conversion of CHAR (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the CHAR data type:

Source	Target	With IEC check	Without IEC check	Explanation
CHAR	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. The remaining bits are filled from the left with "0".
	WORD	-	X	
	DWORD	-	X	
	SINT	-	X	
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	DTL	-	-	
	TOD	-	-	
	DATE	-	-	
	WCHAR	-	-	
	STRING	X	X	The STRING is shortened to length 1 and includes the character.
	WSTRING	-	-	No implicit conversion
x: Conversion possible -: Conversion not possible				

See also

CHAR (Page 2269)

Setting and canceling the IEC check (Page 2467)

Overview of data type conversion (Page 2465)

Explicit conversion of CHAR (Page 2526)

Implicit conversion of WCHAR (S7-1200)

Options for implicit conversion

The following table shows the options for the implicit conversion of the WCHAR data type:

Source	Target	With IEC check	Without IEC check	Explanation
WCHAR	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. The remaining bits are filled from the left with "0".
	WORD	-	X	
	DWORD	-	X	
	SINT	-	X	
	USINT	-	X	
	INT	-	X	
	UINT	-	X	
	DINT	-	X	
	UDINT	-	X	
	REAL	-	-	No implicit conversion
	LREAL	-	-	
	TIME	-	-	
	DTL	-	-	
	TOD	-	-	
	DATE	-	-	
	CHAR	-	-	
	STRING	-	-	
	WSTRING	X	X	The WSTRING is shortened to length 1 and includes the character.
x: Conversion possible -: Conversion not possible				

Implicit conversion of STRING (S7-1200)

Options for implicit conversion

The following table shows the options for implicit conversion of the STRING data type:

Source	Target	With IEC check	Without IEC check	Explanation
STRING	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	DTL	-	-	
	DATE	-	-	
	TOD	-	-	
	CHAR	-	X	The first character of the STRING is returned if the STRING includes one or more characters. Otherwise, the character is output with coding $00.
	WCHAR	-	-	No implicit conversion
	WSTRING	-	-	
x: Conversion possible -: Conversion not possible				

See also

Explicit conversion of STRING (Page 2528)

Implicit conversion of WSTRING (S7-1200)

Options for implicit conversion

The following table shows the options for the implicit conversion of the WSTRING data type:

Source	Target	With IEC check	Without IEC check	Explanation
WSTRING	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	-	
	SINT	-	-	
	USINT	-	-	
	INT	-	-	
	UINT	-	-	
	DINT	-	-	
	UDINT	-	-	
	REAL	-	-	
	LREAL	-	-	
	TIME	-	-	
	DTL	-	-	
	DATE	-	-	
	TOD	-	-	
	CHAR	-	-	
	WCHAR	-	X	The first character of the WSTRING is returned if the WSTRING includes one or more characters. Otherwise, the character is output with coding $0000.
	STRING	-	-	No implicit conversion
x: Conversion possible -: Conversion not possible				

11.3.16.3 Explicit conversions (S7-1200)

Binary numbers (S7-1200)

Explicit conversion of BOOL (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the BOOL data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
BOOL	BYTE	X	Only the LSB (Least Significant Bit) is set in the target data type. The enable output ENO is always "1".	BOOL_TO_BYTE
	WORD	X		BOOL_TO_WORD
	DWORD	X		BOOL_TO_DWORD
	SINT	X		BOOL_TO_SINT
	USINT	X		BOOL_TO_USINT
	INT	X		BOOL_TO_INT
	UINT	X		BOOL_TO_UINT
	DINT	X		BOOL_TO_DINT
	UDINT	X		BOOL_TO_UDINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	-		-
	DTL	-		-
	TOD	-		-
	DATE	-		-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

x: Conversion possible
- : Conversion not possible

See also

BOOL (bit) (Page 2243)

Implicit conversion of BYTE (Page 2469)

Overview of data type conversion (Page 2465)

Bit strings (S7-1200)

Explicit conversion of BYTE (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the BYTE data type:

Source	Target	Conversion	Explanation	Mnemonics of Instruction
BYTE[1)]	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bits are not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	BYTE_TO_BOOL
	WORD[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	BYTE_TO_WORD
	DWORD[1)]	X		BYTE_TO_DWORD
	SINT	X		BYTE_TO_SINT
	USINT	X		BYTE_TO_USINT
	INT	X		BYTE_TO_INT
	UINT	X		BYTE_TO_UINT
	DINT	X		BYTE_TO_DINT
	UDINT	X		BYTE_TO_UDINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	BYTE_TO_TIME
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	BYTE_TO_TOD
	DATE	X		BYTE_TO_DATE
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	BYTE_TO_CHAR
	WCHAR	X		BYTE_TO_WCHAR

Source	Target	Conversion	Explanation	Mnemonics of Instruction
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD) are interpreted as an unsigned integer with the same bit length.. Data type BYTE is interpreted as USINT, WORD as UINT and DWORD as UDINT.				

See also

BYTE (Page 2244)

Implicit conversion of BYTE (Page 2469)

Overview of data type conversion (Page 2465)

Explicit conversion of WORD (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the WORD data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
WORD[1)]	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bit is not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	WORD_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	WORD_TO_BYTE
	DWORD[1)]	X		WORD_TO_DWORD

	SINT	X	ENO = TRUE
			#sint1 := WORD_TO_SINT(16#FFFF); // -1 to
			#sint1 := WORD_TO_SINT(16#FF80); // -128
			#sint1 := WORD_TO_SINT(16#0); // 0 to
			#sint1 := WORD_TO_SINT(16#007F); // 127
			
			ENO = FALSE
			#sint1 := WORD_TO_SINT(16#FF7F); // -129 to
			#sint1 := WORD_TO_SINT(16#8000); // -32768
			#sint1 := WORD_TO_SINT(16#0080); // 128 to
			#sint1 := WORD_TO_SINT(16#7FFF); // 32767
				WORD_TO_SINT

	USINT	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	WORD_TO_USINT
	INT	X		WORD_TO_INT
	UINT	X		WORD_TO_UINT
	DINT	X		WORD_TO_DINT
	UDINT	X		WORD_TO_UDINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	WORD_TO_TIME
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	WORD_TO_TOD
	DATE	X		WORD_TO_DATE

Source	Target	Conversion	Explanation	Mnemonics of the instruction
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	WORD_TO_CHAR
	WCHAR	X		WORD_TO_WCHAR
WORD_BCD16	INT	X	The value to be converted has data type WORD and is accepted as a BCD-coded value between -999 and +999. The result is available after conversion as an integer (in binary notation) of the type INT. A real conversion takes place. If the bit pattern includes an invalid tetrad, a synchronous error is not triggered but only the status bit OV is set instead.	WORD_BCD16_TO_INT
BCD16	INT	X		BCD16_TO_INT

x: Conversion possible

- : Conversion not possible

[1] Bit strings (BYTE, WORD, DWORD) are interpreted as an unsigned integer with the same bit length.. Data type BYTE is interpreted as USINT, WORD as UINT and DWORD as UDINT.

See also

WORD (Page 2245)

Implicit conversion of WORD (Page 2470)

Overview of data type conversion (Page 2465)

Explicit conversion of DWORD (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DWORD data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DWORD[1)]	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bit is not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	DWORD_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DWORD_TO_BYTE
	WORD[1)]	X		DWORD_TO_WORD

	SINT	X	ENO = TRUE
			#sint1 := DWORD_TO_SINT(16#FFFF_FFFF); // -1 bis
			#sint1 := DWORD_TO_SINT(16#FFFF_FF80); // -128
			#sint1 := DWORD_TO_SINT(16#0); // 0 bis
			#sint1 := DWORD_TO_SINT(16#0000_007F); // 127
			
			ENO = FALSE
			#sint1 := DWORD_TO_SINT(16#FFFF_FF7F); // -129
			#sint1 := DWORD_TO_SINT(16#8000_0000); // -2147483648
			#sint1 := DWORD_TO_SINT(16#0000_0080); // 128 bis
			#sint1 := DWORD_TO_SINT(16#7FFF_FFFF); // 2147483647
				DWORD_TO_SINT

	USINT	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DWORD_TO_USINT

	INT	X	ENO = TRUE
			#int1 := DWORD_TO_INT(16#FFFF_FFFF); // -1 bis
			#int1 := DWORD_TO_INT(16#FFFF_8000); //
				DWORD_TO_INT

Source	Target	Conversion	Explanation	Mnemonics of the instruction

			-32768
			#int1 := DWORD_TO_INT(16#0); // 0
			bis
			#int1 :=
			DWORD_TO_INT(16#0000_7FFF); //
			32767
			
			ENO = FALSE
			#int1 :=
			DWORD_TO_INT(16#FFFF_7FFF); //
			-32769
			#int1 :=
			DWORD_TO_INT(16#8000_0000); //
			-2147483648
			#int1 :=
			DWORD_TO_INT(16#8000); // 32768
			bis
			#int1 :=
			DWORD_TO_INT(16#7FFF_FFFF); //
			2147483647	

	UINT	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DWORD_TO_UINT
	DINT	X		DWORD_TO_DINT
	UDINT	X		DWORD_TO_UDINT
	REAL	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If no errors occur during the conversion, the signal state of ENO = 1; if an error occurs during processing, the signal state of ENO = 0.	DWORD_TO_REAL
	LREAL	-	No explicit conversion	-
	TIME	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DWORD_TO_TIME
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DWORD_TO_TOD
	DATE	X		DWORD_TO_DATE
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DWORD_TO_CHAR
	WCHAR	X		DWORD_TO_WCHAR

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DWORD_BCD32	DINT	X	The value to be converted has data type DWORD and is accepted as a BCD-coded value between -9999999 and +9999999. The result is available after conversion as an integer (in binary notation) of the type DINT. A real conversion takes place. If the bit pattern includes an invalid tetrad, a synchronous error is not triggered but only the status bit OV is set instead.	DWORD_BCD32_TO_DINT
BCD32	DINT	X		BCD32_TO_DINT
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD) are interpreted as an unsigned integer with the same bit length.. Data type BYTE is interpreted as USINT, WORD as UINT and DWORD as UDINT.				

See also

DWORD (Page 2245)

Implicit conversion of DWORD (Page 2471)

Overview of data type conversion (Page 2465)

Integers (S7-1200)

Explicit conversion of SINT (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the SINT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
SINT	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bit is not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	SINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0".	SINT_TO_BYTE
	WORD[1)]	X		SINT_TO_WORD
	DWORD[1)]	X		SINT_TO_DWORD
	USINT	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). Enable output ENO is set to "0" if a negative value is converted to an unsigned target data type.	SINT_TO_USINT
	INT	X		SINT_TO_INT
	UINT	X		SINT_TO_UINT
	DINT	X		SINT_TO_DINT
	UDINT	X		SINT_TO_UDINT
	REAL	X	The value is converted into the format of the target data type (the value "-1" will be converted into the value "-1.0" with the "Convert value" (CONVERT) instruction.	SINT_TO_REAL, NORM_X
	LREAL	X		SINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the target data type and interpreted as milliseconds.	SINT_TO_TIME
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). Enable output ENO is set to "0" if a negative value is converted to an unsigned target data type. (interpretation in milliseconds since 0:0)	SINT_TO_TOD
	DATE	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). Enable output ENO is set to "0" if a negative value is converted to an unsigned target data type. (interpretation in days since 1990-1-1)	SINT_TO_DATE
	STRING	X	The value is converted to a character string. • LAD/FBD: The first characters of the string are padded with spaces. The number of	SINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		SINT_TO_WSTRING

Source	Target	Conversion	Explanation	Mnemonics of the instruction
			spaces depends on the length of the numerical value. Positive numeric values are output without a sign. • STL/SCL: No spaces are inserted. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). Enable output ENO is set to "0" if a negative value is converted to an unsigned target data type.	SINT_TO_CHAR
	WCHAR[1)]	X		SINT_TO_WCHAR
x: Conversion possible - : Conversion not possible [1)] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width including sign, and then the bits are copied. The source type determines the interpretation.				

See also

SINT (8-bit integers) (Page 2248)

Implicit conversion of SINT (Page 2472)

Overview of data type conversion (Page 2465)

Explicit conversion of USINT (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the USINT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
USINT	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bit is not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	USINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	USINT_TO_BYTE
	WORD[1)]	X		USINT_TO_WORD
	DWORD[1)]	X		USINT_TO_DWORD
	SINT	X	The bit pattern of the source value is transferred unchanged to the target data type. If the sign is changed during the conversion, the enable output ENO is set to "0".	USINT_TO_SINT
	INT	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	USINT_TO_INT
	UINT	X		USINT_TO_UINT
	DINT	X		USINT_TO_DINT
	UDINT	X		USINT_TO_UDINT
	REAL	X	The value is converted into the format of the target data type (the value "1" will be converted into the value "1.0" with the "Convert value" (CONVERT) instruction.	USINT_TO_REAL, NORM_X
	LREAL	X		USINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the target data type and interpreted as milliseconds.	USINT_TO_TIME
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	USINT_TO_TOD
	DATE	X		USINT_TO_DATE
	STRING	X	The value is converted to a character string. If the permitted length of the character string is violated, the enable output ENO is set to "0".	USINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		USINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the target data type.	USINT_TO_CHAR
	WCHAR[1)]	X		USINT_TO_WCHAR

Source	Target	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width (the non-existing sign is replaced with zeros) and then the bits are copied. The source type determines the interpretation.				

See also

USINT (8-bit integers) (Page 2249)

Implicit conversion of USINT (Page 2473)

Overview of data type conversion (Page 2465)

Explicit conversion of INT (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the INT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
INT	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bit is not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	INT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0".	INT_TO_BYTE
	WORD[1)]	X		INT_TO_WORD
	DWORD[1)]	X		INT_TO_DWORD
	SINT	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0".	INT_TO_SINT
	USINT	X		INT_TO_USINT
	UINT	X		INT_TO_UINT
	DINT	X		INT_TO_DINT
	UDINT	X		INT_TO_UDINT
	REAL	X	The value is converted to the format of the target data type (the value "-1", for example, is converted with the instruction "Convert value" (CONVERT) to the value "-1.0").	INT_TO_REAL, NORM_X
	LREAL	X		INT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the target data type and interpreted as milliseconds.	INT_TO_TIME
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0". (interpretation in milliseconds since 0:0; check for 24h limit)	INT_TO_TOD
	DATE	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0". (interpretation	INT_TO_DATE

Source	Target	Conversion	Explanation	Mnemonics of the instruction
			in days since 1990-1-1; check for negative value)	
	STRING	X	The value is converted to a character string. • LAD/FBD: The first characters of the string are padded with spaces. The number of spaces depends on the length of the numerical value. Positive numeric values are output without a sign. • STL/SCL: No spaces are inserted. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	INT_TO_STRING, S_CONV, VAL_STRG[1)]
	WSTRING	X		INT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0".	INT_TO_CHAR
	WCHAR[1)]	X		INT_TO_WCHAR
	BCD16	X	The value to be converted has type INT and is accepted as an integer with a value between -999 and +999. The result is available after conversion as a BCD-coded number of the type WORD. A real conversion takes place. If the value is outside the target area, a synchronous error is not triggered, but rather only the status bit OV is set.	INT_TO_BCD16
	BCD16_WORD	X		INT_TO_BCD16_WORD

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width including sign, and then the bits are copied. The source type determines the interpretation.

See also

INT (16-bit integers) (Page 2250)

Implicit conversion of INT (Page 2474)

Overview of data type conversion (Page 2465)

Explicit conversion of UINT (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the UINT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
UINT	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bit is not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	UINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If bits are lost in the process, the enable output ENO is set to "0".	UINT_TO_BYTE
	WORD[1)]	X		UINT_TO_WORD
	DWORD[1)]	X		UINT_TO_DWORD
	SINT	X		UINT_TO_SINT
	USINT	X		UINT_TO_USINT
	INT	X	The bit pattern of the source value is transferred unchanged to the target data type. If the sign bit is changed during the conversion, the enable output ENO is set to "0".	UINT_TO_INT
	DINT	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	UINT_TO_DINT
	UDINT	X		UINT_TO_UDINT
	REAL	X	The value is converted to the format of the target data type (the value "1", for example, is converted with the instruction "Convert value" (CONVERT) to the value "1.0").	UINT_TO_REAL, NORM_X
	LREAL	X		UINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the target data type and interpreted as milliseconds.	UINT_TO_TIME
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). Enable output ENO is set to "0" if a negative value is converted to an unsigned target data type. (interpretation in milliseconds since 0:0; check for 24h limit)	UINT_TO_TOD
	DATE	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). Enable output	UINT_TO_DATE, T_CONV

Source	Target	Conversion	Explanation	Mnemonics of the instruction
			ENO is set to "0" if a negative value is converted to an unsigned target data type. (interpretation in days since 1990-1-1; check for negative value)	
	STRING	X	The value is converted to a character string. If the permitted length of the character string is violated, the enable output ENO is set to "0".	UINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		UINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is transferred unchanged to the target data type. The enable output ENO is set to "0" in the event of overflow.	UINT_TO_CHAR
	WCHAR[1)]	X		UINT_TO_WCHAR
x: Conversion possible - : Conversion not possible [1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data type CHAR are first extended to the necessary width (the non-existing sign is replaced with zeros) and then the bits are copied. The source type determines the interpretation.				

See also

UINT (16-bit integers) (Page 2251)

Implicit conversion of UINT (Page 2475)

Overview of data type conversion (Page 2465)

Explicit conversion of DINT (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the DINT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DINT	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bit is not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	DINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0".	DINT_TO_BYTE
	WORD[1)]	X		DINT_TO_WORD
	DWORD[1)]	X		DINT_TO_DWORD
	SINT	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0".	DINT_TO_SINT
	USINT	X		DINT_TO_USINT
	INT	X		DINT_TO_INT
	UINT	X		DINT_TO_UINT
	UDINT	X		DINT_TO_UDINT
	REAL	X	The value is converted to the format of the target data type (the value "-1", for example, is converted with the instruction "Convert value" (CONVERT) to the value "-1.0").	DINT_TO_REAL, NORM_X
	LREAL	X		DINT_TO_LREAL, NORM_X
	TIME	X	The value is transferred to the target data type and interpreted as milliseconds.	DINT_TO_TIME, T_CONV
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0". (interpretation in milliseconds since 0:0)	DINT_TO_TOD
	DATE	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted to an unsigned target	DINT_TO_DATE

Source	Target	Conversion	Explanation	Mnemonics of the instruction
			data type, the enable output ENO is set to "0". (interpretation in days since 1990-1-1)	
	STRING	X	The value is converted to a character string.	DINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X	• LAD/FBD: The first characters of the string are padded with spaces. The number of spaces depends on the length of the numerical value. Positive numeric values are output without a sign. • STL/SCL: No spaces are inserted. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	DINT_TO_WSTRING
	CHAR[1)]	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF)). If a negative value is converted to an unsigned target data type, the enable output ENO is set to "0".	DINT_TO_CHAR
	WCHAR[1)]	X		DINT_TO_WCHAR
	BCD32	X	The value to be converted has type DINT and is accepted as an integer with a value between –999999 and +9999999. The result is available after conversion as a BCD-coded number of the type DWORD. The enable output is set to "0" in the event of overflow. A real conversion takes place. If the value is outside the target area, a synchronous error is not triggered, but rather only the status bit OV is set.	DINT_TO_BCD32
	BCD32_DWORD	X		DINT_TO_BCD32_DWORD

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width including sign, and then the bits are copied. The source type determines the interpretation.

See also

DINT (32-bit integers) (Page 2252)

Implicit conversion of DINT (Page 2476)

Overview of data type conversion (Page 2465)

Explicit conversion of UDINT (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the UDINT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
UDINT	BOOL	X	The following scenarios are possible: • If the source is "0", the target data type is also "0" and enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and enable output ENO is "1". • If bit is not equal to LSB in the source value, the target data type is set according to LSB and enable output ENO is "0".	UDINT_TO_BOOL
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged to the target data type. If bits are lost in the process, the enable output ENO is set to "0".	UDINT_TO_BYTE
	WORD[1)]	X		UDINT_TO_WORD
	DWORD[1)]	X		UDINT_TO_DWORD
	SINT	X		UDINT_TO_SINT
	USINT	X		UDINT_TO_USINT
	INT	X		UDINT_TO_INT
	UINT	X		UDINT_TO_UINT
	DINT	X	The bit pattern of the source value is transferred unchanged to the target data type. If the sign bit is changed during the conversion, the enable output ENO is set to "0".	UDINT_TO_DINT
	REAL	X	The value is converted to the format of the target data type (the value "1", for example, is converted with the instruction "Convert value" (CONVERT) to the value "1.0").	UDINT_TO_REAL, NORM_X
	LREAL	X		UDINT_TO_LREAL, NORM_X
	TIME	X	The bit pattern of the source value is transferred unchanged right-justified and interpreted as milliseconds to the target data type.	UDINT_TO_TIME
	DTL	-	No explicit conversion	-
	TOD	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). Enable output ENO is set to "0" if a negative value is converted to an unsigned target data type. (interpretation in milliseconds since 0:0; check for 24h limit)	UDINT_TO_TOD, T_CONV
	DATE	X	The bit pattern of the source value is converted and transferred to the target data type. (The value "-1" (16#FF) becomes the value "-1" (16#FFFFFFFF). Enable output	UDINT_TO_DATE

Source	Target	Conversion	Explanation	Mnemonics of the instruction
			ENO is set to "0" if a negative value is converted to an unsigned target data type. (interpretation in days since 1990-1-1; check for negative value)	
	STRING	X	The value is converted to a character string. If the permitted length of the character string is violated, the enable output ENO is set to "0".	UDINT_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		UDINT_TO_WCHAR
	CHAR[1)]	X	The bit pattern of the source value is transferred unchanged to the target data type. The enable output ENO is set to "0" in the event of overflow.	UDINT_TO_CHAR
	WCHAR[1)]	X		UDINT_TO_WCHAR
x: Conversion possible - : Conversion not possible [1)] Bit strings (BYTE, WORD, DWORD, LWORD) and the data type CHAR are first extended to the necessary width (the non-existent sign is replaced with zeros) and then the bits are copied. The source type determines the interpretation.				

See also

UDINT (32-bit integers) (Page 2253)

Implicit conversion of UDINT (Page 2477)

Overview of data type conversion (Page 2465)

Floating-point numbers (S7-1200)

Explicit conversion of REAL (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the REAL data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
REAL	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	X	The bit pattern of the source value is transferred unchanged to the target data type.	REAL_TO_DWORD
	SINT	X	The value is converted to the target data type. The result of the conversion depends on the instruction used. Enable output ENO is set to "0" if the valid range of values of the target data type is exceeded during conversion, or if the value to be converted is an invalid floating-point number.	REAL_TO_SINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	USINT	X		REAL_TO_USINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	INT	X		REAL_TO_INT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	UINT	X		REAL_TO_UINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	DINT	X		REAL_TO_DINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	UDINT	X		REAL_TO_UDINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	LREAL	X	The value is converted to the target data type. The result of the conversion depends on the instruction used, e.g. TRUNC(2.5) = 2.0; CEIL(2.5) = 3.0	REAL_TO_LREAL, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	TIME	-	No explicit conversion	-
	DTL	-		-
	TOD	-		-
	DATE	-		-
	STRING	X	The value is converted to a character string. Enable output ENO is set to "0" if the character string length is exceeded, or if the value to be converted is an invalid floating-point number. The string has a minimum length of 14 characters.	REAL_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		REAL_TO_WSTRING
	CHAR	-	No explicit conversion	-
	WCHAR	-		-

Source	Target	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible -: Conversion not possible				

See also

REAL (Page 2257)

Implicit conversion of REAL (Page 2478)

Overview of data type conversion (Page 2465)

Explicit conversion of LREAL (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the LREAL data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
LREAL	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	-		-
	SINT	X	The value is converted to the target data type. The result of the conversion depends on the instruction used. Enable output ENO is set to "0" if the valid range of values of the target data type is exceeded during conversion, or if the value to be converted is an invalid floating-point number.	LREAL_TO_SINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	USINT	X		LREAL_TO_USINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	INT	X		LREAL_TO_INT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	UINT	X		LREAL_TO_UINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	DINT	X		LREAL_TO_DINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	UDINT	X		LREAL_TO_UDINT, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	REAL	X	The value is converted to the target data type. Enable output ENO is set to "0" if the valid range of values of the target data type is exceeded during conversion, or if the value to be converted is an invalid floating-point number. A loss in accuracy is tolerated.	LREAL_TO_LREAL, ROUND, CEIL, FLOOR, TRUNC, NORM_X, SCALE_X
	TIME	-	No explicit conversion	-
	DTL	-		-
	TOD	-		-
	DATE	-		-
	STRING	X	The value is converted to a character string. Enable output ENO is set to "0" if the character string length is exceeded, or if the value to be converted is an invalid floating-point number. The string has a minimum length of 21 characters.	LREAL_TO_STRING, S_CONV, VAL_STRG
	WSTRING	X		LREAL_TO_WSTRING
	CHAR	-	No explicit conversion	-
	WCHAR	-		-
x: Conversion possible -: Conversion not possible				

See also

LREAL (Page 2258)

Overview of data type conversion (Page 2465)

Timers (S7-1200)

Explicit conversion of TIME (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the TIME data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
TIME	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged to the target data type.	TIME_TO_BYTE
	WORD[1)]	X		TIME_TO_WORD
	DWORD[1)]	X		TIME_TO_DWORD
	SINT	X	The bit pattern of the source value is transferred unchanged right-justified and interpreted as milliseconds to the target data type.	TIME_TO_SINT
	USINT	X		TIME_TO_USINT
	INT	X		TIME_TO_INT
	UINT	X		TIME_TO_UINT
	DINT	X	The bit pattern of the source value is transferred unchanged to the target data type. The result of the conversion shows the duration in milliseconds.	TIME_TO_DINT, T_CONV
	UDINT	X	The bit pattern of the source value is transferred unchanged right-justified and interpreted as milliseconds to the target data type. Enable output ENO is set to "0" if the sign changes.	TIME_TO_UDINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	DTL	-		-
	TOD	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If the source value exceeds the range of values of TOD, the target data type remains unchanged.	TIME_TO_TOD
	DATE	-	No explicit conversion	-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

Source	Target	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible -: Conversion not possible [1] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width and then the bits are copied. The source type determines the interpretation.				

See also

TIME (IEC time) (Page 2262)

Implicit conversion of TIME (Page 2480)

Overview of data type conversion (Page 2465)

Clock and calendar (S7-1200)

Explicit conversion of DATE (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the DATE data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DATE	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DATE_TO_BYTE
	WORD[1)]	X		DATE_TO_WORD
	DWORD[1)]	X		DATE_TO_DWORD
	SINT	X	The number of days since 1/1/1990 is returned as result.	DATE_TO_SINT
	USINT	X		DATE_TO_USINT
	INT	X		DATE_TO_INT
	UINT	X		DATE_TO_UINT
	DINT	X		DATE_TO_DINT
	UDINT	X		DATE_TO_UDINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	-		-
	DTL [2)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	DATE_TO_DTL
	TOD	-	No explicit conversion	-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width and then the bits are copied. The source type determines the interpretation.

[2)] The data type DTL is initialized as follows: 1970-1-1-0:0:0. Conversion from DATE_TO_DTL only sets the corresponding part of the DTL. The rest of the DTL remains as it was initialized. This results in the following differences for the instructions "T_COMBINE" and "T_CONV":

- T_COMBINE(D#2015-24-12, LTOD#12:13:14) returns DTL#2015-24-12-12:13:14
- myDTL := T_CONV(D#2015-24-12); myDTL := T_CONV(LTOD#12:13:14) results in myDTL = DTL#1970-1-1-12:13:14

See also

DATE (Page 2264)

Implicit conversion of DATE (Page 2481)

Overview of data type conversion (Page 2465)

Explicit conversion of TOD (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the TOD data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
TOD	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged to the target data type.	TOD_TO_BYTE
	WORD[1)]	X		TOD_TO_WORD
	DWORD[1)]	X		TOD_TO_DWORD
	SINT	X	The number of milliseconds since midnight is returned as result.	TOD_TO_SINT
	USINT	X		TOD_TO_USINT
	INT	X		TOD_TO_INT
	UINT	X		TOD_TO_UINT
	DINT	X		TOD_TO_DINT
	UDINT	X	The result of the conversion corresponds to the number of milliseconds since the start of day (0:00 hrs).	TOD_TO_UDINT, T_CONV
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	X	The duration since midnight is returned as result.	TOD_TO_TIME
	DTL	X	The date is set to 1.1.1970 as a result.	TOD_TO_DTL
	DATE	-	No explicit conversion	-
	STRING	-		-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-

x: Conversion possible

-: Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width including sign, and then the bits are copied. The source type determines the interpretation.

See also

TIME_OF_DAY (TOD) (Page 2264)

Implicit conversion of TOD (Page 2482)

Overview of data type conversion (Page 2465)

Explicit conversion of DTL (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the DTL data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DTL	BYTE	-	No explicit conversion	-
	WORD	-		-
	DWORD	-		-
	SINT	-		-
	USINT	-		-
	INT	-		-
	UINT	-		-
	DINT	-		-
	UDINT	-		-
	REAL	-		-
	LREAL	-		-
	TIME	-		-
	TOD	X	During the conversion, the time-of-day is extracted from the DTL format and written to the target data type.	DTL_TO_TOD, T_CONV
	DATE	X	During the conversion, the date is extracted from the DTL format and written to the target data type. The enable output ENO is set to "0" in the event of overflow.	DTL_TO_DATE, T_CONV
	STRING	-	No explicit conversion	-
	WSTRING	-		-
	CHAR	-		-
	WCHAR	-		-
x: Conversion possible -: Conversion not possible				

See also

DTL (Page 2267)

Overview of data type conversion (Page 2465)

Character strings (S7-1200)

Explicit conversion of CHAR (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the CHAR data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
CHAR	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	CHAR_TO_BYTE
	WORD[1)]	X		CHAR_TO_WORD
	DWORD[1)]	X		CHAR_TO_DWORD
	SINT	X		CHAR_TO_SINT
	USINT	X		CHAR_TO_USINT
	INT	X		CHAR_TO_INT
	UINT	X		CHAR_TO_UINT
	DINT	X		CHAR_TO_DINT
	UDINT	X		CHAR_TO_UDINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	-		-
	DTL	-		-
	TOD	-		-
	DATE	-		-
	STRING	X	The value is converted to the first character in the character string (STRING). If the length of the character string is not defined, the length "1" is set after the conversion. If the length of the character string is defined, it remains unchanged after the conversion.	CHAR_TO_STRING
	WSTRING	-	No explicit conversion	-
	WCHAR	X		CHAR_TO_WCHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width and then the bits are copied. The source type determines the interpretation.

See also

CHAR (Page 2269)

Implicit conversion of CHAR (Page 2483)

Overview of data type conversion (Page 2465)

Explicit conversion of WCHAR (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the WCHAR data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
WCHAR	BOOL	-	No explicit conversion	-
	BYTE[1)]	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	WCHAR_TO_BYTE
	WORD[1)]	X		WCHAR_TO_WORD
	DWORD[1)]	X		WCHAR_TO_DWORD
	SINT	X		WCHAR_TO_SINT
	USINT	X		WCHAR_TO_USINT
	INT	X		WCHAR_TO_INT
	UINT	X		WCHAR_TO_UINT
	DINT	X		WCHAR_TO_DINT
	UDINT	X		WCHAR_TO_UDINT
	REAL	-	No explicit conversion	-
	LREAL	-		-
	TIME	-		-
	DTL	-		-
	TOD	-		-
	DATE	-		-
	STRING	-		-
	WSTRING	X	The value is converted to the first character in the character string (WSTRING). If the length of the character string is not defined, the length "1" is set after conversion. If the length of the character string is defined, it remains unchanged after conversion.	WCHAR_TO_WSTRING
	CHAR	X		WCHAR_TO_CHAR

x: Conversion possible

- : Conversion not possible

[1)] Bit strings (BYTE, WORD, DWORD) and data type CHAR are first extended to the necessary width and then the bits are copied. The source type determines the interpretation.

Explicit conversion of STRING (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the STRING data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
STRING	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	-		-
	SINT	X	Conversion begins with the first character in the character string (STRING) and stops at the end of the string or at the first inadmissible character. The following characters are permitted for conversion: • Digit • Sign • Dot The first character of the string may be a sign (+, -) or a number. Leading spaces are ignored. The dot is used as separation for the conversion of floating-point numbers. The exponential notation "e" or "E" is not permitted. The comma as thousands separator is permitted to the left of the decimal point but will be ignored. If the layout of the character string is invalid for the conversion or an overflow occurs, then the enable output ENO will be set to "0".	STRING_TO_SINT, S_CONV, STRG_VAL
	USINT	X		STRING_TO_USINT, S_CONV, STRG_VAL
	INT	X		STRING_TO_INT, S_CONV, STRG_VAL
	UINT	X		STRING_TO_UINT, S_CONV, STRG_VAL
	DINT	X		STRING_TO_DINT, S_CONV, STRG_VAL
	UDINT	X		STRING_TO_UDINT, S_CONV, STRG_VAL
	REAL	X		STRING_TO_REAL, S_CONV, STRG_VAL
	LREAL	X		STRING_TO_LREAL, S_CONV, STRG_VAL
	TIME	-	No explicit conversion	-
	DTL	-		-
	TOD	-		-
	DATE	-		-
	CHAR	X	The first character in the character string (STRING) is transferred to the destination data type. If the string is empty, then the value "0" will be written in the destination data type.	STRING_TO_CHAR, S_CONV
	WCHAR	-	No explicit conversion	-
	WSTRING	X		STRING_TO_WSTRING
x: Conversion possible - : Conversion not possible				

See also

STRING (Page 2270)

Overview of data type conversion (Page 2465)

Explicit conversion of WSTRING (S7-1200)

Options for explicit conversion

The following table shows the options and instructions for explicit conversion of the WSTRING data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
WSTRING	BOOL	-	No explicit conversion	-
	BYTE	-		-
	WORD	-		-
	DWORD	-		-
	SINT	X	Conversion begins with the first character in the character string (STRING) and stops at the end of the string or at the first inadmissible character. The following characters are permitted for conversion: • Digit • Sign • Dot The first character of the string may be a sign (+, -) or a number. Leading spaces are ignored. The dot is used as separation for the conversion of floating-point numbers. The exponential notation "e" or "E" is not permitted. The comma as thousands separator is permitted to the left of the decimal point but will be ignored. If the layout of the character string is invalid for the conversion, or an overflow occurs, the enable output ENO will be set to "0".	WSTRING_TO_SINT, S_CONV, STRG_VAL
	USINT	X		WSTRING_TO_USINT, S_CONV, STRG_VAL
	INT	X		WSTRING_TO_INT, S_CONV, STRG_VAL
	UINT	X		WSTRING_TO_UINT, S_CONV, STRG_VAL
	DINT	X		WSTRING_TO_DINT, S_CONV, STRG_VAL
	UDINT	X		WSTRING_TO_UDINT, S_CONV, STRG_VAL
	REAL	X		WSTRING_TO_REAL, S_CONV, STRG_VAL
	LREAL	X		WSTRING_TO_LREAL, S_CONV, STRG_VAL
	TIME	-	No explicit conversion	-
	DTL	-		-
	TOD	-		-
	DATE	-		-
	CHAR	-		-
	WCHAR	X	The first character in the character string (WSTRING) is transferred to the target data type. If the string is empty, the value "0" will be written in the target data type.	WSTRING_TO_WCHAR
	STRING	X		WSTRING_TO_STRING
x: Conversion possible - : Conversion not possible				

11.3.17 Data type conversion for S7-300, S7-400 (S7-300, S7-400)

11.3.17.1 Overview of data type conversions (S7-300, S7-400)

Introduction

If you combine several operands in an instruction, you must make sure that the data types are compatible. This also applies when you assign or supply block parameters. If the operands are not of the same data type, a conversion has to be carried out.

There are two options for conversion:

- Implicit conversion
  Implicit conversion is supported by the programming languages LAD, FBD, SCL and GRAPH. Implicit conversion is not possible in the STL programming language.
- Explicit conversion

Note

Converting bit strings to SCL

All bit strings (BYTE, WORD, and DWORD) are handled like the corresponding unsigned integers (USINT, UINT, and UDINT) in expressions. Therefore, implicit conversion from DWORD to REAL is carried out like a conversion from UDINT to REAL, for example.

Implicit conversion

Implicit conversion is executed automatically if the data types of the operands are compatible. This compatibility test can be carried out according to strict or less strict criteria:

- With IEC check
  The following rules apply in the LAD, FBD and GRAPH programming languages when the IEC check has been set:
  - The only data types for which implicit conversion is possible are BYTE and WORD.
  - The bit length of the source data type may not exceed the bit length of the target data type. A WORD data type operand, for example, cannot be specified at a parameter at which the BYTE data type is expected.

  The following rules apply in the SCL programming language when the IEC check has been set:
  - Implicit conversion of bit strings into other data types is not possible. A WORD data type operand, for example, cannot be specified at a parameter at which the INT data type is expected.
  - The bit length of the source data type may not exceed the bit length of the target data type. A WORD data type operand, for example, cannot be specified at a parameter at which the BYTE data type is expected.
- Without IEC check (default setting)
  The following rules apply in the LAD, FBD and GRAPH programming languages when the IEC check has not been set:
  - Implicit conversion of data types BYTE, WORD, DWORD, INT, DINT, TIME, S5TIME, TOD, DATE and CHAR is possible.
  - The bit length of the source data type may not exceed the bit length of the target data type. A DWORD data type operand, for example, cannot be specified at a parameter at which the WORD data type is expected.

  The following rules apply in the SCL programming language when the IEC check has not been set:
  - Bit strings can be implicitly converted into other data types. A WORD data type operand can, for example, be given at a parameter at which the WORD data type is expected.
  - Bit strings cannot be implicitly converted into floating-point numbers. A WORD data type operand, for example, cannot be specified at a parameter at which the REAL data type is expected.
  - Bit strings can only be implicitly converted into the data types TIME, TOD, DATE and CHAR if these have the same bit length. A DWORD data type operand, for example, cannot be specified at a parameter at which the DATE data type is expected.
  - The bit length of the source data type may not exceed the bit length of the target data type. A DINT data type operand, for example, cannot be specified at a parameter at which the INT data type is expected.
  - The bit length of an operand entered at in-out parameters must be the same as the programmed bit length for the parameter in question.

Explicit conversion

If the operands are not compatible and implicit conversion is therefore not possible, you can use an explicit conversion instruction. You can find the conversion instructions in the "Instructions" task card.

Any overflow will be displayed on the ENO enable output. An overflow occurs, for example, if the value of the source data type is greater than the value of the target data type.

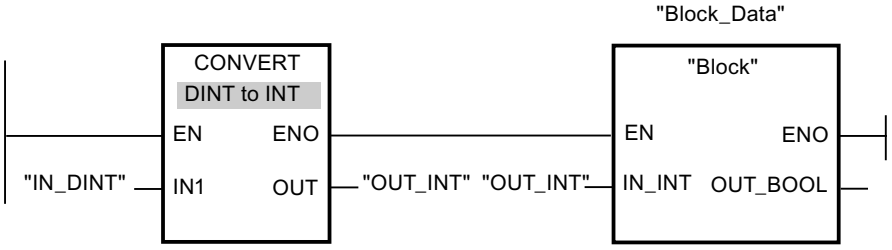
Note

Shifting of bit patterns

If the explicit conversion involves shifting a bit pattern, the enable output ENO is not set.

You can find additional information on explicit conversion under "See also".

The following figure shows an example in which explicit data type conversion must be carried out:



The "Block" function block expects a tag of the INT data type at the "IN_INT" input parameter. The value of the "IN_DINT" tag has therefore to the converted first from DINT to INT. Conversion is performed if the value of the "IN_DINT" tag is within the admissible value range for the INT data type. Otherwise an overflow is reported. However, a conversion takes place even in the event of an overflow, but the values are truncated and the ENO enable output is set to "0".

See also

Setting and disabling the IEC check (Page 2533)

Implicit conversions (Page 2533)

Explicit conversions (Page 2545)

11.3.17.2 Implicit conversions (S7-300, S7-400)

Setting and disabling the IEC check (S7-300, S7-400)

The data types of the tags used are checked for compatibility. This compatibility test can be carried out according to criteria that are more or less strict. If "IEC check" is activated, stricter criteria are applied.

You can set the IEC check centrally for all new blocks of the project or for individual blocks.

Setting IEC check for new blocks

To set the IEC check for all new blocks in the project, proceed as follows:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "PLC programming > General" group in the area navigation.
3. Select or clear the "IEC check" check box in the "Default settings for new blocks" group.
   The IEC check is enabled or disabled for all new blocks in the program.

Setting IEC check for a block

To set the IEC check for a block, proceed as follows:

1. Open the block.
2. Open the "Properties" tab in the inspector window.
3. Select the "Attributes" group in the area navigation.
4. Select or clear the "IEC check" check box.
   The IEC check is enabled or disabled for this block. The setting is stored together with the project.

See also

Overview of data type conversions (Page 2531)

Binary numbers (S7-300, S7-400)

Implicit conversion of BOOL (S7-300, S7-400)

Implicit conversion options

The BOOL data type cannot be implicitly converted.

See also

BOOL (bit) (Page 2243)

Bit strings (S7-300, S7-400)

Implicit conversion of BYTE (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of BYTE data type:

Source	Target	With IEC check	Without IEC check	Explanation
BYTE	BOOL	-	-	No implicit conversion
	WORD	X	X	The bit pattern of the source value is transferred unchanged right-justified to the target data type.
	DWORD	X	X	
	INT	-	-	No implicit conversion
	DINT	-	X	The bit pattern of the source value is transferred unchanged right-justified to the target data type.
	REAL	-	-	No implicit conversion
	TIME	-	-	
	S5TIME	-	-	
	DT	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	CHAR	-	X	The bit pattern of the source value is transferred unchanged to the target data type.
x: Conversion possible -: Conversion not possible				

See also

BYTE (Page 2244)

Overview of data type conversions (Page 2531)

Explicit conversion of BYTE (Page 2546)

Setting and disabling the IEC check (Page 2533)

Implicit conversion of WORD (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of WORD data type:

Source	Target	With IEC check	Without IEC check	Explanation
WORD	BOOL	-	-	No implicit conversion
	BYTE	-	X	Only the low byte is transferred to the destination data type.
	DWORD	X	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.
	INT	-	X	
	DINT	-	X	
	REAL	-	-	No implicit conversion
	TIME	-	-	
	S5TIME	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	DT	-	-	No implicit conversion
	TOD	-	-	
	DATE	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	STRING	-	-	No implicit conversion
	CHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

WORD (Page 2245)
Overview of data type conversions (Page 2531)
Explicit conversion of WORD (Page 2548)
Setting and disabling the IEC check (Page 2533)

Implicit conversion of DWORD (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of DWORD data type:

Source	Target	With IEC check	Without IEC check	Explanation
DWORD	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	WORD	-	X	
	INT	-	-	No implicit conversion
	DINT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	REAL	-	-	No implicit conversion
	TIME	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	S5TIME	-	-	No implicit conversion
	DT	-	-	
	TOD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	DATE	-	-	No implicit conversion
	STRING	-	-	
	CHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

DWORD (Page 2245)

Overview of data type conversions (Page 2531)

Explicit conversion of DWORD (Page 2550)

Setting and disabling the IEC check (Page 2533)

Integers (S7-300, S7-400)

Implicit conversion of INT (S7-300, S7-400)

Options for implicit conversion

The following table shows the options for the implicit conversion of INT data type:

Source	Target	With IEC check	Without IEC check	Explanation
INT	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	DWORD	-	-	No implicit conversion
	DINT [1)]	X	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	REAL	X	X	
	TIME	-	-	No implicit conversion
	S5TIME	-	-	
	DT	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	CHAR	-	-	

x: Conversion possible
-: Conversion not possible
[1)]: Only possible in SCL

See also

INT (16-bit integers) (Page 2250)
Overview of data type conversions (Page 2531)
Explicit conversion of INT (Page 2553)
Setting and disabling the IEC check (Page 2533)

Implicit conversion of DINT (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of DINT data type:

Source	Target	With IEC check	Without IEC check	Explanation
DINT	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	INT	-	-	No implicit conversion
	REAL	-	-	
	TIME	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	S5TIME	-	-	No implicit conversion
	DT	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	CHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

DINT (32-bit integers) (Page 2252)

Overview of data type conversions (Page 2531)

Explicit conversion of STRING (Page 2566)

Setting and disabling the IEC check (Page 2533)

Floating-point numbers (S7-300, S7-400)

Implicit conversion of REAL (S7-300, S7-400)

Implicit conversion options

The REAL data type cannot be implicitly converted.

See also

REAL (Page 2257)

Overview of data type conversions (Page 2531)

Explicit conversion of CHAR (Page 2565)

Setting and disabling the IEC check (Page 2533)

Timers (S7-300, S7-400)

Implicit conversion of TIME (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of TIME data type:

Source	Target	With IEC check	Without IEC check	Explanation
TIME	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion shows the duration in milliseconds.
	INT	-	-	No implicit conversion
	DINT	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion shows the duration in milliseconds.
	REAL	-	-	No implicit conversion
	S5TIME	-	-	
	DT	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	CHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

TIME (IEC time) (Page 2262)

Overview of data type conversions (Page 2531)

Explicit conversion of TIME (Page 2560)

Setting and disabling the IEC check (Page 2533)

Implicit conversion of S5TIME (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of S5TIME data type:

Source	Target	With IEC check	Without IEC check	Explanation
S5TIME	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion shows the duration in milliseconds.
	DWORD	-	-	No implicit conversion
	INT	-	-	
	DINT	-	-	
	REAL	-	-	
	TIME	-	-	
	DT	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	-	-	
	CHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

S5TIME (duration) (Page 2261)

Overview of data type conversions (Page 2531)

Explicit conversion of S5TIME (Page 2561)

Setting and disabling the IEC check (Page 2533)

Date and time (S7-300, S7-400)

Implicit conversion of DATE (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of DATE data type:

Source	Target	With IEC check	Without IEC check	Explanation
DATE	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion corresponds to the number of days since 01/01/1990.
	DWORD	-	-	No implicit conversion
	INT	-	-	
	DINT	-	-	
	REAL	-	-	
	TIME	-	-	
	S5TIME	-	-	
	DT	-	-	
	TOD	-	-	
	STRING	-	-	
	CHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

DATE (Page 2264)
Overview of data type conversions (Page 2531)
Setting and disabling the IEC check (Page 2533)

Implicit conversion of TOD (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of TOD data type:

Source	Target	With IEC check	Without IEC check	Explanation
TOD	BOOL	-	-	No implicit conversion
	BYTE	-	-	
	WORD	-	-	
	DWORD	-	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of the conversion corresponds to the number of milliseconds since the start of day (0:00).
	INT	-	-	No implicit conversion
	DINT	-	-	
	REAL	-	-	
	TIME	-	-	
	S5TIME	-	-	
	DT	-	-	
	DATE	-	-	
	STRING	-	-	
	CHAR	-	-	
x: Conversion possible -: Conversion not possible				

See also

TIME_OF_DAY (TOD) (Page 2264)

Overview of data type conversions (Page 2531)

Setting and disabling the IEC check (Page 2533)

Implicit conversion of DT (S7-300, S7-400)

Implicit conversion options

The DT data type cannot be implicitly converted.

See also

DATE_AND_TIME (date and time of day) (Page 2265)

Overview of data type conversions (Page 2531)

Explicit conversion of DT (Page 2564)

Setting and disabling the IEC check (Page 2533)

Character strings (S7-300, S7-400)

Implicit conversion of CHAR (S7-300, S7-400)

Implicit conversion options

The following table shows the options for the implicit conversion of CHAR data type:

Source	Target	With IEC check	Without IEC check	Explanation
CHAR	BOOL	-	-	No implicit conversion
	BYTE	-	X	The bit pattern of the source value is transferred unchanged to the destination data type.
	WORD	-	-	No implicit conversion
	DWORD	-	-	
	INT	-	-	
	DINT	-	-	
	REAL	-	-	
	TIME	-	-	
	S5TIME	-	-	
	DT	-	-	
	TOD	-	-	
	DATE	-	-	
	STRING	X	X	The bit pattern of the source value is transferred unchanged to the destination data type.
x: Conversion possible -: Conversion not possible				

See also

CHAR (Page 2269)

Overview of data type conversions (Page 2531)

Explicit conversion of CHAR (Page 2565)

Setting and disabling the IEC check (Page 2533)

Implicit conversion of STRING (S7-300, S7-400)

Implicit conversion options

The STRING data type cannot be implicitly converted.

See also

STRING (Page 2270)

Overview of data type conversions (Page 2531)

Explicit conversion of STRING (Page 2566)

Setting and disabling the IEC check (Page 2533)

11.3.17.3 Explicit conversions (S7-300, S7-400)

Binary numbers (S7-300, S7-400)

Explicit conversion of BOOL (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the BOOL data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
BOOL	BYTE	X	Only the LSB (Least Significant Bit) is set in the destination data type. The enable output ENO is always "1".	CONVERT
	WORD	X		
	DWORD	X		
	INT	X		BOOL_TO_INT
	DINT	X		BOOL_TO_DINT
	REAL	-	No explicit conversion	-
	TIME	-		
	S5TIME	-		
	DT	-		
	TOD	-		
	DATE	-		
	STRING	-		
	CHAR	-		
x: Conversion possible - : Conversion not possible				

See also

Overview of data type conversions (Page 2531)

BOOL (bit) (Page 2243)

Implicit conversions (Page 2533)

Bit strings (S7-300, S7-400)

Explicit conversion of BYTE (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the BYTE data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
BYTE	BOOL	X	The following possibilities can occur: • If the source is "0", the target data type is also "0" and the enable output ENO "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the target data type is also "1" and the enable output ENO "1". • If bits are not equal to LSB in the source value, the target data type is set according to LSB and the enable output ENO is "0".	CONVERT
	WORD	X	The bit pattern of the source value is transferred unchanged right-justified to the target data type. The remaining bits are set to "0".	CONVERT
	DWORD	X		
	INT	X		
	DINT	X		
	REAL	-	No explicit conversion	-
	TIME	-		
	S5TIME	-		
	DT	-		
	TOD	-		
	DATE	-		
	STRING	-		
	CHAR	X	The bit pattern of the source value is transferred unchanged right-justified to the target data type. The remaining bits are set to "0".	CONVERT
x: Conversion possible - : Conversion not possible				

See also

BYTE (Page 2244)

Overview of data type conversions (Page 2531)

Implicit conversion of BYTE (Page 2535)

Explicit conversion of WORD (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the WORD data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
WORD	BOOL	X	The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO is "1". • If bit is not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	CONVERT
	BYTE	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If the permitted value range of the destination data type is violated, the enable output ENO is set to "0". The result of the conversion is invalid in such a case.	
	DWORD	X		
	INT	X		
	DINT	X		
	REAL	-	No explicit conversion	-
	TIME	-		
	S5TIME	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	MOVE
	DT	-	No explicit conversion	-
	TOD	-		
	DATE	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	MOVE
	STRING	-	No explicit conversion	-
	CHAR	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	WORD_TO_CHAR
	BLOCK_DB	X	The bit pattern of WORD is interpreted as a data block number.	WORD_TO_BLOCK_DB

Source	Target	Conversion	Explanation	Mnemonics of the instruction
WORD_BCD[1) 2)]	INT	X	If the permitted value range of the destination data type is violated or if there is an invalid tetrad in the A - F range, the enable output ENO is set to "0". The result of the conversion is invalid in such a case.	WORD_BCD_TO_INT
BCD[1) 2)]	INT	X		BCD_TO_INT

x: Conversion possible

- : Conversion not possible

[1)] The value to be converted has data type WORD and is accepted as a BCD-coded value between -999 and +999. The result is available after conversion as an integer (in binary notation) of the type INT.

[2)] Valid for SCL: If the BCD code contains a pseudo tetrad, the CPU reports a programming error and opens the organization block "OB121". If the organization block OB121 is not present, the CPU changes to STOP mode.

See also

WORD (Page 2245)

Overview of data type conversions (Page 2531)

Implicit conversion of WORD (Page 2536)

Explicit conversion of DWORD (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DWORD data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DWORD	BOOL	X	The following possibilities can occur: • If the source is "0", the target data type is also "0" and the enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO is "1". • If bit is not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	CONVERT
	BYTE	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	
	WORD	X		

	INT	X	ENO = TRUE
			#int1 := DWORD_TO_INT(16#FFFF_FFFF); // -1 bis
			#int1 := DWORD_TO_INT(16#FFFF_8000); // -32768
			#int1 := DWORD_TO_INT(16#0); // 0 bis
			#int1 := DWORD_TO_INT(16#0000_7FFF); // 32767
			
			ENO = FALSE
			#int1 := DWORD_TO_INT(16#FFFF_7FFF); // -32769
			#int1 := DWORD_TO_INT(16#8000_0000); // -2147483648
			#int1 := DWORD_TO_INT(16#8000); // 32768 bis
			#int1 := DWORD_TO_INT(16#7FFF_FFFF); // 2147483647	

	DINT	X	The bit pattern of the source value is transferred un-	MOVE
	REAL	X		

Source	Target	Conversion	Explanation	Mnemonics of the instruction
	TIME	X	changed, right-justified, to the target data type.	
	S5TIME	-	No explicit conversion	-
	DT	-		
	TOD	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	MOVE
	DATE	-	No explicit conversion	-
	STRING	-		
	CHAR	-		
DWORD_BCD[1) 2)]	DINT	X	If the permitted value range of the destination data type is violated or if there is an invalid tetrad in the A - F range, the enable output ENO is set to "0". The result of the conversion is invalid in such a case.	DWORD_BCD_TO_DINT
BCD[1) 2)]	DINT	X		BCD_TO_DINT

x: Conversion possible

- : Conversion not possible

[1)] The value to be converted has data type DWORD and is accepted as a BCD-coded value between -9999999 and +9999999. The result is available after conversion as an integer (in binary notation) of the type DINT.

[2)] Valid for SCL: If the BCD code contains a pseudo tetrad, the CPU reports a programming error and opens the organization block "OB121". If the organization block OB121 is not present, the CPU changes to STOP mode.

See also

DWORD (Page 2245)

Overview of data type conversions (Page 2531)

Implicit conversion of DWORD (Page 2537)

Integers (S7-300, S7-400)

Explicit conversion of INT (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the INT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
INT	BOOL	X	The value is first converted to WORD and then to BOOL. The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO is "1". • If bit is not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	INT_TO_BOOL
	BYTE	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. The enable output ENO is set to "0" if a negative value is converted to an unsigned destination data type or if overflow occurs.	CONVERT
	WORD	X		MOVE
	DWORD	X		
	DINT	X		CONVERT, ITD
	REAL	X	The value is converted into the format of the destination data type (the value "1", for example, is converted with the instruction "Convert value" (CONVERT) into the value "1.0").	CONVERT, SCALE, ITR
	TIME	-	No explicit conversion	-
	S5TIME	-		
	DT	-		
	TOD	-		
	DATE	-		
	STRING	X	The value is converted into a character string. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	S_CONV, CONVERT
	CHAR	X	The bit pattern of the source value is transferred unchanged to the destination data type. With the conversion of negative values or with an overflow, the enable output ENO will be set to "0".	CONVERT
	BCD[1) 2)]	X	If the permitted value range of the destination data type is violated or if there is an invalid tetrad in the A - F range, the enable output ENO is set to "0". The result of the conversion is invalid in such a case.	INT_TO_BCD
	BCD_WORD[1) 2)]	X		INT_TO_BCD_WORD

Source	Target	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] The value to be converted has type INT and is accepted as an integer with a value between -999 and +999. The result is available after conversion as a BCD-coded number of the type WORD. [2] Valid for SCL: If the BCD code contains a pseudo tetrad, the CPU reports a programming error and opens the organization block "OB121". If the organization block OB121 is not present, the CPU changes to STOP mode.				

See also

INT (16-bit integers) (Page 2250)

Overview of data type conversions (Page 2531)

Implicit conversion of INT (Page 2538)

Explicit conversion of DINT (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DINT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DINT	BOOL	X	The value is first converted to DWORD and then to BOOL. The following possibilities can occur: • If the source is "0", the destination data type is also "0" and the enable output ENO is "1". • If only the LSB (Least Significant Bit) "1" is set in the source value, the destination data type is also "1" and the enable output ENO is "1". • If bit is not equal to LSB in the source value, the destination data type is set according to LSB and the enable output ENO is "0".	DINT_TO_BOOL
	BYTE	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type. The enable output ENO is set to "0" if a negative value is converted to an unsigned destination data type or if overflow occurs.	CONVERT
	WORD	X		
	DWORD	X		MOVE
	INT	X		CONVERT
	REAL	X	The value is converted into the format of the destination data type (the value "1", for example, is converted with the instruction "Convert value" (CONVERT) into the value "1.0").	CONVERT, DTR, SCALE
	TIME	X	The bit pattern of the source value is transferred unchanged to the destination data type.	T_CONV, MOVE
	S5TIME	-	No explicit conversion	-
	DT	-		
	TOD	X	The bit pattern of the source value is transferred unchanged, right-justified, to the target data type.	CONVERT
	DATE	X		
	STRING	X	The value is converted into a character string. The character string is shown preceded by a sign. If the permitted length of the character string is violated, the enable output ENO is set to "0".	S_CONV, CONVERT
	CHAR	X	The bit pattern of the source value is transferred unchanged to the destination data type. With the conversion of negative values or with an overflow, the enable output ENO will be set to "0".	DINT_TO_CHAR
	BCD[1) 2)]	X	If the permitted value range of the destination data type is violated or if there is an invalid tetrad in the A - F range, the enable output ENO is set to "0". The result of the conversion is invalid in such a case.	DINT_TO_BCD
	BCD_DWORD[1) 2)]	X		DINT_TO_BCD_DWORD

Source	Target	Conversion	Explanation	Mnemonics of the instruction
x: Conversion possible - : Conversion not possible [1] The value to be converted has data type DWORD and is accepted as a BCD-coded value between -9999999 and +9999999. The result is available after conversion as an integer (in binary notation) of the type DINT. [2] Valid for SCL: If the BCD code contains a pseudo tetrad, the CPU reports a programming error and opens the organization block "OB121". If the organization block OB121 is not present, the CPU changes to STOP mode.				

See also

DINT (32-bit integers) (Page 2252)

Overview of data type conversions (Page 2531)

Implicit conversions (Page 2533)

Floating-point numbers (S7-300, S7-400)

Explicit conversion of REAL (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the REAL data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
REAL	BOOL	-	No explicit conversion	-
	BYTE	-		
	WORD	-		
	DWORD	X	The bit pattern of the source value is transferred unchanged to the destination data type.	CONVERT, MOVE
	INT	X	The value is converted into the destination data type. The result of conversion depends on the instruction employed. The enable output ENO is set to "0" if the admissible range of values of the destination data type is exceeded during conversion or the value to be converted has an invalid floating-point number.	CONVERT, ROUND, RND, CEIL, RND+, FLOOR, RND-, TRUNC, UNSCALE
	DINT	X		
	TIME	-	No explicit conversion	-
	S5TIME	-		
	DT	-		
	TOD	-		
	DATE	-		
	STRING	X	The value is converted into a character string. The enable output ENO is set to "0" if the character string length is exceeded or the value to be converted has an invalid floating-point number.	S_CONV, CONVERT
	CHAR	-	No explicit conversion	-
x: Conversion possible - : Conversion not possible				

See also

REAL (Page 2257)

Overview of data type conversions (Page 2531)

Implicit conversions (Page 2533)

Timer operations (S7-300, S7-400)

Explicit conversion of TIME (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the TIME data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
TIME	BOOL	-	No explicit conversion	-
	BYTE	-		
	WORD	-		
	DWORD	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion shows the duration in milliseconds.	CONVERT
	INT	-	No explicit conversion	-
	DINT	X	The bit pattern of the source value is transferred unchanged to the destination data type. The result of conversion shows the duration in milliseconds.	T_CONV, CONVERT
	REAL	-	No explicit conversion	-
	S5TIME	X	The value is converted to the destination data type format. The enable output ENO is set to "0" in the event of overflow.	T_CONV, CONVERT
	DT	-	No explicit conversion	-
	TOD	-		
	DATE	-		
	STRING	-		
	CHAR	-		
x: Conversion possible - : Conversion not possible				

See also

TIME (IEC time) (Page 2262)

Overview of data type conversions (Page 2531)

Implicit conversion of TIME (Page 2540)

Explicit conversion of S5TIME (S7-300, S7-400)

Options for explicit conversion in LAD, FBD, STL and GRAPH

The following table shows the options and instructions for the explicit conversion of the S5TIME data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
S5TIME	BOOL	-	No explicit conversion	-
	BYTE	-		
	WORD	X	The value is converted to the destination data type format.	S5TIME_TO_WORD
	DWORD	-	No explicit conversion	-
	INT	-		
	DINT	-		
	REAL	-		
	TIME	X	The value is converted to the destination data type format.	T_CONV, CONVERT
	DT	-	No explicit conversion	-
	TOD	-		
	DATE	-		
	STRING	-		
	CHAR	-		
x: Conversion possible - : Conversion not possible				

See also

S5TIME (duration) (Page 2261)

Overview of data type conversions (Page 2531)

Implicit conversion of S5TIME (Page 2541)

Date and time (S7-300, S7-400)

Explicit conversion of DATE (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DATE data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DATE	BOOL	-	No explicit conversion	-
	BYTE	-		
	WORD	X	The value is converted to the destination data type format.	DATE_TO_WORD
	DWORD	-	No explicit conversion	-
	INT	X	The value is converted to the destination data type format.	DATE_TO_INT
	DINT	X		DATE_TO_DINT
	REAL	-	No explicit conversion	-
	TIME	-		
	S5TIME	-		
	DT	-		
	TOD	-		
	STRING	-		
	CHAR	-		
x: Conversion possible - : Conversion not possible				

See also

DATE (Page 2264)

Overview of data type conversions (Page 2531)

Implicit conversion of DATE (Page 2542)

Explicit conversion of TOD (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the TOD data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
TOD	BOOL	-	No explicit conversion	-
	BYTE	-		
	WORD	-		
	DWORD	X	The value is converted to the destination data type format.	TOD_TO_DWORD
	INT	-	No explicit conversion	-
	DINT	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	TOD_TO_DINT
	REAL	-	No explicit conversion	-
	TIME	-		
	S5TIME	-		
	DT	-		
	DATE	-		
	STRING	-		
	CHAR	-		
x: Conversion possible - : Conversion not possible				

See also

TIME_OF_DAY (TOD) (Page 2264)

Overview of data type conversions (Page 2531)

Implicit conversion of TOD (Page 2543)

Explicit conversion of DT (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the DT data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
DT	BOOL	-	No explicit conversion	-
	BYTE	-		
	WORD	-		
	DWORD	-		
	INT	-		
	DINT	-		
	REAL	-		
	TIME	-		
	S5TIME	-		
	TOD	X	During conversion, times are extracted from the DTL format and transferred to the destination data type.	T_CONV, CONVERT
	DATE	X	During the conversion, the date information is extracted from the DTL format and transferred to the destination data type.	
	STRING	-	No explicit conversion	-
	CHAR	-		
x: Conversion possible - : Conversion not possible				

See also

DATE_AND_TIME (date and time of day) (Page 2265)

Overview of data type conversions (Page 2531)

String (S7-300, S7-400)

Explicit conversion of CHAR (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the CHAR data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
CHAR	BOOL	-	No explicit conversion	-
	BYTE	X	The bit pattern of the source value is transferred unchanged right-justified to the destination data type.	CONVERT
	WORD	X		
	DWORD	X		
	INT	X		
	DINT	X		
	REAL	-	No explicit conversion	-
	TIME	-		
	S5TIME	-		
	DT	-		
	TOD	-		
	DATE	-		
	STRING	X	The value is converted into the first character in the character string (STRING). The length "1" is set after conversion if the character string length is not defined. If the character string length is defined, it remains unchanged following conversion.	S_CONV, CONVERT
x: Conversion possible - : Conversion not possible				

See also

CHAR (Page 2269)

Overview of data type conversions (Page 2531)

Implicit conversion of CHAR (Page 2544)

Explicit conversion of STRING (S7-300, S7-400)

Options for explicit conversion

The following table shows the options and instructions for the explicit conversion of the STRING data type:

Source	Target	Conversion	Explanation	Mnemonics of the instruction
STRING	BOOL	-	No explicit conversion	-
	BYTE	-		
	WORD	-		
	DWORD	-		
	INT	X	Conversion begins with the first character in the character string (STRING) and stops at the end of the string or at the first inadmissible character. The following characters are admissible for conversion: • Digit • Sign • Dot The first character in the character string may be a sign (+, -) or a digit. Leading spaces are ignored. The dot is used as separation for the conversion of floating-point numbers. The exponential notation "e" or "E" is not permitted. The comma as thousand separator to the left of the decimal point is permitted but is ignored. If the layout of the character string is invalid for the conversion or an overflow occurs, then the enable output ENO will be set to "0".	S_CONV, CONVERT
	DINT	X		
	REAL	X		
	TIME	-	No explicit conversion	-
	S5TIME	-		
	DT	-		
	TOD	-		
	DATE	-		
	CHAR	X	The first character in the character string (STRING) is transferred to the target data type. The value "0" is written to the target data type if the character string is empty.	S_CONV, CONVERT
x: Conversion possible - : Conversion not possible				

See also

STRING (Page 2270)

Overview of data type conversions (Page 2531)

Implicit conversions (Page 2533)

Additional conversion functions (S7-300, S7-400)

Additional explicit conversion functions in SCL (S7-300, S7-400)

Additional options for explicit conversion in SCL

The following table shows the additional options and operations for explicit conversion in SCL:

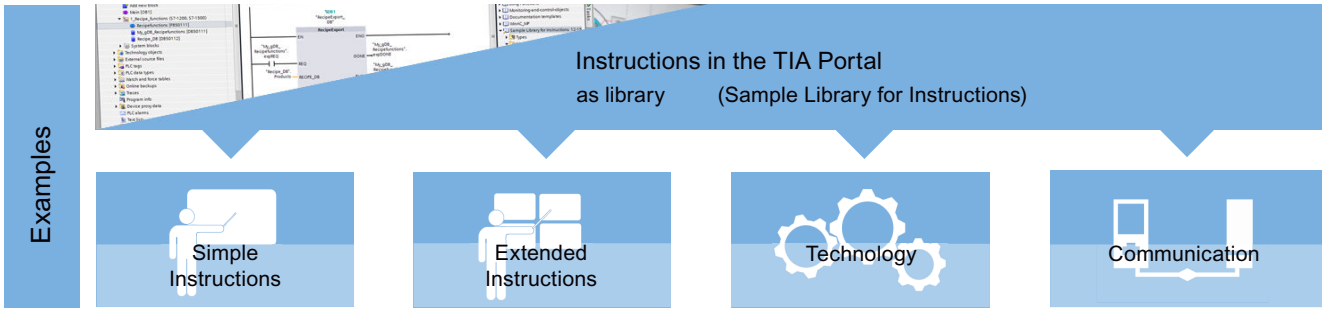
Source	Target	Explanation	Mnemonics of the instruction
WORD	BLOCK_DB	The bit pattern of WORD is interpreted as a data block number.	WORD_TO_BLOCK_DB
BLOCK_DB	WORD	The data block number is interpreted as a bit pattern of WORD.	BLOCK_DB_TO_WORD

11.4 Instructions (S7-1200, S7-1500)

11.4.1 Example libraries (S7-1200, S7-1500)

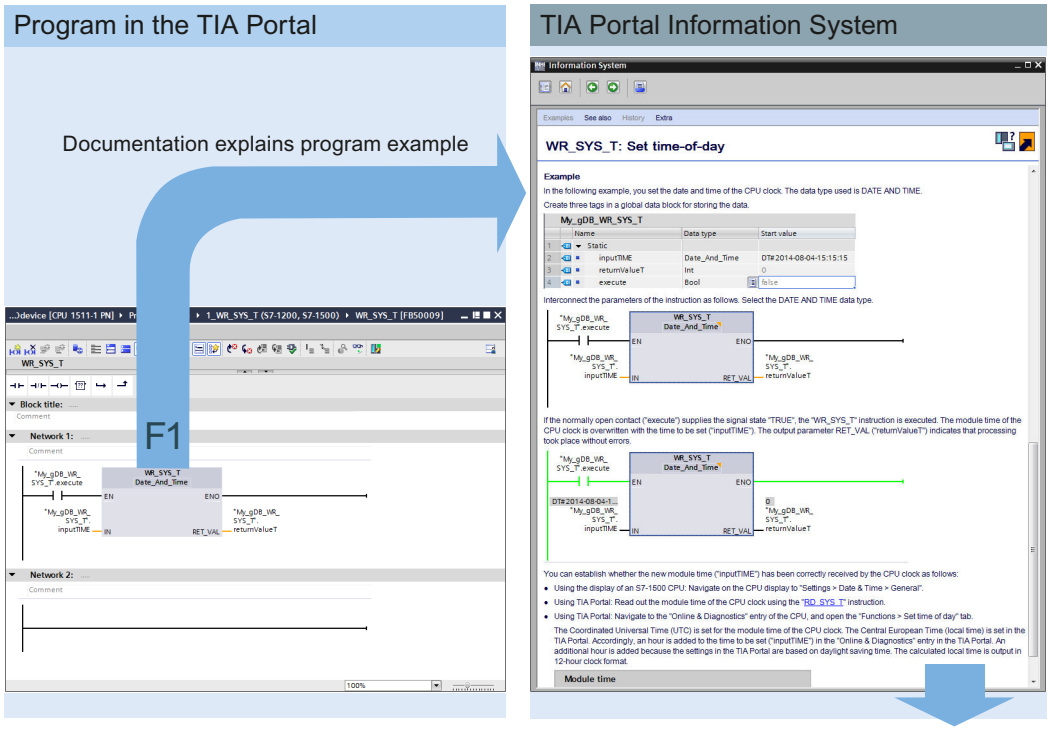
11.4.1.1 Sample Library for Instructions (S7-1200, S7-1500)

The "Sample Library for Instructions" is the global library for the program examples using the instructions in STEP 7 (TIA Portal). This library offers you support during programming with its compact, easy to understand program examples in the programming language LAD. The precondition for using the library is at least one CPU from the SIMATIC S7-1200 / S7-1500 series.



Programming examples in the TIA Portal information system

Programming examples in the "Sample Library for Instructions" match the instructions used in the programming examples in the TIA Portal information system. This means that you can call the respective programming example documentation via the help (<F1>) in the TIA Portal.



Library contains program example from the documentation

Downloading the library

The "Sample Library for Instructions" is available at the FAQ page (https://support.industry.siemens.com/cs/ww/en/view/109476781).

Note

Adding program examples to a project correctly

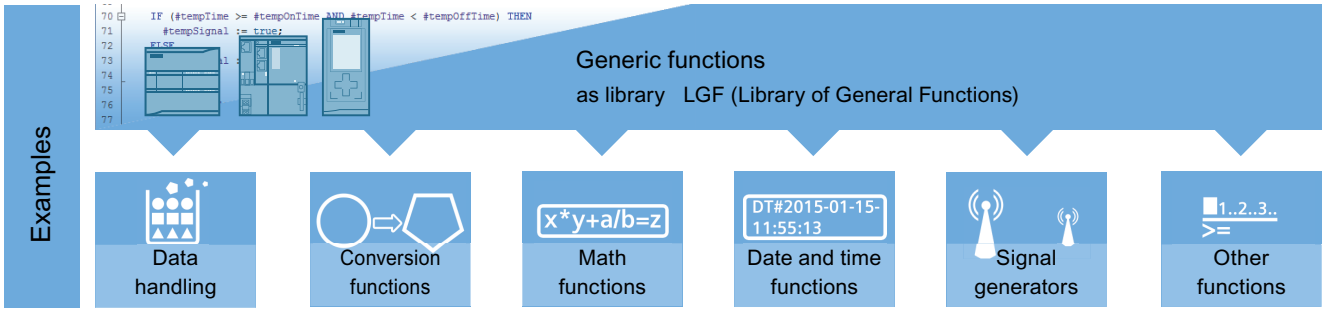
Due to the differing preconditions and functions, program examples may contain not only program blocks but also other components (e.g. PLC tags, PLC data types, watch tables or the like). There are also program examples that require special hardware (multiple CPUs, distributed I/O, input modules and the like).

Please observe the following rules:

- Never simply add all program examples or the entire "Sample Library for Instructions" folder to a CPU.
- Make sure that the corresponding program example does not consist of components for multiple CPUs.
- Always add the individual components of the program examples one at a time to the appropriate project folder. Also make sure that you select the appropriate CPU.

11.4.1.2 Library of General Functions (S7-1200, S7-1500)

The "Library of General Functions" is the global library for supplementary functions for the instructions in STEP 7 (TIA Portal). To expand the scope of instructions, this library offers blocks with useful basic functions that are often required in automation projects. All contents of the library can be used with a CPU of the series SIMATIC S7-1200(F) / S7-1500(F).



General functions

The general functions of the "Library of General Functions" are provided as blocks in the SCL programming language. The library includes the following functions: FIFO, search function, matrix calculations, astro clock. You can immediately use the functions by means of parameter assignment and implement them universally.

Downloading the library

The "Library of General Functions" is stored on the FAQ page (https://support.industry.siemens.com/cs/ww/en/view/109479728).

11.4.2 Asynchronous instructions (S7-1200, S7-1500)

11.4.2.1 Difference between synchronous and asynchronous instructions (S7-1200, S7-1500)

Asynchronous instructions

In program processing, a differentiation is made between synchronous and asynchronous instructions.

The "synchronous" and "asynchronous" properties relate to the temporal relationship between the call and execution of the instruction.

- The following applies to synchronous instructions: When the call of a synchronous instruction is ended, the execution is also ended.
- This is different in the case of asynchronous instructions: The execution of an asynchronous instruction can extend over multiple calls. The CPU processes asynchronous instructions in parallel with the cyclic user program. Asynchronous instructions occupy resources in the CPU while they are being processed.

Asynchronous instructions are usually instructions for transferring data (data records for modules, communication data, diagnostics data).

Additional information about asynchronous instructions can be found here: SIMATIC S7-1500 / ET 200MP Manual Collection (https://support.industry.siemens.com/cs/ww/en/view/86140384) in the system manual under Basics of program processing > Asynchronous instructions

Identification of the job

To execute an instruction over multiple calls, the CPU must be able to uniquely assign a subsequent call to a running job of the instruction. To assign a call to a job, the CPU uses one of the following two mechanisms, depending on the type of the instruction:

- The instance of the instruction (with type "SFB" system function block)
- The input parameter of the instruction identifying the job

If you use asynchronous instructions to trigger a process interrupt, output control commands to DP slaves, start a data transfer, or abort a non-configured connection and then call the instruction again before the current job is completed, the reaction of the instruction will depend on whether the second call involves the same job.

Passing parameters to program blocks that are to be executed asynchronously

Code blocks (FBs/FCs) and data blocks (DBs) can be created with different access types ("standard" and "optimized"). In code blocks, you can call any instructions. Some instructions (e.g. "WRIT_DBL" and "READ_DBL") are executed asynchronously. These blocks cannot be supplied with tags from TEMP because the data must not be changed during execution.

Ensure that you do not use these instructions in programs in which code blocks of different access types are called reciprocally. This could cause the following to occur:

- A structure from a standard data block is directly or indirectly passed to an optimized code block, which forwards this structure directly or indirectly to one of the blocks mentioned above.
- The reverse scenario, whereby a structure from an optimized code block is directly or indirectly passed to a standard data block, which forwards this structure directly or indirectly to one of the program blocks mentioned above.

Otherwise, a hidden copy of the passed data is created in TEMP, which leads to the asynchronous program blocks returning a negative acknowledgment.

Parameter REQ

The input parameter REQ (request) is used solely to start the job:

- Trigger the job by setting the input parameter REQ to "1" (case 1).
- If a particular job has been started and not yet completed and you call the instruction again to perform the same job (for example, in a cyclic interrupt OB), then REQ is not evaluated by the instruction (case 2).
- If the job is completed, but the input parameter REQ is still set to "1", the job is immediately started again.

Note

REQ bit = "1"

Please note that, when the REQ bit is set, the asynchronous instruction will be started again after the previous call is completed. This may not be desired. To make the project clearer and easier to maintain, it is therefore advisable to reset the REQ bit to "0" as early as possible.

Parameter RET_VAL and BUSY

The output parameters RET_VAL and BUSY indicate the status of the job.

Pay attention to the note in section: AUTOHOTSPOT

- In case 1 (first call with REQ = 1), code W#16#7001 will be entered in RET_VAL and BUSY will be set to "1" if system resources are available and the supply of the input parameters is correct.
  If the required system resources are currently being used or the input parameters contain an error, the corresponding error code is entered in RET_VAL and BUSY receives the value "0".

  Note

  Call with REQ = 0 when system resources are assigned

  If an asynchronous instruction with REQ = 0 is called for an S7-1500 CPU, all available resources are assigned at this time and this call not an intermediate call of a job that is already running, the error code W#16#80C3 (temporary resource error) is entered in RET_VAL.

  For an S7-300 CPU, the error code W#16#7000 (first call with REQ = 0) is entered in RET_VAL under these boundary conditions.

- In case 2 (intermediate call), the code W#16#7002 is entered in RET_VAL (this corresponds to the message that the call is currently still being processed) and BUSY is set to "1".
- The following applies to the last call for a job:
  - For instruction "DPNRM_DG (Page 3715)", the number of data in bytes will entered as integer in RET_VAL in case there are no errors in data transmission. BUSY has the value "0" in this case.
    If there is an error, then the error information will be entered in RET_VAL and you should not evaluate BUSY in this case.
  - For all other instructions, "0" will be entered in RET_VAL if the job was executed without errors and BUSY will receive the value "0". If there is an error, the error code will be entered in RET_VAL and BUSY will receive the value "0".

Note

If the first and last call coincide, the reaction for RET_VAL and BUSY is the same as for the last call.

Overview

The following table provides you with an overview of the relationships explained above. In particular, it shows the possible values of the output parameters if the execution of the job is not completed after an instruction call has been completed.

Note

Following every call, you must evaluate the relevant output parameters in your program.

Relationship between call, REQ, RET_VAL and BUSY during a "running" job.

Number of the call	Type of call	REQ	RET_VAL	BUSY	DONE	ERROR
1	First call	1	W#16#7001	1	0	0
			Error code	0	0	1
2 to (n - 1)	Intermediate call	irrelevant	W#16#7002	1	0	0
n	Last call	irrelevant	W#16#0000 (exceptions: RD_REC (Page 3694) and RD_DPARA (Page 3789): The successfully read data length is written to the parameter STATUS.)	0	1	0
			Error code if errors occurred	0	0	1

11.4.3 Basic instructions (S7-1200, S7-1500)

11.4.3.1 LAD (S7-1200, S7-1500)

Bit logic operations (S7-1200, S7-1500)

---| |---: Normally open contact (S7-1200, S7-1500)

Description

The activation of the normally open contact depends on the signal state of the associated operand. When the operand has signal state "1", the normally open contact closes and the signal state at the output is set to the signal state of the input.

When the operand has signal state "0", the normally open contact is not activated and the signal state at the output of the instruction is reset to "0".

Two or more normally open contacts are linked bit-by-bit by AND when connected in series. With a series connection, power flows when all contacts are closed.

The normally open contacts are linked by OR when connected in parallel. With a parallel connection, power flows when one of the contacts is closed.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Operand whose signal state is queried.

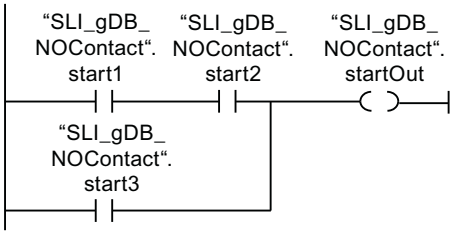
Example

The following example shows how the instruction works.

Create a global data block with the following content for this purpose:

Name of the block: SLI_gDB_NOContact	
Name	**Data type**
start1	BOOL
start2	BOOL
start3	BOOL
startOut	BOOL

Write the following program code:



Operand "SLI_gDB_NOContact".startOut is set when one of the following conditions is fulfilled:

- The operands "SLI_gDB_NOContact".start1 and "SLI_gDB_NOContact".start2 have the signal state "1".
- The operand "SLI_gDB_NOContact".start3 has the signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

---| / |---: Normally closed contact (S7-1200, S7-1500)

Description

The activation of the normally closed contact depends on the signal state of the associated operand. When the operand has signal state "1", the normally closed contact opens and the signal state at the output of the instruction is reset to "0".

When the operand has signal state "0", the normally closed contact is not enabled and the signal state of the input is transferred to the output.

Two or more normally closed contacts are linked bit-by-bit by AND when connected in series. With a series connection, power flows when all contacts are closed.

The normally closed contacts are linked by OR when connected in parallel. With a parallel connection, power flows when one of the contacts is closed.

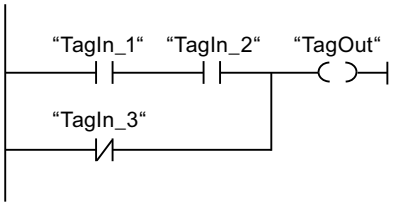
Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Operand whose signal state is queried.

Example

The following example shows how the instruction works:



The "TagOut" operand is set when one of the following conditions is fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The operand "TagIn_3" has the signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

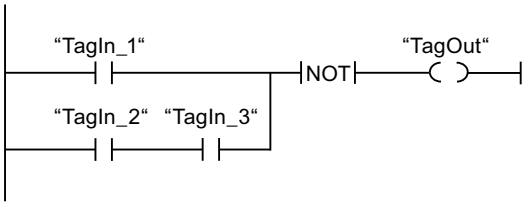
--|NOT|--: Invert RLO (S7-1200, S7-1500)

Description

You use the "Invert RLO" instruction to invert the signal state of the result of logic operation (RLO). If the signal state is "1" at the input of the instruction, the output of the instruction has signal state "0". If the signal state is "0" at the input of the instruction, the output has the signal state "1".

Example

The following example shows how the instruction works:



Operand "TagOut" is reset when one of the following conditions is fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The signal state of the operands "TagIn_2" and "TagIn_3" is "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

---( )---: Assignment (S7-1200, S7-1500)

Description

You can use the "Assignment" instruction to set the bit of a specified operand. If the result of logic operation (RLO) at the input of the coil has signal state "1", the specified operand is set to signal state "1". If the signal state is "0" at the input of the coil, the bit of the specified operand is reset to "0".

The instruction does not influence the RLO. The RLO at the input of the coil is sent directly to the output.

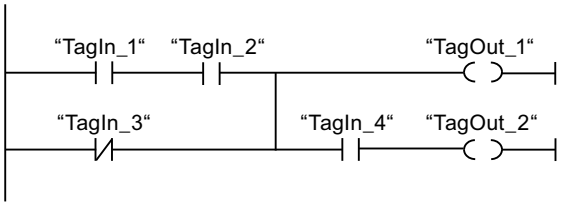
Parameter

The following table shows the parameters of the "Assignment" instruction:

Parameter	Declaration	Data type	Memory area	Description
<Operand>	Output	BOOL	I, Q, M, D, L	Operand to which the RLO is assigned.

Example

The following example shows how the instruction works:



The "TagOut_1" operand is set when one of the following conditions is fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state of the operand "TagIn_3" is "0".

The "TagOut_2" operand is set when one of the following conditions is fulfilled:

- Operands "TagIn_1", "TagIn_2", and "TagIn_4" have signal state "1".
- The signal state of the "TagIn_3" operand is "0" and the signal state of the "TagIn_4" operand is "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

--(/)--: Negate assignment (S7-1200, S7-1500)

Description

The "Negate assignment" instruction inverts the result of logic operation (RLO) and assigns it to the specified operand. When the RLO at the input of the coil is "1", the operand is reset. When the RLO at the input of the coil is "0", the operand is set to signal state "1".

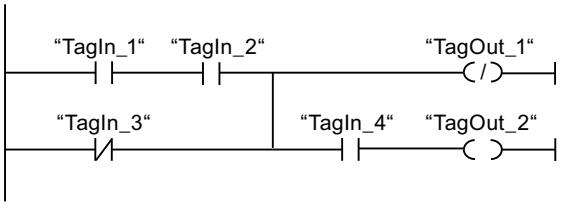
Parameter

The following table shows the parameters of the "Negate assignment" instruction:

Parameter	Declaration	Data type	Memory area	Description
<Operand>	Output	BOOL	I, Q, M, D, L	Operand to which the RLO is assigned.

Example

The following example shows how the instruction works:



Operand "TagOut_1" is reset when one of the following conditions is fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state of the operand "TagIn_3" is "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

---(R)---: Reset output (S7-1200, S7-1500)

Description

You can use the "Reset output" instruction to reset the signal state of a specified operand to "0".

The instruction is only executed if the result of logic operation (RLO) at the input of the coil is "1". If power flows to the coil (RLO = "1"), the specified operand is reset to "0". If the RLO at the input of the coil is "0" (no signal flow to the coil), the signal state of the specified operand remains unchanged.

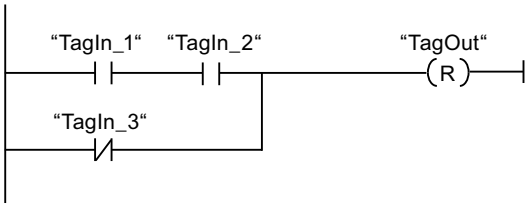
Parameter

The following table shows the parameters of the "Reset output" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand>	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Operand that is reset when RLO = "1".

Example

The following example shows how the instruction works:



Operand "TagOut" is reset when one of the following conditions is fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state of the operand "TagIn_3" is "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

---(S)---: Set output (S7-1200, S7-1500)

Description

You can use the "Set output" instruction to set the signal state of a specified operand to "1".

The instruction is only executed if the result of logic operation (RLO) at the input of the coil is "1". If power flows to the coil (RLO = "1"), the specified operand is set to "1". If the RLO at the input of the coil is "0" (no signal flow to the coil), the signal state of the specified operand remains unchanged.

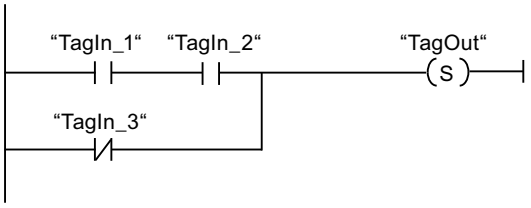
Parameter

The following table shows the parameters of the "Set output" instruction:

Parameter	Declaration	Data type	Memory area	Description
<Operand>	Output	BOOL	I, Q, M, D, L	Operand which is set with RLO = "1".

Example

The following example shows how the instruction works:



The "TagOut" operand is set when one of the following conditions is fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state of the operand "TagIn_3" is "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

SET_BF: Set bit field (S7-1200, S7-1500)

Description

You use the "Set bit field" instruction to set several bits starting from a certain address.

You determine the number of bits to be set using the value of <Operand1>. The address of the first bit to be set is defined by <Operand2>. If the value of <Operand1> may not be greater than the number of bits in a selected byte. If this value is greater, the instruction is not executed and the error message "Range violation for index <Operand1>" is displayed. The bits remain set until they are explicitly reset by another instruction.

Specify <Operand1> at the operand placeholder below the instruction. Specify <Operand2> at the operand placeholder above the instruction.

The instruction is only executed if the result of logic operation (RLO) at the input of the coil is "1". If the RLO at the input of the coil is "0", the instruction does not execute.

Bit fields of the type PLC data type, STRUCT or ARRAY

With structures of the type PLC data type, STRUCT or ARRAY, the number of bits contained in the structure represents the maximum number of bits that can be reset:

- If you specify the value "20" at <Operand1>, for example, and the structure only contains 10 bits, then only these 10 bits are set.
- If you specify the value "5" at <Operand1>, for example, and the structure contains 10 bits, then exactly 5 bits are set.

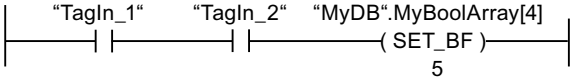
Parameter

The following table shows the parameters of the "Set bit field" instruction:

Parameter	Declaration	Data type	Memory area	Description
<Operand1>	Input	UINT	Constant	Number of bits to be set
<Operand2>	Output	BOOL	I, Q, M In the case of a DB or an IDB, an element of an ARRAY[..] of BOOL	Pointer to the first bit to be set.

Example

The following example shows how the instruction works:



If the operands "TagIn_1" and "TagIn_2" have the signal state "1", 5 bits are set starting at the address of the operand "MyDB".MyBoolArray[4].

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

RESET_BF: Reset bit field (S7-1200, S7-1500)

Description

You use the "Reset bit field" instruction to reset several bits starting from a certain address.

You specify the number of bits to be reset using the value of <Operand1>. The address of the first bit to be reset is defined by <Operand2>. If the value of <Operand 1> may not be greater than the number of bits in a selected byte. If this value is greater, the instruction is not executed

and the error message "Range violation for index <Operand 1>" is displayed. The bits remain set until they are explicitly reset by another instruction.

Specify <Operand1> at the operand placeholder below the instruction. Specify <Operand2> at the operand placeholder above the instruction.

The instruction is only executed if the result of logic operation (RLO) at the input of the coil is "1". If the RLO at the input of the coil is "0", the instruction does not execute.

Bit fields of the type PLC data type, STRUCT or ARRAY

With structures of the type PLC data type, STRUCT or ARRAY, the number of bits contained in the structure represents the maximum number of bits that can be set:

- If you specify the value "20" at <Operand1>, for example, and the structure only contains 10 bits, then only these 10 bits are reset.
- If you specify the value "5" at <Operand1>, for example, and the structure contains 10 bits, then exactly 5 bits are reset.

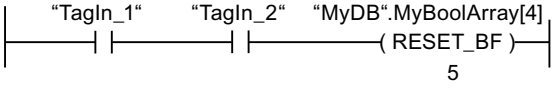
Parameter

The following table shows the parameters of the "Reset bit field" instruction:

Parameter	Declaration	Data type	Memory area	Description
<Operand1>	Input	UINT	Constant	Number of bits to be reset
<Operand2>	Output	BOOL	I, Q, M With a DB or an instance DB, an element of an ARRAY[..] of BOOL	Pointer to the first bit to be reset.

Example

The following example shows how the instruction works:



If the operands "TagIn_1" and "TagIn_2" have the signal state "1", 5 bits are reset starting at the address of the operand "MyDB".MyBoolArray[4]

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

SR: Set/reset flip-flop (S7-1200, S7-1500)

Description

Use the instruction "Set/reset flip-flop" to set or reset the bit of the specified operand, depending on the signal state of the inputs S and R1. If the signal state is "1" at input S and "0" at input R1, the specified operand is set to "1". If the signal state is "0" at input S and "1" at input R1, the specified operand will be reset to "0".

Input R1 takes priority over input S. When the signal state is "1" on both inputs S and R1, the signal state of the specified operand is reset to "0".

The instruction is not executed if the signal state at the two inputs S and R1 is "0". The signal state of the operand then remains unchanged.

The current signal state of the operand is transferred to output Q and can be queried there.

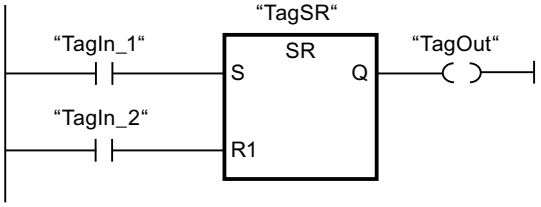
Parameter

The following table shows the parameters of the "Set/reset flip-flop" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
S	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable setting
R1	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable resetting
<Operand>	InOut	BOOL	I, Q, M, D, L	I, Q, M, D, L	Operand that is set or reset.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Signal state of the operand

Example

The following example shows how the instruction works:



The operands "TagSR" and "TagOut" are set when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The operand "TagIn_2" has the signal state "0".

The operands "TagSR" and "TagOut" are reset when one of the following conditions is fulfilled:

- The operand "TagIn_1" has signal state "0" and the operand "TagIn_2" has signal state "1".
- The operands "TagIn_1" and "TagIn_2" have signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

RS: Reset/set flip-flop (S7-1200, S7-1500)

Description

You can use the "Reset/set flip-flop" instruction to reset or set the bit of a specified operand based on the signal state of the inputs R and S1. If the signal state is "1" at input R and "0" at input S1, the specified operand will be reset to "0". If the signal state is "0" at input R and "1" at input S1, the specified operand is set to "1".

Input S1 takes priority over input R. When the signal state is "1" at both inputs R and S1, the signal state of the specified operand is set to "1".

The instruction is not executed if the signal state at the two inputs R and S1 is "0". The signal state of the operand then remains unchanged.

The current signal state of the operand is transferred to output Q and can be queried there.

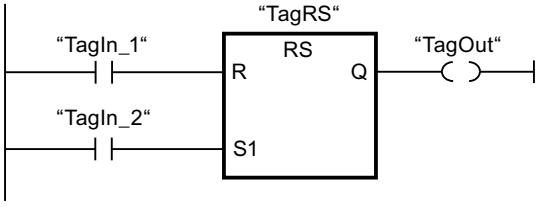
Parameter

The following table shows the parameters of the "Reset/set flip-flop" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
R	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable resetting
S1	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable setting
<Operand>	InOut	BOOL	I, Q, M, D, L	I, Q, M, D, L	Operand that is reset or set.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Signal state of the operand

Example

The following example shows how the instruction works:



The operands "TagRS" and "TagOut" are reset when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The operand "TagIn_2" has the signal state "0".

The operands "TagRS" and "TagOut" are set when one of the following conditions is fulfilled:

- The operand "TagIn_1" has signal state "0" and the operand "TagIn_2" has signal state "1".
- The operands "TagIn_1" and "TagIn_2" have signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

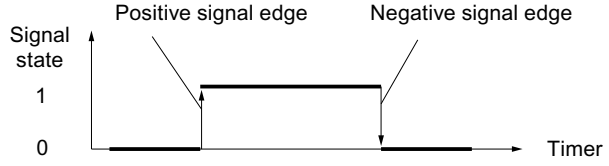
Basic information on LAD (Page 4878)

--|P|--: Scan operand for positive signal edge (S7-1200, S7-1500)

Description

You can use the "Scan operand for positive signal edge" instruction to determine if there is a "0" to "1" change in the signal state of a specified operand (<Operand1>). The instruction compares the current signal state of <Operand1> with the signal state of the previous scan, which is saved in an edge memory bit (<Operand2>). If the instruction detects a change in the result of logic operation (RLO) from "0" to "1", there is a positive, rising edge.

The following figure shows the change in the signal state in case of a negative and a positive signal edge:



The positive signal edge is queried each time the instruction executes. When a positive signal edge is detected, <Operand1> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

Specify the operand to be queried (<Operand1>) in the operand placeholder above the instruction. Specify the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This step influences the edge evaluation and the result is therefore no longer unique. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

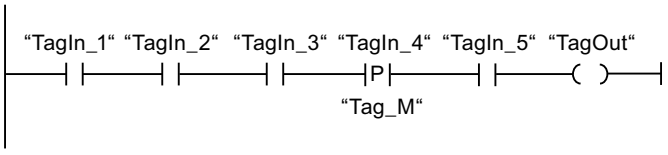
Parameter

The following table shows the parameters of the "Scan operand for positive signal edge" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand1>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Signal to be scanned
<Operand2>	InOut	BOOL	I, Q, M, D, L	I, Q, M, D, L	Edge memory bit in which the signal state of the previous query is saved.

Example

The following example shows how the instruction works:



Operand "TagOut" is set when the following conditions are fulfilled:

- The operands "TagIn_1", "TagIn_2", and "TagIn_3" have signal state "1".
- There is a rising edge at operand "TagIn_4". The signal state of the previous scan is stored in the edge memory bit "Tag_M".
- The signal state of the operand "TagIn_5" is "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

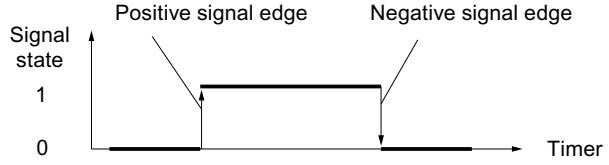
Basic information on LAD (Page 4878)

--|N|--: Scan operand for negative signal edge (S7-1200, S7-1500)

Description

You can use the "Scan operand for negative signal edge" instruction to determine if there is a "1" to "0" change in the signal state of a specified operand (<Operand1>). The instruction compares the current signal state of <Operand1> with the signal state of the previous scan that is saved in an edge memory bit <Operand2>. If the instruction detects a change in the result of logic operation (RLO) from "1" to "0", there is a negative, falling edge.

The following figure shows the change in the signal state in case of a negative and a positive signal edge:



The negative signal edge is queried each time the instruction executes. When a negative signal edge is detected, <Operand1> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

Specify the operand to be queried (<Operand1>) in the operand placeholder above the instruction. Specify the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This step influences the edge evaluation and the result is therefore no longer unique. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

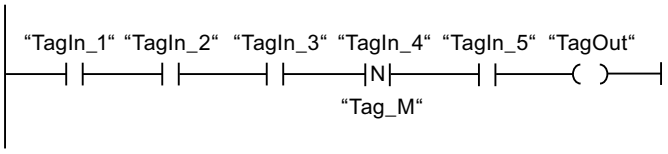
Parameter

The following table shows the parameters of the "Scan operand for negative signal edge" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand1>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Signal to be scanned
<Operand2>	InOut	BOOL	I, Q, M, D, L	I, Q, M, D, L	Edge memory bit in which the signal state of the previous query is saved.

Example

The following example shows how the instruction works:



Operand "TagOut" is set when the following conditions are fulfilled:

- The operands "TagIn_1", "TagIn_2", and "TagIn_3" have signal state "1".
- There is a negative signal edge at operand "TagIn_4". The signal state of the previous scan is stored in the edge memory bit "Tag_M".
- The signal state of the operand "TagIn_5" is "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

--(P)--: Set operand on positive signal edge (S7-1200, S7-1500)

Description

You can use the "Set operand on positive signal edge" instruction to set a specified operand (<Operand1>) when there is a "0" to "1" change in the result of logic operation (RLO). The instruction compares the current RLO with the RLO from the previous query, which is saved in the edge memory bit (<Operand2>). If the instruction detects a change in the RLO from "0" to "1", there is a positive signal edge.

The positive signal edge is queried each time the instruction executes. When a positive signal edge is detected, <Operand1> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

You specify the operand (<Operand1>) to be set in the operand placeholder above the instruction. Specify the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This step influences the edge evaluation and the result is therefore no longer unique. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

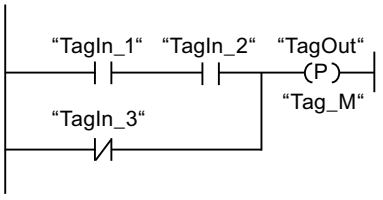
Parameter

The following table shows the parameters of the instruction "Set operand on positive signal edge":

Parameter	Declaration	Data type	Memory area	Description
<Operand1>	Output	BOOL	I, Q, M, D, L	Operand which is set by a positive edge.
<Operand2>	InOut	BOOL	I, Q, M, D, L	Edge memory bit

Example

The following example shows how the instruction works:



Operand "TagOut" is set for one program cycle, when the signal state at the input of the coil switches from "0" to "1" (positive signal edge). In all other cases, the operand "TagOut" has the signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

--(N)--: Set operand on negative signal edge (S7-1200, S7-1500)

Description

You can use the "Set operand on negative signal edge" instruction to set a specified operand (<Operand1>) when there is a "1" to "0" change in the result of logic operation (RLO). The instruction compares the current RLO with the RLO from the previous query, which is saved in the edge memory bit (<Operand2>). If the instruction detects a change in the RLO from "1" to "0", there is a negative signal edge.

The negative signal edge is queried each time the instruction executes. When a negative signal edge is detected, <Operand1> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

You specify the operand (<Operand1>) to be set in the operand placeholder above the instruction. Specify the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This step influences the edge evaluation and the result is therefore no longer unique. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

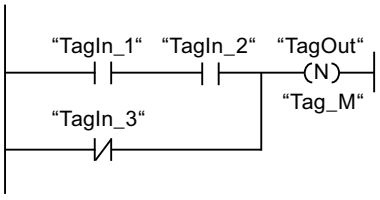
Parameter

The following table shows the parameters of the instruction "Set operand on negative signal edge":

Parameter	Declaration	Data type	Memory area	Description
<Operand1>	Output	BOOL	I, Q, M, D, L	Operand which is set by a negative edge.
<Operand2>	InOut	BOOL	I, Q, M, D, L	Edge memory bit

Example

The following example shows how the instruction works:



Operand "TagOut" is set for one program cycle, when the signal state at the input of the coil switches from "1" to "0" (negative signal edge). In all other cases, the operand "TagOut" has the signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

P_TRIG: Scan RLO for positive signal edge (S7-1200, S7-1500)

Description

Use the "Scan RLO for positive signal edge" instruction to query a "0" to "1" change in the signal state of the result of logic operation (RLO). The instruction compares the current signal state of the RLO with the signal state of the previous query, which is saved in an edge memory bit (<operand>). If the instruction detects a change in the RLO from "0" to "1", there is a positive signal edge.

The positive signal edge is queried each time the instruction executes. As soon as a positive single edge is detected, the output Q of the instruction returns the signal state "1" for the length of a program cycle. In all other cases, the output returns the signal state "0".

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This step influences the edge evaluation and the result is therefore no longer unique. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

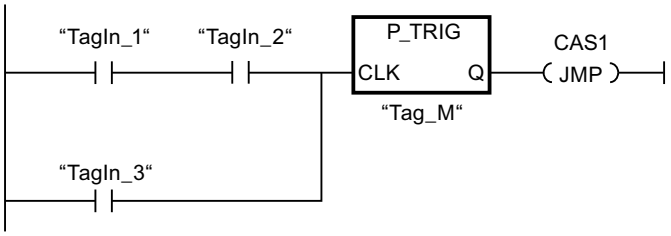
Parameter

The following table shows the parameters of the instruction "Scan RLO for positive signal edge":

Parameter	Declaration	Data type	Memory area	Description
CLK	Input	BOOL	I, Q, M, D, L	Current RLO
<Operand>	InOut	BOOL	M, D	Edge memory bit in which the RLO of the previous query is saved.
Q	Output	BOOL	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:



The RLO of the previous query is saved in the edge memory bit "Tag_M". If a "0" to "1" change is detected in the signal state of the RLO, the program jumps to jump label CAS1.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

N_TRIG: Scan RLO for negative signal edge (S7-1200, S7-1500)

Description

Use the "Scan RLO for negative signal edge" instruction to query a "1" to "0" change in the signal state of the result of logic operation (RLO). The instruction compares the current signal state of the RLO with the signal state of the previous query, which is saved in an edge memory bit (<operand>). If the instruction detects a change in the RLO from "1" to "0", there is a negative signal edge.

The negative signal edge is queried each time the instruction executes. As soon as a negative single edge is detected, the output Q of the instruction returns the signal state "1" for the length of a program cycle. In all other cases, the signal state at the output of the instruction is "0".

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This step influences the edge evaluation and the result is therefore no longer unique. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

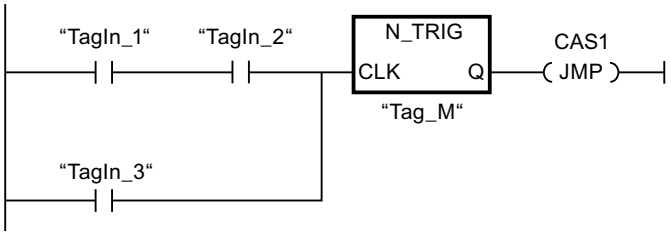
Parameter

The following table shows the parameters of the "Scan RLO for negative signal edge" instruction:

Parameters	Declaration	Data type	Memory area	Description
CLK	Input	BOOL	I, Q, M, D, L	Current RLO
<Operand>	InOut	BOOL	M, D	Edge memory bit in which the RLO of the previous query is saved.
Q	Output	BOOL	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:



The RLO of the previous query is saved in the edge memory bit "Tag_M". If a "1" to "0" change is detected in the signal state of the RLO, the program jumps to jump label CAS1.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

R_TRIG: Detect positive signal edge (S7-1200, S7-1500)

Description

With the "Detect positive signal edge" instruction, you can detect a state change from "0" to "1" at the CLK input. The instruction compares the current value at the CLK input with the state of the previous query (edge memory bit) that is saved in the specified instance. If the instruction detects a state change at the CLK input from "0" to "1", a positive signal edge is generated at the Q output, i.e., the output has the value TRUE or "1" for exactly one cycle.

In all other cases, the signal state at the output of the instruction is "0".

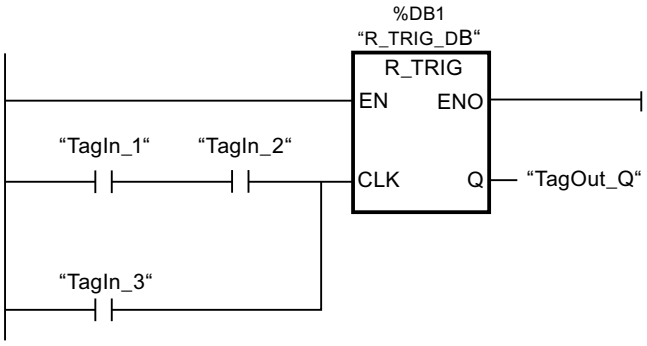
Parameter

The following table shows the parameters of the "Detect positive signal edge" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
CLK	Input	BOOL	I, Q, M, D, L or constant	Incoming signal, the edge of which is to be queried.
Q	Output	BOOL	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:



The previous state of the tag at the CLK input is stored in the "R_TRIG_DB" tag. If a change in the signal state from "0" to "1" is detected in the "TagIn_1" and "TagIn_2" operands or in the "TagIn_3" operand, the "TagOut_Q" output has signal state "1" for one cycle.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

F_TRIG: Detect negative signal edge (S7-1200, S7-1500)

Description

With the "Detect negative signal edge" instruction, you can detect a state change from "1" to "0" at the CLK input. The instruction compares the current value at the CLK input with the state of the previous query (edge memory bit) that is saved in the specified instance. If the instruction detects a state change at the CLK input from "1" to "0", a negative signal edge is generated at the Q output, i.e., the output has the value TRUE or "1" for exactly one cycle.

In all other cases, the signal state at the output of the instruction is "0".

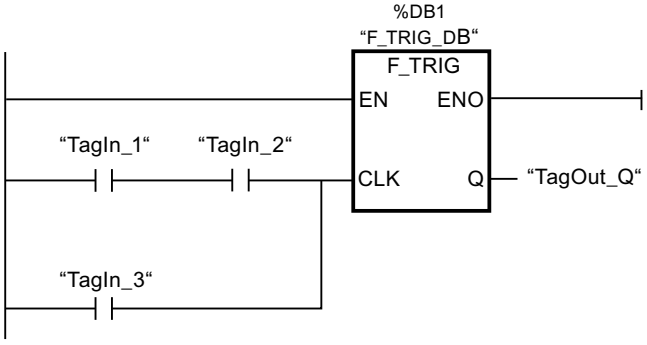
Parameter

The following table shows the parameters of the "Detect negative signal edge" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
CLK	Input	BOOL	I, Q, M, D, L or constant	Incoming signal, the edge of which is to be queried.
Q	Output	BOOL	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:



The previous state of the tag at the CLK input is stored in the "F_TRIG_DB" tag. If a change in the signal state from "1" to "0" is detected in the "TagIn_1" and "TagIn_2" operands or in the "TagIn_3" operand, the "TagOut_Q" output has signal state "1" for one cycle.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Timer operations (S7-1200, S7-1500)

TP: Generate pulse (S7-1200, S7-1500)

Description

You can use the "Generate pulse" instruction to set the Q output for a programmed duration. The instruction is started when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive signal edge). The programmed time PT begins when the instruction starts. Output Q is set for the duration PT, regardless of the subsequent course of the input signal. Even if a new positive signal edge is detected, the signal state at the output Q is not affected as long as the PT time duration is running.

You can scan the current time value at the ET output. The timer value starts at T#0s and ends when the value of the time duration PT is reached. When the time PT has elapsed and the signal state at input IN is "0", the ET output is reset.

Each call of the "Generate pulse" instruction must be assigned to an IEC timer in which the instruction data is stored.

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

For S7-1200 CPU

An IEC timer is a structure of the data type IEC_TIMER or TP_TIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TP_TIME, TP_LTIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

An IEC Timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TP_TIME or TP_LTIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TP_TIME, TP_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- When the instruction is called if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

The execution of the "Generate pulse" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

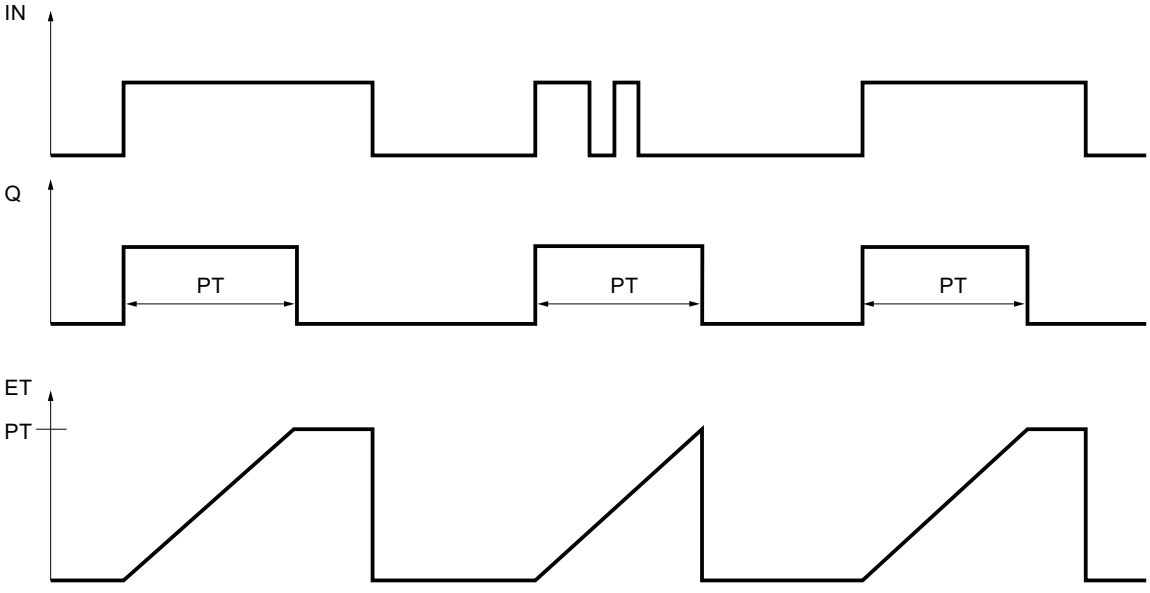
Parameters

The following table shows the parameters of the "Generate pulse" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Duration of the pulse The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Pulse output
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L	I, Q, M, D, L, P	Current time value

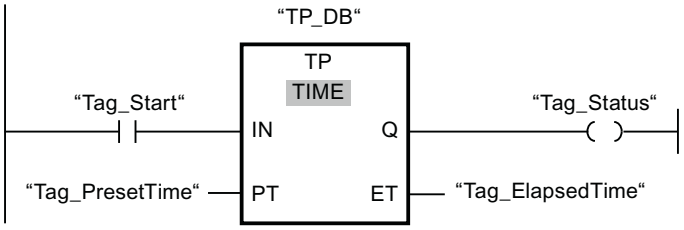
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Generate pulse" instruction:



Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Start	Signal transition "0" => "1"
PT	Tag_PresetTime	T#10s
Q	Tag_Status	TRUE
ET	Tag_ElapsedTime	from T#0s => T#10s

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started and the "Tag_Status" operand is set to "1". The current time value is stored in the "Tag_ElapsedTime" operand. When the time expires, the "Tag_Status" operand is reset to signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Example of controlling room temperature (Page 4933)

Basic information on LAD (Page 4878)

TON: Generate on-delay (S7-1200, S7-1500)

Description

You can use the "Generate on-delay" instruction to delay setting of the Q output by the programmed time PT. The instruction is started when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive signal edge). The programmed time PT begins when the instruction starts. When the time PT has elapsed, the output Q has the signal state "1". Output Q remains set as long as the start input is still "1". When the signal state at the start input changes from "1" to "0", the Q output is reset. The timer function is started again when a new positive signal edge is detected at the start input.

The current time value can be queried at the ET output. The timer value starts at T#0s and ends when the value of the time duration PT is reached. The ET output is reset as soon as the signal state at the IN input changes to "0".

Each call of the "Generate on-delay" instruction must be assigned to an IEC timer in which the instruction data is stored.

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

For S7-1200 CPU

An IEC timer is a structure of the data type IEC_TIMER or TON_TIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TON_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

An IEC timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TON_TIME or TON_LTIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TON_TIME, TON_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- When the instruction is called if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

The execution of the "Generate on-delay" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

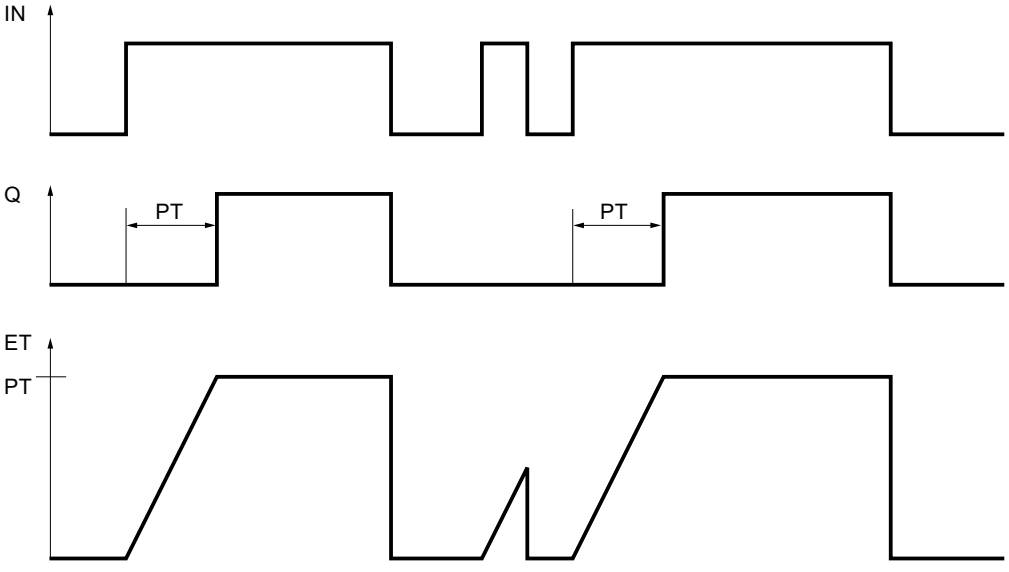
Parameters

The following table shows the parameters of the "Generate on-delay" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Duration of the on-delay The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Output that is set when the time PT expires.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L	I, Q, M, D, L, P	Current time value

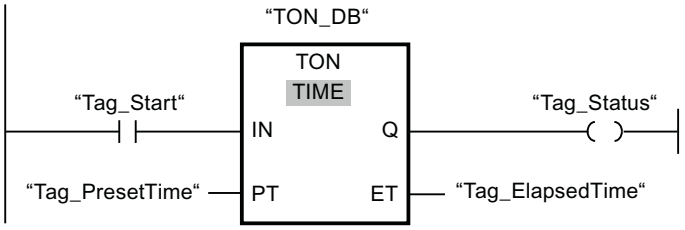
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Generate on-delay" instruction:



Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Start	Signal transition "0" => "1"
PT	Tag_PresetTime	T#10s
Q	Tag_Status	FALSE; after 10 s => TRUE
ET	Tag_ElapsedTime	from T#0s => T#10s

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started. When the time duration expires, the "Tag_Status" operand is set to the signal state "1". The Tag_Status operand remains set to "1" as long as the Tag_Start operand has signal state "1". The current time value is stored in the "Tag_ElapsedTime" operand. When the signal state at the operand "Tag_Start" changes from "1" to "0", the "Tag_Status" operand is reset.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

TOF: Generate off-delay (S7-1200, S7-1500)

Description

You can use the "Generate off-delay" instruction to delay resetting of the Q output by the programmed time PT. The Q output is set when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive signal edge). When the signal state at input IN changes back to "0", the programmed time PT starts. Output Q remains set as long as the time duration PT is running. When the PT time duration expires, the Q output is reset. If the signal state at input IN changes to "1" before the PT time duration expires, the timer is reset. The signal state at the output Q continues to be "1".

The current time value can be queried at the ET output. The timer value starts at T#0s and ends when the value of the time duration PT is reached. When the PT time duration expires, the ET output remains set to the current value until input IN changes back to "1". If input IN switches to "1" before the duration PT has expired, the ET output is reset to the value T#0s.

Each call of the "Generate off-delay" instruction must be assigned to an IEC Timer in which the instruction data is stored.

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

For S7-1200 CPU

An IEC timer is a structure of the data type IEC_TIMER or TOF_TIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TOF_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

An IEC timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TOF_TIME or TOF_LTIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TOF_TIME, TOF_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- When the instruction is called if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

The execution of the "Generate off-delay" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

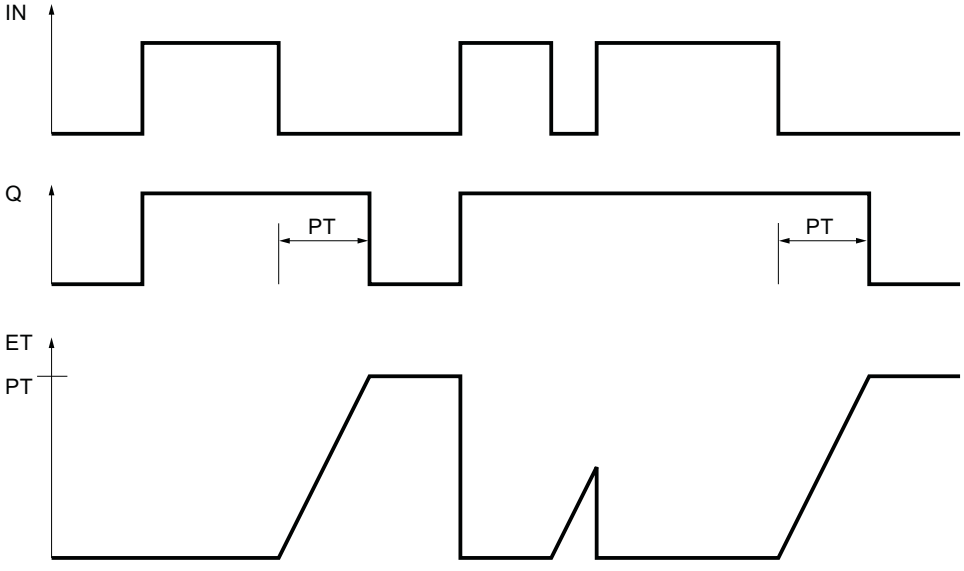
Parameters

The following table shows the parameters of the "Generate off-delay" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Duration of the off delay The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Output that is reset when the time PT expires.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L	I, Q, M, D, L, P	Current time value

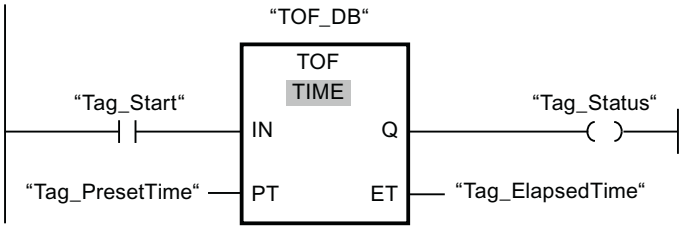
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Generate off-delay" instruction:



Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Start	Signal transition "0" => "1"; signal transition "1" => "0"
PT	Tag_PresetTime	T#10s
Q	Tag_Status	TRUE
ET	Tag_ElapsedTime	from T#10s => T#0s

When the signal state of the operand "Tag_Start" changes from "0" to "1", the "Tag_Status" operand is set to signal state "1". When the signal state of the "Tag_Start" operand changes from "1" to "0", the time programmed for the PT parameter is started. As long as the time is running, the "Tag_Status" operand remains set to TRUE. When the time has expired, the "Tag_Status" operand is reset to FALSE. The current time value is stored in the "Tag_ElapsedTime" operand.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

TONR: Time accumulator (S7-1200, S7-1500)

Description

The instruction "Time accumulator" is used to accumulate time values within a period set by the parameter PT. When the signal state at the IN input changes from "0" to "1" (positive signal edge), the instruction is executed and time PT starts. While the time PT is running, the time values are accumulated that are recorded when the IN input has signal state "1". The accumulated time is written to the ET output and can be queried there. When the duration PT expires, the output Q has signal state "1". The Q parameter remains set to "1", even when the signal state at the IN parameter changes from "1" to "0" (negative signal edge).

The R input resets the ET and Q outputs regardless of the signal state of the start input.

Each call of the "Time accumulator" instruction must be assigned an IEC timer in which the instruction data is stored.

For S7-1200 CPU

An IEC Timer is a structure of the data type IEC_TIMER or TONR_TIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TONR_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

An IEC timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TONR_TIME or TONR_LTIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TONR_TIME, TONR_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- When the instruction is called, if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

The execution of the Time accumulator" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

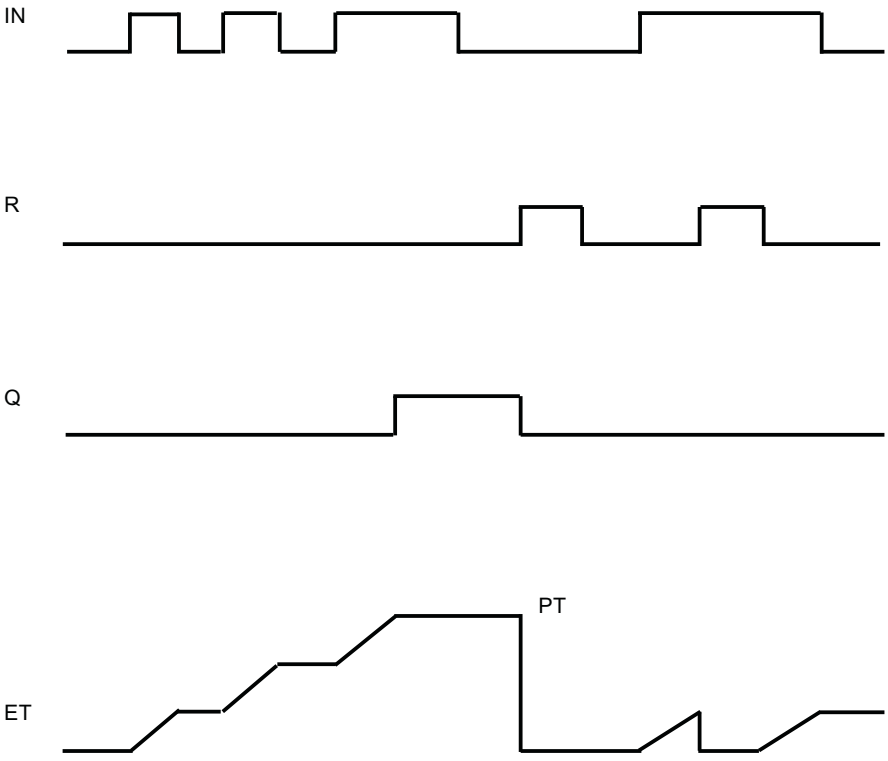
Parameters

The following table shows the parameters of the "Time accumulator" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Start input
R	Input	BOOL	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Reset input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Maximum duration of time recording The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Output that is set when time PT expires.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L	I, Q, M, D, L, P	Accumulated time

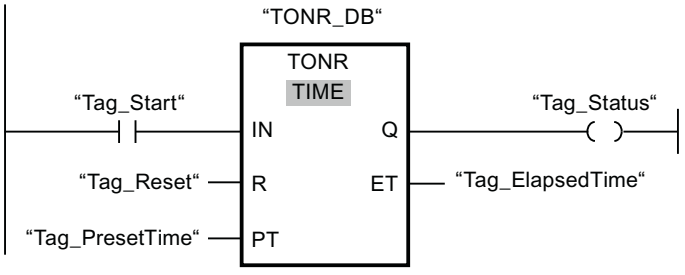
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Time accumulator" instruction:



Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Start	Signal transition "0" => "1"
PT	Tag_PresetTime	T#10s

Parameters	Operand	Value
Q	Tag_Status	FALSE; after 10 s => TRUE
ET	Tag_ElapsedTime	Signal transition from "0" => "1" The time T#10s expires. After 5 s signal transition from "1" => "0": The time at the "Tag_Elapsed-Time" operand remains at T#5s. After approx. 2 s, renewed signal transition from "0" => "1": The time at the "Tag_Elapsed-Time" operand continues to run at T#5s.

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started. The time continues to run as long as the "Tag_Start" operand has the signal state "1". When the signal state of the "Tag_Start" operand changes from "1" to "0", the time stops and the current time value at the Tag_ElapsedTime operand is recorded. When the signal state at the "Tag_Start" operand changes again from "0" to "1", the time continues to run starting at the time value that was recorded at the transition from "1" to "0". When the time value specified for the PT parameter is reached, the "Tag_Status" operand is set to the signal state "1". The current time value is stored in the "Tag_ElapsedTime" operand.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(TP)---: Start pulse timer (S7-1200, S7-1500)

Description

Use the "Start pulse timer" instruction to start an IEC timer with a specified duration as pulse. The IEC timer is started when the result of logic operation (RLO) changes from "0" to "1" (positive signal edge). The IEC timer runs for the specified time regardless of any subsequent changes in the RLO. The run of the IEC timer is also not affected by the detection of a new positive signal edge. As long as the IEC timer is running, the querying of the timer status for "1" returns the signal state "1". When the IEC timer has expired, the timer status returns the signal state "0".

Specify the duration of the pulse under the instruction at <Operand1> (duration) and specify the IEC time to be started above the instruction at <Operand2> (IEC time).

Note

You can start and query the IEC timer at various execution levels, as each querying of the outputs Q or ET updates the IEC_TIMER structure.

For S7-1200 CPU

The instruction "Start pulse timer" stores its data in a structure of the data type IEC_TIMER or TP_TIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TP_LTIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

The instruction "Start pulse timer" stores its data in a structure of the data type IEC_TIMER, IEC_LTIMER, TP_TIME or TP_LTIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TP_TIME, TP_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- The IEC_Timer structure is updated when the instruction is called. The time value at the ET output is only updated if the ET or Q outputs are scanned (e.g. "MyTimer".Q or "MyTimer".ET).
- When the assigned timer is accessed.

The current timer status is stored in the Q structure component of the IEC timer. You can use a normally open contact to query timer status for "1" or a normally closed contact for "0".

The execution of the "Start pulse timer" instruction assumes a preceding logic operation. It can be placed only at the end of the network.

Parameters

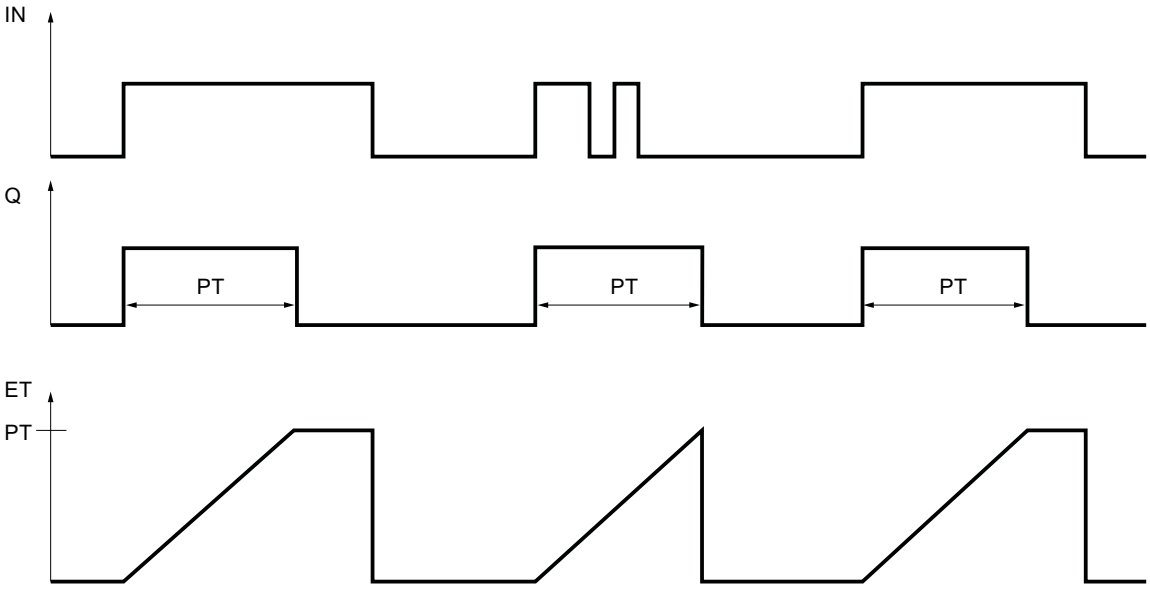
The following table shows the parameters of the "Start pulse timer" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<Time duration>	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC timer runs.
<IEC timer>	InOut	IEC_TIMER, TP_TIME	IEC_TIMER, IEC_LTIMER, TP_TIME, TP_LTIME	D, L	IEC timer that is started.

You can find additional information on valid data types under "See also".

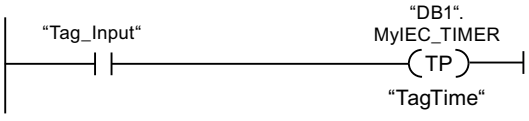
Pulse timing diagram

The following figure shows the pulse timing diagram of the instruction:

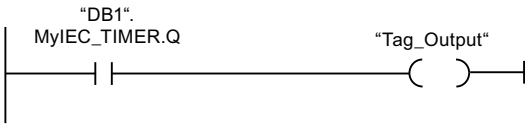


Example

The following example shows how the instruction works:



The "Start pulse timer" instruction is executed when the signal state of the operand "Tag_Input" changes from "0" to "1". The timer "DB1".MyIEC_TIMER is started for the time stored in the operand "TagTime".



As long as the timer "DB1". MyIEC_TIMER is running, the timer status ("DB1".MyIEC_TIMER.Q) has signal state "1" and the operand "Tag_Output" is set. When the IEC timer has expired, the signal state of the time status changes back to "0" and the "Tag_Output" operand is reset.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(TON)---: Start on-delay timer (S7-1200, S7-1500)

Description

Use the "Start on-delay timer" instruction to start an IEC timer with a specified duration as on-delay. The IEC timer is started when the result of logic operation (RLO) changes from "0" to "1" (positive signal edge). The IEC timer runs for the specified time. The output returns the signal state "1" if the RLO at the input of the instruction has the signal state "1". If the RLO changes to "0" before the time expires, the IEC timer is reset. In this case, querying the timer status for "1" returns signal state "0". The IEC timer restarts when the next positive signal edge is detected at the input of the instruction.

Specify the duration of the ON delay under the instruction at <Operand1> (duration) and specify the IEC time to be started above the instruction at <Operand2> (IEC time).

Note

You can start and query the IEC timer at various execution levels, as each querying of the outputs Q or ET updates the IEC_TIMER structure.

For S7-1200 CPU

The instruction "Start on-delay timer" stores its data in a structure of the data type IEC_TIMER or TON_TIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TON_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

The "Start on-delay timer" instruction stores its data in a structure of the data type IEC_TIMER, IEC_LTIMER, TON_TIME or TON_LTIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TON_TIME, TON_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- The IEC_Timer structure is updated when the instruction is called. The time value at the ET output is only updated if the ET or Q outputs are scanned (e.g. "MyTimer".Q or "MyTimer".ET).
- When the assigned timer is accessed.

The current timer status is stored in the ET structure component of the IEC timer. You can use a normally open contact to query timer status for "1" or a normally closed contact for "0".

The execution of the "Start on-delay timer" instruction assumes a preceding logic operation. It can be placed only at the end of the network.

Parameters

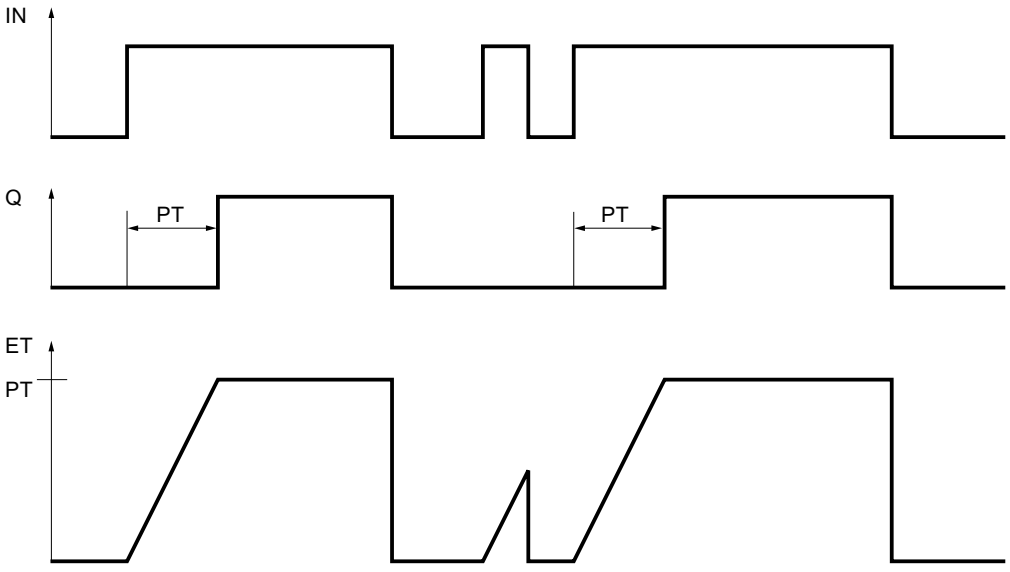
The following table shows the parameters of the "Start on-delay timer" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<Time duration>	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC timer runs.
<IEC timer>	InOut	IEC_TIMER, TON_TIME	IEC_TIMER, IEC_LTIMER, TON_TIME, TON_LTIME	D, L	IEC timer that is started.

You can find additional information on valid data types under "See also".

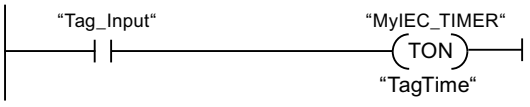
Pulse timing diagram

The following figure shows the pulse timing diagram of the instruction:

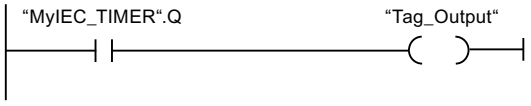


Example

The following example shows how the instruction works:



The "Start on-delay timer" instruction is executed when the signal state of the operand "Tag_Input" changes from "0" to "1". The "MyIEC_TIMER timer is started for the time stored in the "TagTime" operand.



If the timer "MyIEC_TIMER" has expired and the operand "Tag_Input" has the signal state "1", querying the timer status ("MyIEC_TIMER).Q) returns signal state "1" and the "Tag_Output" operand is set. When the signal state of the operand "Tag_Input" changes to "0", the querying of the timer status returns the signal state "0" and the operand "Tag_Output" is reset.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(TOF)---: Start off-delay timer (S7-1200, S7-1500)

Description

Use the "Start off-delay timer" instruction to start an IEC timer with a specified duration as on-delay. The query of the timer status for "1" returns the signal state "0" if the result of logic operation (RLO) at the input of the instruction has the signal state "1". When the RLO changes from "1" to "0" (negative signal edge), the IEC timer starts with the specified time. The timer status remains at signal state "1" as long as the IEC timer is running. When the timer has run out and the RLO at the input of the instruction has the signal state "0", the timer status is set to the signal state "0". If the RLO changes to "1" before the time expires, the IEC timer is reset and the timer status remains at signal state "1".

Specify the duration of the OFF delay under the instruction at <Operand1> (duration) and specify the IEC time to be started above the instruction at <Operand2> (IEC time).

Note

You can start and query the IEC timer at various execution levels, as each querying of the outputs Q or ET updates the IEC_TIMER structure.

For S7-1200 CPU

The instruction "Start off-delay timer" stores its data in a structure of the data type IEC_TIMER or TOF_TIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TOF_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

The "Start off-delay timer" instruction stores its data in a structure of the data type IEC_TIMER, IEC_LTIMER, TOF_TIME or TOF_LTIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TOF_TIME, TOF_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- The IEC_Timer structure is updated when the instruction is called. The time value at the ET output is only updated if the ET or Q outputs are scanned (e.g. "MyTimer".Q or "MyTimer".ET).
- When the assigned timer is accessed.

The current timer status is stored in the ET structure component of the IEC timer. You can use a normally open contact to query timer status for "1" or a normally closed contact for "0".

The execution of the "Start off-delay timer" instruction assumes a preceding logic operation. It can be placed only at the end of the network.

Parameters

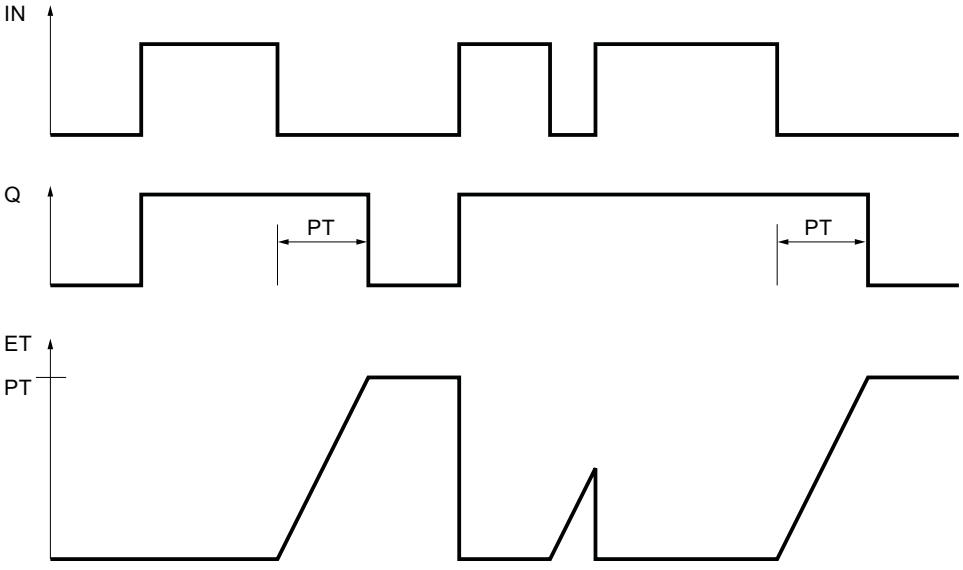
The following table shows the parameters of the "Start off-delay timer" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<Time duration>	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC timer runs.
<IEC timer>	InOut	IEC_TIMER, TOF_TIME	IEC_TIMER, IEC_LTIMER, TOF_TIME, TOF_LTIME	D, L	IEC timer that is started.

You can find additional information on valid data types under "See also".

Pulse timing diagram

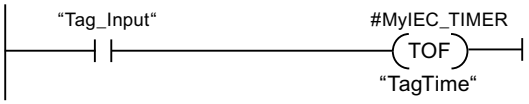
The following figure shows the pulse timing diagram of the instruction:



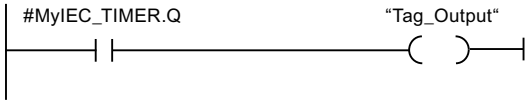
Example

As long as timer #MyIEC_TIMER is running, the query of the time status (#MyIEC_TIMER.Q) returns the signal state "1" and operand "Tag_Output" is set. If the timer has expired and the operand "Tag_Input" has the signal state "0", the query of the timer status returns the signal state "0". If the signal state of the operand "Tag_Input" changes to "1" before timer #MyIEC_TIMER expires, the timer is reset. When the signal state of the operand "Tag_Input" is "1", the query of the timer status returns the signal state "1".

The following example shows how the instruction works:



The "Start off-delay timer" instruction is executed when the signal state of the operand "Tag_Input" changes from "1" to "0". The timer #MyIEC_TIMER is started for the time stored in the operand "TagTime".



For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(TONR)---: Time accumulator (S7-1200, S7-1500)

Description

You can use the "Time accumulator" instruction to record how long the signal is at the input of instruction "1". The instruction is started when the result of logic operation (RLO) changes from "0" to "1" (positive signal edge). The time is recorded as long at the RLO is "1". If the RLO changes to "0", the instruction is halted. If the RLO changes back to "1", the time recording is continued. The query of the timer status for "1" returns the signal state "1" if the recorded time exceeds the value of the specified duration and the RLO at the input of coil is "1".

The timer status and the currently expired timer can be reset to "0" using the "Reset timer" instruction.

Specify the duration under the instruction at <Operand1> (duration) and specify the IEC time to be started above the instruction at <Operand2> (IEC time).

Note

You can start and query the IEC timer at various execution levels, as each querying of the outputs Q or ET updates the IEC_TIMER structure.

The execution of the "Time accumulator" instruction requires a preceding logic operation. It can be placed only at the end of the network.

For S7-1200 CPU

The "Time accumulator" instruction stores its data in a structure of the data type IEC_TIMER or TONR_TIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TONR_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

The "Time accumulator" instruction stores its data in a structure of the data type IEC_TIMER, IEC_LTIMER, TONR_TIME or TONR_LTIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TONR_TIME, TONR_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- The IEC_Timer structure is updated when the instruction is called. The time value at the ET output is only updated if the ET or Q outputs are scanned (e.g. "MyTimer".Q or "MyTimer".ET).
- When the assigned timer is accessed.

The current timer status is stored in the ET structure component of the IEC timer. You can use a normally open contact to query timer status for "1" or a normally closed contact for "0".

Parameters

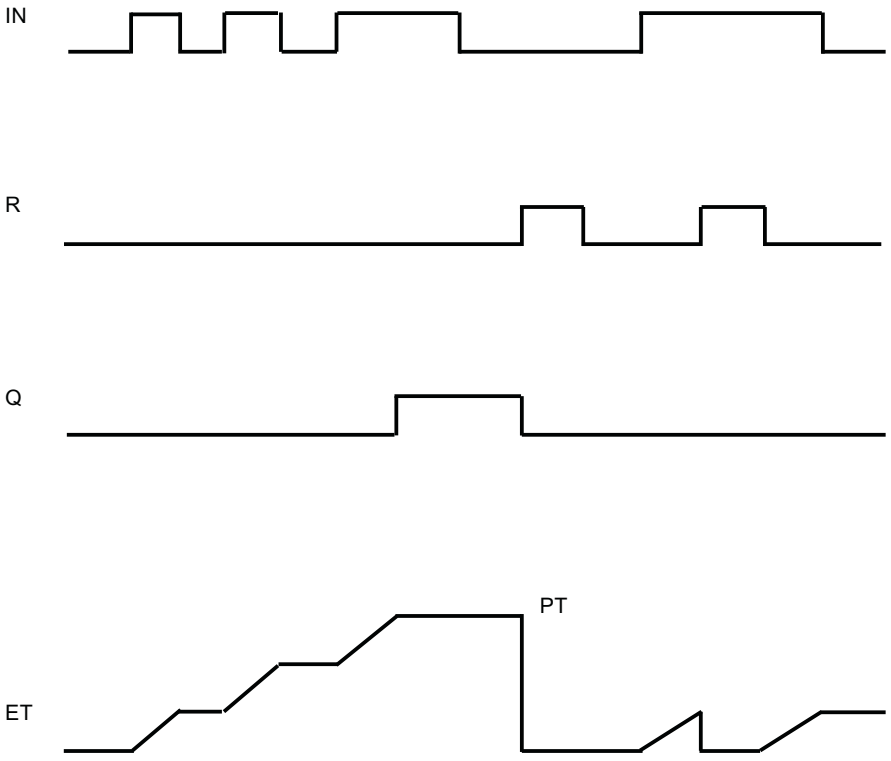
The following table shows the parameters of the "Time accumulator" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<Time duration>	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC timer runs.
<IEC timer>	InOut	IEC_TIMER, TONR_TIME	IEC_TIMER, IEC_LTIMER, TONR_TIME, TONR_LTIME	D, L	IEC timer that is started.

You can find additional information on valid data types under "See also".

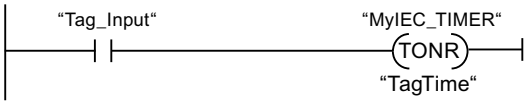
Pulse timing diagram

The following figure shows the pulse timing diagram of the instruction:

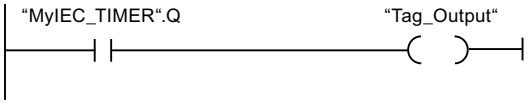


Example

The following example shows how the instruction works:



The "Time accumulator" instruction executes on a positive signal edge in the RLO. The time is recorded as long as the operand "Tag_Input" has the signal state "1".



If the recorded time exceeds the value of the operand "TagTime", then the query of the timer status ("MyIEC_TIMER".Q) will return the signal state "1" and the operand "Tag_Output" will be set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(RT)---: Reset timer (S7-1200, S7-1500)

Description

You can use the "Reset timer" instruction to reset an IEC timer to "0". The instruction is only executed if the result of logic operation (RLO) at the input of the coil is "1". If current is flowing to the coil (RLO is "1"), the structure components of the timer in the specified data block are reset to "0". If the RLO at the input of the instruction is "0", the timer remains unchanged.

The instruction does not influence the RLO. The RLO at the input of the coil is sent directly to the output of the coil.

You assign the "Reset timer" instruction an IEC timer that has been declared in the program.

The instruction data is updated only when the instruction is called and not each time the assigned IEC timer is accessed. Querying the data is only identical from the call of the instruction to the next call of the instruction.

Parameters

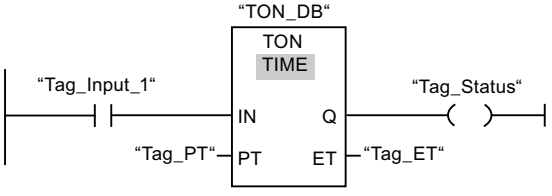
The following table shows the parameters of the "Reset timer" instruction":

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<IEC timer>	Output	IEC_TIMER, TP_TIME, TON_TIME, TOF_TIME, TONR_TIME	IEC_TIMER, IEC_LTIMER, TP_TIME, TP_LTIME, TON_TIME, TON_LTIME, TOF_TIME, TOF_LTIME, TONR_TIME, TONR_LTIME	D, L	IEC timer that is reset.

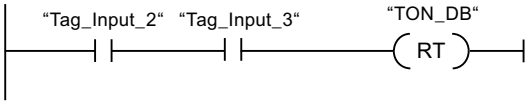
You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The "Generate on-delay" instruction executes when the signal state of the "Tag_Input_1" operand changes from "0" to "1". The timer stored in the "TON_DB" instance data block starts running for the time duration specified by operand "Tag_PT".



If operands "Tag_Input_2" and "Tag_Input_3" have the signal state "1", the "Reset timer" instruction is executed and the timer stored in the "TON_DB" data block.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(PT)---: Load time duration (S7-1200, S7-1500)

Description

You can use the "Load time duration" instruction to set the time for an IEC timer. The instruction is executed in every cycle when the result of logic operation (RLO) at the input of the instruction has the signal state "1". The instruction writes the specified time to the structure of the specified IEC timer.

Specify the duration that is loaded under the instruction at <Operand1> (duration) and specify the IEC time to be started above the instruction at <Operand2> (IEC time).

Note

If the specified IEC timer is running while the instruction executes, the instruction overwrites the current time of the specified IEC timer. This can change the timer status of the IEC timer.

You assign an IEC timer declared in the program to the "Load time duration" instruction.

The instruction data is updated only when the instruction is called and each time the assigned IEC timer is accessed. The query on Q or ET (for example, "MyTimer".Q or "MyTimer".ET) updates the IEC_TIMER structure.

Parameter

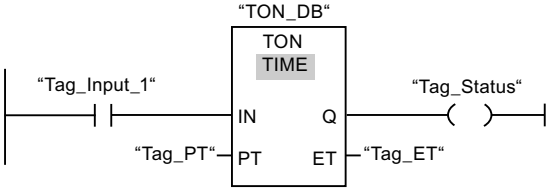
The following table shows the parameters of the "Load time duration" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<Time duration>	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC timer runs.
<IEC timer>	Output	IEC_TIMER, TP_TIME, TON_TIME, TOF_TIME, TONR_TIME	IEC_TIMER, IEC_LTIMER, TP_TIME, TP_LTIME, TON_TIME, TON_LTIME, TOF_TIME, TOF_LTIME, TONR_TIME, TONR_LTIME	D, L	IEC timer, the duration of which is set.

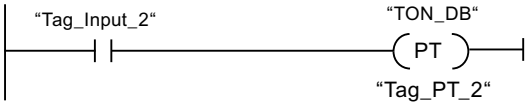
You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The "Generate on-delay" instruction executes when the signal state of the "Tag_Input_1" operand changes from "0" to "1". The IEC timer stored in the instance data block "TON_DB" is started with the time duration that is specified by the operand "Tag_PT".



The "Load time duration" instruction is executed when the operand "Tag_Input_2" has the signal state "1". The instruction writes the time duration "Tag_PT_2" in the instance data block "TON_DB" and at the same time overwrites the value of the operand "Tag_PT" within the data block. The signal state of the timer status may therefore change at the next query or when "MyTimer".Q or "MyTimer".ET are accessed.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Legacy (S7-1500)

S_PULSE: Assign pulse timer parameters and start (S7-1500)

Description

The "Assign pulse timer parameters and start" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV) as soon as the signal state at input S is "1". If the signal state at input S changes to "0" before the programmed duration expires, the timer is stopped. In this case, the signal state at output Q is "0".

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down towards zero. The time base indicates the time period with which the time value is changed. The current time value is output BI-coded at output BI and BCD-coded at output BCD.

If the timer is running and the signal state at input R changes to "1" then the current time value and the time base are also set to zero. If the timer is not running, the signal state "1" at the R input has no effect.

The "Assign pulse timer parameters and start" instruction requires a preceding logic operation for edge evaluation and can be placed within or at the end of the network.

The instruction data is updated at every access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

For an example of how a time cell can be structured, refer to See also "L: Load timer value".

Parameters

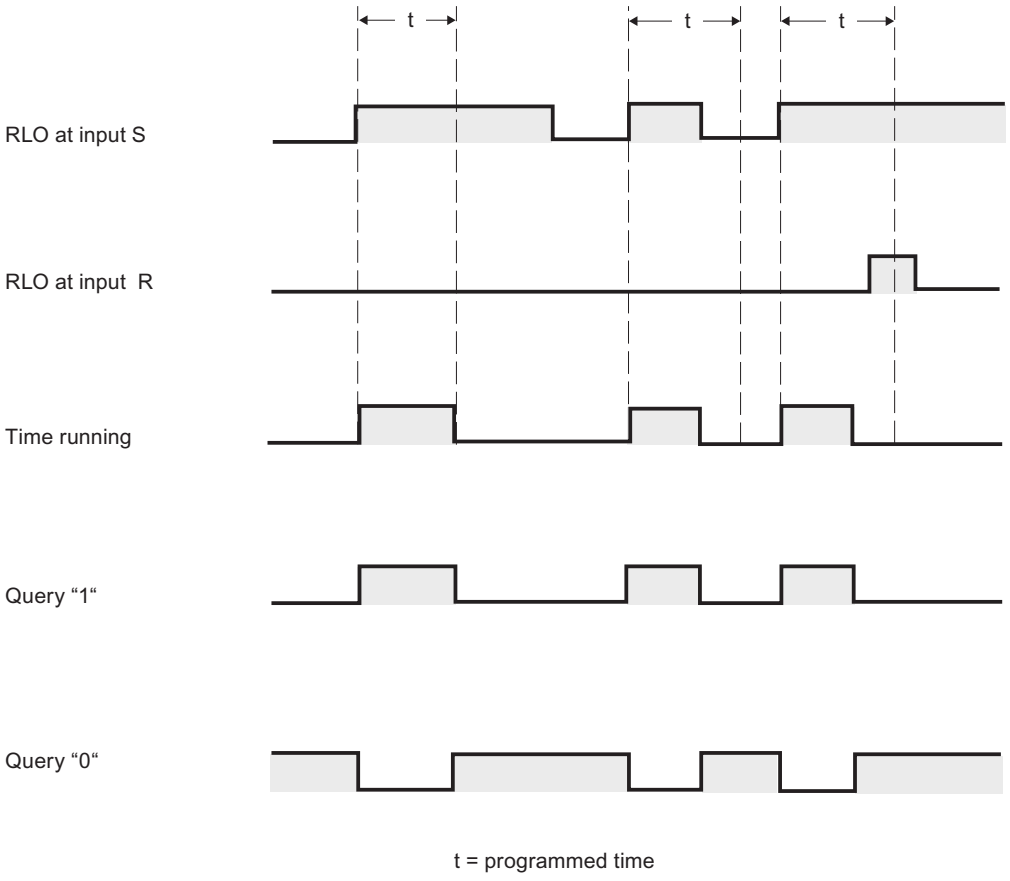
The following table shows the parameters of the instruction "Assign pulse timer parameters and start":

Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Preset time value
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (BI-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the timer

You can find additional information on valid data types under "See also".

Pulse timing diagram

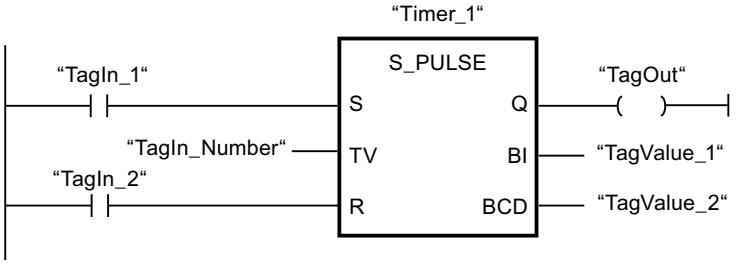
The following figure shows the pulse timing diagram of the instruction "Assign pulse timer parameters and start":



t = programmed time

Example

The following example shows how the instruction works:



The timer "Timer_1" is started when the signal state of the operand "TagIn_1" changes from "0" to "1". The timer expires with the time value of the operand "TagIn_Number" as long as the operand "TagIn_1" has the signal state "1". If the signal state of the operand "TagIn_1" changes from "1" to "0" before the timer expires, the timer "Timer_1" is stopped. The operand "TagOut" is reset to "0" in this case.

The operand "TagOut" has the signal state "1" as long as the timer is running and the operand "TagIn_1" has the signal state "1". When the timer expires or is reset, the operand "TagOut" is reset to "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

S_PEXT: Assign extended pulse timer parameters and start (S7-1500)

Description

The "Assign extended pulse timer parameters and start" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV), even if the signal state at input S changes to "0". As long as the timer is running, the output Q has the signal state "1". When the timer expires, the output Q is reset to "0". If the signal state at input S changes from "0" to "1" while the timer is running, the timer is restarted with the duration programmed at input TV.

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down towards zero. The time base indicates the time period with which the time value is changed. The current time value is output BI-coded at output BI and BCD-coded at output BCD.

If the timer is running and the signal state at input R changes to "1" then the current time value and the time base are also set to zero. If the timer is not running, the signal state "1" at the R input has no effect.

The "Assign extended pulse timer parameters and start" instruction requires a preceding logic operation for edge evaluation and can be placed within or at the end of the network.

The instruction data is updated at every access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

For an example of how a time cell can be structured, refer to See also "L: Load timer value".

Parameters

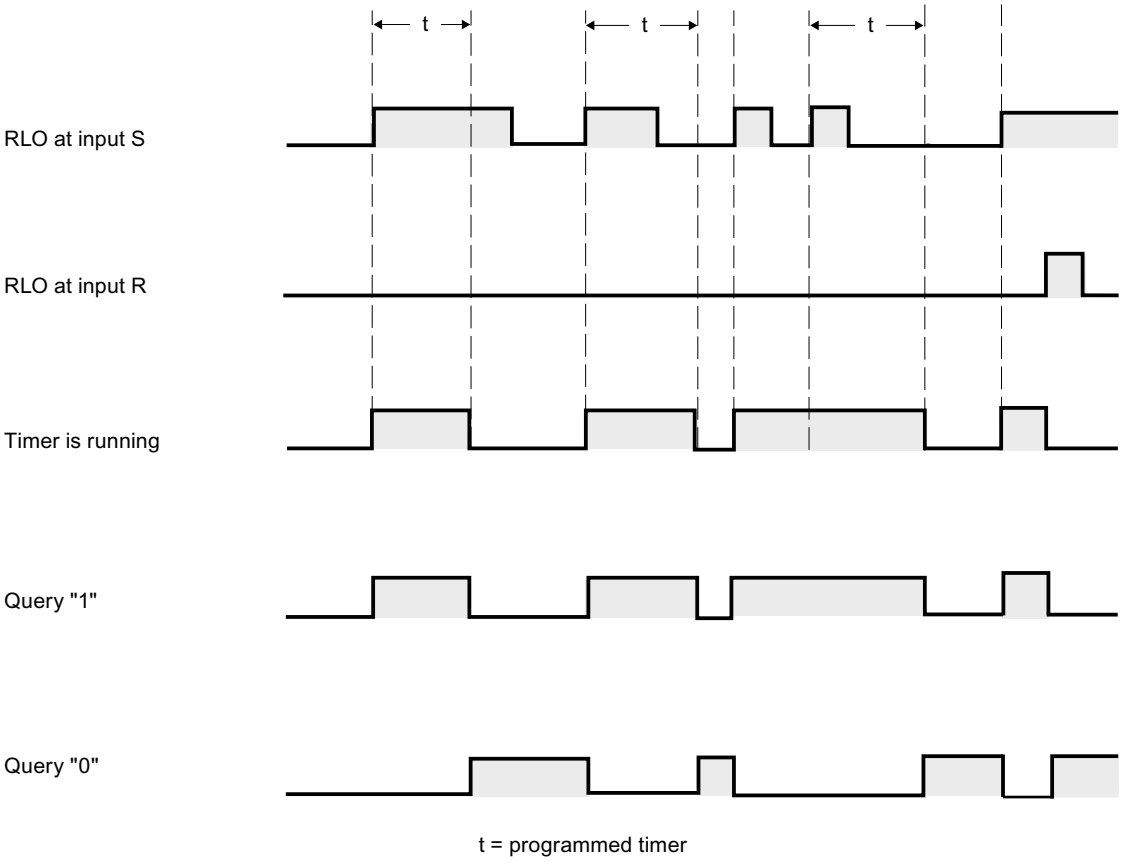
The following table shows the parameters of the instruction "Assign extended pulse timer parameters and start":

Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Preset time value
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (BI-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the timer

You can find additional information on valid data types under "See also".

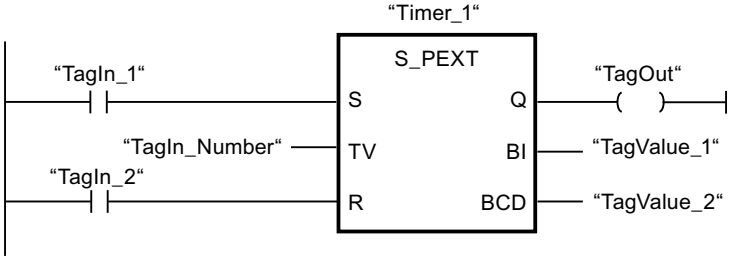
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign extended pulse timer parameters and start" instruction:



Example

The following example shows how the instruction works:



The timer "Timer_1" is started when the signal state of the operand "TagIn_1" changes from "0" to "1". The timer expires with the time value of the operand "TagIn_Number" without being affected by a negative edge at input S. If the signal state of the operand "TagIn_1" changes from "0" to "1" before the timer expires, the timer is restarted.

The operand "TagOut" has the signal state "1" as long as the timer is running. When the timer expires or is reset, the operand "TagOut" is reset to "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

S_ODT: Assign on-delay timer parameters and start (S7-1500)

Description

The "Assign on-delay timer parameters and start" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV) as soon as the signal state at input S is "1". If the timer expires correctly and input S still has signal state "1" then output Q returns signal state "1". If the signal state at input S changes from "1" to "0" while the timer is running, the timer is stopped. In this case, output Q is reset to signal state "0".

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down towards zero. The time base indicates the time period with which the time value is changed. The current time value is output BI-coded at output BI and BCD-coded at output BCD.

If the time is running and the signal state at input R changes from "0" to "1" then the current time value and the time base are also set to zero. In this case, the signal state at output Q is "0". The timer is reset if the signal state is "1" at the R input even if the timer is not running and the RLO at input S is "1".

Specify the timer of the instruction in the placeholder above the box. The timer must be declared with the data type TIMER.

The "Assign on-delay timer parameters and start" instruction requires a preceding logic operation for edge evaluation and can be placed within or at the end of the network.

The instruction data is updated at every access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

For an example of how a time cell can be structured, refer to See also "L: Load timer value".

Parameters

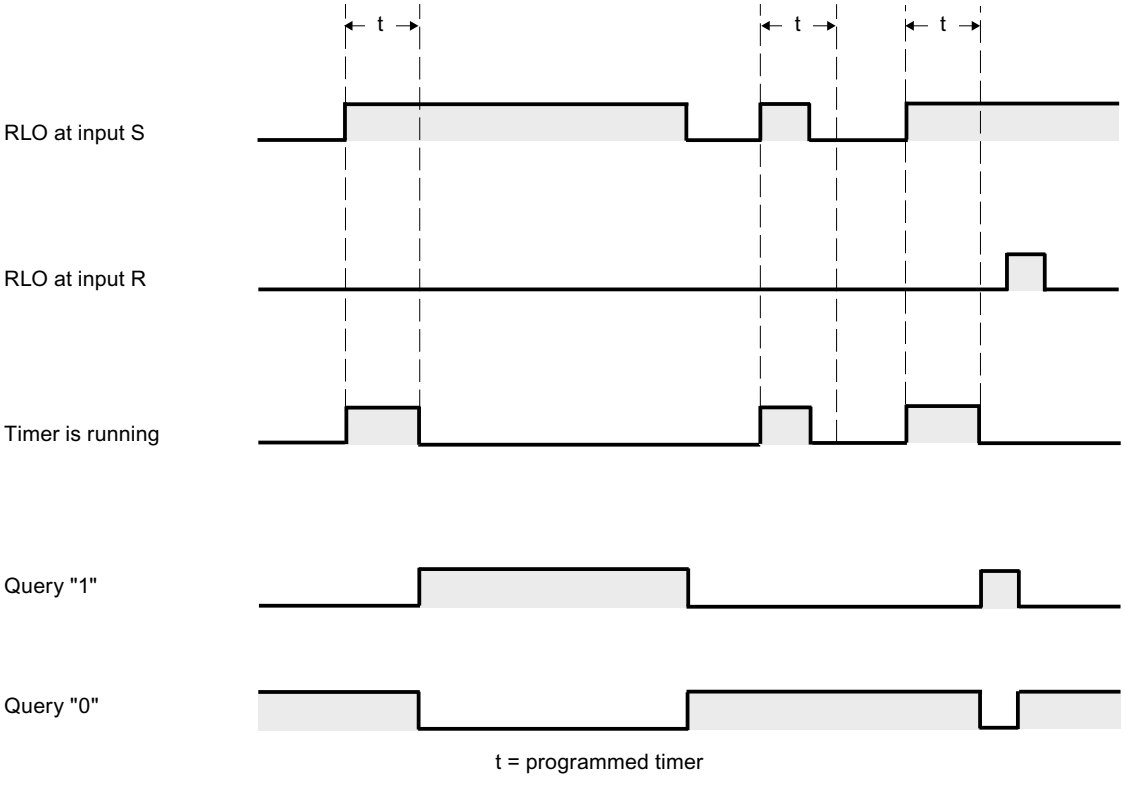
The following table shows the parameters of the "Assign on-delay timer parameters and start" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Preset time value
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (BI-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the timer

You can find additional information on valid data types under "See also".

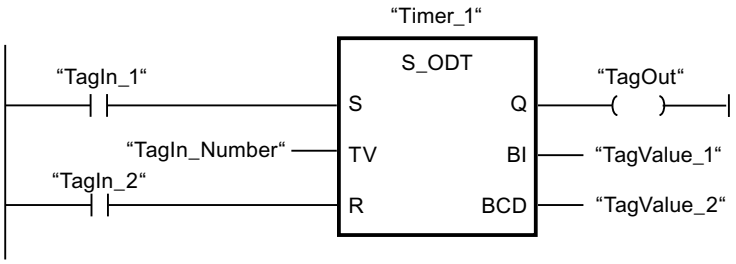
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign on-delay timer parameters and start" instruction:



Example

The following example shows how the instruction works:



The timer "Timer_1" is started when the signal state of the operand "TagIn_1" changes from "0" to "1". The timer expires with the time value of the operand "TagIn_Number". If the timer expires and the operand has the signal state "1", the operand "TagOut" is set to "1". If the signal state of the operand "TagIn_1" changes from "1" to "0" before the timer expires, the timer is stopped. The operand "TagOut" has the signal state "0" in this case.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

S_ODTS: Assign retentive on-delay timer parameters and start (S7-1500)

Description

The "Assign retentive on-delay timer parameters and start" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV), even if the signal state at input S changes to "0". If the timer expires, the "Q" output returns signal state "1" regardless of the signal state at input "S". If the signal state at input S changes from "0" to "1" while the timer is running, the timer is restarted with the duration programmed at input (TV).

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down towards zero. The time base indicates the time period with which the time value is changed. The current time value is output BI-coded at output BI and BCD-coded at output BCD.

Signal state "1" at input R resets the current time value and time base to "0" regardless of the signal state at start input S. In this case, the signal state at output Q is "0".

The "Assign retentive on-delay timer parameters and start" instruction requires a preceding logic operation for edge evaluation and can be placed within or at the end of the network.

The instruction data is updated at every access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

For an example of how a time cell can be structured, refer to See also "L: Load timer value".

Parameters

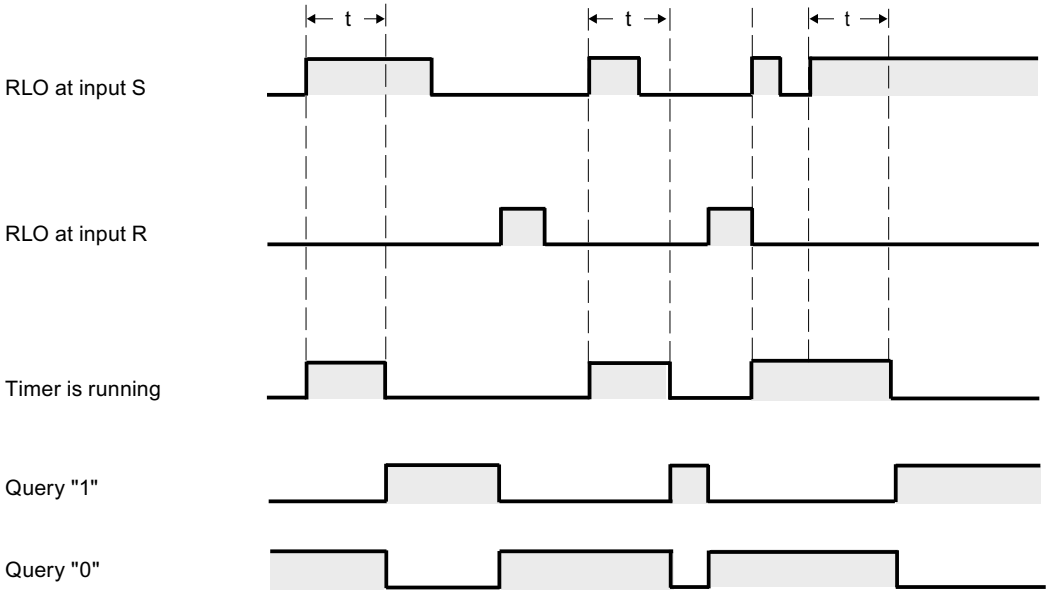
The following table shows the parameters of the "Assign retentive on-delay timer parameters and start" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Preset time value
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (BI-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the timer

You can find additional information on valid data types under "See also".

Pulse timing diagram

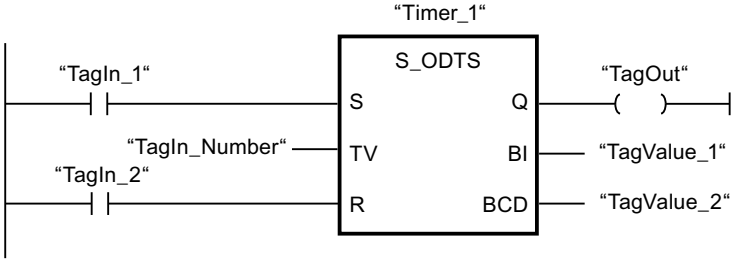
The following figure shows the pulse timing diagram of the "Assign retentive on-delay timer parameters and start" instruction:



t = programmed timer

Example

The following example shows how the instruction works:



The timer "Timer_1" is started when the signal state of the operand "TagIn_1" changes from "0" to "1". The timer expires with the time value of the operand "TagIn_Number", even if the signal state of the operand "TagIn_1" changes to "0". When the timer expires, the operand "TagOut" is set to "1". If the signal state of the operand "TagIn_1" changes from "0" to "1" while the timer is running, the timer is restarted.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

S_OFFDT: Assign off-delay timer parameters and start (S7-1500)

Description

The "Assign off-delay timer parameters and start" instruction starts a programmed timer when a change from "1" to "0" (negative signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV). As long as the timer is running or input S returns signal state "1", then output Q has signal state "1". When the timer expires and the signal state at input S is "0", output Q is reset to the signal state "0". If the signal state at input S changes from "0" to "1" while the timer is running, the timer is stopped. The timer is only restarted after a falling signal edge is detected at input S.

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down towards zero. The time base indicates the time period with which the time value is changed. The current time value is output BI-coded at output BI and BCD-coded at output BCD.

Signal state "1" at input R resets the current time value and time base to "0". In this case, the signal state at output Q is "0".

The "Assign off-delay timer parameters and start" instruction requires a preceding logic operation for edge evaluation and can be placed within or at the end of the network.

The instruction data is updated at every access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

For an example of how a time cell can be structured, refer to See also "L: Load timer value".

Parameters

The following table shows the parameters of the "Assign off-delay timer parameters and start" instruction:

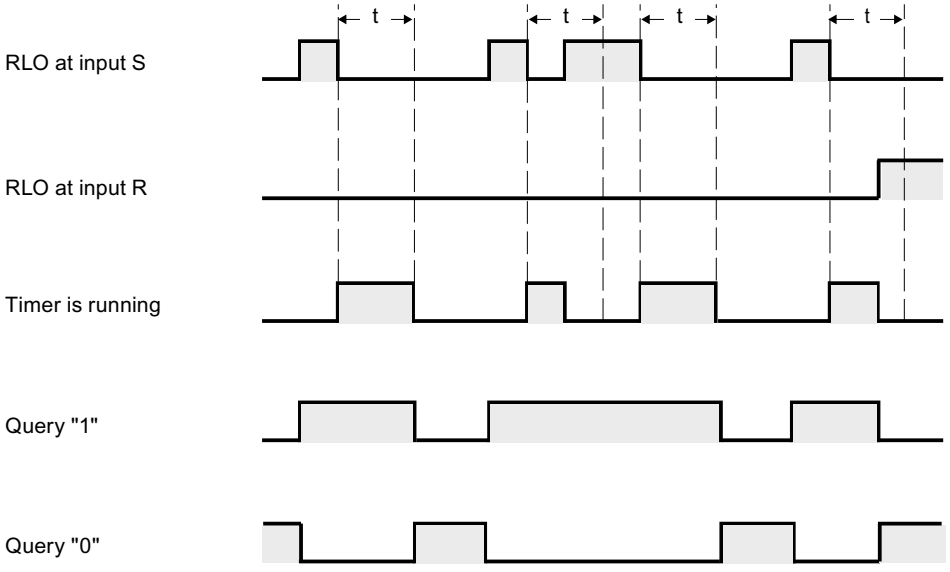
Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input

Parameters	Declaration	Data type	Memory area	Description
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Preset time value
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (BI-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the timer

You can find additional information on valid data types under "See also".

Pulse timing diagram

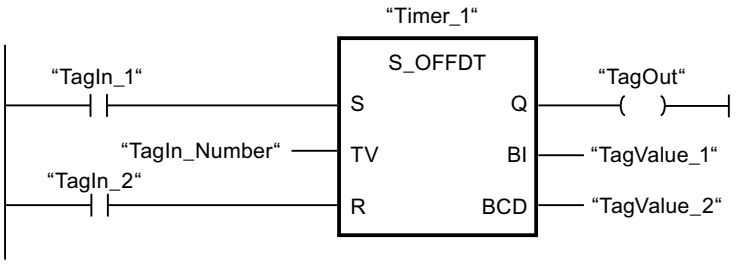
The following figure shows the pulse timing diagram of the "Assign off-delay timer parameters and start" instruction:



t = programmed timer

Example

The following example shows how the instruction works:



The timer "Timer_1" is started when the signal state of the operand "TagIn_1" changes from "1" to "0". The timer expires with the time value of the operand "TagIn_Number". The operand "TagOut" is set to "1" when the timer is running and when the operand "TagIn_1" has the signal state "0". If the signal state of the operand "TagIn_1" changes from "0" to "1" while the timer is running, the timer is reset.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(SP): Start pulse timer (S7-1500)

Description

The "Start pulse timer" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO). The timer runs with the specified duration as long as the RLO has the signal state "1". As long as the timer is running, the query of timer status "1" returns the signal state "1". If there is a change from "1" to "0" in the RLO before the time value has elapsed, the timer stops. In this case, the querying of the timer status for "1" returns the signal state "0".

The duration is made up internally of a time value and a time base. When the instruction is started, the programmed time value is counted down to zero. The time base indicates the time period with which the time value is changed.

Specify the duration with which the time will run down at the operand placeholder <Operand1> (duration) and specify the time to be started above the instruction in <Operand2>.

The "Start pulse timer" instruction requires a preceding logic operation for edge evaluation and can only be placed at the right side of the network.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

For an example of how a time cell can be structured, refer to See also "L: Load timer value".

Parameters

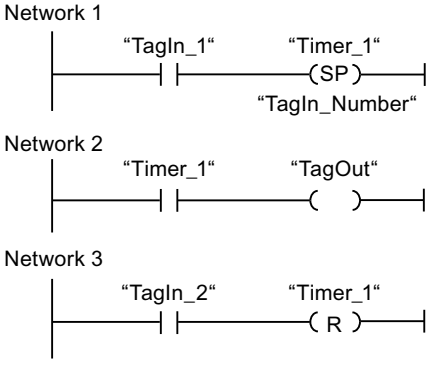
The following table shows the parameters of the instruction "Start pulse timer":

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	S5TIME, WORD	I, Q, M, D, L or constant	Duration with which the timer expires.
<Operand2>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

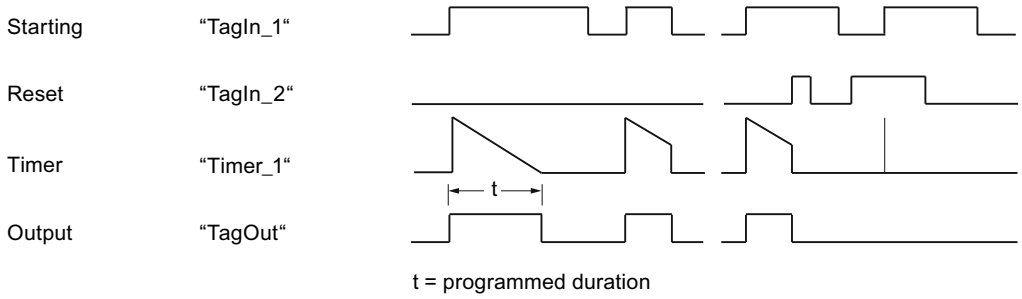
Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer expires with the time value of the "TagIn_Number" operand as long as the signal state of the "TagIn_1" operand is "1". If the signal state of the operand "TagIn_1" changes from "1" to "0" before the timer expires, the timer is stopped. As long as the timer is running, the "TagOut" operand returns signal state "1". A signal state change of the operand "TagIn_2" from "0" to "1" resets the timer, which stops the timer and sets the current time value to "0".

The figure below shows the timer diagram of the example:



For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(SE): Start extended pulse timer (S7-1500)

Description

The "Start extended pulse timer" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO). The timer runs for the specified time period even when the RLO changes to signal state "0". As long as the timer is running, the query of timer status "1" returns the signal state "1". If the RLO changes from "0" to "1" while the timer is running, the timer is restarted with the programmed time period. The querying of the timer status for "1" returns the signal state "0" when the timer expires.

The duration is made up internally of a time value and a time base. When the instruction is started, the programmed time value is counted down to zero. The time base indicates the time period with which the time value is changed.

Specify the duration with which the time will run down at the operand placeholder <Operand1> (duration) and specify the time to be started above the instruction in <Operand2>.

The "Start extended pulse timer" instruction requires a preceding logic operation for edge evaluation and can only be placed at the right side of the network.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed here: AUTOHOTSPOT

Parameters

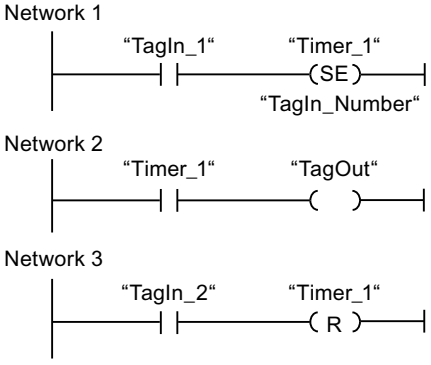
The following table shows the parameters of the "Start extended pulse timer" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	S5TIME, WORD	I, Q, M, D, L or constant	Duration with which the timer expires.
<Operand2>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

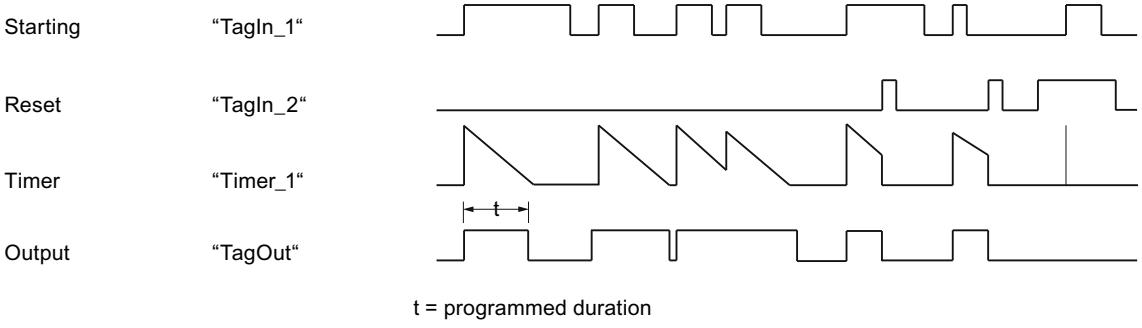
Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer expires with the time value of the operand "TagIn_Number" without being affected by a negative signal edge in the RLO. As long as the timer is running, the "TagOut" operand returns signal state "1". If the signal state of the operand "TagIn_1" changes again from "0" to "1" before the timer expires, the timer is restarted.

The figure below shows the timer diagram of the example:



For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(SD): Start on-delay timer (S7-1500)

Description

The "Start on-delay timer" instruction starts a programmed timer when the signal state "1" is detected at the start input. As long as this signal state remains "1", the timer expires with the specified duration. When the timer expires and the signal state at the start input is still "1", the query for timer status also returns "1". If the signal at the start input is "0", the timer is reset. In this case, querying the timer status returns the signal state "0". As soon as the signal at the start input changes to "1" again, the timer starts running again.

The signal state of the timer output is identical to the signal state of the start input. The start input is directly interconnected to the output and is not linked to the timer.

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down towards zero.

Specify the duration with which the time will run down at the operand placeholder <Operand1> (duration) and specify the time to be started above the instruction in <Operand2>.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed here: AUTOHOTSPOT

Parameters

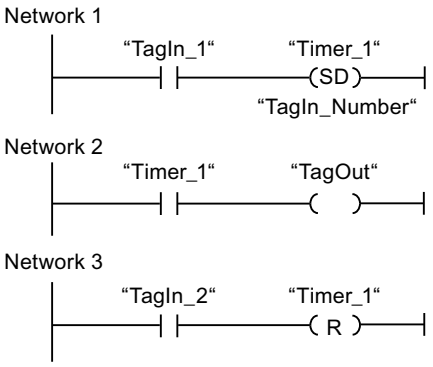
The following table shows the parameters of the "Start on-delay timer" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	S5TIME, WORD	I, Q, M, D, L or constant	Duration with which the timer expires.
<Operand2>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



Network 1:

"#Timer_1" starts when the signal state of the "#TagIn_1" operand changes from "0" to "1". The timer expires with the value of operand "#TagIn_Number". If the signal state of the "#TagIn_1" operand changes from "1" to "0" before the timer expires, the timer is reset.

Network 2:

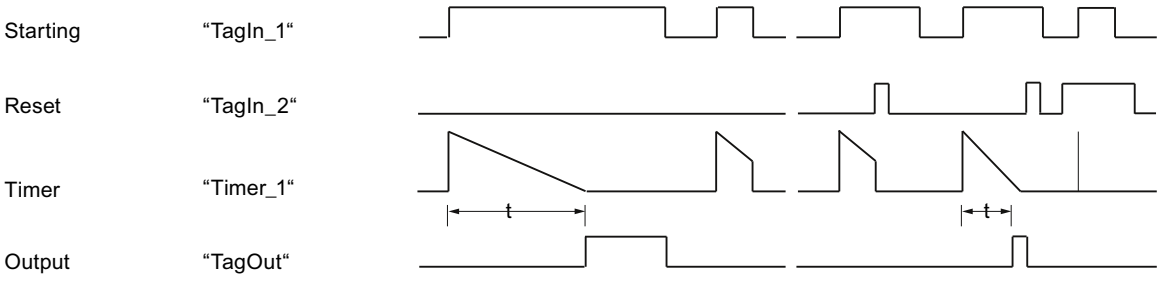
When the timer has expired, the #TagIn_1 operand at the start input has signal state "1" and the timer was not reset, the #TagOut operand is "1".

Network 3:

When the signal state of the #TagIn_2 operand has signal state "1", the timer #Timer_1 and the output #TagOut are reset.

If you want to restart #Timer_1, the signal state at the #TagIn_2 operand must be "0" and the signal state at the start input #TagIn_1 must change from "0" to "1".

The figure below shows the timer diagram of the example:



t = programmed duration

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(SS): Start retentive on-delay timer (S7-1500)

Description

The "Start retentive on-delay timer" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO). The timer expires with the specified duration, even if the RLO changes to the signal state "0". When the timer expires, the querying of the timer status for "1" returns the signal state "1". After expiry of the timer, the timer can only be restarted if it is explicitly reset.

The duration is made up internally of a time value and a time base. When the instruction is started, the programmed time value is counted down towards zero. The time base indicates the time period with which the time value is changed.

Specify the duration with which the time will run down at the operand placeholder <Operand1> (duration) and specify the time to be started above the instruction in <Operand2>.

The "Start retentive on-delay timer" instruction requires a preceding logic operation for edge evaluation and can only be placed at the right side of the network.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed here: AUTOHOTSPOT

Parameters

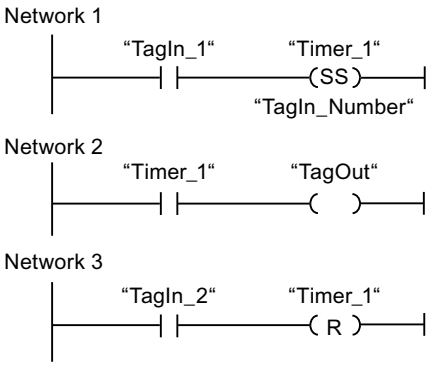
The following table shows the parameters of the "Start retentive on-delay timer" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	S5TIME, WORD	I, Q, M, D, L or constant	Duration with which the timer expires.
<Operand2>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

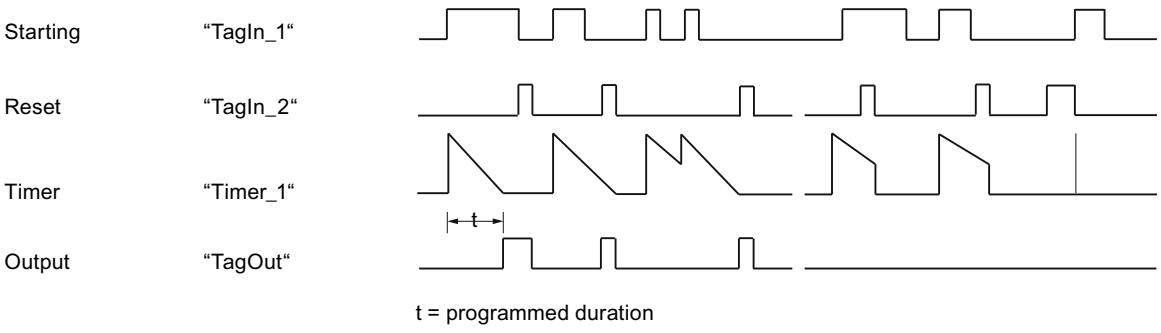
Example

The following example shows how the instruction works:



The timer "Timer_1" is started when the signal state of the operand "TagIn_1" changes from "0" to "1". The timer expires with the time value of the operand "TagIn_Number". When the timer expires, the operand "TagOut" is set to "1". If the signal state of the operand "TagIn_1" changes from "0" to "1" while the timer is running, the timer is restarted. If the signal state of the operand "TagIn_2" is "1", the timer "Timer_1" is reset, which stops the timer and sets the current time value to "0".

The figure below shows the timer diagram of the example:



For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(SF): Start off-delay timer (S7-1500)

Description

The "Start off-delay timer" instruction starts a programmed timer when a change from "1" to "0" (negative signal edge) is detected in the result of logic operation (RLO). The timer expires with the specified duration. As long as the timer is running, the query of timer status "1" returns the signal state "1". If the RLO changes from "0" to "1" while the timer is running, the timer is reset. The timer is always restarted when the RLO changes from "1" to "0".

The duration is made up internally of a time value and a time base. When the instruction is started, the programmed time value is counted down towards zero. The time base indicates the time period with which the time value is changed.

The queries of the timer status for "1" return the query result "1" if the signal state of the result of logic operation is "1" at the execution of the instruction. If the RLO is "0", queries of the timer status for "1" return the query result "0".

Specify the duration with which the time will run down at the operand placeholder <Operand1> (duration) and specify the time to be started above the instruction in <Operand2>.

The "Start off-delay timer" instruction requires a preceding logic operation for edge evaluation and can only be placed at the right side of the network.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

For an example of how a time cell can be structured, refer to See also "L: Load timer value".

Parameters

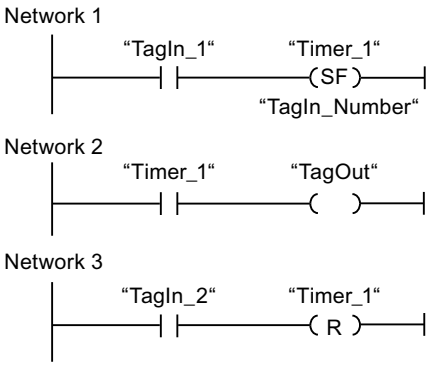
The following table shows the parameters of the instruction "Start off-delay timer":

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	S5TIME, WORD	I, Q, M, D, L or constant	Duration with which the timer expires.
<Operand2>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

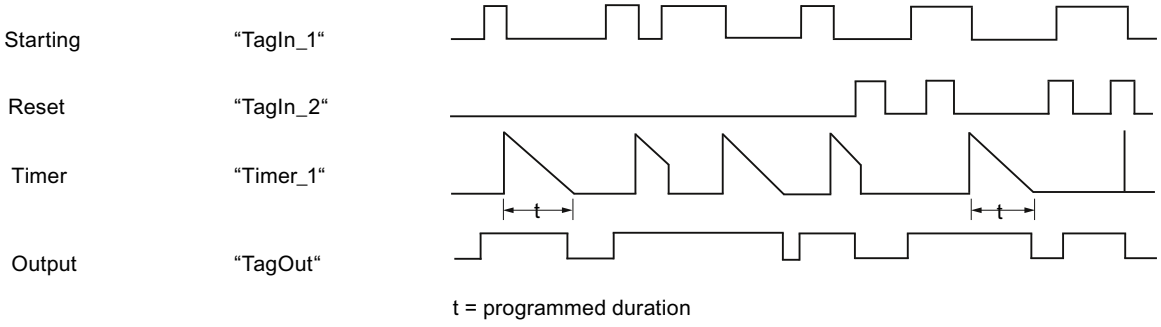
Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "1" to "0". The timer expires with the time value of the operand "TagIn_Number". As long as the timer is running, the "TagOut" operand is set to "1". If the signal state at the "TagIn_1" operand changes from "1" to "0" while the timer is running, the timer is restarted. If the signal state of the "TagIn_2" operand is "1", "Timer_1" is reset, i.e. the timer is stopped and the current time value is set to "0".

The figure below shows the timer diagram of the example:



t = programmed duration

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Counter operations (S7-1200, S7-1500)

CTU: Count up (S7-1200, S7-1500)

Description

You can use the "Count up" instruction to increment the value at output CV. When the signal state at the CU input changes from "0" to "1" (positive signal edge), the instruction is executed and the current counter value at the CV output is incremented by one. The counter value is incremented each time a positive signal edge is detected, until it reaches the high limit for the data type specified at the output CV. When the high limit is reached, the signal state at the CU input no longer has an effect on the instruction.

You can query the counter status in the Q output. The signal state at the Q output is determined by the PV parameter. If the current counter value is greater than or equal to the value of the PV parameter, the Q output is set to signal state "1". In all other cases, the Q output has signal state "0".

The value at the CV output is reset to zero when the signal state at input R changes to "1". As long as the R input has signal state "1", the signal state at the CU input has no effect on the instruction.

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count up" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For S7-1200 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER	• CTU_SINT / CTU_USINT • CTU_INT / CTU_UINT • CTU_DINT / CTU_UDINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER

For S7-1500 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER	• CTU_SINT / CTU_USINT • CTU_INT / CTU_UINT • CTU_DINT / CTU_UDINT • CTU_LINT / CTU_ULINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER

You can declare an IEC counter as follows:

- Declaration of a data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER")
- Declaration as a local tag of the type CTU_<Data type> or IEC_<Counter> in the "Static" section of a block (for example #MyIEC_COUNTER)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive. For additional information on setting retentivity in an instance data block, refer to "See also".

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

The execution of the "Count up" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

Parameters

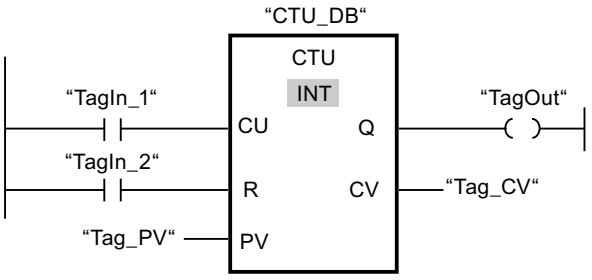
The following table shows the parameters of the "Count up" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
CU	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Count input
R	Input	BOOL	I, Q, M, D, L, P or constant	I, Q, M, T, C, D, L, P or constant	Reset input
PV	Input	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Value at which the output Q is set.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Counter status
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	I, Q, M, D, L, P	Current counter value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



When the signal state of the "TagIn_1" operand changes from "0" to "1", the "Count up" instruction is executed and the current counter value of the "Tag_CV" operand is incremented by one. With each additional positive signal edge, the counter value is incremented until the high limit of the data type (INT = 32767) is reached.

The value of the PV parameter is adopted as the limit for determining the "TagOut" output. The "TagOut" output has signal state "1" as long as the current counter value is greater than or equal to the value of the "Tag_PV" operand. In all other cases, the "TagOut" output has signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Setting the retentivity of local tags (Page 4769)

Setting retentivity in an instance data block (Page 4809)

Basic information on LAD (Page 4878)

CTD: Count down (S7-1200, S7-1500)

Description

You can use the "Count down" instruction to decrement the value at output CV. When the signal state at the CD input changes from "0" to "1" (positive signal edge), the instruction is executed and the current counter value at the CV output is decremented by one. Each time a positive signal edge is detected, the counter value is decremented until it reaches the low limit of the specified data type. When the low limit is reached, the signal state at the CD input no longer has an effect on the instruction.

You can query the counter status in the Q output. If the current counter value is less than or equal to zero, the Q output is set to signal state "1". In all other cases, the Q output has signal state "0".

The value at the CV output is set to the value of the PV parameter when the signal state at the LD input changes to "1". As long as the LD input has signal state "1", the signal state at the CD input has no effect on the instruction.

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count down" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For S7-1200 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER	• CTD_SINT / CTD_USINT • CTD_INT / CTD_UINT • CTD_DINT / CTD_UDINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER

For S7-1500 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER	• CTD_SINT / CTD_USINT • CTD_INT / CTD_UINT • CTD_DINT / CTD_UDINT • CTD_LINT / CTD_ULINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER

You can declare an IEC counter as follows:

- Declaration of a data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER")
- Declaration as a local tag of the type CTD_<Data type> or IEC_<Counter> in the "Static" section of a block (for example #MyIEC_COUNTER)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive. For additional information on setting retentivity in an instance data block, refer to "See also".

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

The execution of the "Count down" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

Parameters

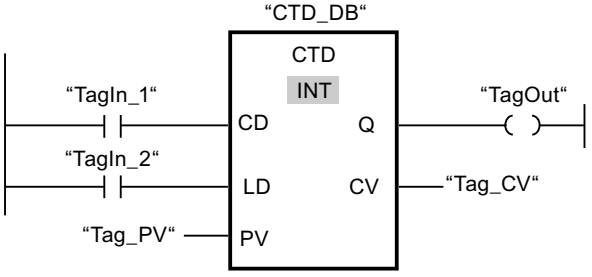
The following table shows the parameters of the "Count down" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
CD	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Count input
LD	Input	BOOL	I, Q, M, D, L, P or constant	I, Q, M, T, C, D, L, P or constant	Load input
PV	Input	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Value to which the CV output is set with LD = 1.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Counter status
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	I, Q, M, D, L, P	Current counter value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



When the signal state of the "TagIn_1" operand changes from "0" to "1", the instruction is executed and the value at the "Tag_CV" output is decremented by one. With each additional positive signal edge, the counter value is decremented until the low limit of the specified data type (INT = -32768) is reached.

The "TagOut" output has signal state "1" as long as the current counter value is less than or equal to zero. In all other cases, the "TagOut" output has signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Setting retentivity in an instance data block (Page 4809)

Setting the retentivity of local tags (Page 4769)

Basic information on LAD (Page 4878)

CTUD: Count up and down (S7-1200, S7-1500)

Description

You can use the "Count up and down" instruction to increment and decrement the counter value at the CV output. If the signal state at the CU input changes from "0" to "1" (positive signal edge), the current counter value is incremented by one and stored at the CV output. If the signal state at the CD input changes from "0" to "1" (positive signal edge), the counter value at the CV output is decremented by one. If there is a positive signal edge at the CU and CD inputs in one program cycle, the current counter value at the CV output remains unchanged.

The counter value can be incremented until it reaches the high limit of the data type specified at the CV output. When the high limit is reached, the counter value is no longer incremented on a positive signal edge. The counter value is no longer decremented once the low limit of the specified data type has been reached.

When the signal state at the LD input changes to "1", the counter value at the CV output is set to the value of the PV parameter. As long as the LD input has signal state "1", the signal state at the CU and CD inputs has no effect on the instruction.

The counter value is set to zero when the signal state at the R input changes to "1". As long as the R input has signal state "1", a change in the signal state of the CU, CD and LD inputs has no effect on the "Count up and down" instruction.

You can query the status of the up counter at the QU output. If the current counter value is greater than or equal to the value of the PV parameter, the QU output is set to signal state "1". In all other cases, the QU output has signal state "0".

You can query the status of the down counter at the QD output. If the current counter value is less than or equal to zero, the QD output is set to signal state "1". In all other cases, the QD output has signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count up and down" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For S7-1200 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER	• CTUD_SINT / CTUD_USINT • CTUD_INT / CTUD_UINT • CTUD_DINT / CTUD_UDINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER

For S7-1500 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER	• CTUD_SINT / CTUD_USINT • CTUD_INT / CTUD_UINT • CTUD_DINT / CTUD_UDINT • CTUD_LINT / CTUD_ULINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER

You can declare an IEC counter as follows:

- Declaration of a data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER")
- Declaration as a local tag of the type CTUD_<Data type> or IEC_<Counter> in the "Static" section of a block (for example #MyIEC_COUNTER)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive. For additional information on setting retentivity in an instance data block, refer to "See also".

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

The execution of the "Count up and down" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

Parameters

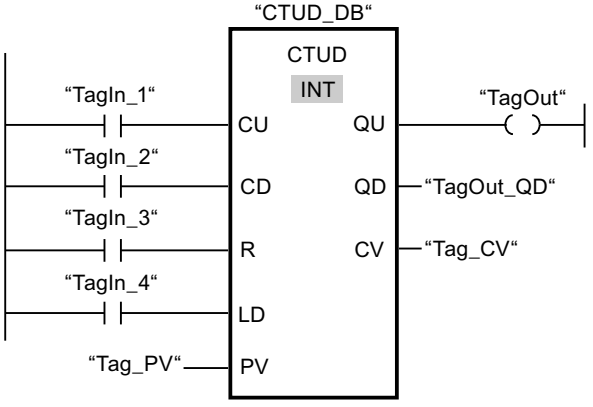
The following table shows the parameters of the "Count up and down" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
CU	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Count up input
CD	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Count down input
R	Input	BOOL	I, Q, M, D, L, P or constant	I, Q, M, T, C, D, L, P or constant	Reset input
LD	Input	BOOL	I, Q, M, D, L, P or constant	I, Q, M, T, C, D, L, P or constant	Load input
PV	Input	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Value at which the QU output is set / value to which the CV output is set when LD = 1.
QU	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Status of the up counter
QD	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Status of the down counter
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	I, Q, M, D, L, P	Current counter value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



If the signal state at the "TagIn_1" or "TagIn_2" input changes from "0" to "1" (positive signal edge), the "Count up and down" instruction is executed. When there is a positive signal edge at the "TagIn_1" input, the current counter value is incremented by one and stored at the "Tag_CV" output. When there is a positive signal edge at the "TagIn_2" input, the counter value

is decremented by one and stored at the "Tag_CV" output. When there is a positive signal edge at the CU input, the counter value is incremented until it reaches the high limit of 32767. If input CD has a positive signal edge, the counter value is decremented until it reaches the low limit of INT = -32768.

The "TagOut" output has signal state "1" as long as the current counter value is greater than or equal to the value at the "Tag_PV" input. In all other cases, the "TagOut" output has signal state "0".

The "TagOut_QD" output has signal state "1" as long as the current counter value is less than or equal to zero. In all other cases, the "TagOut_QD" output has signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Setting retentivity in an instance data block (Page 4809)

Setting the retentivity of local tags (Page 4769)

Example of detecting the fill level of a storage area (Page 4928)

Basic information on LAD (Page 4878)

Legacy (S7-1500)

S_CU: Assign parameters and count up (S7-1500)

Description

You can use the "Assign parameters and count up" instruction to increment the value of a counter. If the signal state at the CU input changes from "0" to "1" (positive signal edge), the current counter value is incremented by one. The current counter value is output as a hexadecimal value at output CV and BCD-coded at output CV_BCD. The counter value can be incremented until the limit of "999" is reached. When the limit is reached, the counter value is no longer incremented on a positive signal edge.

When the signal state at input S changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if RLO at input CU is "1", the counter will count accordingly in the next scan cycle, even when no change has been detected in the signal edge.

The counter value is set to zero when the signal state at the R input changes to "1". As long as the R input has the signal state "1", processing of the signal state of the CU and S inputs has no effect on the counter value.

The signal state at output Q is "1" if the counter value is greater than zero. If the counter value is equal to zero, output Q has the signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

The "Assign parameters and count up" instruction requires a preceding logic operation for edge evaluation and can be placed within or at the end of the network.

Parameters

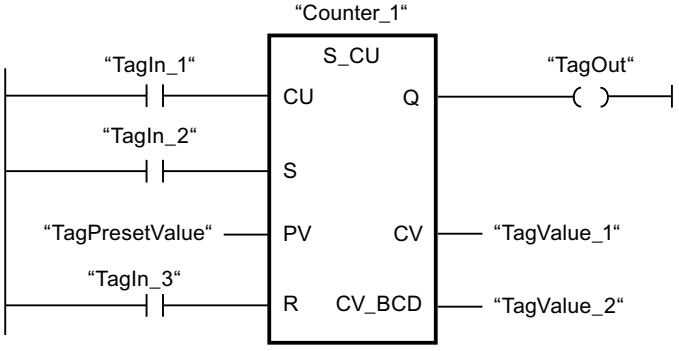
The following table shows the parameters of the "Assign parameters and count up" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Counter>	InOut/Input	COUNTER	C	Counter of the instruction The number of counters depends on the CPU.
CU	Input	BOOL	I, Q, M, D, L	Count up input
S	Input	BOOL	I, Q, M, D, L, T, C or constant	Input for presetting counter
PV	Input	WORD	I, Q, M, D, L or constant	Preset counter value (C#0 to C#999)
R	Input	BOOL	I, Q, M, D, L, T, C or constant	Reset input
BI	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (hexadecimal)
BCD	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the counter

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



When the signal state at the "TagIn_1" input changes from "0" to "1" (positive signal edge) and the current counter value is less than "999", the counter value is incremented by one. When the signal state at input "TagIn_2" changes from "0" to "1", the counter value is set to the value of the operand "TagPresetValue". The counter value is reset to "0" when the "TagIn_3" operand has signal state "1".

The current counter value is saved as a hexadecimal value in the operand "TagValue_1" and BCD-coded in the operand "TagValue_2".

The "TagOut" output has the signal state "1" as long as the current counter value is not equal to "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

S_CD: Assign parameters and count down (S7-1500)

Description

You can use the "Assign parameters and count down" instruction to decrement the value of a counter. If the signal state at the CD input changes from "0" to "1" (positive signal edge), the counter value is decremented by one. The current counter value is output as a hexadecimal value at output CV and BCD-coded at output CV_BCD. The counter value is decremented until the low limit of "0" is reached. When the low limit is reached, the counter value is no longer decremented on a positive signal edge.

When the signal state at input S changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if RLO at input CD is "1", the counter will count accordingly in the next scan cycle, even when no change has been detected in the signal edge.

The counter value is set to zero when the signal state at the R input changes to "1". As long as the R input has the signal state "1", processing of the signal state of the CD and S inputs has no effect on the counter value.

The signal state at output Q is "1" if the counter value is greater than zero. If the counter value is equal to zero, output Q has the signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

The "Assign parameters and count down" instruction requires a preceding logic operation for edge evaluation and can be placed within or at the end of the network.

Parameters

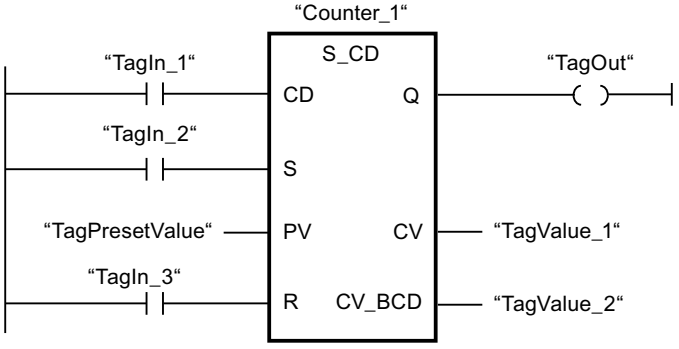
The following table shows the parameters of the "Assign parameters and count down" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Counter>	InOut/Input	COUNTER	C	Counter of the instruction The number of counters depends on the CPU.
CD	Input	BOOL	I, Q, M, D, L or constant	Count down input
S	Input	BOOL	I, Q, M, D, L, T, C or constant	Input for presetting counter
PV	Input	WORD	I, Q, M, D, L or constant	Preset counter value (C#0 to C#999)
R	Input	BOOL	I, Q, M, D, L, T, C or constant	Reset input
BI	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (hexadecimal)
BCD	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the counter

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



When the signal state at the "TagIn_1" input changes from "0" to "1" (positive signal edge) and the current counter value is greater than "0", the counter value is decremented by one. When the signal state at input "TagIn_2" changes from "0" to "1", the counter value is set to the value of the operand "TagPresetValue". The counter value is reset to "0" when the "TagIn_3" operand has signal state "1".

The current counter value is saved as a hexadecimal value in the operand "TagValue_1" and BCD-coded in the operand "TagValue_2".

The "TagOut" output has the signal state "1" as long as the current counter value is not equal to "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

S_CUD: Assign parameters and count up / down (S7-1500)

Description

You can use the "Assign parameters and count up / down" instruction to increment or decrement the value of a counter. If the signal state at the CU input changes from "0" to "1" (positive signal edge), the current counter value is incremented by one. If the signal state at the CD input changes from "0" to "1" (positive signal edge), the counter value is decremented by one. The current counter value is output as a hexadecimal value at output CV and BCD-coded at output CV_BCD. If there is a positive signal edge at the CU and CD inputs in one program cycle, the counter value remains unchanged.

The counter value can be incremented until the high limit of "999" is reached. When the high limit value is reached, the counter value is no longer incremented on a positive signal edge. When the low limit "0" is reached, the counter value is not decremented any further.

When the signal state at input S changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if RLO is "1" at the inputs CU and CD, the counter counts accordingly in the next scan cycle, even if no change in the signal edge was detected.

The counter value is set to zero when the signal state at the R input changes to "1". As long as the R input has the signal state "1", processing of the signal state of the CU, CD and S inputs has no effect on the counter value.

The signal state at output Q is "1" if the counter value is greater than zero. If the counter value is equal to zero, output Q has the signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

The "Assign parameters and count up / down" instruction requires a preceding logic operation for edge evaluation and can be placed within or at the end of the network.

Parameters

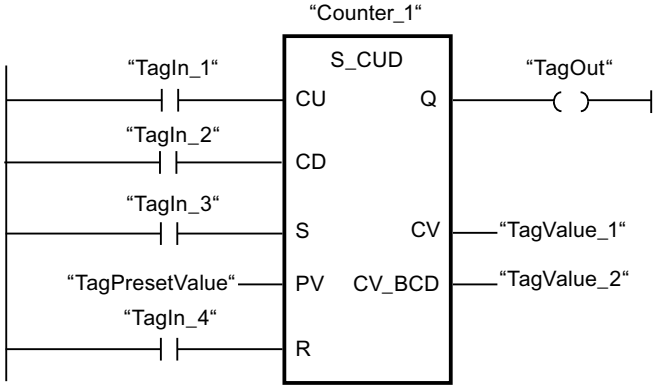
The following table shows the parameters of the "Assign parameters and count up / down" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Counter>	InOut/Input	COUNTER	C	Counter of the instruction The number of counters depends on the CPU.
CU	Input	BOOL	I, Q, M, D, L	Count up input
CD	Input	BOOL	I, Q, M, D, L, T, C or constant	Count down input
S	Input	BOOL	I, Q, M, D, L, T, C or constant	Input for presetting counter
PV	Input	WORD	I, Q, M, D, L or constant	Preset counter value (C#0 to C#999)
R	Input	BOOL	I, Q, M, D, L, T, C or constant	Reset input
BI	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (hexadecimal)
BCD	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the counter

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the signal state at the "TagIn_1"" or "TagIn_2" input changes from "0" to "1" (positive signal edge), the "Assign parameters and count up / down" instruction is executed. When there is a positive signal edge at the "TagIn_1" input and the current counter value is less than "999", the counter value is incremented by one. When there is a positive signal edge at the "TagIn_2" input and the current counter value is greater than "0", the counter value is decremented by one.

When the signal state at input "TagIn_3" changes from "0" to "1", the counter value is set to the value of the operand "TagPresetValue". The counter value is reset to "0" when the "TagIn_4" operand has signal state "1".

The current counter value is saved as a hexadecimal value in the operand "TagValue_1" and BCD-coded in the operand "TagValue_2".

The "TagOut" output has the signal state "1" as long as the current counter value is not equal to "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(SC): Set counter value (S7-1500)

Description

You can use the "Set counter value" instruction to set the value of a counter. The instruction is executed when the result of logic operation (RLO) at the input changes from "0" to "1". When the instruction is executed, the counter is set to the specified counter value.

At the operand placeholder <Operand1> below the instruction, specify the duration with which counter is preset and specify the counter above the instruction in <Operand2>.

The "Set counter value" instruction needs a preceding logic operation for the edge evaluation and can only be placed on the right edge of the network.

Parameters

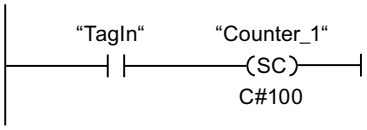
The following table lists the parameters of the "Set counter value" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	WORD	I, Q, M, D, L or constant	Value with which the counter is preset in the BCD format. (C#0 to C#999)
<Operand2>	InOut/Input	COUNTER	C	Counter that is preset.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



the counter "Counter_1" starts with the value "100" when the signal state of the "TagIn" operand changes from "0" to "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(CU): Count up (S7-1500)

Description

With the "Count up" instruction you can increment the value of the specified counter by one if there is a positive signal edge in the result of logic operation (RLO). The counter value can be incremented until the limit of "999" is reached. When the limit is reached, the counter value is no longer incremented on a positive signal edge.

The "Count up" instruction requires a preceding logic operation for edge evaluation and can only be placed at the right side of the network.

Parameters

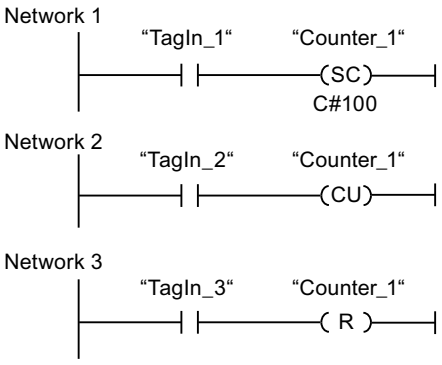
The following table shows the parameters of the "Count up" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Counter>	InOut/Input	COUNTER	C	Counter whose value is incremented.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



When the signal state of the "TagIn_1" operand changes from "0" to "1" (positive signal edge), the counter "Counter_1" is preset with the value "100".

The value of the counter "Counter_1" is incremented by one when the signal state of the operand "TagIn_2" changes from "0" to "1".

When the operand "TagIn_3" has the signal state "1", the value of the counter "Counter_1" is reset to "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(CD): Count down (S7-1500)

Description

With the "Count down" instruction you can increment the value of the specified counter by one if there is a positive signal edge in the result of logic operation (RLO). The counter value can be decremented until the limit of "0" is reached. When the limit is reached, the counter value is no longer changed on a positive signal edge.

The "Count down" instruction requires a preceding logic operation for edge evaluation and can only be placed at the right side of the network.

Parameters

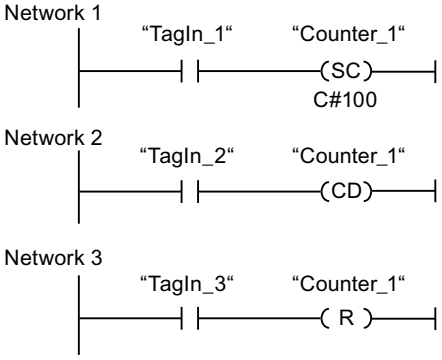
The following table shows the parameters of the "Count down" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Counter>	InOut/Input	COUNTER	C	Counter whose value is decremented.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



When the signal state of the "TagIn_1" operand changes from "0" to "1" (positive signal edge), the counter "Counter_1" is preset with the value "100".

The value of the counter "Counter_1" is decremented by one when the signal state of the operand "TagIn_2" changes from "0" to "1".

When the operand "TagIn_3" has the signal state "1", the value of the counter "Counter_1" is reset to "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Comparator operations (S7-1200, S7-1500)

CMP ==: Equal (S7-1200, S7-1500)

Description

You can use the "Equal" instruction to determine if a first comparison value (<Operand1>) is equal to a second comparison value (<Operand2>).

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify the first comparison value (<Operand1>) at the operand placeholder above the instruction. Specify the second comparison value (<Operand2>) at the operand placeholder below the instruction.

If IEC check is enabled, the operands to be compared must be of the same data type. If IEC check is not enabled, the width of the operands must be the same.

Comparison of floating-point numbers

If you want to compare the data types REAL or LREAL, then instead of the instruction "CMP ==: Equal" we recommend using the instruction "IN_RANGE: Value within range".

When floating-point numbers are compared, the operands to be compared must have the same data type regardless of the setting for the IEC Check.

The special bit patterns of invalid floating-point numbers (NaN) that are the outcome of undefined results (e.g. root of -1) are not comparable. This means that if one of the operands has the value NaN, the instruction "CMP ==: Equal" has the result FALSE.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison.

The following table shows examples of string comparisons:

<Operand1>	<Operand2>	RLO of the instruction
'AA'	'AA'	1
'Hello World'	'HelloWorld'	0
'AA'	'aa'	0
'aa'	'aaa'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

Bit patterns of invalid timers, date and times, e.g. DT#2015-13-33-25:62:99.999_999_999, cannot be compared. This means that if one of the operands has an invalid value, the instruction "CMP ==: Equal" has the result FALSE.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Comparison of hardware data types

To be able to compare the operands of the PORT data type, you need to select the WORD data type from the drop-down list in the instruction box.

If you want to compare the two hardware data types HW_IO and HW_DEVICE, you must first create a tag of data type HW_ANY in the section "Temp" of the block interface and then copy the LADDR (of data type HW_DEVICE) to the tag. It is then possible to compare HW_ANY and HW_IO.

Comparison of structures

Note

Availability of comparison of structures

The option to compare structures is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

You can compare the values of two structured operands when both tags are of the same structure data type. When structures are compared, the operands to be compared must have the same data type regardless of the setting for the IEC Check. An exception is comparisons in which one of the two operands is a VARIANT or an ANY. If the data type is not yet known when the program is created, you can use the VARIANT data type. In this case you can also compare the operand with a structured tag of any data type. You can also compare two tags of the data type VARIANT or ANY with each other.

You can select the data type VARIANT for the comparison of structures from the drop-down list in the instruction box. The following data types are possible:

- PLC data type
- STRUCT (the structure of the data type STRUCT must be contained in a PLC data type (UDT) or the two structures to be compared are two elements of an ARRAY of STRUCT. Anonymous structures are not permitted.)
- Tag to which ANY is pointing
- Tag to which VARIANT is pointing

The following requirements must be met to compare two tags of the data type ARRAY with VARIANT selected:

- The elements must each have the same data type.
- The two ARRAYs must have the same dimension.
- All dimensions must have the same number of elements. The exact ARRAY limits do not have to match.

Note

ARRAY of BOOL

When you compare two operands of the data type ARRAY of BOOL with each other, you have selected the data type VARIANT from the drop-down list, and the number of elements cannot be divided by 8, the fill bits are compared as well. This may affect the comparison result.

If an element within the structures that you want to compare is an invalid STRING/WSTRING, an invalid time or date or an invalid floating-point number, the comparison result in the RLO (result of logic operation) will return the signal state "0".

The table below shows an example of a comparison of structures:

<Operand1>			<Operand2>			RLO of the instruction
Tag of data type A <PLC data type>		Tag value	Tag of data type A <PLC data type>		Tag value	1
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

<Operand1>			<Operand2>			RLO of the instruction
Tag of data type A <PLC data type>		Tag value	Tag of data type B <PLC data type>		Tag value	0
	BOOL	FALSE		BOOL	TRUE	
	INT	2		INT	3	

<Operand1>			<Operand2>			RLO of the instruction
Tag of data type A <PLC data type>		Tag value	VARIANT (supplied with tag of data type A)		Tag value	1
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

Parameters

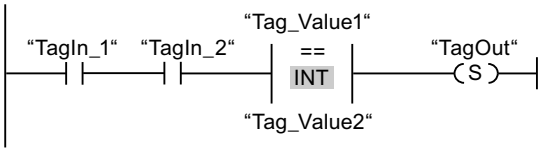
The following table shows the parameters of the instruction "Equal":

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time, ARRAY of <data type> with fixed and variable ARRAY limits, STRUCT, VARIANT, ANY, PLC data type	I, Q, M, D, L, P, or constant	First comparison value
<Operand2>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time, ARRAY of <data type> with fixed and variable ARRAY limits, STRUCT, VARIANT, ANY, PLC data type	I, Q, M, D, L, P, or constant	Second value to compare
As detailed above, the data types ARRAY, STRUCT (in a PLC data type), VARIANT, ANY and PLC data type (UDT) are only available as of firmware version 2.0 or 4.2.				

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" = "Tag_Value2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

CMP <>: Not equal (S7-1200, S7-1500)

Description

You can use the "Not equal" instruction to determine if a first comparison value (<Operand1>) is not equal to a second comparison value (<Operand2>).

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify the first comparison value (<Operand1>) at the operand placeholder above the instruction. Specify the second comparison value (<Operand2>) at the operand placeholder below the instruction.

If IEC check is enabled, the operands to be compared must be of the same data type. If IEC check is not enabled, the width of the operands must be the same.

Comparison of floating-point numbers

If you want to compare the data types REAL or LREAL, then instead of the instruction "CMP <>: Not equal" we recommend using the instruction "OUT_RANGE: Value outside range".

If floating-point numbers are being compared, the operands to be compared must be of the same data type regardless of the IEC check setting.

The special bit patterns of invalid floating-point numbers (NaN) that are the outcome of undefined results (e.g. root of -1) are not comparable. This means that if one of the operands has the value NaN, the instruction "CMP <>: Not equal" has the result FALSE.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison.

The following table shows examples of string comparisons:

<Operand1>	<Operand2>	RLO of the instruction
'AA'	'aa'	1
'Hello World'	'HelloWorld'	1
'AA'	'AA'	0
'aa'	'aaa'	1

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

Bit patterns of invalid timers, date and times, e.g. DT#2015-13-33-25:62:99.999_999_999, cannot be compared. This means that if one of the operands has an invalid value, the instruction "CMP <>: Not equal" has the result FALSE.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Comparison of hardware data types

To be able to compare the operands of the PORT data type, you need to select the WORD data type from the drop-down list in the instruction box.

If you want to compare the two hardware data types HW_IO and HW_DEVICE, you must first create a tag of data type HW_ANY in the section "Temp" of the block interface and then copy the LADDR (of data type HW_DEVICE) to the tag. It is then possible to compare HW_ANY and HW_IO.

Comparison of structures

Note

Availability of comparison of structures

The option to compare structures is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

You can compare the values of two structured operands when both tags are of the same structure data type. When structures are compared, the operands to be compared must have the same data type regardless of the setting for the IEC Check. An exception is comparisons in which one of the two operands is a VARIANT or an ANY. If the data type is not yet known when the program is created, you can use the VARIANT data type. In this case you can also

compare the operand with a structured tag of any data type. You can also compare two tags of the data type VARIANT or ANY with each other.

You can select the data type VARIANT for the comparison of structures from the drop-down list in the instruction box. The following data types are possible:

- PLC data type
- STRUCT (the structure of the data type STRUCT must be contained in a PLC data type (UDT) or the two structures to be compared are two elements of an ARRAY of STRUCT. Anonymous structures are not permitted.)
- Tag to which ANY is pointing
- Tag to which VARIANT is pointing

The following requirements must be met to compare two tags of the data type ARRAY with VARIANT selected:

- The elements must each have the same data type.
- The two ARRAYs must have the same dimension.
- All dimensions must have the same number of elements. The exact ARRAY limits do not have to match.

Note

ARRAY of BOOL

When you compare two operands of the data type ARRAY of BOOL with each other, you have selected the data type VARIANT from the drop-down list, and the number of elements cannot be divided by 8, the fill bits are compared as well. This may affect the comparison result.

If an element within the structures that you want to compare is an invalid STRING/WSTRING, an invalid time or date or an invalid floating-point number, the comparison result in the RLO (result of logic operation) will return the signal state "0".

The table below shows an example of a comparison of structures:

<Operand1>			<Operand2>			RLO of the instruction
Tag of data type A <PLC data type>		Tag value	Tag of data type A <PLC data type>		Tag value	0
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

<Operand1>			<Operand2>			RLO of the instruction
Tag of data type A <PLC data type>		Tag value	Tag of data type B <PLC data type>		Tag value	1
	BOOL	FALSE		BOOL	TRUE	
	INT	2		INT	3	

<Operand1>			<Operand2>			RLO of the instruction
Tag of data type A <PLC data type>		Tag value	VARIANT (supplied with tag of data type A)		Tag value	0
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

Parameters

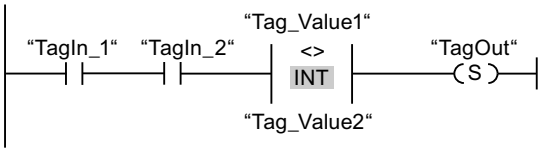
The following table shows the parameters of the "Not equal" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time, ARRAY of <data type> with fixed and variable ARRAY limits, STRUCT, VARIANT, ANY, PLC data type	I, Q, M, D, L, P, or constant	First comparison value
<Operand2>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time, ARRAY of <data type> with fixed and variable ARRAY limits, STRUCT, VARIANT, ANY, PLC data type	I, Q, M, D, L, P, or constant	Second value to compare
As detailed above, the data types ARRAY, STRUCT (in a PLC data type), VARIANT, ANY and PLC data type (UDT) are only available as of firmware version 2.0 or 4.2.				

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" <> "Tag_Value2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

CMP >=: Greater or equal (S7-1200, S7-1500)

Description

You can use the "Greater or equal" instruction to determine if a first comparison value (<Operand1>) is greater than or equal to a second comparison value (<Operand2>). Both values to be compared must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify the first comparison value (<Operand1>) at the operand placeholder above the instruction. Specify the second comparison value (<Operand2>) at the operand placeholder below the instruction.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the longer string is considered greater.

The following table shows examples of string comparisons:

<Operand1>	<Operand2>	RLO of the instruction
'BB'	'AA'	1
'AAA'	'AA'	1
'Hello World'	'Hello World'	1
'Hello World'	'HelloWorld'	0
'AA'	'aa'	0
'AAA'	'a'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

When time values are compared, the RLO of the instruction is "1" if the point of time at <Operand1> is greater (more recent) than or equal to the point of time at <Operand2>.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Parameters

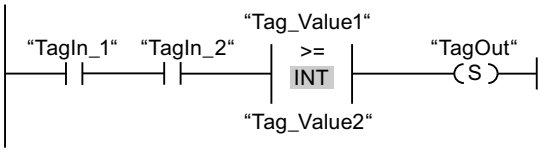
The following table shows the parameters of the "Greater or equal" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	First comparison value
<Operand2>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	Second value to compare

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" >= "Tag_Value2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Example of detecting the fill level of a storage area (Page 4928)

Basic information on LAD (Page 4878)

CMP <=: Less or equal (S7-1200, S7-1500)

Description

You can use the "Less or equal" instruction to determine if a first comparison value (<Operand1>) is less than or equal to a second comparison value (<Operand2>). Both values to be compared must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify the first comparison value (<Operand1>) at the operand placeholder above the instruction. Specify the second comparison value (<Operand2>) at the operand placeholder below the instruction.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the shorter string is considered smaller.

The following table shows examples of string comparisons:

<Operand1>	<Operand2>	RLO of the instruction
'AA'	'aa'	1
'AAA'	'a'	1
'Hello World'	'Hello World'	1
'HelloWorld'	'Hello World'	0
'BB'	'AA'	0
'AAA'	'AA'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

When time values are compared, the RLO of the instruction is "1" if the point of time at <Operand1> is less (less recent) than or equal to the point of time at <Operand2>.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Parameters

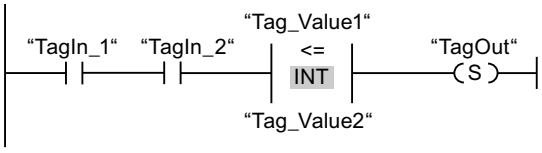
The following table shows the parameters of the "Less or equal" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	First comparison value
<Operand2>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	Second value to compare

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" <= "Tag_Value2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

CMP >: Greater than (S7-1200, S7-1500)

Description

You can use the "Greater than" instruction to determine if a first comparison value (<Operand1>) is greater than a second comparison value (<Operand2>). Both values to be compared must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify the first comparison value (<Operand1>) at the operand placeholder above the instruction. Specify the second comparison value (<Operand2>) at the operand placeholder below the instruction.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the longer string is considered greater.

The following table shows examples of string comparisons:

<Operand1>	<Operand2>	RLO of the instruction
'BB'	'AA'	1
'AAA'	'AA'	1
'AA'	'aa'	0
'AAA'	'a'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

When time values are compared, the RLO of the instruction is "1" if the point of time at <Operand1> is greater (more recent) than the point of time at <Operand2>.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Parameters

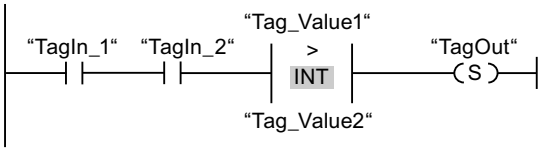
The following table shows the parameters of the "Greater than" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	First comparison value
<Operand2>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	Second value to compare

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" > "Tag_Value2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

CMP <: Less than (S7-1200, S7-1500)

Description

You can use the "Less than" instruction to determine if a first comparison value (<Operand1>) is less than a second comparison value (<Operand2>). Both values to be compared must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify the first comparison value (<Operand1>) at the operand placeholder above the instruction. Specify the second comparison value (<Operand2>) at the operand placeholder below the instruction.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the shorter string is considered smaller.

The following table shows examples of string comparisons:

<Operand1>	<Operand2>	RLO of the instruction
'AA'	'aa'	1
'AAA'	'a'	1
'BB'	'AA'	0
'AAA'	'AA'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

When time values are compared, the RLO of the instruction is "1" if the point of time at <Operand1> is less (less recent) than the point of time at <Operand2>.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Parameters

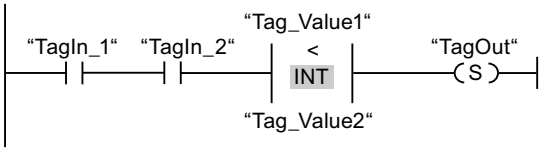
The following table shows the parameters of the "Less than" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	First comparison value
<Operand2>	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	Second value to compare

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" < "Tag_Value2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Example of detecting the fill level of a storage area (Page 4928)

Basic information on LAD (Page 4878)

IN_RANGE: Value within range (S7-1200, S7-1500)

Description

You can use the "Value within range" instruction to query whether of the value at input VAL is within a specific value range.

You specify the limits of the value range with the MIN and MAX inputs. The "Value within range" instruction compares the value at the VAL input with the values of the MIN and MAX inputs and sends the result to the box output. If the value at input VAL satisfies the comparison MIN <= VAL or VAL <=MAX, the box output has the signal state "1". If the comparison is not fulfilled, the box output has the signal state "0".

If the box input has the signal state "0", the "Value within range" instruction is not executed.

The comparison function can only execute if the values to be compared are of the same data type and the box input is interconnected.

Parameters

The following table shows the parameters of the "Value within range" instruction:

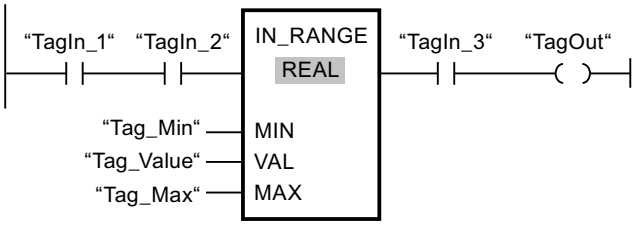
Parameters	Declaration	Data type	Memory area	Description
Box input	Input	BOOL	I, Q, M, D, L	Result of the previous logic operation
MIN	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Low limit of the value range
VAL	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Comparison value
MAX	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	High limit of the value range
Box output	Output	BOOL	I, Q, M, D, L	Result of the comparison

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also":

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The value of the operand "Tag_Value" is within the value range that is specified by the current values of the operands "Tag_Min" and "Tag_Max" (MIN <= VAL or VAL <= MAX).
- The operand "TagIn_3" has the signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

OUT_RANGE: Value outside range (S7-1200, S7-1500)

Description

You can use the "Value outside range" instruction to query whether of the value at input VAL is outside a specific value range.

You specify the limits of the value range with the MIN and MAX inputs. The "Value outside range" instruction compares the value at input VAL with the values of the inputs MIN and MAX and sends the result to the box output. If the value at input VAL satisfies the comparison MIN > VAL or VAL >MAX, the box output has the signal state "1". The box output also has the signal state "1" if a specified operand with the REAL data type shows an invalid value.

The box output returns the signal state "0", if the value at input VAL does not satisfy the MIN > VAL or VAL > MAX condition.

If the box input has the signal state "0", the "Value outside range" instruction is not executed.

The comparison function can only execute if the values to be compared are of the same data type and the box input is interconnected.

Parameters

The following table shows the parameters of the "Value outside range" instruction:

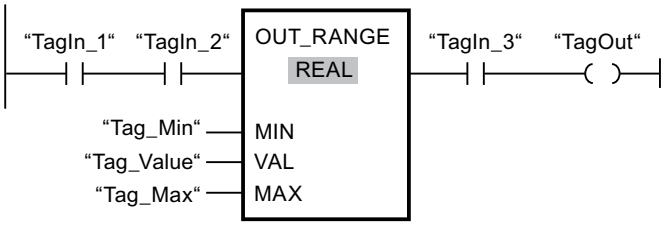
Parameters	Declaration	Data type	Memory area	Description
Box input	Input	BOOL	I, Q, M, D, L	Result of the previous logic operation
MIN	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Low limit of the value range
VAL	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Comparison value
MAX	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	High limit of the value range
Box output	Output	BOOL	I, Q, M, D, L	Result of the comparison

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The value of the operand "Tag_Value" is outside the value range that is specified by the values of the operands "Tag_Min" and "Tag_Max" (MIN > VAL or VAL > MAX).
- The operand "TagIn_3" has the signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

----I OK I----: Check validity (S7-1200, S7-1500)

Description

You can use the "Check validity" instruction to check if the value of an operand (<operand>) is a valid floating-point number. The query is started in each program cycle when the signal state at the input of the instruction is "1".

The output of the instruction has signal state "1" when the value of the operand is a valid floating-point number at the time of the query and the input of the instruction has signal state "1". In all other cases, the signal state at the output of the "Check validity" instruction is "0".

You can use the "Check validity" instruction together with the EN mechanism. If you connect the instruction box to an EN enable input, the enable input is set only when the result of the validity query of the value is positive. You can use this function to ensure that an instruction is enabled only when the value of the specified operand is a valid floating-point number.

Parameters

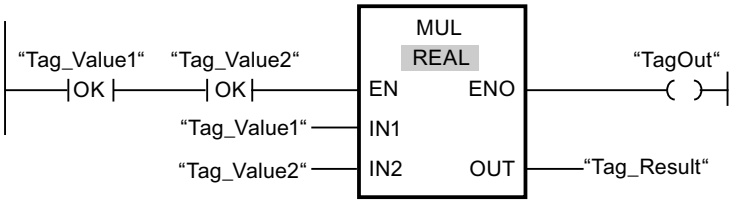
The following table shows the parameters of the "Check validity" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	Floating-point numbers	I, Q, M, D, L	Value to be queried.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



When the values of the operands "Tag_Value1" and "Tag_Value2" show valid floating-point numbers, the "Multiply" instruction is executed. The value of operand "Tag_Value1" is multiplied by the value of operand "Tag_Value2". The product of the multiplication is then written in the operand "Tag_Result". If the instruction is executed without errors, the TagOut operand at the "ENO" enable output is set to signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

----| NOT_OK |----: Check invalidity (S7-1200, S7-1500)

Description

You can use the "Check invalidity" instruction to check if the value of an operand (<operand>) is an invalid floating-point number. The query is started in each program cycle when the signal state at the input of the instruction is "1".

The output of the instruction has signal state "1" when the value of the operand is an invalid floating-point number at the time of the query and the input of the instruction has signal state "1". In all other cases, the signal state at the output of the "Check invalidity" instruction is "0".

Parameters

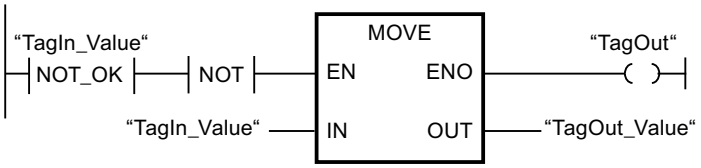
The following table shows the parameters of the instruction "Check invalidity":

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	Floating-point numbers	I, Q, M, D, L	Value to be queried.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



When the value of operand "TagIn_Value" is an invalid floating-point number, the instruction "Move value" is not executed. The "TagOut" operand at the ENO enable output is set to the signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

VARIANT (S7-1200, S7-1500)

EQ_Type: Compare data type for EQUAL with the data type of a tag (S7-1200, S7-1500)

Description

You can use the "Compare data type for EQUAL with the data type of a tag" instruction to query the data type of a tag to which a VARIANT points. You are comparing the data type of the tag (<Operand1>), which you declared in the block interface, with the data type of a tag (<Operand2>) for "Equal to".

<Operand1> must have the VARIANT data type. <Operand2> can be an elementary data type or a PLC data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify <Operand1> at the operand placeholder above the instruction. Specify <Operand2> at the operand placeholder below the instruction.

Comparison of timers, date and time

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparison of structures

For the comparison of structures, you can use the instruction "CMP ==". Anonymous structures cannot generally be compared unless they are part of the same ARRAY.

For additional information on the comparison of structures, please refer to: CMP ==: Equal (Page 2662)

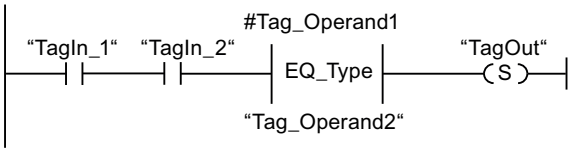
Parameters

The following table shows the parameters of the "Compare data type for EQUAL with the data type of a tag" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand1>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand
<Operand2>	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Second operand

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the #Tag_Operand1 operand is equal to "Tag_Operand2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

NE_Type: Compare data type for UNEQUAL with the data type of a tag (S7-1200, S7-1500)

Description

You can use the "Compare data type for UNEQUAL with the data type of a tag" instruction to query which data type a tag to which a VARIANT points does not have. You are comparing the data type of the tag (<Operand1>), which you declared in the block interface, with the data type of a tag (<Operand2>) for "Not equal to".

<Operand1> must have the VARIANT data type. <Operand2> can be an elementary data type or a PLC data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify <Operand1> at the operand placeholder above the instruction. Specify <Operand2> at the operand placeholder below the instruction.

Comparison of timers, date and time

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparison of structures

For the comparison of structures, you can use the instruction "CMP <>". Anonymous structures cannot generally be compared unless they are part of the same ARRAY.

For additional information on the comparison of structures, please refer to: CMP <>: Not equal (Page 2667)

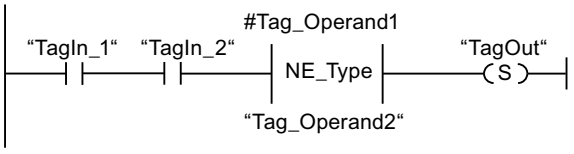
Parameters

The following table shows the parameters of the "Compare data type for UNEQUAL with the data type of a tag" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand1>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand
<Operand2>	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Second operand

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the #Tag_Operand1 operand is not equal to "Tag_Operand2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

EQ_ElemType: Compare data type of an ARRAY element for EQUAL with the data type of a tag (S7-1200, S7-1500)

Description

You can use the "Compare data type of an ARRAY element for EQUAL with the data type of a tag" instruction to query the data type of a tag to which a VARIANT points. You are comparing the data type of the tag (<Operand1>), which you declared in the block interface, with the data type of a tag (<Operand2>) for "Equal to".

<Operand1> must have the VARIANT data type. <Operand2> can be an elementary data type or a PLC data type.

If the data type of the VARIANT tag (<Operand1>) is an ARRAY, the data type of the ARRAY elements is compared.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify <Operand1> at the operand placeholder above the instruction. Specify <Operand2> at the operand placeholder below the instruction.

Comparison of timers, date and time

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparison of structures

For the comparison of structures, you can use the instruction "CMP ==". Anonymous structures cannot generally be compared unless they are part of the same ARRAY.

For additional information on the comparison of structures, please refer to: CMP ==: Equal (Page 2662)

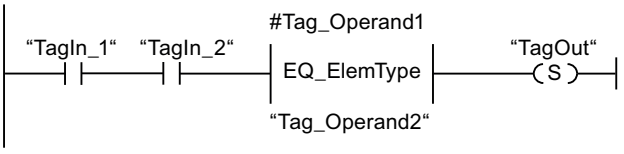
Parameters

The following table shows the parameters of the "Compare data type of an ARRAY element for EQUAL with the data type of a tag" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand1>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand
<Operand2>	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Second operand

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the #Tag_Operand1 operand is equal to "Tag_Operand2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

NE_ElemType: Compare data type of an ARRAY element for UNEQUAL with the data type of a tag (S7-1200, S7-1500)

Description

You can use the "Compare data type of an ARRAY element for UNEQUAL with the data type of a tag" instruction to query which data type a tag does not have to which a VARIANT points. You are comparing the data type of the tag (<Operand1>), which you declared in the block interface, with the data type of a tag (<Operand2>) for "Not equal to".

<Operand1> must have the VARIANT data type. <Operand2> can be an elementary data type or a PLC data type.

If the data type of the VARIANT tag (<Operand1>) is an ARRAY, the data type of the ARRAY elements is compared.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0". The RLO of the instruction is logically combined with the RLO of the entire rung as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Specify <Operand1> at the operand placeholder above the instruction. Specify <Operand2> at the operand placeholder below the instruction.

Comparison of timers, date and time

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparison of structures

For the comparison of structures, you can use the instruction "CMP <>". Anonymous structures cannot generally be compared unless they are part of the same ARRAY.

For additional information on the comparison of structures, please refer to: CMP <>: Not equal (Page 2667)

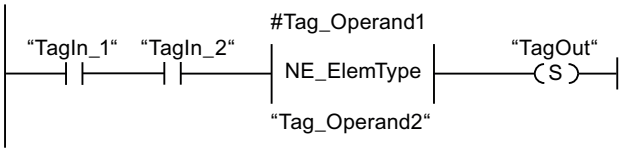
Parameters

The following table shows the parameters of the "Compare data type of an ARRAY element for UNEQUAL with the data type of a tag" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand1>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand
<Operand2>	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Second operand

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the #Tag_Operand1 operand is not equal to "Tag_Operand2".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

IS_NULL: Check for EQUALS NULL pointer (S7-1200, S7-1500)

Description

You can use the instruction "Check for EQUALS NULL pointer" to query whether the VARIANT points to a NULL pointer and therefore does not point to an object.

<Operand> must have the VARIANT data type.

Note

VARIANT tag points to an ANY pointer

If the VARIANT tag points to an ANY pointer, the instruction always returns the result RLO = "0" even if the ANY pointer is NULL.

Parameters

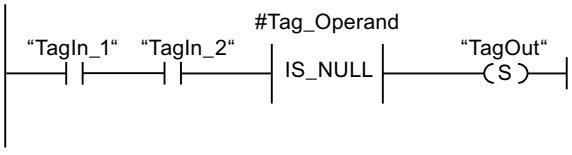
The following table shows the parameters of the "Check for EQUALS NULL pointer" instruction:

Parameter	Declaration	Data type	Memory area	Description
<Operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Operand that is compared for EQUALS NULL

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the #Tag_Operand operand does not point to an object.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

NOT_NULL: Check for UNEQUALS NULL pointer (S7-1200, S7-1500)

Description

You can use the instruction "Check for UNEQUALS NULL pointer" to query whether the VARIANT does not point to a NULL pointer and therefore points to an object.

<Operand> must have the VARIANT data type.

Note

VARIANT tag points to an ANY pointer

If the VARIANT tag points to an ANY pointer, the instruction always returns the result RLO = "1" even if the ANY pointer is NULL.

Parameters

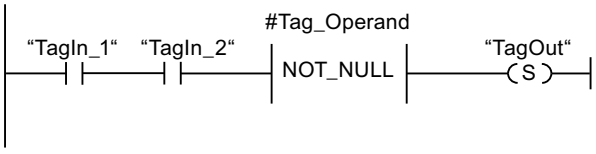
The following table shows the parameters of the "Check for UNEQUALS NULL pointer" instruction:

Parameter	Declaration	Data type	Memory area	Description
<Operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Operand that is compared for UNEQUALS NULL

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the #Tag_Operand operand points to an object.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

IS_ARRAY: Check for ARRAY (S7-1200, S7-1500)

Description

You can use the "Check for ARRAY" instruction to query whether the VARIANT points to a tag of the ARRAY data type.

<Operand> must have the VARIANT data type.

Parameters

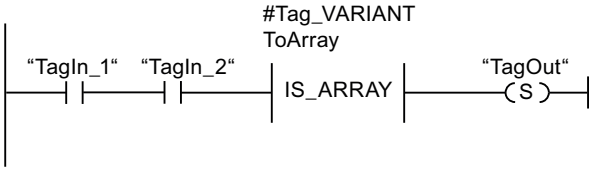
The following table shows the parameters of the "Check for ARRAY" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Operand that is queried for ARRAY

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the "#Tag_VARIANTToArray" operand is of the ARRAY data type.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Math functions (S7-1200, S7-1500)

CALCULATE: Calculate (S7-1200, S7-1500)

Description

The "Calculate" instruction is used to define and execute an expression for the calculation of mathematical operations or complex logic operations depending on the selected data type.

You can select the data type of the instruction from the "???" drop-down list of the instruction box. Depending on the data type selected, you can combine the functions of certain instructions to perform a complex calculation. The expression to be calculated is specified via a dialog you can open via the "Calculator" icon at the top of the instruction box. The expression can contain names of input parameters and the syntax of the instructions. Operand names and operand addresses cannot be specified.

In its initial state, the instruction box contains at least 2 inputs (IN1 and IN2). The number of inputs can be extended. The inserted inputs are numbered in ascending order in the box.

The values of the inputs are used to execute the specified expression. Not all of the defined inputs have to be used in the expression. The result of the instruction is transferred to the output OUT.

Note

If one of the mathematical operations fails in the expression, no result is transferred to the output OUT and the enable output ENO returns the signal state "1".

If you use inputs in the expression that are not available in the box, they are inserted automatically. This requires that there are no gaps in the numbering of the inputs to be newly defined in the expression. For example, you cannot use the IN4 input in the expression unless the IN3 input has been defined.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The result of the "Calculate" instruction is outside the range permitted for the data type specified at the OUT output.
- A floating-point number has an invalid value.
- An error occurred during execution of one of the instructions in the expression.

The following table shows the instructions that can be executed together in the expression of the instruction "Calculate", depending on the selected data type:

Data type	Instruction	Syntax	Example
Bit strings	AND: AND logic operation	AND	IN1 AND IN2 OR IN3
	OR: OR logic operation	OR	
	XOR: EXCLUSIVE OR logic operation	XOR	
	INV: Create ones complement	NOT	
	SWAP: Swap [1)]	SWAP	

Data type	Instruction	Syntax	Example
Integers	ADD: Add	+	(IN1 + IN2) * IN3; (ABS(IN2)) * (ABS(IN1))
	SUB: Subtract	-	
	MUL: Multiply	*	
	DIV: Divide	/	
	MOD: Return remainder of division	MOD	
	INV: Create ones complement	NOT	
	NEG: Create twos complement	-(in1)	
	ABS: Form absolute value	ABS( )	
Floating-point numbers	ADD: Add	+	((SIN(IN2) * SIN(IN2) + (SIN(IN3) * SIN(IN3)) / IN3)); (SQR(SIN(IN2)) + (SQR(COS(IN3)) / IN2))
	SUB: Subtract	-	
	MUL: Multiply	*	
	DIV: Divide	/	
	EXPT: Exponentiate	**	
	ABS: Form absolute value	ABS( )	
	SQR: Form square	SQR( )	
	SQRT: Form square root	SQRT( )	
	LN: Form natural logarithm	LN( )	
	EXP: Form exponential value	EXP( )	
	FRAC: Return fraction	FRAC( )	
	SIN: Form sine value	SIN( )	
	COS: Form cosine value	COS( )	
	TAN: Form tangent value	TAN( )	
	ASIN: Form arcsine value	ASIN( )	
	ACOS: Form arccosine value	ACOS( )	
	ATAN: Form arctangent value	ATAN( )	
	NEG: Create twos complement	-(in1)	
	TRUNC: Truncate numerical value	TRUNC( )	
	ROUND: Round numerical value	ROUND( )	
	CEIL: Generate next higher integer from floating-point number	CEIL( )	
	FLOOR: Generate next lower integer from floating-point number	FLOOR( )	

[1] Not possible for data type BYTE.

Parameters

The following table shows the parameters of the "Calculate" instruction:

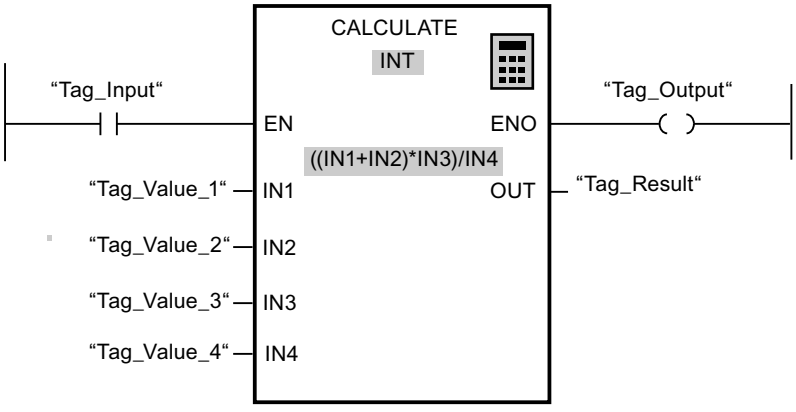
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Bit strings, integers, floating-point numbers	I, Q, M, D, L, P, or constant	First available input

Parameters	Declaration	Data type	Memory area	Description
IN2	Input	Bit strings, integers, floating-point numbers	I, Q, M, D, L, P, or constant	Second available input
INn	Input	Bit strings, integers, floating-point numbers	I, Q, M, D, L, P, or constant	Additionally inserted inputs
OUT	Output	Bit strings, integers, floating-point numbers	I, Q, M, D, L, P	Output to which the end result is to be transferred.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN1	Tag_Value_1	4
IN2	Tag_Value_2	4
IN3	Tag_Value_3	3
IN4	Tag_Value_4	2
OUT	Tag_Result	12

If the "Tag_Input" input has the signal state"1", the "Calculate" instruction is executed. The value of operand "Tag_Value_1" is added to the value of operand "Tag_Value_2". The sum is multiplied with the value of operand "Tag_Value_3". The product is divided by the value of operand "Tag_Value_4". The quotient is transferred as end result to the operand "Tag_Result" at the OUT output of the instruction. If the instruction is executed without errors, the enable output ENO and operand "Tag_Output" are set to "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Using the "Calculate" instruction (Page 4903)

Overview of the valid data types (Page 2239)

Example of calculating an equation (Page 4931)

Basic information on LAD (Page 4878)

ADD: Add (S7-1200, S7-1500)

Description

You can use the "Add" instruction to add the value at input IN1 and the value at input IN2 and query the sum at output OUT (OUT := IN1+IN2).

In its initial state, the instruction box contains at least 2 inputs (IN1 and IN2). The number of inputs can be extended. The inserted inputs are numbered in ascending order in the box. When the instruction is executed, the values of all available input parameters are added. The sum is stored at the OUT output.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The result of the instruction is outside the range permitted for the data type specified at the OUT output.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Add" instruction:

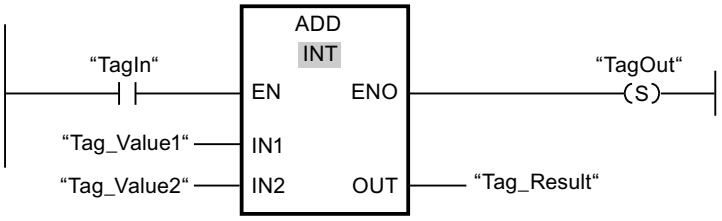
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	First number to be added
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Second number to be added
INn	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Optional input values that are added.
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	Sum

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also":

Example

The following example shows how the instruction works:



If operand "TagIn" has the signal state "1", the "Add" instruction is executed. The value of operand "Tag_Value1" is added to the value of operand "Tag_Value2". The result of the addition is stored in operand "Tag_Result". If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Selecting a data type (Page 4894)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

SUB: Subtract (S7-1200, S7-1500)

Description

You can use the "Subtract" instruction to subtract the value at input IN2 from the value at input IN1 and query the difference at output OUT (OUT := IN1-IN2).

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The result of the instruction is outside the range permitted for the data type specified at the OUT output.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Subtract" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output

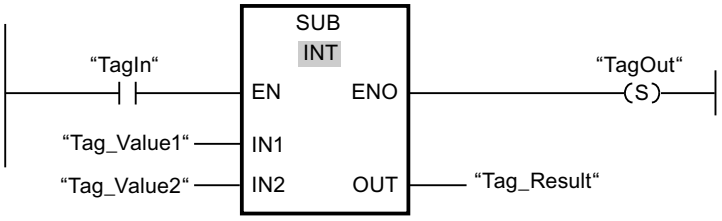
Parameters	Declaration	Data type	Memory area	Description
IN1	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Minuend
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Subtracting
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	Difference

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If operand "TagIn" has the signal state "1", the "Subtract" instruction is executed. The value of operand "Tag_Value2" is subtracted from the value of operand "Tag_Value1". The result of the subtraction is stored in operand "Tag_Result". If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

MUL: Multiply (S7-1200, S7-1500)

Description

You can use the "Multiply" instruction to multiply the value at input IN1 with the value at input IN2 and query the product at output OUT (OUT := IN1*IN2).

The number of inputs can be expanded in the instruction box. The added inputs are numbered in ascending order in the box. When the instruction is executed, the values of all available input parameters are multiplied. The product is stored at the OUT output.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN input has the signal state "0".
- The result is outside the range permitted for the data type specified at output OUT.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Multiply" instruction:

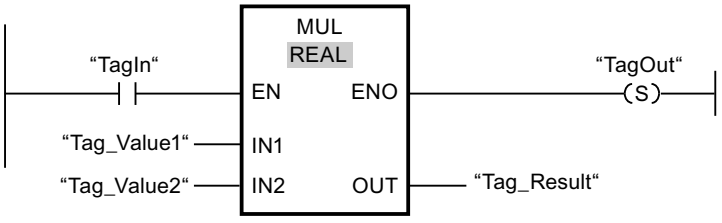
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Multiplier
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Number being multiplied
INn	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Optional input values that can be multiplied.
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	Product

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If operand "TagIn" has the signal state "1", the "Multiply" instruction is executed. The value of operand "Tag_Value1" is multiplied by the value of operand "Tag_Value2". The result of the multiplication is stored in operand "Tag_Result". If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

DIV: Divide (S7-1200, S7-1500)

Description

You can use the "Divide" instruction to divide the value at input IN1 by the value at input IN2 and query the quotient at output OUT (OUT := IN1/IN2).

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The result of the instruction is outside the range permitted for the data type specified at the OUT output.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Divide" instruction:

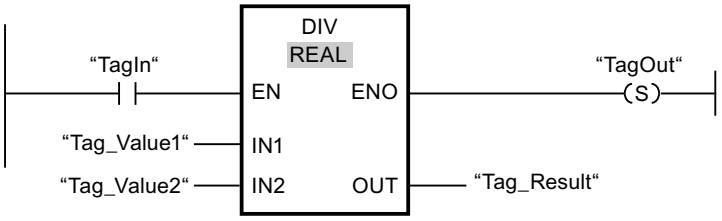
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Dividend
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Divisor
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	Quotient value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If operand "TagIn" has the signal state "1", the "Divide" instruction is executed. The value of operand "Tag_Value1" is divided by the value of operand "Tag_Value2". The division result is stored in operand "Tag_Result". If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

LAD programming examples (Page 4924)

Basic information on LAD (Page 4878)

MOD: Return remainder of division (S7-1200, S7-1500)

Description

You can use the "Return remainder of division" instruction to divide the value at input IN1 by the value at input IN2 and query the remainder of division at output OUT.

Parameters

The following table shows the parameters of the "Return remainder of division" instruction:

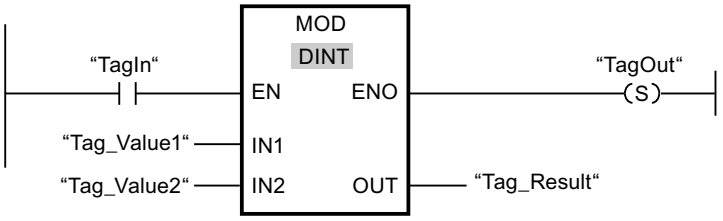
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Integers	I, Q, M, D, L, P, or constant	Dividend
IN2	Input	Integers	I, Q, M, D, L, P, or constant	Divisor
OUT	Output	Integers	I, Q, M, D, L, P	Remainder of division

You can select the data type of the instruction from the "Auto (???)" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If operand "TagIn" has the signal state "1", the "Return remainder of division" instruction is executed. The value of operand "Tag_Value1" is divided by the value of operand "Tag_Value2". The remainder of division is stored in operand "Tag_Result". If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

NEG: Create twos complement (S7-1200, S7-1500)

Description

You can use the "Create twos complement" instruction to change the sign of the value at the IN input and query the result at the OUT output. If there is a positive value at input IN, for example, the negative equivalent of this value is sent to output OUT.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The result of the instruction is outside the range permitted for the data type specified at the OUT output.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Create twos complement" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output

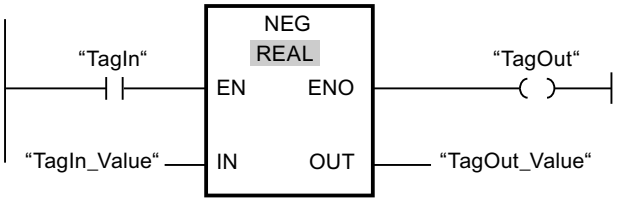
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P	Twos complement of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If operand "TagIn" has the signal state "1", the "Create twos complement" instruction is executed. The sign of the value at input "TagIn_Value" is changed and the result is provided at output "TagOut_Value". If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

INC: Increment (S7-1200, S7-1500)

Description

You can use the "Increment" instruction to change the value of the operand at the IN/OUT parameter to the next higher value and query the result. The "Increment" instruction is only started when the signal state at the EN enable input is "1". If no overflow error occurs during the execution, the ENO enable output also has the signal state "1".

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Increment" instruction:

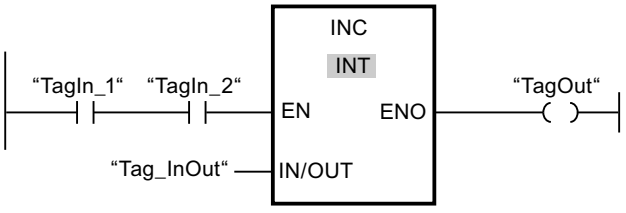
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN/OUT	InOut	Integers	I, Q, M, D, L	Value to be incremented.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If operands "TagIn_1" and "TagIn_2" have the signal state "1", the value of operand "Tag_InOut" is incremented by one and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

DEC: Decrement (S7-1200, S7-1500)

Description

You can use the "Decrement" instruction to change the value of the operand at the IN/OUT parameter to the next lower value and query the result. The "Decrement" instruction is only started when the signal state at the EN enable input is "1". If the range of values of the selected data type is not exceeded during processing, the ENO output also has the signal state "1".

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Decrement" instruction:

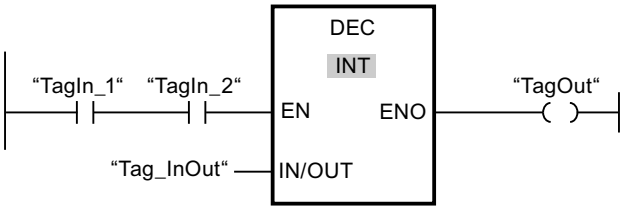
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN/OUT	InOut	Integers	I, Q, M, D, L	Value to be decremented.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If operands "TagIn_1" and "TagIn_2" have the signal state "1", the value of operand "Tag_InOut" is decremented by one and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ABS: Form absolute value (S7-1200, S7-1500)

Description

You can use the "Form absolute value" instruction to calculate the absolute value of the value specified at input IN. The result of the instruction is sent to the OUT output and can be queried there.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Form absolute value" instruction:

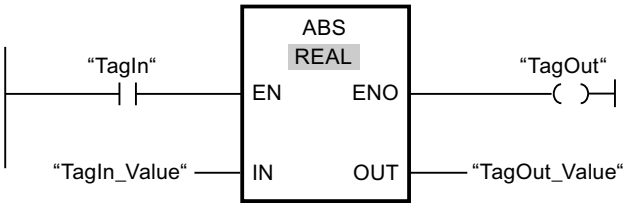
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN	Input	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P	Absolute value of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	TagIn_Value	-6.234
OUT	TagOut_Value	6.234

If operand "TagIn" has the signal state "1", the "Form absolute value" instruction is executed. The instruction calculates the absolute value of the value at input "TagIn_Value" and sends the result to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

MIN: Get minimum (S7-1200, S7-1500)

Description

The "Get minimum" instruction compares the values at the available inputs and writes the lowest value to the OUT output. The number of inputs can be expanded at the instruction box by additional inputs. The inputs are numbered in ascending order in the box.

A minimum of two and a maximum of 100 inputs must be specified for the execution of the instruction.

The ENO enable output has the signal state "0" if one of the following conditions is fulfilled:

- Enable input EN has the signal state "0".
- The implicit conversion of the data types fails during execution of the instruction.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Get minimum" instruction:

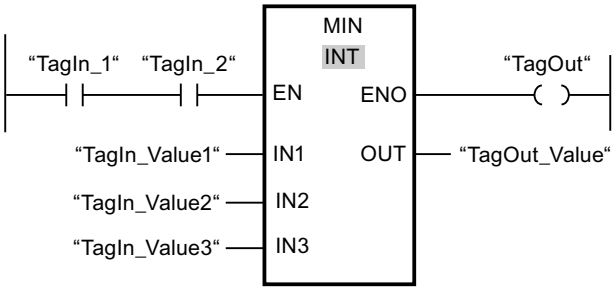
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P, or constant	First input value
IN2	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P, or constant	Second input value
INn	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P, or constant	Additionally inserted inputs whose values are to be compared
OUT	Output	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P	Result
If the IEC check is not activated, you can also use tags with the data types TIME, LTIME, TOD, LTOD, DATE and LDT by selecting a bit string or integer with the same length as the data type of the instruction. (for example, instead of TIME => DINT, UDINT or DWORD = 32 bits).				

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN1	TagIn_Value1	12222
IN2	TagIn_Value2	14444
IN3	TagIn_Value3	13333
OUT	TagOut_Value	12222

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The instruction compares the values of the specified operands and copies the lowest value ("TagIn_Value1") to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

MAX: Get maximum (S7-1200, S7-1500)

Description

The "Get maximum" instruction compares the values at the available inputs and writes the highest value to the OUT output. The number of inputs can be expanded at the instruction box by additional inputs. The inputs are numbered in ascending order in the box.

A minimum of two and a maximum of 100 inputs must be specified for the execution of the instruction.

The ENO enable output has the signal state "0" if one of the following conditions is fulfilled:

- Enable input EN has the signal state "0".
- The implicit conversion of the data types fails during execution of the instruction.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Get maximum" instruction:

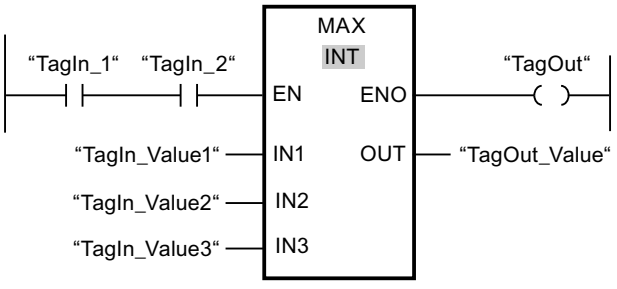
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P, or constant	First input value
IN2	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P, or constant	Second input value
INn	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P, or constant	Additionally inserted inputs whose values are to be compared
OUT	Output	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P	Result
If the IEC check is not activated, you can also use tags with the data types TIME, LTIME, TOD, LTOD, DATE and LDT by selecting a bit string or integer with the same length as the data type of the instruction. (for example, instead of TIME => DINT, UDINT or DWORD = 32 bits).				

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN1	TagIn_Value1	12222
IN2	TagIn_Value2	14444
IN3	TagIn_Value3	13333
OUT	TagOut_Value	14444

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The instruction compares the values of the specified operands and copies the highest value ("TagIn_Value2") to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

LIMIT: Set limit value (S7-1200, S7-1500)

Description

You can use the "Set limit value" instruction to limit the value at input IN to the values at the inputs MN and MX. If the value at the IN input meets the MN condition <= IN <= MX, it is copied to the OUT output. If the condition is not fulfilled and the input value IN is below the low limit MN, the output OUT is set to the value of the MN input. If the high limit MX is exceeded, output OUT is set to the value of the input MX.

If the value at the MN input is greater than at the MX input, the result is the value specified at the IN parameter and the enable output ENO is "0".

The ENO enable output has the signal state "0" if one of the following conditions is fulfilled:

- Enable input EN has the signal state "0".
- The specified tags are not of the same data type.
- An operand has an invalid value.
- The value at the MN input is greater than the value of the MX input.

Parameters

The following table shows the parameters of the "Set limit value" instruction:

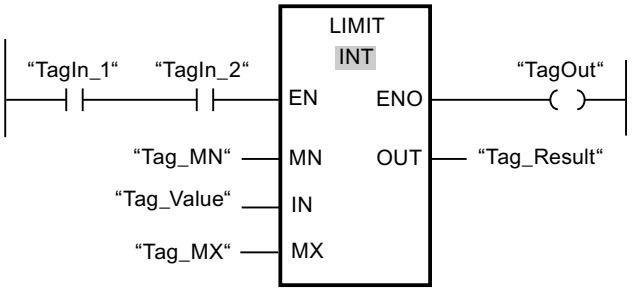
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
MN	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL, DT	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DTL, DT	I, Q, M, D, L, P, or constant	Low limit
IN	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL, DT	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DTL, DT	I, Q, M, D, L, P, or constant	Input value
MX	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL, DT	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DTL, DT	I, Q, M, D, L, P, or constant	High limit
OUT	Output	Integers, floating-point numbers, TIME, TOD, DATE, DTL, DT	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DTL, DT	I, Q, M, D, L, P	Result
The data types TOD, LTOD, DATE, and LDT can only be used if the IEC test is not enabled.					

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
MN	Tag_MN	12000
IN	Tag_Value	8000
MX	Tag_MX	16000
OUT	Tag_Result	12000

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The value of operand "Tag_Value" is compared with the values of operands "Tag_MN" and "Tag_MX". Since the value of operand "Tag_Value" is less than the low limit, the value of operand "Tag_MN" is copied to the "Tag_Result" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

SQR: Form square (S7-1200, S7-1500)

Description

The instruction "Form square" is used to square the value of a floating-point number at input IN and to write the result to output OUT.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form square":

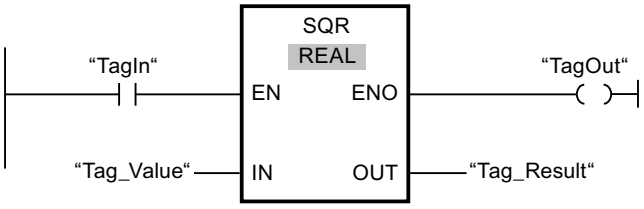
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Square of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	5.0
OUT	Tag_Result	25.0

If the operand "TagIn" has the signal state "1", the "Form square" instruction is executed. The instruction squares the value of the operand "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Inserting LAD elements (Page 4890)

Editing LAD elements (Page 4907)

Inserting operands into LAD instructions (Page 4913)

Selecting the data type of a LAD element (Page 4894)

SQRT: Form square root (S7-1200, S7-1500)

Description

The instruction "Form square root" is used form the square root from the value of the floating-point number at input IN and to write the result to output OUT. The instruction has a positive result if the input value is greater than zero. If input values are less than zero, output OUT returns an invalid floating-point number. If the value at input IN is "0", the result is also "0".

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.
- The value at input IN is negative.

Parameters

The following table shows the parameters of the instruction "Form square root":

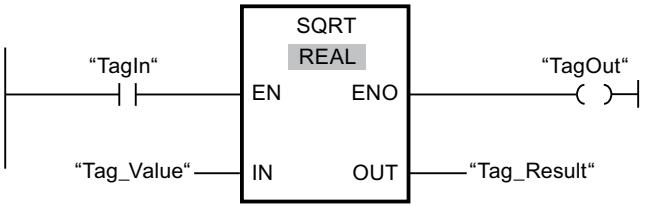
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Square root of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	25.0
OUT	Tag_Result	5.0

If the operand "TagIn" has the signal state "1", the "Form square root" instruction is executed. The instruction calculates the square root of the operand "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Inserting LAD elements (Page 4890)

Editing LAD elements (Page 4907)

Inserting operands into LAD instructions (Page 4913)

Selecting the data type of a LAD element (Page 4894)

LN: Form natural logarithm (S7-1200, S7-1500)

Description

You can use the "Form natural logarithm" instruction to calculate the natural logarithm to base e (e = 2.718282) of the value at input IN. The result is sent to output OUT and can be queried there. The instruction has a positive result if the input value is greater than zero. If input values are less than zero, output OUT returns an invalid floating-point number.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.
- The value at input IN is negative.

Parameters

The following table shows the parameters of the instruction "Form natural logarithm":

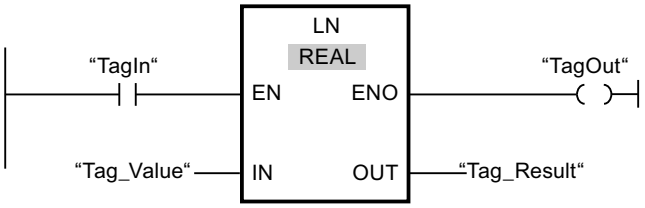
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Natural logarithm of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the operand "TagIn" has the signal state "1", the "Form natural logarithm" instruction is executed. The instruction forms the natural logarithm of the value at input "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

EXP: Form exponential value (S7-1200, S7-1500)

Description

You can use the "Form exponential value" instruction to calculate the exponent from the base e (e = 2.718282) and the value specified at input IN. The result is provided at output OUT and can be queried there (OUT = e^{IN}).

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form exponential value":

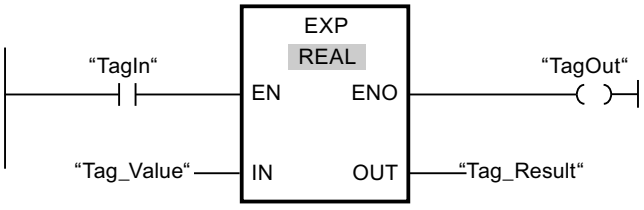
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Exponential value of input value IN

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the operand "TagIn" has the signal state "1", the "Form exponential value" instruction is executed. The instruction forms the exponent from base e and the value of the operand "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

SIN: Form sine value (S7-1200, S7-1500)

Description

Use the "Form sine value" instruction to calculate the sine of the angle. The size of the angle is specified in radians at the IN input. The result of the instruction is sent to the OUT output and can be queried there.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.

Parameters

The following table shows the parameters of the "Form sine value" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output

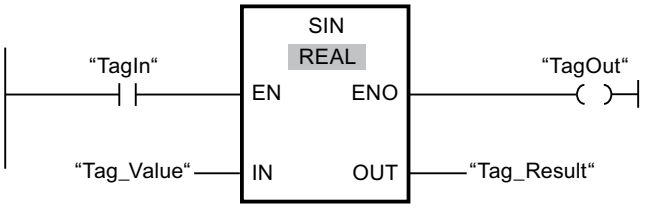
Parameters	Declaration	Data type	Memory area	Description
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Size of angle in radians
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Sine of the specified angle

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	+1.570796 (π/2)
OUT	Tag_Result	1.0

If operand "TagIn" has the signal state "1", the "Form sine value" instruction is executed. The instruction calculates the sine of the angle specified at the "Tag_Value" input and stores the result in the "Tag_Result" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

COS: Form cosine value (S7-1200, S7-1500)

Description

Use the "Form cosine value" instruction to calculate the cosine of the angle. The size of the angle is specified in radians at the IN input. The result of the instruction is sent to the OUT output and can be queried there.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.

Parameters

The following table shows the parameters of the "Form cosine value" instruction:

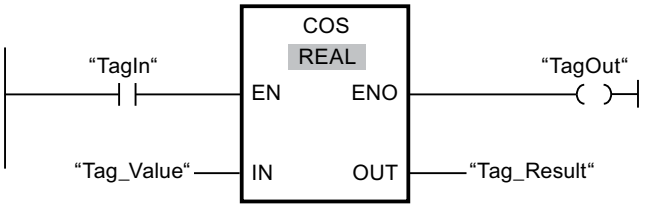
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Size of angle in radians
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Cosine of the specified angle

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	+1.570796 (π/2)
OUT	Tag_Result	0

If operand "TagIn" has the signal state "1", the "Form cosine value" instruction is executed. The instruction calculates the cosine of the angle specified at the "Tag_Value" input and stores the result in the "Tag_Result" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

TAN: Form tangent value (S7-1200, S7-1500)

Description

Use the "Form tangent value" instruction to calculate the tangent of the angle. The size of the angle is specified in radians at the IN input. The result of the instruction is sent to the OUT output and can be queried there.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.

Parameters

The following table shows the parameters of the "Form tangent value" instruction:

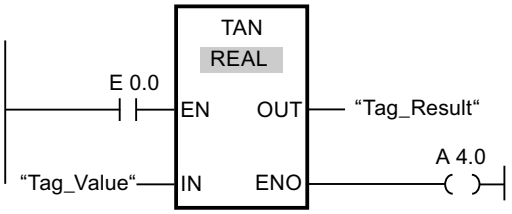
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Size of angle in radians
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Tangent of the specified angle

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	+3.141593 (π)
OUT	Tag_Result	0

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction calculates the tangent of the angle specified at the "Tag_Value" input and stores the result in the "Tag_Result" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ASIN: Form arcsine value (S7-1200, S7-1500)

Description

You can use the "Form arcsine value" instruction to calculate the size of the angle from the arcsine value specified at input IN, which corresponds to this value. Only valid floating-point numbers within the range -1 to +1 can be specified at the IN input. The calculated angle size is output in radians at the OUT output and can range in value from -π/2 to +π/2.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.
- The value at the IN input is outside the permitted value range (-1 to +1).

Parameters

The following table shows the parameters of the "Form arcsine value" instruction:

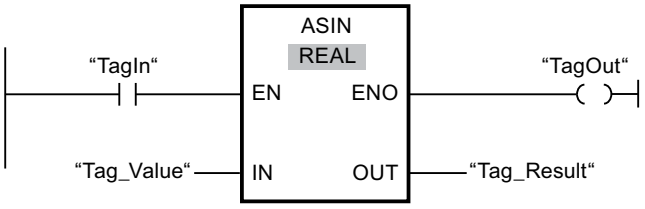
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Sine value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Size of angle in radians

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	1.0
OUT	Tag_Result	+1.570796 (π/2)

If operand "TagIn" has the signal state "1", the "Form arcsine value" instruction is executed. The instruction calculates the size of the angle corresponding to the sine value at the "Tag_Value" input. The result of the instruction is stored in the "Tag_Result" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ACOS: Form arccosine value (S7-1200, S7-1500)

Description

You can use the "Form arccosine value" instruction to calculate the size of the angle from the cosine value specified at input IN, which corresponds to this value. Only valid floating-point numbers within the range -1 to +1 can be specified at the IN input. The calculated angle size is output in radians at the OUT output and can range in value from 0 to +π.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.
- The value at the IN input is outside the permitted value range (-1 to +1).

Parameters

The following table shows the parameters of the "Form arccosine value" instruction:

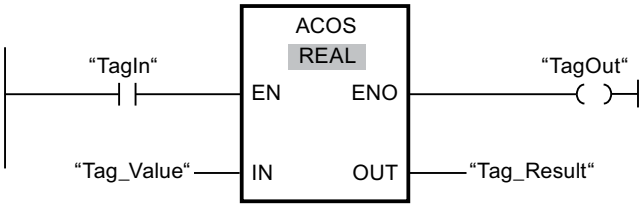
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Cosine value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Size of angle in radians

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	0
OUT	Tag_Result	+1.570796 (π/2)

If operand "TagIn" has the signal state "1", the "Form arccosine value" instruction is executed. The instruction calculates the size of the angle corresponding to the cosine value at the "Tag_Value" input. The result of the instruction is stored in the "Tag_Result" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ATAN: Form arctangent value (S7-1200, S7-1500)

Description

You can use the "Form arctangent value" instruction to calculate the size of the angle from the tangent value specified at input IN, which corresponds to this value. It is only permitted to specify valid floating-point numbers (or -NaN/+NaN) at input IN. The calculated angle size is output in radians at the OUT output and can range in value from -π/2 to +π/2.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form arctangent value":

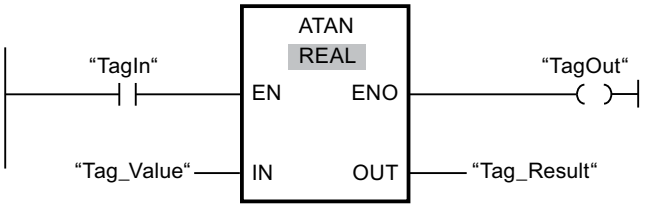
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Tangent value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Size of angle in radians

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	1.0
OUT	Tag_Result	+0.785398 (π/4)

If the operand "TagIn" has the signal state "1", the "Form arctangent value" instruction is executed. The instruction calculates the size of the angle corresponding to the tangent value at input "Tag_Value". The result of the instruction is stored in the "Tag_Result" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Invalid floating-point numbers (Page 2259)

Basic information on LAD (Page 4878)

FRAC: Return fraction (S7-1200, S7-1500)

Description

You can use the "Return fraction" instruction to determine the decimal places of the value at the IN input. The result of the query is stored at the OUT output and can be queried there. If, for example, the IN input has the value 123.4567, the OUT output returns the value 0.4567.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the IN input is not a valid floating-point number.

Parameters

The following table shows the parameters of the "Return fraction" instruction:

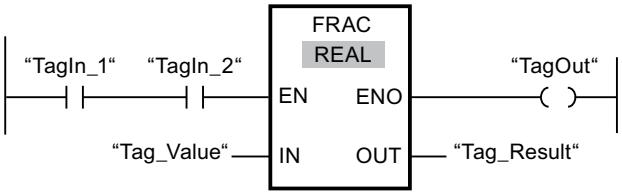
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Value, whose decimal places are to be determined.
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Decimal places of the value at the IN input

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	2.555
OUT	Tag_Result	0.555

If operands "TagIn_1" and "TagIn_2" have the signal state "1", the "Return fraction" instruction is started. The decimal places from the value of operand "Tag_Value" are copied to operand "Tag_Result". If the instruction is executed without errors, the ENO output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

EXPT: Exponentiate (S7-1200, S7-1500)

Description

You can use the "Exponentiate" instruction to raise the value at the IN1 input to a power specified with the value at the IN2 input. The result of the instruction is provided at the OUT output and can be queried there (OUT =$IN1^{IN2}$).

The IN1 input can only be assigned valid floating-point numbers. Integers can also be assigned to the IN2 input.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- Errors occur during execution of the instruction, for example, an overflow occurs.

Parameters

The following table shows the parameters of the "Exponentiate" instruction:

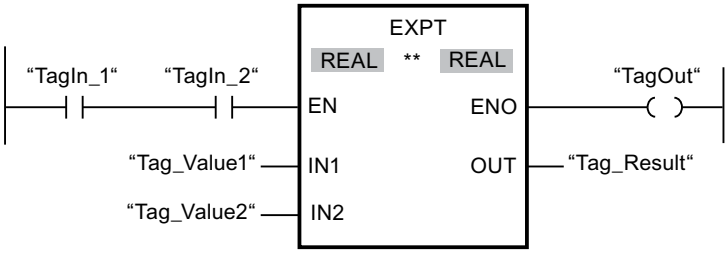
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Base value
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P, or constant	Value with which the base value is exponentiated
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	Result

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If operands "TagIn_1" and "TagIn_2" have the signal state "1", the "Exponentiate" instruction is started. The value of operand "Tag_Value1" is raised to a power specified with the value of operand "Tag_Value2". The result is stored in the "Tag_Result" output. If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Move operations (S7-1200, S7-1500)

MOVE: Move value (S7-1200, S7-1500)

Description

You use the "Move value" instruction to transfer the content of the operand at the IN input to the operand at the OUT1 output. The transfer is always made in the direction of ascending address.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The data type at the IN parameter does not correspond to the specified data type at the OUT1 parameter.

The following table lists the possible transfers for the S7-1200 CPU series:

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
BYTE	BYTE, WORD, DWORD	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE ,TOD, CHAR
WORD	WORD, DWORD	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD, CHAR
DWORD	DWORD	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, CHAR
SINT	SINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
USINT	USINT, UINT, UDINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
INT	INT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
UINT	UINT, UDINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
DINT	DINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
UDINT	UDINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
REAL	REAL	DWORD, REAL
LREAL	LREAL	LREAL
TIME	TIME	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME
DATE	DATE	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, DATE
TOD	TOD	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TOD
DTL	DTL	DTL
CHAR	CHAR	BYTE, WORD, DWORD, CHAR, Character of a string[1)]

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
WCHAR	WCHAR	BYTE, WORD, DWORD, CHAR, WCHAR, character of a character string[1)]
Character of a string[1)]	Character of a string	CHAR, WCHAR, character of a string
ARRAY[2)]	ARRAY	ARRAY
STRUCT	STRUCT	STRUCT
PLC data type (UDT)	PLC data type (UDT)	PLC data type (UDT)
IEC_TIMER	IEC_TIMER	IEC_TIMER
IEC_SCOUNT-ER	IEC_SCOUNTER	IEC_SCOUNTER
IEC_US-COUNTER	IEC_USCOUNTER	IEC_USCOUNTER
IEC_COUNTER	IEC_COUNTER	IEC_COUNTER
IEC_UCOUNT-ER	IEC_UCOUNTER	IEC_UCOUNTER
IEC_DCOUNT-ER	IEC_DCOUNTER	IEC_DCOUNTER
IEC_UD-COUNTER	IEC_UDCOUNTER	IEC_UDCOUNTER

The following table lists the possible transfers for the S7-1500 CPU series:

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
BYTE	BYTE, WORD, DWORD, LWORD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE ,TOD, LTOD, CHAR
WORD	WORD, DWORD, LWORD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, S5TIME, TIME, LTIME, LDT, DATE, TOD, LTOD, CHAR
DWORD	DWORD, LWORD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, TIME, LTIME, LDT, DATE, TOD, LTOD, CHAR
LWORD	LWORD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, LREAL, TIME, LTIME, LDT, DATE, TOD, LTOD, CHAR
SINT	SINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
USINT	USINT, UINT, UDINT, ULINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
INT	INT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
UINT	UINT, UDINT, ULINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD. LTOD
DINT	DINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
UDINT	UDINT, ULINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
LINT	LINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
ULINT	ULINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
REAL	REAL	DWORD, REAL
LREAL	LREAL	LWORD, LREAL
S5TIME	S5TIME	WORD, S5TIME
TIME	TIME	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME
LTIME	LTIME	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, LTIME
DATE	DATE	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, DATE
DT	DT	DT
LDT	LDT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, LDT
TOD	TOD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TOD
LTOD	LTOD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, LTOD
DTL	DTL	DTL
CHAR	CHAR	BYTE, WORD, DWORD, LWORD, CHAR, Character of a string[1)]
WCHAR	WCHAR	BYTE, WORD, DWORD, LWORD, CHAR, WCHAR, Character of a string[1)]
Character of a string[1)]	Character of a string	CHAR, WCHAR, character of a string
ARRAY[2)]	ARRAY	ARRAY
STRUCT	STRUCT	STRUCT
COUNTER	COUNTER, WORD, INT	WORD, DWORD, INT, UINT, DINT, UDINT
TIMER	TIMER, WORD, INT	WORD, DWORD, INT, UINT, DINT, UDINT
PLC data type (UDT)	PLC data type (UDT)	PLC data type (UDT)
IEC_TIMER	IEC_TIMER	IEC_TIMER
IEC_LTIMER	IEC_LTIMER	IEC_LTIMER

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
IEC_SCOUNT-ER	IEC_SCOUNTER	IEC_SCOUNTER
IEC_US-COUNTER	IEC_USCOUNTER	IEC_USCOUNTER
IEC_COUNTER	IEC_COUNTER	IEC_COUNTER
IEC_UCOUNT-ER	IEC_UCOUNTER	IEC_UCOUNTER
IEC_DCOUNT-ER	IEC_DCOUNTER	IEC_DCOUNTER
IEC_UD-COUNTER	IEC_UDCOUNTER	IEC_UDCOUNTER
IEC_LCOUNT-ER	IEC_LCOUNTER	IEC_LCOUNTER
IEC_UL-COUNTER	IEC_ULCOUNTER	IEC_ULCOUNTER

[1] You can also use the "Move value" instruction to transfer individual characters of a string to operands of the CHAR or WCHAR data type. The number of the character to be transferred is specified in square brackets next to the operand name. "MyString[2]", for example, transfers the second character of the "MyString" string. The transfer of the operand of the data type CHAR or WCHAR to the individual character of a string is also possible. You can also replace a specific character of a string with the character of another string.

[2] Transferring entire arrays (ARRAY) is possible only when the array components of the operands at input IN and at output OUT1 are of the same data type.

If the bit length of the data type at input IN exceeds the bit length of the data type at output OUT1, the higher-order bits of the source value are lost. If the bit length of the data type at input IN is less than the bit length of the data type at output OUT1, the higher-order bits of the destination value will be overwritten with zeros.

In its initial state, the instruction box contains 1 output (OUT1). The number of outputs can be extended. The added outputs are numbered in ascending order on the box. During the execution of the instruction, the content of the operand at the input IN is transferred to all available outputs. The instruction box cannot be extended if structured data types (DTL, STRUCT, ARRAY) or characters of a string are transferred.

You can also use the "Move block" (MOVE_BLK) and "Move block uninterruptible" (UMOVE_BLK) instructions to move operands of the ARRAY data type. Operand of the data type STRING or WSTRING can be copied with the instruction "Move character string" (S_MOVE).

Parameter

The following table shows the parameters of the "Move value" instruction:

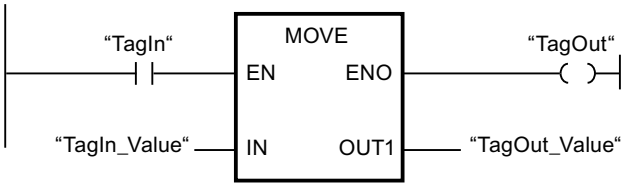
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Bit strings, integers, floating-point numbers, timers, date and time, CHAR, WCHAR, STRUCT, ARRAY, IEC data types, PLC data type (UDT)	Bit strings, integers, floating-point numbers, timers, date and time, CHAR; WCHAR, STRUCT, ARRAY, TIMER, COUNTER, IEC data type, PLC data type (UDT)	I, Q, M, D, L or constant	Source value
OUT1	Output	Bit strings, integers, floating-point numbers, timers, date and time, CHAR, WCHAR, STRUCT, ARRAY, IEC data types, PLC data type (UDT)	Bit strings, integers, floating-point numbers, timers, date and time, CHAR; WCHAR, STRUCT, ARRAY, TIMER, COUNTER, IEC data type, PLC data type (UDT)	I, Q, M, D, L	Operands to which in the source value is transferred.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	TagIn_Value	0011 1111 1010 1111
OUT1	TagOut_Value	0011 1111 1010 1111

If the "TagIn" operand returns signal state "1", the instruction is executed. The instruction copies the contents of operand "TagIn_Value" to operand "TagOut_Value" and sets output "TagOut" to signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

Note

You can find more information on the MOVE instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

MOVE_BLK: Move block (Page 2743)

UMOVE_BLK: Move block uninterruptible (Page 2761)

S_MOVE: Move character string (Page 3490)

Basic information on LAD (Page 4878)

Deserialize: Deserialize (S7-1200, S7-1500)

Description

You can use the "Deserialize" instruction to convert back the sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type> and to fill its entire contents.

The memory area in which the sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type> is located must have the ARRAY of BYTE data type and be declared with standard access in version 1.0. Optimized memory areas also permitted as of version 2.0. The capacity of the standard memory area is 64 KB. Make sure that there is enough memory space prior to the conversion. If the memory area is filled using the instruction "Serialize" and it is possible that fill bytes were inserted, their conversion is not taken into account.

It is recommended to define the low limit of the ARRAY with "0", because then the index within ARRAY corresponds to the value of the POS parameter, e.g. ARRAY[0] = POS 0. The description and example below have been formed based on this.

The instruction enables you to convert multiple sequential representations of converted data back to their original state step-by-step.

If you only want to convert back a single sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type>, you can also directly use the instruction "TRCV: Receive data via communication connection".

Size of the memory area

The alignment rules mean that simple structures in the optimized memory area do not contain filling bytes. This results in a structure in the optimized memory area being smaller than in the standard memory area. ARRAYs of structures and structures in which other structures are located contain filling bytes. There is no general rule as to the memory area in which a composed structure requires more space.

Applies to CPUs of the S7-1500 series:

For a block with the block property "Optimized block access", the BOOL has a length of 1 byte. A structure that consists largely of the data type BOOL can therefore be larger in the optimized memory area than in the standard memory area. Composed structures with a low proportion of BOOL data types are smaller in the optimized memory area than in the standard memory area.

Optimized memory area

To deserialize larger structures, you can declare the memory area for sequential representation with optimized access for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0. The content of the sequential representation remains unchanged, as for a standard memory area. Only symbolic access to the bytes in the ARRAY is possible.

Parameters

The following table shows the parameters of the "Deserialize" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRC_ARRAY	Input	VARIANT ARRAY[*] of BYTE [1)]	D, block interface of an FB (The sections Input, Output, Static and Temp are possible.)	Tag of ARRAY of BYTE data type in which the generate data flow is stored
DEST_VARIABLE	InOut	VARIANT	D, block interface of an FB	Tag in which the reconverted PLC data type (UDT), STRUCT, ARRAY of <data type> is stored
POS	InOut	DINT	I, Q, M, D, L	The operand at the POS parameter stores the index of the first byte based on the number of bytes occupied by the converted customer data. The POS parameter is calculated zero-based.
RET_VAL	Output	INT	I, Q, M, D, L	Error information

[1)] Possible for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B0	The memory areas for the SRC_ARRAY and DEST_VARIABLE parameters overlap.
8136	The tag at the SRC_ARRAY parameter is not in a block with standard access.
8150	The VARIANT data type at the SRC_ARRAY parameter contains a ZERO pointer.
8151	No valid reference at the SRC_ARRAY parameter
8153	There is not enough free memory available at the SRC_ARRAY parameter.
8154	Invalid data type at the SRC_ARRAY parameter
8250	The VARIANT data type at the DEST_VARIABLE parameter contains a ZERO pointer.
8251	No valid reference at the DEST_VARIABLE parameter
8382	The value at the POS parameter is outside the limits of the array.
9512	For a CPU of the S7-1200 module and a block with the block property "Optimized block access": The BOOL data type is used at the DEST_VARIABLE parameter. This makes the object smaller than a full byte.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

For a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0, the following error code has a different meaning:

Error code* (W#16#...)	Explanation
8136	Access to the memory area at the SRC_ARRAY parameter is invalid.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following table shows the declaration of the operands:

Operand	Data type	Declaration
DeliverPos	INT	In the "Input" section of the block interface of an FB or FC.
BufferPos	DINT	In the "Temp" section of the block interface of an FB or FC.
Error	INT	
Label	STRING[4]	

The following table shows the declarations of the PLC data types:

Name of the PLC data types	Name	Data type
Article	Number	DINT
	Declaration	STRING
	Colli	INT

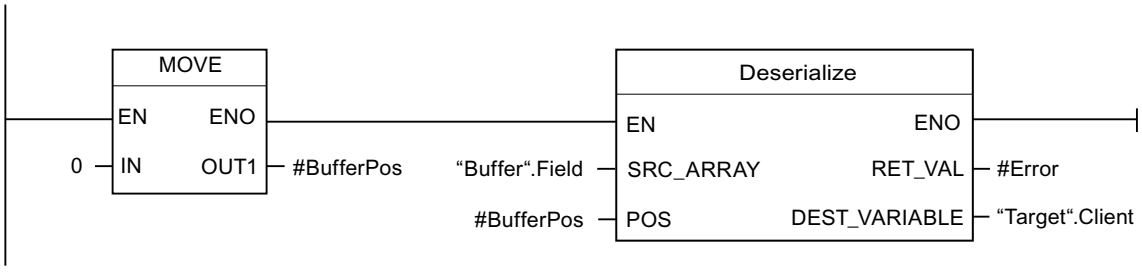
Name of the PLC data types	Name	Data type
Client	Title	INT
	First name	STRING[10]
	Surname	STRING[10]

The following table shows the declaration of the data blocks:

Name of the data blocks	Name	Data type
Target	Client	"Client" (PLC data type)
	Article	Array[0..10] of "Article" (PLC data type)
	Bill	Array[0..10] of INT
Buffer	Field	Array[0..294] of BYTE

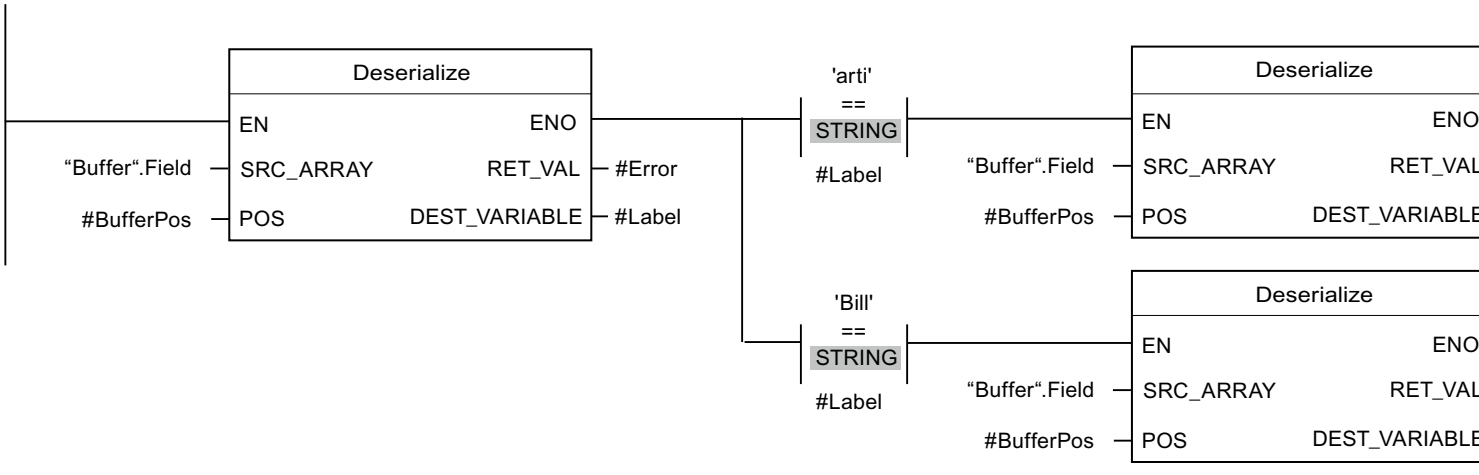
The following example shows how the instruction works:

Network 1:



The "Move value" instruction moves the value "0" to the #BufferPos operand. The "Deserialize" instruction deserializes the sequential representation of the customer data from the "Buffer" tag and writes it to the "Target" tag. The #BufferPos operand stores the index of the first byte based on the number of bytes occupied by the converted customer data.

Network 2:



The "Deserialize" instruction deserializes the sequential representation of the separator sheet, which was stored in the sequential representation after the customer data, from the "Buffer" tag and writes the characters to the #Label operand. The characters are compared using

comparison instructions "arti" and "Bill". If the comparison for "arti" = TRUE, these are article data that have been deserialized and written to the "Target" tag. If the comparison for "Bill" = TRUE, these are billing data that have been deserialized and written to the "Target" tag.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

Basics of PLC data types (UDT) (Page 2277)

Structure of an ARRAY tag (Page 2290)

Structure of a STRUCT tag (Page 2284)

Structure of a STRING tag (Page 2272)

Basic information on LAD (Page 4878)

Serialize: Serialize (S7-1200, S7-1500)

Description

You can use the "Serialize" instruction to convert several PLC data types (UDT), STRUCT or ARRAY of <data type> to a sequential representation without losing parts of their structure.

You use the instruction to temporarily save multiple structured data items from your program in a buffer, which should preferably be in a global data block, and send them to another CPU. The memory area in which the converted data are stored must have the ARRAY of BYTE data type and be declared with standard access in version 1.0. Optimized data is also permitted as of version 2.0. Fill bytes or fill bits of a data area (e.g. ARRAY, STRUCT or PLC data type (UDT)) as well as the characters of a string currently not in use are undefined in the destination ARRAY.

The capacity of the standard memory area is 64 KB. Structures that are larger than 64 KB in accordance with the rules for standard memory areas cannot be serialized if the operand at the DEST_ARRAY parameter is located in a standard memory area.

It is recommended to define the low limit of the ARRAY with "0", because then the index within ARRAY corresponds to the value of the POS parameter, e.g. ARRAY[0] = POS 0. The description and example below have been formed based on this.

The operand at the POS parameter contains information about the number of bytes used by the data.

If you want to send a single PLC data type (UDT), STRUCT or ARRAY of <data type>, you can directly call the instruction "TSEND: Send data via communication connection".

Note

Comparison of structures

To compare structures, they have to be serialized first. Use the "CMP" instruction instead.

You can find additional information in the section "Comparator".

Size of the memory area

The alignment rules mean that simple structures in the optimized memory area do not contain filling bytes. This results in a structure in the optimized memory area being smaller than in the standard memory area. ARRAYs of structures and structures in which other structures are located contain filling bytes. There is no general rule as to the memory area in which a composed structure requires more space.

Applies to CPUs of the S7-1500 series:

For a block with the block property "Optimized block access", the BOOL has a length of 1 byte. A structure that consists largely of the data type BOOL can therefore be larger in the optimized memory area than in the standard memory area. Composed structures with a low proportion of BOOL data types are smaller in the optimized memory area than in the standard memory area.

Optimized memory area

To serialize larger structures, you can declare the memory area with optimized access for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0. The sequential representation remains unchanged, as for a standard memory area.

Parameters

The following table shows the parameters of the instruction "Serialize":

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRC_VARIABLE	Input	VARIANT	D, block interface of an FB	PLC data type (UDT), STRUCT or ARRAY of <data type> that is converted to sequential representation.
DEST_ARRAY	InOut	VARIANT	D, block interface of an FB (the sections Input, Output, Static and Temp are possible.)	Data block in which the generated data stream is stored.

Parameters	Declaration	Data type	Memory area	Description
POS	InOut	DINT	I, Q, M, D, L	The operand at the POS parameter stores the index of the first byte based on the total number of bytes that the converted customer data has occupied. The POS parameter is calculated zero-based.
RET_VAL	Output	INT	I, Q, M, D, L	Error information

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B0	The memory areas for the SRC_VARIABLE and DEST_ARRAY parameters overlap.
8150	The VARIANT data type at the SRC_VARIABLE parameter contains a ZERO pointer.
8151	No valid reference at the SRC_VARIABLE parameter
8236	The tag at the SRC_ARRAY parameter is not in a block with standard access.
8250	The VARIANT data type at the DEST_ARRAY parameter contains a ZERO pointer.
8251	No valid reference at the DEST_ARRAY parameter
8253	There is not enough free memory available at the DEST_ARRAY parameter.
8254	Invalid data type at the DEST_ARRAY parameter
8382	The value at the POS parameter is outside the limits of the array.
9512	For a CPU of the S7-1200 module and a block with the block property "Optimized block access": The BOOL data type is used at the DEST_VARIABLE parameter. This makes the object smaller than a full byte.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

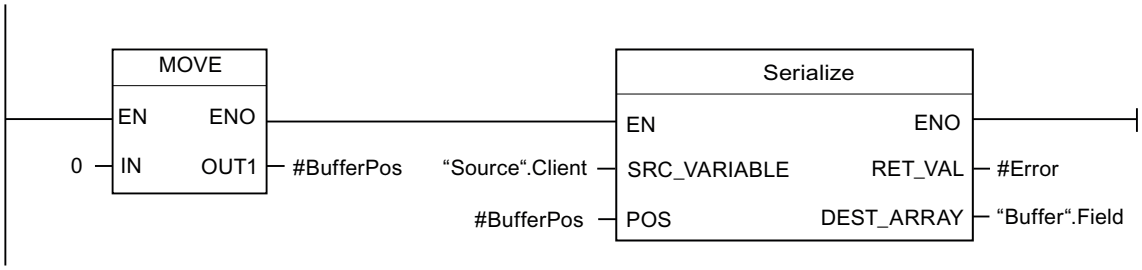
For a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0, the following error code has a different meaning:

Error code* (W#16#...)	Explanation
8236	Access to the memory area at the DEST_ARRAY parameter is invalid.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

Example

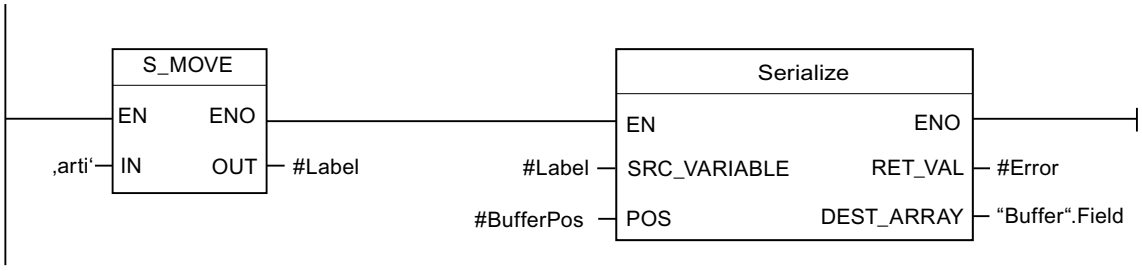
The following example shows how the instruction works:

Network 1:



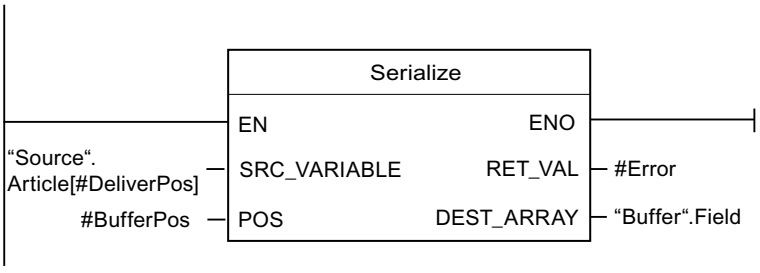
The "Move value" instruction moves the value "0" to the #BufferPos operand. The "Deserialize" instruction deserializes the customer data from the "Source" tag and writes the sequential representation to the "Buffer" tag. The index of the next unwritten byte of the "Buffer".Field operand is saved in the #BufferPos operand.

Network 2:



A kind of separator sheet is inserted now to make it easier to deserialize the sequential representation later. The "Move character string" instruction moves the "arti" characters to the #Label operand. The "Serialize" instruction writes these characters after the customer data to the "Buffer" tag. The value of the #BufferPos operand is incremented accordingly.

Network 3:



The "Serialize" instruction serializes the data of a specific article, which is calculated in runtime, from the "Source" tag and writes it in sequential representation to the "Buffer" tag after the "arti" characters.

The following table shows the declaration of the operands:

Operand	Data type	Declaration
DeliverPos	INT	In the "Input" section of the block interface
BufferPos	DINT	In the "Temp" section of the block interface

Operand	Data type	Declaration
Error	INT	In the "Temp" section of the block interface
Label	STRING[4]	In the "Temp" section of the block interface

The following table shows the declarations of the PLC data types:

Name of the PLC data types	Name	Data type
Article	Number	DINT
	Declaration	STRING
	Colli	INT
Client	Title	INT
	First name	STRING[10]
	Surname	STRING[10]

The following table shows the declaration of the data blocks:

Name of the data blocks	Name	Data type
Source	Client	"Client" (PLC data type)
	Article	Array[0..10] of "Article" (PLC data type)
Buffer	Field	Array[0..294] of BYTE

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

Basics of PLC data types (UDT) (Page 2277)

Structure of an ARRAY tag (Page 2290)

Structure of a STRUCT tag (Page 2284)

Structure of a STRING tag (Page 2272)

Basic information on LAD (Page 4878)

MOVE_BLK: Move block (S7-1200, S7-1500)

Description

You use the "Move block" instruction to move the content of a memory area (source range) to another memory area (target range). The number of elements to be moved to the target range is specified at input COUNT. The width of the elements to be moved is defined by the width of the element at input IN.

The instruction can only be executed if the source range and the target range have the same data type.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- More data is moved than is made available at input IN or output OUT.

When a ARRAY of BOOL is copied, the enable output ENO for an overflow is set to "1" until the byte limit of the ARRAY structure is exceeded. If the byte limit of the ARRAY structure is exceeded by the value at the COUNT input, the ENO enable output is reset to "0".

Note

Use of ARRAYs

The instruction copies the contents from the defined element n elements (n = depending on the value at the parameter COUNT) from the source range to the target range, starting at the specified index.

Parameters

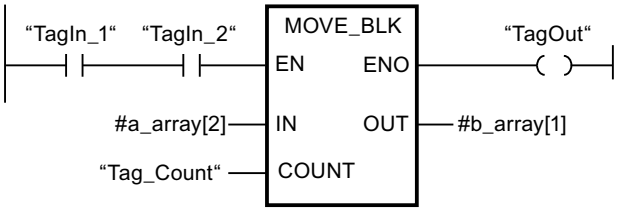
The following table shows the parameters of the "Move block" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN [1]	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	The first element of the source area that is being copied
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P or constant	Number of elements to be moved from the source range to the target range.
OUT [1]	Output	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	The first element of the target range to which the contents of the source range are being copied
[1] The specified data types can only be used as elements of an ARRAY structure.					

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN	a_array[2]	The data type of the "a_array" operand is Array [0..5] of INT. It consists of six elements of data type INT.
COUNT	Tag_Count	3
OUT	b_array[1]	The data type of the "b_array" operand is Array [0..6] of INT. It consists of seven elements of data type INT.

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. Starting from the third element, the instruction selects three INT elements from the #a_array tag and copies their contents to the #b_array output tag, beginning with the second element. If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

Note

You can find more information on the MOVE_BLK instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

SCATTER: Parse the bit sequence into individual bits (S7-1200, S7-1500)

Description

The instruction "Parse the bit sequence into individual bits" splits a tag of the BYTE, WORD, DWORD or LWORD data type into individual bits and saves them in an ARRAY of BOOL.

Note

Length of the ARRAY

The ARRAY of BOOL must have exactly the number of elements that is specified by the bit sequence.

This means for the data type BYTE, for example, the ARRAY must have exactly 8 elements (WORD = 16, DWORD = 32 and LWORD = 64).

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

This way you can, for example, parse a status word, and read and change the status of the individual bits using the index. Using GATHER, you can merge the bits once again into a bit sequence.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- If the ARRAY does not provide enough BOOL elements.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	BYTE, WORD, DWORD, LWORD	I, Q, M, D, L	Bit sequence that is parsed
OUT	Output	ARRAY[*] of BOOL *: 8, 16, 32 or 64 elements	I, Q, M, D, L	ARRAY in which the individual bits are saved

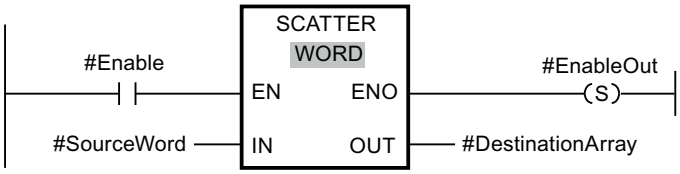
You can find additional information on valid data types under "See also".

Example

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceWord		WORD
EnableOut	Output	BOOL
DestinationArray		ARRAY[0..15] of BOOL

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceWord	WORD (16 bits)
OUT	DestinationArray	The operand "DestinationArray" is of the data type ARRAY[0..15] of BOOL. It consists of 16 elements and is therefore just as large as the WORD that is to be parsed.

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. The operand #SourceWord of the data type WORD is parsed into its individual bits (16) and assigned to the individual elements of the #DestinationArray operand. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

SCATTER_BLK: Parse elements of an ARRAY of bit sequence into individual bits (S7-1200, S7-1500)

Description

The instruction "Parse elements of an ARRAY of bit sequence into individual bits" parses one or more elements of an ARRAY of BYTE, WORD, DWORD or LWORD into individual bits and saves them in an ARRAY of BOOL. At the COUNT_IN parameter you specify how many elements of the source ARRAY are going to be parsed. The source ARRAY at the IN parameter may have more elements than specified at the COUNT_IN parameter. The ARRAY of BOOL must have sufficient elements to save the bits of the parsed bit strings. However, the destination ARRAY may also be larger.

Note

If the low ARRAY limit of the target ARRAY is not "0", note the following:

For performance reasons the index must always start at a BYTE, WORD, DWORD or LWORD limit. This means the index must be calculated starting at the low limit of the ARRAY. The following formula is used as basis for this calculation:

Valid indices = Low ARRAY limit + n(number of bit strings) × number of bits of the desired bit string

For an ARRAY[-2..45] of BOOL and the bit string WORD the calculation looks as follows:

- Valid index (-2) = -2 + 0 × 16
- Valid index (14) = -2 + 1 × 16
- Valid index (30) = -2 + 2 × 16

You can find an example described below.

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

This way you can, for example, parse status words, and read and change the status of the individual bits using the index. Using GATHER, you can combine the bits once again into a bit string.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- The source ARRAY has fewer elements than specified at the COUNT_IN parameter.
- The index of the destination ARRAY does not start at a BYTE, WORD, DWORD or LWORD limit. In this case, no result is written to the ARRAY of BOOL.
- The ARRAY[*] of BOOL does not provide the required number of elements. In this case as many bit strings as possible are parsed and written to the ARRAY of BOOL. The remaining bit strings are no longer taken into account.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Element of an ARRAY[*] of <bit string>	I, Q, M, D, L	ARRAY of <bit string> that is parsed
COUNT_IN	Input	UDINT	I, Q, M, D, L	Counter for the number of elements of the source ARRAY that are to be parsed
OUT	Output	Element of an ARRAY[*] of BOOL	I, Q, M, D, L	ARRAY in which the individual bits are saved

You can select the required bit string from the "???" drop-down list of the instruction box.

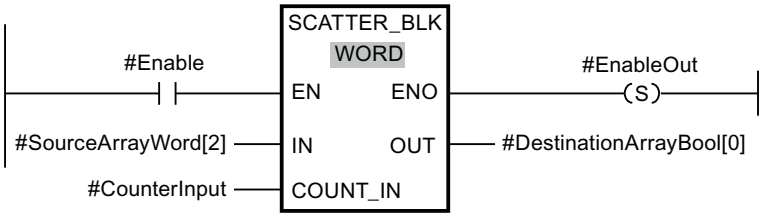
You can find additional information on valid data types under "See also".

Example of a destination ARRAY wit the low limit "0"

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArrayWord		ARRAY[0..5] of WORD
CounterInput		UDINT
EnableOut	Output	BOOL
DestinationArrayBool		ARRAY[0..95] of BOOL

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayWord[2]	ARRAY[0..5] of WORD (96 bits can be parsed.)
COUNT_IN	CounterInput = 3	UDINT3 (3 WORDs or 48 bits are to be parsed. This means at least 48 bits must be available in the destination ARRAY.)
OUT	DestinationArrayBool[0]	The operand "DestinationArrayBool" is of the data type ARRAY[0..95] of BOOL. This means it provides 96 BOOL elements.

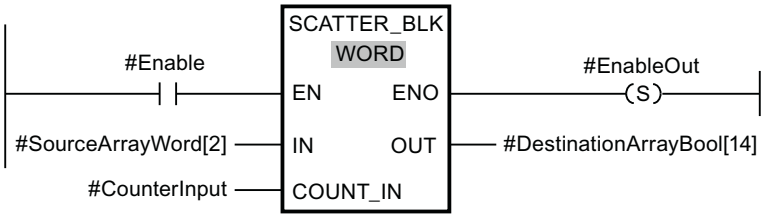
If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. The 3rd, 4th and 5th WORD of the operand #SourceArrayWord is parsed into its individual bits (48) and as of the first element assigned to the individual elements of the #DestinationArrayBool operand. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

Example of a destination ARRAY wit the low limit "-2"

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArrayWord		ARRAY[0..5] of WORD
CounterInput		UDINT
EnableOut	Output	BOOL
DestinationArrayBool		ARRAY[-2..93] of BOOL

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayWord[2]	ARRAY[0..5] of WORD (96 bits can be parsed.)
COUNT_IN	CounterInput = 3	UDINT3 (3 WORDs or 48 bits are to be parsed. This means at least 48 bits must be available in the destination ARRAY.)
OUT	DestinationArrayBool[14]	The operand "DestinationArray-Bool" is of the data type AR-RAY[-2..93] of BOOL. This means it provides 96 BOOL ele-ments.

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. The 3rd, 4th and 5th WORD of the operand #SourceArrayWord is parsed into its individual bits (48) and as of the 16th element assigned to the individual elements of the #DestinationArrayBool operand. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO. The remaining 32 bits are not written.

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

GATHER: Merge individual bits into a bit sequence (S7-1200, S7-1500)

Description

The instruction "Merge individual bits into a bit sequence" combines the bits from an ARRAY of BOOL into a bit sequence. The bit sequence is saved in a tag of the data type BYTE, WORD, DWORD or LWORD.

Note

Length of the ARRAY

The ARRAY of BOOL must have exactly the number of elements that is specified by the bit sequence.

This means for the data type BYTE, for example, the ARRAY must have exactly 8 elements (WORD = 16, DWORD = 32 and LWORD = 64).

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- If the ARRAY has fewer or more BOOL elements than specified by the bit sequence, they are not transferred and the ENO returns signal state "0". The ENO also returns "0" if fewer than the necessary number of bits is available.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	ARRAY[*] of BOOL *: 8, 16, 32 or 64 elements	I, Q, M, D, L	ARRAY whose bits are combined into a bit sequence
OUT	Output	BYTE, WORD, DWORD, LWORD	I, Q, M, D, L	Combined bit sequence, saved in a tag

You can select the required bit sequence from the "???" drop-down list of the instruction box.

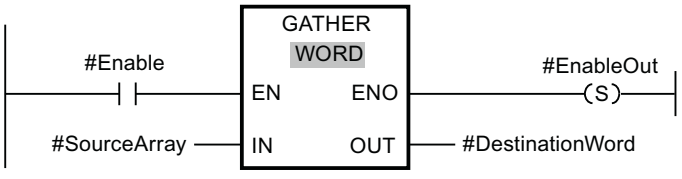
You can find additional information on valid data types under "See also".

Example

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArray		WORD
EnableOut	Output	BOOL
DestinationWord		ARRAY[0..15] of BOOL

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArray	The operand "SourceArray" is of the data type ARRAY[0..15] of BOOL. It consists of 16 elements and is therefore just as large as the WORD in which the bits are to be combined.
OUT	DestinationWord	WORD (16 bits)

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. The bits of the operand #SourceArray are combined into a WORD. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

GATHER_BLK: Merge individual bits into multiple elements of an ARRAY of bit sequence (S7-1200, S7-1500)

Description

The instruction "Merge individual bits into multiple elements of an ARRAY of bit sequence" combines the bits from an ARRAY of BOOL into one or multiple elements of an ARRAY of <bit string>. At the COUNT_OUT parameter you specify how many elements of the destination ARRAY are going to be written. With this step you also implicitly specify how many elements of the ARRAY of BOOL are required. The destination ARRAY at the OUT parameter may have more elements than specified at the COUNT_OUT parameter. The ARRAY of <bit string> must have sufficient elements to save the bits that are going to be combined. However, the destination ARRAY may also be larger.

Note

If the low ARRAY limit of the target ARRAY is not "0", note the following:

For performance reasons the index must always start at a BYTE, WORD, DWORD or LWORD limit. This means the index must be calculated starting at the low limit of the ARRAY. The following formula is used as basis for this calculation:

Valid indices = Low ARRAY limit + n(number of bit strings) × number of bits of the desired bit string

For an ARRAY[-2..45] of BOOL and the bit string WORD the calculation looks as follows:

- Valid index (-2) = -2 + 0 × 16
- Valid index (14) = -2 + 1 × 16
- Valid index (30) = -2 + 2 × 16

You can find an example described below.

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- The index of the source ARRAY does not start at a BYTE, WORD, DWORD or LWORD limit. In this case, no result is written to the ARRAY of <bit string>.
- The ARRAY[*] of <bit string> does not provide the required number of elements. In this case as many bit strings as possible are combined and written to the ARRAY of <bit string>. The remaining bits are no longer taken into account.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Element of an ARRAY[*] of BOOL	I, Q, M, D, L	ARRAY of BOOL whose bits are combined (source ARRAY)
COUNT_OUT	Input	UDINT	I, Q, M, D, L	Counter of the number of elements of the destination ARRAY that are going to be written
OUT	Output	Element of an ARRAY[*] of <bit string>	I, Q, M, D, L	ARRAY of <bit string> to which the bits are saved (destination ARRAY)

You can select the required bit string from the "???" drop-down list of the instruction box.

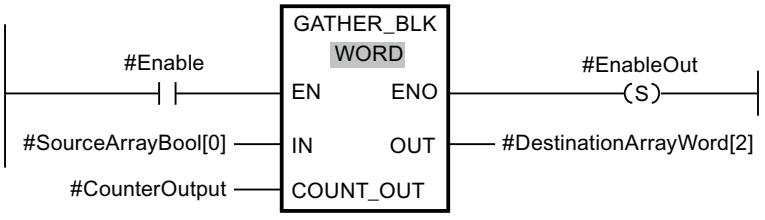
You can find additional information on valid data types under "See also".

Example of a source ARRAY wit the low limit "0"

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArrayBool		ARRAY[0..95] of BOOL
CounterOutput		UDINT
EnableOut	Output	BOOL
DestinationArrayWord		ARRAY[0..5] of WORD

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayBool[0]	The operand "SourceArrayBool" is of the data type ARRAY[0..95] of BOOL. This means it provides 96 BOOL elements that can combined into words once again.
COUNT_OUT	CounterOutput = 3	UDINT3 (3 words are to be written. This means 48 bits must be available in the source ARRAY.)
OUT	DestinationArrayWord[2]	The operand "DestinationArrayWord" is of the data type ARRAY[0..5] of WORD. This means 6 WORD elements are available.

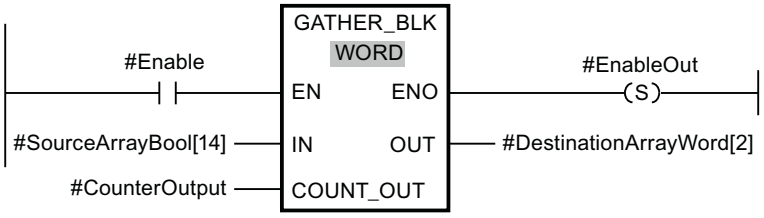
If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. Starting with the 1st element of the operand #SourceArrayBool, 48 bits are combined in the operand #DestinationArrayWord. The starting point in the destination ARRAY is the 3rd element. This means the first 16 bits are written to the 3rd word, the second 16 bits to the 4th word, and the third 16 bits to the 5th word of the destination ARRAY. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

Example of a source ARRAY wit the low limit "-2"

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArrayBool		ARRAY[-2..93] of BOOL
CounterOutput		UDINT
EnableOut	Output	BOOL
DestinationArrayWord		ARRAY[0..5] of WORD

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayBool[14]	The operand "SourceArrayBool" is of the data type ARRAY[-2..93] of BOOL. Because the starting point is the 14th element, only 80 BOOL elements that can be combined into words again are available.
COUNT_OUT	CounterOutput = 3	UDINT3 (3 words are to be written. This means 48 bits must be available in the source ARRAY.)
OUT	DestinationArrayWord[2]	The operand "DestinationArrayWord" is of the data type ARRAY[0..5] of WORD. This means 6 WORD elements are available.

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. Starting with the 14th element of the operand #SourceArrayBool, 48 bits are combined in the operand #DestinationArrayWord. The starting point in the destination ARRAY is the 3rd element. This means the first 16 bits of the source ARRAY are ignored. The second 16 bits are written to the 3rd word, the third 16 bits to the 4th word, and the fourth 16 bits to the 5th word of the destination ARRAY. The remaining 64 bits of the source ARRAY are not taken into account either. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

MOVE_BLK_VARIANT: Move block (S7-1200, S7-1500)

Description

You use the "Move block" instruction to move the content of a memory area (source range) to another memory area (target range). You can copy a complete ARRAY or elements of an ARRAY to another ARRAY of the same data type. The size (number of elements) of source and destination ARRAY may be different. You can copy multiple or single elements within an ARRAY.

The number of elements that are to be copied must not exceed the selected source range or target range.

If you use the instruction at the time the block is being created, the ARRAY does not yet have to be known, as the source and the target are transferred per VARIANT.

Counting at the parameters SRC_INDEX and DEST_INDEX always begins with the low limit "0", regardless of the later declaration of the ARRAY.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- More data is copied than is made available.

Parameters

The following table shows the parameters of the "Move block" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRC	Input [2)]	VARIANT (which points to an ARRAY or an individual ARRAY element), ARRAY of <Data_type>	L (A declaration in the "Temp" section of the block interface is not possible.)	Source block from which to copy
COUNT	Input	UDINT	I, Q, M, D, L or constant	Number of elements which are copied Assign the value "1" to the parameter COUNT, if no ARRAY is specified at parameter SRC or at parameter DEST.
SRC_INDEX	Input	DINT	I, Q, M, D, L or constant	Defines the first element to be copied: • The SRC_INDEX parameter is calculated zero-based. If an ARRAY is specified at parameter SRC, the integer at parameter SRC_INDEX specifies the first element within the source area from which it is to be copied. Independent of the declared ARRAY limits. • If no ARRAY is specified at parameter SRC or only one single element of an ARRAY is specified, then assign the value "0" at parameter SRC_INDEX.

Parameter	Declaration	Data type	Memory area	Description
DEST_INDEX	Input	DINT	I, Q, M, D, L or constant	Defines the start of the destination memory area: • The DEST_INDEX parameter is calculated zero-based. If an ARRAY is specified at parameter DEST, the integer at parameter DEST_INDEX specifies the first element within the target range that is to be copied into. Independent of the declared ARRAY limits. • If no ARRAY is specified at parameter DEST, then assign the value "0" at parameter DEST_INDEX.
DEST	Output [1]	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Destination area into which the contents of the source block are copied.
RET_VAL	Output	INT	I, Q, M, D, L	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.
1) The DEST parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface. 2) With the SRC parameter, the data types BOOL and ARRAY of BOOL are not permitted.				

You can find additional information on valid data types under "See also".

Parameter RET_VAL

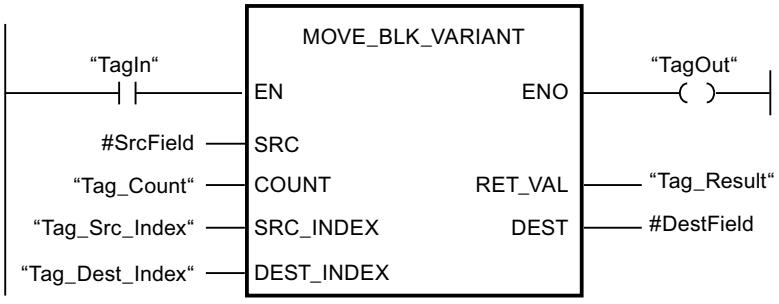
The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The data types do not correspond. Instead of an ARRAY of <Struct>, use an ARRAY of <PLC data type (UDT)>.
8151	Access to the SRC parameter is not possible.
8152	The operand at the SRC parameter is not typed.
8153	Code generation error at the SRC parameter
8154	The operand at the SRC parameter has the data type BOOL.
8281	The COUNT parameter has an invalid value
8382	The value at the SRC_INDEX parameter is outside the limits of the VARIANT.

Error code* (W#16#...)	Explanation
8383	The value at the SRC_INDEX is outside the high limit of the ARRAY.
8482	The value at the DEST_INDEX parameter is outside the limits of the VARIANT.
8483	The value at the DEST_INDEX is outside the high limit of the ARRAY.
8534	The DEST parameter is write protected
8551	Access to the DEST parameter is not possible.
8552	The operand at the DEST parameter is not typed.
8553	Code generation error at the DEST parameter
8554	The operand at the DEST parameter has the data type BOOL.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Declaration in the block interface	Operand	Value
SRC	Input	#SrcField	The local operand #SrcField uses a PLC data type that was still unknown at the time when the block was programmed. (ARRAY[0..10] of "MOVE_UDT")
COUNT	Input	Tag_Count	2
SRC_INDEX	Input	Tag_Src_Index	3
DEST_INDEX	Input	Tag_Dest_Index	3
DEST	InOut	#DestField	The local operand #DestField uses a PLC data type that was still unknown at the time when the block was programmed. (ARRAY[10..20] of "MOVE_UDT")

If the "TagIn" operand returns signal state "1", the instruction is executed. Two elements are moved from the source range, starting from the fourth element of the ARRAY [0..10] of MOVE_UDT, to the target range. The copies are pasted in the ARRAY [10..20] of MOVE_UDT starting from the fourth element. If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

Note

You can find more information on the MOVE_BLK_VARIANT instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

VariantGet: Read out VARIANT tag value (Page 2783)

Example of moving data (Page 2316)

Basic information on VARIANT (Page 2308)

Basic information on LAD (Page 4878)

UMOVE_BLK: Move block uninterruptible (S7-1200, S7-1500)

Description

You use the "Move block uninterruptible" instruction to move the content of a memory area (source range) to another memory area (target range). The instruction cannot be interrupted. The number of elements to be moved to the target range is specified with the COUNT parameter. The width of the elements to be moved is defined by the width of the element at input IN.

The instruction can only be executed if the source range and the target range have the same data type.

Note

The move operation cannot be interrupted by other operating system activities. This is why the interrupt reaction times of the CPU increase during the execution of the "Move block uninterruptible" instruction.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- More data is moved than is made available at input IN or output OUT.

When a ARRAY of BOOL is copied, the enable output ENO for an overflow is set to "1" until the byte limit of the ARRAY structure is exceeded. If the byte limit of the ARRAY structure is exceeded by the value at the COUNT input, the ENO enable output is reset to "0".

Note

Use of ARRAYs

The instruction copies the contents from the defined element n elements (n = depending on the value at the parameter COUNT) from the source range to the target range, starting at the specified index.

You use the "Move block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Parameters

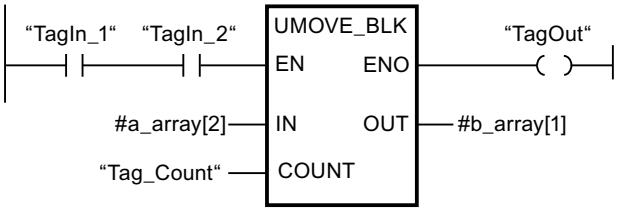
The following table shows the parameters of the "Move block uninterruptible" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN [1)]	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	The first element of the source area that is being copied
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P or constant	Number of elements to be moved from the source range to the target range.
OUT [1)]	Output	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	The first element of the target range to which the contents of the source range are being copied
[1)] The specified data types can only be used as elements of an ARRAY structure.					

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN	a_array[2]	The data type of the "a_array" operand is Array [0..5] of INT. It consists of six elements of data type INT.
COUNT	Tag_Count	3
OUT	b_array[1]	The data type of the "b_array" operand is Array [0..6] of INT. It consists of seven elements of data type INT.

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. Starting from the third element, the instruction selects three INT elements from the #a_array tag and copies their contents to the #b_array output tag, beginning with the second element. The move operation cannot be interrupted by other operating system activities. If the instruction is executed without errors, the ENO output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

Note

You can find more information on the UMOVE_BLK instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

FILL_BLK: Fill block (S7-1200, S7-1500)

Description

You can use the "Fill block" instruction to fill a memory area (target range) with the value of the IN input. The target range is filled beginning with the address specified at the OUT output. The number of repeated copy operations is specified with the COUNT parameter. When the instruction is executed, the value at the input IN is moved to the target range as often as specified by the value of the COUNT parameter.

The instruction can only be executed if the source range and the target range have the same data type.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The maximum number of elements changed is the number of elements in the ARRAY or structure. If you copy more data than there are elements at the OUT output, you will get an unintended result.

When a ARRAY of BOOL is copied, the enable output ENO for an overflow is set to "1" until the byte limit of the ARRAY structure is exceeded. If the byte limit of the ARRAY structure is exceeded by the value at the COUNT input, the ENO enable output is reset to "0".

Note

Use of ARRAYs

The instruction reads the content from the selected element from the source range and copies this content n-times (n = depending on the value at the parameter COUNT) to the destination, starting at the specified index.

Filling structures

As well as the elements of an ARRAY, you can also fill multiple elements of a structure (STRUCT, PLC data type) with the same value. The structure whose elements you want to fill must only contain individual elements of the same elementary data type. The structure itself can, however, be embedded in another structure.

Parameters

The following table shows the parameters of the "Fill block" instruction:

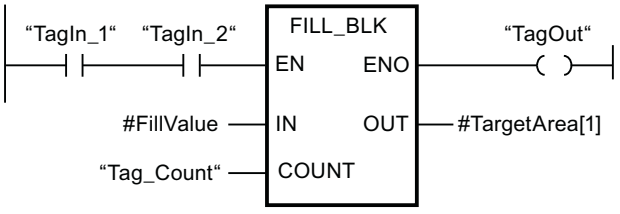
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	I, Q, M, D, L, P, or constant	Element used to fill the target range
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P, or constant	Number of repeated move operations
OUT	Output	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	Address in target range from which filling starts

For additional information on valid data types, refer to "See also".

Example with an ARRAY

The following example shows how the instruction works when you want to fill an ARRAY:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	FillValue	The data type of the operand is INT.
COUNT	Tag_Count	3
OUT	TargetArea	The data type of the operand TargetArea is ARRAY[1..5] of INT. It consists of five elements of data type INT.

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The instruction copies the value of the operand #FillValue to the #TargetArea output tag three times, starting with the first element. If the instruction is executed without errors, the TagOut operand at the "ENO" enable output is set to signal state "1".

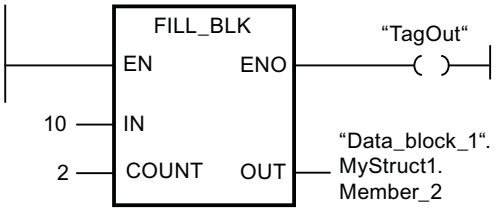
Examples with a structure

The following examples show how the instruction works when you want to fill a structure:

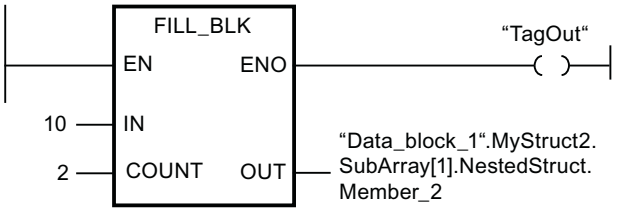
Create a global data block with the following elements:

Data_block_1					Data type
MyStruct1					STRUCT
	Member_1				INT
	Member_2				INT
	Member_3				INT
	Member_4				INT
MyStruct2					STRUCT
	SubArray				ARRAY[1..2] of STRUCT
		SubArray[1]			STRUCT
			NestedStruct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT
		SubArray[2]			STRUCT
			Nested-Struct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT

Generate the following program code to address the MyStruct1 tag:



Generate the following program code to address the MyStruct2 tag:



In each of the two examples, the value 10 of the IN parameter is copied twice to the operand at the OUT parameter, starting with the element Member_2. This means that the value 10 is copied to the elements Member_2 and Member_3. The other two elements, Member_1 and

Member_4, are not changed. If the instruction is executed without errors, the TagOut operand is set to signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

UFILL_BLK: Fill block uninterruptible (S7-1200, S7-1500)

Description

You can use the "Fill block uninterruptible" instruction to fill a memory area (target range) with the value of the IN input. The instruction cannot be interrupted. The target range is filled beginning with the address specified at the OUT output. The number of repeated copy operations is specified with the COUNT parameter. When the instruction is executed, the value at the input IN is moved to the target range as often as specified by the value of the COUNT parameter.

The instruction can only be executed if the source range and the target range have the same data type.

Note

The move operation cannot be interrupted by other operating system activities. This is why the alarm reaction times of the CPU increase during the execution of the "Fill block uninterruptible" instruction.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The maximum number of elements changed is the number of elements in the ARRAY or structure. If you copy more data than there are elements at the OUT output, you will get an unintended result.

When a ARRAY of BOOL is copied, the enable output ENO for an overflow is set to "1" until the byte limit of the ARRAY structure is exceeded. If the byte limit of the ARRAY structure is exceeded by the value at the COUNT input, the ENO enable output is reset to "0".

Note

Use of ARRAYs

The instruction reads the content from the selected element from the source range and copies this content n-times (n = depending on the value at the parameter COUNT) to the destination, starting at the specified index.

You use the "Fill block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Filling structures

As well as the elements of an ARRAY, you can also fill multiple elements of a structure (STRUCT, PLC data type) with the same value. The structure whose elements you want to fill must only contain individual elements of the same elementary data type. The structure itself can, however, be embedded in another structure.

Parameters

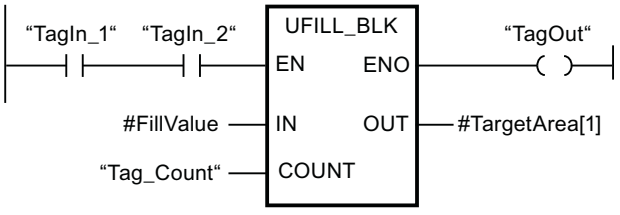
The following table shows the parameters of the "Fill block uninterruptible" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	I, Q, M, D, L, P, or constant	Element used to fill the target range.
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P, or constant	Number of repeated move operations
OUT	Output	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	Address in target range from which filling begins.

For additional information on valid data types, refer to "See also".

Example with an ARRAY

The following example shows how the instruction works when you want to fill an ARRAY:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	FillValue	The data type of the operand is INT.
COUNT	Tag_Count	3
OUT	TargetArea	The data type of the operand TargetArea is ARRAY[1..5] of INT. It consists of five elements of data type INT.

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The instruction copies the value of the operand #FillValue to the #TargetArea output tag three times, starting with the first element. The move operation cannot be interrupted by other operating system activities. If the instruction is executed without errors, the TagOut operand at the "ENO" enable output is set to signal state "1".

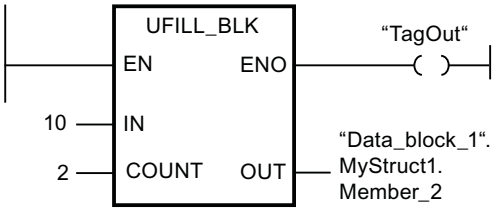
Examples with a structure

The following examples show how the instruction works when you want to fill a structure:

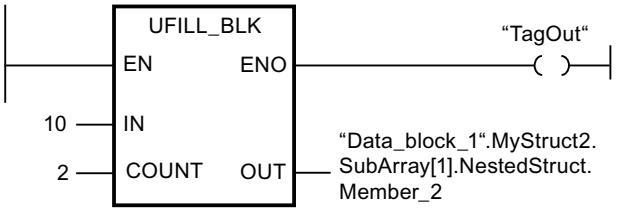
Create a global data block with the following elements:

Data_block_1					Data type
MyStruct1					STRUCT
	Member_1				INT
	Member_2				INT
	Member_3				INT
	Member_4				INT
MyStruct2					STRUCT
	SubArray				ARRAY[1..2] of STRUCT
		SubArray[1]			STRUCT
			NestedStruct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT
		SubArray[2]			STRUCT
			Nested-Struct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT

Generate the following program code to address the MyStruct1 tag:



Generate the following program code to address the MyStruct2 tag:



In each of the two examples, the value 10 of the IN parameter is copied twice to the operand at the OUT parameter, starting with the element Member_2. This means that the value 10 is copied to the elements Member_2 and Member_3. The other two elements, Member_1 and Member_4, are not changed. If the instruction is executed without errors, the TagOut operand is set to signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

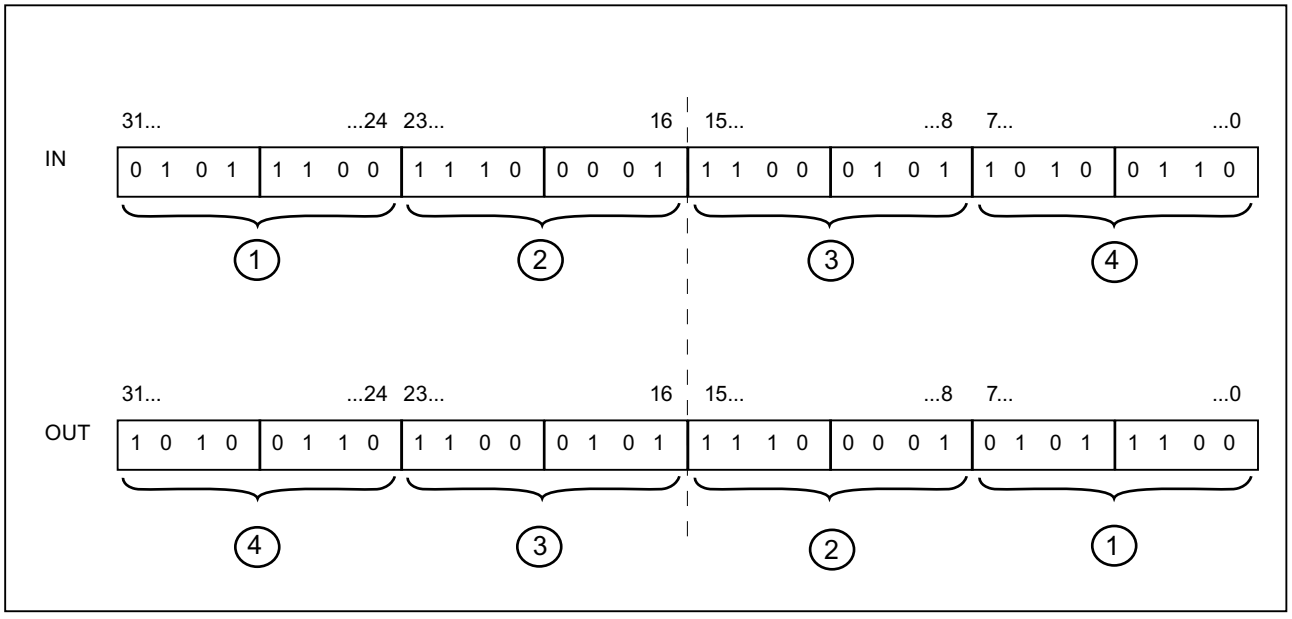
Basic information on LAD (Page 4878)

SWAP: Swap (S7-1200, S7-1500)

Description

You can use the "Swap" instruction to change the order of the bytes at input IN and query the result at output OUT.

The following figure shows how the bytes of an operand of the DWORD data type are swapped using the "Swap" instruction:



Parameters

The following table lists the parameters of the "Swap" instruction:

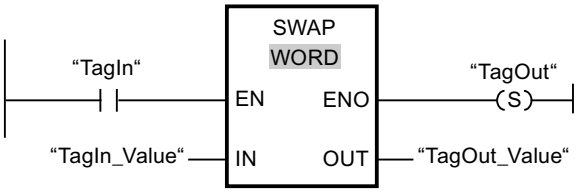
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN	Input	WORD, DWORD	WORD, DWORD, LWORD	I, Q, M, D, L or, P constant	Operand whose bytes are swapped.
OUT	Output	WORD, DWORD	WORD, DWORD, LWORD	I, Q, M, D, L, P	Result

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	TagIn_Value	0000 1111 0101 0101
OUT	TagOut_Value	0101 0101 0000 1111

If operand "TagIn" has the signal state "1", the "Swap" instruction is executed. The order of the bytes is changed and stored in operand "TagOut_Value".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ARRAY DB (S7-1500)

ReadFromArrayDB: Read from ARRAYdata block (S7-1500)

Description

The instruction "Read from ARRAY data block" is used to read the element from a data block of the ARRAY DB block type to which the index references, and write the value of the element to the target range.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

Enable output ENO returns the signal state FALSE if one of the following conditions applies:

- Enable input EN returns the signal state FALSE.
- If an error occurs during the execution of the instruction.

Parameters

The following table shows the parameters of the "Read from ARRAY data block" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
DB	Input	DB_ANY	I, Q, M, D, L	Data block that is read
INDEX	Input	DINT	I, Q, M, D, L, P or constant	Element in the DB that is read. The specification can be a constant, a global tag or an indexed value.

Parameters	Declaration	Data type	Memory area	Description
VALUE	Output [1)]	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Value that is read and output
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.
1) The VALUE parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

You can find additional information on valid data types under "See also".

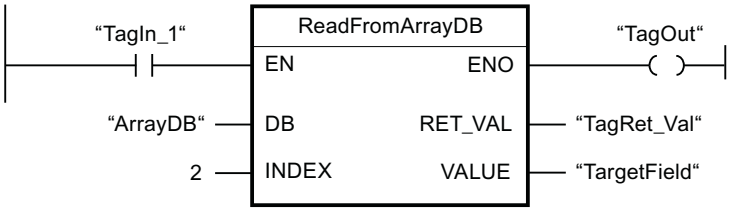
Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
80B5	The copying was interrupted.
8132	The data block does not exist, is too short, write protected, or is located in load memory.
8135	The ARRAY data block contains invalid values.
8154	The data block has the incorrect data type.
8282	The value at the INDEX parameter is outside the limits of the ARRAY.
8450	The data type VARIANT at parameter VALUE provides the value "0".
8452	Code generation error
8453	There are two possible causes of the error: • The size of the VALUE parameter does not match the element length in the ARRAY data block. • The two tags are not in a memory area with optimized access. For additional information on the memory area access types, refer to: AUTOHOTSPOT
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type Array [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	TargetField	The "TargetField" operand is a global tag of the data type INT.

If the "TagIn1" operand returns signal state "1", the instruction is executed. The 2nd element is read from "ArrayDB" and written to the "TargetField" operand. If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

Basic information on LAD (Page 4878)

WriteToArrayDB: Write to ARRAY data block (S7-1500)

Description

The instruction "Write to ARRAY data block" is used to write the element to which the index references to a data block of the ARRAY DB block type.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

Enable output ENO returns the signal state FALSE if one of the following conditions applies:

- Enable input EN returns the signal state FALSE.
- If an error occurs during the execution of the instruction.

Parameters

The following table shows the parameters of the "Write to ARRAY data block" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
DB	Input	DB_ANY	I, Q, M, D, L	Data block to which data is written
INDEX	Input	DINT	I, Q, M, D, L, P or constant	Element in the DB to which data is written. The specification can be a constant, a global tag or an indexed value.
VALUE	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Value to be written
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.

You can find additional information on valid data types under "See also".

Parameter RET_VAL

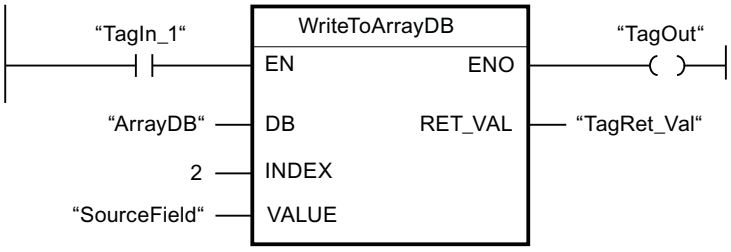
The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
80B5	The copying was interrupted.
8132	The data block does not exist, is too short, or is located in load memory.
8134	The data block is write protected.
8135	The data block is not an ARRAY data block.
8154	The data block has the incorrect data type.
8282	The value at the INDEX parameter is outside the limits of the ARRAY.
8350	The data type VARIANT at parameter VALUE provides the value "0".
8352	Code generation error

Error code* (W#16#...)	Explanation
8353	There are two possible causes of the error: • The size of the VALUE parameter does not match the element length in the ARRAY data block. • The two tags are not in a memory area with optimized access. For additional information on the memory area access types, refer to: AUTOHOTSPOT
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type Array [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	SourceField	The "SourceField" operand is a global tag of the data type INT.

If the "TagIn1" operand returns signal state "1", the instruction is executed. The value of the "SourceField" operand is written to the 2nd element of the ARRAY DB. If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

Basic information on LAD (Page 4878)

ReadFromArrayDBL: Read from ARRAY data block in load memory (S7-1500)

Description

The instruction "Read from ARRAY data block in load memory" is used to read the element ti which the index references from a data block of the ARRAY DB block type in the load memory and write it to the target range.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

If the ARRAY data block has been designated with the block attribute "Only store in load memory", it will only be stored in the load memory.

The instruction is executed when a positive signal edge is detected at the REQ parameter. The BUSY parameter then has the signal state "1". The instruction is terminated if a negative signal edge is detected at the BUSY parameter. The DONE parameter has the signal state "1" for one program cycle and the read value is output at the VALUE parameter within this cycle. With all other program cycles, the value at the VALUE parameter is not changed.

Enable output ENO returns the signal state FALSE if one of the following conditions applies:

- Enable input EN returns the signal state FALSE.
- If an error occurs during the execution of the instruction.

Parameters

The following table shows the parameters of the "Read from ARRAY data block in load memory" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = "1": Begin with the reading of the ARRAY DB
DB [1)]	Input	DB_ANY	I, Q, M, D, L	ARRAY data block that is read.
INDEX	Input	DINT	I, Q, M, D, L, P or constant	Element in the DB that is read. The specification can be a constant, a global tag or an indexed value.
VALUE [1)]	InOut	VARIANT	D (element of a global data block) L (A declaration in the "Temp" section of the block interface is not possible.)	Pointer to the DB in the work memory that is to be read and the value of which is to be written. No local constants or tags from the TEMP section must be used.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = "1": The array DB is still being read

Parameter	Declaration	Data type	Memory area	Description
DONE	Output	BOOL	I, Q, M, D, L	DONE = "1": The instruction was executed successfully
ERROR	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the ERROR parameter.
[1] The data blocks must be created with the "Optimized" block property.				

Parameter ERROR

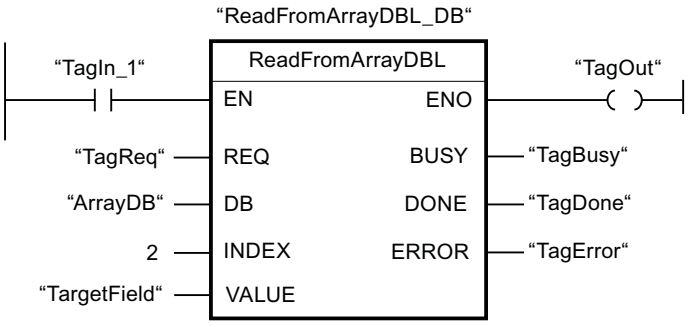
The following table shows the meaning of the values of the ERROR parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
8230	The data block number is incorrect.
8231	The data block does not exist.
8232	The data block is too short, or is not located in load memory.
8235	The data block is not an ARRAY DB.
8254	The data block has the incorrect data type.
8382	The value at the INDEX parameter is outside the limits of the ARRAY.
8750	The data type VARIANT at parameter VALUE provides the value "0".
8751	Code generation error
8752	Code generation error
8753	The size of the VALUE parameter does not match the element length in the ARRAY data block.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

You can find descriptions of the error codes triggered by the instructions "READ_DBL" and "WRIT_DBL" in the corresponding descriptions of the instructions.

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
REQ	TagReq	BOOL
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type ARRAY [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	TargetField	The "TargetField" operand is a global tag of the data type INT.
BUSY	TagBusy	BOOL
DONE	TagDone	BOOL

If the "TagIn1" operand has the signal state "1" and a positive edge is detected at the "TagReq" operand, the instruction is executed. The 2nd element is read from "ArrayDB" and output at the VALUE parameter. As soon as a negative signal edge is detected at the "TagBusy" operand, the instruction is terminated and the value at the VALUE parameter is no longer changed. If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set. After the instruction has been processed, the "TagDone" operand has the signal state TRUE.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

READ_DBL: Read from data block in the load memory (Page 4050)

WRIT_DBL: Write to data block in the load memory (Page 4054)

Basic information on VARIANT (Page 2308)

Difference between synchronous and asynchronous instructions (Page 2571)

Basic information on LAD (Page 4878)

WriteToArrayDBL: Write to ARRAY data block in load memory (S7-1500)

Description

The instruction "Write to ARRAY data block in load memory" is used to write the element to which the index references to a data block of the ARRAY DB block type in load memory.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

If the ARRAY data block has been designated with the block attribute "Only store in load memory", it will only be stored in the load memory.

The instruction is executed when a positive signal edge is detected at the REQ parameter. The BUSY parameter then has the signal state "1". If a negative signal edge is detected at the BUSY parameter, the instruction is terminated and the value at the VALUE parameter is written to the data block. The DONE parameter then has the signal state "1" for one program cycle.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- If an error occurs during the execution of the instruction.

Parameters

The following table shows the parameters of the "Write to ARRAY data block in load memory" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = "1": Start writing to the array DB
DB [1)]	Input	DB_ANY	I, Q, M, D, L	ARRAY data block to which data is written
INDEX	Input	DINT	I, Q, M, D, L, P or constant	Element in the DB to which data is written. The specification can be a constant, a global tag or an indexed value.
VALUE [1)]	Input	VARIANT	D (element of a global data block) L (A declaration in the "Temp" section of the block interface is not possible.)	Pointer to the DB in the work memory that is to be read and the value of which is to be written. No local constants or tags from the TEMP section must be used.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = "1": The array DB is still being written
DONE	Output	BOOL	I, Q, M, D, L	DONE = "1": The instruction was executed successfully

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the ERROR parameter.
[1] The data blocks must be created with the "Optimized" block property.				

Parameter ERROR

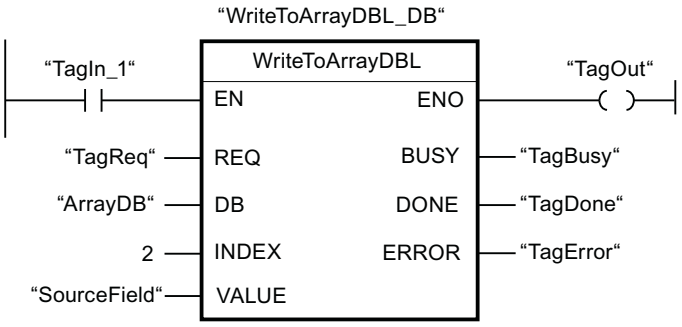
The following table shows the meaning of the values of the ERROR parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
8230	The data block number is incorrect.
8231	The data block does not exist.
8232	The data block is too short, or is not located in load memory.
8234	The data block is write protected.
8235	The data block is not an ARRAY DB.
8254	The data block has the incorrect data type.
8382	The value at the INDEX parameter is outside the limits of the ARRAY.
8450	The data type VARIANT at parameter VALUE provides the value "0".
8751	Code generation error
8752	Code generation error
8753	The size of the VALUE parameter does not match the element length in the ARRAY data block.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

You can find descriptions of the error codes triggered by the instructions "READ_DBL" and "WRIT_DBL" in the corresponding descriptions of the instructions.

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
REQ	TagReq	BOOL
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type ARRAY [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	SourceField	The "SourceField" operand is a global tag of the data type INT.
BUSY	TagBusy	BOOL
DONE	TagDone	BOOL

If the "TagIn1" operand has the signal state "1" and a positive edge is detected at the "TagReq" operand, the instruction is executed. As soon as a negative signal edge is detected at the "TagBusy" operand, the instruction is terminated and the value at the VALUE parameter is written to the 2nd element of the "ArrayDB". If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set. After the instruction has been processed, the "TagDone" operand has the signal state TRUE.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

READ_DBL: Read from data block in the load memory (Page 4050)

WRIT_DBL: Write to data block in the load memory (Page 4054)

Basic information on VARIANT (Page 2308)

Difference between synchronous and asynchronous instructions (Page 2571)

Basic information on LAD (Page 4878)

VARIANT (S7-1200, S7-1500)

VariantGet: Read out VARIANT tag value (S7-1200, S7-1500)

Description

You can use the "Read out VARIANT tag value" instruction to read the value of the tag to which the VARIANT at the SRC parameter points and write it in the tag at the DST parameter.

The SRC parameter has the VARIANT data type. Any data type except for VARIANT can be specified at the DST parameter.

The data type of the tag at the DST parameter must match the data type to which the VARIANT points.

Note

To copy structures and ARRAYs, you can use the "MOVE_BLK_VARIANT: Move block" instruction. For additional information, refer to "See also".

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The data types do not correspond (no value is transferred).

Parameter

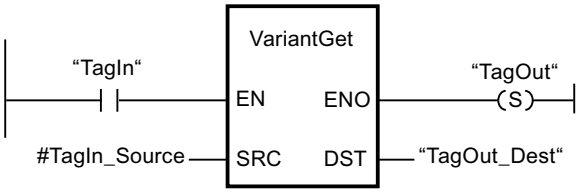
The following table shows the parameters of the instruction "Read out VARIANT tag value":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
SRC	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	Tag to be read
DST	Output	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY elements, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Result of the instruction

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The value of the tag to which the VARIANT at the "#TagIn_Source" operand points, is read and written to the "TagOut_Dest" operand.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

MOVE_BLK_VARIANT: Move block (Page 2757)

Basic information on LAD (Page 4878)

VariantPut: Write VARIANT tag value (S7-1200, S7-1500)

Description

You can use the "Write VARIANT tag value" instruction to write the value of the tag at the SRC parameter to the memory at the DST parameter to which the VARIANT points.

The DST parameter has the VARIANT data type. Any data type except for VARIANT can be specified at the SRC parameter.

The data type of the tag at the SRC parameter must match the data type to which the VARIANT points.

Note

To copy structures and ARRAYs, you can use the "MOVE_BLK_VARIANT instruction: Move block" instruction. For additional information, refer to "See also".

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The data types do not correspond. (No value is transferred.)

Parameters

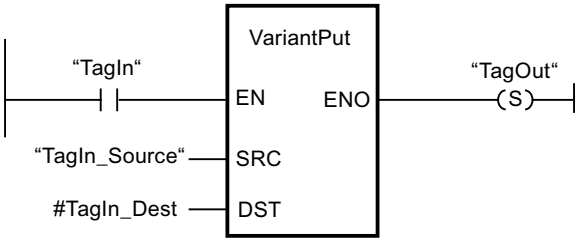
The following table shows the parameters of the instruction "Write VARIANT tag value":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
SRC	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY elements, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Tag to be read
DST	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	Result of the instruction

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand returns signal state "1", the instruction is executed. The value of the "TagIn_Source" operand is written to the tag to which the VARIANT at the #TagIn_Dest operand points.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

MOVE_BLK_VARIANT: Move block (Page 2757)

Basic information on LAD (Page 4878)

CountOfElements: Get number of ARRAY elements (S7-1200, S7-1500)

Description

You use the "Get number of ARRAY elements" instruction to query how many ARRAY elements a tag to which VARIANT points has.

If it is a multi-dimensional ARRAY, the number of ARRAY elements is output as the result. (The difference between the high and low limit +1). If it is a multi-dimensional ARRAY, the number of all dimensions is output as the result.

Note

Instances

The VARIANT pointer can point to no instance and therefore also to no multi-instance or an ARRAY of multi-instances.

Note

ARRAY within a data block

If you want to query the number of the elements of an ARRAY which is located in a data block, the block attribute "Data block write-protected in the device" must not be activated in this block. Otherwise, the RET_VAL parameter returns the result "0", irrespective of how many elements the ARRAY contains.

The result is also "0" if the VARIANT tag is not an ARRAY.

If the VARIANT points to an ARRAY of BOOL, the fill elements are included in the count. (For example, 8 is returned for an ARRAY[0..1] of BOOL).

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The VARIANT tag is not an ARRAY. (The result is "0".)

Parameters

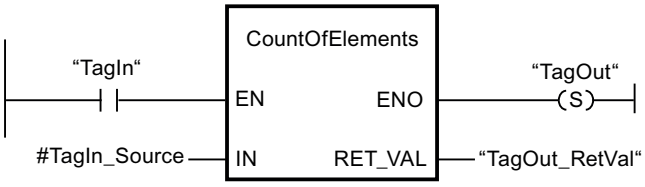
The following table shows the parameters of the "Get number of ARRAY elements" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Tag to be queried
RET_VAL	Output	UDINT	I, Q, M, D, L	Result of the instruction

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand returns signal state "1", the instruction is executed. The number of ARRAY elements of the tag to which the VARIANT at the #TagIn_Source operand points is read and output at the "TagOut_RetVal" operand.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ARRAY[*] (S7-1200, S7-1500)

LOWER_BOUND: Read out low ARRAY limit (S7-1200, S7-1500)

Description

You can declare tags of the data type ARRAY[*] in the block interface of a function block or function. For these local tags, you can read out the limits of the ARRAY. You will need to specify the required dimension at the DIM parameter.

The "Read out ARRAY low limit" instruction is used to read out the variable low limit of the ARRAY.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The dimension specified at input DIM does not exist.

Note

Availability of the instruction

The instruction is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

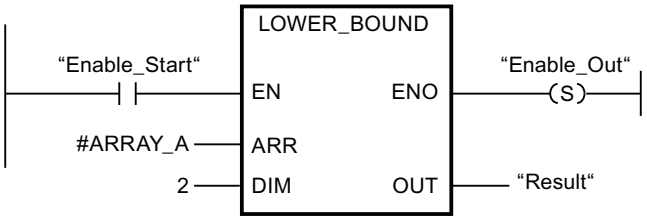
Parameters

The following table shows the parameters of the instruction "Read out ARRAY low limit":

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
ARR	Input	ARRAY[*]	FB => Section In-Out FC => Sections Input and InOut	ARRAY of which the variable low limit is to be read.
DIM	Input	UDINT	I, Q, M, D, L, P or constant	Dimension of the ARRAY of which the variable low limit is to be read.
OUT	Output	DINT	I, Q, M, D, L, P	Result

Example

The following example shows how the instruction works:



If the "Enable_Start" operand returns signal state "1", the instruction is executed. It reads out the variable low limit of the ARRAY #ARRAY_A from the second dimension. If the instruction is executed without errors, the result is written to operand "Enable_Out" and operand "Result" is set.

See also

Overview of the valid data types (Page 2239)

Basic information on ARRAY (Page 2286)

Basic information on LAD (Page 4878)

UPPER_BOUND: Read out high ARRAY limit (S7-1200, S7-1500)

Description

You can declare tags of the data type ARRAY[*] in the block interface of a function block or function. For these local tags, you can read out the limits of the ARRAY. You will need to specify the required dimension at the DIM parameter.

The instruction "Read out ARRAY high limit" is used to read out the variable high limit of the ARRAY.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The dimension specified at input DIM does not exist.

Note

Availability of the instruction

The instruction is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

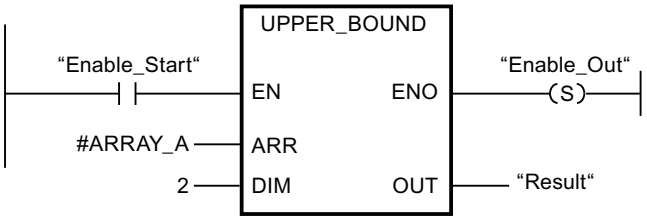
Parameters

The following table shows the parameters of the instruction "Read out ARRAY high limit":

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
ARR	Input	ARRAY[*]	FB => Section In-Out FC => Sections Input and InOut	ARRAY of which the variable high limit is to be read.
DIM	Input	UDINT	I, Q, M, D, L, P or constant	Dimension of the ARRAY of which the variable high limit is to be read.
OUT	Output	DINT	I, Q, M, D, L, P	Result

Example

The following example shows how the instruction works:



If the "Enable_Start" operand returns signal state "1", the instruction is executed. It reads out the variable high limit of the ARRAY #ARRAY_A from the second dimension. If the instruction is executed without errors, the result is written to operand "Enable_Out" and operand "Result" is set.

See also

Overview of the valid data types (Page 2239)

Basic information on ARRAY (Page 2286)

Basic information on LAD (Page 4878)

Legacy (S7-1500)

FieldRead: Read field (S7-1500)

Description

You can use the "Read field" instruction to read out a specific component from the field specified at input MEMBER and transfer its content to the tag at output VALUE. You specify the index of the field component to be read at input INDEX. Specify the first component of the field from which reading is to occur at input MEMBER.

The data types of the field component at parameter MEMBER, the index and the tags at parameter VALUE must correspond to the data type of the instruction "Read field" because implicit conversion is not possible.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The field component specified at input INDEX is not defined in the field specified at output MEMBER.
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the instruction "Read field":

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
INDEX	Input	DINT	DINT	I, Q, M, D, L, P, or constant	Index of field component whose content is read out

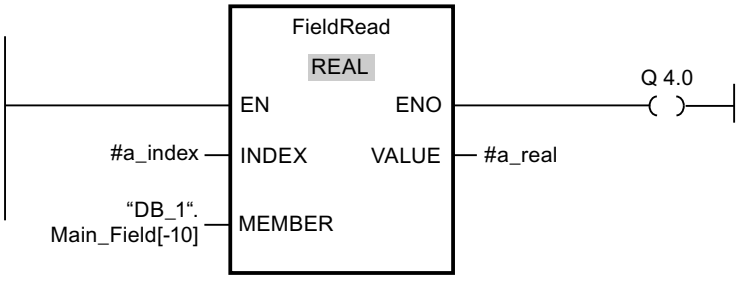
Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
MEMBER	Input	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR and WCHAR as components of an ARRAY tag	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD, CHAR and WCHAR as components of an ARRAY tag	D, L	First component of the field from which reading occurs.
VALUE	Output	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD CHAR, WCHAR	I, Q, M, D, L, P	Operand to which the content of the field component is transferred

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Tag	Value
INDEX	a_index	4
MEMBER	"DB_1".Main_Field[-10]	First component of the "Main_Field[-10..10] of REAL" field in the data block "DB_1"
VALUE	a_real	Component with index 4 of the "Main_Field[-10..10] of REAL" field

The field component with index 4 is read from the "Main_Field[-10...10] of REAL" field and written to the "a_real" tag. The field component to be read is specified by the value at input INDEX.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

FieldWrite: Write field (S7-1500)

Description

The "Write field" instruction is used to transfer the content of the tag at the VALUE input to a specific component of the field at the MEMBER output. You can use the value at the INDEX input to specify the index of the field component that is described. At the MEMBER output, enter the first component of the field which is to be written to.

The data types of the field component at parameter MEMBER, the index and the tags at parameter VALUE must correspond to the data type of the instruction "Read field" because implicit conversion is not possible.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The field component specified at the input INDEX is not defined in the field specified at the output MEMBER.
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the instruction "Write field":

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
INDEX	Input	DINT	DINT	I, Q, M, D, L, P, or constant	Index of the field component that is being written with the content of VALUE.

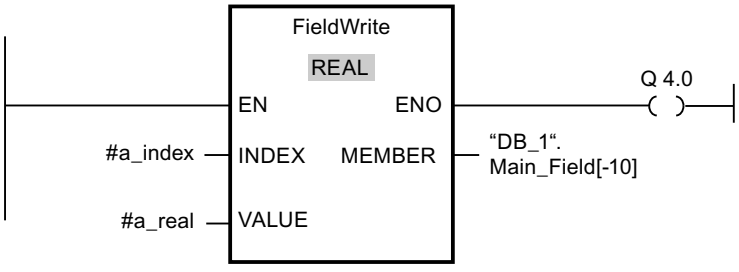
Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
VALUE	Input	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD, CHAR, WCHAR	I, Q, M, D, L, P, or constant	Operand whose content is copied.
MEMBER	Output	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR and WCHAR as components of an ARRAY tag	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD, CHAR and WCHAR as components of an ARRAY tag	D, L	First component of the field to which the content of VALUE is written.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
INDEX	a_index	4
VALUE	a_real	10.54
MEMBER	"DB_1".Main_Field[-10]	First component of the "Main_Field[-10..10] of REAL" field in the data block "DB_1"

The value "10.54" of the "a_real" tag is written to the field component with index 4 of the "Main_Field[-10..10] of REAL" field. The index of the field component to which the content of the tag "a_real" is transferred is specified by the value at the input INDEX.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

BLKMOV: Move block (S7-1500)

Description

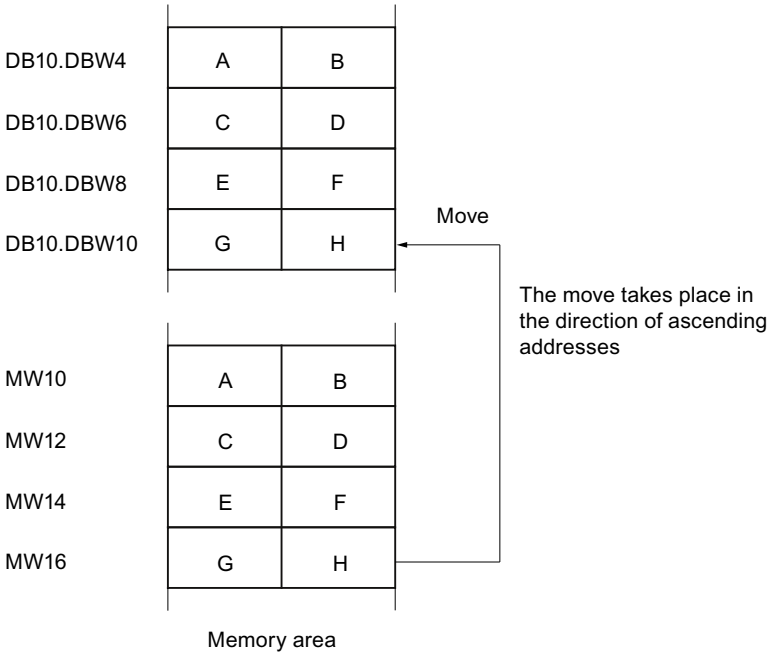
You can use the "Move block" instruction to move the content of a memory area (source area) to another memory area (destination area). The move operation takes place in the direction of ascending addresses. You use VARIANT to define the source and destination areas.

Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can use this tag in memory areas "with optimized block access".

The following figure shows the principle of the move operation:



Consistency of source and destination data

Make sure that the source data remain unchanged during the execution of the "Move block" instruction. Otherwise, consistency of the destination data cannot be guaranteed.

Interruptibility

There is no limit to the nesting depth.

Memory areas

You can use the "Move block" instruction to move the following memory areas:

- Areas of a data block
- Bit memory
- Process image input
- Process image output

General rules for moving

The source and destination areas must not overlap. If the source and destination areas have different lengths, only the length of the smaller area will be moved.

If the source area is smaller than the destination area, the entire source area will be written to the destination area. The remaining bytes of the destination area remain unchanged.

If the destination area is smaller than the source area, the entire destination area will be written. The remaining bytes of the source area are ignored.

If a block of data type BOOL is moved, the tag must be addressed absolutely and the specified length of the area must be divisible by 8, otherwise the instruction cannot be executed.

Rules for moving character strings

You can use the "Move block" instruction to also move source and destination areas of the STRING data type. If only the source area is STRING data type, the characters will be moved that are actually contained in the character string. Information on the actual and maximum length is also written to the destination area. If the source and destination area are both STRING data type, the current length of the character string in the destination area is set to the number of characters actually moved.

If you want to move the information on the maximum and actual length of a character string, specify the areas in bytes in the SRCBLK and DSTBLK parameters.

Parameters

The following table shows the parameters of the "Move block" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRCBLK	Input	VARIANT	I, Q, M, D, L, P	Specifies the memory area to be moved (source area).

Parameters	Declaration	Data type	Memory area	Description
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.
DSTBLK	Output [1]	VARIANT	I, Q, M, D, L, P	Specifies the memory area to which the block is to be moved (destination area).
1) The DSTBLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

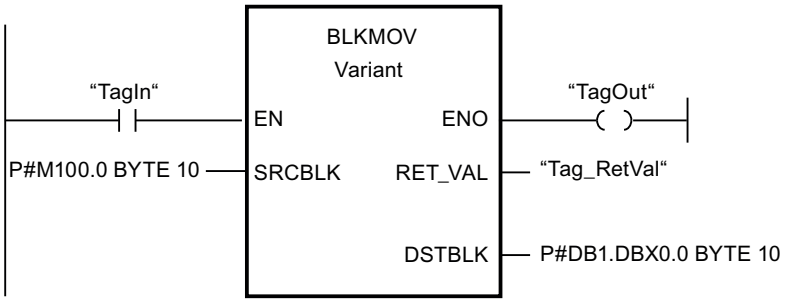
Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8092	The source or destination area is located in the load memory only.
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the SRCBLK parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the DSTBLK parameter.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction copies 10 bytes starting from MB100 and writes them to DB1. If an error occurs during the move operation, its error code is output in the "Tag_RetVal" tag.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 2840)

Basic information on LAD (Page 4878)

UBLKMOV: Move block uninterruptible (S7-1500)

Description

You can use the "Move block uninterruptible" instruction to move the content of a memory area (source area) to another memory area (destination area). The move operation takes place in the direction of ascending addresses. You use VARIANT to define the source and destination areas.

The move operation cannot be interrupted by other operating system activities. As a result, the interrupt reaction time of the CPU can increase during the execution of the "Move block uninterruptible" instruction.

Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can use this tag in memory areas "with optimized block access".

Memory areas

You can use the "Move block uninterruptible" instruction to move the following memory areas:

- Areas of a data block
- Bit memory
- Process image input
- Process image output

General rules for moving

The source and destination areas must not overlap during the execution of the "Move block uninterruptible" instruction. If the source area is smaller than the destination area, the entire source area will be written to the destination area. The remaining bytes of the destination area remain unchanged.

If the destination area is smaller than the source area, the entire destination area will be written. The remaining bytes of the source area are ignored.

If a destination or source area defined as a formal parameter is less than a destination or source area specified in the SRCBLK or DSTBLK parameter, no data is transferred.

If a block of data type BOOL is moved, the tag must be addressed absolutely and the specified length of the area must be divisible by 8, otherwise the instruction cannot be executed.

You can use the "Move block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Rules for moving character strings

You can use the "Move block uninterruptible" instruction to also move source and destination areas of the STRING data type. If only the source area is STRING data type, the characters will be moved that are actually contained in the character string. Information on the actual and maximum length are not written in the destination area. If the source and destination area are both STRING data type, the current length of the character string in the destination area is set to the number of characters actually moved. If areas of the STRING data type are moved, specify "1" as area length.

Parameters

The following table shows the parameters of the "Move block uninterruptible" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRCBLK	Input	VARIANT	I, Q, M, D, L, P	Specifies the memory area to be moved (source area).
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.
DSTBLK	Output [1)]	VARIANT	I, Q, M, D, L, P	Specifies the memory area to which the block is to be moved (destination area).
1) The DSTBLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

Parameter RET_VAL

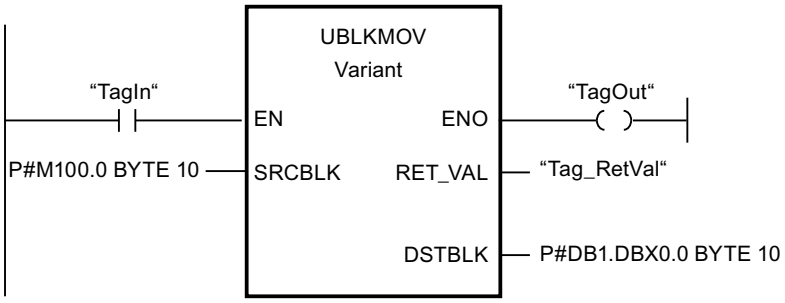
The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8091	The source or destination area is located in the load memory only.
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the SRCBLK parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the DSTBLK parameter.

Error code* (W#16#...)	Explanation
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction copies 10 bytes starting from MB100 and writes them to DB1. If an error occurs during the move operation, its error code is output in the "Tag_RetVal" tag.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 2840)

Basic information on LAD (Page 4878)

FILL: Fill block (S7-1500)

Description

You can use the "Fill block" instruction to fill a memory area (destination area) with the content of another memory area (source area). The "Fill block" instruction moves the content of the source area to the destination area until the destination area is completely written. The move operation takes place in the direction of ascending addresses.

You use VARIANT to define the source and destination areas.

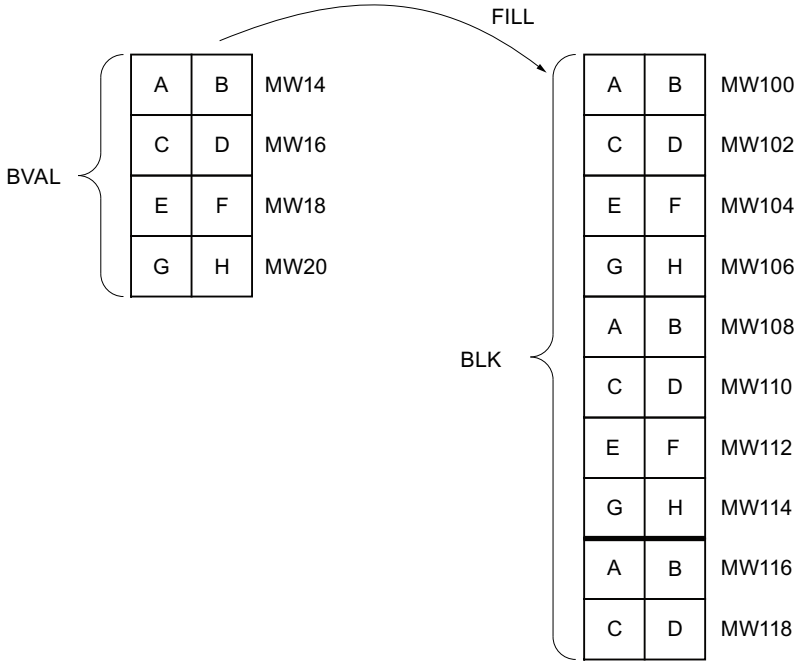
Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can use this tag in memory areas "with optimized block access".

For blocks with the "Optimized block access" attribute, you can use the "FILL_BLK instruction. Fill block" instruction.

The following figure shows the principle of the move operation:



Example: The contents of the range MW100 to MW118 are to be preassigned with the contents of the memory words MW14 to MW20.

Consistency of source and destination data

Note that while the instruction "Fill block" is being executed, the source data remains unchanged; otherwise the consistency of the destination data cannot be guaranteed.

Memory areas

You can use the "Fill block" instruction to move the following memory areas:

- Areas of a data block
- Bit memory

- Process image input
- Process image output

General rules for moving

The source and destination areas must not overlap. If the destination area to be preset is not an integer multiple of the length of the input parameter BVAL, the destination area is nevertheless written up to the last byte.

If the destination area to be preset is smaller than the source area, the function only copies as much data as can be written to the destination area.

If the destination or source area actually present is smaller than the assigned memory area for the source or destination area (BVAL, BLK parameters), no data is transferred.

If the ANY pointer (source or destination) is of the data type BOOL, it must be addressed absolutely and the length specified must be divisible by 8; otherwise the instruction is not executed.

If the destination area is STRING data type, the instruction writes the entire string including the administration information.

Rules for moving structures

When you transfer a structure as an input parameter you must bear in mind that the length of a structure is always oriented to an even number of bytes. The structure will need one byte of additional memory space if you declare a structure with an odd number of bytes.

Parameters

The following table shows the parameters of the "Fill block" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
BVAL	Input	VARIANT	I, Q, M, D, L, P	Specification of the memory area (source area), the content of which is used to fill the destination area at the BLK parameter.
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.
BLK	Output [1)]	VARIANT	I, Q, M, D, L, P	Specification of the memory area that will be filled with the content of the source area.
1) The BLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

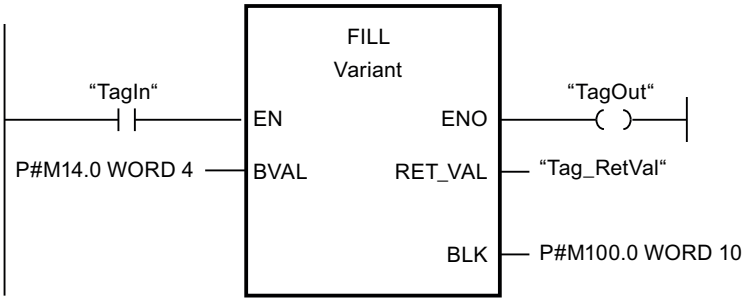
Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8092	The source or destination area is located in the load memory only.
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the BVAL parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the BLK parameter.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction copies the source area from MW14 to MW20 and fills the destination area from MW100 to MW118 with the content of the 4 words contained in the memory area in the BVAL parameter.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 2840)

Basic information on LAD (Page 4878)

Conversion operations (S7-1200, S7-1500)

CONVERT: Convert value (S7-1200, S7-1500)

Description

The "Convert value" instruction reads the content of the IN parameter and converts it according to the data types selected in the instruction box. The converted value is output at the OUT output.

For information on possible conversions, refer to the "Explicit conversion" section at "See also".

The enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- Errors, such as an overflow, occur during execution.

Conversion options for bit strings

BYTE and WORD bit strings cannot be selected in the instruction box. However, it is possible to specify an operand of data type DWORD or LWORD at a parameter of the instruction if the lengths of the input and output operands match. The operand is then interpreted by the data type of a bit string according to the data type of the input or output parameter and is implicitly converted. The data type DWORD is for example interpreted as DINT/UDINT and LWORD as LINT/ULINT. These conversion options are also available to you when "IEC check" is enabled.

Note

For CPUs of the S7-1500 series: The data types DWORD and LWORD can only be converted to or from data type REAL or LREAL.

During the conversion, the bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If no errors occur during the conversion, the signal state of enable output ENO = 1; if an error occurs during processing, the signal state of the enable output ENO = 0.

Parameters

The following table shows the parameters of the "Convert value" instruction:

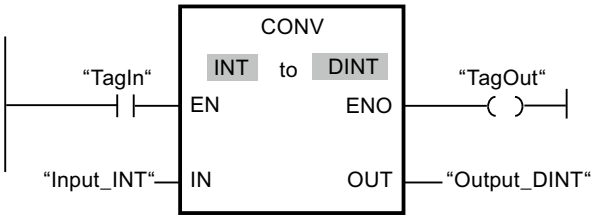
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output

Parameter	Declaration	Data type	Memory area	Description
IN	Input	Bit strings, integers, floating-point numbers, CHAR, WCHAR, BCD16, BCD32	I, Q, M, D, L, P or constant	Value to be converted.
OUT	Output	Bit strings, integers, floating-point numbers, CHAR, WCHAR, BCD16, BCD32	I, Q, M, D, L, P	Result of the conversion

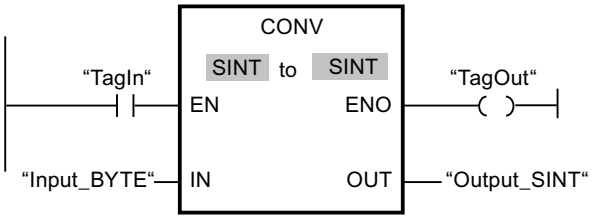
You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

Examples

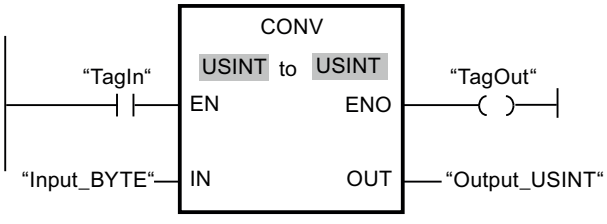
The following example shows the conversion of an integer (16 bit) to another integer (32 bit):



The following example shows the conversion of a byte (8 bit) to the integer SINT (8 bit):



The following example shows the conversion of a byte (8 bit) to the unsigned integer USINT (8 bit):



The conversions are possible, because the length of the two operands is the same.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

Basic information on LAD (Page 4878)

ROUND: Round numerical value (S7-1200, S7-1500)

Description

You can use the "Round numerical value" instruction to round the value at input IN to the nearest integer. The instruction interprets the value at input IN as a floating-point number and converts this to an integer of data type DINT. If the input value is exactly between an even and odd number, the even number is selected. The result of the instruction is sent to the OUT output and can be queried there.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the "Round numerical value" instruction:

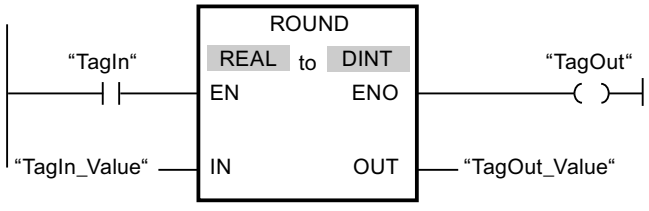
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Input value to be rounded.
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	Result of rounding

You can select the data type for the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value	
IN	TagIn_Value	1.50000000	-1.50000000
OUT	TagOut_Value	2	-2

If the "TagIn" operand has signal state "1", the instruction is executed. The floating-point number at input "TagIn_Value" is rounded to the nearest even integer and sent to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

CEIL: Generate next higher integer from floating-point number (S7-1200, S7-1500)

Description

You can use the "Generate next higher integer from floating-point number" instruction to round the value at input IN to the next higher integer. The instruction interprets the value at the IN input as a floating-point number and converts this number to the next higher integer. The result of the instruction is sent to the OUT output and can be queried there. The output value can be greater than or equal to the input value.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the "Generate next higher integer from floating-point number" instruction:

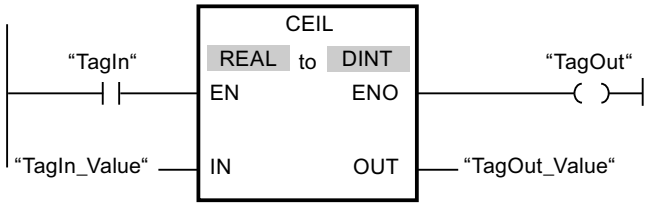
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	Result with next higher integer

You can select the data type for the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value	
IN	TagIn_Value	0.50000000	-0.50000000
OUT	TagOut_Value	1	0

If the "TagIn" operand has signal state "1", the instruction is executed. The floating-point number at the "TagIn_Value" input is rounded to the next higher integer and sent to the "TagOut_Value" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

FLOOR: Generate next lower integer from floating-point number (S7-1200, S7-1500)

Description

You can use the "Generate next lower integer from floating-point number" instruction to round the value at input IN to the next lower integer. The instruction interprets the value at input IN as a floating-point number and converts this to the next lower integer. The result of the instruction is sent to the OUT output and can be queried there. The output value can be less than or equal to the input value.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the "Generate next lower integer from floating-point number" instruction:

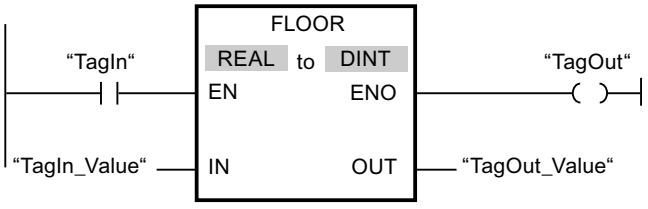
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	Result with next lower integer

You can select the data type for the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value	
IN	TagIn_Value	0.50000000	-0.50000000
OUT	TagOut_Value	0	-1

If the "TagIn" operand has signal state "1", the instruction is executed. The floating-point number at input "TagIn_Value" is rounded to the next lower integer and sent to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

TRUNC: Truncate numerical value (S7-1200, S7-1500)

Description

You can use the "Truncate numerical value" instruction to form an integer from the value at the IN input. The value at the IN input is interpreted as a floating-point number. The instruction selects only the integer part of the floating-point number and sends this to the OUT output without decimal places.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the "Truncate numerical value" instruction:

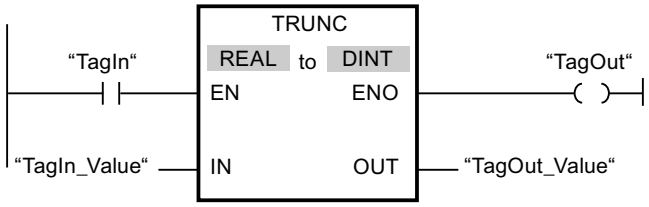
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L or constant	Input value
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L	Integer part of the input value

You can select the data type for the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value	
IN	TagIn_Value	1.50000000	-1.50000000
OUT	TagOut_Value	1	-1

If the "TagIn" operand has signal state "1", the instruction is executed. The integer part of the floating-point number at the "TagIn_Value" input is converted to an integer and sent to the "TagOut_Value" output. If the instruction is executed without errors, the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

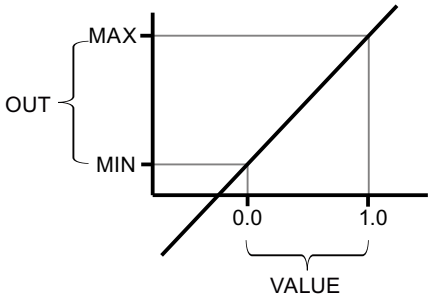
Basic information on LAD (Page 4878)

SCALE_X: Scale (S7-1200, S7-1500)

Description

You can use the "Scale" instruction to scale the value at the VALUE input by mapping it to a specified value range. When the "Scale" instruction is executed, the floating-point value at the VALUE input is scaled to the value range that was defined by the MIN and MAX parameters. The result of the scaling is an integer, which is stored in the OUT output.

The following figure shows an example of how values can be scaled:



The "Scale" instruction works with the following equation:

OUT = [VALUE * (MAX – MIN)] + MIN

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the MIN input is greater than or equal to the value at the MAX input.
- The value of a specified floating-point number is outside the range of the normalized numbers according to IEEE-754.
- An overflow occurs.
- The value at the VALUE input is NaN (Not a Number = result of an invalid arithmetic operation).

Note

For more information on the converting of analog values, refer to the respective manual.

Parameters

The following table shows the parameters of the "Scale" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
MIN	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Low limit of the value range
VALUE	Input	Floating-point numbers	I, Q, M, D, L or constant	Value to be scaled. If you enter a constant, you must declare it.
MAX	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	High limit of the value range
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L	Result of scaling

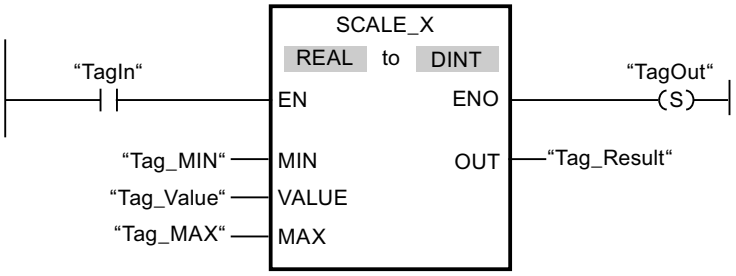
You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

You can find additional information on valid data types under "See also".

For additional information on declaring constants, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
MIN	Tag_MIN	10
VALUE	Tag_Value	0.5
MAX	Tag_MAX	30
OUT	Tag_Result	20

If the "TagIn" operand has signal state "1", the instruction is executed. The value at the "Tag_Value" input is scaled to the range of values defined by the values at the "Tag_MIN" and "Tag_MAX" inputs. The result is stored in the "Tag_Result" output. If the instruction is executed without errors, the ENO enable output has signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

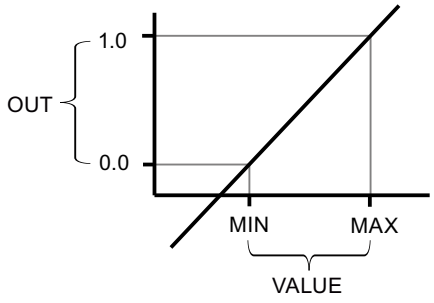
Basic information on LAD (Page 4878)

NORM_X: Normalize (S7-1200, S7-1500)

Description

You can use the instruction "Normalize" to normalize the value of the tag at the VALUE input by mapping it to a linear scale. You can use the MIN and MAX parameters to define the limits of a value range that is applied to the scale. The result at the OUT output is calculated and stored as a floating-point number depending on the location of the value to be normalized within this value range. If the value to be normalized is equal to the value at the MIN input, the OUT output has the value "0.0". If the value to be normalized is equal to the value at input MAX, output OUT returns the value "1.0".

The following figure shows an example of how values can be normalized:



The "Normalize" instruction works with the following equation:

OUT = (VALUE – MIN) / (MAX – MIN)

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value at the MIN input is greater than or equal to the value at the MAX input.
- The value of a specified floating-point number is outside the range of the normalized numbers according to IEEE-754.
- The value at the VALUE input is NaN (result of an invalid arithmetic operation).

Note

For more information on the converting of analog values, refer to the respective manual.

Parameters

The following table shows the parameters of the "Normalize" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
MIN [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Low limit of the value range
VALUE [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Value to be normalized.
MAX [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	High limit of the value range

Parameter	Declaration	Data type	Memory area	Description
OUT	Output	Floating-point numbers	I, Q, M, D, L	Result of the normalization
[1] If you use constants in these three parameters, you only need to declare one of them.				

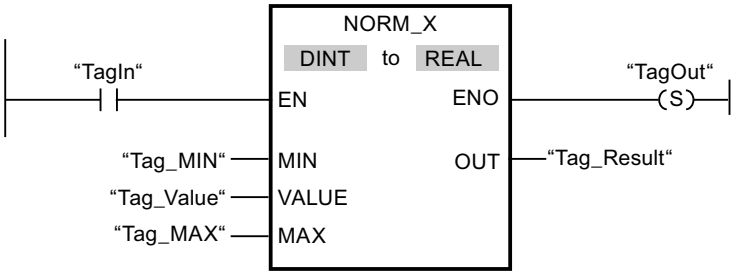
You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

You can find additional information on valid data types under "See also".

For additional information on declaring constants, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
MIN	Tag_MIN	10
VALUE	Tag_Value	20
MAX	Tag_MAX	30
OUT	Tag_Result	0.5

If the "TagIn" operand has signal state "1", the instruction is executed. The value at the "Tag_Value" input is mapped to the range of values that were defined by the values at the "Tag_MIN" and "Tag_MAX" inputs. The tag value at the "Tag_Value" input is normalized to the defined value range. The result is stored as a floating-point number in the "Tag_Result" output. If the instruction is executed without errors, the ENO enable output has signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Legacy (S7-1500)

SCALE: Scale (S7-1500)

Description

Use the "Scale" instruction to convert the integer at the IN parameter into a floating-point number, which can be scaled in physical units between a low limit value and a high limit value. You can use the LO_LIM and HI_LIM parameters to specify the low limit and high limit of the value range to which the input value is scaled. The result of the instruction is output on the OUT parameter.

The "Scale" instruction works with the following equation:

OUT = [((FLOAT (IN) – K1)/(K2–K1)) * (HI_LIM–LO_LIM)] + LO_LIM

The values of the constants "K1" and "K2" are determined by the signal state on the BIPOLAR parameter. The following signal states are possible on the BIPOLAR parameter:

- Signal state "1": It is assumed that the value at the IN parameter is bipolar and in a value range between -27648 and 27648. In this case the "K1" constant has the value "-27648.0" and the "K2" constant the value "+27648.0".
- Signal state "0": It is assumed that the value at the IN parameter is unipolar and in a value range between 0 and 27648. In this case the "K1" constant has the value "0,0" and the "K2" constant the value "+27648.0".

When the value at the IN parameter is greater than the value of the constant "K2", the result of the instruction is set to the value of the high limit (HI_LIM) and an error is output.

When the value at the IN parameter is less than the value of the constant "K1", the result of the instruction is set to the value of the low limit value (LO_LIM) and an error is output.

When the indicated low limit value is greater than the high limit value (LO_LIM > HI_LIM), the result is scaled in reverse proportion to the input value.

Parameter

The following table shows the parameters of the "Scale" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	INT	I, Q, M, D, L, P, or constant	Input value to be scaled.
HI_LIM	Input	REAL	I, Q, M, D, L, P, or constant	High limit
LO_LIM	Input	REAL	I, Q, M, D, L, P, or constant	Low limit

Parameter	Declaration	Data type	Memory area	Description
BIPOLAR	Input	BOOL	I, Q, M, D, L or constant	Indicates if the value at IN parameter is to be interpreted as bipolar or unipolar. The parameter can assume the following values: 1: Bipolar 0: Unipolar
OUT	Output	REAL	I, Q, M, D, L, P	Result of the instruction
RET_VAL	Output	WORD	I, Q, M, D, L, P	Error information

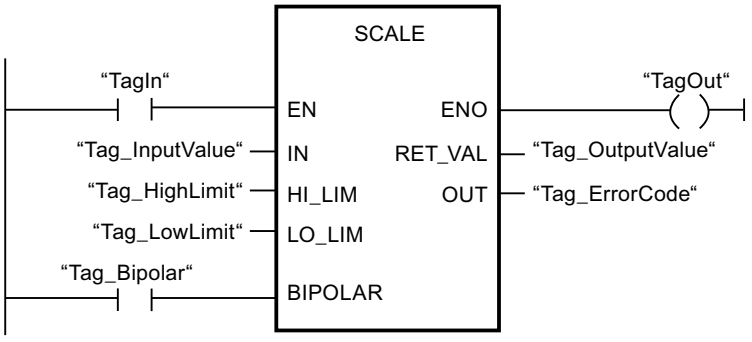
Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
0008	The value of the IN parameter is greater than the value of the constant "K2" or less than the value of the constant "K1"
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN	Tag_InputValue	22
HI_LIM	Tag_HighLimit	100.0
LO_LIM	Tag_LowLimit	0.0

Parameter	Operand	Value
BIPOLAR	Tag_Bipolar	1
OUT	Tag_OutputValue	50.03978588
RET_VAL	Tag_ErrorCode	W#16#0000

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 2840)

Basic information on LAD (Page 4878)

UNSCALE: Unscale (S7-1500)

Description

You use the "Unscale" instruction to unscale the floating-point number in the IN parameter into physical units between a low limit and a high limit and convert it into an integer. You use the LO_LIM and HI_LIM parameters to specify the low limit and high limit of the value range to which the input value is unscaled. The result of the instruction is output at the OUT parameter.

The "Unscale" instruction works with the following equation:

OUT = [((IN–LO_LIM)/(HI_LIM–LO_LIM)) * (K2–K1)] + K1

The values of the "K1" and "K2" constants are determined by the signal state at the BIPOLAR parameter. The following signal states are possible at the BIPOLAR parameter:

- Signal state "1": It is assumed that the value at the IN parameter is bipolar and in a value range between -27648 and 27648. In this case the "K1" constant has the value "-27648.0" and the "K2" constant the value "+27648.0".
- Signal state "0": It is assumed that the value at the IN parameter is unipolar and in a value range between 0 and 27648. In this case the "K1" constant has the value "0.0" and the "K2" constant the value "+27648.0".

When the value at parameter IN is not within the limits defined by HI_LIM and LO_LIM, an error is output.

When the indicated low limit value is greater than the high limit value (LO_LIM > HI_LIM), the result is scaled inversely proportionate to the input value.

Parameter

The following table shows the parameters of the "Unscale" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Input	BOOL	I, Q, M, D, L	Enable output

Parameters	Declaration	Data type	Memory area	Description
IN	Input	REAL	I, Q, M, D, L, P or constant	Input value to be unscaled to an integer value.
HI_LIM	Input	REAL	I, Q, M, D, L, P or constant	High limit
LO_LIM	Input	REAL	I, Q, M, D, L, P or constant	Low limit
BIPOLAR	Input	BOOL	I, Q, M, D, L or constant	Indicates whether the value at the IN parameter is to be interpreted as bipolar or unipolar. The parameter can assume the following values: 1: Bipolar 0: Unipolar
OUT	Output	INT	I, Q, M, D, L, P	Result of the instruction
RET_VAL	Output	WORD	I, Q, M, D, L, P	Error information

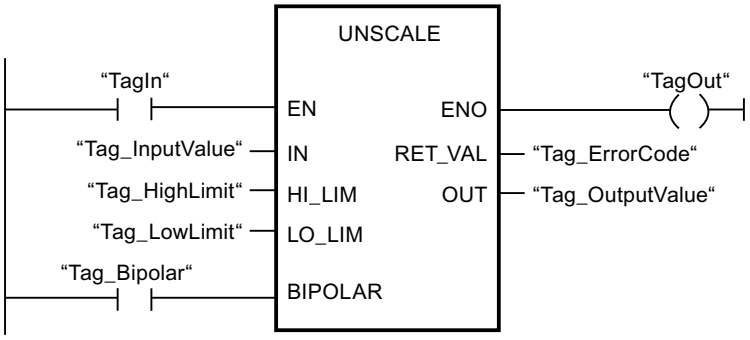
RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
0008	The value of the IN parameter is greater than the value of the high limit (HI_LIM) or less than the value of the low limit (LO_LIM).
General error information	See also: "GET_ERR_ID: Get error ID locally"
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats here: AUTOHOTSPOT	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN	Tag_InputValue	50.03978588
HI_LIM	Tag_HighLimit	100.0
LO_LIM	Tag_LowLimit	0.0
BIPOLAR	Tag_Bipolar	1
OUT	Tag_OutputValue	22
RET_VAL	Tag_ErrorCode	W#16#0000

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 2840)

Basic information on LAD (Page 4878)

Program control operations (S7-1200, S7-1500)

---(JMP): Jump if RLO = 1 (S7-1200, S7-1500)

Description

You can use the "Jump if RLO = 1" instruction to interrupt the linear execution of the program and resume it in another network. The destination network must be identified by a jump label (LABEL). The name of this jump label is specified in the placeholder above the instruction.

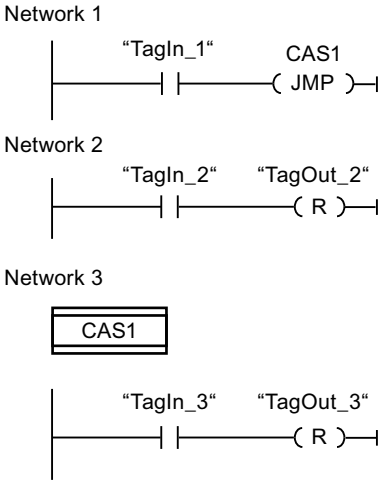
The specified jump label must be in the same block in which the instruction is executed. The name you specify can only occur once in the block. Only one jumping coil is permitted within a network.

If the result of logic operation (RLO) at the input of the instruction is "1", the jump to the network identified by the specified jump label is executed. The jump direction can be towards higher or lower network numbers.

If the condition at the input of the instruction is not fulfilled (RLO = 0), execution of the program continues in the next network.

Example

The following example shows how the instruction works:



If operand "TagIn_1" has the signal state "1", the "Jump if RLO = 1" instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. If the "TagIn_3" input has signal state "1", the "TagOut_3" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

---(JMPN): Jump if RLO = 0 (S7-1200, S7-1500)

Description

You can use the instruction "Jump if RLO = 0" to interrupt the linear execution of the program and resume it in another network, when the result of logic operation at the input of the instruction is "0". The destination network must be identified by a jump label (LABEL). The name of this jump label is specified in the placeholder above the instruction.

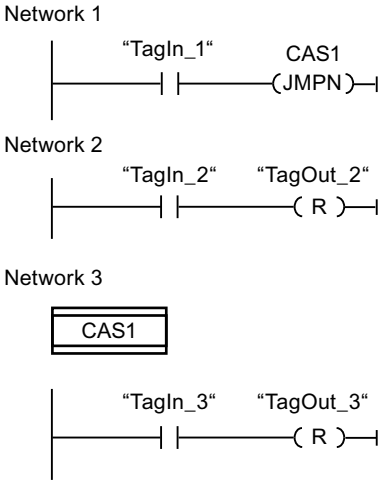
The specified jump label must be in the same block in which the instruction is executed. The name you specify can only occur once in the block. Only one jumping coil is permitted within a network.

If the result of logic operation (RLO) at the input of the instruction is "0", the jump to the network identified by the specified jump label is executed. The jump direction can be towards higher or lower network numbers.

If the result of logic operation at the input of the instruction is "1", execution of the program continues in the next network.

Example

The following example shows how the instruction works:



If operand "TagIn_1" has the signal state "0", the "Jump if RLO = 0" instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. If the "TagIn_3" input has signal state "1", the "TagOut_3" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

LABEL: Jump label (S7-1200, S7-1500)

Description

You can use a jump label to identify a destination network, in which the program execution should resume when a jump is executed.

The jump label and the instruction in which the jump label is specified must be located in the same block. The name of a jump label can only be assigned once in a block. You can declare up to 32 jump labels when you use a CPU S7-1200 and a maximum of 256 jump labels when you use a CPU S7-1500.

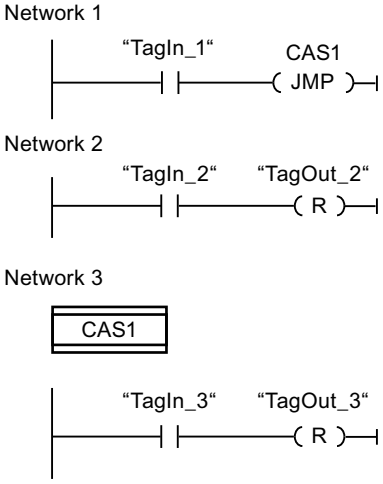
Only one jump label can be placed in a network. Each jump label can jump to several locations.

Adhere to the following grammatical rules for jump labels:

- Letters (a to z, A to Z)
- A combination of letters and numbers. Note the order, i.e. first the letters and then the letters (a - z, A - Z, 0 - 9).
- Special characters or a combination of letters and numbers in reverse order, i.e. first the numbers and then the letters (0 - 9, a - z, A - Z) must not be used.

Example

The following example shows how the instruction works:



If operand "TagIn_1" has the signal state "1", the "Jump if RLO = 1" instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. If the "TagIn_3" input has signal state "1", the "TagOut_3" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

JMP_LIST: Define jump list (S7-1200, S7-1500)

Description

You can use the "Define jump list" instruction to define several conditional jumps and continue the program execution in a specific network depending on the value of the K parameter.

You define the jumps with jump labels (LABEL), which you specify at the outputs of the instruction box. The number of outputs can be expanded in the instruction box. You can declare up to 32 outputs when you use a CPU S7-1200 and a maximum of 256 outputs when you use a CPU S7-1500.

The numbering of the outputs begins with the value "0" and is continued in ascending order with each new output. Only jump labels can be specified at the outputs of the instruction. Instructions or operands cannot be specified.

The value of the K parameter specifies the number of the output and thus the jump label where the program execution is to be resumed. If the value in the K parameter is greater than the number of available outputs, the program execution is resumed in the next network of the block.

The "Define jump list" instruction is only executed if the signal state is "1" at the EN enable input.

Parameters

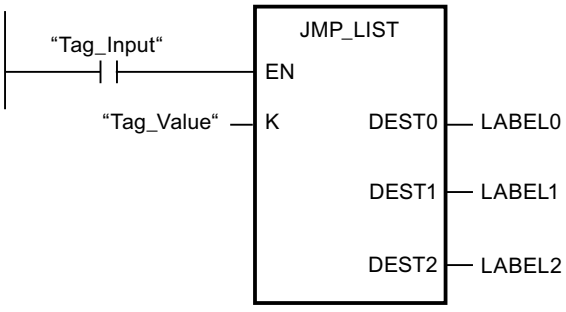
The following table shows the parameters of the "Define jump list" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
K	Input	UINT	I, Q, M, D, L or constant	Specifies the number of the output and thus the jump that is to be made.
DEST0	-	-	-	First jump label
DEST1	-	-	-	Second jump label
DESTn	-	-	-	Optional jump labels

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand/jump label	Value
K	Tag_Value	1
Dest0	LABEL0	Jump to the network that is identified with the jump label "LABEL0".
Dest1	LABEL1	Jump to the network that is identified with the jump label "LABEL1".
Dest2	LABEL2	Jump to the network that is identified with the jump label "LABEL2".

If operand "Tag_Input" has the signal state "1", the "Define jump list" instruction is executed. The execution of the program is continued according to the value of the operand "Tag_Value" in the network that is identified with the jump label "LABEL1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

SWITCH: Jump distributor (S7-1200, S7-1500)

Description

You can use the "Jump distributor" instruction to define multiple program jumps to be executed depending on the result of one or more comparison instructions.

You specify the value to be compared in the K parameter. This value is compared with the values that are provided by the various inputs. You can select the comparison method for each individual input. The availability of the various comparison instructions depends on the data type of the instruction.

The following table shows the comparison instructions that are available depending on the selected data type:

Data type		Instruction	Syntax
S7-1200	S7-1500		
Bit strings	Bit strings	Equal	==
		Not equal	<>

Data type		Instruction	Syntax
S7-1200	S7-1500		
Integers, floating-point numbers, TIME, DATE, TOD	Integers, floating-point numbers, TIME, LTIME, DATE, TOD, LTOD, LDT	Equal	==
		Not equal	<>
		Greater or equal	>=
		Less or equal	<=
		Greater than	>
		Less than	<

You can select the data type of the instruction from the "???" drop-down list of the instruction box. If you select a comparison instruction and the data type of the instruction is not yet defined, then the "???" drop-down list will only list those data types that are permitted for the selected comparison instruction.

Execution of the instruction begins with the first comparison and runs until a comparison condition is met. If a comparison condition is met, the subsequent comparison conditions are not considered. If none of the specified comparison conditions are fulfilled, the jump is executed at output ELSE. If no program jump is defined at the ELSE output, execution of the program continues in the next network.

The number of outputs can be expanded in the instruction box. The numbering of the outputs begins with the value "0" and is continued in ascending order with each new output. Specify jump labels (LABEL) at the outputs of the instruction. Instructions or operands cannot be specified at the outputs of the instruction.

An input is automatically inserted to each additional output. The jump programmed at an output is executed if the comparison condition of the corresponding input is fulfilled.

Parameters

The following table shows the parameters of the instruction "Jump distributor":

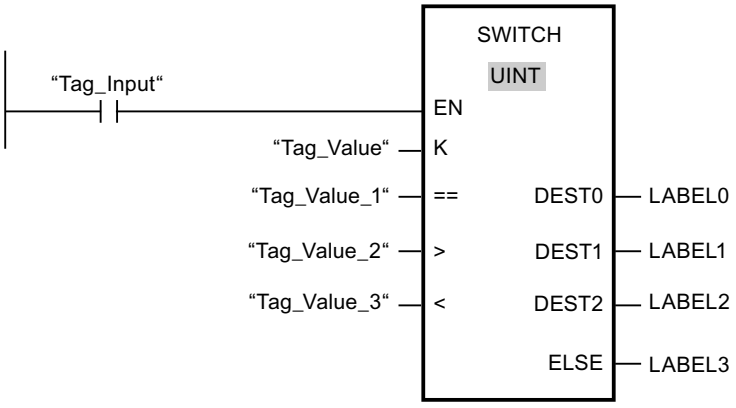
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
K	Input	UINT	UINT	I, Q, M, D, L or constant	Specifies the value to be compared.
<Comparison values>	Input	Bit strings, integers, floating-point numbers, TIME, DATE, TOD	Bit strings, integers, floating-point numbers, TIME, LTIME, DATE, TOD, LTOD, LDT	I, Q, M, D, L or constant	Input values with which the value of the parameter K is compared.
DEST0	-	-	-	-	First jump label
DEST1	-	-	-	-	Second jump label

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
DEST(n)	-	-	-	-	Optional jump labels: • S7-1200: n = 2 to 32 • S7-1500: n = 2 to 256
ELSE	-	-	-	-	Program jump which is executed if none of the comparison conditions are fulfilled.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand/jump label	Value
K	Tag_Value	23
==	Tag_Value_1	20
>	Tag_Value_2	21
<	Tag_Value_3	19
DEST0	LABEL0	Jump to jump label "LABEL0", if the value of parameter K is equal to 20.
DEST1	LABEL1	Jump to jump label "LABEL1", if the value of parameter K is greater than 21.
DEST2	LABEL2	Jump to jump label "LABEL2", if the value of parameter K is less than 19.
ELSE	LABEL3	Jump to jump label "LABEL3", if the none of the comparison conditions are fulfilled.

If the operand "Tag_Input" changes to signal state "1", the instruction "Jump distributor" is executed. The execution of the program is continued in the network that is identified with the jump label "LABEL1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

--(RET): Return (S7-1200, S7-1500)

Description

You can use the "Return" instruction to stop the execution of a program block conditionally or absolutely. The signal state of the return value (operand) is mapped in the enable output ENO of the calling program block when the program block is exited.

There are three ways that the execution of a program block can be ended:

Termination of the program block	Description
Without a call of the instruction	The program block is exited after the execution of the last network. The enable output ENO of the calling program block is set to signal state "1".
Call of the instruction with preceding logic operation (see example)	If the preceding logic operation is fulfilled, the program execution in the currently called program block is ended and the program block is exited. (Conditional program block end) The program execution is continued in the calling program block (e.g. an OB) after the program block call. The enable output ENO of the calling program block corresponds to the operand.
Call of the instruction without preceding logic or the instruction is directly connected to the left power rail.	The program block is exited unconditionally. (Absolute program block end) The enable output ENO of the calling program block corresponds to the operand.

If an organization block (OB) is ended, another program block is selected by the execution level system and started or further executed.

- If the OB program cycle is ended, it is restarted.
- If an OB is ended that has interrupted another program block (e.g. an interrupt OB), then the interrupted program block (e.g. program cycle OB) is further executed.

Note

RET instruction in relation with JMP and JMPN

If a network already contains the "JMP: Jump if RLO = 1" or "JMPN: Jump if RLO = 0 instruction, the RET: Return" instruction must not be used. Only one jumping coil may be used in a network

Possible settings of the return value (operand)

The return value of the instruction can assume the following values:

- Ret (RLO, corresponds to the result of logic operation RLO. The signal state "1" is supplied to the enable output ENO of the calling program block because the RET instruction is only then executed as a conditional instruction when the condition is TRUE.)
- Ret True or Ret False (the respective value of the constant, TRUE or FALSE, is supplied to the calling program block.)
- Ret Value (the value of the Boolean tag <Operand> is supplied to the calling program block.)

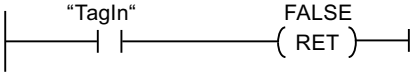
To set the return value of the instruction, click the small yellow triangle of the instruction and select the desired value in the drop-down list.

The following table shows the status of the calling program block when the instruction is programmed in a network within the called program block:

RLO	Return value	ENO of the calling program block
1	RLO	1
	TRUE	1
	FALSE	0
	<Operand> Boolean tag with possible memory areas I, Q, M, D, L, T, and C	<Operand>
0	RLO	The program execution is continued in the next network of the called program block.
	TRUE	
	FALSE	
	<Operand>	

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. Program execution is ended in the called program block and continues in the calling program block. The enable output ENO of the calling program block is reset to signal state "0".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Runtime control (S7-1200, S7-1500)

ENDIS_PW: Limit and enable password legitimation (S7-1200, S7-1500)

Description

You use the "Limit and enable password legitimation" instruction to specify whether configured passwords may be legitimated or not for the CPU. You can prevent legitimated connections, even when the correct password is known.

When you call the command and the REQ parameter has the signal state "0", only the currently set state is displayed at the output parameters. If changes are made to the input parameters, these changes are not transferred to the output parameters.

When the instruction is called and the REQ parameter has the signal state "1", the signal state is taken from the input parameters (F_PWD, FULL_PWD, R_PWD, HMI_PWD).

- If signal state "0" is present, authorization via password is not allowed.
- If signal state "1" is present, the password can be used.

Disable or enabling of the passwords can be allowed or prohibited individually. For example, all passwords can be prohibited, except the fail-safe password. You can thus limit access options to a small user group. The output parameters (F_PWD_ON, FULL_PWD_ON, R_PWD_ON, HMI_PWD_ON) always show the current status of the password use, regardless of the REQ parameter.

Non-configured passwords must have the TRUE signal state at the input and return the TRUE signal state at the output. The fail-safe password can only be configured for an F-CPU and, therefore, must always be interconnected with theTRUE signal state in a standard CPU. If the instruction return an error, the call remains without effect, i.e. the previous lock remains in force.

Disabled passwords can be enabled again under the following conditions:

- The CPU is reset to its factory settings.
- The front panel of the S7-1500 CPU supports a dialog that allows you to navigate to the appropriate menu in which the passwords can be enabled again.
- When you call the "Limit and enable password legitimation" instruction, the input parameter of the desired password has the signal state "1".
- Set the operating mode switch to STOP. The restriction to password legitimation is re-established as soon as the switch is set back to RUN.
- Inserting an empty memory card (transfer card or program card) into S7-1200 CPU.
- The transition from POWER OFF to POWER ON disables the protection in the S7-1200 CPU. The "Limit and enable password legitimation" instruction must be called again in the program (e.g. in the startup OB).

Note

The "Limit and enable password legitimation" instruction blocks access to the HMI system, if the HMI password has not been enabled.

Note

Existing legitimized connections retain their access rights and you cannot use the "Limit and enable password legitimation" instruction to restrict them.

Preventing accidental lockout out with an S7-1500 CPU

The same setting can be made on the front panel of the S7-1500 CPU and the CPU saves the most recent setting.

To prevent accidental lockout, the protection can be disabled by setting the mode selector to STOP with an S7-1500 CPU. The protection is automatically enabled again by setting mode selector to RUN without having to call the "Limit and enable password legitimation" instruction again or taking additional actions on the front panel.

Preventing accidental lockout with an S7-1200 CPU

The protection for POWER OFF - POWER ON is disabled because a S7-1200 CPU has no operating mode switch. As a result it is possible and advisable to use certain program sequences within your program to prevent an accidental lockout.

To do this, program a time control using either a cyclic interrupt OB or a timer in the main OB (OB 1). This provides you with the option to call the "Limit and enable password legitimation" instruction again without delay in the corresponding OB (e.g. OB 1 or OB 35) after a transition from POWER OFF - POWER ON. Call the instruction in the startup OB (OB 100) in order to keep as short as possible the time window in which the application is not active and thus ensure that no restrictions exist in the password legitimation. This procedure offers you the largest possible protection against unauthorized access.

If an unintended lockout occurs you can skip the call in the startup OB (for example, by querying an input parameter) and have the set time (for example 10 seconds to 1 minute), to establish a connection to the CPU before the lock becomes active again.

If you have set no timer in your program code and are locked out, insert an empty transfer card or empty program card into the CPU. The empty transfer card or program card deletes the internal load memory of the CPU. You must then load the user program again from STEP 7 in the CPU.

Procedure for lost passwords with an S7-1200 CPU

If you have lost the password for a password-protected S7-1200 CPU, delete the password-protected program with an empty transfer card or program card. The empty transfer card or program card deletes the internal load memory of the CPU. You can then load a new user program from STEP 7 Basic in the CPU.

WARNING

Inserting empty transfer card

When you insert a transfer card in a CPU during runtime, the CPU goes to STOP mode. If operating states are unstable, controllers may fail and thus cause the uncontrolled operation of the controlled devices. This leads to an unforeseen operation of the automation system which can cause deadly or serious injuries and/or damage to property.

After you remove the transfer card the content of the transfer card is in the internal load memory. Make sure at this point that the card contains no program.

WARNING

Inserting empty program card

When you insert a program card in a CPU during runtime, the CPU goes to STOP mode. If operating states are unstable, controllers may fail and thus cause the uncontrolled operation of the controlled devices. This leads to an unforeseen operation of the automation system which can cause deadly or serious injuries and/or damage to property.

Make sure that the program card is empty. The internal load memory is copied to the empty program card. The internal load memory is empty after the removal of the formerly empty program card.

You must remove the transfer card or program card before you put the CPU into RUN mode.

Effects of password use on the operating modes

The following table shows what effects the password use by means of the "Limit and enable password legitimation" instruction has on the operating modes and the corresponding user actions.

Action	Password protection by means of the instruction
Basic state after • Operating mode switch to STOP • Reset memory manually (PG, switch, change of MC (Motion Control)) • Reset to factory setting	Not activated (no restrictions)
Basic state after POWER ON	• S7-1200-CPU: the lock is disabled and the instruction must be called again in the program (in the startup OB, for example). • S7-1500 CPU: Enabled (if a lock was activated before POWER OFF). The option of not allowing passwords is retentive.
Operating mode transition RUN/STARTUP/HOLD -> STOP (by terminating the instruction, an error or communication) or STOP -> STARTUP/RUN/HOLD	Activated Passwords still cannot be used.

Parameters

The following table shows the parameters of the "Limit and enable password legitimation" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
REQ	Input	BOOL	I, Q, M, D, L or constant	When the REQ parameter has the signal state "0", the currently set signal state of the passwords is queried.
F_PWD	Input	BOOL	I, Q, M, D, L or constant	Read/write access including fail-safe • F_PWD = "0": Do not allow password • F_PWD = "1": Allow password
FULL_PWD	Input	BOOL	I, Q, M, D, L or constant	Read/write access • FULL_PWD = "0": Do not allow password • FULL_PWD = "1": Allow password

Parameters	Declaration	Data type	Memory area	Description
R_PWD	Input	BOOL	I, Q, M, D, L or constant	Read access • R_PWD = "0": Do not allow password • R_PWD = "1": Allow password
HMI_PWD	Input	BOOL	I, Q, M, D, L or constant	HMI access • HMI_PWD = "0": Do not allow password • HMI_PWD = "1": Allow password
F_PWD_ON	Output	BOOL	I, Q, M, D, L	Read/write access including fail-safe • F_PWD_ON = "0": Password not allowed • F_PWD_ON = "1": Password allowed
FULL_PWD_ON	Output	BOOL	I, Q, M, D, L	Read/write access status • FULL_PWD_ON = "0": Password not allowed • FULL_PWD_ON = "1": Password allowed
R_PWD_ON	Output	BOOL	I, Q, M, D, L	Read-access status • R_PWD_ON = "0": Password not allowed • R_PWD_ON = "1": Password allowed
HMI_PWD_ON	Output	BOOL	I, Q, M, D, L	HMI-access status • HMI_PWD_ON = "0": Password not allowed • HMI_PWD_ON = "1": Password allowed
RET_VAL	Output	WORD	I, Q, M, D, L	Error information

You can find additional information on valid data types under "See also".

Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8090	The "Limit and enable password legitimation" instruction is not supported
80D0	The password for fail-safe is not configured. With standard CPUs, the signal state must be TRUE.
80D1	The read/write access is not configured

Error code* (W#16#...)	Explanation
80D2	The read access is not configured
80D3	The HMI access is not configured
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

PC systems (S7-1200, S7-1500)

SHUT_DOWN: Shut down target system (S7-1200, S7-1500)

Description

You can use the instruction "SHUT_DOWN: Shutdown target system" to shut down the PC-based automation system and restart the S7 software controller CPU 150xS or Windows on the PC-based automation system.

The instruction is available in the "Instructions" task card under Basic instructions > Program control > Runtime control.

A restart is, for example, advisable in the following situations:

- An industrial UPS (uninterruptible power supply) reports a power failure via a digital input.
- Windows stops responding or shows a "blue screen".
- Too many error OBs are called in the user program.

Parameter

The following table shows the parameters of the "Shutdown target system" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	UINT	I, Q, M, D, L or constant	MODE = 1: Shut down CPU 150xS and Windows The CPU goes to STOP and saves the retentive data. Then the CPU and Windows are shut down. The system must be restarted manually. MODE = 2: Restart CPU 150xS The CPU goes to STOP and saves the retentive data. Then the CPU is shut down and restarted. MODE = 3: Restart Windows The CPU remains in RUN. Windows is restarted (as of TIA Portal V14, MODE 3 is only retained for backward compatibility. We recommend using MODE 4 or MODE 5). MODE = 4: Windows is shut down and restarted correctly. The CPU remains in RUN. MODE = 5: Restart Windows (Comparable with MODE 3; Exception: MODE 5 should only be used if Windows has crashed).
COM-MENT	Input	STRING	I, Q, M, D, L	With Mode = 1, 3 and 4, you can specify a reason for the restart. The reason is output in the Windows Event Log.
Ret_Val	Return	WORD	Q, M, D, L	Ret_Val = 0: No faults Ret_Val = 8090: The value passed to MODE is not supported. Ret_Val = 8091: Windows does not respond to the call of the Shut_Down instruction (only valid for Mode 3 and 4). Ret_Val = 8092: Please contact SIMATIC Customer Support if this error occurs (only valid for Mode 3 and 4).

RE_TRIGR: Restart cycle monitoring time (S7-1200, S7-1500)

Description

The "Restart cycle monitoring time" instruction is used to restart the cycle monitoring time of the CPU. The cycle monitoring time then restarts with the time you have set in the CPU configuration.

If the instruction is called in a block with a higher priority, such as a hardware interrupt or diagnostic error interrupt, the instruction is not executed and the enable output ENO returns signal state "0".

The instruction executes completely within a time span (10 times the maximum program cycle), regardless of the number of calls. Once this time has expired, the program cycle can no longer be prolonged.

Calling the instruction

The following call conditions apply:

- The following applies to a CPU of the S7-1200 series:
  In firmware versions <= 2.2, you can call the instruction only in a program cycle organization block of priority 1. This corresponds to the lowest priority of all organization blocks. If the instruction is called in an organization block with a higher priority, the instruction is not executed and the result (BR bit, enable output ENO) is always "0".
  In firmware versions >= 2.2, you can call the instruction regardless of the priority in all organization blocks.
- The following applies to a CPU of the S7-1500 series:
  You can call the instruction regardless of the priority in all organization blocks.

Parameters

The instruction "Restart cycle monitoring time" has no parameters.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

STP: Exit program (S7-1200, S7-1500)

Description

You use the "Exit program" instruction to set the CPU to STOP mode and thus end the program execution. The effects at a transition from RUN to STOP depend on the CPU configuration.

When the result of logic operation (RLO) at the input of the instruction is "1", the CPU changes to STOP mode and the program execution is ended. The signal state at the output of the instruction is not evaluated.

When the RLO is "0" at the input of the instruction, then the instruction will not be executed.

Parameters

The "Exit program" instruction has no parameters.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

GET_ERROR: Get error locally (S7-1200, S7-1500)

Description

The "Get error locally" instruction is used to query the occurrence of errors within a program block. This normally involves programming or access errors. If the system reports an error during the execution of the program block, detailed information is stored in the operand at the ERROR output for the first error to occur during the execution of the block since the last execution of the instruction.

Only operands of the "ErrorStruct" system data type can be specified at the ERROR output. The "ErrorStruct" system data type specifies the exact structure in which the information about the error is stored. Using additional instructions, you can evaluate this structure and program an appropriate response. If several errors occur in the program block, the error information about the next error to occur is output by the instruction only after the first error that occurred has been remedied.

Note

Output ERROR

The ERROR output is only changed if error information is present. To set the output back to "0" after handling an error, you have the following options:

- Declare the tag in the "Temp" section of the block interface.
- Reset the tag to "0" before calling the instruction.
- Query the enable output ENO.

The enable output ENO is only set if the enable input EN returns signal state "1" and error information is present. If one of these conditions does not apply, then the remaining program execution is not affected by the instruction.

Note

Activation of the local error handling

As soon as you insert the instruction in the program code of a program block, the local error handling is activated and default system reactions are ignored when errors occur.

Options for error handling

You can find an overview of the options for error handling here: AUTOHOTSPOT

For a detailed example of local error handling that contains several error handling options, refer to: AUTOHOTSPOT

Parameters

The following table shows the parameters of the "Get error locally" instruction:

Parameters	Declaration	Data type	Memory area	Description
ERROR	Output	ErrorStruct	D, L	Error information

Data type "ErrorStruct"

The "ErrorStruct" data type can be inserted in a global data block or in the block interface. You can insert the data type more then once if you assign a different name for the data structure each time. The data structure and the name of individual structure elements cannot be changed. If you save the error information in a global data block, it can also be read out by other program blocks.

The following table shows the structure of the "ErrorStruct" data type:

Structure component		Data type	Description
ERROR_ID		WORD	Error ID
FLAGS		BYTE	Shows whether the error occurred during a program block call. 16#01: Error during a program block call 16#00: No error during a program block call
REACTION		BYTE	Default reaction: 0: Ignore (write error) 1: Continue with substitute value "0" (read error) 2: Skip instruction (system error)
CODE_ADDRESS		CREF	Information about the address and type of the program block
	BLOCK_TYPE	BYTE	Program block type in which the error occurred: 1: Organization block (OB) 2: Function (FC) 3: Function block (FB)
	CB_NUMBER	UINT	Number of the code block
	OFFSET	UDINT	Reference to the internal memory
MODE		BYTE	Information about the address of an operand
OPERAND_NUMBER		UINT	Operand number of the machine command
POINTER_NUMBER_LOCATION		UINT	(A) Internal pointer
SLOT_NUMBER_SCOPE		UINT	(B) Storage area in internal memory
DATA_ADDRESS		NREF	Information about the address of an operand
	AREA	BYTE	(C) Memory area: L: 16#40...16#7F, 16#86, 16#87, 16#8E, 16#8F, 16#C0...16#FF I: 16#81 Q: 16#82 M: 16#83 DB: 16#40, 16#84, 16#85, 16#8A, 16#8B PI: 16#01 PQ: 16#02 Technological objects: 16#04

Structure component		Data type	Description
	DB_NUMBER	UINT	(D) Number of the data block
	OFFSET	UDINT	(E) Relative address of the operand

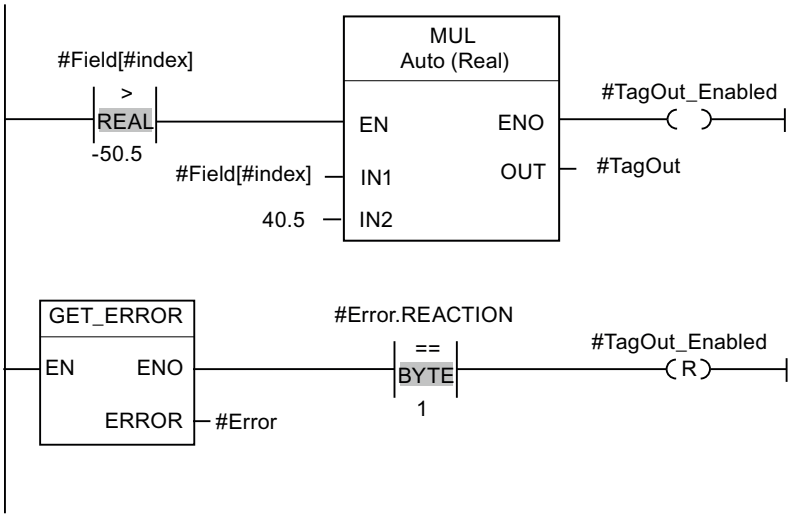
Structure component "ERROR_ID"

The following table shows the values that can be output at the structure component "ERROR_ID":

ID* (hexadecimal)	ID* (decimal)	Description
0	0	No error
2503	9475	Invalid pointer
2520	9504	Invalid STRING
2522	9506	Read error: Operand outside the valid range
2523	9507	Write error: Operand outside the valid range
2524	9508	Read error: Invalid operand
2525	9509	Write error: Invalid operand
2528	9512	Read error: Data alignment
2529	9513	Write error: Data alignment
252C	9516	Invalid pointer
2530	9520	Write error: Data block
2533	9523	Invalid reference used
2538	9528	Access error: DB does not exist
2539	9529	Access error: Wrong DB used
253A	9530	Global data block does not exist
253C	9532	Faulty information or the function does not exist
253D	9533	System function does not exist
253E	9534	Faulty information or the function block does not exist
253F	9535	System block does not exist
2550	9552	Access error: DB does not exist
2551	9553	Access error: Wrong DB used
2575	9589	Error in the program nesting depth
2577	9591	The block property "Parameter passing via registers" is not selected.
2576	9590	Error in the local data distribution
25A0	9632	Internal error in TP
25A1	9633	Tag is write-protected
25A2	9634	Invalid numerical value of tag
2942	10562	Read error: Input
2943	10563	Write error: Output
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".		

Example

The following example shows how the instruction works:



An error occurred accessing the #Field[#index] tag. The enable output ENO of the "Multiply" instruction and the #TagOut_Enabled operand have signal state "1" despite the read/access error and the multiplication is performed with the value "0.0". If this error scenario occurs, it is recommended to program the "Get error locally" instruction after the "Multiply" instruction to get the error. The error information supplied by the "Get error locally" instruction is evaluated with the comparison instruction "Equal". If the "#Error.REACTION" structure component has the value "1", this involves a read/access error and the #TagOut_Enabled output is reset.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

GET_ERR_ID: Get error ID locally (S7-1200, S7-1500)

Description

The "Get error ID locally" instruction is used to query the occurrence of errors within a block. This is usually to access error. If the system reports during the block processing errors within the block execution since the last execution of the instruction, the error ID of the first error that occurred in the tag is stored at the ID output.

Only operands of the WORD data type can be specified at the ID output. If several errors occur in the block, the error ID of the next error to occur in the instruction is output only after the first error that occurred has been remedied.

Note

The ID output is only changed if error information is present. To set the output back to "0" after handling an error, you have the following options:

- Declare the tag in the "Temp" section of the block interface.
- Reset the tag to "0" before calling the instruction.
- Query the enable output ENO.

The enable output ENO of the instruction "Get error ID locally" instruction is set only if the enable input EN returns signal state "1" and error information is present. If one of these conditions is not fulfilled, the remaining program execution is not affected by the "Get error ID locally" instruction.

You can find an example of how you can implement the instruction in combination with other troubleshooting options under "See also".

Note

The "Get error ID locally" instruction enables local error handling within a block. If the "Get error ID locally" instruction is inserted in the program code of a block, any predefined system reactions are ignored if errors occur.

Options for error handling

You can find an overview of the options for error handling here: AUTOHOTSPOT

For a detailed example of local error handling that contains several error handling options, refer to: AUTOHOTSPOT

Parameters

The following table shows the parameters of the "Get error ID locally" instruction:

Parameters	Declaration	Data type	Memory area	Description
ID	Output	WORD	I, Q, M, D, L	Error ID

Parameter ID

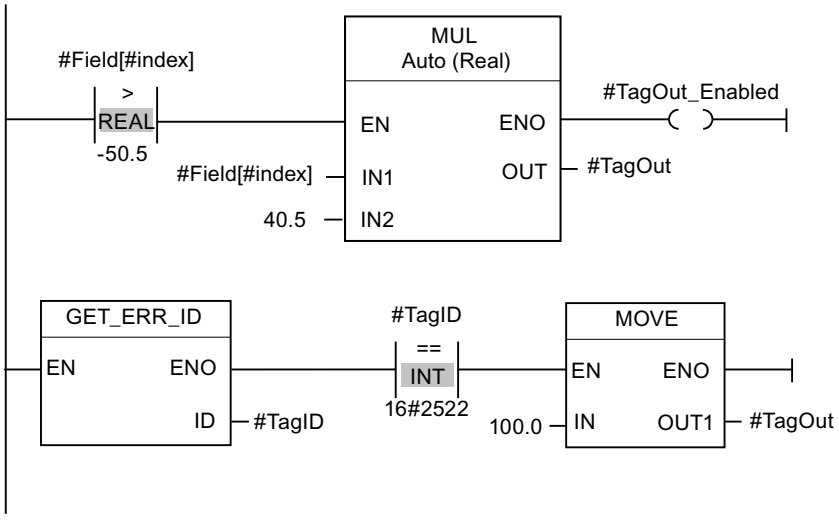
The following table shows the values that can be output at the ID parameter:

ID* (hexadecimal)	ID* (decimal)	Description
0	0	No error
2503	9475	Invalid pointer
2520	9504	Invalid STRING

ID* (hexadecimal)	ID* (decimal)	Description
2522	9506	Read error: Operand outside the valid range
2523	9507	Write error: Operand outside the valid range
2524	9508	Read error: Invalid operand
2525	9509	Write error: Invalid operand
2528	9512	Read error: Data alignment
2529	9513	Write error: Data alignment
252C	9516	Invalid pointer
2530	9520	Write error: Data block
2533	9523	Invalid reference used
2538	9528	Access error: DB does not exist
2539	9529	Access error: Wrong DB used
253A	9530	Global data block does not exist
253C	9532	Faulty information or the function does not exist
253D	9533	System function does not exist
253E	9534	Faulty information or the function block does not exist
253F	9535	System block does not exist
2550	9552	Access error: DB does not exist
2551	9553	Access error: Wrong DB used
2575	9589	Error in the program nesting depth
2576	9590	Error in the local data distribution
2577	9591	The block property "Parameter passing via registers" is not selected.
25A0	9632	Internal error in TP
25A1	9633	Tag is write-protected
25A2	9634	Invalid numerical value of tag
2942	10562	Read error: Input
2943	10563	Write error: Output
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".		

Example

The following example shows how the instruction works:



An error occurred accessing the #Field[#index] tag. The enable output ENO of the "Multiply" instruction and the #TagOut_Enabled operand have signal state "1" despite the read/access error and the multiplication is performed with the value "0.0". If this error scenario occurs, it is recommended to program the "Get error ID locally" instruction after the "Multiply" instruction to get the error. The error information supplied by the "Get error ID locally" instruction is evaluated with the comparison instruction "Equal". If the #TagID operand returns the ID 2522, this involves a read/access error and the value "100.0" is written to the #TagOut output.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

INIT_RD: Initialize all retain data (S7-1500)

Description

You can use the "Initialize all retain data" instruction to reset the retain data of all data blocks, bit memories and SIMATIC timers and counters at the same time. The instruction can only be executed within a startup OB because the execution would exceed the program cycle duration.

Parameters

The following table shows the parameters of the "Initialize all retain data" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
REQ	Input	BOOL	I, Q, M, D, L, T, C or constant	If the input "REQ" has the signal state "1", all retain data are reset.
RET_VAL	Output	INT	I, Q, M, D, L	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.

You can find additional information on valid data types under "See also".

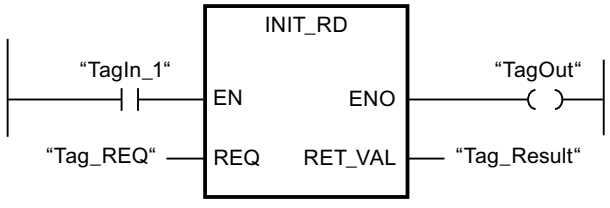
Parameters RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B5	The instruction cannot be executed because it was not programmed within a startup OB.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



If the operands "TagIn_1" and "Tag_REQ" have signal state "1", the instruction is executed. The retain data of all data blocks, bit memories and SIMATIC timers and counters are reset If the instruction is executed without errors, the ENO enable output has the signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 2840)

Basic information on LAD (Page 4878)

WAIT: Configure time delay (S7-1500)

Description

The "Configure time delay" instruction pauses the program execution for a specific period of time. You indicate the period of time in microseconds on the WT parameter of the instruction.

You can configure time delays from -32768 up to +32767 microseconds (µs). The smallest possible delay time depends on the CPU and corresponds to the execution time of the instruction.

The execution of the instruction can be interrupted by higher priority events and does not return any error information.

Note

Negative delay time

If you specify a negative delay time at parameter WT, the enable output ENO, or the RLO and the BR bit, will return the signal state FALSE. A negative delay time does not affect the CPU. The following instructions linked to enable output ENO are not executed in LAD or FBD.

Parameters

The following table shows the parameters of the "Configure time delay" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
WT	Input	INT	I, Q, M, D, L, P, or constant	Time delay in microseconds (µs)

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

RUNTIME: Measure program runtime (S7-1200, S7-1500)

Description

The "Measure program runtime" instruction is used to measure the runtime of the entire program, individual blocks or command sequences.

If you want to measure the runtime of your entire program, call the instruction "Measure program runtime" in OB1. Measurement of the runtime is started with the first call and the output RET_VAL returns the runtime of the program after the second call. The measured runtime includes all CPU processes that can occur during the program execution, for example, interruptions caused by higher-level events or communication. The instruction "Measure program runtime" reads an internal counter of the CPU and write the value to the IN-OUT parameter MEM. The instruction calculates the current program runtime according to the internal counter frequency and writes it to output RET_VAL.

If you want to measure the runtime of individual blocks or individual command sequences, you need three separate networks. Call the instruction "Measure program runtime" in an individual network within your program. You set the starting point of the runtime measurement with this first call of the instruction. Then you call the required program block or the command sequence in the next network. In another network, call the "Measure program runtime" instruction a second time and assign the same memory to the IN-OUT parameter MEM as you did during the first call of the instruction. The "Measure program runtime" instruction in the third network reads an internal CPU counter and calculates the current runtime of the program block or the command sequence according to the internal counter frequency and writes it to the output RET_VAL.

The "Measure program runtime" instruction uses an internal high-frequency counter to calculate the time. If the counter overruns, the instruction returns values <= 0.0. This can occur up to once per minute for S7-1200 CPUs with firmware version <V4.2. Ignore these runtime values.

Note

The runtime of a command sequence cannot be determined exactly, because the sequence of instructions within a command sequence is changed during optimized compilation of the program.

Parameter

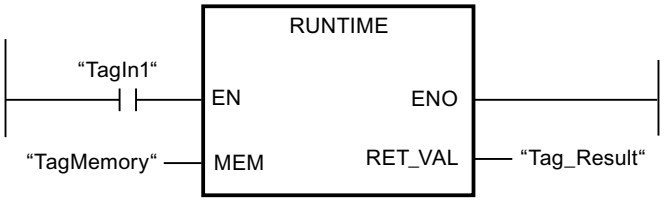
The following table lists the parameters of the "Measure program runtime" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
MEM	InOut	LREAL	I, Q, M, D, L	The content is for internal purposes only.
RET_VAL	Output	LREAL	I, Q, M, D, L	Returns the measured runtime in seconds

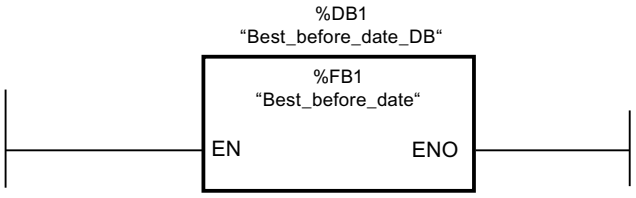
Example

The following example shows how the instruction works based on the runtime calculation of a program block:

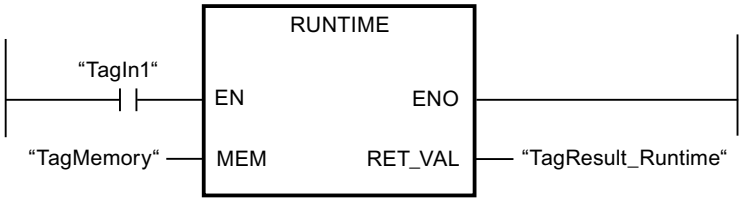
Network 1:



Network 2:



Network 3:



When the "TagIn1" operand in network 1 has the signal state "1", the instruction is executed. The starting point for the runtime measurement is set with the first call of the instruction and buffered as reference for the second call of the instruction in the "TagMemory" operand.

The "Best_before_date" program block FB1 is called in network 2.

When the FB1 program block has been executed and the "TagIn1" operand has the signal state "1", the instruction in network 3 is executed. The second call of the instruction calculates the runtime of the program block and writes the result to the output RET_VAL.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

You can find a more detailed example of how you can measure the total cycle time of a program in the Siemens Industry Online Support under the following entry ID: 87668055 (https://support.industry.siemens.com/cs/ww/en/view/87668055)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Word logic operations (S7-1200, S7-1500)

AND: AND logic operation (S7-1200, S7-1500)

Description

You can use the "AND logic operation" instruction to combine the value at the IN1 input and the value at the IN2 input bit-by-bit by AND logic and query the result at the OUT output.

When the instruction is executed, bit 0 of the value at the IN1 input and bit 0 of the value at the IN2 input are logically ANDed. The result is stored in bit 0 of output OUT. The same logic operation is executed for all other bits of the specified values.

The number of inputs can be expanded in the instruction box. The added inputs are numbered in ascending order in the box. When the instruction is executed, the values of all available input parameters are combined with AND logic (ANDed). The result is stored in the OUT output.

The result bit has signal state "1" only when both of the bits in the logic operation also have signal state "1". If one of the two bits of the logic operation has signal state "0", the corresponding result bit is reset.

Parameters

The following table shows the parameters of the "AND logic operation" instruction:

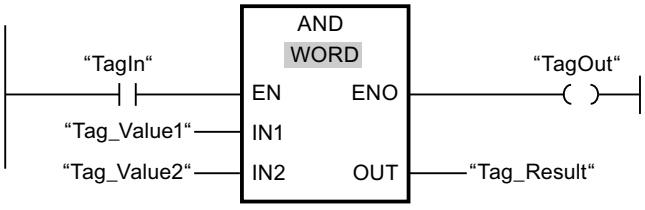
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Bit strings	I, Q, M, D, L, P, or constant	First value for logic operation
IN2	Input	Bit strings	I, Q, M, D, L, P, or constant	Second value for logic operation
INn	Input	Bit strings	I, Q, M, D, L, P, or constant	Other inputs whose values are logically combined.
OUT	Output	Bit strings	I, Q, M, D, L, P	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN1	Tag_Value1	0101 0101 0101 0101
IN2	Tag_Value2	0000 0000 0000 1111
OUT	Tag_Result	0000 0000 0000 0101

If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" and the value of operand "Tag_Value2" are ANDed. The result is mapped bit-for-bit and output in operand "Tag_Result". The enable output ENO and the output "TagOut" are set to signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

OR: OR logic operation (S7-1200, S7-1500)

Description

You can use the "OR logic operation" instruction to combine the value at the IN1 input and the value at the IN2 input bit-by-bit by OR logic and query the result at the OUT output.

When the instruction is executed, bit 0 of the value at the IN1 input and bit 0 of the value at the IN2 input are combined by OR logic operation. The result is stored in bit 0 of output OUT. The same logic operation is executed for all bits of the specified tags.

The number of inputs can be expanded in the instruction box. The added inputs are numbered in ascending order in the box. When the instruction is executed, the values of all available input parameters are combined with OR logic (ORed). The result is stored in the OUT output.

The result bit has signal state "1" when at least one of the two bits in the logic operation has signal state "1". If both of the bits of the logic operation have signal state "0", the corresponding result bit is reset.

Parameters

The following table shows the parameters of the "OR logic operation" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Bit strings	I, Q, M, D, L, P, or constant	First value for logic operation
IN2	Input	Bit strings	I, Q, M, D, L, P, or constant	Second value for logic operation

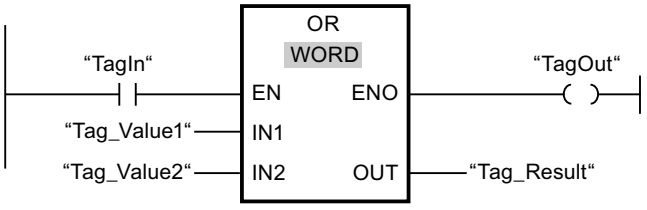
Parameter	Declaration	Data type	Memory area	Description
INn	Input	Bit strings	I, Q, M, D, L, P, or constant	Other inputs whose values are logically combined.
OUT	Output	Bit strings	I, Q, M, D, L, P	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN1	Tag_Value1	0101 0101 0101 0101
IN2	Tag_Value2	0000 0000 0000 1111
OUT	Tag_Result	0101 0101 0101 1111

If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" and the value of operand "Tag_Value2" are ORed. The result is mapped bit-for-bit and output in operand "Tag_Result". The enable output ENO and the output "TagOut" are set to signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

XOR: EXCLUSIVE OR logic operation (S7-1200, S7-1500)

Description

You can use the "EXCLUSIVE OR logic operation" instruction to combine the value at the IN1 input and the value at the IN2 input bit-by-bit by EXCLUSIVE OR logic and query the result at the OUT output.

When the instruction is executed, bit 0 of the value at the IN1 input and bit 0 of the value at the IN2 input are combined by EXCLUSIVE OR logic operation. The result is stored in bit 0 of output OUT. The same logic operation is executed for all other bits of the specified value.

The number of inputs can be expanded in the instruction box. The added inputs are numbered in ascending order in the box. When the instruction is executed, the values of all available input parameters are combined with EXCLUSIVE OR logic. The result is stored in the OUT output.

The result bit has signal state "1" when one of the two bits in the logic operation has signal state "1". If both of the bits of the logic operation have signal state "1" or "0", the corresponding result bit is reset.

Parameters

The following table shows the parameters of the "EXCLUSIVE OR logic operation" instruction:

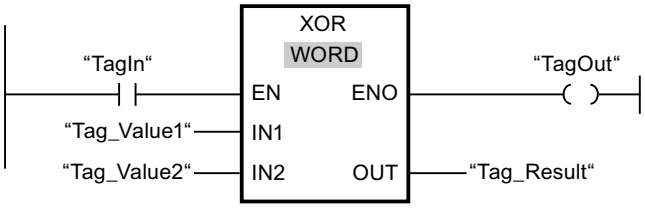
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN1	Input	Bit strings	I, Q, M, D, L, P, or constant	First value for logic operation
IN2	Input	Bit strings	I, Q, M, D, L, P, or constant	Second value for logic operation
INn	Input	Bit strings	I, Q, M, D, L, P, or constant	Other inputs whose values are logically combined.
OUT	Output	Bit strings	I, Q, M, D, L, P	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value
IN1	Tag_Value1	0101 0101 0101 0101
IN2	Tag_Value2	0000 0000 0000 1111
OUT	Tag_Result	0101 0101 0101 1010

If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" and the value of operand "Tag_Value2" are combined by EXCLUSIVE OR logic.

The result is mapped bit-for-bit and output in operand "Tag_Result". The enable output ENO and the output "TagOut" are set to signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

INVERT: Create ones complement (S7-1200, S7-1500)

Description

You can use the instruction "Create ones complement" to invert the signal status of the bits at input IN. When the instruction is processed, the value at input IN is linked to EXCLUSIVE OR by a hexadecimal mask (W#16#FFFF for 16-bit numbers or DW#16#FFFF FFFF for 32-bit number). This inverts the signal state of the individual bits that are then stored at output OUT.

Parameters

The following table shows the parameters of the instruction "Create ones complement":

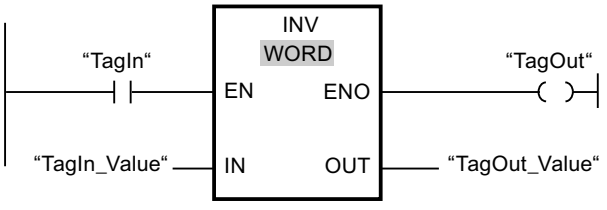
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	I, Q, M, D, L, P, or constant	Input value
OUT	Output	Bit strings, integers	I, Q, M, D, L, P	Ones complement of the value at input IN

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value	
IN	TagIn_Value	W#16#000F	W#16#7E
OUT	TagOut_Value	W#16#FFF0	W#16#81

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction inverts the signal state of the individual bits at input "TagIn_Value" and writes the result to output "TagOut_Value". The enable output ENO and the output "TagOut" are set to signal state "1".

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

DECO: Decode (S7-1200, S7-1500)

Description

You can use the "Decode" instruction to set a bit in the output value that is specified by the input value.

The "Decode" instruction reads the value at the IN input and sets the bit in the output value whose bit position corresponds to the read value. The other bits in the output value are filled with zeroes. When the value at the IN input is greater than 31, a modulo 32 instruction is executed.

Parameters

The following table shows the parameters of the "Decode" instruction:

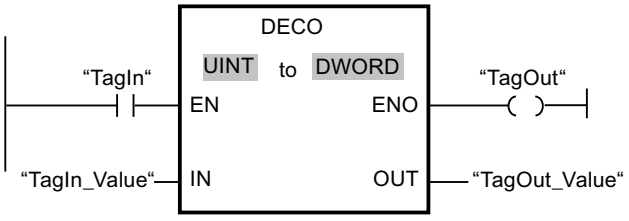
Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	UINT	I, Q, M, D, L, P, or constant	Position of the bit in the output value which is set.
OUT	Output	Bit strings	I, Q, M, D, L, P	Output value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

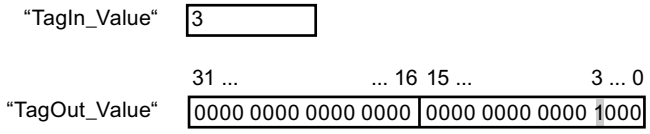
You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following figure shows how the instruction works using specific operand values:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction reads bit number "3" from the value of the operand "TagIn_Value" at the input and sets the third bit to the value of the operand "TagOut_Value" at the output.

If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ENCO: Encode (S7-1200, S7-1500)

Description

You can use the "Encode" instruction to read the bit number of the least significant bit in the input value and to send it to the OUT output.

The "Encode" instruction selects the least significant bit of the value at the IN input and writes its bit number to the tag in the OUT output.

Parameters

The following table shows the parameters of the "Encode" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output

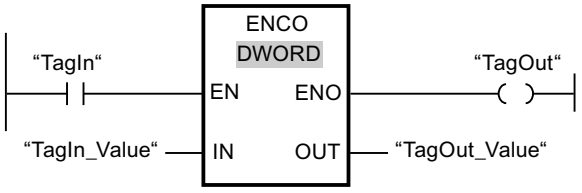
Parameter	Declaration	Data type	Memory area	Description
IN	Input	Bit strings	I, Q, M, D, L, P, or constant	Input value
OUT	Output	INT	I, Q, M, D, L, P	Output value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

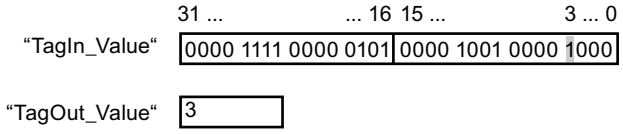
You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following figure shows how the instruction works using specific operand values:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction selects the least significant bit at the "TagIn_Value" input and writes bit position "3" to the tag in the "TagOut_Value" output.

If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

SEL: Select (S7-1200, S7-1500)

Description

Depending on a switch (G input), the "Select" instruction selects one of the inputs, IN0 or IN1 and copies its content to the OUT output. When the input G has signal state "0", the value at the input IN0 is moved. When the input G has signal state "1", the value at the input IN1 is moved to the output OUT.

All tags at all parameters must have the same data type.

Parameters

The following table shows the parameters of the "Select" instruction:

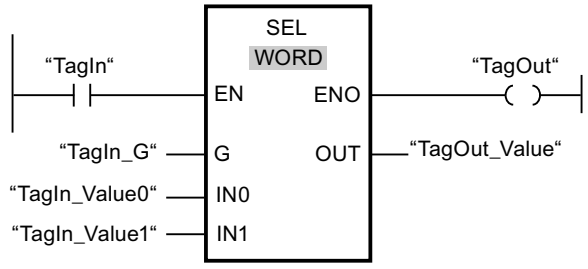
Parameter	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
G	Input	BOOL	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L, T, C or constant	Switch
IN0	Input	Bit strings, integers, floating-point numbers, timers, TOD, CHAR, WCHAR, DATE	Bit strings, integers, floating-point numbers, timers, TOD, LTOD, LDT, CHAR, WCHAR, DATE	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	First input value
IN1	Input	Bit strings, integers, floating-point numbers, timers, TOD, CHAR, WCHAR, DATE	Bit strings, integers, floating-point numbers, timers, TOD, LTOD, LDT, CHAR, WCHAR, DATE	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	Second input value
OUT	Output	Bit strings, integers, floating-point numbers, timers, TOD, CHAR, WCHAR, DATE	Bit strings, integers, floating-point numbers, timers, TOD, LTOD, LDT, CHAR, WCHAR, DATE	I, Q, M, D, L, P	I, Q, M, D, L, P	Result

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value	
G	TagIn_G	0	1
IN0	TagIn_Value0	W#16#0000	W#16#4C
IN1	TagIn_Value1	W#16#FFFF	W#16#5E
OUT	TagOut_Value	W#16#0000	W#16#5E

If the "TagIn" operand has signal state "1", the instruction is executed. Based on the signal state at the "TagIn_G" input, the value at the "TagIn_Value0" or "TagIn_Value1" input is selected and moved to the "TagOut_Value" output. If the instruction is executed without errors, the enable output ENO has signal state "1" and the output "TagOut" is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

MUX: Multiplex (S7-1200, S7-1500)

Description

You can use the instruction "Multiplex" to copy the content of a selected input to output OUT. The number of selectable inputs of the instruction box can be expanded. You can declare a maximum of 32 inputs.

The inputs are numbered automatically in the box. Numbering starts at IN0 and is incremented continuously with each new input. You can use the K parameter to define the input whose content is to be copied to the OUT output. If the value of the parameter K is greater than the number of available inputs, the content of the ELSE parameter is copied to the OUT output and the ENO enable output is assigned the signal state "0".

The "Multiplex" instruction can only be executed if the tags have the same data type in all inputs and in the OUT output. The K parameter is an exception, since only integers can be specified for it.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The input at the K parameter is located outside the available inputs. This applies regardless of whether the ELSE input is used. The value at the OUT output remains changed.
- Errors occurred during execution of the instruction.

Parameters

The following table shows the parameters of the "Multiplex" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
K	Input	Integers	Integers	I, Q, M, D, L, P, or constant	Specifies the input whose content is to be copied. • If K = 0 => Parameter IN0 • If K = 1 => Parameter IN1, etc.
IN0	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P, or constant	First input value
IN1	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P, or constant	Second input value

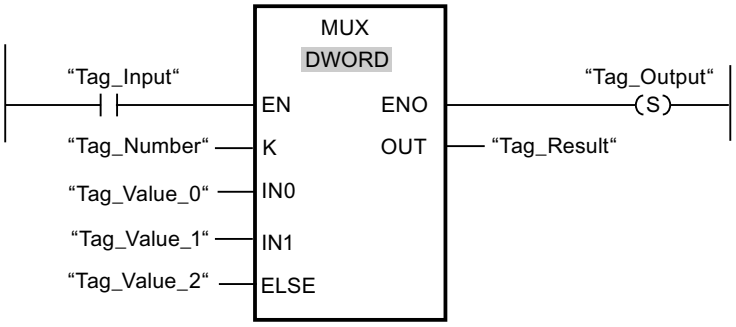
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
INn	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P, or constant	Optional input values
ELSE	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P, or constant	Specifies the value to be copied when K > n.
OUT	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	Output to which the value is to be copied.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
K	Tag_Number	1
IN0	Tag_Value_0	DW#16#00000000
IN1	Tag_Value_1	DW#16#003E4A7D
ELSE	Tag_Value_2	DW#16#FFFF0000
OUT	Tag_Result	DW#16#003E4A7D

If the "Tag_Input" operand has signal state "1", the instruction is executed. Depending on the value of the operand "Tag_Number", the value at input "Tag_Value_1" is copied and assigned to the operand at output "Tag_Result". If the instruction is executed without errors, the "ENO" and "Tag_Output" enable outputs are set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

DEMUX: Demultiplex (S7-1200, S7-1500)

Description

You can use the instruction "Demultiplex" to copy the content of the input IN to a selected output. The number of selectable outputs can be extended in the instruction box. The outputs are numbered automatically in the box. Numbering starts at OUT0 and continues consecutively with each new output. You can use the K parameter to define the output to which the content of the IN input is to be copied. The other outputs are not changed. If the value of the parameter K is greater than the number of available outputs, then the content of input IN is copied to the parameter ELSE and the enable output ENO is assigned the signal state "0".

The instruction "Demultiplex" can only be executed if the tags at the input IN and at all outputs are of the same data type. The K parameter is an exception, since only integers can be specified for it.

The enable output ENO is reset if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The value of the K parameter is greater than the number of available outputs.
- Errors occurred during execution of the instruction.

Parameters

The following table shows the parameters of the instruction "Demultiplex":

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
K	Input	Integers	Integers	I, Q, M, D, L, P, or constant	Specifies the output to which the input value (IN) is copied. • If K = 0 => Parameter OUT0 • If K = 1 => Parameter OUT1, etc.
IN	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P, or constant	Input value
OUT0	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	First output

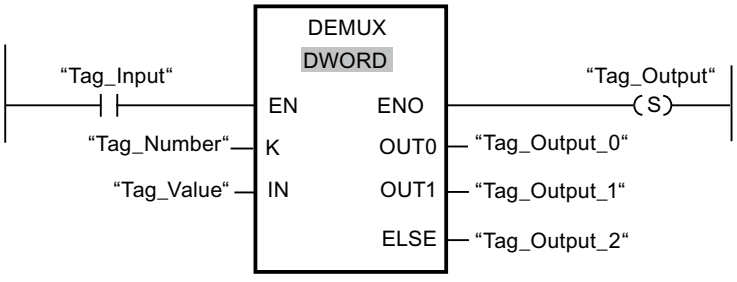
Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
OUT1	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	Second output
OUTn	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	Optional outputs
ELSE	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	Output to which the input value (IN) at K > n is copied.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on available data types, refer to "See also".

Example

The following example shows how the instruction works:



The following tables show how the instruction works using specific operand values:

Table 11-14 Input values of the "Demultiplex" instruction before network execution

Parameter	Operand	Values	
K	Tag_Number	1	4
IN	Tag_Value	DW#16#FFFFFFFF	DW#16#003E4A7D

Table 11-15 Output values of the "Demultiplex" instruction after network execution

Parameter	Operand	Values	
OUT0	Tag_Output_0	Unchanged	Unchanged
OUT1	Tag_Output_1	DW#16#FFFFFFFF	Unchanged
ELSE	Tag_Output_2	Unchanged	DW#16#003E4A7D

If the "Tag_Input" has signal state "1", the instruction is executed. Depending on the value of the operand "Tag_Number", the value at input "IN" is copied to the corresponding output.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Shift and rotate (S7-1200, S7-1500)

SHR: Shift right (S7-1200, S7-1500)

Description

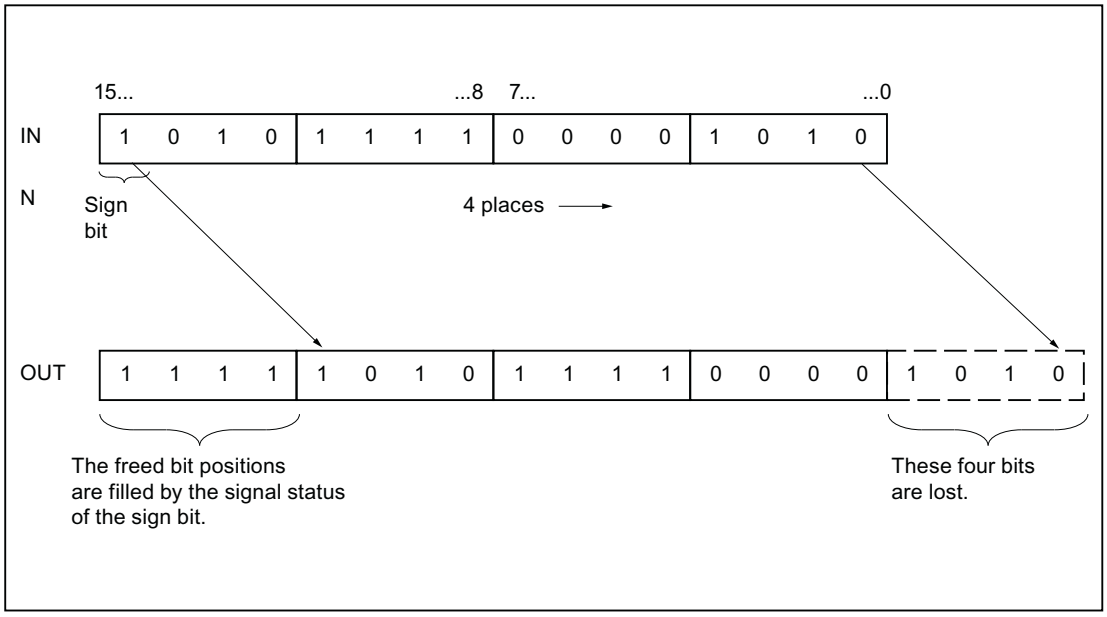
You can use the "Shift right" instruction to shift the content of the operand at the input IN bit-by-bit to the right and query the result at the OUT output. You use the N parameter to specify the number of bit positions by which the specified value is to be shifted.

If the value at the parameter N is "0", the value at input IN is copied to the operand at output OUT.

If the value at the parameter N is greater than the number of bit positions, the operand value at input IN is shifted to the right by the available number of bit positions.

The freed bit positions in the left area of the operand are filled by zeroes when values without signs are shifted. If the specified value has a sign, the free bit positions are filled with the signal state of the sign bit.

The following figure show how the content of an operand of integer data type is shifted four bit positions to the right:



Parameters

The following table shows the parameters of the "Shift right" instruction:

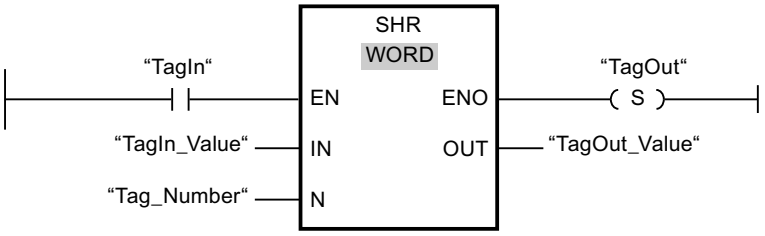
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L or constant	Value to be shifted
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L or constant	Number of bit positions by which the value is shifted
OUT	Output	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	TagIn_Value	0011 1111 1010 1111
N	Tag_Number	3
OUT	TagOut_Value	0000 0111 1111 0101

If operand "TagIn" has the signal state "1", the "Shift right" instruction is executed. The content of operand "TagIn_Value" is shifted three bit positions to the right. The result is sent to the "TagOut_Value" output. If the instruction is executed without errors, the ENO enable output has signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

SHL: Shift left (S7-1200, S7-1500)

Description

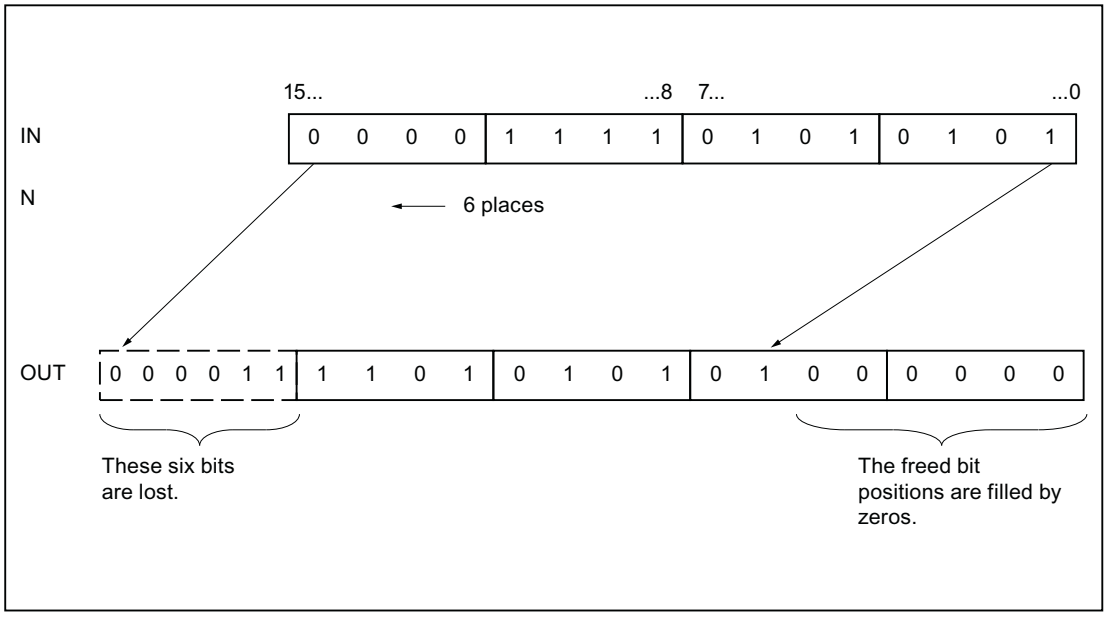
You can use the "Shift left" instruction to shift the content of the operand at the input IN bit-by-bit to the left and query the result at the OUT output. You use the N parameter to specify the number of bit positions by which the specified value is to be shifted.

If the value at the parameter N is "0", the value at input IN is copied to the operand at output OUT.

If the value at the parameter N is greater than the number of bit positions, the operand value at input IN is shifted to the right by the available number of bit positions.

The bit positions in the right part of the operand freed by shifting are filled with zeros.

The following figure shows how the content of an operand of the WORD data type is shifted by six bit positions to the left:



Parameters

The following table shows the parameters of the "Shift left" instruction:

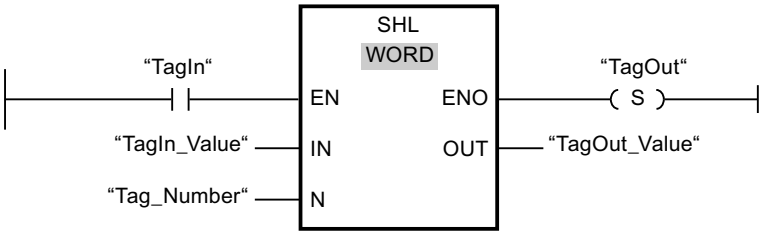
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L or constant	Value to be shifted
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L or constant	Number of bit positions by which the value is shifted
OUT	Output	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	TagIn_Value	0011 1111 1010 1111
N	Tag_Number	4
OUT	TagOut_Value	1111 1010 1111 0000

If operand "TagIn" has the signal state "1", the "Shift left" instruction is executed. The content of operand "TagIn_Value" is shifted four bit positions to the left. The result is sent to the "TagOut_Value" output. If the instruction is executed without errors, the ENO enable output has signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ROR: Rotate right (S7-1200, S7-1500)

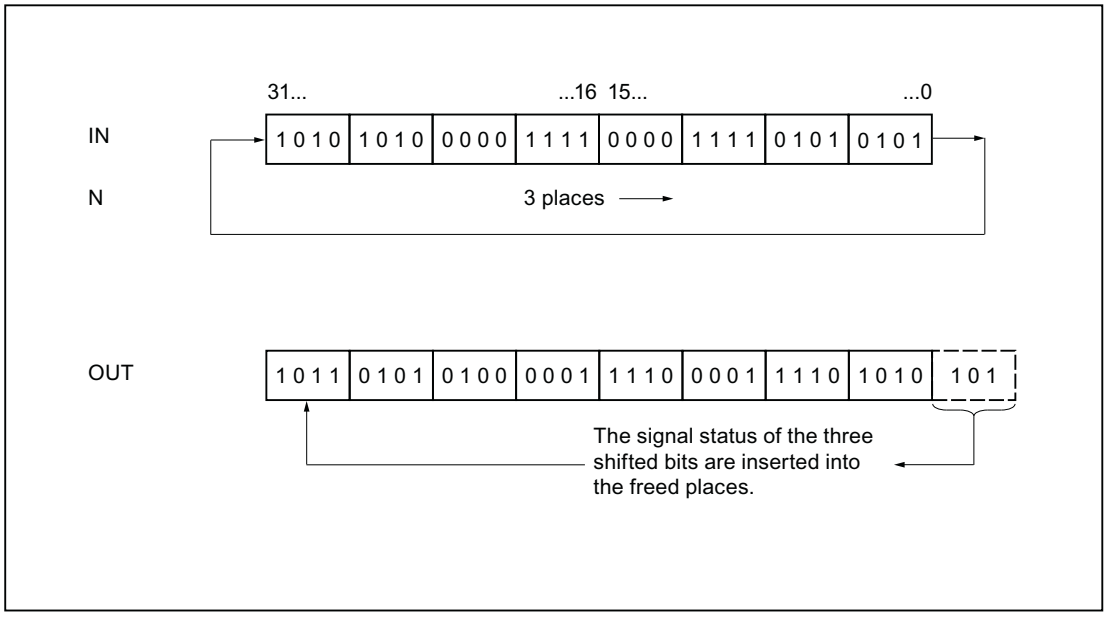
Description

You can use the "Rotate right" instruction to rotate the content of the operand at the input IN bit-by-bit to the right and query the result at the OUT output. You use the N parameter to specify the number of bit positions by which the specified value is to be rotated. The bit positions freed by rotating are filled with the bit positions that are pushed out.

If the value at the parameter N is "0", the value at input IN is copied to the operand at output OUT.

If the value at the parameter N is greater than the number of available bit positions, the operand value at input IN is nevertheless rotated by the specified number of bit positions.

The following figure shows how the content of an operand of DWORD data type is rotated three positions to the right:



Parameters

The following table shows the parameters of the instruction "Rotate right":

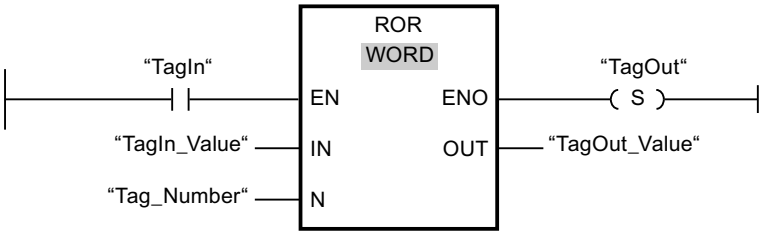
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L or constant	Value to be rotated
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L or constant	Number of bit positions by which the value is rotated
OUT	Output	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	TagIn_Value	0000 1111 1001 0101
N	Tag_Number	5
OUT	TagOut_Value	1010 1000 0111 1100

If operand "TagIn" has the signal state "1", the "Rotate right" instruction is executed. The content of operand "TagIn_Value" is rotated five bit positions to the right. The result is sent to the "TagOut_Value" output. If the instruction is executed without errors, the ENO enable output has signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

ROL: Rotate left (S7-1200, S7-1500)

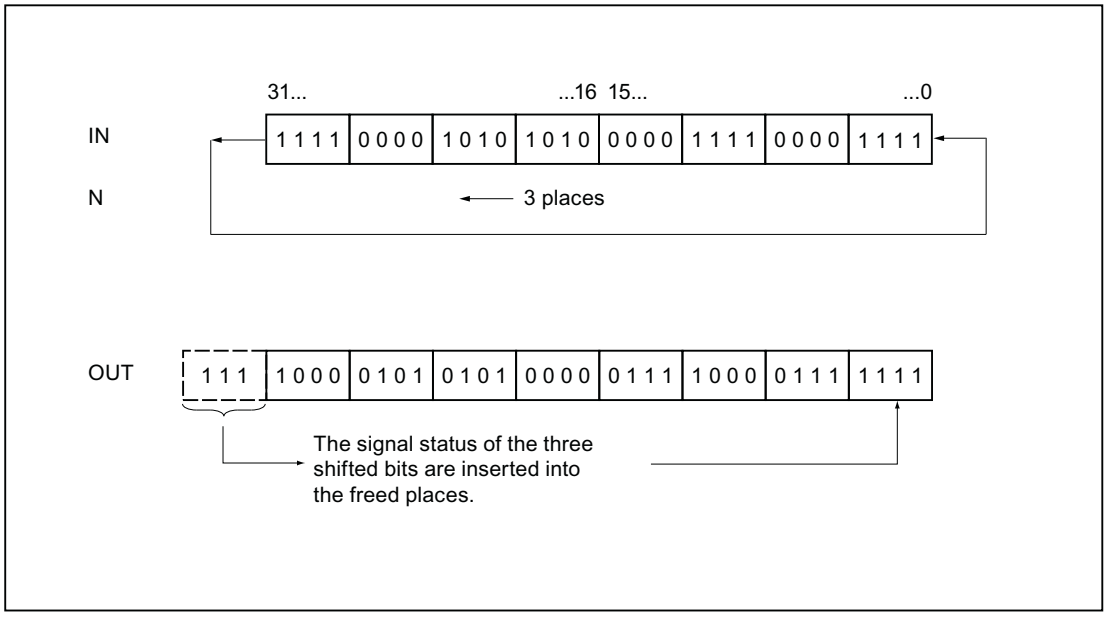
Description

You can use the "Rotate left" instruction to rotate the content of the operand at the input IN bit-by-bit to the left and query the result at the OUT output. You use the N parameter to specify the number of bit positions by which the specified value is to be rotated. The bit positions freed by rotating are filled with the bit positions that are pushed out.

If the value at the parameter N is "0", the value at input IN is copied to the operand at output OUT.

If the value at the parameter N is greater than the number of available bit positions, the operand value at input IN is nevertheless rotated by the specified number of bit positions.

The following figure shows how the content of an operand of DWORD data type is rotated three positions to the left:



Parameters

The following table shows the parameters of the instruction "Rotate left":

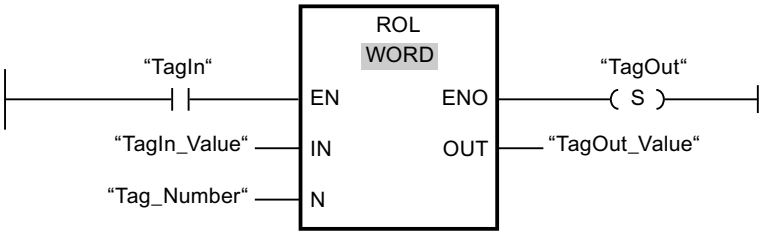
Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L or constant	Value to be rotated
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L or constant	Number of bit positions by which the value is rotated
OUT	Output	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	TagIn_Value	1010 1000 1111 0110
N	Tag_Number	5
OUT	TagOut_Value	0001 1110 1101 0101

If input "TagIn" has the signal state "1", the instruction "Rotate left" is executed. The content of operand "TagIn_Value" is rotated five bit positions to the left. The result is sent to the "TagOut_Value" output. If the instruction is executed without errors, the ENO enable output has signal state "1" and the "TagOut" output is set.

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

Legacy (S7-1500)

DRUM: Implement sequencer (S7-1500)

Description

You can use the "Implement sequencer" instruction to assign the programmed values of the OUT_VAL parameter of the corresponding step to the programmed output bits (OUT1 to OUT16) and the output word (OUT_WORD). The specific step must thereby satisfy the conditions of the programmed enable mask on the S_MASK parameter while the instruction remains at this step. The instruction advances to the next step if the event for the step is true and the programmed time for the current step elapses, or if the value at the JOG parameter changes from "0" to "1". The instruction is reset when the signal state of the RESET parameter changes to "1". The current step is hereby equated to the preset step (DSP).

The amount of time spent on a step is determined by the product of the preset timebase (DTBP) and the preset counter value (S_PRESET) for each step. At the start of a new step, this calculated value is loaded into the DCC parameter, which contains the time remaining for the

current step. If, for example the value at the DTBP parameter is "2" and the preset value for the first step is "100" (100 ms), the DCC parameter has the value "200" (200 ms).

A step can be programmed with a timer value, an event, or both. Steps that have an event bit and the timer value "0" advance to the next step as soon as the signal state of the event bit is "1". Steps that are programmed only with a timer value start the time immediately. Steps that are programmed with an event bit and a timer value greater than "0" start the time when the signal state of the event bit is "1". The event bits are initialized with a signal state of "1".

When the sequencer is on the last programmed step (LST_STEP) and the time for this step has expired, the signal state on the Q parameter is set to "1"; otherwise it is set to "0". When the parameter Q is set, the instruction remains on the step until it is reset.

In the configurable mask (S_MASK) you can select the individual bits in the output word (OUT_WORD) and set or reset the output bits (OUT1 to OUT16) by means of the output values (OUT_VAL). When a bit of the configurable mask has a signal state of "1", the OUT_VAL value sets/resets the corresponding bit. If the signal state of a bit of the configurable mask is "0", the corresponding bit is left unchanged. All the bits of the configurable mask for all 16 steps are initialized with a signal state of "1".

The output bit at the OUT1 parameter corresponds to the least significant bit of the output word (OUT_WORD). The output bit at the OUT16 parameter corresponds to the most significant bit of the output word (OUT_WORD).

Parameters

The following table shows the parameters of the "Implement sequencer" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
RESET	Input	BOOL	I, Q, M, D, L or constant	The signal state "1" indicates a reset condition.
JOG	Input	BOOL	I, Q, M, D, L or constant	When the signal state changes from "0" to "1", the instruction advances to the next step.
DRUM_EN	Input	BOOL	I, Q, M, D, L or constant	The signal state "1" allows the sequencer to advance based on the event and time criteria.
LST_STEP	Input	BYTE	I, Q, M, D, L or constant	Maximum step number (for example: LST_STEP = 16#08; a maximum of 8 steps is possible.)
EVENT(i), 1 ≤ i ≤ 16	Input	BOOL	I, Q, M, D, L or constant	Event bit (i); Initial signal state is "1".
OUT(j), 1 ≤ j ≤ 16	Output	BOOL	I, Q, M, D, L	Output bit (j)
Q	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that the time for the last step has elapsed.

Parameter	Declaration	Data type	Memory area	Description
OUT_WORD	Output	WORD	I, Q, M, D, L, P	Word address to which the sequencer writes the output values.
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
JOG_HIS	Static	BOOL	I, Q, M, D, L	JOG parameter history bit
EOD	Static	BOOL	I, Q, M, D, L or constant	The signal state "1" indicates that the time for the last step has elapsed.
DSP	Static	BYTE	I, Q, M, D, L, P or constant	Preset step of the sequencer
DSC	Static	BYTE	I, Q, M, D, L, P or constant	Current step of the sequencer
DCC	Static	DWORD	I, Q, M, D, L, P or constant	Current numerical value of the sequencer
DTBP	Static	WORD	I, Q, M, D, L, P or constant	Preset timebase of the sequencer
PrevTime	Static	TIME	I, Q, M, D, L or constant	Previous system time
S_PRESET	Static	ARRAY[1..16] of WORD	I, Q, M, D, L or constant	Preset counter value for each step [1 to 16] where one clock pulse = 1 ms.
OUT_VAL	Static	ARRAY[1..16, 0..15] of BOOL	I, Q, M, D, L or constant	Output values for each step [1 to 16, 0 to 15].
S_MASK	Static	ARRAY[1..16, 0..15] of BOOL	I, Q, M, D, L or constant	Configurable mask for each step [1 to 16, 0 to 15]. Initial signal states are "1".

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

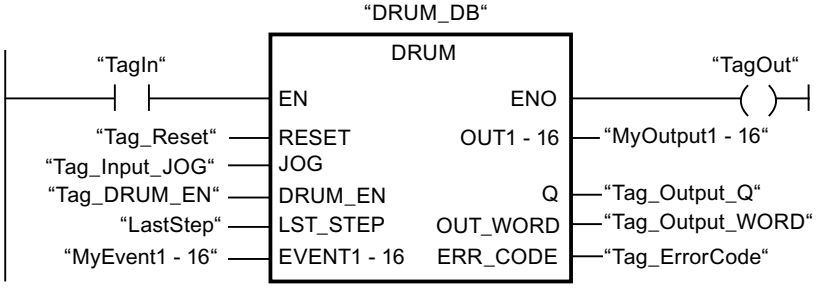
ERR_CODE*	Explanation
W#16#0000	No error
W#16#000B	The value at the LST_STEP parameter is less than 1 or greater than 16.
W#16#000C	The value at the DSC parameter is less than 1 or greater than the value of the LST_STEP parameter.
W#16#000D	The value at the DSP parameter is less than 1 or greater than the value of the LST_STEP parameter.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

In the following example the instruction advances from step 1 to step 2. The output bits (OUT1 to OUT16) and the output word (OUT_WORD) are set according to the mask configured for step 2 and the values of the OUT_VAL parameter.

Note

You can initialize static parameters in the data block.



The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for initializing the input parameters:

Parameter	Operand	Address	Value
RESET	Tag_Reset	M0.0	FALSE
JOG	Tag_Input_JOG	M0.1	FALSE
DRUM_EN	Tag_Input_DrumEN	M0.2	TRUE
LST_STEP	Tag_Number_LastStep	MB1	B#16#08
EVENT2	MyTag_Event_2	M20.0	FALSE
EVENT4	MyTag_Event_4	M20.1	FALSE
EVENT6	MyTag_Event_6	M20.2	FALSE
EVENT8	MyTag_Event_8	M20.3	FALSE
EVENT10	MyTag_Event_10	M20.4	FALSE
EVENT12	MyTag_Event_12	M20.5	FALSE
EVENT14	MyTag_Event_14	M20.6	FALSE
EVENT16	MyTag_Event_16	M20.7	FALSE

The following values are saved in the "DRUM_DB" instance data block of the instruction:

Parameter	Address	Value
JOG_HIS	DBX12.0	FALSE
EOD	DBX12.1	FALSE
DSP	DBB13	W#16#0001
DSC	DBB14	W#16#0001
DCC	DBD16	DW#16#0000000A

Parameter	Address	Value
DTBP	DBW20	W#16#0001
S_PRESET[1]	DBW26	W#16#0064
S_PRESET[2]	DBW28	W#16#00C8
OUT_VAL[1,0]	DBX58.0	TRUE
OUT_VAL[1,1]	DBX58.1	TRUE
OUT_VAL[1,2]	DBX58.2	TRUE
OUT_VAL[1,3]	DBX58.3	TRUE
OUT_VAL[1,4]	DBX58.4	TRUE
OUT_VAL[1,5]	DBX58.5	TRUE
OUT_VAL[1,6]	DBX58.6	TRUE
OUT_VAL[1,7]	DBX58.7	TRUE
OUT_VAL[1,8]	DBX59.0	TRUE
OUT_VAL[1,9]	DBX59.1	TRUE
OUT_VAL[1,10]	DBX59.2	TRUE
OUT_VAL[1,11]	DBX59.3	TRUE
OUT_VAL[1,12]	DBX59.4	TRUE
OUT_VAL[1,13]	DBX59.5	TRUE
OUT_VAL[1,14]	DBX59.6	TRUE
OUT_VAL[1,15]	DBX59.7	TRUE
OUT_VAL[2,0]	DBX60.0	FALSE
OUT_VAL[2,1]	DBX60.1	FALSE
OUT_VAL[2,2]	DBX60.2	FALSE
OUT_VAL[2,3]	DBX60.3	FALSE
OUT_VAL[2,4]	DBX60.4	FALSE
OUT_VAL[2,5]	DBX60.5	FALSE
OUT_VAL[2,6]	DBX60.6	FALSE
OUT_VAL[2,7]	DBX60.7	FALSE
OUT_VAL[2,8]	DBX61.0	FALSE
OUT_VAL[2,9]	DBX61.1	FALSE
OUT_VAL[2,10]	DBX61.2	FALSE
OUT_VAL[2,11]	DBX61.3	FALSE
OUT_VAL[2,12]	DBX61.4	FALSE
OUT_VAL[2,13]	DBX61.5	FALSE
OUT_VAL[2,14]	DBX61.6	FALSE
OUT_VAL[2,15]	DBX61.7	FALSE
S_MASK[2,0]	DBX92.0	FALSE
S_MASK[2,1]	DBX92.1	TRUE
S_MASK[2,2]	DBX92.2	TRUE
S_MASK[2,3]	DBX92.3	TRUE
S_MASK[2,4]	DBX92.4	TRUE
S_MASK[2,5]	DBX92.5	FALSE
S_MASK[2,6]	DBX92.6	TRUE
S_MASK[2,7]	DBX92.7	TRUE

Parameter	Address	Value
S_MASK[2,8]	DBX93.0	FALSE
S_MASK[2,9]	DBX93.1	FALSE
S_MASK[2,10]	DBX93.2	TRUE
S_MASK[2,11]	DBX93.3	TRUE
S_MASK[2,12]	DBX93.4	TRUE
S_MASK[2,13]	DBX93.5	TRUE
S_MASK[2,14]	DBX93.6	FALSE
S_MASK[2,15]	DBX93.7	TRUE

The output parameters are set to the following values before the execution of the instruction:

Parameter	Operand	Address	Value
Q	Tag_Output_Q	M6.0	FALSE
OUTWORD	Tag_OutputWord	MW8	W#16#FFFF
OUT1	MyTag_Output_1	M4.0	TRUE
OUT2	MyTag_Output_2	M4.1	TRUE
OUT3	MyTag_Output_3	M4.2	TRUE
OUT4	MyTag_Output_4	M4.3	TRUE
OUT5	MyTag_Output_5	M4.4	TRUE
OUT6	MyTag_Output_6	M4.5	TRUE
OUT7	MyTag_Output_7	M4.6	TRUE
OUT8	MyTag_Output_8	M4.7	TRUE
OUT9	MyTag_Output_9	M5.0	TRUE
OUT10	MyTag_Output_10	M5.1	TRUE
OUT11	MyTag_Output_11	M5.2	TRUE
OUT12	MyTag_Output_12	M5.3	TRUE
OUT13	MyTag_Output_13	M5.4	TRUE
OUT14	MyTag_Output_14	M5.5	TRUE
OUT15	MyTag_Output_15	M5.6	TRUE
OUT16	MyTag_Output_16	M5.7	TRUE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameter	Operand	Address	Value
OUT1	MyTag_Output_1	M4.0	TRUE
OUT2	MyTag_Output_2	M4.1	FALSE
OUT3	MyTag_Output_3	M4.2	FALSE
OUT4	MyTag_Output_4	M4.3	FALSE
OUT5	MyTag_Output_5	M4.4	FALSE
OUT6	MyTag_Output_6	M4.5	TRUE
OUT7	MyTag_Output_7	M4.6	FALSE

Parameter	Operand	Address	Value
OUT8	MyTag_Output_8	M4.7	FALSE
OUT9	MyTag_Output_9	M5.0	TRUE
OUT10	MyTag_Output_10	M5.1	TRUE
OUT11	MyTag_Output_11	M5.2	FALSE
OUT12	MyTag_Output_12	M5.3	FALSE
OUT13	MyTag_Output_13	M5.4	FALSE
OUT14	MyTag_Output_14	M5.5	FALSE
OUT15	MyTag_Output_15	M5.6	TRUE
OUT16	MyTag_Output_16	M5.7	FALSE
Q	Tag_Output_Q	M6.0	FALSE
OUTWORD	Tag_OutputWord	MW8	W#16#4321
ERR_CODE	Tag_ErrorCode	MW10	W#16#0000

The following values are changed in the instance data block "DRUM_DB" of the instruction after the execution of the instruction:

Parameter	Address	Value
JOG_HIS	DBX12.0	FALSE
EOD	DBX12.1	FALSE
DSC	DBB14	W#16#0002
DCC	DBD16	DW#16#000000C8

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

DCAT: Discrete control-timer alarm (S7-1500)

Description

You can use the "Discrete control-timer alarm" to count the time from the point at which the CMD parameter issues the command to open or close. The time is accumulated until the preset time (PT) is exceeded or the information is received that the device was opened or closed (O_FB or C_FB) within the specified time. If the preset time is exceeded before the information on the opening or closing of the device is received, the corresponding alarm is activated. If the signal state of the command input changes before the preset time, the time is restarted.

The "Discrete control-timer alarm" instruction has the following reactions to the input conditions:

- When the signal state of the CMD parameter changes from "0" to "1", the signal states of the parameters Q, CMD_HIS, ET (only if ET is < PT) OA and CA are influenced as follows:
  - The parameters Q and CMD_HIS are set to "1".
  - The parameters ET, OA and CA are reset to "0".
- When the signal state of the parameter CMD changes from "1" to "0", the parameters Q, ET (only if ET < PT), OA, CA and CMD_HIS are reset to "0".
- When the signal state of the parameters CMD and CMD_HIS is "1" and the parameter O_FB is set to "0", the time difference (ms) since the last execution of the instruction is added to the value at the parameter ET. If the value of the ET parameter exceeds the value of the PT parameter, the signal state at the OA parameter is set to "1". If the value of the ET parameter does not exceed the value of the PT parameter, the signal state of the OA parameter is reset to "0". The value at the parameter CMD_HIS is reset to the value of the parameter CMD.
- If the signal state of the CMD, CMD_HIS and O_FB parameters are set to "1" and the parameter C_FB has the value "0", the signal state of the parameter OA is set to "0". The value of the ET parameter is set to the value of the PT parameter. If the signal state of the O_FB parameter changes to "0", the alarm is set the next time the instruction is executed. The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the parameters CMD, CMD_HIS and C_FB have the value "0", the time difference (ms) since the last execution of the instruction is added to the value of the parameter ET. If the value of the ET parameter exceeds the value of the PT parameter, the signal state of the parameter CA is reset to "1". If the value at the parameter PT is not exceeded, the parameter CA has the signal state "0". The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the parameters CMD, CMD_HIS and O_FB have the signal state "0" and the parameter C_FB is set to "1", the parameter CA is set to "0". The value of the ET parameter is set to the value of the PT parameter. If the signal state of the C_FB parameter changes to "0", the alarm is set the next time the instruction is executed. The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the parameters O_FB and C_FB simultaneously have the signal state "1", the signal states of both alarm outputs are set to "1".

The "Discrete control-timer alarm" instruction has no error information.

Parameters

The following table shows the parameters of the "Discrete control-timer alarm" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
CMD	Input	BOOL	I, Q, M, D, L or constant	A signal state of "0" indicates a "close" command. A signal state of "1" indicates an "open" command.

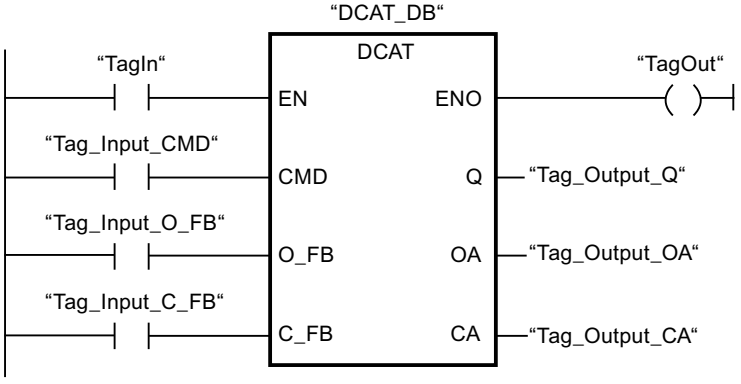
Parameter	Declaration	Data type	Memory area	Description
O_FB	Input	BOOL	I, Q, M, D, L or constant	Feedback input when opening
C_FB	Input	BOOL	I, Q, M, D, L or constant	Feedback input when closing
Q	Output	BOOL	I, Q, M, D, L	Shows the status of the parameter CMD
OA	Output	BOOL	I, Q, M, D, L	Alarm output when opening
CA	Output	BOOL	I, Q, M, D, L	Alarm output when closing
ET	Static	DINT	D, L or constant	Currently elapsed time, where one clock pulse = 1 ms
PT	Static	DINT	D, L or constant	Preset timer value, where one clock pulse = 1 ms
PREV_TIME	Static	DWORD	D, L or constant	Previous system time
CMD_HIS	Static	BOOL	D, L or constant	CMD history bit

Example

In the following example the parameter CMD changes from "0" to "1". After the execution of the instruction the parameter Q is set to "1" and the two alarm outputs OA and CA have the signal state "0". The CMD_HIS parameter of the instance data block is set to the signal state "1" and the ET parameter is reset to "0".

Note

You can initialize static parameters in the data block.



The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for the input and output parameters:

Parameter	Operand	Value
CMD	Tag_Input_CMD	TRUE
O_FB	Tag_Input_O_FB	FALSE
C_FB	Tag_Input_C_FB	FALSE

Parameter	Operand	Value
Q	Tag_Output_Q	FALSE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE

The following values are saved in the instance data block "DCAT_DB" of the instruction:

Parameter	Address	Value
ET	DBD4	L#12
PT	DBD8	L#222
CMD_HIS	DBX16.0	FALSE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameter	Operand	Value
Q	Tag_Output_Q	TRUE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE

The following values are saved in the instance data block "DCAT_DB" of the instruction:

Parameter	Address	Value
ET	DBD4	L#0
CMD_HIS	DBX16.0	TRUE

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

MCAT: Motor control-timer alarm (S7-1500)

Description

The "Motor control-timer alarm" instruction is used to accumulate the time from the point at which one of the command inputs (opening or closing) is switched on. The time is accumulated until the preset time is exceeded or the relevant feedback input indicates that the device has executed the requested operation within the specified time. If the preset time is exceeded before the feedback is received, the corresponding alarm is triggered.

Parameters

The following table shows the parameters of the "Motor control-timer alarm" instruction:

Parameters	Declaration	Data type	Memory space	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
O_CMD	Input	BOOL	I, Q, M, D, L or constant	"Open" command input
C_CMD	Input	BOOL	I, Q, M, D, L or constant	"Close" command input
S_CMD	Input	BOOL	I, Q, M, D, L or constant	"Stop" command input
O_FB	Input	BOOL	I, Q, M, D, L or constant	Feedback input when opening
C_FB	Input	BOOL	I, Q, M, D, L or constant	Feedback input when closing
OO	Output	BOOL	I, Q, M, D, L	"Open" output
CO	Output	BOOL	I, Q, M, D, L	"Close" output
OA	Output	BOOL	I, Q, M, D, L	Alarm output when opening
CA	Output	BOOL	I, Q, M, D, L	Alarm output when closing
Q	Output	BOOL	I, Q, M, D, L	A signal state of "0" indicates an error condition.
ET	Static	DINT	D, L or constant	Currently elapsed time, where one count = 1 ms
PT	Static	DINT	D, L or constant	Preset timer value, where one clock pulse = 1 ms
PREV_TIME	Static	DWORD	D, L or constant	Previous system time
O_HIS	Static	BOOL	D, L or constant	"Open" history bit
C_HIS	Static	BOOL	D, L or constant	"Close" history bit

Execution of the "Motor control-timer alarm" instruction

The following table shows the reactions of the "Motor control-timer alarm" instruction to the various input conditions:

Input parameters								Output parameters								
ET	O_HIS	C_HIS	O_CMD	C_CMD	S_CMD	O_FB	C_FB	OO	CO	OA	CA	ET	O_HIS	C_HIS	Q	Status
X	1	1	X	X	X	X	X	0	0	1	1	PT	0	0	0	Alarm
X	X	X	X	X	X	1	1	0	0	1	1	PT	0	0	0	Alarm
X	X	X	X	X	1	X	X	0	0	0	0	X	0	0	1	Stop
X	X	X	1	1	X	X	X	0	0	0	0	X	0	0	1	Stop
X	0	X	1	0	0	X	X	1	0	0	0	0	1	0	1	Start opening
<PT	1	0	X	0	0	0	X	1	0	0	0	INC	1	0	1	Open
X	1	0	X	0	0	1	0	0	0	0	0	PT	1	0	1	Opened

Input parameters								Output parameters								
>=PT	1	0	X	0	0	0	X	0	0	1	0	PT	1	0	0	Opening alarm
X	X	0	0	1	0	X	X	0	1	0	0	0	0	1	1	Start closing
<PT	0	1	0	X	0	X	0	0	1	0	0	INC	0	1	1	Close
X	0	1	0	X	0	0	1	0	0	0	0	PT	0	1	1	Closed
>=PT	0	1	0	X	0	X	0	0	0	0	1	PT	0	1	0	Closing alarm
X	0	0	0	0	0	X	X	0	0	0	0	X	0	0	1	Stopped

Legend:	
INC	Add the time difference (ms) since the last processing of the FB to ET
PT	PT is set to the same value as ET
X	Cannot be used
<PT	ET < PT
>=PT	ET >= PT

If the input parameters O_HIS and C_HIS both have the signal state "1", they are immediately set to signal state "0". In this case, the last row in the table (X) mentioned above is valid. Because it is therefore no longer possible to check whether the input parameters O_HIS and C_HIS have the signal state "1", the output parameters are set as follows in this case:

OO = FALSE

CO = FALSE

OA = FALSE

CA = FALSE

ET = PT

Q = TRUE

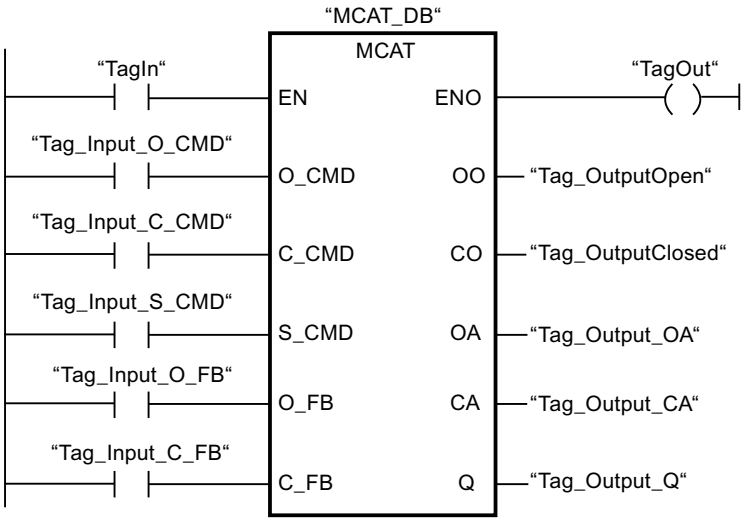
The "Motor control-timer alarm" instruction returns no error information.

Example

The following example shows how the instruction works:

Note

You can initialize static parameters in the data block.



The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for the input and output parameters:

Parameter	Operand	Value
O_CMD	Tag_Input_O_CMD	TRUE
C_CMD	Tag_Input_C_CMD	FALSE
S_CMD	Tag_Input_S_CMD	FALSE
O_FB	Tag_Input_O_FB	FALSE
C_FB	Tag_Input_C_FB	FALSE
OO	Tag_OutputOpen	FALSE
CO	Tag_OutputClosed	FALSE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE
Q	Tag_Output_Q	FALSE

The following values are saved in the instance data block "MCAT_DB" of the instruction:

Parameter	Address	Value
ET	DBD4	L#2
PT	DBD8	L#22
O_HIS	DBX16.0	TRUE
C_HIS	DBX16.1	FALSE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameter	Operand	Value
OO	Tag_OutputOpen	TRUE
CO	Tag_OutputClosed	FALSE

Parameter	Operand	Value
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE
Q	Tag_Output_Q	TRUE

The following values are saved in the instance data block "MCAT_DB" of the instruction:

Parameter	Address	Value
ET	DBD4	L#0
O_HIS	DBX16.0	TRUE
CMD_HIS	DBX16.1	FALSE

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

IMC: Compare input bits with the bits of a mask (S7-1500)

Description

You can use the "Compare input bits with the bits of a mask" instruction to compare the signal state of up to 16 programmed input bits (IN_BIT0 to IN_BIT15) with the corresponding bits of a mask. Up to 16 steps with masks can be programmed. The value of the IN_BIT0 parameter is compared with the value of the mask CMP_VAL[x,0], with "x" indicating the step number. In the CMP_STEP parameter, you specify the step number of the mask that is used for the comparison. All programmed values are compared in the same manner. Unprogrammed input bits or unprogrammed bits of the mask have the default signal state FALSE.

If a match is found in the comparison, the signal state of the OUT parameter is set to "1". Otherwise, the OUT parameter is set to "0".

If the value of the CMP_STEP parameter is greater than 15, the instruction is not executed. An error message is output at the ERR_CODE parameter.

Parameters

The following table shows the parameters of the "Compare input bits with the bits of a mask" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN_BIT0	Input	BOOL	I, Q, M, D, L or constant	Input bit 0 is compared with bit 0 of the mask.

Parameter	Declaration	Data type	Memory area	Description
IN_BIT1	Input	BOOL	I, Q, M, D, L or constant	Input bit 1 is compared with bit 1 of the mask.
IN_BIT2	Input	BOOL	I, Q, M, D, L or constant	Input bit 2 is compared with bit 2 of the mask.
IN_BIT3	Input	BOOL	I, Q, M, D, L or constant	Input bit 3 is compared with bit 3 of the mask.
IN_BIT4	Input	BOOL	I, Q, M, D, L or constant	Input bit 4 is compared with bit 4 of the mask.
IN_BIT5	Input	BOOL	I, Q, M, D, L or constant	Input bit 5 is compared with bit 5 of the mask.
IN_BIT6	Input	BOOL	I, Q, M, D, L or constant	Input bit 6 is compared with bit 6 of the mask.
IN_BIT7	Input	BOOL	I, Q, M, D, L or constant	Input bit 7 is compared with bit 7 of the mask.
IN_BIT8	Input	BOOL	I, Q, M, D, L or constant	Input bit 8 is compared with bit 8 of the mask.
IN_BIT9	Input	BOOL	I, Q, M, D, L or constant	Input bit 9 is compared with bit 9 of the mask.
IN_BIT10	Input	BOOL	I, Q, M, D, L or constant	Input bit 10 is compared with bit 10 of the mask.
IN_BIT11	Input	BOOL	I, Q, M, D, L or constant	Input bit 11 is compared with bit 11 of the mask.
IN_BIT12	Input	BOOL	I, Q, M, D, L or constant	Input bit 12 is compared with bit 12 of the mask.
IN_BIT13	Input	BOOL	I, Q, M, D, L or constant	Input bit 13 is compared with bit 13 of the mask.
IN_BIT14	Input	BOOL	I, Q, M, D, L or constant	Input bit 14 is compared with bit 14 of the mask.
IN_BIT15	Input	BOOL	I, Q, M, D, L or constant	Input bit 15 is compared with bit 15 of the mask.
CMP_STEP	Input	BYTE	I, Q, M, D, L, P or constant	The step number of the mask used for the comparison.
OUT	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that a match was found. A signal state of "0" indicates that no match was found.
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
CMP_VAL	Static	ARRAY OF WORD	I, Q, M, D, L or constant	Comparison masks [0 to 15, 0 to 15]: The first number of the index is the step number and the second number is the bit number of the mask.

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

Error code* (W#16#...)	Explanation
0000	No error
000A	The value at the CMP_STEP parameter is greater than 15.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

SMC: Compare scan matrix (S7-1500)

Description

You can use the "Compare scan matrix" instruction to compare the signal state of up to 16 programmed input bits (IN_BIT0 to IN_BIT15) with the corresponding bit of the comparison masks for each step. Processing starts at step 1 and is continued until the last programmed step (LAST) or until a match is found. The input bit of the IN_BIT0 parameter is compared with the value of the mask CMP_VAL[x,0], with "x" indicating the step number. All programmed values are compared in the same manner. If a match is found the signal state of the OUT parameter is set to "1" and the step number with the matching mask is written in the OUT_STEP parameter. Unprogrammed input bits or unprogrammed bits of the mask have the default signal state FALSE. If more than one step has a matching mask, only the first one found is indicated in the OUT_STEP parameter. If no match is found, the signal state of the OUT parameter is set to "0". In this case the value at the OUT_STEP parameter is greater by "1" than the value at the LAST parameter.

Parameters

The following table shows the parameters of the "Compare scan matrix" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN_BIT0	Input	BOOL	I, Q, M, D, L or constant	Input bit 0 is compared with bit 0 of the mask.
IN_BIT1	Input	BOOL	I, Q, M, D, L or constant	Input bit 1 is compared with bit 1 of the mask.
IN_BIT2	Input	BOOL	I, Q, M, D, L or constant	Input bit 2 is compared with bit 2 of the mask.

Parameter	Declaration	Data type	Memory area	Description
IN_BIT3	Input	BOOL	I, Q, M, D, L or constant	Input bit 3 is compared with bit 3 of the mask.
IN_BIT4	Input	BOOL	I, Q, M, D, L or constant	Input bit 4 is compared with bit 4 of the mask.
IN_BIT5	Input	BOOL	I, Q, M, D, L or constant	Input bit 5 is compared with bit 5 of the mask.
IN_BIT6	Input	BOOL	I, Q, M, D, L or constant	Input bit 6 is compared with bit 6 of the mask.
IN_BIT7	Input	BOOL	I, Q, M, D, L or constant	Input bit 7 is compared with bit 7 of the mask.
IN_BIT8	Input	BOOL	I, Q, M, D, L or constant	Input bit 8 is compared with bit 8 of the mask.
IN_BIT9	Input	BOOL	I, Q, M, D, L or constant	Input bit 9 is compared with bit 9 of the mask.
IN_BIT10	Input	BOOL	I, Q, M, D, L or constant	Input bit 10 is compared with bit 10 of the mask.
IN_BIT11	Input	BOOL	I, Q, M, D, L or constant	Input bit 11 is compared with bit 11 of the mask.
IN_BIT12	Input	BOOL	I, Q, M, D, L or constant	Input bit 12 is compared with bit 12 of the mask.
IN_BIT13	Input	BOOL	I, Q, M, D, L or constant	Input bit 13 is compared with bit 13 of the mask.
IN_BIT14	Input	BOOL	I, Q, M, D, L or constant	Input bit 14 is compared with bit 14 of the mask.
IN_BIT15	Input	BOOL	I, Q, M, D, L or constant	Input bit 15 is compared with bit 15 of the mask.
OUT	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that a match was found. The signal state "0" indicates that no match was found.
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
OUT_STEP	Output	BYTE	I, Q, M, D, L, P	Contains the step number with the matching mask or the step number which is greater by "1" than the value of the LAST parameter, provided no match is found.
LAST	Static	BYTE	I, Q, M, D, L, P or constant	Specifies the step number of the last step to be scanned for a matching mask.
CMP_VAL	Static	ARRAY OF WORD	I, Q, M, D, L or constant	Comparison masks [0 to 15, 0 to 15]: The first number of the index is the step number and the second number is the bit number of the mask.

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

Error code* (W#16#...)	Explanation
0000	No error
000E	The value at the LAST parameter is greater than 15.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

LEAD_LAG: Lead and lag algorithm (S7-1500)

Description

You can use the "Lead and lag algorithm" instruction to process signals with an analog tag. The gain value at the GAIN parameter must be greater than zero. The result of the instruction "Lead and lag algorithm" is calculated using the following equation:

$$OUT = \left[\frac{LG_TIME}{LG_TIME + SAMPLE_T}\right] PREV_OUT + GAIN \left[\frac{LD_TIME + SAMPLE_T}{LG_TIME + SAMPLE_T}\right] IN - GAIN \left[\frac{LD_TIME}{LG_TIME + SAMPLE_T}\right] * PREV_IN$$

The "Lead and lag algorithm" instruction supplies plausible results only when processing is in fixed program cycles. The same units must be specified for the LD_TIME, LG_TIME, and SAMPLE_T parameters. At LG_TIME > 4 + SAMPLE_T, the instruction approaches the following function:

OUT = GAIN * ((1 + LD_TIME * s) / (1 + LG_TIME * s)) * IN

When the value of the GAIN parameter is less than or equal to zero, the calculation is not performed and an error information is output on the ERR_CODE parameter.

You can use the instruction "Lead and lag algorithm" in conjunction with loops as a compensator in dynamic feed-forward control. The instruction consists of two operations. The "Lead" operation shifts the phase of the OUT output so that the output leads the input. The "Lag" operation, on the other hand, shifts the output so that the output lags behind the input. Because the "Lag" operation is equivalent to an integration, it can be used as a noise suppressor or as a low-pass filter. The "Lead" operation is equivalent to a differentiation and can therefore be used as a high-pass filter. The two operations together (Lead and Lag) result in the output phase lagging behind the the input at lower frequencies and leading it at higher frequencies. This means that the "Lead and lag algorithm" instruction can be used as a band pass filter.

Parameters

The following table shows the parameters of the "Lead and lag algorithm" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	REAL	I, Q, M, D, L, P or constant	The input value of the current sample time (cycle time) to be processed. Constants can also be specified on the IN parameter.
SAMPLE_T	Input	INT	I, Q, M, D, L, P or constant	Sample time Constants can also be specified on the SAMPLE_T parameter.
OUT	Output	REAL	I, Q, M, D, L	Result of the instruction
ERR_CODE	Output	WORD	I, Q, M, D, L	Error information
LD_TIME	Static	REAL	I, Q, M, D, L, P or constant	Lead time in the same unit as sample time.
LG_TIME	Static	REAL	I, Q, M, D, L, P or constant	Lag time in in the same unit as sample time.
GAIN	Static	REAL	I, Q, M, D, L, P or constant	Gain as % / % (the ratio of the change in output to a change in input as a steady state).
PREV_IN	Static	REAL	I, Q, M, D, L, P or constant	Previous input
PREV_OUT	Static	REAL	I, Q, M, D, L, P or constant	Previous output

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

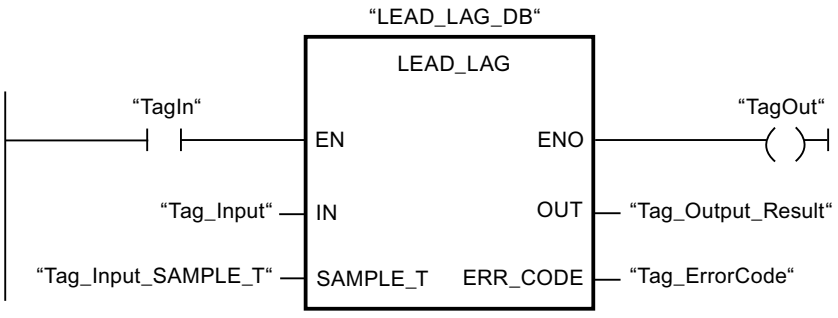
Error code* (W#16#...)	Explanation
0000	No error
0009	The value at the GAIN parameter is less than or equal to zero.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:

Note

You can initialize static parameters in the data block.



The following tables show how the instruction works using specific values.

Before processing

In this example the following values are used for the input parameters:

Parameter	Operand	Value
IN	Tag_Input	2.0
SAMPLE_T	Tag_InputSampleTime	10

The following values are saved in the "LEAD_LAG_DB" instance data block of the instruction:

Parameter	Address	Value
LD_TIME	DBD12	2.0
LG_TIME	DBD16	2.0
GAIN	DBD20	1.0
PREV_IN	DBD24	6.0
PREV_OUT	DBD28	6.0

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameter	Operand	Value
OUT	Tag_Output_Result	2.0

The following values are saved in the instance data block "LEAD_LAD_DB" of the instruction:

Parameter	Operand	Value
PREV_IN	DBD24	2.0
PREV_OUT	DBD28	2.0

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

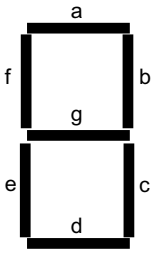
Basic information on LAD (Page 4878)

SEG: Create bit pattern for seven-segment display (S7-1500)

Description

The instruction "Create bit pattern for seven-segment display" is used to convert each of the four hexadecimal digits of the specified source word (IN) into an equivalent bit pattern for a seven-segment display. The result of the instruction is output in the double word on the OUT parameter.

The following relation exists between the hexadecimal digits and the assignment of the 7 segments (a, b, c, d, e, f, g):

Input digit (Binary)	Assignment of the segments - g f e d c b a	Display (Hexadecimal)	Seven-segment display
0000	00111111	0	
0001	00000110	1	
0010	01011011	2	
0011	01001111	3	
0100	01100110	4	
0101	01101101	5	
0110	01111101	6	
0111	00000111	7	
1000	01111111	8	
1001	01100111	9	
1010	01110111	A	
1011	01111100	B	
1100	00111001	C	
1101	01011110	D	
1110	01111001	E	
1111	01110001	F	

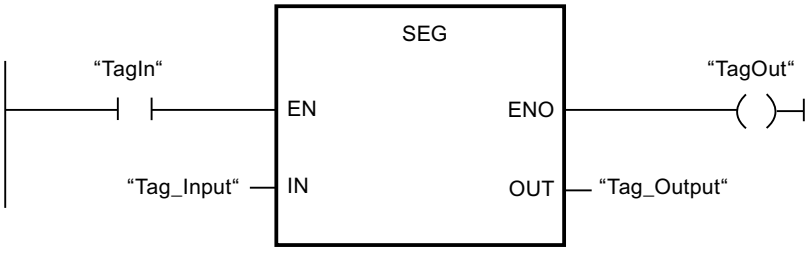
Parameter

The following table shows the parameters of the "Create bit pattern for seven-segment display" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	WORD	I, Q, M, D, L, P, or constant	Source word with four hexadecimal digits
OUT	Output	DWORD	I, Q, M, D, L, P	Bit pattern for the seven-segment display

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Value	
Hexadecimal	Binary		
IN	Tag_Input	W#16#1234	0001 0010 0011 0100
OUT	Tag_Output	DW#16065B4F66	000 00110 0101 1011 0100 1111 0110 0110 Display: 1234

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

BCDCPL: Create tens complement (S7-1500)

Description

You can use the "Create tens complement" instruction to create the tens complement of a seven-digit BCD number specified in the IN parameter. This instruction uses the following mathematical formula to calculate:

10000000 (as BCD)

– 7-digit BCD value

Tens complement (as BCD)

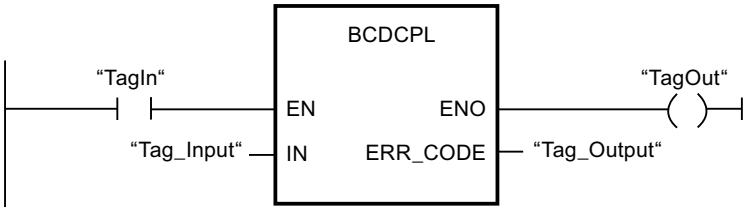
Parameters

The following table shows the parameters of the "Create tens complement" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Bit strings	I, Q, M, D, L, P, or constant	7-digit BCD number
ERR_CODE	Output	DWORD	I, Q, M, D, L, P	Result of the instruction

Example

The following example shows how the instruction works:



The following table shows how the instruction functions using specific values:

Parameter	Operand	Value*
IN	Tag_Input	DW#16#01234567
ERR_CODE	Tag_Output	DW#16#08765433
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".		

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

BITSUM: Count number of set bits (S7-1500)

Description

The "Count number of set bits" instruction is used to count the number of bits of an operand that is set to the signal state "1". The operand whose bits are to be counted is specified on the IN parameter. The result of the instruction is output on the RET_VAL parameter.

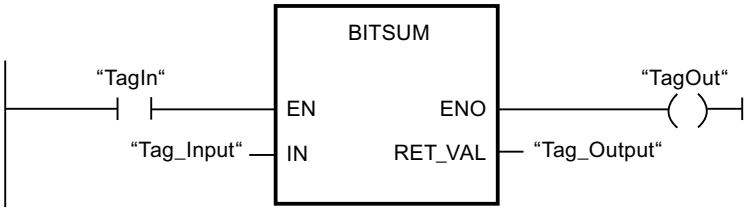
Parameters

The following table shows the parameters of the "Count number of set bits" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	DWORD	I, Q, M, D, L, P, or constant	Operand whose set bits are counted
RET_VAL	Output	INT	I, Q, M, D, L, P	Number of bits to be set

Example

The following example shows how the instruction works:



The following table shows how the instruction functions using specific values:

Parameter	Operand	Value*
IN	Tag_Input	DW#16#12345678
RET_VAL	Tag_Output	W#16#000D (13 Bits)
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".		

For more information and the program code to the above named example, refer to: Sample Library for Instructions (Page 2568)

See also

Overview of the valid data types (Page 2239)

Basic information on LAD (Page 4878)

11.4.3.2 FBD (S7-1200, S7-1500)

Bit logic operations (S7-1200, S7-1500)

&: AND logic operation (S7-1200, S7-1500)

Description

You can use the instruction "AND logic operation" to query the signal states of two or more specified operands and evaluate them according to the AND truth table.

If the signal state of all the operands is "1", then the condition is fulfilled and the instruction returns the result "1". If the signal state of one of the operands is "0", then the condition is not fulfilled and the instruction generates the result "0".

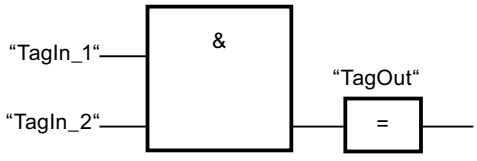
Parameters

The following table shows the parameters of the instruction "AND logic operation":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	The operand indicates the bit whose signal state will be queried.

Example

The following example shows how the instruction works:



Output "TagOut" is set, when the signal state of the operands "TagIn_1" and "TagIn_2" is "1" and reset when the state of the operands "TagIn_1" and "TagIn_2" is "0".

See also

AND truth table (Page 2896)
Example of detecting the direction of a conveyor belt (Page 4980)
Example of controlling room temperature (Page 4988)
Overview of the valid data types (Page 2239)
Adding additional inputs and outputs to FBD elements (Page 4967)
Insert input (Page 2900)
Example of detecting the fill level of a storage area (Page 4982)
Inserting FBD elements (Page 4948)
Editing FBD elements (Page 4964)
Inserting operands in FBD instructions (Page 4970)
FBD programming examples (Page 4978)

AND truth table (S7-1200, S7-1500)

Results of the logic operation

The following table shows the results that arise from the AND logic operation of two operands:

Signal state of the first operand	Signal state of the second operand	Result of the logic operation
1	1	1
0	1	0
1	0	0
0	0	0

See also

&: AND logic operation (Page 2895)
Overview of the valid data types (Page 2239)
Inserting FBD elements (Page 4948)
Editing FBD elements (Page 4964)
Inserting operands in FBD instructions (Page 4970)

>=1: OR logic operation (S7-1200, S7-1500)

Description

You can use the instruction "OR logic operation" to query the signal states of two or more specified operands and evaluate them according to the OR truth table.

If the signal state of one of the operands is "1", then the condition is fulfilled and the instruction returns the result "1". If the signal state of all the operands is "0", then the condition is not fulfilled and the instruction generates the result "0".

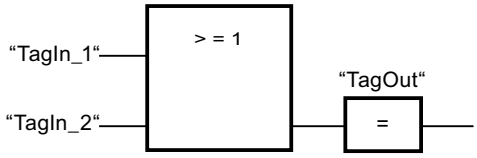
Parameters

The following table shows the parameters of the instruction "OR logic operation":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	The operand indicates the bit whose signal state will be queried.

Example

The following example shows how the instruction works:



Output "TagOut" is set, when the signal state of the operands "TagIn_1" or "TagIn_2" is "1".

See also

OR truth table (Page 2898)

Example of controlling a conveyor belt (Page 4978)

Adding additional inputs and outputs to FBD elements (Page 4967)

Overview of the valid data types (Page 2239)

Insert input (Page 2900)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

FBD programming examples (Page 4978)

OR truth table (S7-1200, S7-1500)

Results of the logic operation

The following table shows the results that arise from the OR logic operation of two operands:

Signal state of the first operand	Signal state of the second operand	Result of the logic operation
1	0	1
0	1	1
1	1	1
0	0	0

See also

>=1: OR logic operation (Page 2896)

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

X: EXCLUSIVE OR logic operation (S7-1200, S7-1500)

Description

You can use the "EXCLUSIVE OR logic operation" instruction to query the result of a signal state query according to the EXCLUSIVE OR truth table.

With an "EXCLUSIVE OR logic operation" instruction, the signal state is "1" when the signal state of one of the two specified operands is "1". When more than two operands are queried, the common RLO is "1" if an odd number of the queried operands returns the result "1".

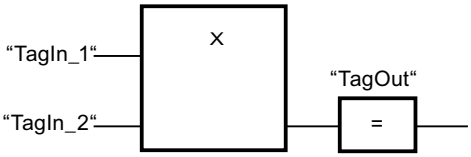
Parameters

The following table shows the parameters of the instruction "EXCLUSIVE OR logic operation":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	The operand indicates the bit whose signal state will be queried.

Example

The following example shows how the instruction works:



Output "TagOut" is set when the signal state of the operands "TagIn_1" and "TagIn_2" is "1". When both operands return the signal state "1" or "0", the output "TagOut" is reset.

See also

EXCLUSIVE OR truth table (Page 2899)

Adding additional inputs and outputs to FBD elements (Page 4967)

Overview of the valid data types (Page 2239)

Insert input (Page 2900)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

EXCLUSIVE OR truth table (S7-1200, S7-1500)

Results of the logic operation

The following table shows the results that arise from the EXCLUSIVE OR logic operation of two operands:

Signal state of the first operand	Signal state of the second operand	Result of the logic operation
1	0	1
0	1	1
1	1	0
0	0	0

The following table shows the results that arise from the EXCLUSIVE OR logic operation of three operands:

Signal state of the first operand	Signal state of the second operand	Signal state of the third operand	Result of the logic operation
1	0	0	1
0	1	1	0
0	1	0	1
1	0	1	0
0	0	1	1

Signal state of the first operand	Signal state of the second operand	Signal state of the third operand	Result of the logic operation
1	1	0	0
1	1	1	1
0	0	0	0

See also

X: EXCLUSIVE OR logic operation (Page 2898)

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

Insert input (S7-1200, S7-1500)

Description

The "Insert input" instruction is used to add an input to the box of one of the following instructions:

- "AND logic operation"
- "OR logic operation"
- "EXCLUSIVE OR logic operation"

You can query the signal state of several operands by the extension of an instruction box.

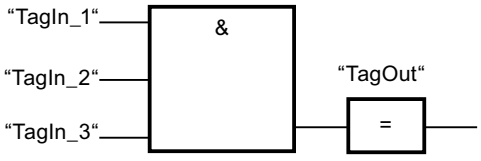
Parameters

The following table shows the parameters of the "Insert input" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	The operand indicates the bit whose signal state will be queried.

Example

The following example shows how the instruction works:



The box of the "AND logic operation" instruction was extended by an additional input at which the signal state of the operand "TagIn_3" is queried. The "TagOut" output is set, when the operands "TagIn_1" and "TagIn_2" and "TagIn_3" return signal state "1".

See also

&: AND logic operation (Page 2895)

>=1: OR logic operation (Page 2896)

X: EXCLUSIVE OR logic operation (Page 2898)

Overview of the valid data types (Page 2239)

Inserting operands in FBD instructions (Page 4970)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

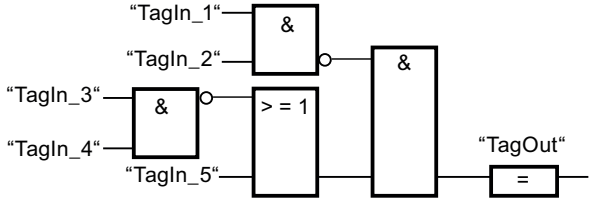
Invert RLO (S7-1200, S7-1500)

Description

You use the "Invert RLO" instruction to invert the signal state of the result of logic operation (RLO).

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The input "TagIn_1" and/or "TagIn_2" has signal state "0".
- The input "TagIn_3" and/or "TagIn_4" has signal state "0" or the input "TagIn_5" has signal state "1".

See also

Overview of the valid data types (Page 2239)

Example of detecting the direction of a conveyor belt (Page 4980)

Example of detecting the fill level of a storage area (Page 4982)

Example of controlling room temperature (Page 4988)

FBD programming examples (Page 4978)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

=: Assignment (S7-1200, S7-1500)

Description

You can use the instruction "Assignment" to set the bit of a specified operand. If the result of logic operation (RLO) at the box input has the signal state "1", the specified operand is set to signal state "1". If the signal state at the box input is "0", the bit of the specified operand is reset to "0".

The instruction does not influence the RLO. The RLO at the box input is assigned directly to the operand above the assignment box.

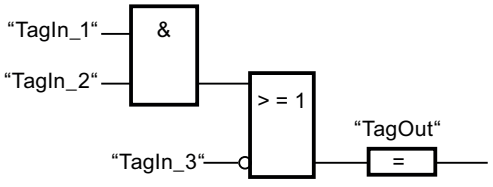
Parameters

The following table shows the parameters of the instruction "Assignment":

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Output	BOOL	I, Q, M, D, L	Operand to which the RLO is assigned.

Example

The following example shows how the instruction works:



The operand "TagOut" is set at the output of the "Assignment" instruction when one of the following conditions is fulfilled:

- The inputs "TagIn_1" and "TagIn_2" have the signal state "1".
- The signal state at the input "TagIn_3" is "0".

See also

Overview of the valid data types (Page 2239)

Example of detecting the fill level of a storage area (Page 4982)

Example of controlling room temperature (Page 4988)

FBD programming examples (Page 4978)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

/=: Negate assignment (S7-1200, S7-1500)

Description

The instruction "Negate assignment" inverts the result of logic operation (RLO) and assigns this to the operand above the box. If the RLO at the input of the box is "1", the binary operand is reset. If the RLO at the input of the box is "0", the operand is set to signal state "1".

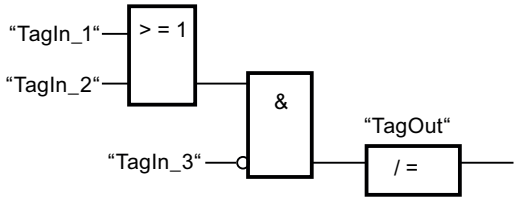
Parameters

The following table shows the parameters of the instruction "Negate assignment":

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Output	BOOL	I, Q, M, D, L	Operand to which the negated RLO is assigned.

Example

The following example shows how the instruction works:



The operand "TagOut" is reset when the following conditions are fulfilled:

- The operand "TagIn_1" or "TagIn_2" has the signal state "1".
- The operand "TagIn_3" has the signal state "0".

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

R: Reset output (S7-1200, S7-1500)

Description

You can use the "Reset output" instruction to reset the signal state of a specified operand to "0".

The instruction is only executed if the result of logic operation (RLO) at the box input is "1". If the box input has the signal state "1", the specified operand is reset to "0". If there is an RLO of "0" at the box input, the signal state of the specified operand remains unchanged.

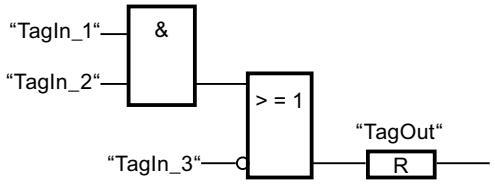
Parameters

The following table shows the parameters of the "Reset output" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand>	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Operand that is reset if RLO = "1".

Example

The following example shows how the instruction works:



Operand "TagOut" is reset when one of the following conditions is fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The operand "TagIn_3" has the signal state "0".

See also

Example of controlling a conveyor belt (Page 4978)

Example of detecting the direction of a conveyor belt (Page 4980)

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

FBD programming examples (Page 4978)

S: Set output (S7-1200, S7-1500)

Description

You can use the instruction "Set output" to set the signal state of a specified operand to "1".

The instruction is only executed if the result of logic operation (RLO) at the box input is "1". If the box input has the signal state "1", the specified operand is set to "1". If there is an RLO of "0" at the box input, the signal state of the specified operand remains unchanged.

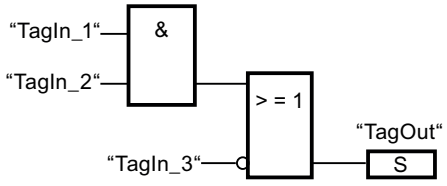
Parameters

The following table shows the parameters of the instruction "Set output":

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Output	BOOL	I, Q, M, D, L	Operand which is set with RLO = "1".

Example

The following example shows how the instruction works:



The "TagOut" operand is set when one of the following conditions is fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The operand "TagIn_3" has the signal state "0".

See also

Overview of the valid data types (Page 2239)

Example of detecting the direction of a conveyor belt (Page 4980)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

FBD programming examples (Page 4978)

SET_BF: Set bit field (S7-1200, S7-1500)

Description

You use the "Set bit field" instruction to set several bits starting from a certain address.

You specify the number of bits to be set with the value of the constant at the N input. The address of the first bit to be set is defined by <Operand>. If the value of <Operand> may not be greater than the number of bits in a selected byte. If this value is greater, the instruction is not executed and the error message "Range violation for index <Operand1>" is displayed. The bits remain set until they are explicitly reset by another instruction.

The instruction is only executed if the enable input EN has the signal state "1".

Bit fields of the type PLC data type, STRUCT or ARRAY

With structures of the type PLC data type, STRUCT or ARRAY, the number of bits contained in the structure represents the maximum number of bits that can be reset:

- If you specify the value "20" at the N input, for example, and the structure only contains 10 bits, then only these 10 bits are set.
- If you specify the value "5" at the N input, for example, and the structure contains 10 bits, then exactly 5 bits are set.

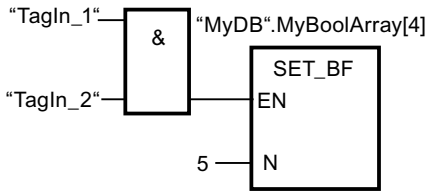
Parameters

The following table shows the parameters of the "Set bit field" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
N	Input	UINT	Constant	Constant	Number of bits to be set
<Operand>	Output	BOOL	I, Q, M With a DB or an IDB, an element of an ARRAY[..] of BOOL	I, Q, M With a DB or an IDB, an element of an ARRAY[..] of BOOL	Pointer to the first bit to be set.

Example

The following example shows how the instruction works:



If the operands "TagIn_1" and "TagIn_2" have the signal state "1", 5 bits are set starting at the address of the operand "MyDB".MyBoolArray[4].

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

RESET_BF: Reset bit field (S7-1200, S7-1500)

Description

You use the "Reset bit field" instruction to reset several bits starting from a certain address.

You specify the number of bits to be reset with the value of the constant at the N input. The address of the first bit to be reset is defined by <Operand>. If the value of the constant may not be greater than the number of bits in a selected byte. If this value is greater, the instruction is not executed and the error message "Range violation for index <Operand 1>" is displayed. The bits remain set until they are explicitly reset by another instruction.

The instruction is only executed if the enable input EN has the signal state "1".

Bit fields of the type PLC data type, STRUCT or ARRAY

With structures of the type PLC data type, STRUCT or ARRAY, the number of bits contained in the structure represents the maximum number of bits that can be set:

- If you specify the value "20" at the N input, for example, and the structure only contains 10 bits, then only these 10 bits are reset.
- If you specify the value "5" at the N input, for example and the structure contains 10 bits, then exactly 5 bits are reset.

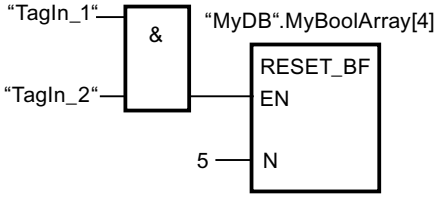
Parameters

The following table shows the parameters of the "Reset bit field" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
N	Input	UINT	Constant	Constant	Number of bits to be reset
<Operand>	Output	BOOL	I, Q, M With a DB or an IDB, an element of an ARRAY[..] of BOOL	I, Q, M With a DB or an IDB, an element of an ARRAY[..] of BOOL	Pointer to the first bit to be reset.

Example

The following example shows how the instruction works:



If the operands "TagIn_1" and "TagIn_2" have the signal state "1", 5 bits are reset starting at the address of the operand "MyDB".MyBoolArray[4].

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

SR: Set/reset flip-flop (S7-1200, S7-1500)

Description

You can use the "Set/reset flip-flop" instruction to set or reset the bit of a specified operand based on the signal state of the inputs S and R1. If the signal state is "1" at input S and "0" at input R1, the specified operand is set to "1". If the signal state is "0" at input S and "1" at input R1, the specified operand will be reset to "0".

Input R1 takes priority over input S. When the signal state is "1" on both inputs S and R1, the signal state of the specified operand is reset to "0".

The instruction is not executed if the signal state at the two inputs S and R1 is "0". The signal state of the operand then remains unchanged.

The current signal state of the operand is transferred to output Q and can be queried there.

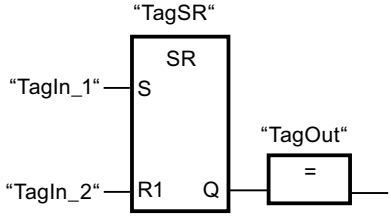
Parameters

The following table shows the parameters of the "Set/reset flip-flop" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
S	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable setting
R1	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable resetting
<Operand>	InOut	BOOL	I, Q, M, D, L	I, Q, M, D, L	Operand that is set or reset
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Signal state of the operand

Example

The following example shows how the instruction works:



The operands "TagSR" and "TagOut" are set when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The operand "TagIn_2" has the signal state "0".

The operands "TagSR" and "TagOut" are reset when one of the following conditions is fulfilled:

- The operand "TagIn_1" has signal state "0" and the operand "TagIn_2" has signal state "1".
- The operands "TagIn_1" and "TagIn_2" have signal state "1".

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

RS: Reset/set flip-flop (S7-1200, S7-1500)

Description

You can use the "Reset/set flip-flop" instruction to reset or set the bit of a specified operand based on the signal state of the inputs R and S1. If the signal state is "1" at input R and "0" at input S1, the specified operand will be reset to "0". If the signal state is "0" at input R and "1" at input S1, the specified operand is set to "1".

Input S1 takes priority over input R. When the signal state is "1" at both inputs R and S1, the signal state of the specified operand is set to "1".

The instruction is not executed if the signal state at the two inputs R and S1 is "0". The signal state of the operand then remains unchanged.

The current signal state of the operand is transferred to output Q and can be queried there.

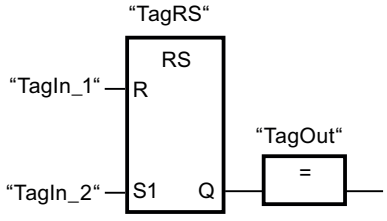
Parameters

The following table shows the parameters of the "Reset/set flip-flop" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
R	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable resetting
S1	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable setting
<Operand>	InOut	BOOL	I, Q, M, D, L	I, Q, M, D, L	Operand that is reset or set.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Signal state of the operand

Example

The following example shows how the instruction works:



The operands "TagRS" and "TagOut" are reset when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The operand "TagIn_2" has the signal state "0".

The operands "TagRS" and "TagOut" are set when the following conditions are fulfilled:

- The operand "TagIn_1" has signal state "0" and the operand "TagIn_2" has signal state "1".
- The operands "TagIn_1" and "TagIn_2" have signal state "1".

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

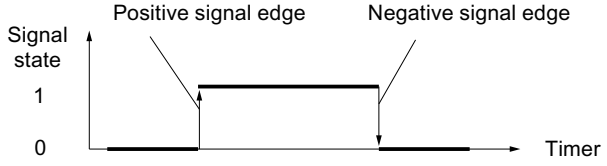
Inserting operands in FBD instructions (Page 4970)

P: Scan operand for positive signal edge (S7-1200, S7-1500)

Description

You can use the "Scan operand for positive signal edge" instruction to determine whether there is a "0" to "1" change in the signal state of a specified operand (<Operand1>). The instruction compares the current signal state of <Operand1> with the signal state of the previous scan, which is saved in an edge memory bit (<Operand2>). If the instruction detects a change in the result of logic operation (RLO) from "0" to "1", there is a positive, rising edge.

The following figure shows the change in the signal state in case of a negative and a positive signal edge:



The positive signal edge is queried each time the instruction executes. When a positive signal edge is detected, <Operand1> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

Specify the operand to be queried (<Operand1>) in the operand placeholder above the instruction. Specify the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This would influence edge evaluation and the result would no longer be unequivocal. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

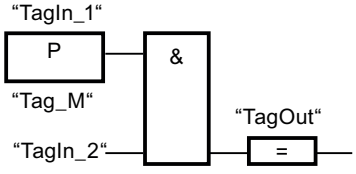
Parameters

The following table shows the parameters of the "Scan operand for positive signal edge" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand1>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Signal to be scanned
<Operand2>	InOut	BOOL	I, Q, M, D, L	I, Q, M, D, L	Edge memory bit in which the signal state of the previous scan is saved.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- There is a rising edge at input "TagIn_1".
- The signal state of the operand "TagIn_2" is "1".

See also

Overview of the valid data types (Page 2239)

Example of detecting the direction of a conveyor belt (Page 4980)

FBD programming examples (Page 4978)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

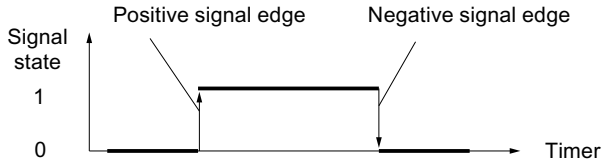
Inserting operands in FBD instructions (Page 4970)

N: Scan operand for negative signal edge (S7-1200, S7-1500)

Description

You can use the "Scan operand for negative signal edge" instruction to determine whether there is a "1" to "0" change in the signal state of a specified operand (<Operand1>). The instruction compares the current signal state of <Operand1> with the signal state of the previous scan, which is saved in an edge memory bit (<Operand2>). If the instruction detects a change in the result of logic operation (RLO) from "1" to "0", there is a negative, falling edge.

The following figure shows the change in the signal state in case of a negative and a positive signal edge:



The negative signal edge is queried each time the instruction executes. When a negative signal edge is detected, <Operand1> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

Specify the operand to be queried (<Operand1>) in the operand placeholder above the instruction. Specify the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This would influence edge evaluation and the result would no longer be unequivocal. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

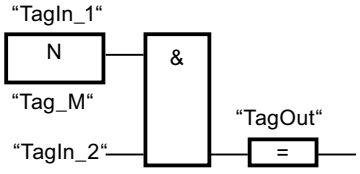
Parameters

The following table shows the parameters of the "Scan operand for negative signal edge" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
<Operand1>	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Signal to be scanned
<Operand2>	InOut	BOOL	I, Q, M, D, L	I, Q, M, D, L	Edge memory bit in which the signal state of the previous scan is saved.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- There is a falling edge at input "TagIn_1".
- The signal state of the operand "TagIn_2" is "1".

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Inserting operands in FBD instructions (Page 4970)

Editing FBD elements (Page 4964)

P=: Set operand on positive signal edge (S7-1200, S7-1500)

Description

You can use the instruction "Set operand on positive signal edge" to set a specified operand (<Operand2>) when there is a "0" to "1" change in the result of logic operation (RLO). The instruction compares the current RLO with the RLO from the previous query, which is saved in the edge memory bit (<Operand1>). If the instruction detects a change in the RLO from "0" to "1", there is a positive signal edge.

The positive signal edge is queried each time the instruction executes. When a positive edge is detected, <Operand2> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

You specify the operand (<Operand2>) to be set in the operand placeholder above the instruction. Specify the edge memory bit (<Operand1>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This would influence edge evaluation and the result would no longer be unequivocal. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

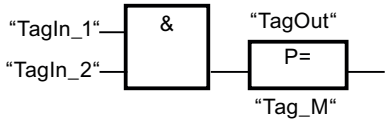
Parameters

The following table shows the parameters of the instruction "Set operand on positive signal edge":

Parameters	Declaration	Data type	Memory area	Description
<Operand2>	Output	BOOL	I, Q, M, D, L	Operand which is set when there is a positive signal edge.
<Operand1>	InOut	BOOL	I, Q, M, D, L	Edge memory bit

Example

The following example shows the parameters of the instruction:



The "TagOut" output is set for one program cycle, when the signal state at the input of the instruction box switches from "0" to "1" (positive signal edge). In all other cases, the "TagOut" output has signal state "0".

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

N=: Set operand on negative signal edge (S7-1200, S7-1500)

Description

You can use the instruction "Set operand on negative signal edge" to set a specified operand (<Operand1>) when there is a "1" to "0" change in the result of logic operation (RLO). The instruction compares the current RLO with the RLO from the previous query, which is saved in the edge memory bit (<Operand2>). If the instruction detects a change in the RLO from "1" to "0", there is a negative signal edge.

The negative signal edge is queried each time the instruction executes. When a negative signal edge is detected, <Operand1> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

You specify the operand (<Operand1>) to be set in the operand placeholder above the instruction. Specify the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This would influence edge evaluation and the result would no longer be unequivocal. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

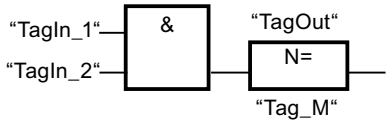
Parameters

The following table shows the parameters of the instruction "Set operand on negative signal edge":

Parameters	Declaration	Data type	Memory area	Description
<Operand1>	Output	BOOL	I, Q, M, D, L	Operand which is set when there is a negative signal edge.
<Operand2>	InOut	BOOL	I, Q, M, D, L	Edge memory bit

Example

The following example shows how the instruction works:



The operand "TagOut" is set for one program cycle if the signal state at the input of the instruction box changes from "1" to "0" (negative signal edge). In all other cases, the operand "TagOut" has the signal state "0".

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Inserting operands in FBD instructions (Page 4970)

Editing FBD elements (Page 4964)

P_TRIG: Scan RLO for positive signal edge (S7-1200, S7-1500)

Description

Use the instruction "Scan RLO for positive signal edge" to query a "0" to "1" change in the signal state of the result of logic operation (RLO). The instruction compares the current signal state of the RLO with the signal state of the previous query, which is saved in an edge memory bit (<operand>). If the instruction detects a change in the RLO from "0" to "1", there is a positive signal edge.

The positive signal edge is queried each time the instruction executes. As soon as a positive single edge is detected, the output Q of the instruction returns the signal state "1" for the length of a program cycle. In all other cases, the output returns the signal state "0".

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This would influence edge evaluation and the result would no longer be unequivocal. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

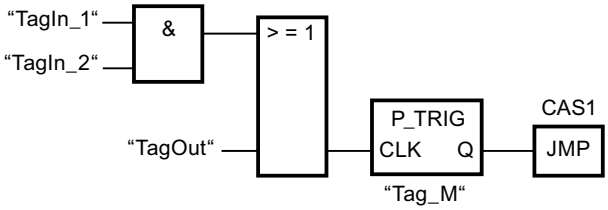
Parameters

The following table shows the parameters of the instruction "Scan RLO for positive signal edge":

Parameter	Declaration	Data type	Memory area	Description
CLK	Input	BOOL	I, Q, M, D, L	Current RLO
<Operand>	InOut	BOOL	M, D	Edge memory bit in which the RLO of the previous query is saved.
Q	Output	BOOL	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:



The RLO of the preceding bit logic operation is saved in the edge memory bit "Tag_M". If a "0" to "1" change is detected in the signal state of the RLO, the program jumps to jump label CAS1.

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

N_TRIG: Scan RLO for negative signal edge (S7-1200, S7-1500)

Description

Use the instruction "Scan RLO for negative signal edge" to query a "1" to "0" change in the signal state of the result of logic operation (RLO). The instruction compares the current signal state of the RLO with the signal state of the previous query saved in the edge memory bit (<Operand>). If the instruction detects a change in the RLO from "1" to "0", there is a negative signal edge.

The negative signal edge is queried each time the instruction executes. As soon as a negative single edge is detected, the output Q of the instruction returns the signal state "1" for the length of a program cycle. In all other cases, the signal state at the output of the instruction is "0".

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This would influence edge evaluation and the result would no longer be unequivocal. The memory area of the edge memory bit has to be located in a DB (static area for FB) or in the bit memory area.

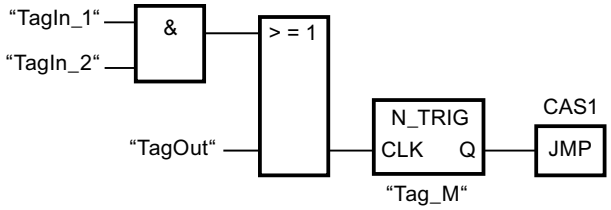
Parameters

The following table shows the parameters of the "Scan RLO for negative signal edge" instruction:

Parameter	Declaration	Data type	Memory area	Description
CLK	Input	BOOL	I, Q, M, D, L	Current RLO
<Operand>	InOut	BOOL	M, D	Edge memory bit in which the RLO of the previous query is saved.
Q	Output	BOOL	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:



The RLO of the preceding bit logic operation is saved in the edge memory bit "Tag_M". If a "1" to "0" change is detected in the signal state of the RLO, the program jumps to jump label CAS1.

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

R_TRIG: Detect positive signal edge (S7-1200, S7-1500)

Description

With the "Detect positive signal edge" instruction, you can detect a state change from "0" to "1" at the CLK input. The instruction compares the current value at the CLK input with the state of the previous query (edge memory bit) that is saved in the specified instance. If the instruction detects a state change at the CLK input from "0" to "1", a positive signal edge is generated at the Q output, i.e., the output has the value TRUE or "1" for exactly one cycle.

In all other cases, the signal state at the output of the instruction is "0".

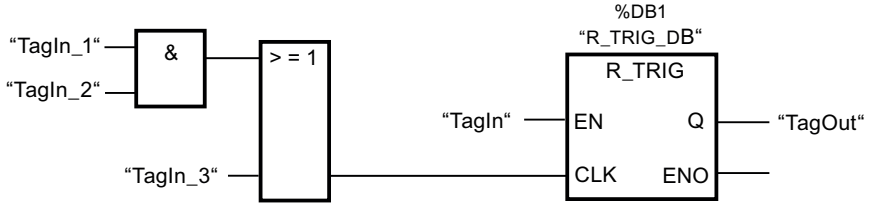
Parameters

The following table shows the parameters of the "Detect positive signal edge" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
CLK	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L, T, C or constant	Incoming signal, the edge of which is to be queried.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:



The previous state of the tag at the CLK input is stored in the "R_TRIG_DB" tag. If a change in the signal state from "0" to "1" is detected in the "TagIn_1" and "TagIn_2" operands or in the "TagIn_3" operand, the "TagOut_Q" output has signal state "1" for one cycle.

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

F_TRIG: Detect negative signal edge (S7-1200, S7-1500)

Description

With the "Detect negative signal edge" instruction, you can detect a state change from "1" to "0" at the CLK input. The instruction compares the current value at the CLK input with the state of the previous query (edge memory bit) that is saved in the specified instance. If the instruction detects a state change at the CLK input from "1" to "0", a negative signal edge is generated at the Q output, i.e., the output has the value TRUE or "1" for exactly one cycle.

In all other cases, the signal state at the output of the instruction is "0".

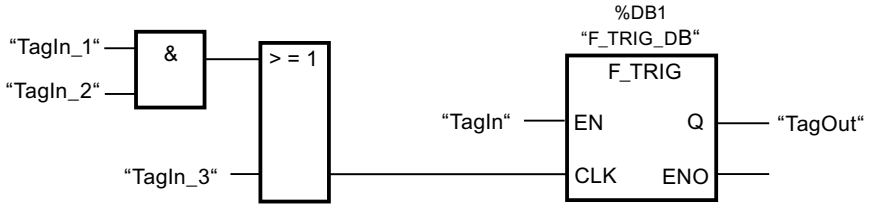
Parameters

The following table shows the parameters of the "Detect negative signal edge" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
CLK	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L, T, C or constant	Incoming signal, the edge of which is to be queried.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:



The previous state of the tag at the CLK input is stored in the "F_TRIG_DB" tag. If a change in the signal state from "1" to "0" is detected in the "TagIn_1" and "TagIn_2" operands or in the "TagIn_3" operand, the "TagOut_Q" output has signal state "1" for one cycle.

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

Timer operations (S7-1200, S7-1500)

TP: Generate pulse (S7-1200, S7-1500)

Description

You can use the "Generate pulse" instruction to set the Q output for a programmed duration. The instruction is started when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive signal edge). The programmed time PT begins when the instruction starts. Output Q is set for the duration PT, regardless of the subsequent course of the input signal. Even when a new positive signal edge is detected, the signal state of the Q output is not affected as long as the PT duration is running.

You can query the current time value at the ET output. The timer value starts at T#0s and ends when the value of the time duration PT is reached. If the configured time duration PT is reached and the signal state at input IN is "0", the ET output is reset.

Each call of the "Generate pulse" instruction must be assigned to an IEC timer in which the instruction data is stored.

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

For S7-1200 CPU

An IEC timer is a structure of the data type IEC_TIMER or TP_TIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TP_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

An IEC Timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TP_TIME or TP_LTIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TP_TIME, TP_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- When the instruction is called, if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

The execution of the "Generate pulse" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

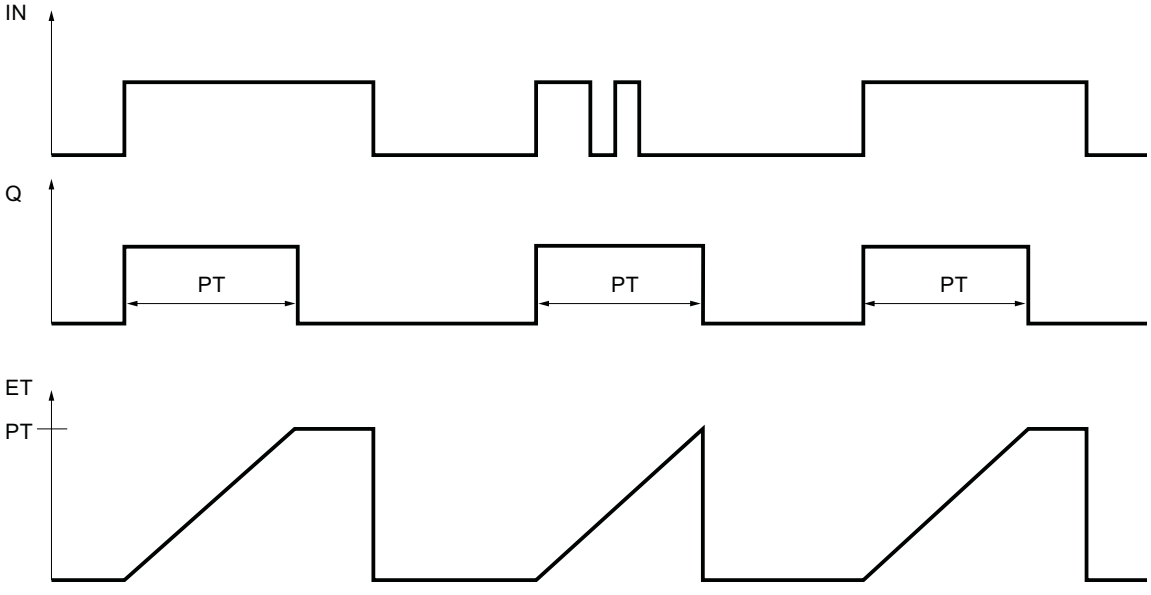
Parameters

The following table shows the parameters of the "Generate pulse" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Duration of the pulse. The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Pulse output
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L	I, Q, M, D, L, P	Current timer value

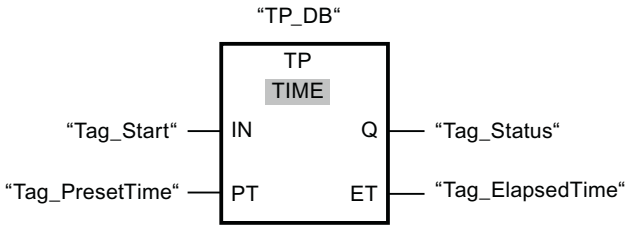
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Generate pulse" instruction:



Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Start	Signal transition "0" => "1"
PT	Tag_PresetTime	T#10s
Q	Tag_Status	TRUE
ET	Tag_ElapsedTime	from T#0s => T#10s

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started and the "Tag_Status" operand is set to "1". The current time value is stored in the "Tag_ElapsedTime" operand. When the time expires, the "Tag_Status" operand is reset to signal state "0".

See also

Overview of the valid data types (Page 2239)

Example of controlling room temperature (Page 4988)

FBD programming examples (Page 4978)

Basic information on FBD (Page 4937)

TON: Generate on-delay (S7-1200, S7-1500)

Description

You can use the instruction "Generate on-delay" to delay setting of the Q output by the time configured with the PT time. The instruction is started when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive signal edge). The programmed time PT begins when the instruction starts. When the time PT has elapsed, the output Q has the signal state "1". Output Q remains set as long as the start input is still "1". When the signal state at the start input changes from "1" to "0", the Q output is reset. The timer function is started again when a new positive signal edge is detected at the start input.

The current time value can be queried at the output ET. The timer value starts at T#0s and ends when the value of the time duration PT is reached. The ET output is reset as soon as the signal state at the IN input changes to "0".

Each call of the "Generate on-delay" instruction must be assigned to an IEC timer in which the instruction data is stored.

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

For S7-1200 CPU

An IEC Timer is a structure of the data type IEC_TIMER or TON_TIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TON_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

An IEC Timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TON_TIME or TON_LTIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TON_TIME, TON_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- When the instruction is called if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

The execution of the "Generate on-delay" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

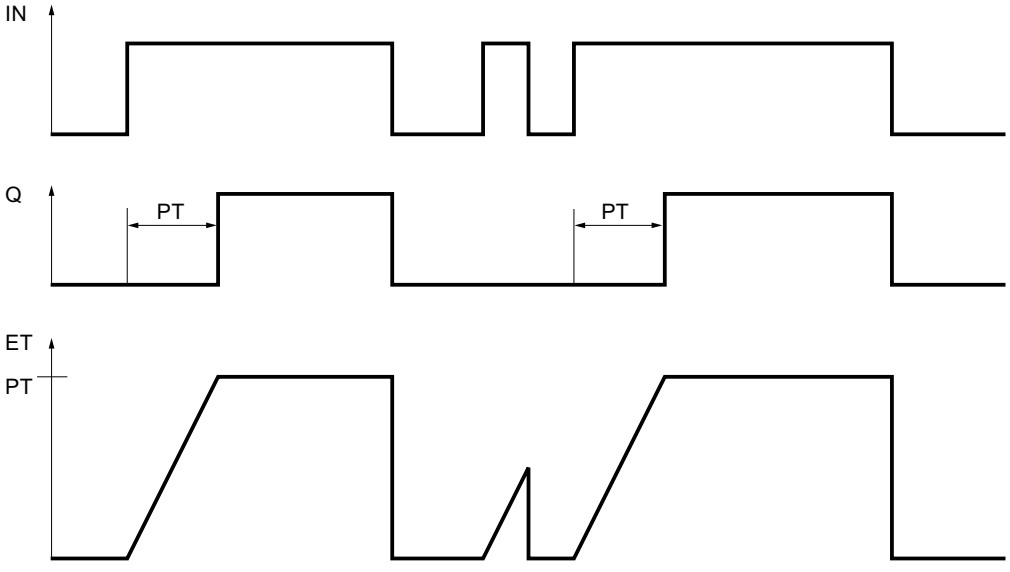
Parameters

The following table shows the parameters of the "Generate on-delay" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Duration of the on delay. The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Output that is set when the time PT expires.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L	I, Q, M, D, L, P	Current timer value

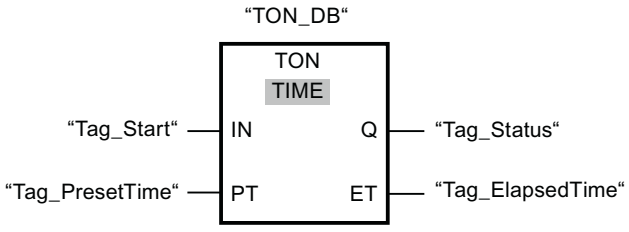
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Generate on-delay" instruction:



Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Start	Signal transition "0" => "1"
PT	Tag_PresetTime	T#10s
Q	Tag_Status	FALSE; after 10 s => TRUE
ET	Tag_ElapsedTime	from T#0s => T#10s

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started. When the time duration expires, the "Tag_Status" operand is set to the signal state "1". The Tag_Status operand remains set to "1" as long as the Tag_Start operand has signal state "1". The current time value is stored in the "Tag_ElapsedTime" operand. When the signal state at the operand "Tag_Start" changes from "1" to "0", the "Tag_Status" operand is reset.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

TOF: Generate off-delay (S7-1200, S7-1500)

Description

You can use the "Generate off-delay" instruction to delay setting of the Q output by the time configured with the PT time. The output Q is set when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive signal edge). When the signal state at input IN changes back to "0" (positive signal edge), the configured time duration PT starts. Output Q remains set as long as the time duration PT is running. When the PT time duration expires, the Q output is reset. If the signal state at input IN changes to "1" before the PT time duration expires, the timer is reset. The signal state at the output Q continues to be "1".

The current time value can be queried at the output ET. The timer value starts at T#0s and ends when the value of the time duration PT is reached. When the PT time duration expires, the ET output remains set to the current value until input IN changes back to "1". If input IN changes to "1" before the time duration PT has expired, the ET output is reset to the value T#0s.

Each call of the "Generate off-delay" instruction must be assigned to an IEC Timer in which the instruction data is stored.

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

For S7-1200 CPU

An IEC Timer is a structure of the data type IEC_TIMER or TOF_TIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TOF_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

An IEC Timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TOF_TIME or TOF_LTIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TOF_TIME, TOF_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- When the instruction is called if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

The execution of the "Generate off-delay" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

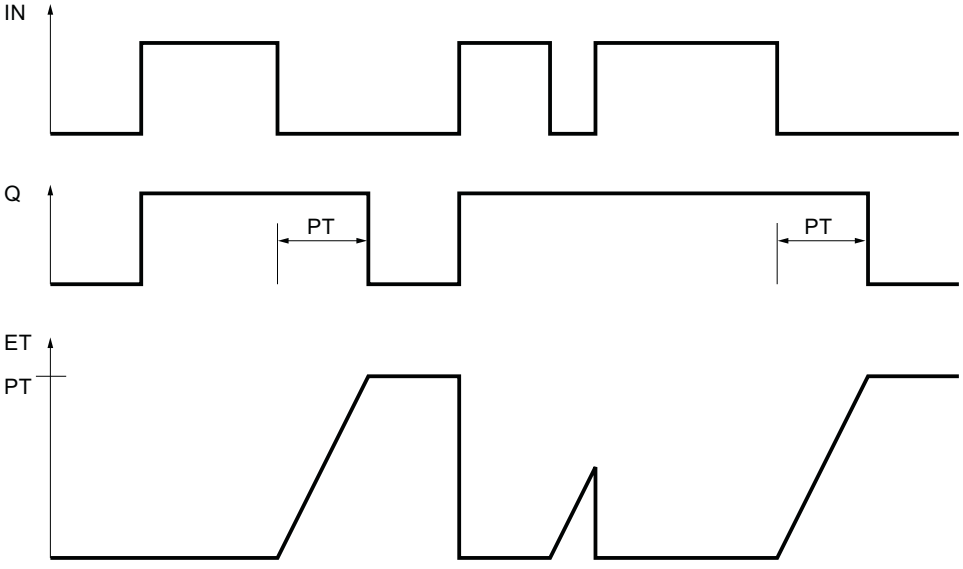
Parameters

The following table shows the parameters of the "Generate off-delay" instruction:

Parameter	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Duration of the off delay The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Output that is reset when the timer PT expires.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L	I, Q, M, D, L, P	Current timer value

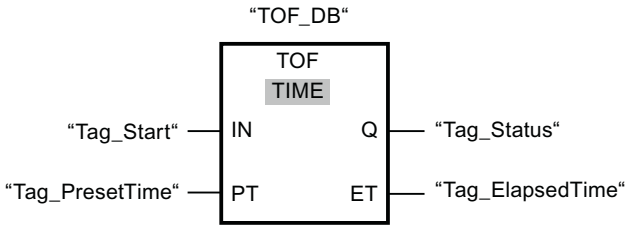
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Generate off-delay" instruction:



Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Start	Signal transition "0" => "1"; signal transition "1" => "0"
PT	Tag_PresetTime	T#10s
Q	Tag_Status	TRUE
ET	Tag_ElapsedTime	from T#10s => T#0s

When the signal state of the operand "Tag_Start" changes from "0" to "1", the "Tag_Status" operand is set to signal state "1". When the signal state of the "Tag_Start" operand changes from "1" to "0", the time programmed for the PT parameter is started. As long as the time is running, the "Tag_Status" operand remains set to TRUE. When the time has expired, the "Tag_Status" operand is reset to FALSE. The current time value is stored in the "Tag_ElapsedTime" operand.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

TONR: Time accumulator (S7-1200, S7-1500)

Description

The instruction "Time accumulator" is used to accumulate time values within a period set by the parameter PT. The instruction is executed and the configured time duration PT is started when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive edge). While the time PT is running, the time values are accumulated that are recorded when the IN input has signal state "1". The accumulated time is written to the ET output and can be queried there. When the current time value PT is reached, the output Q has the signal state "1". Output Q remains set at "1", even when the signal state at input IN changes to "0".

The R input resets the outputs ET and Q regardless of the signal state at the start input.

Each call of the "Time accumulator" instruction must be assigned an IEC timer in which the instruction data is stored.

For S7-1200 CPU

An IEC Timer is a structure of the data type IEC_TIMER or TONR_TIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TONR_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

An IEC timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TONR_TIME or TONR_LTIME that you can declare as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TONR_TIME, TONR_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- When the instruction is called, if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

The execution of the Time accumulator" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

Parameters

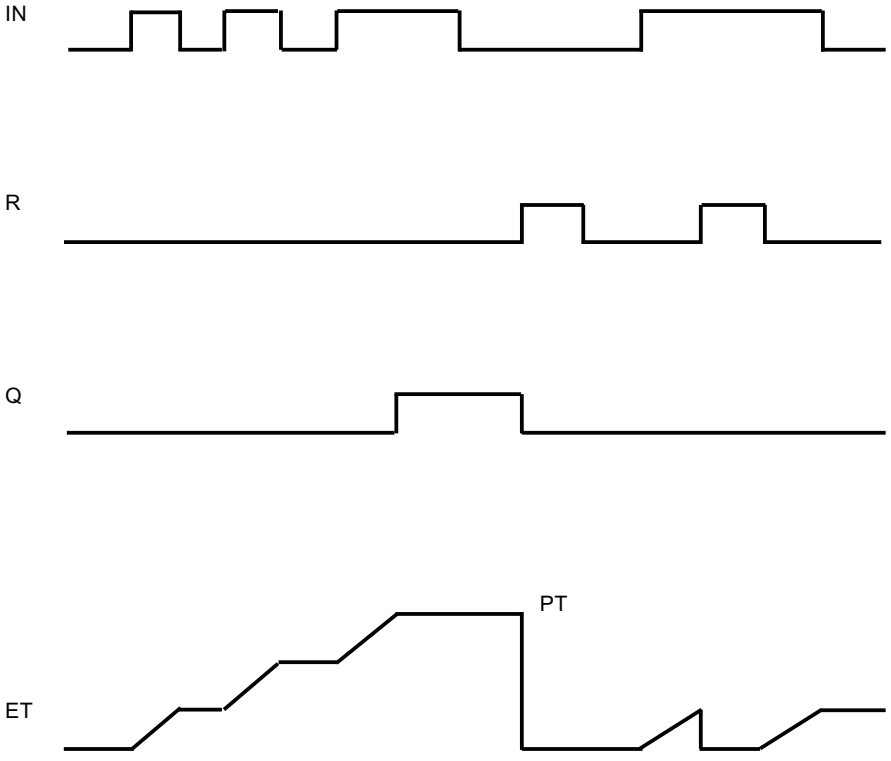
The following table shows the parameters of the "Time accumulator" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C, P	Start input
R	Input	BOOL	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Reset input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	I, Q, M, D, L, P or constant	Maximum duration of time recording. The value of the PT parameter must be positive.

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
Q	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, P	Output that is set when time PT expires.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L	I, Q, M, D, L, P	Current timer value

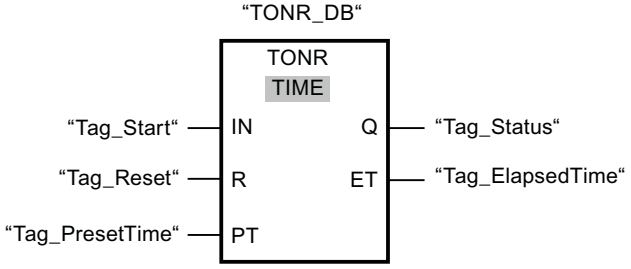
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Time accumulator" instruction:



Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Start	Signal transition "0" => "1"
PT	Tag_PresetTime	T#10s
Q	Tag_Status	FALSE; after 10 s => TRUE
ET	Tag_ElapsedTime	Signal transition from "0" => "1" The time T#10s expires. After 5 s signal transition from "1" => "0": The time at the "Tag_Elapsed-Time" operand remains at T#5s. After approx. 2 s, renewed signal transition from "0" => "1": The time at the "Tag_Elapsed-Time" operand continues to run at T#5s.

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started. The time continues to run as long as the "Tag_Start" operand has the signal state "1". When the signal state of the "Tag_Start" operand changes from "1" to "0", the time stops and the current time value at the Tag_ElapsedTime operand is recorded. When the signal state at the "Tag_Start" operand changes again from "0" to "1", the time continues to run starting at the time value that was recorded at the transition from "1" to "0". When the time value specified for the PT parameter is reached, the "Tag_Status" operand is set to the signal state "1". The current time value is stored in the "Tag_ElapsedTime" operand.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

TP: Start pulse timer (S7-1200, S7-1500)

Description

Use the instruction "Start pulse timer" to start an IEC Timer with a specified duration as pulse. The IEC Timer is started when the result of logic operation (RLO) changes from "0" to "1" (positive signal edge). The IEC Timer runs for the specified time duration regardless of the subsequent course of the RLO. The expiry of the IEC Timer is also not affected by the detection of a new rising edge. As long as the IEC Timer is running, the querying of the timer status for "1" returns the signal state "1". When the IEC Timer has expired, the timer status returns the signal state "0".

Note

The start and the query of the IEC Timer may be on different expiry levels as each query of the outputs Q or ET updates the IEC_TIMER structure.

For S7-1200 CPU

The instruction "Start pulse timer" stores its data in a structure of the data type IEC_TIMER or TP_TIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TP_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

The instruction "Start pulse timer" stores its data in a structure of the data type IEC_TIMER, IEC_LTIMER, TP_TIME or TP_LTIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TP_TIME, TP_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- The IEC_Timer structure is updated when the instruction is called. The time value at the ET output is only updated if the ET or Q outputs are scanned (e.g. "MyTimer".Q or "MyTimer".ET).
- When the assigned timer is accessed.

The current timer status is saved in the structure components Q of the IEC Timer. You can query the timer status with the help of a binary logic operation.

The execution of the instruction "Start pulse timer" assumes a preceding logic operation. It can be placed only at the end of the network.

Parameters

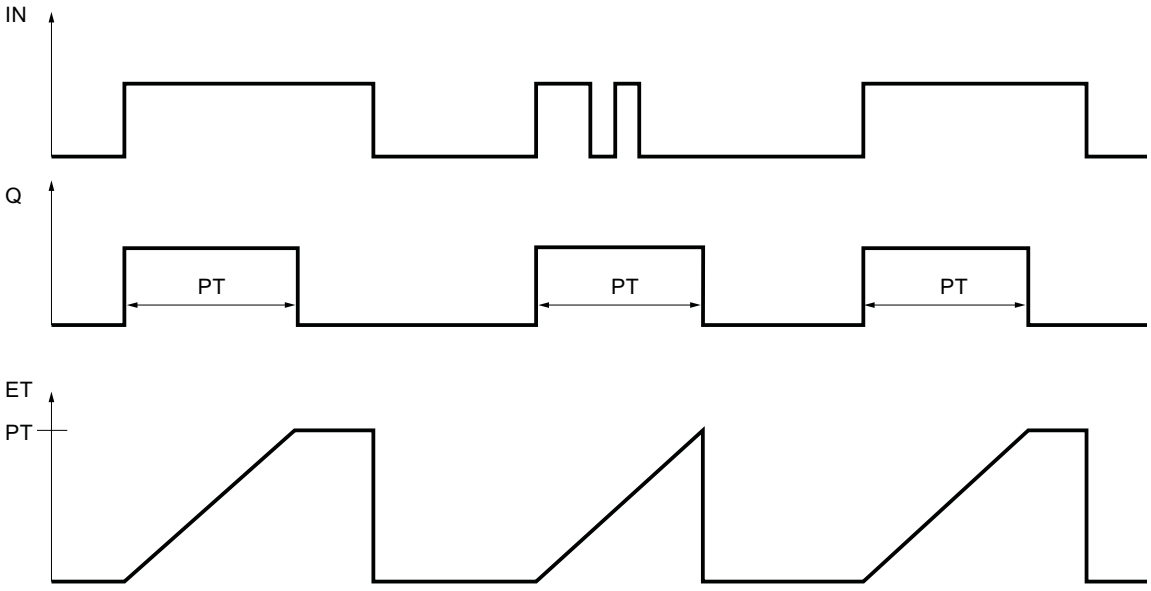
The following table shows the parameters of the instruction "Start pulse timer":

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
VALUE	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC Timer runs.
<IEC Timer>	InOut	IEC_TIMER, TP_TIME	IEC_TIMER, IEC_LTIMER, TP_TIME, TP_LTIME	D, L	IEC Timer which is started.

You can find additional information on valid data types under "See also".

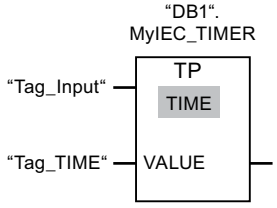
Pulse timing diagram

The following figure shows the pulse timing diagram of the instruction:

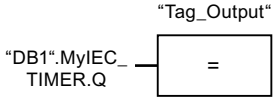


Example

The following example shows how the instruction works:



The instruction "Start pulse timer" is executed when the signal state of the operand "Tag_Input" changes from "0" to "1". The timer "DB1".MyIEC_TIMER is started for the time stored in the operand "TagTime".



As long as the timer "DB1".MyIEC_TIMER is running, the timer status ("DB1".MyIEC_TIMER.Q) has signal state "1" and the operand "Tag_Output" is set. When the IEC Timer has expired, the signal state of the time status changes back to "0" and the "Tag_Output" operand is reset.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

TON: Start on-delay timer (S7-1200, S7-1500)

Description

Use the "Start on-delay timer" instruction to start an IEC Timer with a specified duration as on-delay. The IEC Timer is started when the result of logic operation (RLO) changes from "0" to "1" (positive signal edge). The IEC Timer runs for the specified time duration. The output returns the signal state "1" if the RLO at the input of the instruction has the signal state "1". If the RLO changes to "0" before the end of the timer, the running IEC Timer is reset. The query of the timer status for "1" returns the signal state "0". The IEC Timer restarts when the next rising signal edge is detected at the input of the instruction.

Note

The start and the query of the IEC Timer may be on different expiry levels as each query of the outputs Q or ET updates the IEC_TIMER structure.

For S7-1200 CPU

The instruction "Start on-delay timer" stores its data in a structure of the data type IEC_TIMER or TON_TIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TON_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

The "Start on-delay timer" instruction stores its data in a structure of the data type IEC_TIMER, IEC_LTIMER, TON_TIME or TON_LTIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TON_TIME, TON_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- The IEC_Timer structure is updated when the instruction is called. The time value at the ET output is only updated if the ET or Q outputs are scanned (e.g. "MyTimer".Q or "MyTimer".ET).
- When the assigned timer is accessed.

The current timer status is saved in the structure components ET of the IEC Timer. You can query the timer status with the help of a binary logic operation.

The execution of the "Start on-delay timer" instruction assumes a preceding logic operation. It can be placed only at the end of the network.

Parameters

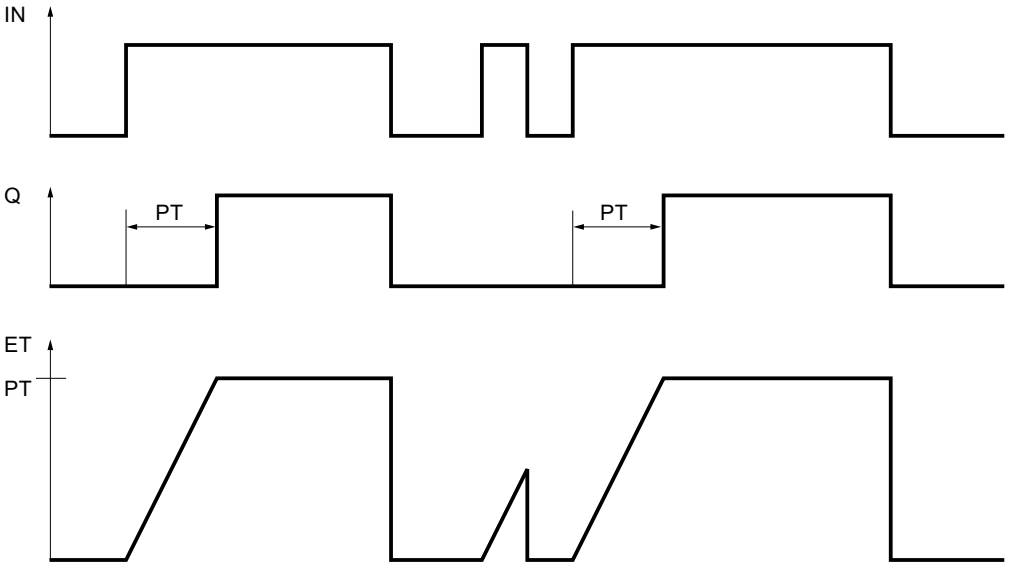
The following table shows the parameters of the instruction "Start on-delay timer":

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
VALUE	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC Timer runs.
<IEC Timer>	InOut	IEC_TIMER, TON_TIME	IEC_TIMER, IEC_LTIMER, TON_TIME, TON_LTIME	D, L	IEC Timer which is started.

You can find additional information on valid data types under "See also".

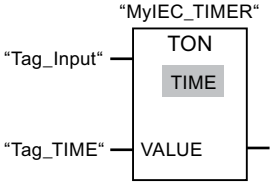
Pulse timing diagram

The following figure shows the pulse timing diagram of the instruction:

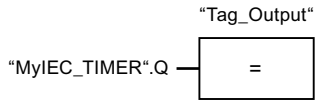


Example

The following example shows how the instruction works:



The "Start on-delay timer" instruction is executed when the signal state of the operand "Tag_Input" changes from "0" to "1". The "MyIEC_TIMER" timer is started for the time stored in the "Tag_TIME" operand.



If the timer "MyIEC_TIMER" has expired and the operand "Tag_Input" has the signal state "1", querying the timer status ("MyIEC_TIMER".Q) returns signal state "1" and the "Tag_Output" operand is set. When the signal state of the operand "Tag_Input" changes to "0", the querying of the timer status returns the signal state "0" and the operand "Tag_Output" is reset.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

TOF: Start off-delay timer (S7-1200, S7-1500)

Description

Use the "Start off-delay timer" instruction to start an IEC Timer with a specified duration as off-delay. The query of the timer status for "1" returns the signal state "0" if the result of the logic operation (RLO) at the input of the instruction has the signal state "1". When the RLO changes from "1" to "0" (negative signal edge), the IEC Timer starts with the specified time duration. The timer status remains at signal state "1" as long as the IEC Timer is running. When the timer has run out and the RLO at the input of the instruction has the signal state "0", the timer status is set to the signal state "0". If the RLO changes to "1" before the end of the timer, the running IEC Timer is reset and the timer status remain at the signal state "1".

Note

The start and the query of the IEC Timer may be on different expiry levels as each query of the outputs Q or ET updates the IEC_TIMER structure.

For S7-1200 CPU

The instruction "Start off-delay timer" stores its data in a structure of the data type IEC_TIMER or TOF_TIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TOF_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

The "Start off-delay timer" instruction stores its data in a structure of the data type IEC_TIMER, IEC_LTIMER, TOF_TIME or TOF_LTIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TOF_TIME, TOF_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- The IEC_Timer structure is updated when the instruction is called. The time value at the ET output is only updated if the ET or Q outputs are scanned (e.g. "MyTimer".Q or "MyTimer".ET).
- When the assigned timer is accessed.

The current timer status is saved in the structure components ET of the IEC Timer. You can query the timer status with the help of a binary logic operation.

The execution of the "Start off-delay timer" instruction assumes a preceding logic operation. It can be placed only at the end of the network.

Parameters

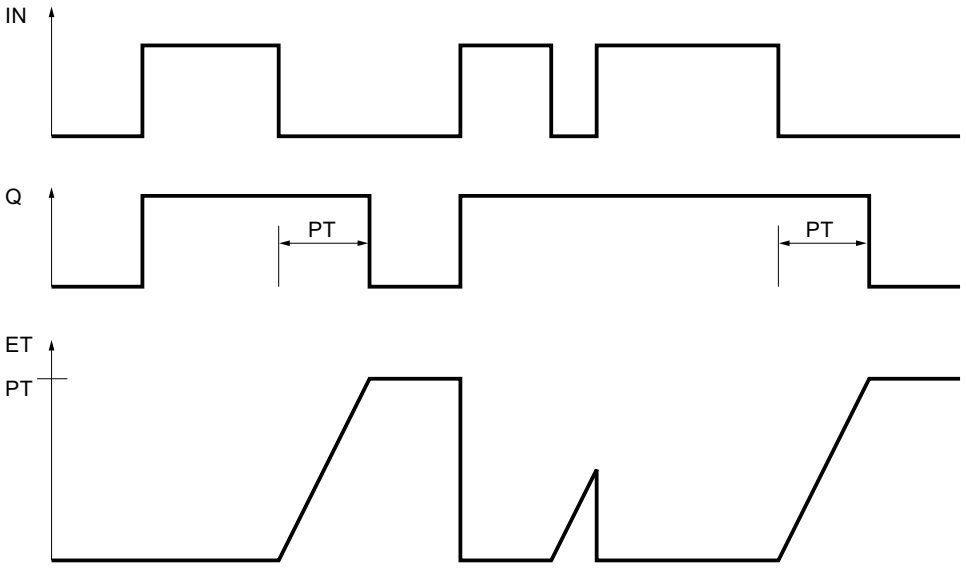
The following table shows the parameters of the instruction "Start off-delay timer":

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
VALUE	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC Timer runs.
<IEC Timer>	InOut	IEC_TIMER, TOF_TIME	IEC_TIMER, IEC_LTIMER, TOF_TIME, TOF_LTIME	D, L	IEC Timer which is started.

You can find additional information on valid data types under "See also".

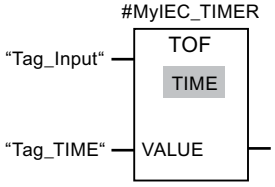
Pulse timing diagram

The following figure shows the pulse timing diagram of the instruction:

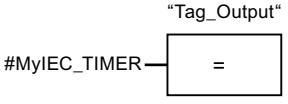


Example

The following example shows how the instruction works:



The "Start off-delay timer" instruction is executed when the signal state of the operand "Tag_Input" changes from "1" to "0". The #MyIEC_TIMER timer is started for the time stored in the operand "Tag_TIME".



As long as timer #MyIEC_TIMER is running, the query of the time status (#MyIEC_TIMER.Q) returns the signal state "1" and operand "Tag_Output" is set. If the timer has expired and the operand "Tag_Input" has the signal state "0", the query of the timer status returns the signal state "0". If the signal state of the operand "Tag_Input" changes to "1" before timer #MyIEC_TIMER expires, the timer is reset. When the signal state of the operand "Tag_Input" is "1", the query of the timer status returns the signal state "1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

TONR: Time accumulator (S7-1200, S7-1500)

Description

You can use the "Time accumulator" instruction to record how long the signal is at the input of instruction "1". The instruction is started when the result of logic operation (RLO) changes from "0" to "1" (positive signal edge). The time is recorded as long at the RLO is "1". If the RLO changes to "0", the instruction is halted. If the RLO changes back to "1", the time recording is continued. The query of the timer status for "1" returns the signal state "1" if the recorded time exceeds the value of the specified time duration and the RLO at the input of coil is "1".

The timer status and the currently expired timer can be reset to "0" using the "Reset timer" instruction.

Note

The start and the query of the IEC Timer may be on different expiry levels as each query of the outputs Q or ET updates the IEC_TIMER structure.

For S7-1200 CPU

The "Time accumulator" instruction stores its data in a structure of the data type IEC_TIMER or TONR_TIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TONR_TIME or IEC_TIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

For S7-1500 CPU

The "Time accumulator" instruction stores its data in a structure of the data type IEC_TIMER, IEC_LTIMER, TONR_TIME or TONR_LTIME. You can declare the structure as follows:

- Declaration of a data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER")
- Declaration as a local tag of the type TONR_TIME, TONR_LTIME, IEC_TIMER or IEC_LTIMER in the "Static" section of a block (for example, #MyIEC_TIMER)

The instruction data is updated in the following cases:

- The IEC_Timer structure is updated when the instruction is called. The time value at the ET output is only updated if the ET or Q outputs are scanned (e.g. "MyTimer".Q or "MyTimer".ET).
- When the assigned timer is accessed.

The current timer status is saved in the structure components ET of the IEC Timer. You can query the timer status with the help of a binary logic operation.

The execution of the "Time accumulator" instruction requires a preceding logic operation. It can be placed only at the end of the network.

Parameters

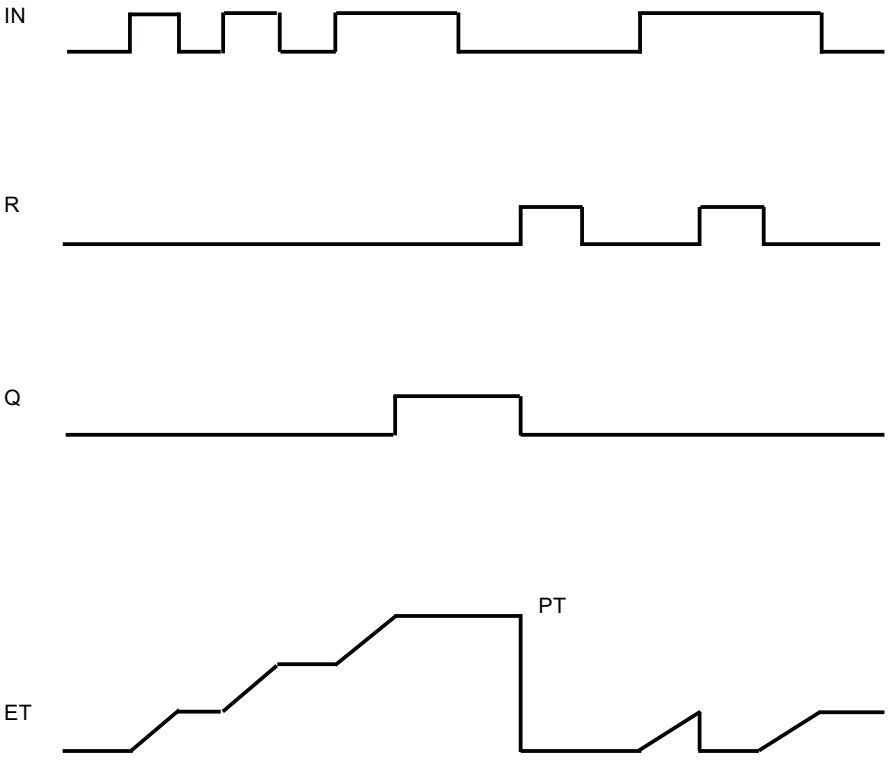
The following table shows the parameters of the "Time accumulator" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
VALUE	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Duration with which the IEC Timer runs.
<IEC Timer>	InOut	IEC_TIMER, TONR_TIME	IEC_TIMER, IEC_LTIMER, TONR_TIME, TONR_LTIME	D, L	IEC Timer which is started.

You can find additional information on valid data types under "See also".

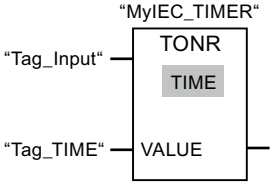
Pulse timing diagram

The following figure shows the pulse timing diagram of the instruction:

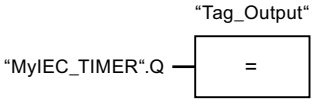


Example

The following example shows how the instruction works:



The "Time accumulator" instruction is executed if there is a positive signal edge in the RLO. The time is recorded as long as the operand "Tag_Input" has the signal state "1".



If the recorded time exceeds the value of the operand "Tag_TIME", then the query of the timer status ("MyIEC_TIMER".Q) will return the signal state "1" and the operand "Tag_Output" will be set.

See also

Overview of the valid data types (Page 2239)

RT: Reset timer (Page 2944)

Basic information on FBD (Page 4937)

RT: Reset timer (S7-1200, S7-1500)

Description

You can use the "Reset timer" instruction to reset an IEC Timer to "0". You specify the IEC Timer to be reset by entering the name of the data block that contains the structure of the IEC Timer in the placeholder above the instruction.

The instruction is only executed if the result of logic operation (RLO) at the box input is "1". When the instruction is executed the structure components of the IEC Timer are reset to "0" in the specified data block. If the RLO at box input is "0", the instruction is not executed.

The instruction does not influence the RLO. The RLO at the box input is transferred directly to the box output.

You must assign a IEC Timer declared in the program to the "Reset timer" instruction.

The instruction data is updated only when the instruction is called and not each time the assigned IEC Timer is accessed. The query of the data is only identical from the call of the instruction to the next call of the instruction.

Parameters

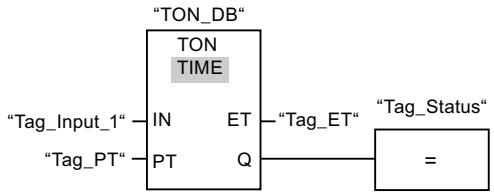
The following table shows the parameters of the instruction "Reset timer":

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<IEC Timer>	InOut	IEC_TIMER, TP_TIME, TON_TIME, TOF_TIME, TONR_TIME	IEC_TIMER, IEC_LTIMER, TP_TIME, TP_LTIME, TON_TIME, TON_LTIME, TOF_TIME, TOF_LTIME, TONR_TIME, TONR_LTIME	D, L	IEC Timer, which is reset.

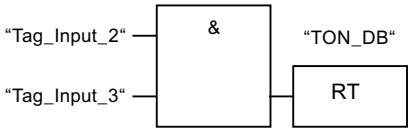
For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The "Generate on-delay" instruction executes when the signal state of the "Tag_Input_1" operand changes from "0" to "1". The IEC Timer stored in the instance data block "TON_DB" is started with the time duration that is specified by the operand "Tag_PT".



If the operands "Tag_Input_2" and "Tag_Input_3" have the signal state "1", the "Reset timer" instruction is executed and the IEC Timer stored in the data block "TON_DB" is reset.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

PT: Load time duration (S7-1200, S7-1500)

Description

Use the "Load time duration" instruction to set the time duration of an IEC Timer. The instruction is executed in every cycle when the result of logic operation (RLO) at the input of the instruction has the signal state "1". The instruction writes the specified time duration to the structure of the specified IEC Timer.

Note

If the specified IEC Timer is running during the execution, the instruction overwrites the current time duration of the specified IEC Timer. As a result, the timer status of the IEC Timer can change.

You must assign a IEC Timer declared in the program to the "Load time duration" instruction.

The instruction data is updated when the instruction is called and each time the assigned IEC Timer is accessed. The query for Q or ET (e. g. "MyTimer".Q or "MyTimer".ET) updates the IEC_TIMER structure.

Parameters

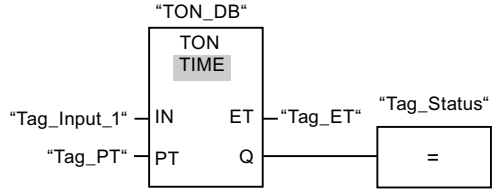
The following table shows the parameters of the instruction "Load time duration":

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L or constant	Time duration
<IEC Timer>	InOut	IEC_TIMER, TP_TIME, TON_TIME, TOF_TIME, TONR_TIME	IEC_TIMER, IEC_LTIMER, TP_TIME, TP_LTIME, TON_TIME, TON_LTIME, TOF_TIME, TOF_LTIME, TONR_TIME, TONR_LTIME	D, L	IEC Timer, the duration of which is set.

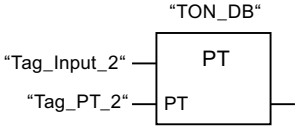
For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The "Generate on-delay" instruction executes when the signal state of the "Tag_Input_1" operand changes from "0" to "1". The IEC Timer stored in the instance data block "TON_DB" is started with the time duration that is specified by the operand "Tag_PT".



The "Load time duration" instruction is executed when the operand "Tag_Input_2" has the signal state "1". The instruction writes the time duration "Tag_PT_2" in the instance data block "TON_DB" and at the same time overwrites the value of the operand "Tag_PT" within the data block. As a result, the signal state of the timer status can change at the next query or upon access to "MyTimer".Q or "MyTimer".ET.

Note

The "Tag_Input_2" is executed as pulse flag in order that the time duration is loaded only throughout one program cycle.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Legacy (S7-1500)

S_PULSE: Assign pulse timer parameters and start (S7-1500)

Description

The "Assign pulse timer parameters and start" instruction starts a programmed timer when a transition from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV) as long as the signal state at input S is "1". If the signal state at input S changes to "0" before the programmed duration expires, the timer is stopped. In this case, the signal state at output Q is "0".

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero. The time base determines the time period of the time value. The current time value is output binary-coded at output BI and BCD-coded at output BCD.

If the timer is running and the signal state at input R changes to "1", the current time value and the time base are also set to zero. If the timer is not running, the signal state "1" at the R input has no effect.

The "Assign pulse timer parameters and start" instruction needs a preceding logic operation for the edge evaluation and can be placed within or at the end of the network.

The instruction data is updated with each access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

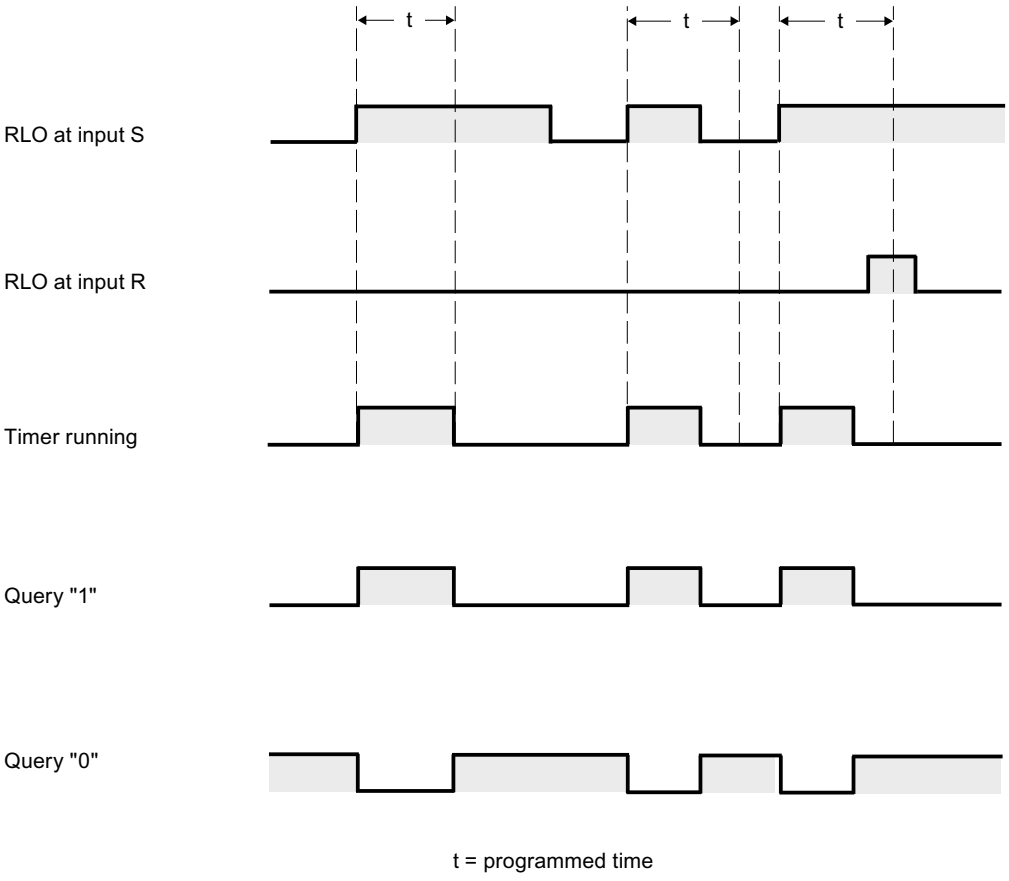
The following table shows the parameters of the "Assign pulse timer parameters and start" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (binary-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer

For additional information on valid data types, refer to "See also".

Pulse timing diagram

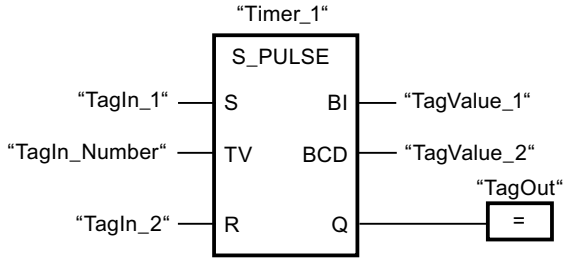
The following figure shows the pulse timing diagram of the "Assign pulse timer parameters and start" instruction:



t = programmed time

Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer runs for the time value of the "TagIn_Number" operand as long as the "TagIn_1" operand has the signal state "1". If the signal state of the operand "TagIn_1" changes from "1" to "0" before the timer expires, the timer "Timer_1" is stopped. The "TagOut" operand is reset to signal state "0".

The "TagOut" operand has the signal state "1" as long as the timer is running and the "TagIn_1" operand has the signal state "1". When the time has expired or is reset, the "TagOut" operand is reset to "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

S_PEXT: Assign extended pulse timer parameters and start (S7-1500)

Description

The "Assign extended pulse timer parameters and start" instruction starts a programmed timer when a transition from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV), even if the signal state at input S changes to "0". As long as the timer is running, output Q has the signal state "1". When the timer has expired, output Q is reset to "0". If the signal state at input S changes from "0" to "1" while the timer is running, the timer is restarted with the duration programmed at input TV.

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero. The time base determines the time period of the time value. The current time value is output binary-coded at output BI and BCD-coded at output BCD.

If the timer is running and the signal state at input R changes to "1", the current time value and the time base are also set to zero. If the timer is not running, the signal state "1" at the R input has no effect.

The "Assign extended pulse timer parameters and start" instruction needs a preceding logic operation for the edge evaluation and can be placed within or at the end of the network.

The instruction data is updated with each access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

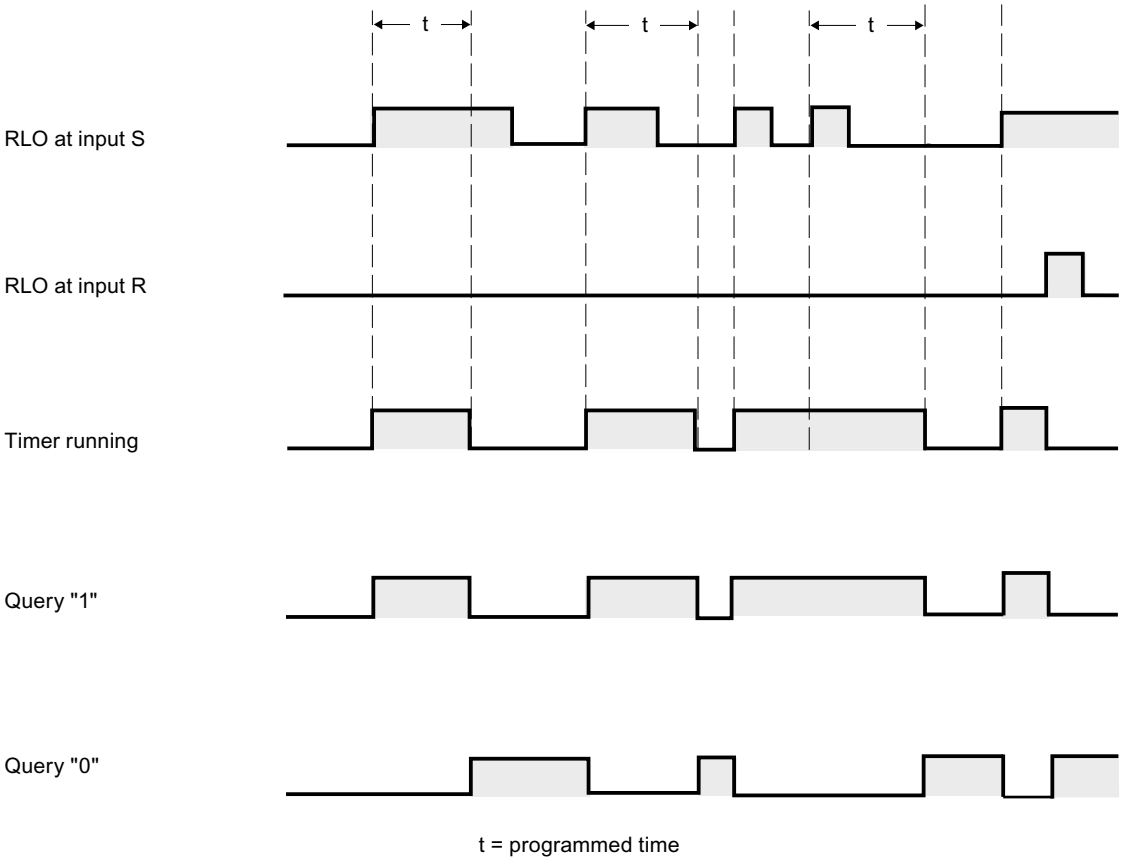
The following table shows the parameters of the "Assign extended pulse timer parameters and start" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (binary-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer

For additional information on valid data types, refer to "See also".

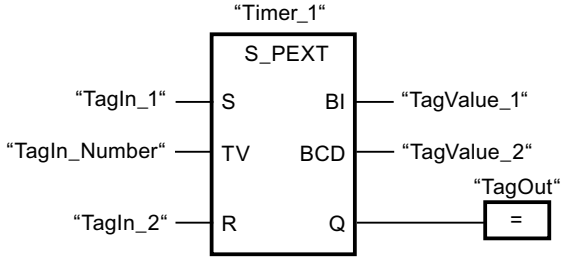
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign extended pulse timer parameters and start" instruction:



Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer runs for the time value of the "TagIn_Number" operand without being affected by a negative edge at the S input. If the signal state at the "TagIn_1" operand changes from "0" to "1" before the timer expires, the timer is restarted.

The "TagOut" operand has the signal state "1" as long as the timer is running. When the time has expired or is reset, the "TagOut" operand is reset to "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

S_ODT: Assign on-delay timer parameters and start (S7-1500)

Description

The "Assign on-delay timer parameters and start" instruction starts a programmed timer when a transition from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV) as long as the signal state at input S is "1". If the timer expires correctly and input S still has signal state "1", output Q returns signal state "1". If the signal state at input S changes from "1" to "0" while the timer is running, the timer is stopped. In this case, output Q is reset to signal state "0".

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero. The time base determines the time period of the time value. The current time value is output binary-coded at output BI and BCD-coded at output BCD.

If the time is running and the signal state at input R changes from "0" to "1", the current time value and the time base are also set to zero. In this case, the signal state at output Q is "0". The timer is reset if the signal state is "1" at the R input even if the timer is not running and the RLO at input S is "1".

The "Assign on-delay timer parameters and start" instruction needs a preceding logic operation for the edge evaluation and can be placed within or at the end of the network.

The instruction data is updated with each access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

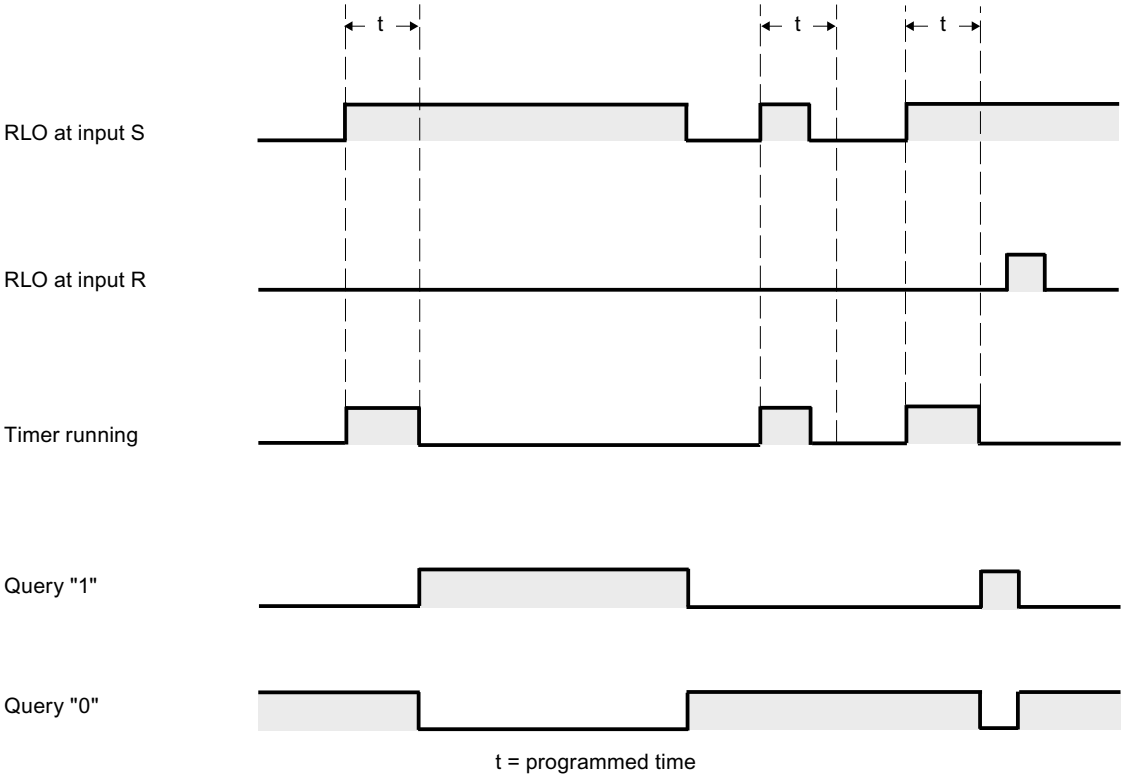
The following table shows the parameters of the "Assign on-delay timer parameters and start" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (binary-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer

For additional information on valid data types, refer to "See also".

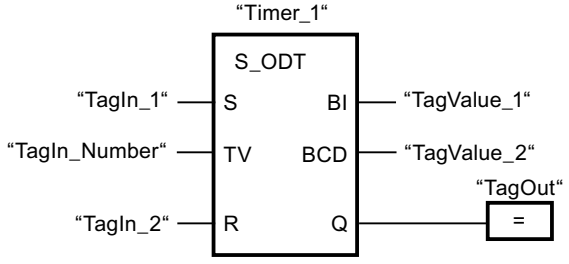
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign on-delay timer parameters and start" instruction:



Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer expires with the value of operand "TagIn_Number". If the timer expires and the signal state of the operand is "1", the "TagOut" operand is set to "1". If the signal state at the "TagIn_1" operand changes from "1" to "0" before the timer expires, the timer is stopped. The "TagOut" operand has the signal state "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

S_ODTS: Assign retentive on-delay timer parameters and start (S7-1500)

Description

The "Assign retentive on-delay timer parameters and start" instruction starts a programmed timer when a transition from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV), even if the signal state at input S changes to "0". If the timer expires, the "Q" output returns signal state "1" regardless of the signal state at input "S". If the signal state at input S changes from "0" to "1" while the timer is running, the timer is restarted with the duration programmed at input (TV).

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero. The time base determines the time period of the time value. The current time value is output binary-coded at output BI and BCD-coded at output BCD.

Signal state "1" at input R resets the current time value and time base to "0" regardless of the signal state at start input S. In this case, the signal state at output Q is "0".

The "Assign retentive on-delay timer parameters and start" instruction needs a preceding logic operation for the edge evaluation and can be placed within or at the end of the network.

The instruction data is updated with each access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

The following table shows the parameters of the "Assign retentive on-delay timer parameters and start" instruction:

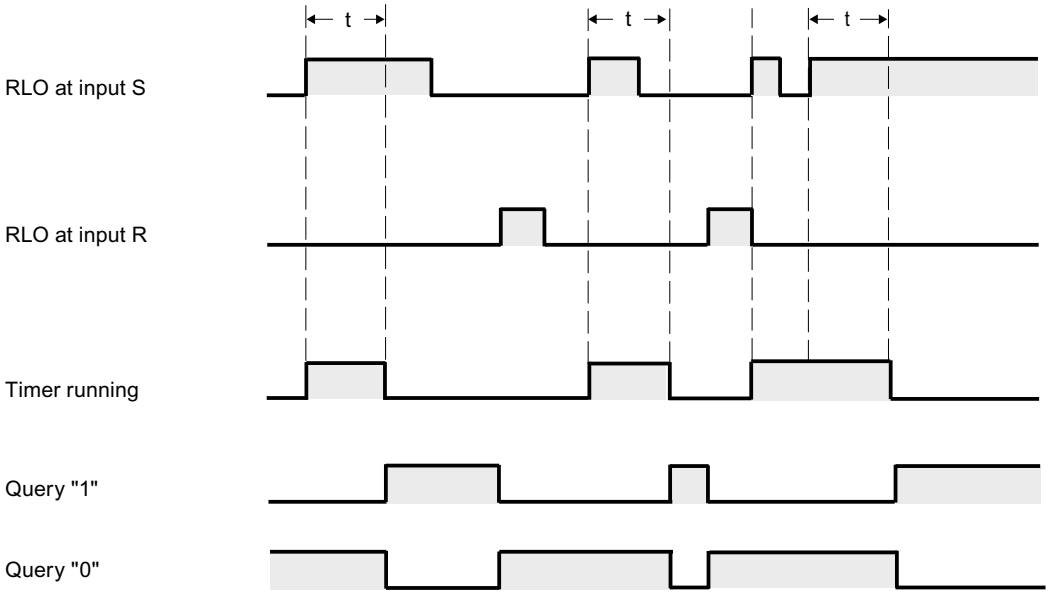
Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, T, C, D, L, P	Start input

Parameters	Declaration	Data type	Memory area	Description
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (binary-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer

For additional information on valid data types, refer to "See also".

Pulse timing diagram

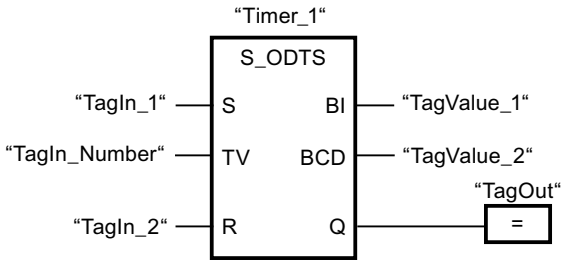
The following figure shows the pulse timing diagram of the "Assign retentive on-delay timer parameters and start" instruction:



t = programmed time

Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer expires with the time value of the "TagIn_Number" operand, even when the signal state of the "TagIn_1" operand changes to "0". When the time expires, the "TagOut" operand is reset to "1". If the signal state at the "TagIn_1" operand changes from "0" to "1" while the timer is running, the timer is restarted.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

S_OFFDT: Assign off-delay timer parameters and start (S7-1500)

Description

The "Assign off-delay timer parameters and start" instruction starts a programmed timer when a transition from "1" to "0" (negative signal edge) is detected in the result of logic operation (RLO) at input S. The timer expires with the programmed duration (TV). As long as the timer is running or input S returns signal state "1", output Q has signal state "1". If the timer expires and the signal state is "0", output Q is reset to signal state "0". If the signal state at input S changes from "0" to "1" while the timer is running, the timer is stopped. The timer is only restarted after a falling signal edge is detected at input S.

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero. The time base determines the time period of the time value. The current time value is output binary-coded at output BI and BCD-coded at output BCD.

Signal state "1" at input R resets the current time value and time base to "0". In this case, the signal state at output Q is "0".

The "Assign off-delay timer parameters and start" instruction needs a preceding logic operation for the edge evaluation and can be placed within or at the end of the network.

The instruction data is updated with each access. It can therefore happen that the query of the data at the start of the cycle returns different values than at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

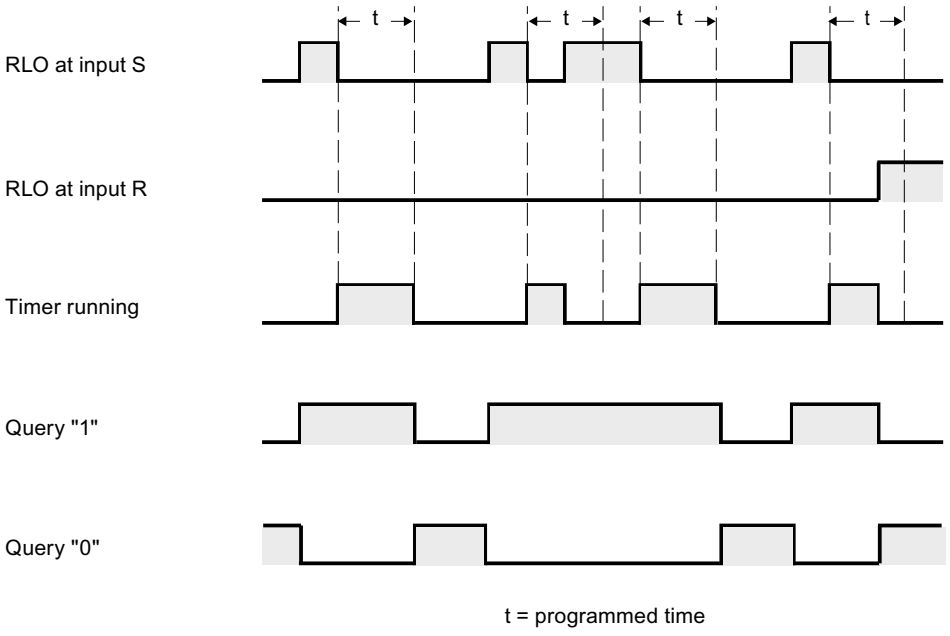
The following table shows the parameters of the "Assign off-delay timer parameters and start" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Timer>	InOut/Input	TIMER	T	Time of the instruction The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
R	Input	BOOL	I, Q, M, T, C, D, L, P or constant	Reset input
BI	Output	WORD	I, Q, M, D, L, P	Current time value (binary-coded)
BCD	Output	WORD	I, Q, M, D, L, P	Current time value (BCD format)
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer

For additional information on valid data types, refer to "See also".

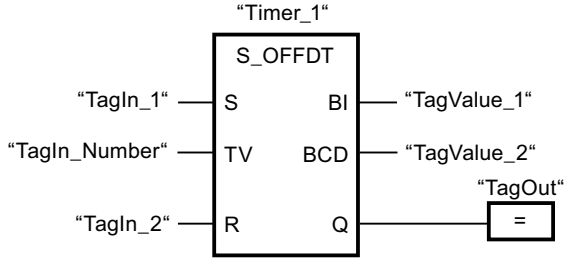
Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign off-delay timer parameters and start" instruction:



Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "1" to "0". The timer expires with the value of operand "TagIn_Number". The "TagOut" operand is set to "1" if the timer is running or the "TagIn_1" operand has the signal state "0". If the signal state at the "TagIn_1" operand changes from "0" to "1" while the timer is running, the timer is reset.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SP: Start pulse timer (S7-1500)

Description

The instruction "Start pulse timer" starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at the start input. The timer runs with the specified duration as long as the RLO has the signal state "1". As long as the timer is running, the query of timer status "1" returns the signal state "1". If there is a change from "1" to "0" in the RLO before the time value has elapsed, the timer stops. In this case, the query for timer status for "1" returns the signal state "0".

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero. The time base determines the time period of the time value.

The "Start pulse timer" instruction requires a preceding logic operation for edge evaluation and can only be placed at the right side of the network.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

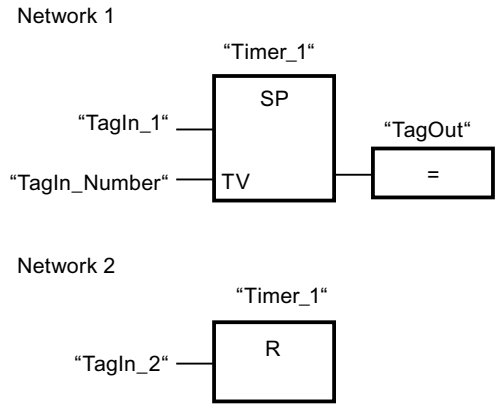
The following table shows the parameters of the instruction "Start pulse timer":

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
<Timer>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

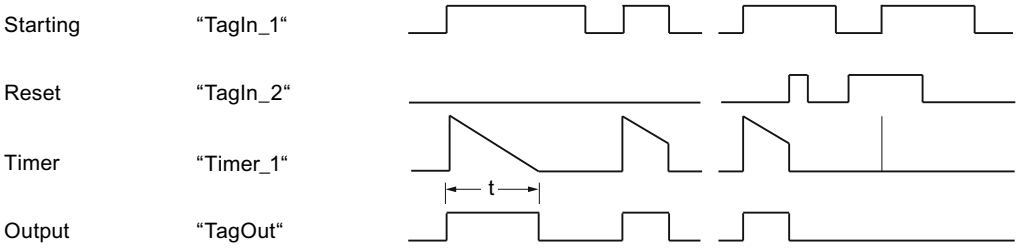
Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer expires with the time value of the "TagIn_Number" operand as long as the signal state of the "TagIn_1" operand is "1". If the signal state of the operand "TagIn_1" changes from "1" to "0" before the timer expires, the timer is stopped. As long as the timer is running, the "TagOut" operand returns signal state "1". A signal state change of the operand "TagIn_2" from "0" to "1" resets the timer, which stops the timer and sets the current time value to "0".

The figure below shows the timer diagram of the example:



t = programmed duration

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SE: Start extended pulse timer (S7-1500)

Description

The "Start extended pulse timer" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at the start input. The timer runs for the specified time period even when the RLO changes to signal state "0". As long as the timer is running, the query of timer status "1" returns the signal state "1". If the RLO changes from "0" to "1" while the timer is running, the timer is restarted with the programmed time period. The querying of the timer status for "1" returns the signal state "0" when the timer expires.

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero. The time base determines the time period of the time value.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed here: AUTOHOTSPOT

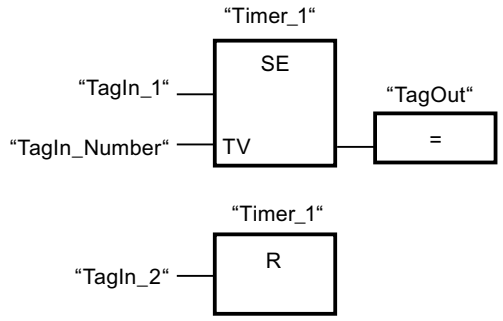
Parameters

The following table shows the parameters of the "Start extended pulse timer" instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
<Timer>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

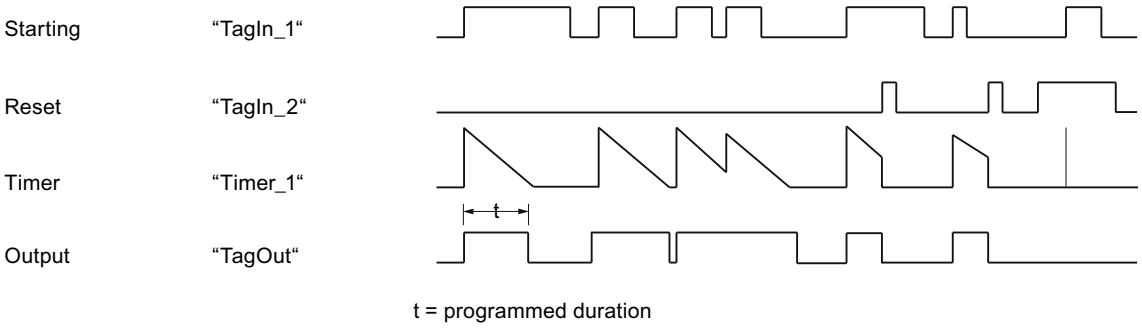
Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer expires with the time value of the operand "TagIn_Number" without being affected by a negative signal edge in the RLO. As long as the timer is running, the "TagOut" operand returns signal state "1". If the signal state of the operand "TagIn_1" changes again from "0" to "1" before the timer expires, the timer is restarted.

The figure below shows the timer diagram of the example:



t = programmed duration

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SD: Start on-delay timer (S7-1500)

Description

The "Start on-delay timer" instruction starts a programmed timer when the signal state "1" is detected at the start input. As long as this signal state remains "1", the timer expires with the specified duration. When the timer expires and the signal state at the start input is still "1", the query for timer status also returns "1". If the signal at the start input is "0", the timer is reset. In this case, querying the timer status returns the signal state "0". As soon as the signal at the start input changes to "1" again, the timer starts running again.

The signal state of the timer output is identical to the signal state of the start input. The start input is directly interconnected to the output and is not linked to the timer.

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed here: AUTOHOTSPOT

Parameters

The following table shows the parameters of the instruction "Start on-delay timer":

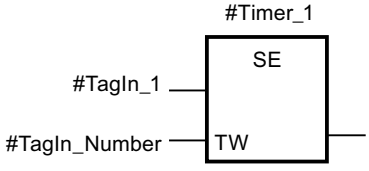
Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
<Timer>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

Example

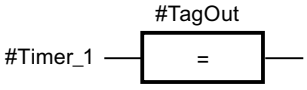
The following example shows how the instruction works:

Network 1:



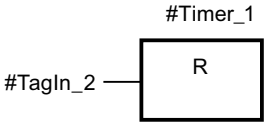
"#Timer_1" starts when the signal state of the "#TagIn_1" operand changes from "0" to "1". The timer expires with the value of operand "#TagIn_Number". If the signal state of the "#TagIn_1" operand changes from "1" to "0" before the timer expires, the timer is reset.

Network 2:



When the timer has expired, the #TagIn_1 operand at the start input has signal state "1" and the timer was not reset, the #TagOut operand is "1".

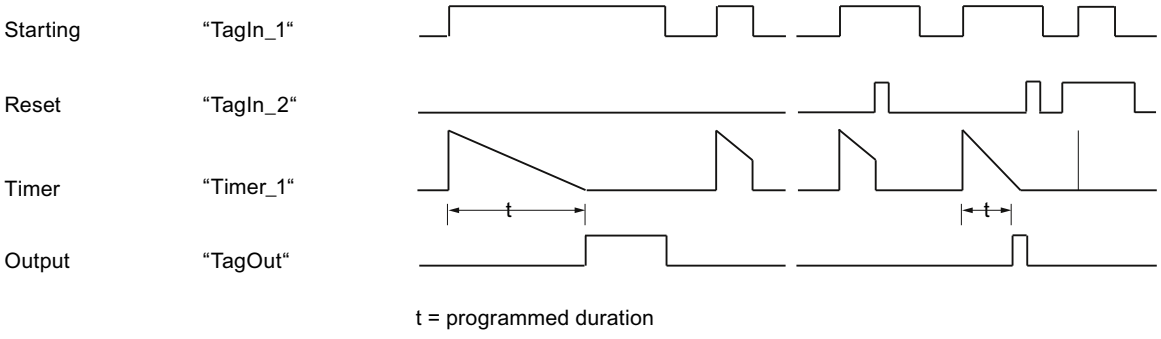
Network 3:



When the signal state of the #TagIn_2 operand has signal state "1", the timer #Timer_1 and the output #TagOut are reset.

If you want to restart #Timer_1, the signal state at the #TagIn_2 operand must be "0" and the signal state at the start input #TagIn_1 must change from "0" to "1".

The figure below shows the timer diagram of the example:



See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SS: Start retentive on-delay timer (S7-1500)

Description

The "Start retentive on-delay timer" instruction starts a programmed timer when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) at the start input. The timer runs for the specified time period even when the RLO changes to signal state "0". When the timer expires, the query for timer status for "1" returns the signal state "1". When the timer expires, the timer can only be restarted if it is explicitly reset.

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down to zero. The time base determines the time period of the time value.

The "Start retentive on-delay timer" instruction needs a preceding logic operation for the edge evaluation and can only be placed on the right edge of the network.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed here: AUTOHOTSPOT

Parameters

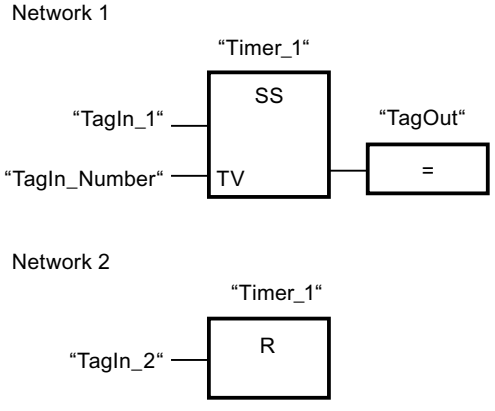
The following table shows the parameters of the instruction "Start retentive on-delay timer":

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
<Timer>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

Example

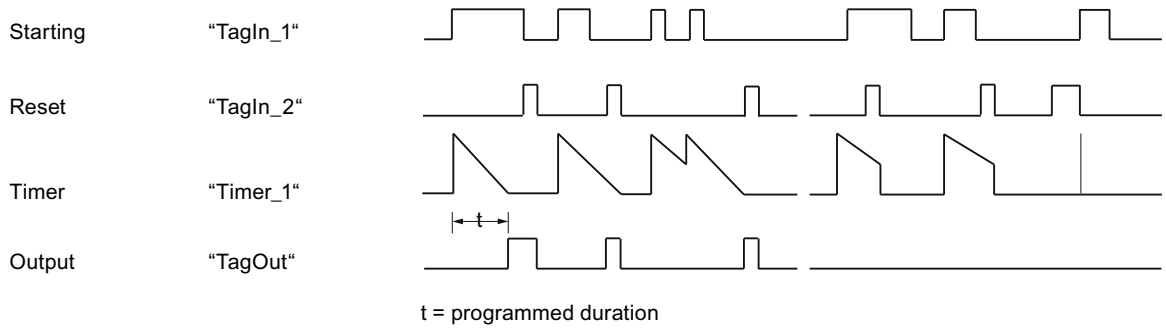
The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "0" to "1". The timer expires with the value of operand "TagIn_Number". When the time expires, the "TagOut" operand is reset to "1". If the signal state at the "TagIn_1" operand changes from "0" to "1"

while the timer is running, the timer is restarted. If the signal state of the "TagIn_2" operand changes to "1", "Timer_1" is reset, i.e. the timer is stopped and the current time value is set to "0".

The figure below shows the timer diagram of the example:



See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SF: Start off-delay timer (S7-1500)

Description

The "Start off-delay timer" instruction starts a programmed timer when a change from "1" to "0" (negative signal edge) is detected in the result of logic operation (RLO) at the start input. The timer expires with the specified duration. As long as the timer is running, the query of timer status "1" returns the signal state "1". If the RLO changes from "0" to "1" while the timer is running, the timer is reset. The timer is always restarted when the RLO changes from "1" to "0".

The duration is made up internally of a time value and a time base and is programmed at parameter TV. When the instruction is started, the programmed time value is counted down towards zero. The time base indicates the time period with which the time value is changed.

The queries of the timer status for "1" return the query result "1" if the signal state of the result of logic operation is "1" at the execution of the instruction. If the RLO is "0", queries of the timer status for "1" return the query result "0".

The "Start off-delay timer" instruction requires a preceding logic operation for edge evaluation and can only be placed at the right side of the network.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed here: AUTOHOTSPOT

Parameters

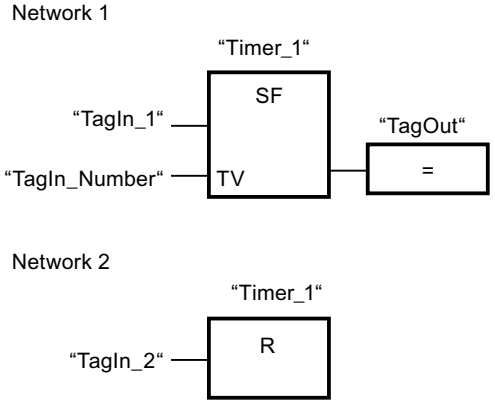
The following table shows the parameters of the instruction "Start off-delay timer":

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	BOOL	I, Q, M, T, C, D, L, P	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L or constant	Time duration
<Timer>	InOut/Input	TIMER	T	Timer which is started. The number of timers depends on the CPU.

You can find additional information on valid data types under "See also".

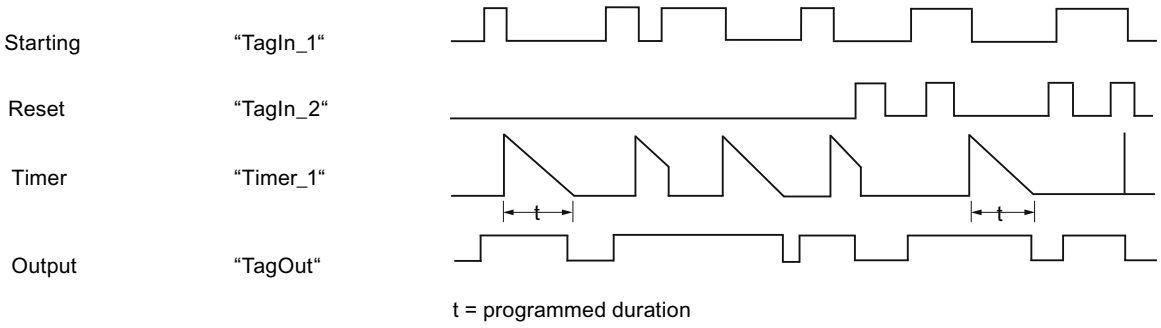
Example

The following example shows how the instruction works:



"Timer_1" starts when the signal state of the "TagIn_1" operand changes from "1" to "0". The timer expires with the time value of the operand "TagIn_Number". As long as the timer is running, the "TagOut" operand is set to "1". If the signal state at the "TagIn_1" operand changes from "1" to "0" while the timer is running, the timer is restarted. If the signal state of the "TagIn_2" operand is "1", "Timer_1" is reset, i.e. the timer is stopped and the current time value is set to "0".

The figure below shows the timer diagram of the example:



See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Counter operations (S7-1200, S7-1500)

CTU: Count up (S7-1200, S7-1500)

Description

You can use the "Count up" instruction to increment the value at output CV. When the signal state at the CU input changes from "0" to "1" (positive signal edge), the instruction is executed and the current counter value at the CV output is incremented by one. The counter value is incremented each time a positive signal edge is detected, until it reaches the high limit for the data type specified at the output CV. When the high limit is reached, the signal state at the CU input no longer has an effect on the instruction.

You can query the counter status in the Q output. The signal state at the Q output is determined by the PV parameter. If the current counter value is greater than or equal to the value of the PV parameter, the Q output is set to signal state "1". In all other cases, the Q output has signal state "0". You can also specify a constant for the PV parameter.

The value at the CV output is reset to "0" and saved to an edge memory bit when the signal state at input R changes to "1". As long as the R input has signal state "1", the signal state at the CU input has no effect on the instruction.

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count up" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For S7-1200 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER	• CTU_SINT / CTU_USINT • CTU_INT / CTU_UINT • CTU_DINT / CTU_UDINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER

For S7-1500 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER	• CTU_SINT / CTU_USINT • CTU_INT / CTU_UINT • CTU_DINT / CTU_UDINT • CTU_LINT / CTU_ULINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER

You can declare an IEC counter as follows:

- Declaration of a data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER")
- Declaration as a local tag of the type CTU_<Data type> or IEC_<Counter> in the "Static" section of a block (for example #MyIEC_COUNTER)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive. For additional information on setting retentivity in an instance data block, refer to "See also".

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

The execution of the "Count up" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

Parameters

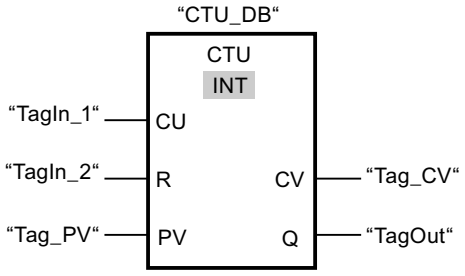
The following table shows the parameters of the "Count up" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
CU	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Count input
R	Input	BOOL	I, Q, M, D, L, P or constant	I, Q, M, T, C, D, L, P or constant	Reset input
PV	Input	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Value at which the output Q is set.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Counter status
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	I, Q, M, D, L, P	Current counter value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



When the signal state of the "TagIn_1" operand changes from "0" to "1", the "Count up" instruction is executed and the current counter value of the "Tag_CV" operand is incremented by one. With each additional positive signal edge, the counter value is incremented until the high limit of the specified data type (INT = 32767) is reached.

The value of the PV parameter is adopted as the limit for determining the "TagOut" output. The "TagOut" output has signal state "1" as long as the current counter value is greater than or equal to the value of the "Tag_PV" operand. In all other cases, the "TagOut" output returns the signal state "0".

See also

Overview of the valid data types (Page 2239)

Setting retentivity in an instance data block (Page 4809)

Setting the retentivity of local tags (Page 4769)

Basic information on FBD (Page 4937)

CTD: Count down (S7-1200, S7-1500)

Description

You can use the "Count down" instruction to decrement the value at output CV. When the signal state at the CD input changes from "0" to "1" (positive signal edge), the instruction is executed and the current counter value at the CV output is decremented by one. Each time a positive signal edge is detected, the counter value is decremented until it reaches the low limit of the specified data type. When the low limit is reached, the signal state at the CD input no longer has an effect on the instruction.

You can query the counter status in the Q output. If the current counter value is less than or equal to "0", the Q output is set to signal state "1". In all other cases, the Q output has signal state "0". You can also specify a constant for the PV parameter.

The value at the CV output is set to the value of the PV parameter and saved to a edge memory bit when the signal state at the LD input changes from "0" to "1". As long as the LD input has signal state "1", the signal state at the CD input has no effect on the instruction.

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count down" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For S7-1200 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER	• CTD_SINT / CTD_USINT • CTD_INT / CTD_UINT • CTD_DINT / CTD_UDINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER

For S7-1500 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER	• CTD_SINT / CTD_USINT • CTD_INT / CTD_UINT • CTD_DINT / CTD_UDINT • CTD_LINT / CTD_ULINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER

You can declare an IEC counter as follows:

- Declaration of a data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER")
- Declaration as a local tag of the type CTD_<Data type> or IEC_<Counter> in the "Static" section of a block (for example #MyIEC_COUNTER)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive. For additional information on setting retentivity in an instance data block, refer to "See also".

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

The execution of the "Count down" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

Parameters

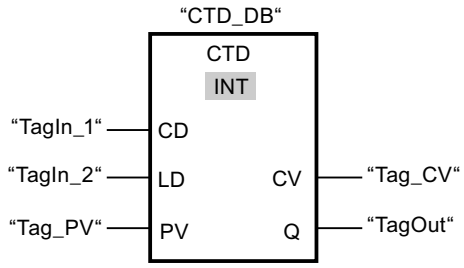
The following table shows the parameters of the "Count down" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
CD	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Count input
LD	Input	BOOL	I, Q, M, D, L, P or constant	I, Q, M, T, C, D, L, P or constant	Load input
PV	Input	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Value to which the CV output is set with LD = 1.
Q	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Counter status
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	I, Q, M, D, L, P	Current counter value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



When the signal state of the "TagIn_1" operand changes from "0" to "1", the instruction is executed and the value at the "Tag_CV" output is decremented by one. With each additional positive signal edge, the counter value is decremented until the low limit of the specified data type (INT = -32768) is reached.

The value of the PV parameter is adopted as the limit for determining the "TagOut" output. The "TagOut" output has signal state "1" as long as the current counter value is less than or equal to "0". In all other cases, the "TagOut" output returns the signal state "0".

See also

Overview of the valid data types (Page 2239)

Setting retentivity in an instance data block (Page 4809)

Setting the retentivity of local tags (Page 4769)

Basic information on FBD (Page 4937)

CTUD: Count up and down (S7-1200, S7-1500)

Description

You can use the "Count up and down" instruction to increment and decrement the counter value at the CV output. If the signal state at the CU input changes from "0" to "1" (positive signal edge), the current counter value is incremented by one and stored at the CV output. If the signal state at the CD input changes from "0" to "1" (positive signal edge), the current counter value at the CV output is decremented by one. If there is a positive signal edge at the CU and CD inputs in one program cycle, the current counter value at the CV output remains unchanged.

The counter value can be incremented until it reaches the high limit of the data type specified at the CV output. When the high limit is reached, the counter value is no longer incremented on a positive signal edge. The counter value is no longer decremented once the low limit of the specified data type has been reached.

When the signal state at the LD input changes to "1", the counter value at the CV output is set to the value of the PV parameter and stored in a edge memory bit. As long as the LD input has signal state "1", the signal state at the CU and CD inputs has no effect on the instruction.

The counter value is set to "0" and stored in an edge memory bit when the signal state at input R changes to "1". As long as the R input has signal state "1", a change in the signal state of the CU, CD and LD inputs has no effect on the "Count up and down" instruction.

You can query the status of the up counter at the QU output. If the current counter value is greater than or equal to the value of the PV parameter, the QU output is set to signal state "1". In all other cases, the QU output has signal state "0". You can also specify a constant for the PV parameter.

You can query the status of the down counter at the QD output. If the current counter value is less than or equal to zero, the QD output is set to signal state "1". In all other cases, the QD output has signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count up and down" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For S7-1200 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER	• CTUD_SINT / CTUD_USINT • CTUD_INT / CTUD_UINT • CTUD_DINT / CTUD_UDINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER

For S7-1500 CPU

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER	• CTUD_SINT / CTUD_USINT • CTUD_INT / CTUD_UINT • CTUD_DINT / CTUD_UDINT • CTUD_LINT / CTUD_ULINT • IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER

You can declare an IEC counter as follows:

- Declaration of a data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER")
- Declaration as a local tag of the type CTUD_<Data type> or IEC_<Counter> in the "Static" section of a block (for example #MyIEC_COUNTER)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive. For additional information on setting retentivity in an instance data block, refer to "See also".

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

The execution of the "Count up and down" instruction requires a preceding logic operation. It can be placed within or at the end of the network.

Parameters

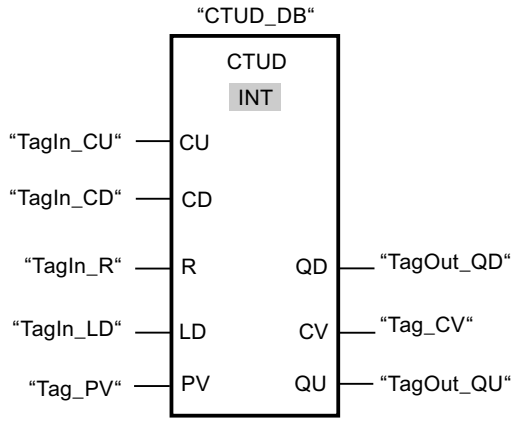
The following table shows the parameters of the "Count up and down" instruction:

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
CU	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Count up input
CD	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Count down input
R	Input	BOOL	I, Q, M, D, L, P or constant	I, Q, M, T, C, D, L, P or constant	Reset input
LD	Input	BOOL	I, Q, M, D, L, P or constant	I, Q, M, T, C, D, L, P or constant	Load input
PV	Input	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Value at which the QU output is set / value to which the CV output is set when LD = 1.
QU	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Up-counter status
QD	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Down-counter status
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	I, Q, M, D, L, P	Current counter value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



If the signal state at the "TagIn_CU" or "TagIn_CD" input changes from "0" to "1" (positive signal edge), the "Count up and down" instruction is executed. When there is a positive signal edge at the "TagIn_CU" input, the current counter value is incremented by one and stored at the "Tag_CV" output. When there is a positive signal edge at the "TagIn_CD" input, the counter value is decremented by one and stored at the "Tag_CV" output. When there is a positive edge at the CU input, the counter value is incremented until it reaches the high limit (INT = 32767). If input CD has a positive signal edge, the counter value is decremented until it reaches the low limit (INT = -32768).

The "TagOut_GU" output has signal state "1" as long as the current counter value is greater than or equal to the value at the "Tag_PV" input. In all other cases, the "TagOut_QU" output returns the signal state "0".

The "TagOut_QD" output has signal state "1" as long as the current counter value is less than or equal to "0". In all other cases, the "TagOut_QD" output has signal state "0".

See also

Overview of the valid data types (Page 2239)

Example of detecting the fill level of a storage area (Page 4982)

Setting retentivity in an instance data block (Page 4809)

Setting the retentivity of local tags (Page 4769)

FBD programming examples (Page 4978)

Basic information on FBD (Page 4937)

Legacy (S7-1500)

S_CU: Assign parameters and count up (S7-1500)

Description

You can use the "Assign parameters and count up" instruction to increment the value of a counter. If the signal state at the CU input changes from "0" to "1" (positive signal edge), the current counter value is incremented by one. The current counter value is output as a hexadecimal value at output CV and BCD-coded at output CV_BCD. The count is incremented until the limit of "999" is reached. When the limit is reached, the counter value is no longer incremented on a positive signal edge.

When the signal state at input S changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if RLO at input CU is "1", the counter will count accordingly in the next scan cycle, even when no change has been detected in the signal edge.

The counter value is set to zero when the signal state at the R input changes to "1". As long as the R input has the signal state "1", processing of the signal state of the CU and S inputs has no effect on the counter value.

The signal state at output Q is "1" if the counter value is greater than zero. If the counter value is equal to zero, output Q has the signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

The "Assign parameters and count up" instruction needs a preceding logic operation for the edge evaluation and can be placed within or at the end of the network.

Parameters

The following table shows the parameters of the "Assign parameters and count up" instruction:

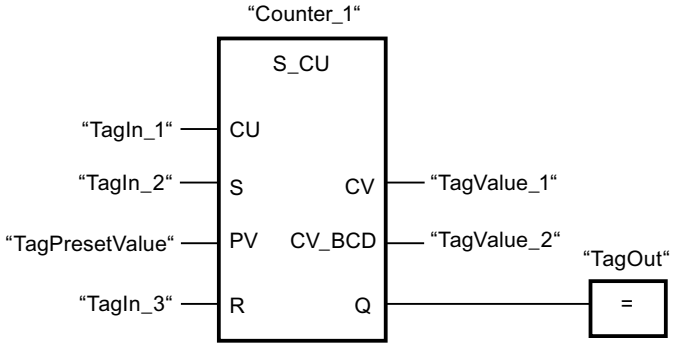
Parameters	Declaration	Data type	Memory area	Description
<counter>	InOut/Input	COUNTER	C	Counter of the instruction The number of counters depends on the CPU.
CU	Input	BOOL	I, Q, M, D, L, T, C	Count up input
S	Input	BOOL	I, Q, M, D, L, T, C or constant	Input for presetting counter
PV	Input	WORD	I, Q, M, D, L or constant	Preset counter value (C#0 to C#999)
R	Input	BOOL	I, Q, M, D, L, T, C or constant	Reset input
CV	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (hexadecimal)

Parameters	Declaration	Data type	Memory area	Description
CV_BCD	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the counter

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



When the signal state at the "TagIn_1" input changes from "0" to "1" (positive signal edge) and the current counter value is less than "999", the counter value is incremented by one. When the signal state at the "TagIn_2" input changes from "0" to "1", the counter value is set to the value of the "TagPresetValue" operand. The counter value is reset to "0" when the "TagIn_3" operand has signal state "1".

The current counter value is hexadecimal in the "TagValue_1" operand and BCD-coded in the "TagValue_2" operand.

The "TagOut" output has the signal state "1" as long as the current counter value is not equal to "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

S_CD: Assign parameters and count down (S7-1500)

Description

You can use the "Assign parameters and count down" instruction to decrement the value of a counter. If the signal state at the CD input changes from "0" to "1" (positive signal edge), the counter value is decremented by one. The current counter value is output as a hexadecimal value at output CV and BCD-coded at output CV_BCD. The count is decremented until the low limit of "0" is reached. When the low limit is reached, the counter value is no longer decremented on a positive signal edge.

When the signal state at input S changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if RLO at input CD is "1", the counter will count accordingly in the next scan cycle, even when no change has been detected in the signal edge.

The counter value is set to zero when the signal state at the R input changes to "1". As long as the R input has the signal state "1", processing of the signal state of the CD and S inputs has no effect on the counter value.

The signal state at output Q is "1" if the counter value is greater than zero. If the counter value is equal to zero, output Q has the signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

The "Assign parameters and count down" instruction needs a preceding logic operation for the edge evaluation and can be placed within or at the end of the network.

Parameters

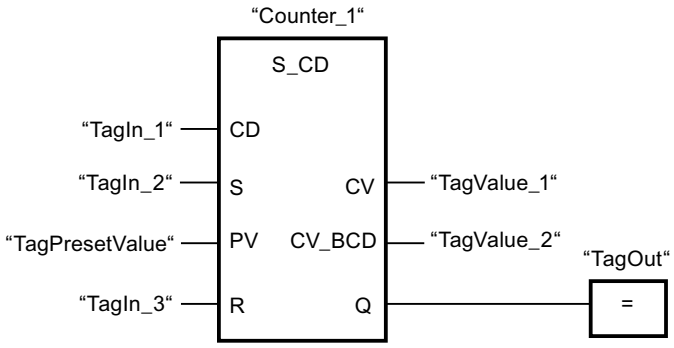
The following table shows the parameters of the "Assign parameters and count down" instruction:

Parameters	Declaration	Data type	Memory area	Description
<counter>	InOut/Input	COUNTER	C	Counter of the instruction The number of counters depends on the CPU.
CD	Input	BOOL	I, Q, M, D, L or constant	Count down input
S	Input	BOOL	I, Q, M, D, L, T, C or constant	Input for presetting counter
PV	Input	WORD	I, Q, M, D, L or constant	Preset counter value (C#0 to C#999)
R	Input	BOOL	I, Q, M, D, L, T, C or constant	Reset input
CV	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (hexadecimal)
CV_BCD	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the counter

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



When the signal state at the "TagIn_1" input changes from "0" to "1" (positive signal edge) and the current counter value is greater than "0", the counter value is decremented by one. When the signal state at the "TagIn_2" input changes from "0" to "1", the counter value is set to the value of the "TagPresetValue" operand. The counter value is reset to "0" when the "TagIn_3" operand has signal state "1".

The current counter value is hexadecimal in the "TagValue_1" operand and BCD-coded in the "TagValue_2" operand.

The "TagOut" output has the signal state "1" as long as the current counter value is not equal to "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

S_CUD: Assign parameters and count up / down (S7-1500)

Description

You can use the "Assign parameters and count up / down" instruction to increment or decrement the value of a counter. If the signal state at the CU input changes from "0" to "1" (positive signal edge), the current counter value is incremented by one. If the signal state at the CD input changes from "0" to "1" (positive signal edge), the counter value is decremented by one. The current counter value is output as a hexadecimal value at output CV and BCD-coded at output CV_BCD. If there is a positive signal edge at the CU and CD inputs in one program cycle, the counter value remains unchanged.

The counter value is incremented until the high limit of "999" is reached. When the high limit value is reached, the counter value is no longer incremented on a positive signal edge. When the low limit of "0" is reached, the counter value is no longer decremented.

When the signal state at input S changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if RLO is "1" at the inputs CU and CD, the counter counts accordingly in the next scan cycle, even if no change in the signal edge is detected.

The counter value is set to zero when the signal state at the R input changes to "1". As long as the R input has the signal state "1", processing of the signal state of the CU, CD and S inputs has no effect on the counter value.

The signal state at output Q is "1" if the counter value is greater than zero. If the counter value is equal to zero, output Q has the signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

The "Assign parameters and count up / down" instruction needs a preceding logic operation for the edge evaluation and can be placed within or at the end of the network.

Parameters

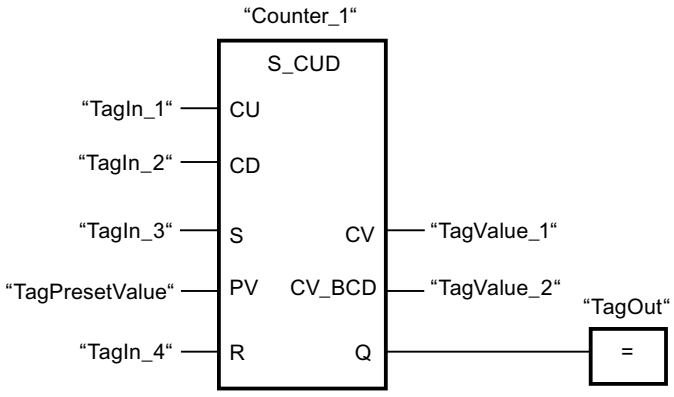
The following table shows the parameters of the "Assign parameters and count up / down" instruction:

Parameters	Declaration	Data type	Memory area	Description
<counter>	InOut/Input	COUNTER	C	Counter of the instruction The number of counters depends on the CPU.
CU	Input	BOOL	I, Q, M, D, L, T, C	Count up input
CD	Input	BOOL	I, Q, M, D, L, T, C or constant	Count down input
S	Input	BOOL	I, Q, M, D, L, T, C or constant	Input for presetting counter
PV	Input	WORD	I, Q, M, D, L or constant	Preset counter value (C#0 to C#999)
R	Input	BOOL	I, Q, M, D, L, T, C or constant	Reset input
CV	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (hexadecimal)
CV_BCD	Output	WORD, S5TIME, DATE	I, Q, M, D, L	Current counter value (BCD format)
Q	Output	BOOL	I, Q, M, D, L	Status of the counter

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the signal state at the input "TagIn_1" or at the input "TagIn_2" changes from "0" to "1" (positive edge), the instruction "Assign parameters and count up / down is executed. When there is a positive signal edge at the "TagIn_1" input and the current counter value is less than "999", the counter value is incremented by one. When there is a positive signal edge at the "TagIn_2" input and the current counter value is greater than "0", the counter value is decremented by one.

When the signal state at the "TagIn_3" input changes from "0" to "1", the counter value is set to the value of the "TagPresetValue" operand. The counter value is reset to "0" when the "TagIn_4" operand has signal state "1".

The current counter value is hexadecimal in the "TagValue_1" operand and BCD-coded in the "TagValue_2" operand.

The "TagOut" output has the signal state "1" as long as the current counter value is not equal to "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SC: Set counter value (S7-1500)

Description

You can use the "Set counter value" instruction to set the value of a counter. The instruction is executed when the result of logic operation (RLO) at the start input of the instruction changes from "0" to "1". When the instruction is executed, the counter is set to the specified counter value.

The "Set counter value" instruction needs a preceding logic operation for the edge evaluation and can only be placed on the right edge of the network.

Parameters

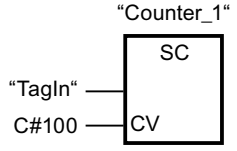
The following table lists the parameters of the "Set counter value" instruction:

Parameters	Declaration	Data type	Memory area	Description
<operand>	Input	BOOL	I, Q, M, T, C, D, L	Start input
PV	Input	WORD	I, Q, M, D, L or constant	Value with which the counter is preset in the BCD format. (C#0 to C#999)
<counter>	InOut/Input	COUNTER	C	Counter that is preset.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



the counter "Counter_1" starts with the value "100" when the signal state of the "TagIn" operand changes from "0" to "1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

CU: Count up (S7-1500)

Description

You use the "Count up" instruction to increment the value of the specified counter by the count of one on a rising edge at the start input. The count is incremented until the limit of "999" is reached. When the limit is reached, the counter value is no longer incremented on a positive signal edge.

The "Count up" instruction needs a preceding logic operation for the edge evaluation and can only be placed on the right edge of the network.

Parameters

The following table shows the parameters of the "Count up" instruction:

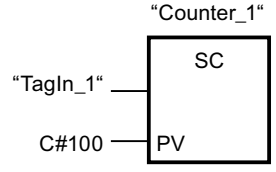
Parameters	Declaration	Data type	Memory area	Description
<operand>	Input	BOOL	I, Q, M, T, C, D, L	Start input
<counter>	InOut/Input	COUNTER	C	Counter whose value is incremented.

For additional information on valid data types, refer to "See also".

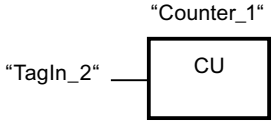
Example

The following example shows how the instruction works:

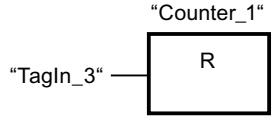
Network 1



Network 2



Network 3



When the signal state of the operand "TagIn_1" changes from "0" to "1" (positive signal edge), the counter "Counter_1" is preset with the value "100".

The value of "Counter_1" is incremented by the count of one when the signal state of the "TagIn_2" operand changes from "0" to "1".

If the "TagIn_3" operand returns signal state "1", the value of "Counter_1" is reset to "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

CD: Count down (S7-1500)

Description

You use the "Count down" instruction to decrement the value of the specified counter by the count of one on a rising edge at the start input. The count is decremented until the limit of "0" is reached. When the limit is reached, the counter value is no longer changed on a positive signal edge.

The "Count down" instruction needs a preceding logic operation for the edge evaluation and can only be placed on the right edge of the network.

Parameters

The following table shows the parameters of the "Count down" instruction:

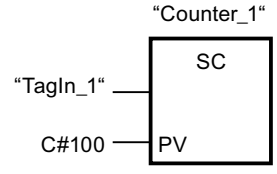
Parameters	Declaration	Data type	Memory area	Description
<operand>	Input	BOOL	I, Q, M, T, C, D, L	Start input
<counter>	InOut/Input	COUNTER	C	Counter whose value is decremented.

For additional information on valid data types, refer to "See also".

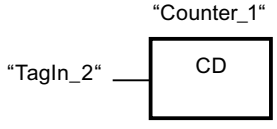
Example

The following example shows how the instruction works:

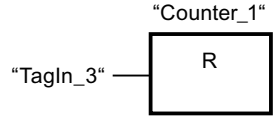
Network 1



Network 2



Network 3



When the signal state of the operand "TagIn_1" changes from "0" to "1" (positive signal edge), the counter "Counter_1" is preset with the value "100".

The value of "Counter_1" is incremented by the count of one when the signal state of the "TagIn_2" operand changes from "0" to "1".

If the "TagIn_3" operand returns signal state "1", the value of "Counter_1" is reset to "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Comparator operations (S7-1200, S7-1500)

CMP ==: Equal (S7-1200, S7-1500)

Description

You can use the "Equal" instruction to query whether the value at input IN1 is equal to the value at input IN2.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

If IEC check is enabled, the operands to be compared must be of the same data type. If IEC check is not enabled, the width of the operands must be the same.

Comparison of floating-point numbers

If you want to compare the data types REAL or LREAL, then instead of the instruction "CMP ==: Equal" we recommend using the instruction "IN_RANGE: Value within range".

If floating-point numbers are being compared, the operands to be compared must be of the same data type regardless of the IEC check setting.

The special bit patterns of invalid floating-point numbers (NaN) that are the outcome of undefined results (e.g. root of -1) are not comparable. This means that if one of the operands has the value NaN, the instruction "CMP ==: Equal" has the result FALSE.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison.

The following table shows examples of string comparisons:

IN1	IN2	RLO of the instruction
'AA'	'AA'	1
'Hello World'	'HelloWorld'	0
'AA'	'aa'	0
'aa'	'aaa'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

Bit patterns of invalid timers, date and times, e.g. DT#2015-13-33-25:62:99.999_999_999, cannot be compared. This means that if one of the operands has an invalid value, the instruction "CMP ==: Equal" has the result FALSE.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Comparison of hardware data types

To be able to compare the operands of the PORT data type, you need to select the WORD data type from the drop-down list in the instruction box.

If you want to compare the two data types, you must first create a tag of data type HW_ANY in the "Temp" section of the the block interface and then copy the LADDR (from data type HW_DEVICE) to the tag. It is then possible to compare HW_ANY and HW_IO.

Comparison of structures

Note

Availability of comparison of structures

The option to compare structures is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

You can compare the values of two structured operands when both tags are of the same structure data type. When structures are compared, the operands to be compared must have the same data type regardless of the setting for the IEC Check. An exception is comparisons in which one of the two operands is a VARIANT or an ANY. If the data type is not yet known when the program is created, you can use the VARIANT data type. In this case you can also compare the operand with a structured tag of any data type. You can also compare two tags of the data type VARIANT or ANY with each other.

You can select the data type VARIANT for the comparison of structures from the drop-down list in the instruction box. The following data types are possible:

- PLC data type
- STRUCT (the structure of the data type STRUCT must be contained in a PLC data type (UDT) or the two structures to be compared are two elements of an ARRAY of STRUCT. Anonymous structures are not permitted.)
- Tag to which ANY is pointing
- Tag to which VARIANT is pointing

The following requirements must be met to compare two tags of the data type ARRAY with VARIANT selected:

- The elements must each have the same data type.
- The two ARRAYs must have the same dimension.
- All dimensions must have the same number of elements. The exact ARRAY limits do not have to match.

Note

ARRAY of BOOL

When you compare two operands of the data type ARRAY of BOOL with each other, you have selected the data type VARIANT from the drop-down list, and the number of elements cannot be divided by 8, the fill bits are compared as well. This may affect the comparison result.

If an element within the structures that you want to compare is an invalid STRING/WSTRING, an invalid time or date or an invalid floating-point number, the comparison result in the RLO (result of logic operation) will return the signal state "0".

The table below shows an example of a comparison of structures:

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	Tag of data type A <PLC data type>		Tag value	1
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	Tag of data type B <PLC data type>		Tag value	0
	BOOL	FALSE		BOOL	TRUE	
	INT	2		INT	3	

<Operand1>			<Operand2>			RLO of the instruction
Tag of data type A <PLC data type>		Tag value	VARIANT (supplied with tag of data type A)		Tag value	1
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

Parameters

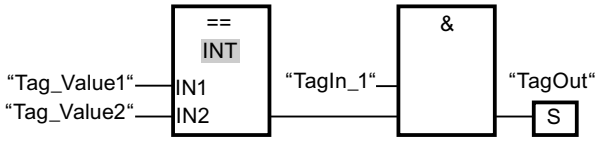
The following table shows the parameters of the instruction "Equal":

Parameters	Declaration	Data type	Memory area	Description
IN1	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time, ARRAY of <data type> with fixed and variable ARRAY limits, STRUCT, VARIANT, ANY, PLC data type	I, Q, M, D, L, P, or constant	First comparison value
IN2	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time, ARRAY of <data type> with fixed and variable ARRAY limits, STRUCT, VARIANT, ANY, PLC data type	I, Q, M, D, L, P, or constant	Second value to compare
As detailed above, the data types ARRAY, STRUCT (in a PLC data type), VARIANT, ANY and PLC data type (UDT) are only available as of firmware version 2.0 or 4.2.				

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" = "Tag_Value2".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

CMP <>: Not equal (S7-1200, S7-1500)

Description

You can use the "Not equal" instruction to query whether the value at input IN1 is not equal to the value at input IN2.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

If IEC check is enabled, the operands to be compared must be of the same data type. If IEC check is not enabled, the width of the operands must be the same.

Comparison of floating-point numbers

If you want to compare the data types REAL or LREAL, then instead of the instruction "CMP <>: Not equal" we recommend using the instruction "OUT_RANGE: Value outside range".

If floating-point numbers are being compared, the operands to be compared must be of the same data type regardless of the IEC check setting.

The special bit patterns of invalid floating-point numbers (NaN) that are the outcome of undefined results (e.g. root of -1) are not comparable. This means that if one of the operands has the value NaN, the instruction "CMP <>: Not equal" has the result FALSE.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison.

The following table shows examples of string comparisons:

IN1	IN2	RLO of the instruction
'AA'	'aa'	1
'Hello World'	'HelloWorld'	1
'AA'	'AA'	0
'aa'	'aaa'	1

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

Bit patterns of invalid timers, date and times, e.g. DT#2015-13-33-25:62:99.999_999_999, cannot be compared. This means that if one of the operands has an invalid value, the instruction "CMP <>: Not equal" has the result FALSE.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Comparison of hardware data types

To be able to compare the operands of the PORT data type with the "Not equal" instruction, you need to select the WORD data type from the drop-down list of the instructions box.

If you want to compare the two hardware data types HW_IO and HW_DEVICE, you must first create a tag of data type HW_ANY in the section "Temp" of the block interface and then copy the LADDR (of data type HW_DEVICE) to the tag. It is then possible to compare HW_ANY and HW_IO.

Comparison of structures

Note

Availability of comparison of structures

The option to compare structures is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

You can compare the values of two structured operands when both tags are of the same structure data type. When structures are compared, the operands to be compared must have the same data type regardless of the setting for the IEC Check. An exception is comparisons in which one of the two operands is a VARIANT or an ANY. If the data type is not yet known when the program is created, you can use the VARIANT data type. In this case you can also compare the operand with a structured tag of any data type. You can also compare two tags of the data type VARIANT or ANY with each other.

You can select the data type VARIANT for the comparison of structures from the drop-down list in the instruction box. The following data types are possible:

- PLC data type
- STRUCT (the structure of the data type STRUCT must be contained in a PLC data type (UDT) or the two structures to be compared are two elements of an ARRAY of STRUCT. Anonymous structures are not permitted.)
- Tag to which ANY is pointing
- Tag to which VARIANT is pointing

The following requirements must be met to compare two tags of the data type ARRAY with VARIANT selected:

- The elements must each have the same data type.
- The two ARRAYs must have the same dimension.
- All dimensions must have the same number of elements. The exact ARRAY limits do not have to match.

Note

ARRAY of BOOL

When you compare two operands of the data type ARRAY of BOOL with each other, you have selected the data type VARIANT from the drop-down list, and the number of elements cannot be divided by 8, the fill bits are compared as well. This may affect the comparison result.

If an element within the structures that you want to compare is an invalid STRING/WSTRING, an invalid time or date or an invalid floating-point number, the comparison result in the RLO (result of logic operation) will return the signal state "0".

The table below shows an example of a comparison of structures:

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	Tag of data type A <PLC data type>		Tag value	0
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	Tag of data type B <PLC data type>		Tag value	1
	BOOL	FALSE		BOOL	TRUE	
	INT	2		INT	3	

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	VARIANT (supplied with tag of data type A)		Tag value	0
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

Parameters

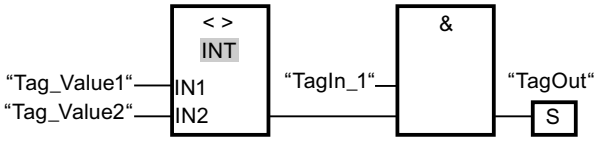
The following table shows the parameters of the "Not equal" instruction:

Parameters	Declaration	Data type	Memory area	Description
IN1	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time, ARRAY of <data type> with fixed and variable ARRAY limits, STRUCT, VARIANT, ANY, PLC data type	I, Q, M, D, L, P, or constant	First comparison value
IN2	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time, ARRAY of <data type> with fixed and variable ARRAY limits, STRUCT, VARIANT, ANY, PLC data type	I, Q, M, D, L, P, or constant	Second value to compare
As detailed above, the data types ARRAY, STRUCT (in a PLC data type), VARIANT, ANY and PLC data type (UDT) are only available as of firmware version 2.0 or 4.2.				

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" <> "Tag_Value2".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

CMP >=: Greater or equal (S7-1200, S7-1500)

Description

You can use the "Greater or equal" instruction to query whether the value at input IN1 is greater or equal to the value at input IN2. Both values to be compared must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the longer string is considered greater.

The following table shows examples of string comparisons:

IN1	IN2	RLO of the instruction
'BB'	'AA'	1
'AAA'	'AA'	1
'Hello World'	'Hello World'	1
'Hello World'	'HelloWorld'	0
'AA'	'aa'	0
'AAA'	'a'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

In comparing timer values, the RLO of the instruction is "1" if the timer at input IN1 is greater (more recent) than or equal to the timer at input IN2.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Parameters

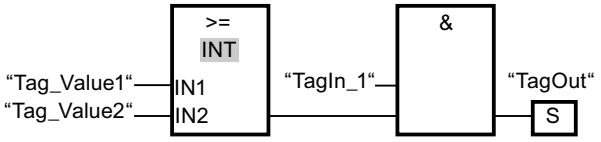
The following table shows the parameters of the "Greater or equal" instruction:

Parameters	Declaration	Data type	Memory area	Description
IN1	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	First comparison value
IN2	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	Second value to compare

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" >= "Tag_Value2".

See also

Overview of the valid data types (Page 2239)

Example of detecting the fill level of a storage area (Page 4982)

FBD programming examples (Page 4978)

Basic information on FBD (Page 4937)

CMP <=: Less or equal (S7-1200, S7-1500)

Description

You can use the "Less or equal" instruction to query whether the value at input IN1 is less or equal to the value at input IN2. Both values to be compared must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the shorter string is considered smaller.

The following table shows examples of string comparisons:

IN1	IN2	RLO of the instruction
'AA'	'aa'	1
'AAA'	'a'	1
'Hello World'	'Hello World'	1
'HelloWorld'	'Hello World'	0
'BB'	'AA'	0
'AAA'	'AA'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

In comparing timer values, the RLO of the instruction is "1" if the timer at input IN1 is smaller (less recent) than or equal to the timer at input IN2.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Parameters

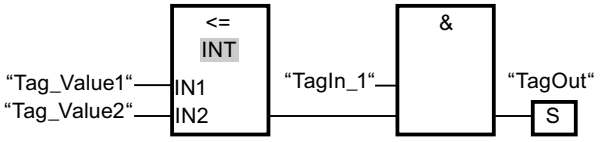
The following table shows the parameters of the "Less or equal" instruction:

Parameters	Declaration	Data type	Memory area	Description
IN1	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	First comparison value
IN2	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	Second value to compare

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" <= "Tag_Value2".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

CMP >: Greater than (S7-1200, S7-1500)

Description

You can use the "Greater than" instruction to query whether the value at input IN1 is greater than the value at input IN2. Both values to be compared must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the longer string is considered greater.

The following table shows examples of string comparisons:

IN1	IN2	RLO of the instruction
'BB'	'AA'	1
'AAA'	'AA'	1
'AA'	'aa'	0
'AAA'	'a'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

In comparing timer values, the RLO of the instruction is "1" if the timer at input IN1 is greater (more recent) than the timer at input IN2.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Parameters

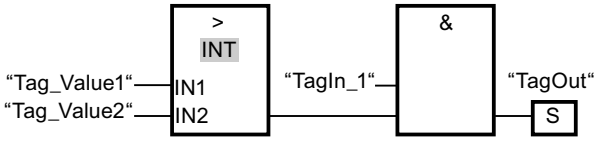
The following table shows the parameters of the "Greater than" instruction:

Parameters	Declaration	Data type	Memory area	Description
IN1	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	First comparison value
IN2	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	Second value to compare

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" > "Tag_Value2".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

CMP <: Less than (S7-1200, S7-1500)

Description

You can use the "Less than" instruction to query whether the value at input IN1 is less than the value at input IN2. Both values to be compared must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the shorter string is considered smaller.

The following table shows examples of string comparisons:

<Operand1>	<Operand2>	RLO of the instruction
'AA'	'aa'	1
'AAA'	'a'	1
'BB'	'AA'	0
'AAA'	'AA'	0

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

In comparing timer values, the RLO of the instruction is "1" if the timer at input IN1 is less (less recent) than the timer at input IN2.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Parameters

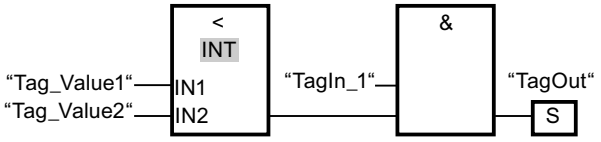
The following table shows the parameters of the "Less than" instruction:

Parameters	Declaration	Data type	Memory area	Description
IN1	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	First comparison value
IN2	Input	Bit strings, integers, floating-point numbers, character strings, timers, date and time	I, Q, M, D, L, P, or constant	Second value to compare

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn_1" has the signal state "1".
- The condition of the comparison instruction is fulfilled if "Tag_Value1" < "Tag_Value2".

See also

Overview of the valid data types (Page 2239)

Example of detecting the fill level of a storage area (Page 4982)

FBD programming examples (Page 4978)

Basic information on FBD (Page 4937)

IN_RANGE: Value within range (S7-1200, S7-1500)

Description

You can use the "Value within range" instruction to query whether of the value at VAL input is within a specific value range.

You specify the limits of the value range with the MIN and MAX inputs. The "Value within range" instruction compares the value at input VAL with the values of the inputs MIN and MAX and sends the result to the box output. If the value at input VAL satisfies the comparison MIN <=

VAL or VAL <=MAX, the box output has the signal state "1". If the comparison is not fulfilled, the signal state is "0" at the box output.

The comparison function can only execute if the values to be compared are of the same data type.

Parameters

The following table shows the parameters of the "Value within range" instruction:

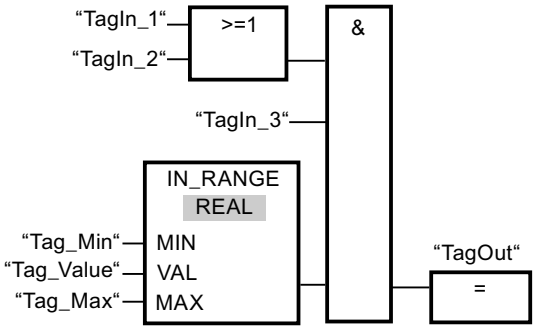
Parameters	Declaration	Data type	Memory area	Description
MIN	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Low limit of the value range
VAL	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Comparison value
MAX	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	High limit of the value range
Box output	Output	BOOL	I, Q, M, D, L	Result of the comparison

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also":

Example

The following example shows how the instruction works:



Output "TagOut" is set when the following conditions are fulfilled:

- The operands "TagIn_1" or "TagIn_2" have signal state "1".
- The operand "TagIn_3" has the signal state "1".
- The value of the operand "Tag_Value" is within the value range that is specified by the current values of the operands "Tag_Min" and "Tag_Max" (MIN <= VAL or VAL <= MAX).

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

OUT_RANGE: Value outside range (S7-1200, S7-1500)

Description

You can use the "Value outside range" instruction to query whether of the value at input VAL is outside a specific value range.

You specify the limits of the value range with the MIN and MAX inputs. The "Value outside range" instruction compares the value at input VAL with the values of the inputs MIN and MAX and sends the result to the box output. If the value at input VAL satisfies the comparison MIN > VAL or VAL >MAX, the box output has the signal state "1". The box output has the signal state "1" if a specified operand of data type REAL has an invalid value.

The box output returns the signal state "0", if the value at input VAL does not satisfy the MIN > VAL or VAL > MAX condition.

The comparison function can only execute if the values to be compared are of the same data type.

Parameters

The following table shows the parameters of the "Value outside range" instruction:

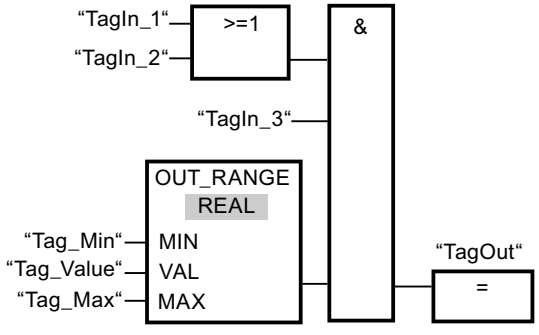
Parameters	Declaration	Data type	Memory area	Description
MIN	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Low limit of the value range
VAL	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	Comparison value
MAX	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	High limit of the value range
Box output	Output	BOOL	I, Q, M, D, L	Result of the comparison

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



Output "TagOut" is set when the following conditions are fulfilled:

- The operands "TagIn_1" and "TagIn_2" have signal state "1".
- The operand "TagIn_3" has the signal state "1".
- The value of the operand "Tag_Value" is outside the value range that is specified by the values of the operands "Tag_Min" and "Tag_Max" (MIN > VAL or VAL > MAX).

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

OK: Check validity (S7-1200, S7-1500)

Description

You can use the "Check validity" instruction to check if the value of an operand (<Operand>) is a valid floating-point number. The check is performed in every program cycle. If the operand value at the time of the query is a valid floating-point number, the output box will return the signal state"1". In all other cases, the signal state at the output of the "Check validity" instruction is "0".

You can use the "Check validity" instruction together with the EN mechanism. If you connect the instruction box to an EN enable input, the enable input is set only when the result of the validity query of the value is positive. You can use this function to ensure that an instruction is enabled only when the value of the specified operand is a valid floating-point number.

Parameters

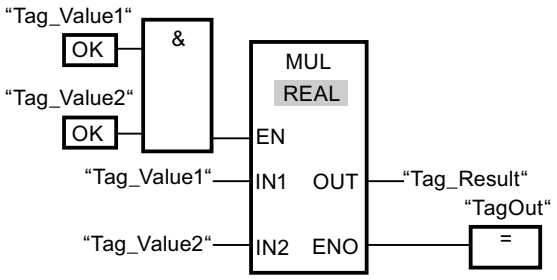
The following example shows how the "Check validity" instruction works:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	Floating-point numbers	I, Q, M, D, L	Value to be checked.

For additional information on valid data types, refer to "See also":

Example

The following example shows how the instruction works:



When the values of the operands "Tag_Value1" and "Tag_Value2" show valid floating-point numbers, the "Multiply" instruction is executed. The value of operand "Tag_Value1" is multiplied with the value of operand "Tag_Value2". The product of the multiplication is then written in the operand "Tag_Result". If the instruction is executed without errors, the TagOut operand at the "ENO" enable output is set to signal state "1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

NOT_OK: Check invalidity (S7-1200, S7-1500)

Description

You can use the "Check invalidity" instruction to check if the value of an operand (<Operand>) is an invalid floating-point number. The check is performed in every program cycle. If the operand value at the time of the query is a valid floating-point number, then the output box will return the signal state "1". In all other cases, the signal state on the output box is "0".

Parameters

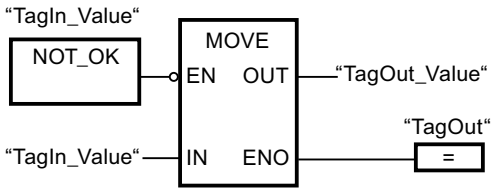
The following table shows the parameters of the instruction "Check invalidity":

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	Floating-point numbers	I, Q, M, D, L	Value to be checked.

For additional information on valid data types, refer to "See also":

Example

The following example shows how the instruction works:



When the value of operand "TagIn_Value" is an invalid floating-point number, the instruction "Move value" is not executed. The "TagOut" operand at the ENO enable output is set to the signal state "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

VARIANT (S7-1200, S7-1500)

EQ_Type: Compare data type for EQUAL with the data type of a tag (S7-1200, S7-1500)

Description

You can use the "Compare data type for EQUAL with the data type of a tag" instruction to query the data type of a tag to which a VARIANT points. You are comparing the data type of the tag at IN1 parameter, which you declared in the block interface, with the data type of a tag at IN2 parameter for "Equal".

The tag at the IN1 parameter must have the VARIANT data type. The tag at the IN2 parameter can be an elementary data type or a PLC data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

Comparison of timers, date and time

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparison of structures

For the comparison of structures, you can use the instruction "CMP ==". Anonymous structures cannot generally be compared unless they are part of the same ARRAY.

For additional information on the comparison of structures, please refer to: CMP ==: Equal (Page 2988)

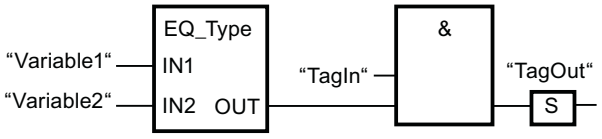
Parameters

The following table shows the parameters of the "Compare data type for EQUAL with the data type of a tag" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
IN1	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand
IN2	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Second operand

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn" has the signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the "Variable1" operand is equal to "Variable2".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

NE_Type: Compare data type for UNEQUAL with the data type of a tag (S7-1200, S7-1500)

Description

You can use the "Compare data type for UNEQUAL with the data type of a tag" instruction to query which data type a tag to which a VARIANT points does not have. You are comparing the data type of the tag at IN1 parameter, which you declared in the block interface, with the data type of a tag at IN2 parameter for "Not equal".

The tag at the IN1 parameter must have the VARIANT data type. The tag at the IN2 parameter can be an elementary data type or a PLC data type.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

Comparison of timers, date and time

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparison of structures

For the comparison of structures, you can use the instruction "CMP <>". Anonymous structures cannot generally be compared unless they are part of the same ARRAY.

For additional information on the comparison of structures, please refer to: CMP <>: Not equal (Page 2992)

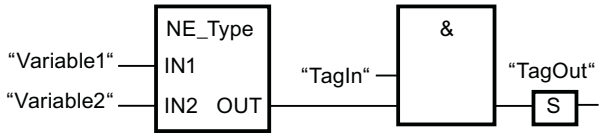
Parameters

The following table shows the parameters of the "Compare data type for UNEQUAL with the data type of a tag" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
IN1	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand
IN2	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Second operand

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn" has the signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the "Variable1" operand is not equal to "Variable2".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

EQ_ElemType: Compare data type of an ARRAY element for EQUAL with the data type of a tag (S7-1200, S7-1500)

Description

You can use the "Compare data type of an ARRAY element for EQUAL with the data type of a tag" instruction to query the data type of a tag to which a VARIANT points. You are comparing the data type of the tag at IN1 parameter, which you declared in the block interface, with the data type of a tag at IN2 parameter for "Equal".

The tag at the IN1 parameter must have the VARIANT data type. The tag at the IN2 parameter can be an elementary data type or a PLC data type.

If the data type of the VARIANT tag is an ARRAY, the data type of the ARRAY elements is compared.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

Comparison of timers, date and time

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparison of structures

For the comparison of structures, you can use the instruction "CMP ==". Anonymous structures cannot generally be compared unless they are part of the same ARRAY.

For additional information on the comparison of structures, please refer to: CMP ==: Equal (Page 2988)

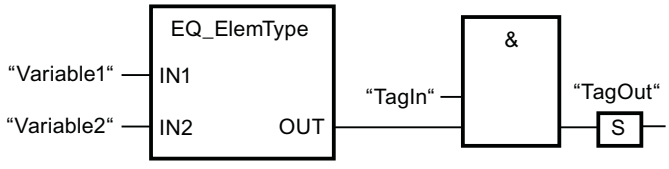
Parameters

The following table shows the parameters of the "Compare data type of an ARRAY element for EQUAL with the data type of a tag" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
IN1	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand
IN2	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Second operand

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn" has the signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the "Variable1" operand is equal to "Variable2".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

NE_ElemType: Compare data type of an ARRAY element for UNEQUAL with the data type of a tag (S7-1200, S7-1500)

Description

You can use the "Compare data type of an ARRAY element for UNEQUAL with the data type of a tag" instruction to query which data type a tag does not have to which a VARIANT points. You are comparing the data type of the tag at IN1 parameter, which you declared in the block interface, with the data type of a tag at IN2 parameter for "Not equal".

The tag at the IN1 parameter must have the VARIANT data type. The tag at the IN2 parameter can be an elementary data type or a PLC data type.

If the data type of the VARIANT tag is an ARRAY, the data type of the ARRAY elements is compared.

If the condition of the comparison is fulfilled, the instruction returns the result of logic operation (RLO) "1". If the comparison condition is not fulfilled, the instruction returns RLO "0".

Comparison of timers, date and time

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparison of structures

For the comparison of structures, you can use the instruction "CMP <>". Anonymous structures cannot generally be compared unless they are part of the same ARRAY.

For additional information on the comparison of structures, please refer to: CMP <>: Not equal (Page 2992)

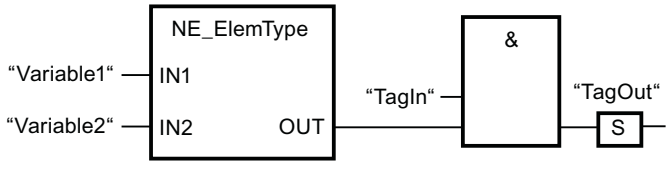
Parameters

The following table shows the parameters of the "Compare data type of an ARRAY element for UNEQUAL with the data type of a tag" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
IN1	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand
IN2	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Second operand

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operand "TagIn" has the signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the "Variable1" operand is not equal to "Variable2".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

IS_NULL: Check for EQUALS NULL pointer (S7-1200, S7-1500)

Description

You use the instruction "Check for EQUALS NULL pointer" to query whether the VARIANT points to a NULL pointer and therefore does not point to an object.

The <operand> must have the VARIANT data type.

Note

VARIANT tag points to an ANY pointer

If the VARIANT tag points to an ANY pointer, the instruction always returns the result RLO = "0" even if the ANY pointer is NULL.

Parameters

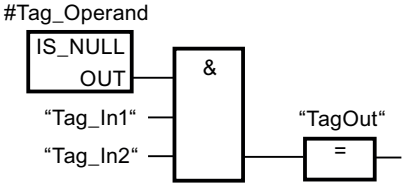
The following table shows the parameters of the "Check for EQUALS NULL pointer" instruction:

Parameters	Declaration	Data type	Memory area	Description
<operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Operand that is compared for EQUALS NULL

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "Tag_In1" and "Tag_In2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the #Tag_Operand operand does not point to an object.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

NOT_NULL: Check for UNEQUALS NULL pointer (S7-1200, S7-1500)

Description

You can use the instruction "Check for UNEQUALS NULL pointer" to query whether the VARIANT does not point to a NULL pointer and therefore points to an object.

The <operand> must have the VARIANT data type.

Note

VARIANT tag points to an ANY pointer

If the VARIANT tag points to an ANY pointer, the instruction always returns the result RLO = "1" even if the ANY pointer is NULL.

Parameters

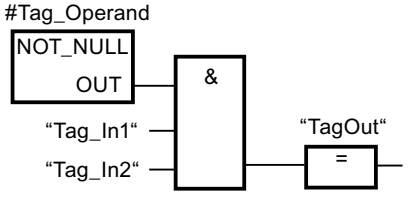
The following table shows the parameters of the "Check for UNEQUALS NULL pointer" instruction:

Parameters	Declaration	Data type	Memory area	Description
<operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Operand that is compared for UNEQUALS NULL

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "Tag_In1" and "Tag_In2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the #Tag_Operand operand points to an object.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

IS_ARRAY: Check for ARRAY (S7-1200, S7-1500)

Description

You can use the "Check for ARRAY" instruction to query whether the VARIANT points to a tag of the ARRAY data type.

The <operand> must have the VARIANT data type.

Parameters

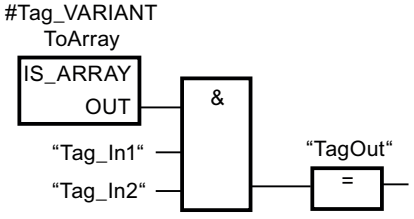
The following table shows the parameters of the "Check for ARRAY" instruction:

Parameters	Declaration	Data type	Memory area	Description
<operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Operand that is queried for ARRAY

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The "TagOut" output is set when the following conditions are fulfilled:

- The operands "Tag_In1" and "Tag_In2" have signal state "1".
- The condition of the comparison instruction is fulfilled, i.e. the "#Tag_VARIANTToArray" operand is of the ARRAY data type.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Math functions (S7-1200, S7-1500)

CALCULATE: Calculate (S7-1200, S7-1500)

Description

The "Calculate" instruction is used to define and execute an expression for the calculation of mathematical operations or complex logic operations depending on the selected data type.

You can select the data type of the instruction from the "???" drop-down list of the instruction box. Depending on the selected data type, you can combine the functionality of specific instructions to execute a complex calculation. The expression to be calculated is specified via a dialog you can open via the "Calculator" icon at the top of the instruction box. The expression can contain the names of the input parameters and the syntax of the instructions. It is not permitted to specify operand names or operand addresses.

In its initial state, the instruction box contains at least 2 inputs (IN1 and IN2). The number of inputs can be extended. The inserted inputs are numbered in ascending order in the box.

The values of the inputs are use to execute the specified expression. Not all defined inputs have to be used in the expression. The result of the instruction is transferred to the box output OUT.

Note

If one of the mathematical operations fails in the expression, then no result is transferred to the OUT output and the ENO enable output has the signal state "1".

If, in the expression, you use inputs that are not available in the box, these inputs are automatically inserted. Provided that there are no gaps in the numbering of the inputs that are to be newly defined in the expression. You cannot, for example, use the input IN4 in the expression if the input IN3 is not defined.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The result or an interim result of the "Calculate" instruction is outside the range permitted for the data type specified at output OUT.
- A floating-point number has an invalid value.
- An error occurred during the execution of one of the instructions specified in the expression.

The following table shows the instructions that, depending on the selected data type, can be combined and executed in the expression of the "Calculate" instruction:

Data type	Instruction	Syntax	Example
Bit strings	AND: AND logic operation	AND	IN1 AND IN2 OR IN3
	OR: OR logic operation	OR	
	XOR: EXCLUSIVE OR logic operation	XOR	
	INV: Create ones complement	NOT	
	SWAP: Swap [1)]	SWAP	

Data type	Instruction	Syntax	Example
Integers	ADD: Add	+	(IN1 + IN2) * IN3; (ABS(IN2)) * (ABS(IN1))
	SUB: Subtract	-	
	MUL: Multiply	*	
	DIV: Divide	/	
	MOD: Return remainder of division	MOD	
	INV: Create ones complement	NOT	
	NEG: Create twos complement	-(in1)	
	ABS: Form absolute value	ABS( )	
Floating-point numbers	ADD: Add	+	((SIN(IN2) * SIN(IN2) + (SIN(IN3) * SIN(IN3)) / IN3)); (SQR(SIN(IN2)) + (SQR(COS(IN3)) / IN2))
	SUB: Subtract	-	
	MUL: Multiply	*	
	DIV: Divide	/	
	EXPT: Exponentiate	**	
	ABS: Form absolute value	ABS( )	
	SQR: Form square	SQR( )	
	SQRT: Form square root	SQRT( )	
	LN: Form natural logarithm	LN( )	
	EXP: Form exponential value	EXP( )	
	FRAC: Return fraction	FRAC( )	
	SIN: Form sine value	SIN( )	
	COS: Form cosine value	COS( )	
	TAN: Form tangent value	TAN( )	
	ASIN: Form arcsine value	ASIN( )	
	ACOS: Form arccosine value	ACOS( )	
	ATAN: Form arctangent value	ATAN( )	
	NEG: Create twos complement	-(in1)	
	TRUNC: Truncate numerical value	TRUNC( )	
	ROUND: Round numerical value	ROUND( )	
	CEIL: Generate next higher integer from floating-point number	CEIL( )	
	FLOOR: Generate next lower integer from floating-point number	FLOOR( )	

[1] Not possible for data type BYTE.

Parameters

The following table shows the parameters of the instruction "Calculate":

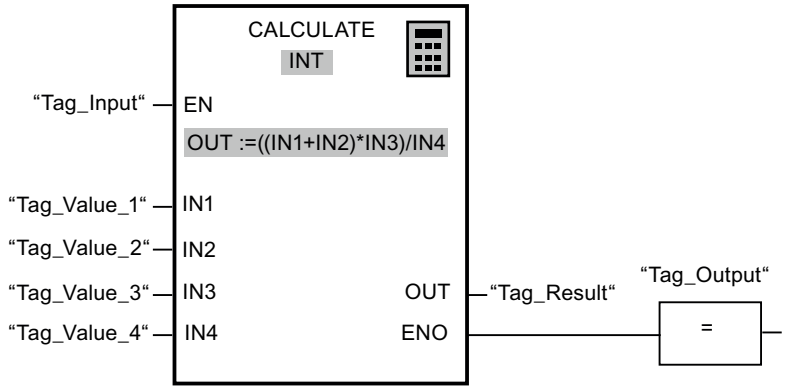
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
IN1	Input	Bit strings, integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First available input
IN2	Input	Bit strings, integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second available input
INn	Input	Bit strings, integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Additionally inserted inputs
OUT	Output	Bit strings, integers, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Output to which the end result is to be transferred.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN1	Tag_Value_1	4
IN2	Tag_Value_2	4
IN3	Tag_Value_3	3
IN4	Tag_Value_4	2
OUT	Tag_Result	12

If the "Tag_Input" has the signal state "1", the instruction is executed. The value of operand "Tag_Value_1" is added to the value of operand "Tag_Value_2". The sum is multiplied with the value of the operand "Tag_Value_3". The product is divided by the value of the operand "Tag_Value_4". The quotient is transferred as end result to the operand "Tag_Result" at the OUT output of the instruction. If no errors occur during the execution of the individual instructions, output ENO and the operand "Tag_Output" are set to "1".

See also

Overview of the valid data types (Page 2239)

Example of calculating an equation (Page 4986)

Basic information on FBD (Page 4937)

ADD: Add (S7-1200, S7-1500)

Description

You can use the "Add" instruction to add the value at input IN1 to the value at input IN2 and query the sum at output OUT (OUT := IN1+IN2).

In its initial state the instruction box contains at least 2 inputs (IN1 and IN2). The number of inputs can be extended. The inserted inputs are numbered in ascending order in the box. During the execution of the instruction, the values of all available input parameters are added. The sum is stored at output OUT.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The result of the instruction is outside the range permitted for the data type specified at output OUT.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the instruction "Add":

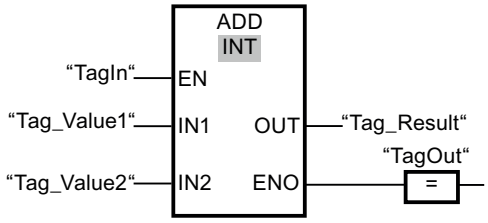
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First number to be added
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second number to be added
INn	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Optional input values, which are added.
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Sum

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also":

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" is added to the value of operand "Tag_Value2". The result of the addition is stored in the operand "Tag_Result". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SUB: Subtract (S7-1200, S7-1500)

Description

You can use the "Subtract" instruction to subtract the value at input IN2 from the value at input IN1 and query the difference at output OUT (OUT := IN1-IN2).

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The result of the instruction is outside the range permitted for the data type specified at output OUT.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the instruction "Subtract":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Minuend

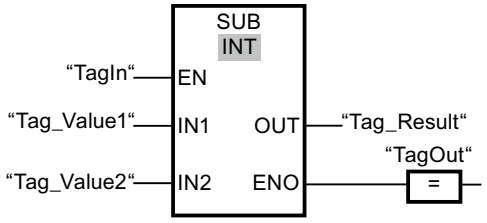
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Subtrahend
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Difference

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value2" is subtracted from the value of operand "Tag_Value1". The result of the subtraction is stored in the operand "Tag_Result". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

MUL: Multiply (S7-1200, S7-1500)

Description

You can use the "Multiply" instruction to multiply the value at input IN1 with the value at input IN2 and query the total at output OUT (OUT := IN1*IN2).

In its initial state the instruction box contains at least 2 inputs (IN1 and IN2). The number of inputs can be extended. The inserted inputs are numbered in ascending order in the box. When the instruction is executed, the values of all available input parameters are multiplied. The product is stored at the OUT output.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The result is outside the range permitted for the data type specified at output OUT.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the instruction "Multiply":

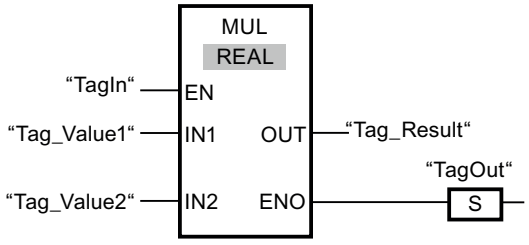
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First value for multiplication
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second value for multiplication
INn	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Optional input values, which are multiplied.
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Product

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" is multiplied with the value of operand "Tag_Value2". The result of the multiplication is stored in the operand "Tag_Result". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

DIV: Divide (S7-1200, S7-1500)

Description

You can use the "Divide" instruction to divide the value at input IN1 by the value at input IN2 and query the quotient at output OUT (OUT := IN1/IN2).

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The result of the instruction is outside the range permitted for the data type specified at output OUT.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the instruction "Divide":

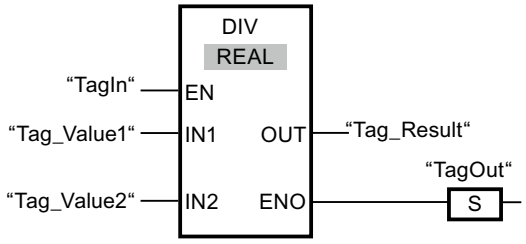
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Dividend
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Divisor
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Quotient value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" is divided by the value of operand "Tag_Value2". The result of the division is stored in the operand "Tag_Result". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

MOD: Return remainder of division (S7-1200, S7-1500)

Description

You can use the "Return remainder of division" instruction to divide the value at input IN1 by the value at input IN2 and query the remainder of division at output OUT.

Parameters

The following table shows the parameters of the "Return remainder of division" instruction:

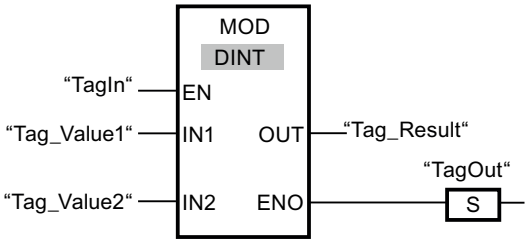
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Dividend
IN2	Input	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Divisor
OUT	Output	Integers	I, Q, M, D, L, P	I, Q, M, D, L, P	Remainder of division

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" is divided by the value of operand "Tag_Value2". The remainder of division is stored in operand "Tag_Result". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

NEG: Create twos complement (S7-1200, S7-1500)

Description

You can use the "Create twos complement" instruction to change the sign of the value at input IN and query the result at output OUT. If there is a positive value at input IN, for example, the negative equivalent of this value is sent to output OUT.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The result of the instruction is outside the range permitted for the data type specified at output OUT.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Create twos complement" instruction:

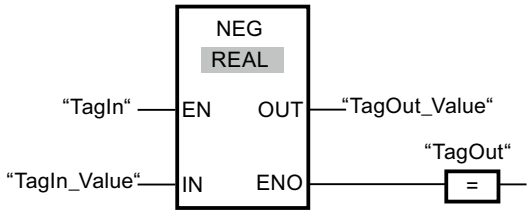
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value
OUT	Output	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Twos complement of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The sign of the value at input "TagIn_Value" is changed and the result is stored at output "TagOut_Value". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

INC: Increment (S7-1200, S7-1500)

Description

You can use the "Increment" instruction to change the value of the operand at parameter IN/OUT to the next higher value and query the result.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Increment" instruction:

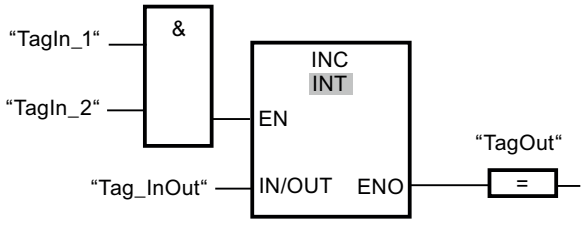
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN/OUT	InOut	Integers	I, Q, M, D, L	I, Q, M, D, L	Value to be incremented.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the operands TagIn_1 and TagIn_2 have the signal state "1", the value of the operand "Tag_InOut" is incremented by one and the output "TagOut" is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

DEC: Decrement (S7-1200, S7-1500)

Description

You can use the "Decrement" instruction to change the value of the operand at parameter IN/OUT to the next lower value and query the result.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the instruction "Decrement":

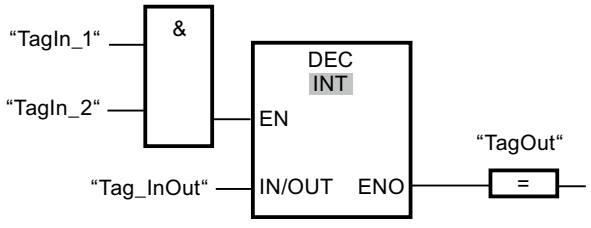
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN/OUT	InOut	Integers	I, Q, M, D, L	I, Q, M, D, L	Value to be decremented.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the operands "TagIn_1" and "TagIn_2" have the signal state "1", the value of the operand "Tag_InOut" is decremented by one and the output "TagOut" is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

ABS: Form absolute value (S7-1200, S7-1500)

Description

You can use the "Form absolute value" instruction to calculate the absolute value of the value specified at input IN. The result of the instruction is output at the OUT output and can be queried there.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the instruction "Form absolute value":

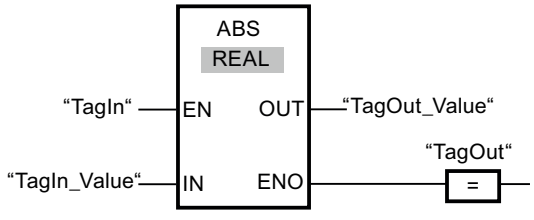
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value
OUT	Output	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Absolute value of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	TagIn_Value	-6.234
OUT	TagOut_Value	6.234

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction calculates the absolute value of the value at input "TagIn_Value" and sends the result to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

MIN: Get minimum (S7-1200, S7-1500)

Description

The "Get minimum" instruction compares the values at the available inputs and writes the lowest value to the OUT output. The number of inputs can be expanded at the instruction box by additional inputs. The inputs are numbered in ascending order in the box.

In its initial state the instruction contains at least two inputs (IN1 and IN2) and no more than 100 inputs.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The implicit conversion of the data types fails during execution of the instruction.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Get minimum" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First input value
IN2	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second input value

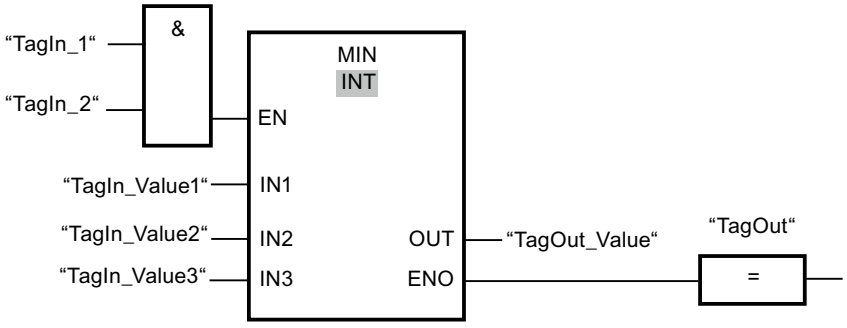
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
INn	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Additionally inserted inputs whose values are to be compared
OUT	Output	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P	I, Q, M, D, L, P	Result
If the IEC check is not enabled, you can also use tags of the data type TIME, LTIME, TOD, LTOD, DATE and LDT by selecting a bit string of the same length as the data type. (e.g. instead of TIME => DINT, UDINT or DWORD = 32 bits)					

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN1	TagIn_Value1	12222
IN2	TagIn_Value2	14444
IN3	TagIn_Value3	13333
OUT	TagOut_Value	12222

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The instruction compares the values of the specified operands and copies the lowest value ("TagIn_Value1") to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

MAX: Get maximum (S7-1200, S7-1500)

Description

The "Get maximum" instruction compares the values at the available inputs and writes the highest value to the OUT output. The number of inputs can be expanded at the instruction box by additional inputs. The inputs are numbered in ascending order in the box.

In its initial state the instruction contains at least two inputs (IN1 and IN2) and no more than 100 inputs.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The implicit conversion of the data types fails during execution of the instruction.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Get maximum" instruction:

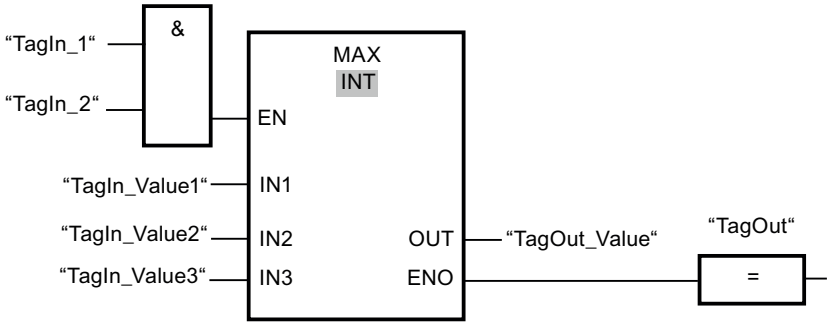
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First input value
IN2	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second input value
INn	Input	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Additionally inserted inputs whose values are to be compared.
OUT	Output	Integers, floating-point numbers, DTL, DT	I, Q, M, D, L, P	I, Q, M, D, L, P	Result
If the IEC check is not enabled, you can also use tags of the data type TIME, LTIME, TOD, LTOD, DATE and LDT by selecting a bit string of the same length as the data type. (e.g. instead of TIME => DINT, UDINT or DWORD = 32 bits)					

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN1	TagIn_Value1	12222
IN2	TagIn_Value2	14444
IN3	TagIn_Value3	13333
OUT	TagOut_Value	14444

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The instruction compares the values of the specified operands and copies the highest value ("TagIn_Value2") to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

LIMIT: Set limit value (S7-1200, S7-1500)

Description

You can use the "Set limit value" instruction to limit the value at input IN to the values at the inputs MN and MX. If the value at the IN input meets the MN condition <= IN <= MX, it is copied to the OUT output. If the condition is not fulfilled and the input value IN is below the low limit MN, output OUT is set to the value of the input MN. If the high limit MX is exceeded, output OUT is set to the value of the input MX.

If the value at the MN input is greater than at the MX input, the result is the value specified at the IN parameter and the enable output ENO is "0".

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The specified tags are not of the same data type.

- An operand has an invalid value.
- The value at the MN input is greater than the value at the MX input.

Parameters

The following table shows the parameters of the "Set limit value" instruction:

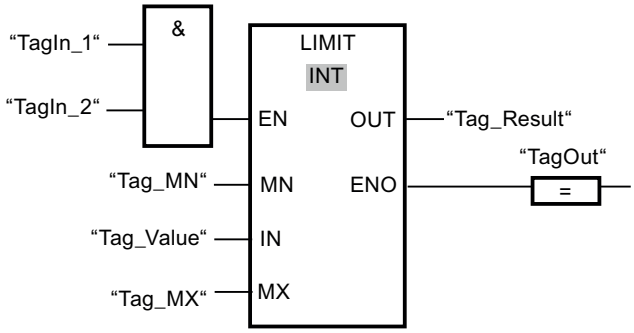
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
MN	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL, DT	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Low limit
IN	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL, DT	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value
MX	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL, DT	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DTL, DT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	High limit
OUT	Output	Integers, floating-point numbers, TIME, TOD, DATE, DTL, DT	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DTL, DT	I, Q, M, D, L, P	I, Q, M, D, L, P	Result
The data types TOD, LTOD, DATE, and LDT can only be used if the IEC test is not enabled.						

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
MN	Tag_MN	12000
IN	Tag_Value	8000
MX	Tag_MX	16000
OUT	Tag_Result	12000

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The value of operand "Tag_Value" is compared with the values of operands "Tag_MN" and "Tag_MX". Since the value of the operand "Tag_Value" is less than the low limit, the value of the operand "Tag_MN" is copied to the "Tag_Result" output. If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SQR: Form square (S7-1200, S7-1500)

Description

The instruction "Form square" is used to square the value of a floating-point number at input IN and to write the result to output OUT.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form square":

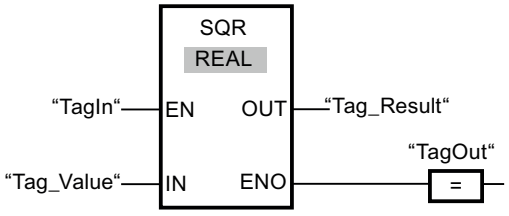
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Square of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_Value	5.0
OUT	Tag_Result	25.0

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction squares the value of the operand "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SQRT: Form square root (S7-1200, S7-1500)

Description

The instruction "Form square root" is used form the square root from the value of the floating-point number at input IN and to write the result to output OUT. The instruction has a positive result if the input value is greater than zero. If input values are less than zero, output OUT returns an invalid floating-point number. If the value at input IN is "0", the result is also "0".

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.
- The value at input IN is negative.

Parameters

The following table shows the parameters of the instruction "Form square root":

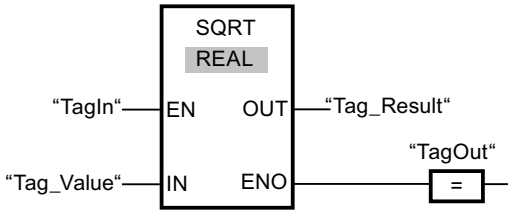
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value
OUT	Output	Floating-point numbers	I, Q, M, D, L	I, Q, M, D, L	Square root of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_Value	25.0
OUT	Tag_Result	5.0

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction calculates the square root of the operand "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

LN: Form natural logarithm (S7-1200, S7-1500)

Description

You can use the "Form natural logarithm" instruction to calculate the natural logarithm to base e (e = 2.718282) of the value at input IN. The result is sent to output OUT and can be queried there. The instruction has a positive result if the input value is greater than zero. If input values are less than zero, output OUT returns an invalid floating-point number.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.
- The value at input IN is negative.

Parameters

The following table shows the parameters of the instruction "Form natural logarithm":

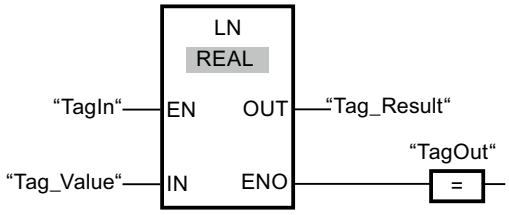
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Natural logarithm of the input value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction forms the natural logarithm of the value at input "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

EXP: Form exponential value (S7-1200, S7-1500)

Description

You can use the "Form exponential value" instruction to calculate the exponent from the base e (e = 2.718282) and the value specified at input IN. The result is provided at output OUT and can be queried there (OUT = e^{IN}).

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form exponential value":

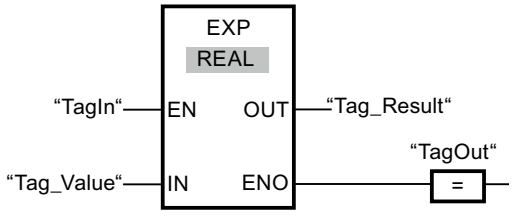
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Exponential value of input value IN

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction forms the exponent from base e and the value of the operand "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SIN: Form sine value (S7-1200, S7-1500)

Description

You can use the "Form sine value" instruction to calculate the sine of an angle. The size of the angle is specified in the radian measure at input IN. The result of the instruction is output at the OUT output and can be queried there.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form sine value":

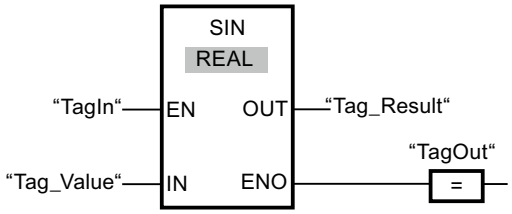
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Size of the angle in radians
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Sine of the specified angle

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_Value	+1.570796 (π/2)
OUT	Tag_Result	1.0

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction calculates the sine of the angle specified at input "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

COS: Form cosine value (S7-1200, S7-1500)

Description

You can use the "Form cosine value" instruction to calculate the cosine of an angle. The size of the angle is specified in the radian measure at input IN. The result of the instruction is output at the OUT output and can be queried there.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form cosine value":

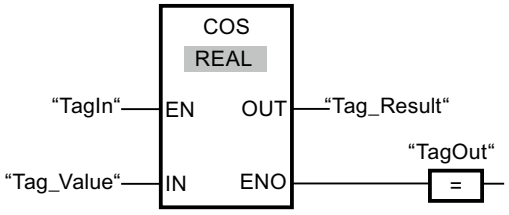
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Size of the angle in radians
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Cosine of the specified angle

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_Value	+1.570796 (π/2)
OUT	Tag_Result	0

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction calculates the cosine of the angle specified at input "Tag_Value" and sends the result to output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

TAN: Form tangent value (S7-1200, S7-1500)

Description

You can use the "Form tangent value" instruction to calculate the tangent of an angle. The size of the angle is specified in the radian measure at input IN. The result of the instruction is provided at output OUT and can be queried there.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form tangent value":

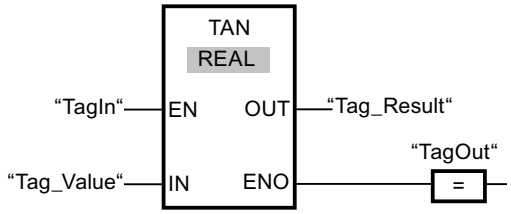
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Size of angle in the radian measure
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Tangent of the specified angle

You can select the data type of the instruction from the "<???>" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	+3.141593 (π)
OUT	Tag_Result	0

If the operand "TagIn" has the signal state "1", the "Form tangent value" instruction is executed. The instruction calculates the tangent of the angle specified at input "Tag_Value" and stores the result at output "Tag_Result". If no errors occur during the execution of the instruction, the output "TagOut" is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

ASIN: Form arcsine value (S7-1200, S7-1500)

Description

You can use the "Form arcsine value" instruction to calculate the size of the angle from the sine value specified at input IN, which corresponds to this value. Only valid floating-point numbers within the range -1 to +1 can be specified at input IN. The calculated angle size is output in radians at the OUT output and can range in value from -π/2 to +π/2.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.
- The value at input IN is outside the permitted range (-1 to +1).

Parameters

The following table shows the parameters of the instruction "Form arcsine value":

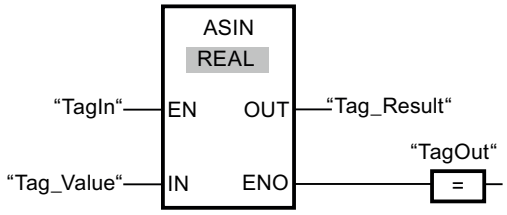
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Sine value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Size of the angle in radians

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_Value	1.0
OUT	Tag_Result	+1.570796 (π/2)

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction calculates the size of the angle corresponding to the sine value at input "Tag_Value". The result of the instruction is stored at output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

ACOS: Form arccosine value (S7-1200, S7-1500)

Description

You can use the "Form arccosine value" instruction to calculate the size of the angle from the cosine value specified at input IN, which corresponds to this value. Only valid floating-point numbers within the range -1 to +1 can be specified at input IN. The calculated angle size is output in radians at output OUT and can range in value from 0 to +π.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.
- The value at input IN is outside the permitted range (-1 to +1).

Parameters

The following table shows the parameters of the instruction "Form arccosine value":

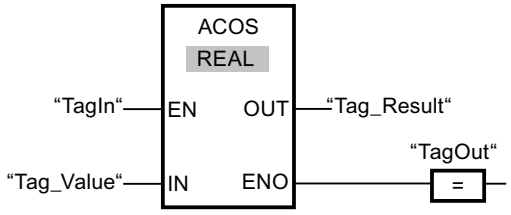
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Cosine value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Size of the angle in radians

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_Value	0
OUT	Tag_Result	+1.570796 (π/2)

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction calculates the size of the angle corresponding to the cosine value at input "Tag_Value". The result of the instruction is stored at output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

ATAN: Form arctangent value (S7-1200, S7-1500)

Description

You can use the "Form arctangent value" instruction to calculate the size of the angle from the tangent value specified at input IN, which corresponds to this value. It is only permitted to specify valid floating-point numbers (or -NaN/+NaN) at input IN. The calculated angle size is output in radians at the OUT output and can range in value from -π/2 to +π/2.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input IN is not a valid floating-point number.

Parameters

The following table shows the parameters of the instruction "Form arctangent value":

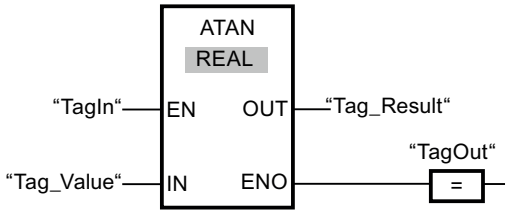
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Tangent value
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Size of the angle in radians

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_Value	1.0
OUT	Tag_Result	+0.785398 (π/4)

If the "TagIn" operand has signal state "1", the instruction is executed. The instruction calculates the size of the angle corresponding to the tangent value at input "Tag_Value". The result of the instruction is stored at output "Tag_Result". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Invalid floating-point numbers (Page 2259)

Basic information on FBD (Page 4937)

FRAC: Return fraction (S7-1200, S7-1500)

Description

You can use the "Return fraction" instruction to determine the decimal places of the value at the IN input. The result of the query is stored at output OUT and can be queried there. If the input IN has e.g. the value 123.4567, the output OUT has the value 0.4567.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- Errors occur during the execution of the instruction, for example there is no valid floating-point number at the input.

Parameters

The following table shows the parameters of the "Return fraction" instruction:

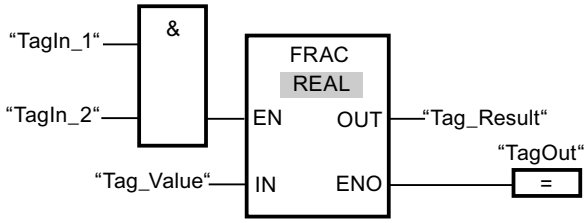
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value, whose decimal places are to be determined.
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Decimal places of the input value at input IN

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_Value	2.555
OUT	Tag_Result	0.555

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is started. The decimal places from the value at the operand "Tag_Value" are copied to the operand "Tag_Result". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

EXPT: Exponentiate (S7-1200, S7-1500)

Description

You can use the "Exponentiate" instruction to raise the value at the input IN1 by a power specified with the value at input IN2. The result of the instruction is stored at output OUT and can be queried there (OUT = $IN1^{IN2}$).

The value at input IN1 must be a valid floating-point number. Integers are also allowed for setting the input IN2.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- Errors occur during the instruction processing, for example, if there is an overflow.

Parameters

The following table shows the parameters of the instruction "Exponentiate":

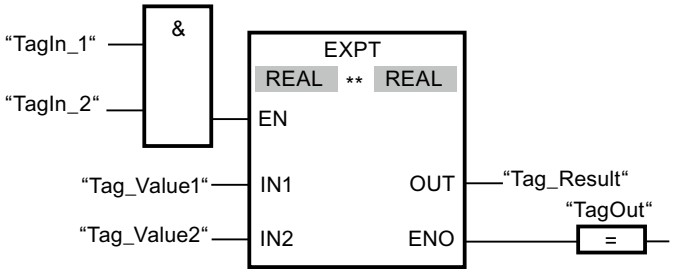
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Base value
IN2	Input	Integers, floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Value with which the base value is exponentiated
OUT	Output	Floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Result

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the operands "TagIn_1" and "TagIn_2" have signal state "1", the "Exponentiate" instruction is executed. The value of operand "Tag_Value1" is raised by the power of the value of the operand "Tag_Value2". The result is stored at output "Tag_Result". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Move operations (S7-1200, S7-1500)

MOVE: Move value (S7-1200, S7-1500)

Description

You use the "Move value" instruction to transfer the content of the operand at the IN input to the operand at the OUT1 output. The transfer is always made in the direction of ascending address.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The data type at the IN parameter does not correspond to the specified data type at the OUT1 parameter.

The following table lists the possible transfers for the S7-1200 CPU series:

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
BYTE	BYTE, WORD, DWORD	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE ,TOD, CHAR
WORD	WORD, DWORD	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD, CHAR
DWORD	DWORD	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, CHAR
SINT	SINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
USINT	USINT, UINT, UDINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
INT	INT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
UINT	UINT, UDINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
DINT	DINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
UDINT	UDINT	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME, DATE, TOD
REAL	REAL	DWORD, REAL
LREAL	LREAL	LREAL
TIME	TIME	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TIME
DATE	DATE	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, DATE
TOD	TOD	BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, TOD
DTL	DTL	DTL
CHAR	CHAR	BYTE, WORD, DWORD, CHAR, Character of a string[1)]

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
WCHAR	WCHAR	BYTE, WORD, DWORD, CHAR, WCHAR, character of a character string[1)]
Character of a string[1)]	Character of a string	CHAR, WCHAR, character of a string
ARRAY[2)]	ARRAY	ARRAY
STRUCT	STRUCT	STRUCT
PLC data type (UDT)	PLC data type (UDT)	PLC data type (UDT)
IEC_TIMER	IEC_TIMER	IEC_TIMER
IEC_SCOUNT-ER	IEC_SCOUNTER	IEC_SCOUNTER
IEC_US-COUNTER	IEC_USCOUNTER	IEC_USCOUNTER
IEC_COUNTER	IEC_COUNTER	IEC_COUNTER
IEC_UCOUNT-ER	IEC_UCOUNTER	IEC_UCOUNTER
IEC_DCOUNT-ER	IEC_DCOUNTER	IEC_DCOUNTER
IEC_UD-COUNTER	IEC_UDCOUNTER	IEC_UDCOUNTER

The following table lists the possible transfers for the S7-1500 CPU series:

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
BYTE	BYTE, WORD, DWORD, LWORD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE ,TOD, LTOD, CHAR
WORD	WORD, DWORD, LWORD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, S5TIME, TIME, LTIME, LDT, DATE, TOD, LTOD, CHAR
DWORD	DWORD, LWORD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, TIME, LTIME, LDT, DATE, TOD, LTOD, CHAR
LWORD	LWORD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, LREAL, TIME, LTIME, LDT, DATE, TOD, LTOD, CHAR
SINT	SINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
USINT	USINT, UINT, UDINT, ULINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
INT	INT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
UINT	UINT, UDINT, ULINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD. LTOD
DINT	DINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
UDINT	UDINT, ULINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
LINT	LINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
ULINT	ULINT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME, LTIME, LDT, DATE, TOD, LTOD
REAL	REAL	DWORD, REAL
LREAL	LREAL	LWORD, LREAL
S5TIME	S5TIME	WORD, S5TIME
TIME	TIME	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TIME
LTIME	LTIME	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, LTIME
DATE	DATE	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, DATE
DT	DT	DT
LDT	LDT	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, LDT
TOD	TOD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, TOD
LTOD	LTOD	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, LTOD
DTL	DTL	DTL
CHAR	CHAR	BYTE, WORD, DWORD, LWORD, CHAR, Character of a string[1)]
WCHAR	WCHAR	BYTE, WORD, DWORD, LWORD, CHAR, WCHAR, Character of a string[1)]
Character of a string[1)]	Character of a string	CHAR, WCHAR, character of a string
ARRAY[2)]	ARRAY	ARRAY
STRUCT	STRUCT	STRUCT
COUNTER	COUNTER, WORD, INT	WORD, DWORD, INT, UINT, DINT, UDINT
TIMER	TIMER, WORD, INT	WORD, DWORD, INT, UINT, DINT, UDINT
PLC data type (UDT)	PLC data type (UDT)	PLC data type (UDT)
IEC_TIMER	IEC_TIMER	IEC_TIMER
IEC_LTIMER	IEC_LTIMER	IEC_LTIMER

Source (IN)	Destination (OUT1)	
	With IEC check	Without IEC check
IEC_SCOUNT-ER	IEC_SCOUNTER	IEC_SCOUNTER
IEC_US-COUNTER	IEC_USCOUNTER	IEC_USCOUNTER
IEC_COUNTER	IEC_COUNTER	IEC_COUNTER
IEC_UCOUNT-ER	IEC_UCOUNTER	IEC_UCOUNTER
IEC_DCOUNT-ER	IEC_DCOUNTER	IEC_DCOUNTER
IEC_UD-COUNTER	IEC_UDCOUNTER	IEC_UDCOUNTER
IEC_LCOUNT-ER	IEC_LCOUNTER	IEC_LCOUNTER
IEC_UL-COUNTER	IEC_ULCOUNTER	IEC_ULCOUNTER

[1] You can also use the "Move value" instruction to transfer individual characters of a string to operands of the CHAR or WCHAR data type. The number of the character to be transferred is specified in square brackets next to the operand name. "MyString[2]", for example, transfers the second character of the "MyString" string. The transfer of the operand of the data type CHAR or WCHAR to the individual character of a string is also possible. You can also replace a specific character of a string with the character of another string.

[2] Transferring entire arrays (ARRAY) is possible only when the array components of the operands at input IN and at output OUT1 are of the same data type.

If the bit length of the data type at input IN exceeds the bit length of the data type at output OUT1, the higher-order bits of the source value are lost. If the bit length of the data type at input IN is less than the bit length of the data type at output OUT1, the higher-order bits of the destination value will be overwritten with zeros.

In its initial state, the instruction box contains 1 output (OUT1). The number of outputs can be extended. The added outputs are numbered in ascending order on the box. During the execution of the instruction, the content of the operand at the input IN is transferred to all available outputs. The instruction box cannot be extended if structured data types (DTL, STRUCT, ARRAY) or characters of a string are transferred.

You can also use the "Move block" (MOVE_BLK) and "Move block uninterruptible" (UMOVE_BLK) instructions to move operands of the ARRAY data type. Operand of the data type STRING or WSTRING can be copied with the instruction "Move character string" (S_MOVE).

Parameters

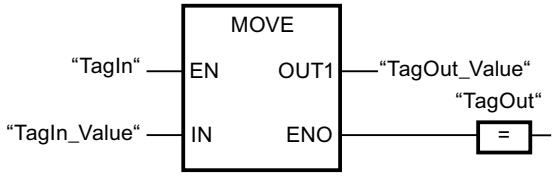
The following table shows the parameters of the "Move value" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers, floating-point numbers, timers, date and time, CHAR, WCHAR, STRUCT, ARRAY, IEC data types, PLC data type (UDT)	Bit strings, integers, floating-point numbers, timers, date and time, CHAR; WCHAR, STRUCT, ARRAY, TIMER, COUNTER, IEC data type, PLC data type (UDT)	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Element used to overwrite the destination address.
OUT1	Output	Bit strings, integers, floating-point numbers, timers, date and time, CHAR, WCHAR, STRUCT, ARRAY, IEC data types, PLC data type (UDT)	Bit strings, integers, floating-point numbers, timers, date and time, CHAR; WCHAR, STRUCT, ARRAY, TIMER, COUNTER, IEC data type, PLC data type (UDT)	I, Q, M, D, L	I, Q, M, D, L	Destination address

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	TagIn_Value	0011 1111 1010 1111
OUT1	TagOut_Value	0011 1111 1010 1111

If the "TagIn" operand returns signal state TRUE, the instruction is executed. The instruction moves the contents of the "TagIn_Value" operand to the "TagOut_Value" operand. If the instruction is executed without errors, the ENO and "TagOut" enable outputs are set to signal state TRUE.

Note

You can find more information on the MOVE instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

Removing instruction inputs and outputs (Page 4968)

MOVE_BLK: Move block (Page 3069)

UMOVE_BLK: Move block uninterruptible (Page 3086)

S_MOVE: Move character string (Page 3490)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

Adding additional inputs and outputs to FBD elements (Page 4967)

Deserialize: Deserialize (S7-1200, S7-1500)

Description

You can use the "Deserialize" instruction to convert back the sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type> and to fill its entire contents.

The memory area in which the sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type> is located must have the ARRAY of BYTE data type and be declared with standard access in version 1.0. Optimized memory areas also permitted as of version 2.0. The capacity of the standard memory area is 64 KB. Make sure that there is enough memory space prior to the conversion. If the memory area is filled using the instruction "Serialize" and it is possible that fill bytes were inserted, their conversion is not taken into account.

It is recommended to define the low limit of the ARRAY with "0", because then the index within ARRAY corresponds to the value of the POS parameter, e.g. ARRAY[0] = POS 0. The description and example below have been formed based on this.

The instruction enables you to convert multiple sequential representations of converted data back to their original state step-by-step.

If you only want to convert back a single sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type>, you can also directly use the instruction "TRCV: Receive data via communication connection".

Size of the memory area

The alignment rules mean that simple structures in the optimized memory area do not contain filling bytes. This results in a structure in the optimized memory area being smaller than in the standard memory area. ARRAYs of structures and structures in which other structures are located contain filling bytes. There is no general rule as to the memory area in which a composed structure requires more space.

Applies to CPUs of the S7-1500 series:

For a block with the block property "Optimized block access", the BOOL has a length of 1 byte. A structure that consists largely of the data type BOOL can therefore be larger in the optimized memory area than in the standard memory area. Composed structures with a low proportion of BOOL data types are smaller in the optimized memory area than in the standard memory area.

Optimized memory area

To deserialize larger structures, you can declare the memory area for sequential representation with optimized access for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0. The content of the sequential representation remains unchanged, as for a standard memory area. Only symbolic access to the bytes in the ARRAY is possible.

Parameters

The following table shows the parameters of the "Deserialize" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRC_ARRAY	Input	VARIANT ARRAY[*] of BYTE [1)]	D, block interface of an FB (The sections Input, Output, Static and Temp are possible.)	Tag of ARRAY of BYTE data type in which the generate data flow is stored
DEST_VARIABLE	InOut	VARIANT	D, block interface of an FB	Tag in which the reconverted PLC data type (UDT), STRUCT, ARRAY of <data type> is stored

Parameters	Declaration	Data type	Memory area	Description
POS	InOut	DINT	I, Q, M, D, L	The operand at the POS parameter stores the index of the first byte based on the number of bytes occupied by the converted customer data. The POS parameter is calculated zero-based.
RET_VAL	Output	INT	I, Q, M, D, L	Error information
[1] Possible for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.				

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B0	The memory areas for the SRC_ARRAY and DEST_VARIABLE parameters overlap.
8136	The tag at the SRC_ARRAY parameter is not in a block with standard access.
8150	The VARIANT data type at the SRC_ARRAY parameter contains a ZERO pointer.
8151	No valid reference at the SRC_ARRAY parameter
8153	There is not enough free memory available at the SRC_ARRAY parameter.
8154	Invalid data type at the SRC_ARRAY parameter
8250	The VARIANT data type at the DEST_VARIABLE parameter contains a ZERO pointer.
8251	No valid reference at the DEST_VARIABLE parameter
8382	The value at the POS parameter is outside the limits of the array.
9512	For a CPU of the S7-1200 module and a block with the block property "Optimized block access": The BOOL data type is used at the DEST_VARIABLE parameter. This makes the object smaller than a full byte.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

For a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0, the following error code has a different meaning:

Error code* (W#16#...)	Explanation
8136	Access to the memory area at the SRC_ARRAY parameter is invalid.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following table shows the declaration of the operands:

Operand	Data type	Declaration
DeliverPos	INT	In the "Input" section of the block interface of an FB or FC.
BufferPos	DINT	In the "Temp" section of the block interface of an FB or FC.
Error	INT	
Label	STRING[4]	

The following table shows the declarations of the PLC data types:

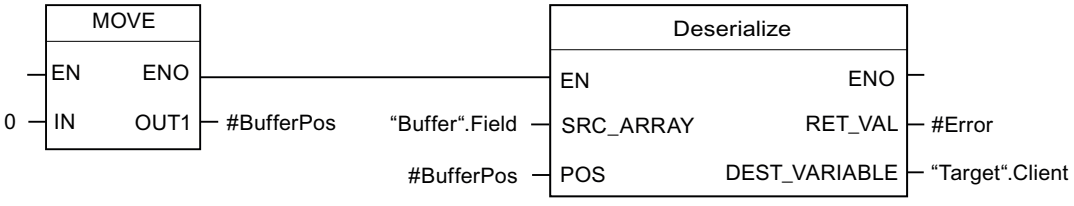
Name of the PLC data types	Name	Data type
Article	Number	DINT
	Declaration	STRING
	Colli	INT
Client	Title	INT
	First name	STRING[10]
	Surname	STRING[10]

The following table shows the declaration of the data blocks:

Name of the data blocks	Name	Data type
Target	Client	"Client" (PLC data type)
	Article	Array[0..10] of "Article" (PLC data type)
	Bill	Array[0..10] of INT
Buffer	Field	Array[0..294] of BYTE

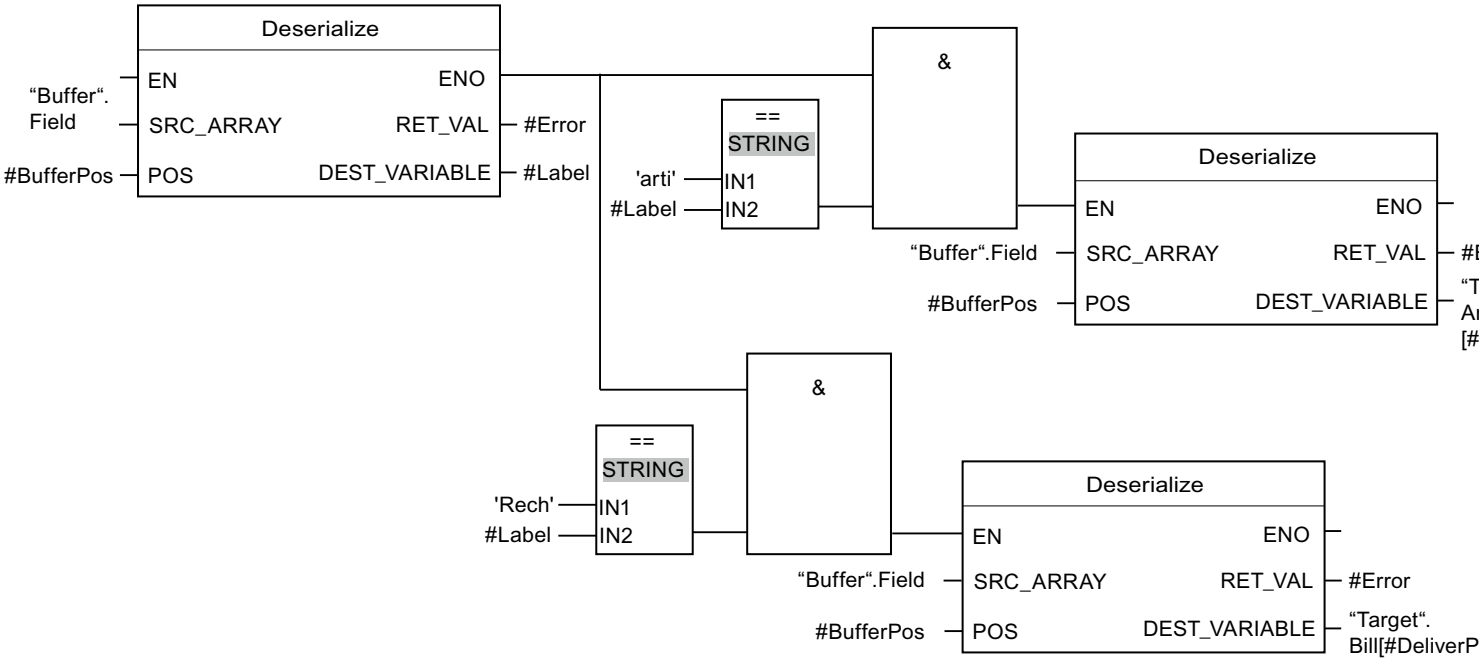
The following example shows how the instruction works:

Network 1:



The "Move value" instruction moves the value "0" to the #BufferPos operand. The "Deserialize" instruction deserializes the sequential representation of the customer data from the "Buffer" tag and writes it to the "Target" tag. The #BufferPos operand stores the index of the first byte based on the number of bytes occupied by the converted customer data.

Network 2:



The "Deserialize" instruction deserializes the sequential representation of the separator sheet, which was stored in the sequential representation after the customer data, from the "Buffer" tag and writes the characters to the #Label operand. The characters are compared using comparison instructions "arti" and "Bill". If the comparison for "arti" = TRUE, these are article data that have been deserialized and written to the "Target" tag. If the comparison for "Bill" = TRUE, these are billing data that have been deserialized and written to the "Target" tag.

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

Basics of PLC data types (UDT) (Page 2277)

Structure of an ARRAY tag (Page 2290)

Structure of a STRUCT tag (Page 2284)

Structure of a STRING tag (Page 2272)

Basic information on FBD (Page 4937)

Serialize: Serialize (S7-1200, S7-1500)

Description

You can use the "Serialize" instruction to convert several PLC data types (UDT), STRUCT or ARRAY of <data type> to a sequential representation without losing parts of their structure.

You use the instruction to temporarily save multiple structured data items from your program in a buffer, which should preferably be in a global data block, and send them to another CPU.

The memory area in which the converted data are stored must have the ARRAY of BYTE data type and be declared with standard access in version 1.0. Optimized data is also permitted as of version 2.0. Fill bytes or fill bits of a data area (e.g. ARRAY, STRUCT or PLC data type (UDT)) as well as the characters of a string currently not in use are undefined in the destination ARRAY.

The capacity of the standard memory area is 64 KB. Structures that are larger than 64 KB in accordance with the rules for standard memory areas cannot be serialized if the operand at the DEST_ARRAY parameter is located in a standard memory area.

It is recommended to define the low limit of the ARRAY with "0", because then the index within ARRAY corresponds to the value of the POS parameter, e.g. ARRAY[0] = POS 0. The description and example below have been formed based on this.

The operand at the POS parameter contains information about the number of bytes used by the converted data.

If you want to send a single PLC data type (UDT), STRUCT or ARRAY of <data type>, you can directly call the instruction "TSEND: Send data via communication connection".

Note

Comparison of structures

To compare structures, they have to be serialized first. Use the "CMP" instruction instead.

You can find additional information in the section "Comparator".

Size of the memory area

The alignment rules mean that simple structures in the optimized memory area do not contain filling bytes. This results in a structure in the optimized memory area being smaller than in the standard memory area. ARRAYs of structures and structures in which other structures are located contain filling bytes. There is no general rule as to the memory area in which a composed structure requires more space.

Applies to CPUs of the S7-1500 series:

For a block with the block property "Optimized block access", the BOOL has a length of 1 byte. A structure that consists largely of the data type BOOL can therefore be larger in the optimized memory area than in the standard memory area. Composed structures with a low proportion of BOOL data types are smaller in the optimized memory area than in the standard memory area.

Optimized memory area

To serialize larger structures, you can declare the memory area with optimized access for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0. The sequential representation remains unchanged, as for a standard memory area.

Parameters

The following table shows the parameters of the instruction "Serialize":

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRC_VARIABLE	Input	VARIANT	D, block interface of an FB	PLC data type (UDT), STRUCT or ARRAY of <data type> that is converted to sequential representation.
DEST_ARRAY	InOut	VARIANT	D, block interface of an FB (the sections Input, Output, Static and Temp are possible.)	Data block in which the generated data stream is stored.
POS	InOut	DINT	I, Q, M, D, L	The operand at the POS parameter stores the index of the first byte based on the total number of bytes that the converted customer data has occupied. The POS parameter is calculated zero-based.
RET_VAL	Output	INT	I, Q, M, D, L	Error information

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B0	The memory areas for the SRC_VARIABLE and DEST_ARRAY parameters overlap.
8150	The VARIANT data type at the SRC_VARIABLE parameter contains a ZERO pointer.
8151	No valid reference at the SRC_VARIABLE parameter
8236	The tag at the SRC_ARRAY parameter is not in a block with standard access.
8250	The VARIANT data type at the DEST_ARRAY parameter contains a ZERO pointer.
8251	No valid reference at the DEST_ARRAY parameter
8253	There is not enough free memory available at the DEST_ARRAY parameter.
8254	Invalid data type at the DEST_ARRAY parameter
8382	The value at the POS parameter is outside the limits of the array.
9512	For a CPU of the S7-1200 module and a block with the block property "Optimized block access": The BOOL data type is used at the DEST_VARIABLE parameter. This makes the object smaller than a full byte.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

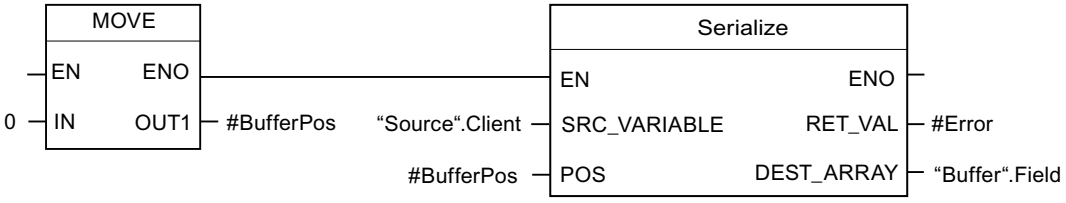
For a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0, the following error code has a different meaning:

Error code* (W#16#...)	Explanation
8236	Access to the memory area at the DEST_ARRAY parameter is invalid.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

Example

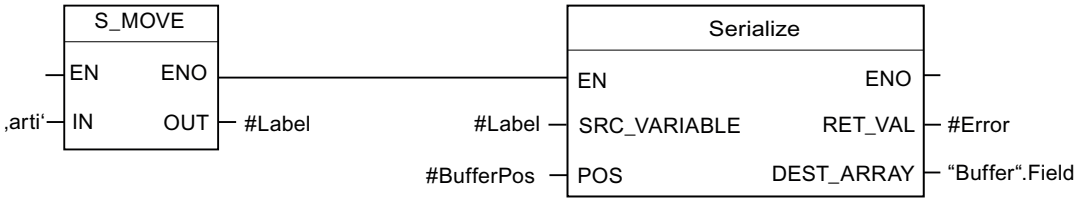
The following example shows how the instruction works:

Network 1:



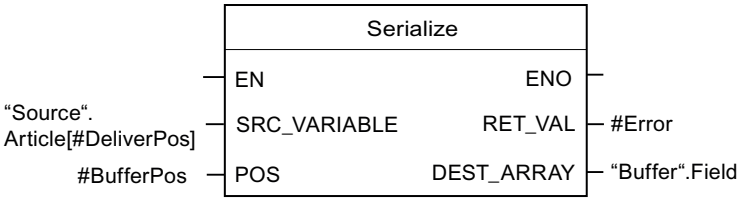
The "Move value" instruction moves the value "0" to the #BufferPos operand. The "Deserialize" instruction deserializes the customer data from the "Source" tag and writes the sequential representation to the "Buffer" tag. The index of the next unwritten byte of the "Buffer".Field operand is saved in the #BufferPos operand.

Network 2:



A kind of separator sheet is inserted now to make it easier to deserialize the sequential representation later. The "Move character string" instruction moves the "arti" characters to the #Label operand. The "Serialize" instruction writes these characters after the customer data to the "Buffer" tag. The value of the #BufferPos operand is incremented accordingly.

Network 3:



The "Serialize" instruction serializes the data of a specific article, which is calculated in runtime, from the "Source" tag and writes it in sequential representation to the "Buffer" tag after the "arti" characters.

The following table shows the declaration of the operands:

Operand	Data type	Declaration
DeliverPos	INT	In the "Input" section of the block interface
BufferPos	DINT	In the "Temp" section of the block interface
Error	INT	In the "Temp" section of the block interface
Label	STRING[4]	In the "Temp" section of the block interface

The following table shows the declarations of the PLC data types:

Name of the PLC data types	Name	Data type
Article	Number	DINT
	Declaration	STRING
	Colli	INT
Client	Title	INT
	First name	STRING[10]
	Surname	STRING[10]

The following table shows the declaration of the data blocks:

Name of the data blocks	Name	Data type
Source	Client	"Client" (PLC data type)
	Article	Array[0..10] of "Article" (PLC data type)
Buffer	Field	Array[0..294] of BYTE

See also

Overview of the valid data types (Page 2239)
Basic information on VARIANT (Page 2308)
Basics of PLC data types (UDT) (Page 2277)
Structure of an ARRAY tag (Page 2290)
Structure of a STRUCT tag (Page 2284)
Structure of a STRING tag (Page 2272)
Basic information on FBD (Page 4937)

MOVE_BLK: Move block (S7-1200, S7-1500)

Description

You use the "Move block" instruction to move the content of a memory area (source range) to another memory area (target range). The number of elements to be moved to the target range is specified with the COUNT parameter. The width of the elements to be moved is defined by the width of the element at input IN.

The instruction can only be executed if the source range and the target range have the same data type.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- More data is moved than is made available at input IN or output OUT.

When a ARRAY of BOOL is copied, the enable output ENO for an overflow is set to "1" until the byte limit of the ARRAY structure is exceeded. If the byte limit of the ARRAY structure is exceeded by the value at the COUNT input, the ENO enable output is reset to "0".

Note

Use of ARRAYs

The instruction copies the contents from the defined element n elements (n = depending on the value at the parameter COUNT) from the source range to the target range, starting at the specified index.

Parameters

The following table shows the parameters of the "Move block" instruction:

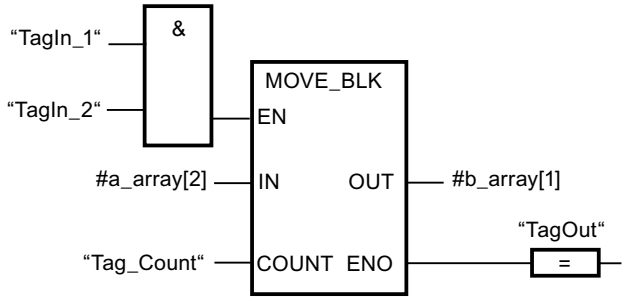
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN [1)]	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	D, L	The first element of the source area that is being copied

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Number of elements to be copied from the source range to the target range
OUT [1]	Output	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	D, L	The first element of the target range to which the contents of the source range are being copied
[1] The specified data types can only be used as elements of an ARRAY structure.						

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	a_array[2]	The data type of the "a_array" operand is ARRAY [0..5] of INT. It consists of six elements of data type INT.
COUNT	Tag_Count	3
OUT	b_array[1]	The data type of the "b_array" operand is ARRAY [0..6] of INT. It consists of seven elements of data type INT.

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. Starting from the third element, the instruction selects three INT elements from the #a_array tag and copies their contents to the #b_array output tag, beginning with the second element.

If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

Note

You can find more information on the MOVE_BLK instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

SCATTER: Parse the bit sequence into individual bits (S7-1200, S7-1500)

Description

The instruction "Parse the bit sequence into individual bits" disassembles a tag of the BYTE, WORD, DWORD or LWORD data type into individual bits and saves them in an ARRAY of BOOL.

Note

Length of the ARRAY

The ARRAY of BOOL must have exactly the number of elements that is specified by the bit sequence.

This means for the data type BYTE, for example, the ARRAY must have exactly 8 elements (WORD = 16, DWORD = 32 and LWORD = 64).

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

This way you can, for example, parse a status word, and read and change the status of the individual bits using the index. Using GATHER, you can merge the bits once again into a bit sequence.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- If the ARRAY does not provide enough BOOL elements.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	BYTE, WORD, DWORD, LWORD	I, Q, M, D, L	Bit sequence that is parsed
OUT	Output	ARRAY[*] of BOOL *: 8, 16, 32 or 64 elements	I, Q, M, D, L	ARRAY in which the individual bits are saved

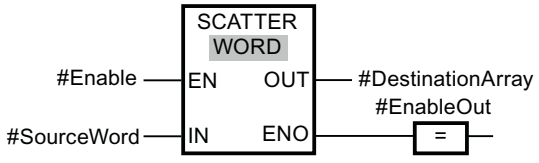
You can find additional information on valid data types under "See also".

Example

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceWord		WORD
EnableOut	Output	BOOL
DestinationArray		ARRAY[0..15] of BOOL

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceWORD	WORD (16 bits)
OUT	DestinationArray	The operand "DestinationArray" is of the data type ARRAY[0..15] of BOOL. It consists of 16 elements and is therefore just as large as the WORD that is to be parsed.

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. The operand #SourceWord of the data type WORD is parsed into its individual bits (16) and assigned to the individual elements of the #DestinationArray operand. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SCATTER_BLK: Parse elements of an ARRAY of bit sequence into individual bits (S7-1200, S7-1500)

Description

The instruction "Parse elements of an ARRAY of bit sequence into individual bits" parses one or more elements of an ARRAY of BYTE, WORD, DWORD or LWORD into individual bits and saves them in an ARRAY of BOOL. At the COUNT_IN parameter you specify how many elements of the source ARRAY are going to be disassembled. The source ARRAY at the IN parameter may have more elements than specified at the COUNT_IN parameter. The ARRAY of BOOL must have sufficient elements to save the bits of the disassembled bit strings. However, the destination ARRAY may also be larger.

Note

If the low ARRAY limit of the target ARRAY is not "0", note the following:

For performance reasons the index must always start at a BYTE, WORD, DWORD or LWORD limit. This means the index must be calculated starting at the low limit of the ARRAY. The following formula is used as basis for this calculation:

Valid indices = Low ARRAY limit + n (number of bit strings) × number of bits of the desired bit string

For an ARRAY[-2..45] of BOOL and the bit string WORD the calculation looks as follows:

- Valid index (-2) = -2 + 0 × 16
- Valid index (14) = -2 + 1 × 16
- Valid index (30) = -2 + 2 × 16

You can find an example described below.

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

This way you can, for example, parse status words, and read and change the status of the individual bits using the index. Using GATHER, you can combine the bits once again into a bit string.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- The source ARRAY has fewer elements than specified at the COUNT_IN parameter.
- The index of the destination ARRAY does not start at a BYTE, WORD, DWORD or LWORD limit. In this case, no result is written to the ARRAY of BOOL.
- The ARRAY[*] of BOOL does not provide the required number of elements. In this case as many bit strings as possible are parsed and written to the ARRAY of BOOL. The remaining bit strings are no longer taken into account.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Element of an ARRAY[*] of <bit string>	I, Q, M, D, L	ARRAY of <bit string> that is disassembled
COUNT_IN	Input	UDINT	I, Q, M, D, L	Counter for the number of elements of the source ARRAY that are going to be disassembled
OUT	Output	Element of an ARRAY[*] of BOOL	I, Q, M, D, L	ARRAY in which the individual bits are saved

You can select the required bit string from the "???" drop-down list of the instruction box.

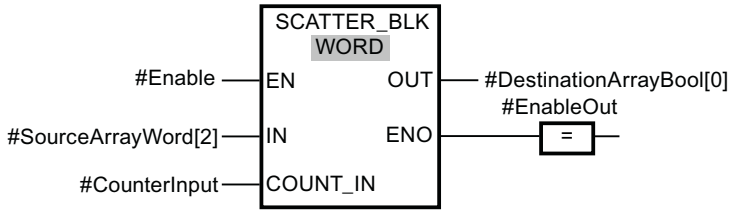
You can find additional information on valid data types under "See also".

Example of a destination ARRAY wit the low limit "0"

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArrayWord		ARRAY[0..5] of WORD
CounterInput		UDINT
EnableOut	Output	BOOL
DestinationArrayBool		ARRAY[0..95] of BOOL

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayWord[2]	ARRAY[0..5] of WORD (96 bits can be disassembled.)
COUNT_IN	CounterInput = 3	UDINT3 (3 WORDs or 48 bits are to be parsed. This means at least 48 bits must be available in the destination ARRAY.)
OUT	DestinationArrayBool[0]	The operand "DestinationArray-Bool" is of the data type AR-RAY[0..95] of BOOL. This means it provides 96 BOOL ele-ments.

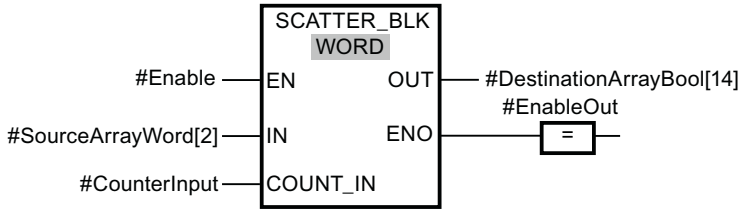
If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. The 3rd, 4th and 5th WORD of the operand #SourceArrayWord is parsed into its individual bits (48) and as of the first element assigned to the individual elements of the #DestinationArrayBool operand. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

Example of a destination ARRAY wit the low limit "-2"

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArrayWord		ARRAY[0..5] of WORD
CounterInput		UDINT
EnableOut	Output	BOOL
DestinationArrayBool		ARRAY[-2..93] of BOOL

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayWord[2]	ARRAY[0..5] of WORD (96 bits can be disassembled.)
COUNT_IN	CounterInput = 3	UDINT3 (3 WORDs or 48 bits are to be parsed. This means at least 48 bits must be available in the destination ARRAY.)
OUT	DestinationArrayBool[14]	The operand "DestinationArray-Bool" is of the data type AR-RAY[-2..93] of BOOL. This means it provides 96 BOOL ele-ments.

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. The 3rd, 4th and 5th WORD of the operand #SourceArrayWord is parsed into its individual bits (48) and as of the 16th element assigned to the individual elements of the #DestinationArrayBool operand. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO. The remaining 32 bits are not written.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

GATHER: Merge individual bits into a bit sequence (S7-1200, S7-1500)

Description

The instruction "Merge individual bits into a bit sequence" combines the bits from an ARRAY of BOOL into a bit sequence. The bit sequence is saved in a tag of the data type BYTE, WORD, DWORD or LWORD.

Note

Length of the ARRAY

The ARRAY of BOOL must have exactly the number of elements that is specified by the bit sequence.

This means for the data type BYTE, for example, the ARRAY must have exactly 8 elements (WORD = 16, DWORD = 32 and LWORD = 64).

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- If the ARRAY has fewer or more BOOL elements than specified by the bit sequence, they are not transferred and the ENO returns signal state "0". The ENO also returns "0" if fewer than the necessary number of bits is available.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	ARRAY[*] of BOOL *: 8, 16, 32 or 64 elements	I, Q, M, D, L	ARRAY whose bits are combined into a bit sequence
OUT	Output	BYTE, WORD, DWORD, LWORD	I, Q, M, D, L	Combined bit sequence, saved in a tag

You can select the required bit sequence from the "???" drop-down list of the instruction box.

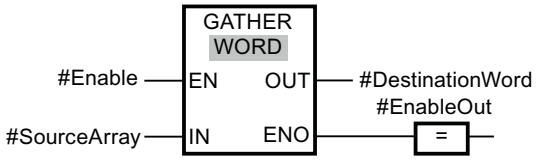
You can find additional information on valid data types under "See also".

Example

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArray		WORD
EnableOut	Output	BOOL
DestinationWord		ARRAY[0..15] of BOOL

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArray	The operand "SourceArray" is of the data type ARRAY[0..15] of BOOL. It consists of 16 elements and is therefore just as large as the WORD in which the bits are to be combined.
OUT	DestinationWord	WORD (16 bits)

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. The bits of the operand #SourceArray are combined into a WORD. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

GATHER_BLK: Merge individual bits into multiple elements of an ARRAY of bit sequence (S7-1200, S7-1500)

Description

The instruction "Merge individual bits into multiple elements of an ARRAY of bit sequence" combines the bits from an ARRAY of BOOL into one or multiple elements of an ARRAY of <bit string>. At the COUNT_OUT parameter you specify how many elements of the destination ARRAY are going to be written. With this step you also implicitly specify how many elements of the ARRAY of BOOL are required. The destination ARRAY at the OUT parameter may have more elements than specified at the COUNT_OUT parameter. The ARRAY of <bit string> must have sufficient elements to save the bits that are going to be combined. However, the destination ARRAY may also be larger.

Note

If the low ARRAY limit of the target ARRAY is not "0", note the following:

For performance reasons the index must always start at a BYTE, WORD, DWORD or LWORD limit. This means the index must be calculated starting at the low limit of the ARRAY. The following formula is used as basis for this calculation:

Valid indices = Low ARRAY limit + n(number of bit strings) × number of bits of the desired bit string

For an ARRAY[-2..45] of BOOL and the bit string WORD the calculation looks as follows:

- Valid index (-2) = -2 + 0 × 16
- Valid index (14) = -2 + 1 × 16
- Valid index (30) = -2 + 2 × 16

You can find an example described below.

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- The index of the source ARRAY does not start at a BYTE, WORD, DWORD or LWORD limit. In this case, no result is written to the ARRAY of <bit string>.
- The ARRAY[*] of <bit string> does not provide the required number of elements. In this case as many bit strings as possible are combined and written to the ARRAY of <bit string>. The remaining bits are no longer taken into account.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Element of an ARRAY[*] of BOOL	I, Q, M, D, L	ARRAY of BOOL whose bits are combined (source ARRAY)
COUNT_OUT	Input	UDINT	I, Q, M, D, L	Counter of the number of elements of the destination ARRAY that are going to be written
OUT	Output	Element of an ARRAY[*] of <bit string>	I, Q, M, D, L	ARRAY of <bit string> to which the bits are saved (destination ARRAY)

You can select the required bit string from the "???" drop-down list of the instruction box.

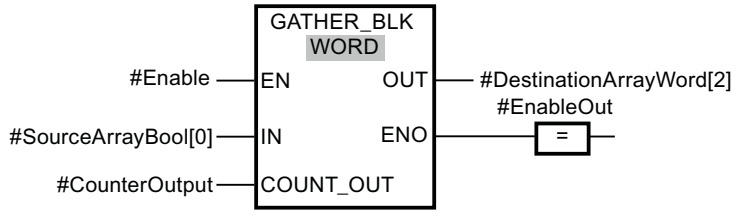
You can find additional information on valid data types under "See also".

Example of a source ARRAY wit the low limit "0"

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArrayBool		ARRAY[0..95] of BOOL
CounterOutput		UDINT
EnableOut	Output	BOOL
DestinationArrayWord		ARRAY[0..5] of WORD

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayBool[0]	The operand "SourceArrayBool" is of the data type ARRAY[0..95] of BOOL. This means it provides 96 BOOL elements that can combined into words once again.
COUNT_OUT	CounterOutput = 3	UDINT3 (3 words are to be written. This means 48 bits must be available in the source ARRAY.)
OUT	DestinationArrayWord[2]	The operand "DestinationArrayWord" is of the data type ARRAY[0..5] of WORD. This means 6 WORD elements are available.

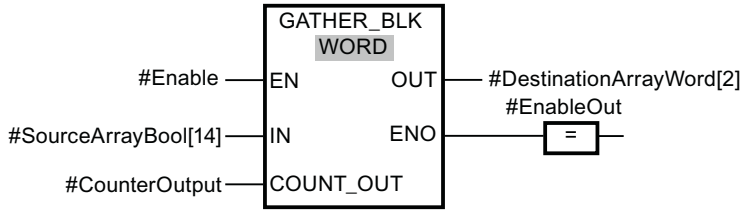
If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. Starting with the 1st element of the operand #SourceArrayBool, 48 bits are combined in the operand #DestinationArrayWord. The starting point in the destination ARRAY is the 3rd element. This means the first 16 bits are written to the 3rd word, the second 16 bits to the 4th word, and the third 16 bits to the 5th word of the destination ARRAY. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

Example of a source ARRAY wit the low limit "-2"

Create the following tags in the block interface:

Tag	Section	Data type
Enable	Input	BOOL
SourceArrayBool		ARRAY[-2..93] of BOOL
CounterOutput		UDINT
EnableOut	Output	BOOL
DestinationArrayWord		ARRAY[0..5] of WORD

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayBool[14]	The operand "SourceArrayBool" is of the data type ARRAY[-2..93] of BOOL. Because the starting point is the 14th element, only 80 BOOL elements that can be combined into words again are available.
COUNT_OUT	CounterOutput = 3	UDINT3 (3 words are to be written. This means 48 bits must be available in the source ARRAY.)
OUT	DestinationArrayWord[2]	The operand "DestinationArrayWord" is of the data type ARRAY[0..5] of WORD. This means 6 WORD elements are available.

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. Starting with the 14th element of the operand #SourceArrayBool, 48 bits are combined in the operand #DestinationArrayWord. The starting point in the destination ARRAY is the 3rd element. This means the first 16 bits of the source ARRAY are ignored. The second 16 bits are written to the 3rd word, the third 16 bits to the 4th word, and the fourth 16 bits to the 5th word of the destination ARRAY. The remaining 64 bits of the source ARRAY are not taken into account either. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

MOVE_BLK_VARIANT: Move block (S7-1200, S7-1500)

Description

You use the "Move block" instruction to move the content of a memory area (source range) to another memory area (target range). You can copy a complete ARRAY or elements of an ARRAY to another ARRAY of the same data type. The size (number of elements) of source and target ARRAY can be different. You can copy multiple or single elements within an ARRAY.

The number of elements that are to be copied must not exceed the selected source range or target range.

If you use the instruction at the time the block is being created, the ARRAY does not yet have to be known, as the source and the target are transferred per VARIANT.

Counting at the parameters SRC_INDEX and DEST_INDEX always begins with the low limit "0", regardless of the later declaration of the ARRAY.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- More data is copied than is made available.

Parameters

The following table shows the parameters of the "Move block" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRC	Input [2)]	VARIANT (which points to an ARRAY or an individual ARRAY element), ARRAY of <Data_type>	L (A declaration in the "Temp" section of the block interface is not possible.)	Source block from which to copy
COUNT	Input	UDINT	I, Q, M, D, L or constant	Number of elements which are copied Assign the value "1" to the parameter COUNT, if no ARRAY is specified at parameter SRC or at parameter DEST.
SRC_INDEX	Input	DINT	I, Q, M, D, L or constant	Defines the first element to be copied: • The SRC_INDEX parameter is calculated zero-based. If an ARRAY is specified at parameter SRC, the integer at parameter SRC_INDEX specifies the first element within the source area from which it is to be copied. Independent of the declared ARRAY limits. • If no ARRAY is specified at parameter SRC or only one single element of an ARRAY is specified, then assign the value "0" at parameter SRC_INDEX.

Parameters	Declaration	Data type	Memory area	Description
DEST_INDEX	Input	DINT	I, Q, M, D, L or constant	Defines the start of the destination memory area: • The DEST_INDEX parameter is calculated zero-based. If an ARRAY is specified at parameter DEST, the integer at parameter DEST_INDEX specifies the first element within the target range that is to be copied into. Independent of the declared ARRAY limits. • If no ARRAY is specified at parameter DEST, then assign the value "0" at parameter DEST_INDEX.
DEST	Output [1]	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Destination area into which the contents of the source block are copied.
RET_VAL	Output	INT	I, Q, M, D, L	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.
1) The DEST parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface. 2) With the SRC parameter, the data types BOOL and ARRAY of BOOL are not permitted.				

You can find additional information on valid data types under "See also".

Parameter RET_VAL

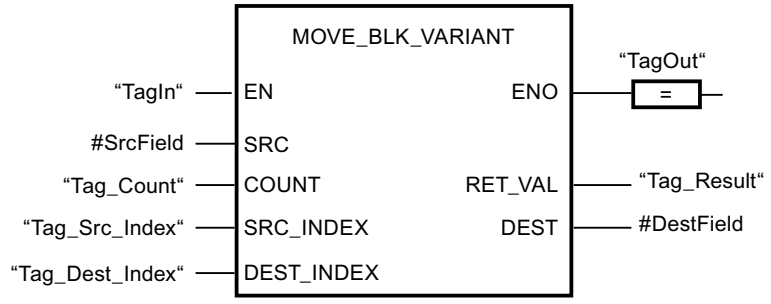
The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The data types do not correspond. Instead of an ARRAY of <Struct>, use an ARRAY of <PLC data type (UDT)>.
8151	Access to the SRC parameter is not possible.
8152	The operand at the SRC parameter is not typed.
8153	Code generation error at the SRC parameter
8154	The operand at the SRC parameter has the data type BOOL.
8281	The COUNT parameter has an invalid value
8382	The value at the SRC_INDEX parameter is outside the limits of the VARIANT.

Error code* (W#16#...)	Explanation
8383	The value at the SRC_INDEX is outside the high limit of the ARRAY.
8482	The value at the DEST_INDEX parameter is outside the limits of the VARIANT.
8483	The value at the DEST_INDEX is outside the high limit of the ARRAY.
8534	The DEST parameter is write protected
8551	Access to the DEST parameter is not possible.
8552	The operand at the DEST parameter is not typed.
8553	Code generation error at the DEST parameter
8554	The operand at the DEST parameter has the data type BOOL.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Declaration in the block interface	Operand	Value
SRC	Input	#SrcField	The local operand #SrcField uses a PLC data type that was still unknown at the time when the block was programmed. (ARRAY[0..10] of "MOVE_UDT")
COUNT	Input	Tag_Count	2
SRC_INDEX	Input	Tag_Src_Index	3
DEST_INDEX	Input	Tag_Dest_Index	3
DEST	InOut	#DestField	The local operand #DestField uses a PLC data type that was still unknown at the time when the block was programmed. (ARRAY[10..20] of "MOVE_UDT")

If the "TagIn" operand returns signal state "1", the instruction is executed. Two elements are moved from the source range, starting from the fourth element of the ARRAY [0..10] of MOVE_UDT, to the target range. The copies are pasted in the ARRAY [10..20] of MOVE_UDT starting from the fourth element. If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set.

Note

You can find more information on the MOVE_BLK_VARIANT instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

VariantGet: Read out VARIANT tag value (Page 3108)

Example of moving data (Page 2316)

Basic information on VARIANT (Page 2308)

Basic information on FBD (Page 4937)

UMOVE_BLK: Move block uninterruptible (S7-1200, S7-1500)

Description

You use the "Move block uninterruptible" instruction to move the content of a memory area (source range) to another memory area (target range). The instruction cannot be interrupted. The number of elements to be moved to the target range is specified with the COUNT parameter. The width of the elements to be moved is defined by the width of the element at input IN.

The instruction can only be executed if the source range and the target range have the same data type.

Note

The move operation cannot be interrupted by other operating system activities. This is why the interrupt reaction times of the CPU increase during the execution of the "Move block uninterruptible" instruction.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- More data is moved than is made available at input IN or output OUT.

When a ARRAY of BOOL is copied, the enable output ENO for an overflow is set to "1" until the byte limit of the ARRAY structure is exceeded. If the byte limit of the ARRAY structure is exceeded by the value at the COUNT input, the ENO enable output is reset to "0".

Note

Use of ARRAYs

The instruction copies the contents from the defined element n elements (n = depending on the value at the parameter COUNT) from the source range to the target range, starting at the specified index.

You use the "Move block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Parameters

The following table shows the parameters of the "Move block uninterruptible" instruction:

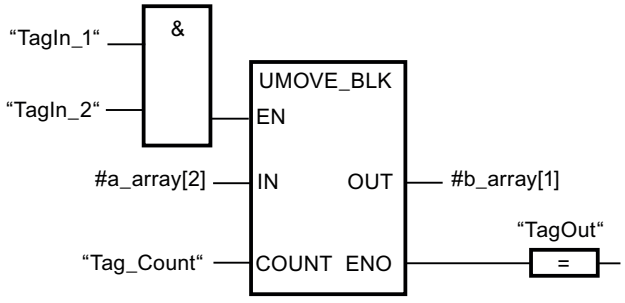
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN [1]	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	D, L	The first element of the source area that is being copied
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Number of elements to be copied from the source range to the target range
OUT [1]	Output	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	D, L	The first element of the target range to which the contents of the source range are being copied

[1] The specified data types can only be used as elements of an array structure.

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	a_array[2]	The data type of the "a_array" operand is ARRAY[0..5] of INT. It consists of six elements of data type INT.
COUNT	Tag_Count	3
OUT	b_array[1]	The data type of the "b_array" operand is ARRAY[0..6] of INT. It consists of seven elements of data type INT.

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. Starting from the third element, the instruction selects three INT elements from the #a_array tag and copies their contents to the #b_array output tag, beginning with the second element. The move operation cannot be interrupted by other operating system activities. If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

Note

You can find more information on the UMOVE_BLK instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

FILL_BLK: Fill block (S7-1200, S7-1500)

Description

You can use the "Fill block" instruction to fill a memory area (target range) with the value of the IN input. The target range is filled beginning with the address specified at the OUT output. The number of repeated copy operations is specified with the COUNT parameter. When the instruction is executed, the value at the input IN is moved to the target range as often as specified by the value of the COUNT parameter.

The instruction can only be executed if the source range and the target range have the same data type.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The maximum number of elements changed is the number of elements in the ARRAY or structure. If you copy more data than there are elements at the OUT output, you will get an unintended result.

When a ARRAY of BOOL is copied, the enable output ENO for an overflow is set to "1" until the byte limit of the ARRAY structure is exceeded. If the byte limit of the ARRAY structure is exceeded by the value at the COUNT input, the ENO enable output is reset to "0".

Note

Use of ARRAYs

The instruction reads the content from the selected element from the source range and copies this content n-times (n = depending on the value at the parameter COUNT) to the destination, starting at the specified index.

Filling structures

As well as the elements of an ARRAY, you can also fill multiple elements of a structure (STRUCT, PLC data type) with the same value. The structure whose elements you want to fill must only contain individual elements of the same elementary data type. The structure itself can, however, be embedded in another structure.

Parameters

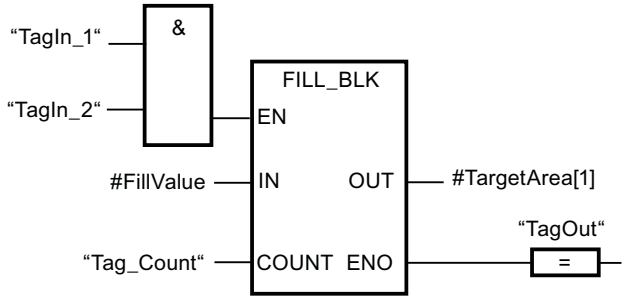
The following table shows the parameters of the "Fill block" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	Element used to fill the target range
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	Number of repeated move operations
OUT	Output	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	D, L	Address in target range from which filling starts

For additional information on valid data types, refer to "See also".

Example with an ARRAY

The following example shows how the instruction works when you want to fill an ARRAY:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	FillValue	The data type of the operand is INT.
COUNT	Tag_Count	3
OUT	TargetArea	The data type of the operand TargetArea is ARRAY[1..5] of INT. It consists of five elements of data type INT.

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The instruction copies the value of the operand #FillValue to the #TargetArea output tag three times, starting with the first element. If the instruction is executed without errors, the TagOut operand at the "ENO" enable output is set to signal state "1".

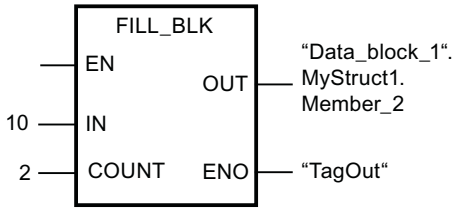
Examples with a structure

The following examples show how the instruction works when you want to fill a structure:

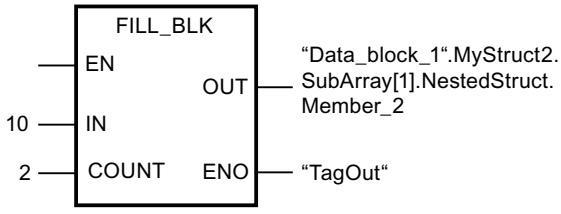
Create a global data block with the following elements:

Data_block_1					Data type
MyStruct1					STRUCT
	Member_1				INT
	Member_2				INT
	Member_3				INT
	Member_4				INT
MyStruct2					STRUCT
	SubArray				ARRAY[1..2] of STRUCT
		SubArray[1]			STRUCT
			NestedStruct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT
		SubArray[2]			STRUCT
			Nested-Struct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT

Generate the following program code to address the MyStruct1 tag:



Generate the following program code to address the MyStruct2 tag:



In each of the two examples, the value 10 of the IN parameter is copied twice to the operand at the OUT parameter, starting with the element Member_2. This means that the value 10 is copied to the elements Member_2 and Member_3. The other two elements, Member_1 and Member_4, are not changed. If the instruction is executed without errors, the TagOut operand is set to signal state "1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

UFILL_BLK: Fill block uninterruptible (S7-1200, S7-1500)

Description

You can use the "Fill block uninterruptible" instruction to fill a memory area (target range) with the value of the IN input. The instruction cannot be interrupted. The target range is filled beginning with the address specified at the OUT output. The number of repeated copy operations is specified with the COUNT parameter. When the instruction is executed, the value at the input IN is moved to the target range as often as specified by the value of the COUNT parameter.

The instruction can only be executed if the source range and the target range have the same data type.

Note

The move operation cannot be interrupted by other operating system activities. This is why the alarm reaction times of the CPU increase during the execution of the "Fill block uninterruptible" instruction.

Enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- The maximum number of elements changed is the number of elements in the ARRAY or structure. If you copy more data than there are elements at the OUT output, you will get an unintended result.

When a ARRAY of BOOL is copied, the enable output ENO for an overflow is set to "1" until the byte limit of the ARRAY structure is exceeded. If the byte limit of the ARRAY structure is exceeded by the value at the COUNT input, the ENO enable output is reset to "0".

Note

Use of ARRAYs

The instruction reads the content from the selected element from the source range and copies this content n-times (n = depending on the value at the parameter COUNT) to the destination, starting at the specified index.

You use the "Fill block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Filling structures

As well as the elements of an ARRAY, you can also fill multiple elements of a structure (STRUCT, PLC data type) with the same value. The structure whose elements you want to fill must only contain individual elements of the same elementary data type. The structure itself can, however, be embedded in another structure.

Parameters

The following table shows the parameters of the "Fill block uninterruptible" instruction:

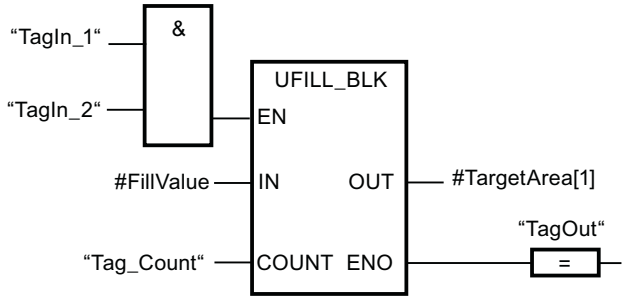
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	Element used to fill the target range

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	Number of repeated move operations
OUT	Output	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	D, L	Address in target range from which filling starts

For additional information on valid data types, refer to "See also".

Example with an ARRAY

The following example shows how the instruction works when you want to fill an ARRAY:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	FillValue	The data type of the operand is INT.
COUNT	Tag_Count	3
OUT	TargetArea	The data type of the operand TargetArea is ARRAY[1..5] of INT. It consists of five elements of data type INT.

If the operands "TagIn_1" and "TagIn_2" have signal state "1", the instruction is executed. The instruction copies the value of the operand #FillValue to the #TargetArea output tag three times, starting with the first element. The move operation cannot be interrupted by other operating system activities. If the instruction is executed without errors, the TagOut operand at the "ENO" enable output is set to signal state "1".

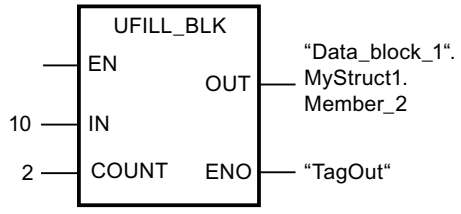
Examples with a structure

The following examples show how the instruction works when you want to fill a structure:

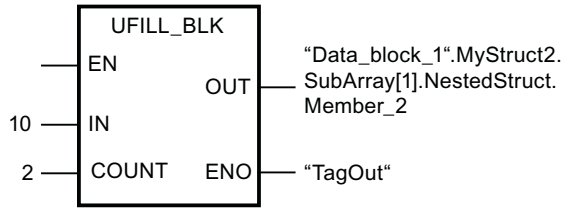
Create a global data block with the following elements:

Data_block_1					Data type
MyStruct1					STRUCT
	Member_1				INT
	Member_2				INT
	Member_3				INT
	Member_4				INT
MyStruct2					STRUCT
	SubArray				ARRAY[1..2] of STRUCT
		SubArray[1]			STRUCT
			NestedStruct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT
		SubArray[2]			STRUCT
			Nested-Struct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT

Generate the following program code to address the MyStruct1 tag:



Generate the following program code to address the MyStruct2 tag:



In each of the two examples, the value 10 of the IN parameter is copied twice to the operand at the OUT parameter, starting with the element Member_2. This means that the value 10 is copied to the elements Member_2 and Member_3. The other two elements, Member_1 and

Member_4, are not changed. If the instruction is executed without errors, the TagOut operand is set to signal state "1".

See also

Overview of the valid data types (Page 2239)

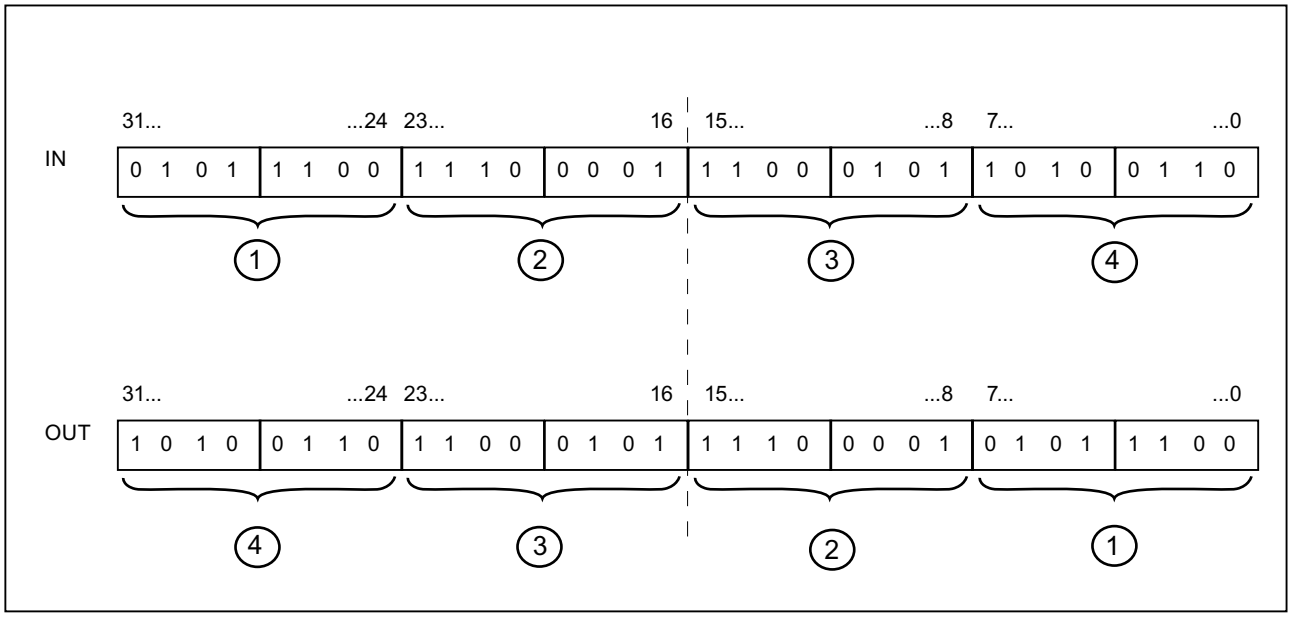
Basic information on FBD (Page 4937)

SWAP: Swap (S7-1200, S7-1500)

Description

You can use the instruction "Swap" to change the order of the bytes within the tag at input IN and query the result at output OUT.

The following figure shows how the bytes of a DWORD data type operand are swapped using the "Swap" instruction:



Parameters

The following table shows the parameters of the "Swap" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output

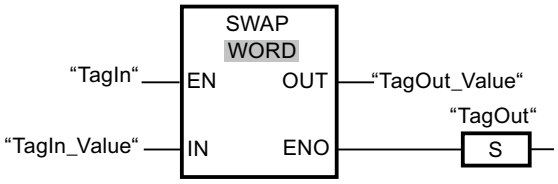
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN	Input	WORD, DWORD	WORD, DWORD, LWORD	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Operand whose bytes are swapped.
OUT	Output	WORD, DWORD	WORD, DWORD, LWORD	I, Q, M, D, L, P	I, Q, M, D, L, P	Result

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	TagIn_Value	0000 1111 0101 0101
OUT	TagOut_Value	0101 0101 0000 1111

If the "TagIn" operand has signal state "1", the instruction is executed. The arrangement of the bytes is changed and stored in the operand "TagOut_Value".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

ARRAY DB (S7-1500)

ReadFromArrayDB: Read from ARRAY data block (S7-1500)

Description

The instruction "Read from ARRAY data block" is used to read the element from a data block of the ARRAY DB block type to which the index references, and write the value of the element to the target range.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

Enable output ENO returns the signal state FALSE if one of the following conditions applies:

- Enable input EN returns the signal state FALSE.
- If an error occurs during the execution of the instruction.

Parameters

The following table shows the parameters of the "Read from ARRAY data block" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
DB	Input	DB_ANY	I, Q, M, D, L	Data block that is read
INDEX	Input	DINT	I, Q, M, D, L, P or constant	Element in the DB that is read. The specification can be a constant, a global tag or an indexed value.
VALUE	Output [1)]	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Value that is read and output
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.
1) The VALUE parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

You can find additional information on valid data types under "See also".

Parameter RET_VAL

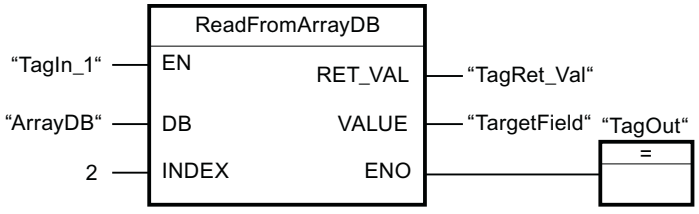
The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
80B5	The copying was interrupted.
8132	The data block does not exist, is too short, write protected, or is located in load memory.
8135	The ARRAY data block contains invalid values.
8154	The data block has the incorrect data type.
8282	The value at the INDEX parameter is outside the limits of the ARRAY.

Error code* (W#16#...)	Explanation
8450	The data type VARIANT at parameter VALUE provides the value "0".
8452	Code generation error
8453	There are two possible causes of the error: • The size of the VALUE parameter does not match the element length in the ARRAY data block. • The two tags are not in a memory area with optimized access. For additional information on the memory area access types, refer to: AUTOHOTSPOT
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type Array [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	TargetField	The "TargetField" operand is a global tag of the data type INT.

If the "TagIn1" operand returns signal state "1", the instruction is executed. The 2nd element is read from "ArrayDB" and written to the "TargetField" operand. If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

Basic information on FBD (Page 4937)

WriteToArrayDB: Write to ARRAY data block (S7-1500)

Description

The instruction "Write to ARRAY data block" is used to write the element to which the index references to a data block of the ARRAY DB block type.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

Enable output ENO returns the signal state FALSE if one of the following conditions applies:

- Enable input EN returns the signal state FALSE.
- If an error occurs during the execution of the instruction.

Parameters

The following table shows the parameters of the "Write to ARRAY data block" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
DB	Input	DB_ANY	I, Q, M, D, L	Data block to which data is written
INDEX	Input	DINT	I, Q, M, D, L, P or constant	Element in the DB to which data is written. The specification can be a constant, a global tag or an indexed value.
VALUE	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Value to be written
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the RET_VAL parameter.

You can find additional information on valid data types under "See also".

Parameter RET_VAL

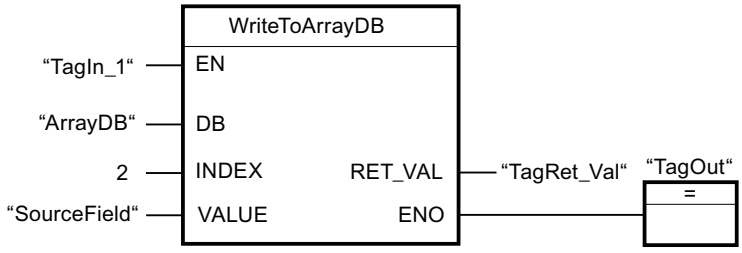
The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.

Error code* (W#16#...)	Explanation
80B5	The copying was interrupted.
8132	The data block does not exist, is too short, or is located in load memory.
8134	The data block is write protected.
8135	The data block is not an ARRAY data block.
8154	The data block has the incorrect data type.
8282	The value at the INDEX parameter is outside the limits of the ARRAY.
8350	The data type VARIANT at parameter VALUE provides the value "0".
8352	Code generation error
8353	There are two possible causes of the error: • The size of the VALUE parameter does not match the element length in the ARRAY data block. • The two tags are not in a memory area with optimized access. For additional information on the memory area access types, refer to: AUTOHOTSPOT
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type Array [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	SourceField	The "SourceField" operand is a global tag of the data type INT.

If the "TagIn1" operand returns signal state "1", the instruction is executed. The value of the "SourceField" operand is written to the 2nd element of the ARRAY DB. If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

Basic information on FBD (Page 4937)

ReadFromArrayDBL: Read from ARRAY data block in load memory (S7-1500)

Description

The instruction "Read from ARRAY data block in load memory" is used to read the element ti which the index references from a data block of the ARRAY DB block type in the load memory and write it to the target range.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

If the ARRAY data block has been designated with the block attribute "Only store in load memory", it will only be stored in the load memory.

The instruction is executed when a positive signal edge is detected at the REQ parameter. The BUSY parameter then has the signal state "1". The instruction is terminated if a negative signal edge is detected at the BUSY parameter. The DONE parameter has the signal state "1" for one program cycle and the read value is output at the VALUE parameter within this cycle. With all other program cycles, the value at the VALUE parameter is not changed.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- If an error occurs during the execution of the instruction.

Parameters

The following table shows the parameters of the "Read from ARRAY data block in load memory" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = "1": Begin with the reading of the ARRAY DB
DB [1)]	Input	DB_ANY	I, Q, M, D, L	ARRAY data block that is read.
INDEX	Input	DINT	I, Q, M, D, L, P or constant	Element in the DB that is read. The specification can be a constant, a global tag or an indexed value.

Parameter	Declaration	Data type	Memory area	Description
VALUE [1]	InOut	VARIANT	D (element of a global data block) L (A declaration in the "Temp" section of the block interface is not possible.)	Pointer to the DB in the work memory that is to be read and the value of which is to be written. No local constants or tags from the TEMP section must be used.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = "1": The array DB is still being read
DONE	Output	BOOL	I, Q, M, D, L	DONE = "1": The instruction was executed successfully
ERROR	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the ERROR parameter.
[1] The data blocks must be created with the "Optimized" block property.				

Parameter ERROR

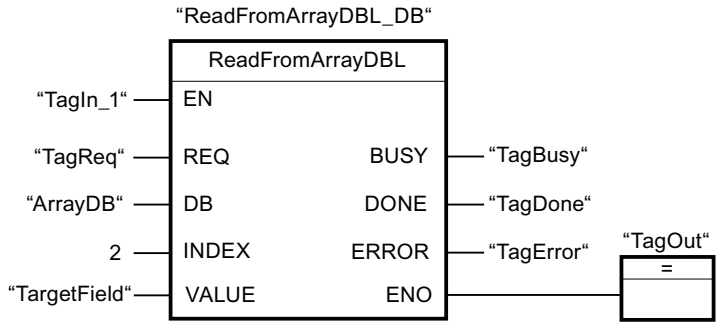
The following table shows the meaning of the values of the ERROR parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
8230	The data block number is incorrect.
8231	The data block does not exist.
8232	The data block is too short, or is not located in load memory.
8235	The data block is not an ARRAY DB.
8254	The data block has the incorrect data type.
8382	The value at the INDEX parameter is outside the limits of the ARRAY.
8750	The data type VARIANT at parameter VALUE provides the value "0".
8751	Code generation error
8752	Code generation error
8753	The size of the parameter does not match the element length in the ARRAY data block.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

You can find descriptions of the error codes triggered by the instructions "READ_DBL" and "WRIT_DBL" in the corresponding descriptions of the instructions.

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
REQ	TagReq	BOOL
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type ARRAY [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	TargetField	The "TargetField" operand is a global tag of the data type INT.
BUSY	TagBusy	BOOL
DONE	TagDone	BOOL

If the "TagIn1" operand has the signal state "1" and a positive edge is detected at the "TagReq" operand, the instruction is executed. The 2nd element is read from "ArrayDB" and output at the VALUE parameter. As soon as a negative signal edge is detected at the "TagBusy" operand, the instruction is terminated and the value at the VALUE parameter is no longer changed. If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set. After the instruction has been processed, the "TagDone" operand has the signal state TRUE.

See also

Overview of the valid data types (Page 2239)

WRIT_DBL: Write to data block in the load memory (Page 4054)

READ_DBL: Read from data block in the load memory (Page 4050)

Basic information on VARIANT (Page 2308)

Difference between synchronous and asynchronous instructions (Page 2571)

Basic information on FBD (Page 4937)

WriteToArrayDBL: Write to ARRAY data block in load memory (S7-1500)

Description

The instruction "Write to ARRAY data block in load memory" is used to write the element to which the index references to a data block of the ARRAY DB block type in load memory.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

If the ARRAY data block has been designated with the block attribute "Only store in load memory", it will only be stored in the load memory.

The instruction is executed when a positive signal edge is detected at the REQ parameter. The BUSY parameter then has the signal state "1". If a negative signal edge is detected at the BUSY parameter, the instruction is terminated and the value at the VALUE parameter is written to the data block. The DONE parameter then has the signal state "1" for one program cycle.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- If an error occurs during the execution of the instruction.

Parameters

The following table shows the parameters of the "Write to ARRAY data block in load memory" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = "1": Start writing to the array DB
DB [1)]	Input	DB_ANY	I, Q, M, D, L	ARRAY data block to which data is written
INDEX	Input	DINT	I, Q, M, D, L, P or constant	Element in the DB to which data is written. The specification can be a constant, a global tag or an indexed value.
VALUE [1)]	Input	VARIANT	D (element of a global data block) L (A declaration in the "Temp" section of the block interface is not possible.)	Pointer to the DB in the work memory that is to be read and the value of which is to be written. No local constants or tags from the TEMP section must be used.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = "1": The array DB is still being written
DONE	Output	BOOL	I, Q, M, D, L	DONE = "1": The instruction was executed successfully

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output at the ERROR parameter.
[1] The data blocks must be created with the "Optimized" block property.				

Parameter ERROR

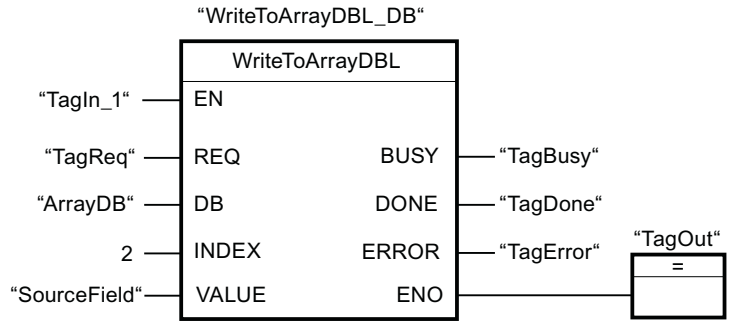
The following table shows the meaning of the values of the ERROR parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
8230	The data block number is incorrect.
8231	The data block does not exist.
8232	The data block is too short, or is not located in load memory.
8234	The data block is write protected.
8235	The data block is not an ARRAY DB.
8254	The data block has the incorrect data type.
8382	The value at the INDEX parameter is outside the limits of the ARRAY.
8450	The data type VARIANT at parameter VALUE provides the value "0".
8751	Code generation error
8752	Code generation error
8753	The size of the VALUE parameter does not match the element length in the ARRAY data block.
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".	

You can find descriptions of the error codes triggered by the instructions "READ_DBL" and "WRIT_DBL" in the corresponding descriptions of the instructions.

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
REQ	TagReq	BOOL
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type ARRAY [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	SourceField	The "SourceField" operand is a global tag of the data type INT.
BUSY	TagBusy	BOOL
DONE	TagDone	BOOL

If the "TagIn1" operand has the signal state "1" and a positive edge is detected at the "TagReq" operand, the instruction is executed. As soon as a negative signal edge is detected at the "TagBusy" operand, the instruction is terminated and the value at the VALUE parameter is written to the 2nd element of the "ArrayDB". If the instruction is executed without errors, the enable output ENO has the signal state TRUE and the "TagOut" output is set. After the instruction has been processed, the "TagDone" operand has the signal state TRUE.

See also

Overview of the valid data types (Page 2239)

WRIT_DBL: Write to data block in the load memory (Page 4054)

READ_DBL: Read from data block in the load memory (Page 4050)

Basic information on VARIANT (Page 2308)

Difference between synchronous and asynchronous instructions (Page 2571)

Basic information on FBD (Page 4937)

VARIANT (S7-1200, S7-1500)

VariantGet: Read out VARIANT tag value (S7-1200, S7-1500)

Description

You can use the "Read out VARIANT tag value" instruction to read the value of the tag to which the VARIANT at the SRC parameter points and write it in the tag at the DST parameter.

The SRC parameter has the VARIANT data type. Any data type except for VARIANT can be specified at the DST parameter.

The data type of the tag specified at the DST parameter must match the data type to which the VARIANT points.

Note

To copy structures and ARRAYs, you can use the "MOVE_BLK_VARIANT instruction: Move block" instruction. For additional information, refer to "See also".

Enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The data types do not match (no value is transferred).

Parameters

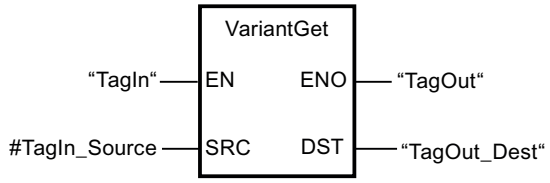
The following table shows the parameters of the "Read out VARIANT tag value" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
SRC	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	Tag to be read
DST	Output	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY elements, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The value of the tag to which the VARIANT at the "#TagIn_Source" operand points, is read and written to the "TagOut_Dest" operand.

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

MOVE_BLK_VARIANT: Move block (Page 3082)

Basic information on FBD (Page 4937)

VariantPut: Write VARIANT tag value (S7-1200, S7-1500)

Description

You can use the "Write VARIANT tag value" instruction to write the value of the tag at the SRC parameter to the memory at the DST parameter to which the VARIANT points.

The DST parameter has the VARIANT data type. Any data type except for VARIANT can be specified at the SRC parameter.

The data type of the tag at the SRC parameter must match the data type to which the VARIANT points.

Note

To copy structures and ARRAYs, you can use the "MOVE_BLK_VARIANT instruction: Move block" instruction. For additional information, refer to "See also".

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The data types do not correspond. (No value is transferred.)

Parameters

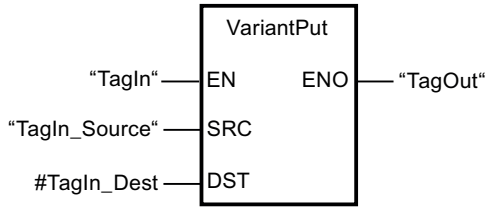
The following table shows the parameters of the instruction "Write VARIANT tag value":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
SRC	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY elements, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Tag to be read
DST	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	Result of the instruction

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand returns signal state "1", the instruction is executed. The value of the "TagIn_Source" operand is written to the tag to which the VARIANT at the #TagIn_Dest operand points.

See also

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

MOVE_BLK_VARIANT: Move block (Page 3082)

Basic information on FBD (Page 4937)

CountOfElements: Get number of ARRAY elements (S7-1200, S7-1500)

Description

You use the "Get number of ARRAY elements" instruction to query how many ARRAY elements a tag to which VARIANT points has.

If it is a multi-dimensional ARRAY, the number of ARRAY elements is output as the result. (The difference between the high and low limit +1). If it is a multi-dimensional ARRAY, the number of all dimensions is output as the result.

Note

Instances

The VARIANT pointer can point to no instance and therefore also to no multi-instance or an ARRAY of multi-instances.

Note

ARRAY within a data block

If you want to query the number of the elements of an ARRAY which is located in a data block, the block attribute "Data block write-protected in the device" must not be activated in this block. Otherwise, the RET_VAL parameter returns the result "0", irrespective of how many elements the ARRAY contains.

The result is also "0" if the VARIANT tag is not an ARRAY.

If the VARIANT points to an ARRAY of BOOL, the fill elements are included in the count. (For example, 8 is returned for an ARRAY[0..1] of BOOL).

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The VARIANT tag is not an ARRAY. (The result is "0".)

Parameters

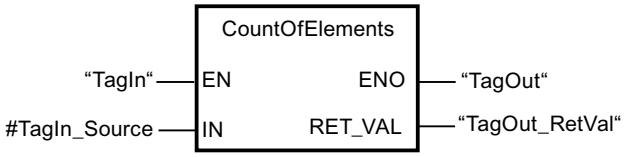
The following table shows the parameters of the "Get number of ARRAY elements" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Tag to be queried
RET_VAL	Output	UDINT	I, Q, M, D, L	Result of the instruction

You can find additional information on valid data types under "See also".

Example

The following example shows how the instruction works:



If the "TagIn" operand returns signal state "1", the instruction is executed. The number of ARRAY elements of the tag to which the VARIANT at the #TagIn_Source operand points is read and output at the "TagOut_RetVal" operand.

See also

Basic information on ARRAY (Page 2286)

Overview of the valid data types (Page 2239)

Basic information on VARIANT (Page 2308)

Basic information on FBD (Page 4937)

ARRAY[*] (S7-1200, S7-1500)

LOWER_BOUND: Read out low ARRAY limit (S7-1200, S7-1500)

Description

You can declare tags of the data type ARRAY[*] in the block interface of a function block or function. For these local tags, you can read out the limits of the ARRAY. You will need to specify the required dimension at the DIM parameter.

The "Read out ARRAY low limit" instruction is used to read out the variable low limit of the ARRAY.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The dimension specified at input DIM does not exist.

Note

Availability of the instruction

The instruction is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

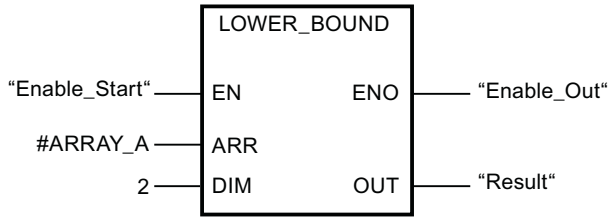
Parameters

The following table shows the parameters of the instruction "Read out ARRAY low limit":

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
ARR	Input	ARRAY[*]	FB => Section In-Out FC => Sections Input and InOut	ARRAY of which the variable low limit is to be read.
DIM	Input	UDINT	I, Q, M, D, L, P or constant	Dimension of the ARRAY of which the variable low limit is to be read.
OUT	Output	DINT	I, Q, M, D, L, P	Result

Example

The following example shows how the instruction works:



If the "Enable_Start" operand returns signal state "1", the instruction is executed. It reads out the variable low limit of the ARRAY #ARRAY_A from the second dimension. If the instruction is executed without errors, the result is written to operand "Enable_Out" and operand "Result" is set.

See also

Overview of the valid data types (Page 2239)

Basic information on ARRAY (Page 2286)

Basic information on FBD (Page 4937)

UPPER_BOUND: Read out high ARRAY limit (S7-1200, S7-1500)

Description

You can declare tags of the data type ARRAY[*] in the block interface of a function block or function. For these local tags, you can read out the limits of the ARRAY. You will need to specify the required dimension at the DIM parameter.

The instruction "Read out ARRAY high limit" is used to read out the variable high limit of the ARRAY.

Enable output ENO returns the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The dimension specified at input DIM does not exist.

Note
Availability of the instruction

The instruction is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

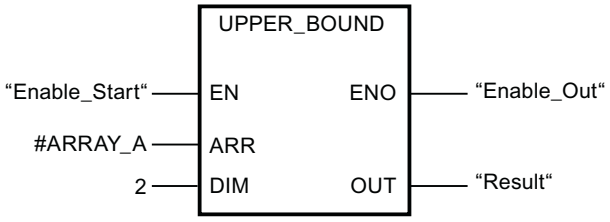
Parameters

The following table shows the parameters of the instruction "Read out ARRAY high limit":

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
ARR	Input	ARRAY[*]	FB => Section In-Out FC => Sections Input and InOut	ARRAY of which the variable high limit is to be read.
DIM	Input	UDINT	I, Q, M, D, L, P or constant	Dimension of the ARRAY of which the variable high limit is to be read.
OUT	Output	DINT	I, Q, M, D, L, P	Result

Example

The following example shows how the instruction works:



If the "Enable_Start" operand returns signal state "1", the instruction is executed. It reads out the variable high limit of the ARRAY #ARRAY_A from the second dimension. If the instruction is executed without errors, the result is written to operand "Enable_Out" and operand "Result" is set.

See also

Overview of the valid data types (Page 2239)

Basic information on ARRAY (Page 2286)

Basic information on FBD (Page 4937)

Legacy (S7-1500)

FieldRead: Read field (S7-1500)

Description

You can use the "Read field" instruction to read out a specific component from the field specified in the MEMBER parameter and transfer its content to the tag in the VALUE parameter. You use the parameter INDEX to define the index of the field components that are to be read. At the parameter MEMBER you specify the first component of the field to be read.

The data types of the field component at parameter MEMBER, the index and the tags at parameter VALUE must correspond to the data type of the instruction "Read field" because implicit conversion is not possible.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The field component specified at the parameter INDEX is not defined in the field specified at the parameter MEMBER.
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the instruction "Read field":

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
INDEX	Input	DINT	DINT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Index of field components whose content is read out

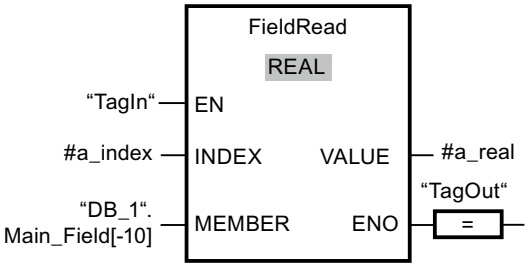
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
MEMBER	Input	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR and WCHAR as components of an ARRAY tag	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD, CHAR and WCHAR as components of an ARRAY tag	D, L	D, L	First component of the field from which will be read
VALUE	Output	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD CHAR, WCHAR	I, Q, M, D, L, P	I, Q, M, D, L, P	Operand to which the contents of the field component are transferred.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Tag	Value
INDEX	a_index	4
MEMBER	"DB_1".Main_Field[-10]	First component of the "Main_Field[-10..10] of REAL" field in the data block "DB_1"
VALUE	a_real	Component with index 4 of the "Main_Field[-10..10] of REAL" field

The field component with index 4 is read from the "Main_Field[-10...10] of REAL" field and written to the "a_real" tag. The field component to be read is defined by the value at the parameter INDEX.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

FieldWrite: Write field (S7-1500)

Description

The "Write field" instruction is used to transfer the content of the tag at the VALUE input to a specific component of the field at the MEMBER output. You use the value at the INDEX input to specify the index of the field component that is described. At the MEMBER output, enter the first component of the field which is to be written to.

The data types of the field component at parameter MEMBER, the index and the tags at parameter VALUE must correspond to the data type of the instruction "Read field" because implicit conversion is not possible.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The field component specified at the input INDEX is not defined in the field specified at the output MEMBER.
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the instruction "Write field":

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
INDEX	Input	DINT	DINT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Index of field component that is written with the content of VALUE

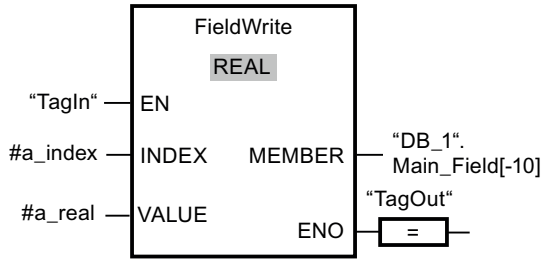
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
VALUE	Input	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD CHAR, WCHAR	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Operand whose contents are copied
MEMBER	Output	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, CHAR and WCHAR as components of an ARRAY tag	Binary numbers, integers, floating-point numbers, timers, DATE, TOD, LTOD, CHAR and WCHAR as components of an ARRAY tag	D, L	D, L	First component of the field to which the content of VALUE is written.

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
INDEX	a_index	4
VALUE	a_real	10.54
MEMBER	"DB_1".Main_Field[-10]	First component of the "Main_Field[-10..10] of REAL" field in the data block "DB_1"

The value "10.54" of the "a_real" tag is written to the field component with index 4 of the "Main_Field[-10 ... 10] of REAL" field. The index of the field component to which the content of the tag "a_real" is transferred is specified by the value at the input INDEX.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

BLKMOV: Move block (S7-1500)

Description

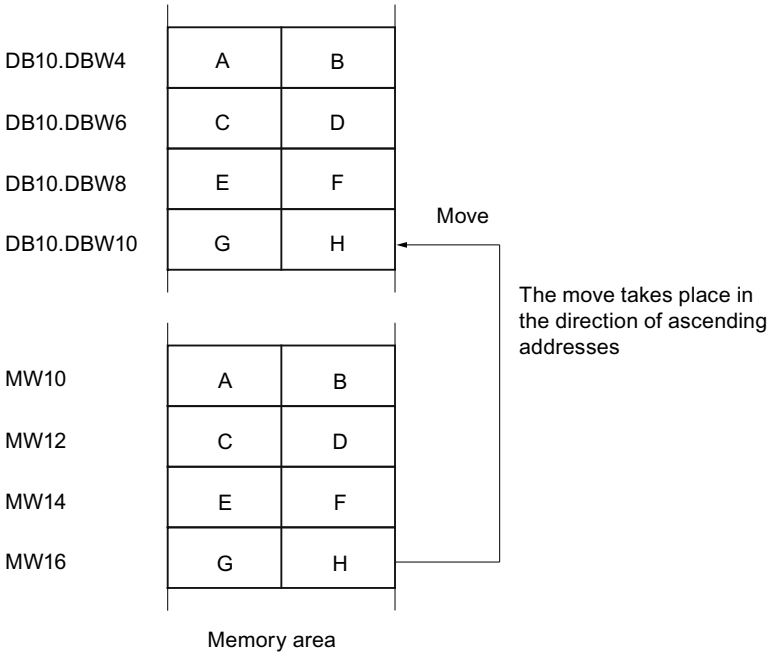
You can use the "Move block" instruction to move the content of a memory area (source area) to another memory area (destination area). The move operation takes place in the direction of ascending addresses. You use VARIANT to define the source and destination areas.

Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can use this tag in memory areas "with optimized block access".

The following figure shows the principle of the move operation:



Consistency of source and destination data

Make sure that the source data remain unchanged during the execution of the "Move block" instruction. Otherwise, consistency of the destination data cannot be guaranteed.

Interruptibility

There is no limit to the nesting depth.

Memory areas

You can use the "Move block" instruction to move the following memory areas:

- Areas of a data block
- Bit memory
- Process image input
- Process image output

General rules for moving

The source and destination areas must not overlap. If the source and destination area have different lengths, only the length of the smaller area will be moved.

If the source area is smaller than the destination area, the entire source area will be written to the destination area. The remaining bytes of the destination area remain unchanged.

If the destination area is smaller than the source area, the entire destination area will be written. The remaining bytes of the source area are ignored.

If a block of data type BOOL is moved, the tag must be addressed absolutely and the specified length of the area must be divisible by 8, otherwise the instruction cannot be executed.

Rules for moving character strings

You can use the "Move block" instruction to also move source and destination areas of the STRING data type. If only the source area is STRING data type, the characters will be moved that are actually contained in the character string. Information on the actual and maximum length is also written to the destination area. If the source and destination area are both STRING data type, the current length of the character string in the destination area is set to the number of characters actually moved.

If you want to move the information on the maximum and actual length of a character string, specify the areas in bytes to the SRCBLK and DSTBLK parameters.

Parameters

The following table shows the parameters of the instruction "Move block":

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output

Parameters	Declaration	Data type	Memory area	Description
SRCBLK	Input	VARIANT	I, Q, M, D, L, P	Specifies the memory area to be moved (source area).
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output in the RET_VAL parameter.
DSTBLK	Output [1)]	VARIANT	I, Q, M, D, L, P	Specifies the memory area to which the block is to be moved (destination area).
1) The DSTBLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

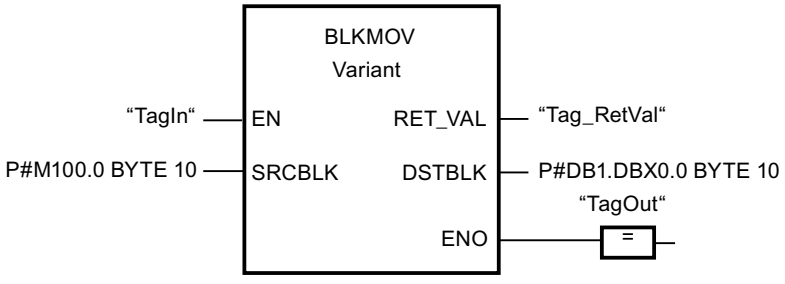
Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8092	The source or destination area is located in the load memory only.
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the SRCBLK parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the DSTBLK parameter.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction copies 10 bytes starting from MB100 and writes them to DB1. If an error occurs during the move operation, its error code is output in the "Tag_RetVal" tag.

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3164)

Basic information on FBD (Page 4937)

UBLKMOV: Move block uninterruptible (S7-1500)

Description

You can use the "Move block uninterruptible" instruction to move the content of a memory area (source area) to another memory area (destination area). The move operation takes place in the direction of ascending addresses. You use VARIANT to define the source and destination areas.

The move operation cannot be interrupted by other operating system activities. As a result the interrupt reaction time of the CPU can increase during the execution of the "Move block uninterruptible" instruction.

Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can use this tag in memory areas "with optimized block access".

Memory areas

You can use the "Move block uninterruptible" instruction to move the following memory areas:

- Areas of a data block
- Bit memory
- Process image input
- Process image output

General rules for moving

The source and destination area must not overlap during the execution of the "Move block uninterruptible" instruction. If the source area is smaller than the destination area, the entire source area will be written to the destination area. The remaining bytes of the destination area remain unchanged.

If the destination area is smaller than the source area, the entire destination area will be written. The remaining bytes of the source area are ignored.

If a source or destination area defined as a formal parameter is smaller than a destination or source area specified in the SRCBLK or DSTBLK parameter, no data is transferred.

If a block of data type BOOL is moved, the tag must be addressed absolutely and the specified length of the area must be divisible by 8, otherwise the instruction cannot be executed.

You can use the "Move block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Rules for moving character strings

You can use the "Move block uninterruptible" instruction to also move source and destination areas of the STRING data type. If only the source area is STRING data type, the characters will be moved that are actually contained in the character string. Information on the actual and maximum length are not written in the destination area. If the source and destination area are both STRING data type, the current length of the character string in the destination area is set to the number of characters actually moved. If areas of the STRING data type are moved, specify "1" as area length.

Parameters

The following table shows the parameters of the "Move block uninterruptible" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
SRCBLK	Input	VARIANT	I, Q, M, D, L, P	Specifies the memory area to be moved (source area).
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output in the RET_VAL parameter.
DSTBLK	Output [1]	VARIANT	I, Q, M, D, L, P	Specifies the memory area to which the block is to be moved (destination area).
1) The DSTBLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

Parameter RET_VAL

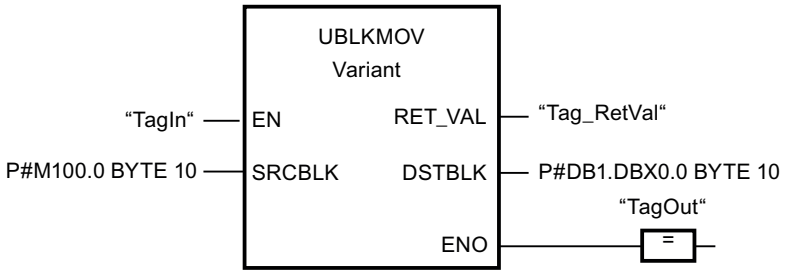
The following table shows the meaning of the values of the RET_VAL parameter:

Error code (W#16#...)	Explanation
0000	No error
8091	The source or destination area is located in the load memory only.
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the SRCBLK parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the DSTBLK parameter.

Error code (W#16#...)	Explanation
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction copies 10 bytes starting from MB100 and writes them to DB1. If an error occurs during the move operation, its error code is output in the "Tag_RetVal" tag.

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3164)

Basic information on FBD (Page 4937)

FILL: Fill block (S7-1500)

Description

You can use the "Fill block" instruction to fill a memory area (destination area) with the content of another memory area (source area). The "Fill block" instruction moves the content of the source area to the destination area until the destination area is completely written. The move operation takes place in the direction of ascending addresses.

You use VARIANT to define the source and destination areas.

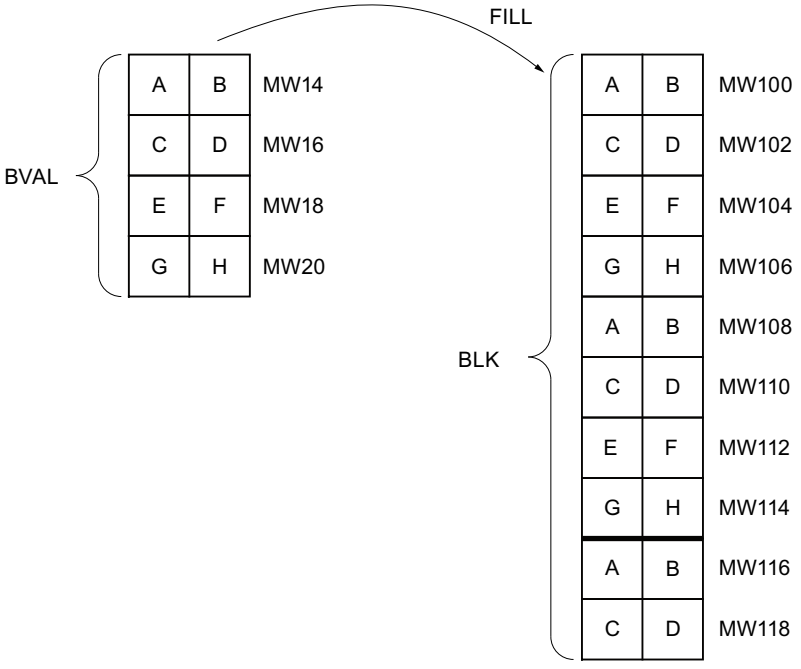
Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can use this tag in memory areas "with optimized block access".

For blocks with the "Optimized block access" attribute, you can use the "FILL_BLK instruction. Fill block" instruction.

The following figure shows the principle of the move operation:



Consistency of source and destination data

Note that while the instruction "Fill block" is being executed, the source data remains unchanged; otherwise the consistency of the destination data cannot be guaranteed.

Memory areas

You can use the "Fill block" instruction to move the following memory areas:

- Areas of a data block
- Bit memory
- Process image input
- Process image output

General rules for moving

The source and destination areas must not overlap. If the destination area to be preset is not an integer multiple of the length of the BVAL input parameter, the destination area is nevertheless written up to the last byte.

If the destination area to be preset is smaller than the source area, the function only copies as much data as can be written to the destination area.

If the destination or source area actually present is smaller than the assigned memory area for the source or destination area (BVAL, BLK parameters), no data is transferred.

If the ANY pointer (source or destination) is of the data type BOOL, it must be addressed absolutely and the length specified must be divisible by 8; otherwise the instruction is not executed.

If the destination area is data type STRING, the instruction writes the entire string including the administration information.

Rules for moving structures

When you transfer a structure as an input parameter you must bear in mind that the length of a structure is always oriented to an even number of bytes. The structure will need one byte of additional memory space if you declare a structure with an odd number of bytes.

Parameters

The following table shows the parameters of the "Fill block" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
BVAL	Input	VARIANT	I, Q, M, D, L, P	Specification of the memory area (source area), the content of which is used to fill the destination area at the BLK parameter.
RET_VAL	Output	INT	I, Q, M, D, L, P	Error information: If an error occurs during execution of the instruction, an error code is output in the RET_VAL parameter.
BLK	Output [1)]	VARIANT	I, Q, M, D, L, P	Specification of the memory area that will be filled with the content of the source area.
1) The BLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

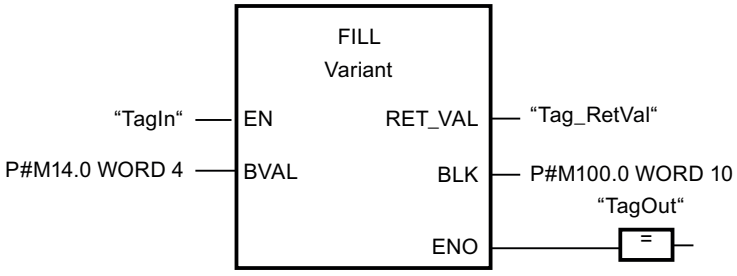
RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8092	The source or destination area is located in the load memory only.
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the BVAL parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the BLK parameter. The WSTRING, WCHAR, and BOOL, data types are not supported.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction copies the source area from MW14 to MW20 and fills the destination area from MW100 to MW118 with the content of the 4 words contained in the memory area in the BVAL parameter.

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3164)

Basic information on FBD (Page 4937)

Conversion operations (S7-1200, S7-1500)

CONVERT: Convert value (S7-1200, S7-1500)

Description

The "Convert value" instruction reads the content of the IN parameter and converts it according to the data types set in the instruction box. The converted value is output at the OUT output.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- The EN enable input has the signal state "0".
- Errors, such as an overflow, occur during execution.

Conversion options for bit strings

The bit strings BYTE and WORD cannot be selected in the instruction box. However, it is possible to specify an operand of data type DWORD or LWORD at a parameter of the instruction if the lengths of the input and output operands match. The operand will be interpreted and implicitly converted from the data type of a bit string according to the data type of the input or output parameter. The data type DWORD is for example interpreted as DINT/ UDINT and LWORD as LINT/ULINT. These conversion options are also available to you when "IEC check" is enabled.

Note

For CPUs of the S7-1500 series: The data types DWORD and LWORD can only be converted to or from data type REAL or LREAL.

During the conversion, the bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If no errors occur during the conversion, the signal state of enable output ENO = 1; if an error occurs during processing, the signal state of the enable output ENO = 0.

Parameters

The following table shows the parameters of the "Convert value" instruction:

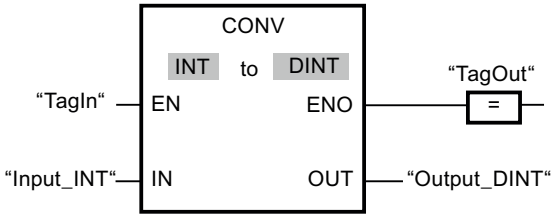
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
IN	Input	Bit strings, integers, floating-point numbers, CHAR, WCHAR, BCD16, BCD32	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Value to be converted.
OUT	Output	Bit strings, integers, floating-point numbers, CHAR, WCHAR, BCD16, BCD32	I, Q, M, D, L, P	I, Q, M, D, L, P	Result of the conversion

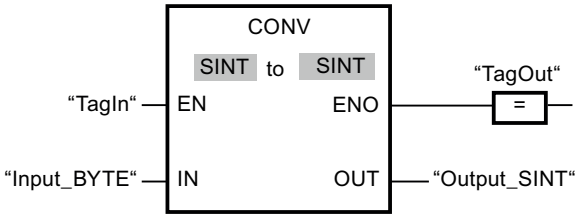
You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

Example

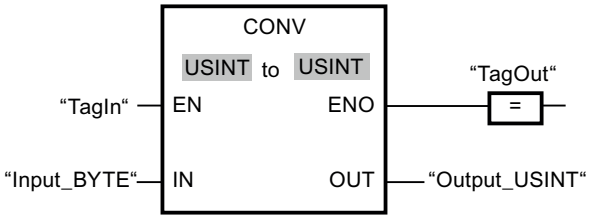
The following example shows conversion of an integer (16 bit) to another integer (32 bit):



The following example shows conversion of a byte (8 bit) to the integer SINT (8 bit):



The following example shows the conversion of a byte (8 bit) to the unsigned integer USINT (8 bit):



The conversions are possible, because the length of the two operands is the same.

See also

Overview of the valid data types (Page 2239)

Data type conversion for S7-1200 (Page 2465)

Basic information on FBD (Page 4937)

ROUND: Round numerical value (S7-1200, S7-1500)

Description

You can use the "Round numerical value" instruction to round the value at input IN to the nearest integer. The instruction interprets the value at input IN as a floating-point number and converts this to the nearest integer. If the input value is exactly between an even and odd number, then the even number is converted. The result of the instruction is output at the OUT output and can be queried there.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the "Round numerical value" instruction:

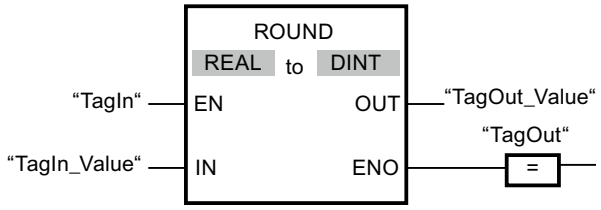
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value to be rounded.
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Result of the rounding

You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value	
IN	TagIn_Value	1.50000000	-1.50000000
OUT	TagOut_Value	2	-2

If the "TagIn" operand has signal state "1", the instruction is executed. The floating-point number at input "TagIn_Value" is rounded to the nearest even integer and sent to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Explicit conversions (Page 2487)

Basic information on FBD (Page 4937)

CEIL: Generate next higher integer from floating-point number (S7-1200, S7-1500)

Description

You can use the "Generate next higher integer from floating-point number" instruction to round the value at input IN to the next higher integer. The instruction interprets the value at input IN as a floating-point number and converts this to the next higher integer. The result of the instruction is output at the OUT output and can be queried there. The output value can be greater than or equal to the input value.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the instruction "Generate next higher integer from floating-point number":

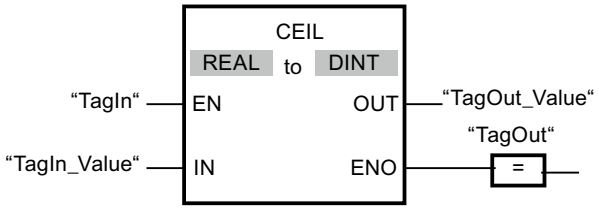
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value as floating-point number
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Result with the next higher integer

You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value	
IN	TagIn_Value	0.50000000	-0.50000000
OUT	TagOut_Value	1	0

If the "TagIn" operand has signal state "1", the instruction is executed. The floating-point number at input "TagIn_Value" is rounded to the next higher integer and sent to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Explicit conversions (Page 2487)

Basic information on FBD (Page 4937)

FLOOR: Generate next lower integer from floating-point number (S7-1200, S7-1500)

Description

You can use the "Generate next lower integer from floating-point number" instruction to round the value at input IN to the next lower integer. The instruction interprets the value at input IN as a floating-point number and converts this to the next lower integer. The result of the instruction is provided at output OUT and can be queried there. The output value can be less than or equal to the input value.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the instruction "Generate next lower integer from floating-point number":

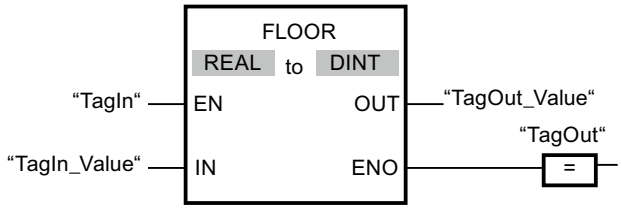
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value as floating-point number
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L, P	I, Q, M, D, L, P	Result with the next lower integer

You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value	
IN	TagIn_Value	0.50000000	-0.50000000
OUT	TagOut_Value	0	-1

If the "TagIn" operand has signal state "1", the instruction is executed. The floating-point number at input "TagIn_Value" is rounded to the next lower integer and displayed at output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Explicit conversions (Page 2487)

Basic information on FBD (Page 4937)

TRUNC: Truncate numerical value (S7-1200, S7-1500)

Description

You can use the "Truncate numerical value" instruction to form an integer from the value at input IN. The value at input IN is interpreted as a floating-point number. The instruction selects only the integer part of the floating-point number and sends this to output OUT without decimal places.

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- Errors, such as an overflow, occur during execution.

Parameters

The following table shows the parameters of the instruction "Truncate numerical value":

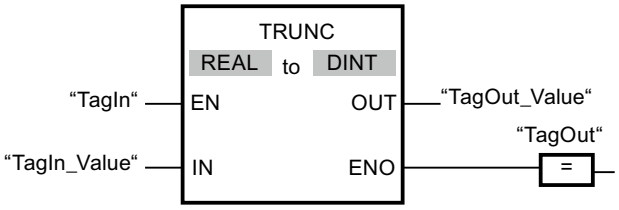
Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	Floating-point numbers	I, Q, M, D, L or constant	Input value as floating-point number
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L	Result with integer part of the floating-point number

You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value	
IN	TagIn_Value	1.50000000	-1.50000000
OUT	TagOut_Value	1	-1

If the "TagIn" operand has signal state "1", the instruction is executed. The integer part of the floating-point number at input "TagIn_Value" is converted to an integer and sent to output "TagOut_Value". If the instruction is executed without errors, the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Explicit conversions (Page 2487)

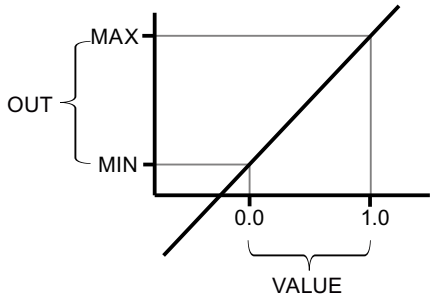
Basic information on FBD (Page 4937)

SCALE_X: Scale (S7-1200, S7-1500)

Description

You can use the "Scale" instruction to scale the value at the VALUE input by mapping it to a specified value range. When the instruction "Scale" is executed, the floating-point value at input VALUE is scaled to the value range, which is defined by the parameters MIN and MAX. The result of the scaling is an integer, which is stored at output OUT.

The following figure shows an example of how values can be scaled:



The "Scale" instruction works with the following equation:

OUT = [VALUE * (MAX – MIN)] + MIN

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input MIN is greater than or equal to the value at input MAX.
- The value of a specified floating-point number is outside the range of the normalized numbers according to IEEE-754.
- An overflow occurs.
- The value at input VALUE is NaN (Not a number = result of an invalid arithmetic operation).

Note

For more information on the conversion of analog values, refer to the respective manual.

Parameters

The following table shows the parameters of the "Scale" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
MIN	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Low limit of the value range

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
VALUE	Input	Floating-point numbers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Value to be scaled. If you enter a constant, you must declare it.
MAX	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	High limit of the value range
OUT	Output	Integers, floating-point numbers	I, Q, M, D, L	I, Q, M, D, L	Result of scaling

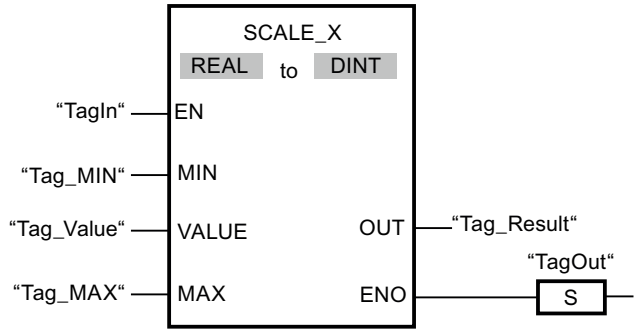
You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

For additional information on valid data types, refer to "See also".

For additional information on declaring constants, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
MIN	Tag_MIN	10
VALUE	Tag_Value	0.5
MAX	Tag_MAX	30
OUT	Tag_Result	20

If the "TagIn" operand has signal state "1", the instruction is executed. The value at input "Tag_Value" is scaled to the range of values defined by the values at inputs "Tag_MIN" and "Tag_MAX". The result is stored at output "Tag_Result". If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

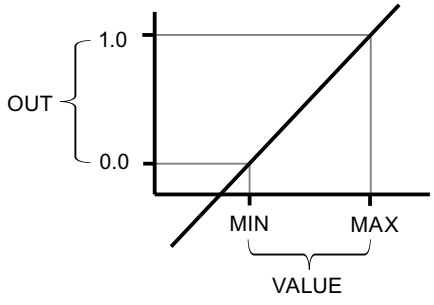
Basic information on FBD (Page 4937)

NORM_X: Normalize (S7-1200, S7-1500)

Description

You can use the instruction "Normalize" to normalize the value of the tag at the VALUE input by mapping it to a linear scale. You can use the parameters MIN and MAX to define the limits of a value range that is applied to the scale. Depending on the location of the normalized value in this value range, the result at output OUT is calculated and stored as a floating-point number. If the value to be normalized is equal to the value at input MIN, output OUT returns the value "0.0". If the value to be normalized is equal to the value at input MAX, output OUT returns the value "1.0".

The following figure shows an example of how values can be normalized:



The "Normalize" instruction works with the following equation:

OUT = (VALUE – MIN) / (MAX – MIN)

The enable output ENO has the signal state "0" if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value at input MIN is greater than or equal to the value at input MAX.
- The value of a specified floating-point number is outside the range of the normalized numbers according to IEEE-754.
- The value at input VALUE is NaN (result of an invalid arithmetic operation).

Note

For more information on the conversion of analog values, refer to the respective manual.

Parameters

The following table shows the parameters of the instruction "Normalize":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
MIN [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Low limit of the value range
VALUE [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Value to be normalized.
MAX [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	High limit of the value range
OUT	Output	Floating-point numbers	I, Q, M, D, L	I, Q, M, D, L	Result of the normalization
[1)] If you use constants in these three parameters, you only need to declare one of them.					

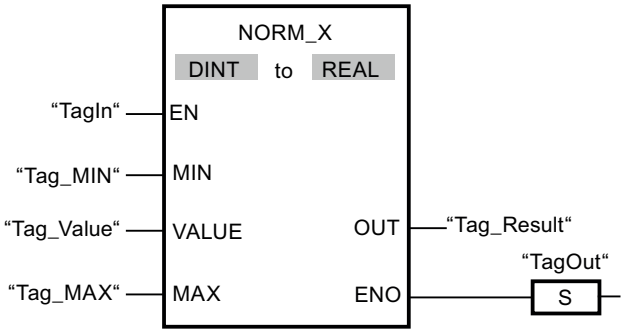
You can select the data types of the instruction from the "???" drop-down lists of the instruction box.

For additional information on valid data types, refer to "See also".

For additional information on declaring constants, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
MIN	Tag_MIN	10
VALUE	Tag_Value	20
MAX	Tag_MAX	30
OUT	Tag_Result	0.5

If the "TagIn" operand has signal state "1", the instruction is executed. The value at input "Tag_Value" is assigned to the range of values defined by the values at inputs "Tag_MIN" and "Tag_MAX". The tag value at input "Tag_Value" is normalized corresponding to the defined value range. The result is stored as a floating-point number at output "Tag_Result". If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Legacy (S7-1500)

SCALE: Scale (S7-1500)

Description

Use the "Scale" instruction to convert the integer in the IN parameter into a floating-point number, which can be scaled in physical units between a low limit value and a high limit value. You use the LO_LIM and HI_LIM parameters to specify the low limit and high limit of the value range to which the input value is scaled. The result of the instruction is output at the OUT parameter.

The "Scale" instruction works with the following equation:

OUT = [((FLOAT (IN) – K1)/(K2–K1)) * (HI_LIM–LO_LIM)] + LO_LIM

The values of the constants "K1" and "K2" are determined by the signal state of the BIPOLAR parameter. The following signal states are possible in the BIPOLAR parameter:

- Signal state "1": It is assumed that the value at the IN parameter is bipolar and in a value range between -27648 and 27648. In this case, the constant "K1" has the value -27648.0 and the constant "K2" the value +27648.0.
- Signal state "0": It is assumed that the value at the IN parameter is unipolar and in a value range between 0 and 27648. In this case, the constant "K1" has the value 0.0 and the constant "K2" the value +27648.0.

When the value at the IN parameter is greater than the value of the constant "K2", the result of the instruction is set to the value of the high limit (HI_LIM) and an error is output.

When the value at the IN parameter is less than the value of the constant "K1", the result of the instruction is set to the value of the low limit value (LO_LIM) and an error is output.

When the indicated low limit value is greater than the high limit value (LO_LIM > HI_LIM), the result is scaled in reverse proportion to the input value.

Parameters

The following table shows the parameters of the "Scale" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	INT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value to be scaled.
HI_LIM	Input	REAL	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	High limit
LO_LIM	Input	REAL	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Low limit
BIPOLAR	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Indicates if the value at IN parameter is to be interpreted as bipolar or unipolar. The parameter can assume the following values: 1: Bipolar 0: Unipolar
OUT	Output	REAL	I, Q, M, D, L, P	I, Q, M, D, L, P	Result of the instruction
RET_VAL	Output	WORD	I, Q, M, D, L, P	I, Q, M, D, L, P	Error information

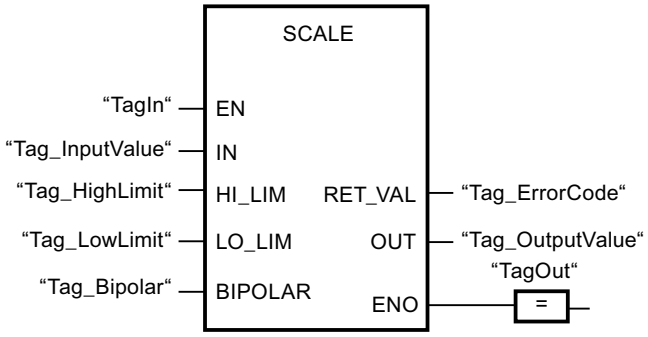
Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
0008	The value of the IN parameter is greater than the value of the "K2" constant or less than the value of the "K1" constant.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal values in the program editor. You will find information on changing display formats in "See also".	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	Tag_InputValue	22
HI_LIM	Tag_HighLimit	100.0
LO_LIM	Tag_LowLimit	0.0
BIPOLAR	Tag_Bipolar	1
OUT	Tag_OutputValue	50.03978588
RET_VAL	Tag_ErrorCode	W#16#0000

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3164)

Basic information on FBD (Page 4937)

UNSCALE: Unscale (S7-1500)

Description

You use the "Unscale" instruction to unscale the floating-point number in the IN parameter into physical units between a low limit and a high limit and convert it into an integer. You use the LO_LIM and HI_LIM parameters to specify the low limit and high limit of the value range to which the input value is unscaled. The result of the instruction is output at the OUT parameter.

The "Unscale" instruction works with the following equation:

OUT = [((IN–LO_LIM)/(HI_LIM–LO_LIM)) * (K2–K1)] + K1

The values of the "K1" and "K2" constants are determined by the signal state at the BIPOLAR parameter. The following signal states are possible at the BIPOLAR parameter:

- Signal state "1": It is assumed that the value at the IN parameter is bipolar and in a value range between -27648 and 27648. In this case, the constant "K1" has the value -27648.0 and the constant "K2" the value +27648.0.
- Signal state "0": It is assumed that the value at the IN parameter is unipolar and in a value range between 0 and 27648. In this case, the constant "K1" has the value 0.0 and the constant "K2" the value +27648.0.

When the value at parameter IN is not within the limits defined by HI_LIM and LO_LIM, an error is output and the result is set to the nearest limit.

When the indicated low limit value is greater than the high limit value (LO_LIM > HI_LIM), the result is scaled in reverse proportion to the input value.

Parameters

The following table shows the parameters of the "Unscale" instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	REAL	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value to be unscaled to an integer value.
HI_LIM	Input	REAL	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	High limit
LO_LIM	Input	REAL	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Low limit
BIPOLAR	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Indicates whether the value at the IN parameter is to be interpreted as bipolar or unipolar. The parameter can assume the following values: 1: Bipolar 0: Unipolar
OUT	Output	INT	I, Q, M, D, L, P	I, Q, M, D, L, P	Result of the instruction
RET_VAL	Output	WORD	I, Q, M, D, L, P	I, Q, M, D, L, P	Error information

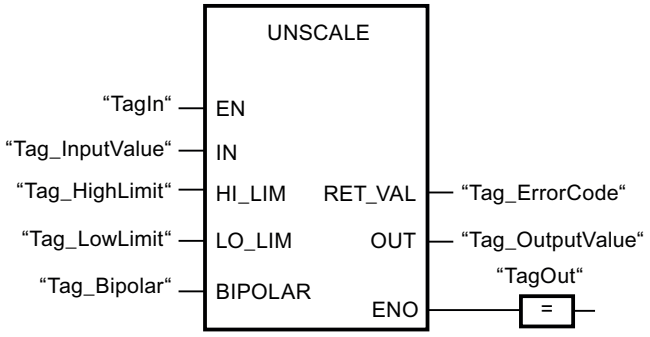
RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
0008	The value of the IN parameter is greater than the value of the high limit (HI_LIM) or less than the value of the low limit (LO_LIM).
General error information	See also: "GET_ERR_ID: Get error ID locally"
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats here: AUTOHOTSPOT	

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_InputValue	50.03978588
HI_LIM	Tag_HighLimit	100.0
LO_LIM	Tag_LowLimit	0.0
BIPOLAR	Tag_Bipolar	1
OUT	Tag_OutputValue	22
RET_VAL	Tag_ErrorCode	W#16#0000

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3164)

Basic information on FBD (Page 4937)

Program control operations (S7-1200, S7-1500)

JMP: Jump if RLO = 1 (S7-1200, S7-1500)

Description

You can use the "Jump if RLO = 1" instruction to interrupt the linear execution of the program and resume it in another network. The target network must be identified by a jump label (LABEL). The jump label description is entered in the placeholder above the instruction box.

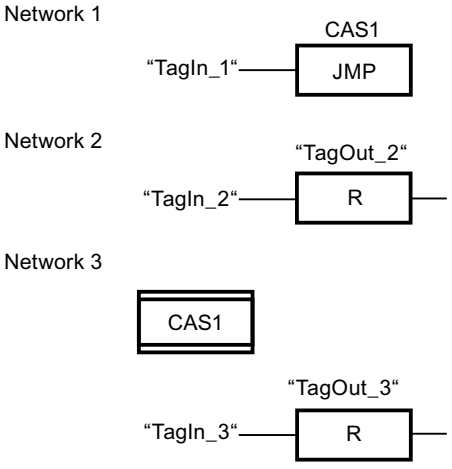
The specified jump label must be in the same block in which the instruction is executed. The name you specify can only occur once in the block. Only one jumping coil can occur in a network.

If the result of logic operation (RLO) at the input of the instruction is "1", the jump to the network identified by the jump label is executed. The jump direction can be towards higher or lower network numbers.

If the condition at the input of the instruction is not fulfilled (RLO = 0), execution of the program continues in the next network.

Example

The following example shows how the instruction works:



If the "TagIn_1" operand has signal state "1", the instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. If input "TagIn_3" has the signal state "1", output "TagOut_3" is reset.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

JMPN: Jump if RLO = 0 (S7-1200, S7-1500)

Description

You can use the instruction "Jump if RLO = 0" to interrupt the linear execution of the program and resume it in another network, when the result of logic operation at the input of the instruction is "0". The target network must be identified by a jump label (LABEL). The jump label description is entered in the placeholder above the instruction box.

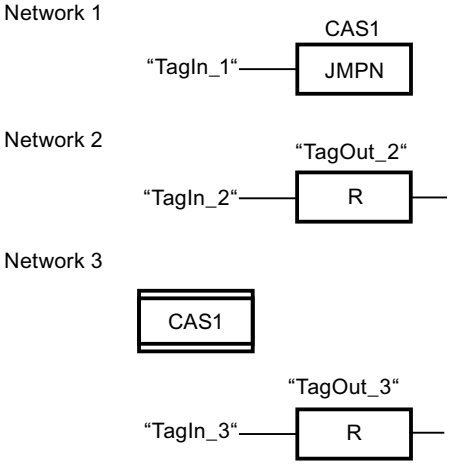
The specified jump label must be in the same block in which the instruction is executed. The name you specify can only occur once in the block. Only one jumping coil can occur in a network.

If the result of logic operation (RLO) at the input of the instruction is "0", the jump to the network identified by the jump label is executed. The jump direction can be towards higher or lower network numbers.

If the result of the logic operation RLO at the input of the instruction is "1", execution of the program continues in the next network.

Example

The following example shows how the instruction works:



If the "TagIn_1" operand has signal state "0", the instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. If input "TagIn_3" has the signal state "1", output "TagOut_3" is reset.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

LABEL: Jump label (S7-1200, S7-1500)

Description

The jump label identifies a destination network in which the execution of the program can be resumed after the execution of a jump instruction.

The jump label and the instruction in which the jump label is specified must be located in the same block. The name of a jump label can only be assigned once in a block. You can declare up to 32 jump labels when you use a CPU S7-1200 and a maximum of 256 jump labels when you use a CPU S7-1500.

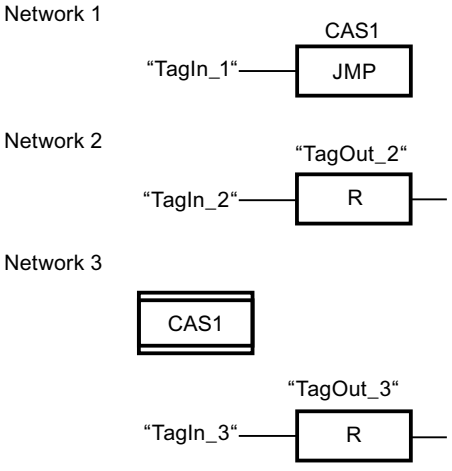
Only one jump label can be placed in a network. Each jump label can jump to several locations.

Adhere to the following grammatical rules for jump labels:

- Letters (a to z, A to Z)
- A combination of letters and numbers. Check that the order is correct, i.e. first the letters, then the numbers (a - z, A - Z, 0 - 9).
- Special characters or a combination of letters and numbers must not be used in reverse order, i.e. first the numbers and then the letters (0-9, a - z, A - Z).

Example

The following example shows how the instruction works:



If the "TagIn_1" operand has signal state "1", the instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. If the "TagIn_3" input has the signal state "1", output "TagOut_3" is reset.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

JMP_LIST: Define jump list (S7-1200, S7-1500)

Description

You can use the "Define jump list" instruction to define several conditional jumps and continue the program execution in a specific network depending on the value of the K parameter.

You define the jumps with jump labels (LABEL), which you specify at the outputs of the instruction box. The number of outputs can be expanded in the instruction box. You can declare up to 32 outputs when you use a CPU S7-1200 and a maximum of 256 outputs when you use a CPU S7-1500.

The numbering of the outputs begins with the value "0" and is continued in ascending order with each new output. Only jump labels can be specified at the outputs of the instruction. Instructions or operands cannot be specified.

The value of the K parameter specifies the number of the output and thus the jump label where the program execution is to be resumed. If the value in the K parameter is greater than the number of available outputs, the program execution is resumed in the next network of the block.

The "Define jump list" instruction is only executed if the signal state is "1" at the EN enable input.

Parameters

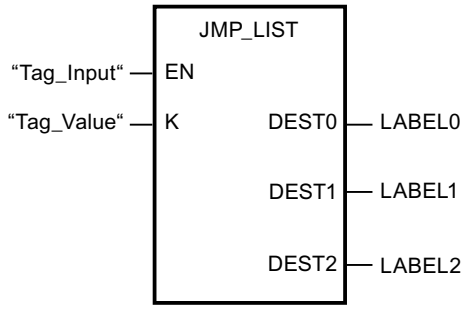
The following table shows the parameters of the "Define jump list" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, L, D	Enable input
K	Input	UINT	I, Q, M, L, D or constant	Specifies the number of the output and thus the jump that is to be made.
DEST0	-	-	-	First jump label
DEST1	-	-	-	Second jump label
DESTn	-	-	-	Optional jump labels

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand/jump label	Value
K	Tag_Value	1
DEST0	LABEL0	Jump to the network that is identified with the jump label "LABEL0".
DEST1	LABEL1	Jump to the network that is identified with the jump label "LABEL1".
DEST2	LABEL2	Jump to the network that is identified with the jump label "LABEL2".

If the "Tag_Input" operand has signal state "1", the instruction is executed. The execution of the program is continued according to the value of the operand "Tag_Value" in the network that is identified with the jump label "LABEL1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SWITCH: Jump distributor (S7-1200, S7-1500)

Description

You can use the "Jump distributor" instruction to define multiple program jumps to be executed depending on the result of one or more comparison instructions.

You specify the value to be compared in the K parameter. This value is compared with the values that are provided by the various inputs. You can select the comparison method for each individual input. The availability of the various comparison instructions depends on the data type of the instruction.

The following table shows the comparison instructions that are available depending on the selected data type:

Data type		Instruction	Syntax
S7-1200	S7-1500		
Bit strings	Bit strings	Equal	==
		Not equal	<>
Integers, floating-point numbers, TIME, DATE, TOD	Integers, floating-point numbers, TIME, LTIME, DATE, TOD, LTOD, LDT	Equal	==
		Not equal	<>
		Greater or equal	>=
		Less or equal	<=
		Greater than	>
		Less than	<

You can select the data type of the instruction from the "???" drop-down list of the instruction box. If you select a comparison instruction and the data type of the instruction is not yet defined, then the "???" drop-down list will only list those data types that are permitted for the selected comparison instruction.

Execution of the instruction begins with the first comparison and runs until a comparison condition is met. If a comparison condition is met, the subsequent comparison conditions are not considered. If none of the specified comparison conditions are fulfilled, the jump is executed at output ELSE. If not jump label is defined at output ELSE, the linear execution of the program is not interrupted but instead continued in the next network.

In its initial state the instruction box contains at least 2 outputs (DEST0 and DEST1). The number of outputs can be extended. The numbering of the outputs begins with the value "0" and is continued in ascending order with each new output. Specify jump labels (LABEL) at the outputs of the instruction. Instructions or operands cannot be specified at the outputs of the instruction.

An input is automatically inserted to each additional output. The jump programmed at an output is executed if the comparison condition of the corresponding input is fulfilled.

Parameters

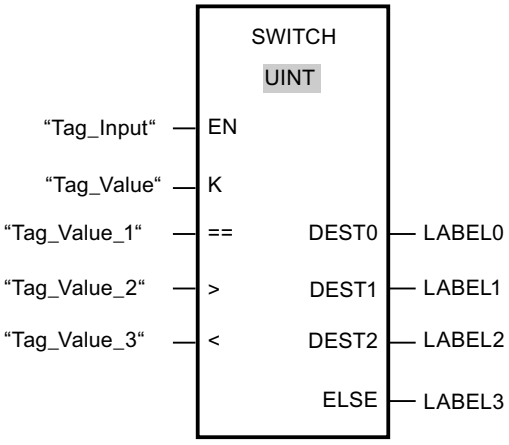
The following table shows the parameters of the instruction "Jump distributor":

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I, Q, M, D, L	Enable input
K	Input	UINT	UINT	I, Q, M, D, L or constant	Specifies the value to be compared.
<Comparison values>	Input	Bit strings, integers, floating-point numbers, TIME, DATE, TOD	Bit strings, integers, floating-point numbers, TIME, LTIME, DATE, TOD, LTOD, LDT	I, Q, M, D, L or constant	Input values with which the value of the parameter K is compared.
DEST0	-	-	-	-	First jump label
DEST1	-	-	-	-	Second jump label
DEST(n)	-	-	-	-	Optional jump labels: • S7-1200: n = 2 to 32 • S7-1500: n = 2 to 256
ELSE	-	-	-	-	Program jump which is executed if none of the comparison conditions are fulfilled.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand/jump label	Value
K	Tag_Value	23
==	Tag_Value_1	20
>	Tag_Value_2	21
<	Tag_Value_3	19
DEST0	LABEL0	Jump to jump label "LABEL0", if the value of parameter K is equal to 20.
DEST1	LABEL1	Jump to jump label "LABEL1", if the value of parameter K is greater than 21.
DEST2	LABEL2	Jump to jump label "LABEL2", if the value of parameter K is less than 19.
ELSE	LABEL3	Jump to jump label "LABEL3", if the none of the comparison conditions are fulfilled.

If the operand "Tag_Input" changes to signal state "1", the instruction is executed. The execution of the program is continued in the network that is identified with the jump label "LABEL1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

RET: Return (S7-1200, S7-1500)

Description

You can use the "Return" instruction to stop the execution of a program block conditionally or absolutely. The signal state of the return value (operand) is mapped in the enable output ENO of the calling program block when the program block is exited.

There are three ways that the execution of a program block can be ended:

Termination of the program block	Description
Without a call of the instruction	The program block is exited after the execution of the last network. The enable output ENO of the calling program block is set to signal state "1".
Call of the instruction with preceding logic operation (see example)	If the preceding logic operation is fulfilled, the program execution in the currently called program block is ended and the program block is exited. (Conditional program block end) The program execution is continued in the calling program block (e.g. an OB) after the program block call. The enable output ENO of the calling program block corresponds to the operand.
Call of the instruction without preceding logic or the instruction is directly connected to the left power rail.	The program block is exited unconditionally. (Absolute program block end) The enable output ENO of the calling program block corresponds to the operand.

If an organization block (OB) is ended, another program block is selected by the execution level system and started or further executed.

- If the OB program cycle is ended, it is restarted.
- If an OB is ended that has interrupted another program block (e.g. an interrupt OB), then the interrupted program block (e.g. program cycle OB) is further executed.

Note

RET instruction in relation with JMP and JMPN

If a network already contains the "JMP: Jump if RLO = 1" or "JMPN: Jump if RLO = 0 instruction, the RET: Return" instruction must not be used. Only one jumping coil may be used in a network

Possible settings of the return value (operand)

The return value of the instruction can assume the following values:

- Ret (RLO, corresponds to the result of logic operation RLO. The signal state "1" is supplied to the enable output ENO of the calling program block because the RET instruction is only then executed as a conditional instruction when the condition is TRUE.)
- Ret True or Ret False (the respective value of the constant, TRUE or FALSE, is supplied to the calling program block.)
- Ret Value (the value of the Boolean tag <Operand> is supplied to the calling program block.)

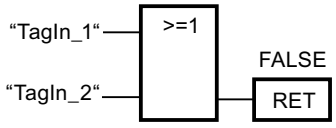
To set the return value of the instruction, click the small yellow triangle of the instruction and select the desired value in the drop-down list.

The following table shows the status of the calling program block when the instruction is programmed in a network within the called program block:

RLO	Return value	ENO of the calling program block
1	RLO	1
	TRUE	1
	FALSE	0
	<Operand> Boolean tag with possible memory areas I, Q, M, D, L, T, and C	<Operand>
0	RLO	The program execution is continued in the next network of the called program block.
	TRUE	
	FALSE	
	<Operand>	

Example

The following example shows how the instruction works:



If one of the operands "TagIn_1" or "TagIn_2" have signal state "1", the instruction is executed. Program execution is ended in the called program block and continues in the calling program block. The enable output ENO of the calling program block is reset to signal state "0".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Runtime control (S7-1200, S7-1500)

ENDIS_PW: Limit and enable password legitimation (S7-1200, S7-1500)

Description

You use the "Limit and enable password legitimation" instruction to specify whether passwords configured for the CPU are legitimized or not. In this way, you can prevent legitimized connections, even if the correct password is known.

When the instruction is called and the REQ parameter has the signal state "0", the currently set state is displayed at the output parameters. If changes have been made to the input parameters, these are not transferred to the output parameters.

When the instruction is called and the REQ parameter has the signal state "1", the signal state is taken from the input parameters (F_PWD, FULL_PWD, R_PWD, HMI_PWD).

- If signal state "0" is present, authorization via password is not allowed.
- If signal state "1" is present, the password can be used.

Disable or enabling of the passwords can be allowed or prohibited individually. For example, all passwords can be prohibited except the fail-safe password. You can thus limit access options to a small user group. The output parameters (F_PWD_ON, FULL_PWD_ON, R_PWD_ON, HMI_PWD_ON) always show the current status of the password use, regardless of the REQ parameter.

Passwords that are not configured must have the signal state TRUE at the input and return the signal state TRUE at the output. The fail-safe password can only be configured for an F-CPU and must therefore always be interconnected with the signal state TRUE in a standard CPU. If the instruction returns an error, the call has no effect and the previous lock is still in effect.

Disabled passwords can be enabled again under the following conditions:

- The CPU is reset to its factory settings.
- The front panel of the S7-1500 CPU supports a dialog that allows you to navigate to the appropriate menu in which the passwords can be enabled again.
- When you call the "Limit and enable password legitimation" instruction, the input parameter of the desired password has the signal state "1".
- Set the mode selector to STOP. The restriction to password legitimation is re-established as soon as the switch is set back to RUN.
- Plugging an empty memory card (transfer module or program card) into an S7-1200 CPU.
- The transition from POWER OFF to POWER ON disables the protection in the S7-1200 CPU. The "Limit and enable password legitimation" instruction must be called once again in the program (e.g. in the startup OB).

Note

The "Limit and enable password legitimation" instruction blocks access from HMI systems if the HMI password has not been enabled.

Note

Existing legitimized connections retain their access rights and you cannot use the "Limit and enable password legitimation" instruction to restrict them.

Preventing unintentional lockout on an S7-1500 CPU

The settings can be made on the front panel of the CPU and the CPU saves the most recent setting.

To prevent accidental lockout, the protection can be disabled by setting the mode selector to STOP with an S7-1500 CPU. The protection is automatically enabled again by setting mode selector to RUN without having to call the "Limit and enable password legitimation" instruction again or taking additional actions on the front panel.

Preventing unintentional lockout on an S7-1200 CPU

Because the S7-1200 CPU does not have a mode switch, protection is disabled with POWER OFF - POWER ON. This means that it is possible and advisable to prevent accidental lockout with the help of specific program sequences within your program.

To do so, program a time control either by means of a cyclic interrupt OB or a timer in the Main OB (OB 1). This gives you the option of calling the "Limit and enable password legitimation" instruction in the respective OB (e.g., OB 1 or OB 35) relatively quickly after a transition from POWER OFF to POWER ON and the associated disabling of the protection. Call the instruction in the startup OB (OB 100) to keep the time window in which the instruction is inactive and there are no limitations for password legitimation relatively small. This procedure will give you the best possible protection from unauthorized access.

If there has been an accidental lockout, you can skip the call in the startup OB (by querying an input parameter, for example) and you have the set time (for example, 10 seconds up to 1 minute) to establish a connection to the CPU before the lock becomes active once again.

If there is no timer in your program code and a lockout has occurred, insert an empty transfer card or an empty program card into the CPU. The empty transfer card or program card deletes the internal load memory of the CPU. You must then download the user program once again from STEP 7 to the CPU.

Procedure for lost passwords with an S7-1200 CPU

If you have lost the password for a password-protected S7-1200 CPU, delete the password-protected program with an empty transfer card or program card. The empty transfer card or program card deletes the internal load memory of the CPU. You can then load a new user program from STEP 7 Basic in the CPU.

⚠ WARNING

Inserting an empty transfer card

When you insert a transfer card in a CPU during runtime, the CPU goes to STOP mode. If operating states are unstable, controllers may fail and thus cause the uncontrolled operation of the controlled devices. This leads to an unforeseen operation of the automation system which can cause deadly or serious injuries and/or damage to property.

Once you have pulled the transfer card, the content of the transfer card is available in the internal load memory. Make sure that the card does not include a program at this point.

⚠ WARNING

Inserting an empty program card

When you insert a program card in a CPU during runtime, the CPU goes to STOP mode. If operating states are unstable, controllers may fail and thus cause the uncontrolled operation of the controlled devices. This leads to an unforeseen operation of the automation system which can cause deadly or serious injuries and/or damage to property.

Make sure that the program card is empty. The internal load memory is copied to the empty program card. Once you have pulled the previously empty program card, the internal load memory is empty.

You must remove the transfer card or program card before you put the CPU into RUN mode.

Effects of password use on the operating modes

The following table shows what effects the password use by means of the "Limit and enable password legitimation" instruction has on the operating modes and the corresponding user actions.

Action	Password protection by means of the instruction
Basic state after • Operating mode switch to STOP • Reset memory manually (PG, switch, change of MC (Motion Control)) • Reset to factory setting	Not activated (no restrictions)
Basic state after POWER ON	• S7-1200 CPU: The lock is disabled and the instruction must be called once again in the program (e.g., in the startup OB). • S7-1500 CPU: Enabled (if a lock was activated before POWER OFF). The option of not allowing passwords is retentive.
Operating mode transition RUN/STARTUP/HOLD -> STOP (by terminating the instruction, an error or communication) or STOP -> STARTUP/RUN/HOLD	Activated Passwords still cannot be used.

Parameters

The following table shows the parameters of the "Limit and enable password legitimation" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
REQ	Input	BOOL	I, Q, M, D, L or constant	When the REQ parameter has the signal state "0", the currently set signal state of the passwords is queried.
F_PWD	Input	BOOL	I, Q, M, D, L or constant	Read/write access including fail-safe • F_PWD = "0": Do not allow password • F_PWD = "1": Allow password

Parameters	Declaration	Data type	Memory area	Description
FULL_PWD	Input	BOOL	I, Q, M, D, L or constant	Read/write access • FULL_PWD = "0": Do not allow password • FULL_PWD = "1": Allow password
R_PWD	Input	BOOL	I, Q, M, D, L or constant	Read access • R_PWD = "0": Do not allow password • R_PWD = "1": Allow password
HMI_PWD	Input	BOOL	I, Q, M, D, L or constant	HMI access • HMI_PWD = "0": Do not allow password • HMI_PWD = "1": Allow password
F_PWD_ON	Output	BOOL	I, Q, M, D, L	Read/write access including fail-safe • F_PWD_ON = "0": Password not allowed • F_PWD_ON = "1": Password allowed
FULL_PWD_ON	Output	BOOL	I, Q, M, D, L	Read/write access status • FULL_PWD_ON = "0": Password not allowed • FULL_PWD_ON = "1": Password allowed
R_PWD_ON	Output	BOOL	I, Q, M, D, L	Read-access status • R_PWD_ON = "0": Password not allowed • R_PWD_ON = "1": Password allowed
HMI_PWD_ON	Output	BOOL	I, Q, M, D, L	HMI-access status • HMI_PWD_ON = "0": Password not allowed • HMI_PWD_ON = "1": Password allowed
RET_VAL	Output	WORD	I, Q, M, D, L	Error information

You can find additional information on valid data types under "See also".

Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8090	The "Limit and enable password legitimation" instruction is not supported
80D0	The password for fail-safe is not configured. The signal state must be TRUE for standard CPUs.
80D1	The read/write access is not configured
80D2	The read access is not configured
80D3	The HMI access is not configured
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

PC systems (S7-1200, S7-1500)

SHUT_DOWN: Shut down target system (S7-1200, S7-1500)

Description

You can use the instruction "SHUT_DOWN: Shutdown target system" to shut down the PC-based automation system and restart the S7 software controller CPU 150xS or Windows on the PC-based automation system.

The instruction is available in the "Instructions" task card under Basic instructions > Program control > Runtime control.

A restart is, for example, advisable in the following situations:

- An industrial UPS (uninterruptible power supply) reports a power failure via a digital input.
- Windows stops responding or shows a "blue screen".
- Too many error OBs are called in the user program.

Parameter

The following table shows the parameters of the "Shutdown target system" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	UINT	I, Q, M, D, L or constant	MODE = 1: Shut down CPU 150xS and Windows The CPU goes to STOP and saves the retentive data. Then the CPU and Windows are shut down. The system must be restarted manually. MODE = 2: Restart CPU 150xS The CPU goes to STOP and saves the retentive data. Then the CPU is shut down and restarted. MODE = 3: Restart Windows The CPU remains in RUN. Windows is restarted (as of TIA Portal V14, MODE 3 is only retained for backward compatibility. We recommend using MODE 4 or MODE 5). MODE = 4: Windows is shut down and restarted correctly. The CPU remains in RUN. MODE = 5: Restart Windows (Comparable with MODE 3; Exception: MODE 5 should only be used if Windows has crashed).
COM-MENT	Input	STRING	I, Q, M, D, L	With Mode = 1, 3 and 4, you can specify a reason for the restart. The reason is output in the Windows Event Log.
Ret_Val	Return	WORD	Q, M, D, L	Ret_Val = 0: No faults Ret_Val = 8090: The value passed to MODE is not supported. Ret_Val = 8091: Windows does not respond to the call of the Shut_Down instruction (only valid for Mode 3 and 4). Ret_Val = 8092: Please contact SIMATIC Customer Support if this error occurs (only valid for Mode 3 and 4).

RE_TRIGR: Restart cycle monitoring time (S7-1200, S7-1500)

Description

The "Restart cycle monitoring time" instruction is used to restart the cycle monitoring time of the CPU. The cycle monitoring time then restarts with the time you have set in the CPU configuration.

If the instruction is called in a block with a higher priority, such as a hardware interrupt or diagnostic error interrupt, the instruction is not executed and the enable output ENO returns signal state "0".

The instruction executes completely within a time span (10 times the maximum program cycle), regardless of the number of calls. Once this time has expired, the program cycle can no longer be prolonged.

Calling the instruction

The following call conditions apply:

- The following applies to a CPU of the S7-1200 series:
  In firmware versions <= 2.2, you can call the instruction only in a program cycle organization block of priority 1. This corresponds to the lowest priority of all organization blocks. If the instruction is called in an organization block with a higher priority, the instruction is not executed and the result (BR bit, enable output ENO) is always "0".
  In firmware versions >= 2.2, you can call the instruction regardless of the priority in all organization blocks.
- The following applies to a CPU of the S7-1500 series:
  You can call the instruction regardless of the priority in all organization blocks.

Parameters

The instruction "Restart cycle monitoring time" has no parameters.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

STP: Exit program (S7-1200, S7-1500)

Description

The instruction "Exit program" instruction is used to set the CPU to STOP mode and therefore to terminate the execution of the program. The effects of changing from RUN to STOP depend on the CPU configuration.

When the (RLO) at the input of the instruction is "1", the CPU changes to STOP mode and the execution of the program is terminated. The signal state at the output of the instruction is not evaluated.

If the RLO at the input of the instruction is "0", then the instruction will not be executed.

Parameters

The instruction "Exit program" has no parameters.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

GET_ERROR: Get error locally (S7-1200, S7-1500)

Description

The "Get error locally" instruction is used to query the occurrence of errors within a program block. This normally involves programming or access errors. If the system reports an error during the execution of the program block, detailed information is stored in the operand at the ERROR output for the first error to occur during the execution of the block since the last execution of the instruction.

Only operands of the "ErrorStruct" system data type can be specified at the ERROR output. The "ErrorStruct" system data type specifies the exact structure in which the information about the error is stored. Using additional instructions, you can evaluate this structure and program an appropriate response. If several errors occur in the program block, the error information about the next error to occur is output by the instruction only after the first error that occurred has been remedied.

Note

Output ERROR

The ERROR output is only changed if error information is present. To set the output back to "0" after handling an error, you have the following options:

- Declare the tag in the "Temp" section of the block interface.
- Reset the tag to "0" before calling the instruction.
- Query the enable output ENO.

The enable output ENO is only set if the enable input EN returns signal state "1" and error information is present. If one of these conditions does not apply, then the remaining program execution is not affected by the instruction.

Note

Activation of the local error handling

As soon as you insert the instruction in the program code of a program block, the local error handling is activated and default system reactions are ignored when errors occur.

Options for error handling

You can find an overview of the options for error handling here: AUTOHOTSPOT

For a detailed example of local error handling that contains several error handling options, refer to: AUTOHOTSPOT

Parameters

The following table shows the parameters of the "Get error locally" instruction:

Parameters	Declaration	Data type	Memory area	Description
ERROR	Output	ErrorStruct	D, L	Error information

Data type "ErrorStruct"

The "ErrorStruct" data type can be inserted in a global data block or in the block interface. You can insert the data type more then once if you assign a different name for the data structure each time. The data structure and the name of individual structure elements cannot be changed. If you save the error information in a global data block, it can also be read out by other program blocks.

The following table shows the structure of the "ErrorStruct" data type:

Structure component		Data type	Description
ERROR_ID		WORD	Error ID
FLAGS		BYTE	Shows whether the error occurred during a program block call. 16#01: Error during a program block call 16#00: No error during a program block call
REACTION		BYTE	Default reaction: 0: Ignore (write error) 1: Continue with substitute value "0" (read error) 2: Skip instruction (system error)
CODE_ADDRESS		CREF	Information about the address and type of the program block
	BLOCK_TYPE	BYTE	Program block type in which the error occurred: 1: Organization block (OB) 2: Function (FC) 3: Function block (FB)
	CB_NUMBER	UINT	Number of the code block
	OFFSET	UDINT	Reference to the internal memory
MODE		BYTE	Information about the address of an operand
OPERAND_NUMBER		UINT	Operand number of the machine command
POINTER_NUMBER_LOCATION		UINT	(A) Internal pointer
SLOT_NUMBER_SCOPE		UINT	(B) Storage area in internal memory
DATA_ADDRESS		NREF	Information about the address of an operand
	AREA	BYTE	(C) Memory area: L: 16#40...16#7F, 16#86, 16#87, 16#8E, 16#8F, 16#C0...16#FF I: 16#81 Q: 16#82 M: 16#83 DB: 16#40, 16#84, 16#85, 16#8A, 16#8B PI: 16#01 PQ: 16#02 Technological objects: 16#04

Structure component		Data type	Description
	DB_NUMBER	UINT	(D) Number of the data block
	OFFSET	UDINT	(E) Relative address of the operand

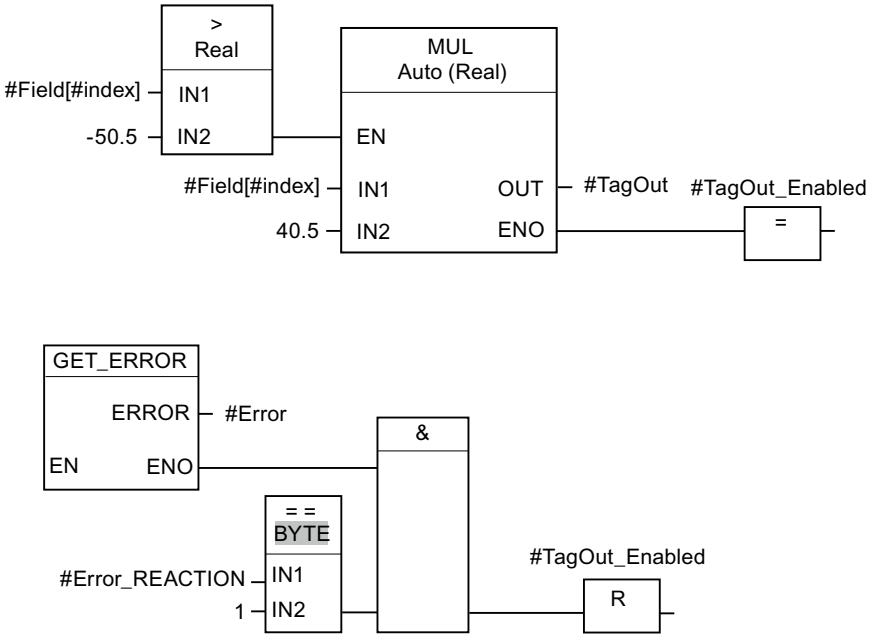
Structure component "ERROR_ID"

The following table shows the values that can be output at the structure component "ERROR_ID":

ID* (hexadecimal)	ID* (decimal)	Description
0	0	No error
2503	9475	Invalid pointer
2520	9504	Invalid STRING
2522	9506	Read error: Operand outside the valid range
2523	9507	Write error: Operand outside the valid range
2524	9508	Read error: Invalid operand
2525	9509	Write error: Invalid operand
2528	9512	Read error: Data alignment
2529	9513	Write error: Data alignment
252C	9516	Invalid pointer
2530	9520	Write error: Data block
2533	9523	Invalid reference used
2538	9528	Access error: DB does not exist
2539	9529	Access error: Wrong DB used
253A	9530	Global data block does not exist
253C	9532	Faulty information or the function does not exist
253D	9533	System function does not exist
253E	9534	Faulty information or the function block does not exist
253F	9535	System block does not exist
2550	9552	Access error: DB does not exist
2551	9553	Access error: Wrong DB used
2575	9589	Error in the program nesting depth
2576	9590	Error in the local data distribution
2577	9591	The block property "Parameter passing via registers" is not selected.
25A0	9632	Internal error in TP
25A1	9633	Tag is write-protected
25A2	9634	Invalid numerical value of tag
2942	10562	Read error: Input
2943	10563	Write error: Output
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".		

Example

The following example shows how the instruction works:



An error occurred accessing the #Field[#index] tag. The enable output ENO of the "Multiply" instruction and the #TagOut_Enabled operand have signal state "1" despite the read/access error and the multiplication is performed with the value "0.0". If this error scenario occurs, it is recommended to program the "Get error locally" instruction after the "Multiply" instruction to get the error. The error information supplied by the "Get error locally" instruction is evaluated with the comparison instruction "Equal". If the "#Error.REACTION" structure component has the value "1", this involves a read/access error and the #TagOut_Enabled output is reset.

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

GET_ERR_ID: Get error ID locally (S7-1200, S7-1500)

Description

The "Get error ID locally" instruction is used to query the occurrence of errors within a block. This is usually to access error. If the system reports during the block processing errors within the block execution since the last execution of the instruction, the error ID of the first error that occurred in the tag is stored at the ID output.

Only operands of the WORD data type can be specified at the ID output. If several errors occur in the block, the error ID of the next error to occur in the instruction is output only after the first error that occurred has been remedied.

Note

The ID output is only changed if error information is present. To set the output back to "0" after handling an error, you have the following options:

- Declare the tag in the "Temp" section of the block interface.
- Reset the tag to "0" before calling the instruction.
- Query the enable output ENO.

The enable output ENO of the instruction "Get error ID locally" instruction is set only if the enable input EN returns signal state "1" and error information is present. If one of these conditions is not fulfilled, the remaining program execution is not affected by the "Get error ID locally" instruction.

You can find an example of how you can implement the instruction in combination with other troubleshooting options under "See also".

Note

The "Get error ID locally" instruction enables local error handling within a block. If the "Get error ID locally" instruction is inserted in the program code of a block, any predefined system reactions are ignored if errors occur.

Options for error handling

You can find an overview of the options for error handling here:AUTOHOTSPOT

For a detailed example of local error handling that contains several error handling options, refer to: AUTOHOTSPOT

Parameters

The following table shows the parameters of the "Get error ID locally" instruction:

Parameters	Declaration	Data type	Memory area	Description
ID	Output	WORD	I, Q, M, D, L	Error ID

Parameter ID

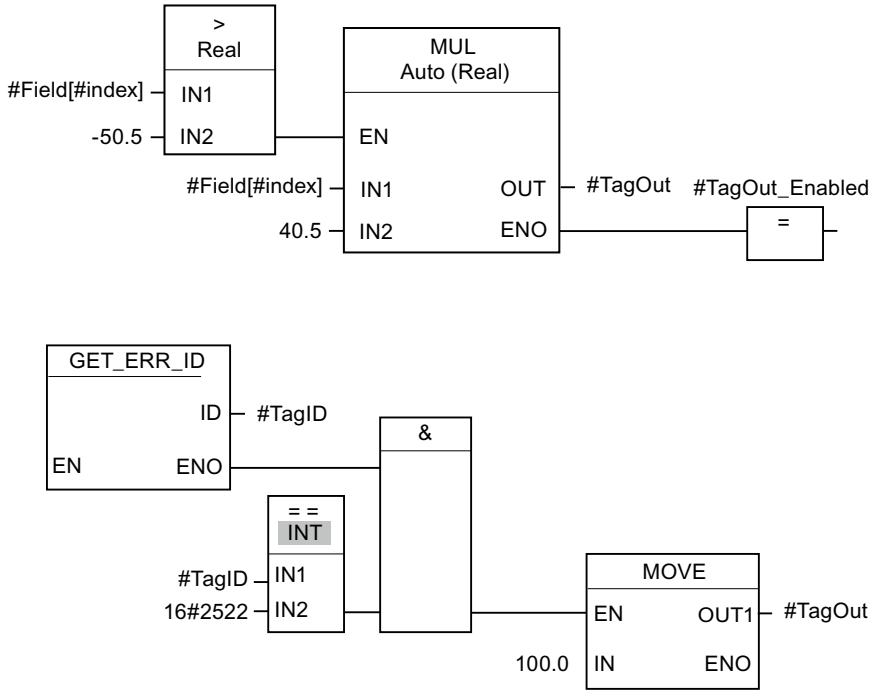
The following table shows the values that can be output at the ID parameter:

ID* (hexadecimal)	ID* (decimal)	Description
0	0	No error
2503	9475	Invalid pointer
2520	9504	Invalid STRING

ID* (hexadecimal)	ID* (decimal)	Description
2522	9506	Read error: Operand outside the valid range
2523	9507	Write error: Operand outside the valid range
2524	9508	Read error: Invalid operand
2525	9509	Write error: Invalid operand
2528	9512	Read error: Data alignment
2529	9513	Write error: Data alignment
252C	9516	Invalid pointer
2530	9520	Write error: Data block
2533	9523	Invalid reference used
2538	9528	Access error: DB does not exist
2539	9529	Access error: Wrong DB used
253A	9530	Global data block does not exist
253C	9532	Faulty information or the function does not exist
253D	9533	System function does not exist
253E	9534	Faulty information or the function block does not exist
253F	9535	System block does not exist
2550	9552	Access error: DB does not exist
2551	9553	Access error: Wrong DB used
2575	9589	Error in the program nesting depth
2576	9590	Error in the local data distribution
2577	9591	The block property "Parameter passing via registers" is not selected.
25A0	9632	Internal error in TP
25A1	9633	Tag is write-protected
25A2	9634	Invalid numerical value of tag
2942	10562	Read error: Input
2943	10563	Write error: Output
* The error codes can be displayed as integer or hexadecimal values in the program editor. You can find information on switching the display formats under "See also".		

Example

The following example shows how the instruction works:



An error occurred accessing the #Field[#index] tag. The enable output ENO of the "Multiply" instruction and the #TagOut_Enabled operand have signal state "1" despite the read/access error and the multiplication is performed with the value "0.0". If this error scenario occurs, it is recommended to program the "Get error ID locally" instruction after the "Multiply" instruction to get the error. The error information supplied by the "Get error ID locally" instruction is evaluated with the comparison instruction "Equal". If the #TagID operand returns the ID 2522, this involves a read/access error and the value "100.0" is written to the #TagOut output.

See also

Overview of the valid data types (Page 2239)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

INIT_RD: Initialize all retain data (S7-1500)

Description

The "Initialize all retain data" instruction is used to reset the retentive data of all data blocks, bit memories and SIMATIC timers and counters at the same time. The instruction can only be executed within a startup OB because the execution exceeds the program cycle duration.

Parameters

The following table shows the parameters of the "Initialize all retain data" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
REQ	Input	BOOL	I, Q, M, D, L, T, C or constant	If the input "REQ" has the signal state "1", all retentive data are reset.
RET_VAL	Output	INT	I, Q, M, D, L	Error information: If an error occurs during execution of the instruction, an error code is output in the RET_VAL parameter.

For additional information on valid data types, refer to "See also".

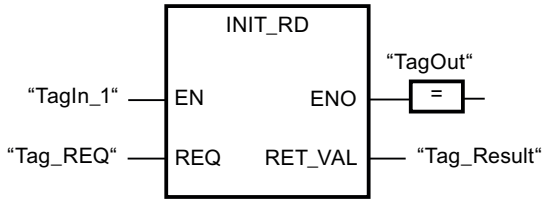
Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B5	The instruction cannot be executed because it was not programmed within a startup OB.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal values in the program editor. You will find information on changing display formats in "See also".	

Example

The following example shows how the instruction works:



If the operands "TagIn_1" and "Tag_REQ" have signal state "1", the instruction is executed. All retentive data of all data blocks, bit memories and SIMATIC timers and counters are reset. If the instruction is executed without errors, the ENO enable output has the signal state "1".

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3164)

Basic information on FBD (Page 4937)

WAIT: Configure time delay (S7-1500)

Description

The "Configure time delay" instruction pauses the program execution for a specific period of time. You indicate the period of time in microseconds on the WT parameter of the instruction.

You can configure time delays from -32768 up to +32767 microseconds (µs). The smallest possible delay time depends on the CPU and corresponds to the execution time of the instruction.

The execution of the instruction can be interrupted by higher priority events and does not return any error information.

Note

Negative delay time

If you specify a negative delay time at parameter WT, the enable output ENO, or the RLO and the BR bit, will return the signal state FALSE. A negative delay time does not affect the CPU. The following instructions linked to enable output ENO are not executed in LAD or FBD.

Parameters

The following table shows the parameters of the "Configure time delay" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
WT	Input	INT	I, Q, M, D, L, P or constant	Time delay in microseconds (µs)

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

RUNTIME: Measure program runtime (S7-1200, S7-1500)

Description

The "Measure program runtime" instruction is used to measure the runtime of the entire program, individual blocks or command sequences.

If you want to measure the runtime of your entire program, call the instruction "Measure program runtime" in OB1. Measurement of the runtime is started with the first call and the output RET_VAL returns the runtime of the program after the second call. The measured runtime includes all CPU processes that can occur during the program execution, for example, interruptions caused by higher-level events or communication. The instruction "Measure program runtime" reads an internal counter of the CPU and write the value to the IN-OUT parameter MEM. The instruction calculates the current program runtime according to the internal counter frequency and writes it to output RET_VAL.

If you want to measure the runtime of individual blocks or individual command sequences, you need three separate networks. Call the instruction "Measure program runtime" in an individual network within your program. You set the starting point of the runtime measurement with this first call of the instruction. Then you call the required program block or the command sequence in the next network. In another network, call the "Measure program runtime" instruction a second time and assign the same memory to the IN-OUT parameter MEM as you did during the first call of the instruction. The "Measure program runtime" instruction in the third network reads an internal CPU counter and calculates the current runtime of the program block or the command sequence according to the internal counter frequency and writes it to the output RET_VAL.

The "Measure program runtime" instruction uses an internal high-frequency counter to calculate the time. If the counter overruns, the instruction returns values <= 0.0. This can occur up to once per minute for S7-1200 CPUs with firmware version <V4.2. Ignore these runtime values.

Note

The runtime of a command sequence cannot be determined exactly, because the sequence of instructions within a command sequence is changed during optimized compilation of the program.

Parameters

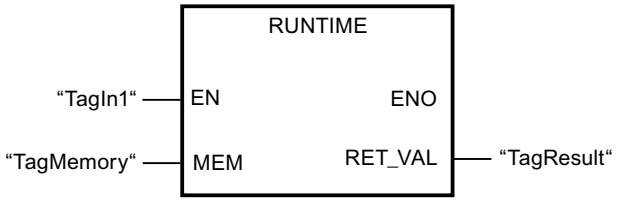
The following table lists the parameters of the "Measure program runtime" instruction:

Parameters	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
MEM	InOut	LREAL	I, Q, M, D, L	The content is for internal purposes only.
RET_VAL	Output	LREAL	I, Q, M, D, L	Returns the measured runtime in seconds

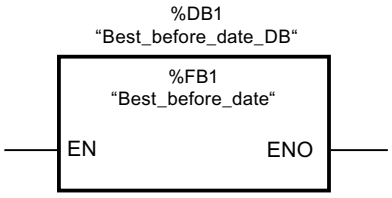
Example

The following example shows how the instruction works based on the runtime calculation of a program block:

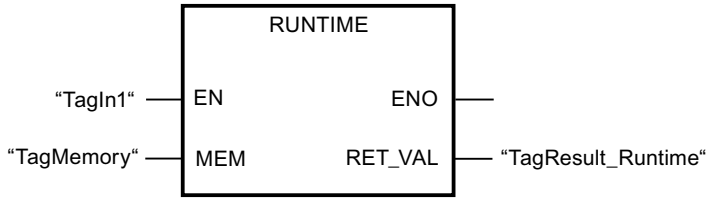
Network 1:



Network 2:



Network 3:



When the "TagIn1" operand in network 1 has the signal state "1", the instruction is executed. The starting point for the runtime measurement is set with the first call of the instruction and buffered as reference for the second call of the instruction in the "TagMemory" operand.

The "Best_before_date" program block FB1 is called in network 2.

When the FB1 program block has been executed and the "TagIn1" operand has the signal state "1", the instruction in network 3 is executed. The second call of the instruction calculates the runtime of the program block and writes the result to the output RET_VAL.

You can find a more detailed example of how you can measure the total cycle time of a program in the Siemens Industry Online Support under the following entry ID: 87668055 (https://support.industry.siemens.com/cs/ww/en/view/87668055)

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Word logic operations (S7-1200, S7-1500)

AND: AND logic operation (S7-1200, S7-1500)

Description

You can use the instruction "AND logic operation" to link the value at input IN1 to the value at input IN2 bit-by-bit by AND logic and query the result at the output OUT.

When the instruction is executed, bit 0 of the value at input IN1 is linked by AND logic to bit 0 of the value at input IN2. The result is stored in bit 0 of output OUT. The same logic operation is executed for all other bits of the specified values.

In its initial state the instruction box contains at least 2 inputs (IN1 and IN2). The number of inputs can be extended. The inserted inputs are numbered in ascending order in the box. During the execution of the instruction, the values of all available input parameters are linked by AND logic. The result is stored at output "OUT".

The result bit has the signal state "1" only when both of the bits in the logic operation also have signal state "1". If one of the two bits of the logic operation has signal state "0", the corresponding result bit is reset.

Parameters

The following table shows the parameters of the instruction "AND logic operation":

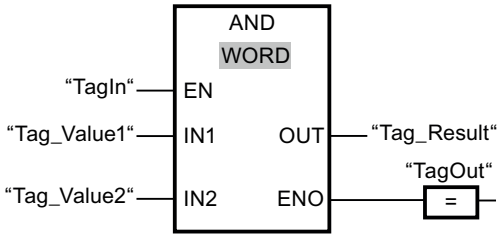
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First value for logic operation
IN2	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second value for logic operation
INn	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Optional input values
OUT	Output	Bit strings	I, Q, M, D, L, P	I, Q, M, D, L, P	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN1	Tag_Value1	0101 0101 0101 0101
IN2	Tag_Value2	0000 0000 0000 1111
OUT	Tag_Result	0000 0000 0000 0101

If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" is linked by AND to the value of the operand "Tag_Value2". The result is mapped bit-for-bit and sent to the operand "Tag_Result". The enable output ENO and the output "TagOut" are set to signal state "1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

OR: OR logic operation (S7-1200, S7-1500)

Description

You can use the instruction "OR logic operation" to link the value at input IN1 to the value at input IN2 bit-by-bit by OR logic and query the result at the output OUT.

When the instruction is executed, bit 0 of the value at input IN1 is linked by OR logic to bit 0 of the value at input IN2. The result is stored in bit 0 of output OUT. The same logic operation is executed for all bits of the specified tags.

In its initial state the instruction box contains at least 2 inputs (IN1 and IN2). The number of inputs can be extended in the instruction box. The inserted inputs are numbered in ascending order in the box. During the execution of the instruction, the values of all available input parameters are linked by OR logic. The result is stored at output "OUT".

The result bit has the signal state "1" when at least one of the two bits in the logic operation has signal state "1". If both of the bits of the logic operation have signal state "0", the corresponding result bit is reset.

Parameters

The following table shows the parameters of the instruction "OR logic operation":

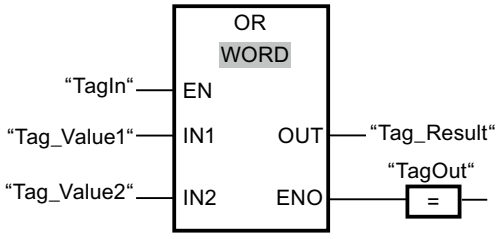
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First value for logic operation
IN2	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second value for logic operation
INn	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Optional input values
OUT	Output	Bit strings	I, Q, M, D, L, P	I, Q, M, D, L, P	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN1	Tag_Value1	0101 0101 0101 0101
IN2	Tag_Value2	0000 0000 0000 1111
OUT	Tag_Result	0101 0101 0101 1111

If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" is linked by OR to the value of the operand "Tag_Value2". The result is mapped bit-for-bit and sent to the operand "Tag_Result". The enable output ENO and the output "TagOut" are set to signal state "1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

XOR: EXCLUSIVE OR logic operation (S7-1200, S7-1500)

Description

You can use the instruction "EXCLUSIVE OR logic operation" to link the value at input IN1 to the value at input IN2 bit-by-bit by EXCLUSIVE OR logic and query the result at the output OUT.

When the instruction is executed, bit 0 of the value at input IN1 is linked by EXCLUSIVE OR logic to bit 0 of the value at input IN2. The result is stored in bit 0 of output OUT. The same logic operation is executed for all other bits of the specified value.

In its initial state the instruction box contains at least 2 inputs (IN1 and IN2). The number of inputs can be extended in the instruction box. The inserted inputs are numbered in ascending order in the box. During the execution of the instruction, the values of all available input parameters are linked by EXCLUSIVE OR logic. The result is stored at output "OUT".

The result bit has the signal state "1" when one of the two bits in the logic operation has signal state "1". If both of the bits of the logic operation have signal state "1" or "0", the corresponding result bit is reset.

Parameters

The following table shows the parameters of the instruction "EXCLUSIVE OR logic operation":

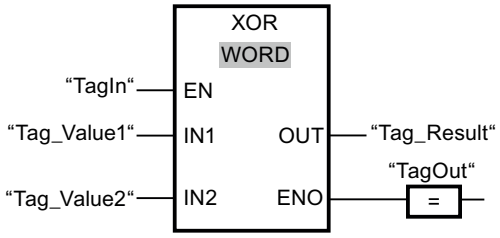
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN1	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First value for logic operation
IN2	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second value for logic operation
INn	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Optional input values
OUT	Output	Bit strings	I, Q, M, D, L, P	I, Q, M, D, L, P	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN1	Tag_Value1	0101 0101 0101 0101
IN2	Tag_Value2	0000 0000 0000 1111
OUT	Tag_Result	0101 0101 0101 1010

If the "TagIn" operand has signal state "1", the instruction is executed. The value of operand "Tag_Value1" is linked by EXCLUSIVE OR to the value of the operand "Tag_Value2". The result is mapped bit-for-bit and sent to the operand "Tag_Result". The enable output ENO and the output "TagOut" are set to signal state "1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

INVERT: Create ones complement (S7-1200, S7-1500)

Description

You can use the instruction "Create ones complement" to invert the signal status of the bits at input IN. When the instruction is processed, the value at input IN is linked to EXCLUSIVE OR by a hexadecimal mask (W#16#FFFF for 16-bit numbers or DW#16#FFFF FFFF for 32-bit number). This inverts the signal state of the individual bits that are then stored at output OUT.

Parameters

The following table shows the parameters of the instruction "Create ones complement":

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output

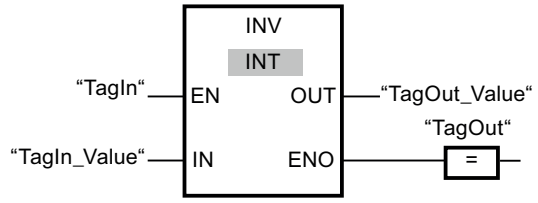
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
IN	Input	Bit strings, integers	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	Input value
OUT	Output	Bit strings, integers	I, Q, M, D, L, P	I, Q, M, D, L, P	Ones complement of the value at input IN

You can select the data type of the instruction from the "<???>" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value	
IN	TagIn_Value	W#16#000F	W#16#7E
OUT	TagOut_Value	W#16#FFF0	W#16#81

If operand "TagIn" has the signal state "1", the "Create ones complement" instruction is executed. The instruction inverts the signal state of the individual bits at input "TagIn_Value" and writes the result to output "TagOut_Value". The enable output ENO and the output "TagOut" are set to signal state "1".

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

DECO: Decode (S7-1200, S7-1500)

Description

You can use the "Decode" instruction to set a bit in the output value specified by the input value.

The "Decode" instruction reads the value at the IN input and sets the bit in the output value whose bit position corresponds to the read value. The other bits in the output value are filled with zeroes. If the value at input IN is greater than 31, a modulo 32 instruction is executed.

Parameters

The following table shows the parameters of the "Decode" instruction:

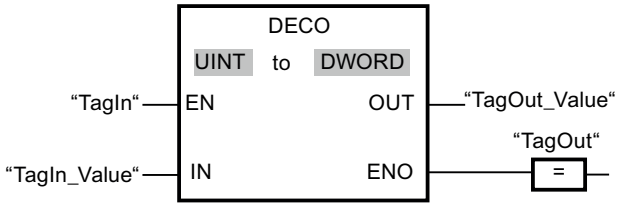
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	UINT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Position of the bit in the output value which is set.
OUT	Output	Bit strings	I, Q, M, D, L, P	I, Q, M, D, L, P	Output value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

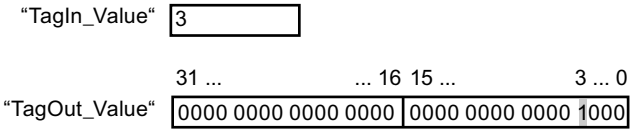
For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following figure shows how the instruction works using concrete operand values:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction reads bit number "3" from the value of the operand "TagIn_Value" and sets the third bit to the value of the operand "TagOut_Value".

If no errors occur during the execution of the instruction, the output ENO has the signal state "1" and the output "TagOut" is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

ENCO: Encode (S7-1200, S7-1500)

Description

The instruction "Encode" is used to read the bit number of the lowest bit in the input value and output it to the output OUT.

The instruction "Encode" selects the least significant bit of the value at the IN input and writes its bit number to the tag in the output OUT.

Parameters

The following table shows the parameters of the instruction "Encode":

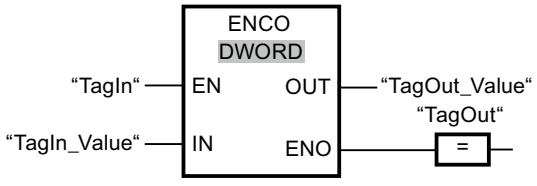
Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
EN	Input	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Bit strings	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Input value
OUT	Output	INT	I, Q, M, D, L, P	I, Q, M, D, L, P	Output value

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

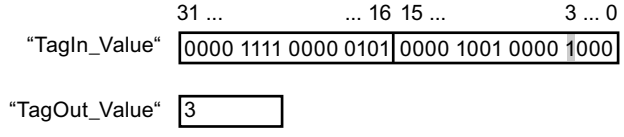
For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following figure shows how the instruction works using concrete operand values:



If the "TagIn" operand has signal state "1", the instruction is executed. The instruction selects bit position "3" as the least significant bit at input "TagIn_Value" and writes the value "3" to the tag at the output "TagOut_Value.

If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SEL: Select (S7-1200, S7-1500)

Description

Depending on the signal state at switch (input G), the "Select" instruction selects one of the inputs IN0 or IN1 and moves its content to the output OUT. When the input G has the signal state "0", the value at the input IN0 is moved. When the input G has the signal state "1", the value at the input IN1 is moved to the output OUT.

The instruction can only be executed if the enable input EN has the signal state "1" and the tags of all parameters are of the same data type.

Parameters

The following table shows the parameters of the "Select" instruction:

Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
G	Input	BOOL	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L, T, C or constant	Switch
IN0	Input	Bit strings, integers, floating-point numbers, timers, TOD, DATE, CHAR, WCHAR	Bit strings, integers, floating-point numbers, timers, TOD, LTOD, DATE, TDT, CHAR, WCHAR	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First input value

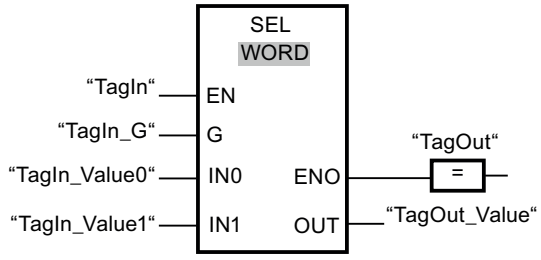
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
IN1	Input	Bit strings, integers, floating-point numbers, timers, TOD, DATE, CHAR, WCHAR	Bit strings, integers, floating-point numbers, timers, TOD, LTOD, DATE, TDT, CHAR, WCHAR	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second input value
OUT	Output	Bit strings, integers, floating-point numbers, timers, TOD, DATE, CHAR, WCHAR	Bit strings, integers, floating-point numbers, timers, TOD, LTOD, DATE, TDT, CHAR, WCHAR	I, Q, M, D, L, P	I, Q, M, D, L, P	Result

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value	
G	TagIn_G	0	1
IN0	TagIn_Value0	W#16#0000	W#16#4C
IN1	TagIn_Value1	W#16#FFFF	W#16#5E
OUT	TagOut_Value	W#16#0000	W#16#5E

If the "TagIn" operand has signal state "1", the instruction is executed. Based on the signal state at the input "TagIn_G", the value at input "TagIn_Value0" or "TagIn_Value1" is selected

and copied to output "TagOut_Value". If the instruction is executed without errors, the enable output ENO has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

MUX: Multiplex (S7-1200, S7-1500)

Description

You can use the instruction "Multiplex" to copy the content of a selected input to output OUT. In its initial state the instruction box contains at least 2 inputs (IN0 and IN1). The number of selectable inputs of the instruction box can be expanded. You can declare a maximum of 32 inputs.

The inputs are numbered automatically in the box. Numbering starts at IN0 and is incremented continuously with each new input. You use the K parameter to define the input whose content is to be copied to the OUT output. If the value of the parameter K is greater than the number of available inputs, the content of the parameter ELSE is copied to output OUT and enable output ENO is assigned the signal state "0".

The "Multiplex" instruction can only be executed if the tags have the same data type in all inputs and in the OUT output. The K parameter is an exception, since only integers can be specified for it.

The enable output ENO is reset if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The input at the K parameter is located outside the available inputs. This is the reaction regardless of whether the ELSE input will be used or not. The value at the OUT output remains unchanged.
- Errors occurred during execution of the instruction.

Parameters

The following table shows the parameters of the "Multiplex" instruction:

Parameters	Declaring	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output

Parameters	Declaring	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
K	Input	Integers	Integers	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Specifies the input whose content is to be copied. • If K = 0 => Parameter IN0 • If K = 1 => Parameter IN1, etc.
IN0	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	First input value
IN1	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Second input value
INn	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Optional input values

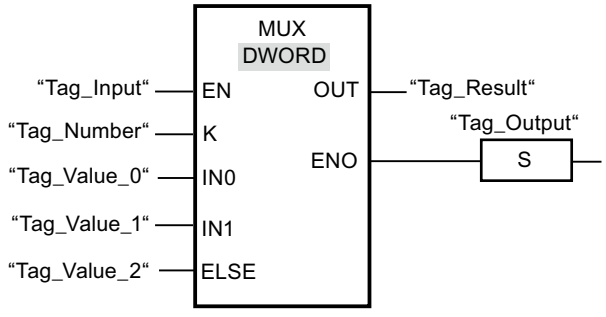
Parameters	Declaring	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
ELSE	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P or constant	I, Q, M, D, L, P or constant	Specifies the value to be copied when K > n
OUT	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	I, Q, M, D, L, P	Output to which the value is to be copied

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
K	Tag_Number	1
IN0	Tag_ValueI_0	DW#16#00000000
IN1	Tag_Value_1	DW#16#003E4A7D
ELSE	Tag_Value_2	DW#16#FFFF0000
OUT	Tag_Result	DW#16#003E4A7D

If the "Tag_Input" operand has signal state "1", the instruction is executed. Depending on the value of the operand "Tag_Number", the value at input "Tag_Value_1" is copied and assigned to the operand at output "Tag_Result". If no errors occur during the execution of the instruction, the outputs ENO and "Tag_Output" are set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

DEMUX: Demultiplex (S7-1200, S7-1500)

Description

You can use the instruction "Demultiplex" to copy the content of the input IN to a selected output. In its initial state the instruction box contains at least 2 outputs (OUT0 and OUT1). The number of selectable outputs can be extended in the instruction box. The outputs are numbered automatically in the box. Numbering starts at OUT0 and continues consecutively with each new output. You use the K parameter to define the output to which the content of the IN input is to be copied. The other outputs are not changed. If the value of the parameter K is greater than the number of available outputs, then the content of input IN is copied to the parameter ELSE and the enable output ENO is assigned the signal state "0".

The instruction "Demultiplex" can only be executed if the tags at the input IN and at all outputs are of the same data type. The K parameter is an exception, since only integers can be specified for it.

The enable output ENO is reset if one of the following conditions applies:

- Enable input EN has the signal state "0".
- The value of the K parameter is greater than the number of available outputs.
- Errors occurred during execution of the instruction.

Parameters

The following table shows the parameters of the instruction "Demultiplex":

Parameter	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output

Parameter	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
K	Input	Integers	Integers	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	Specifies the output to which the input value (IN) is copied. • If K = 0 => Parameter OUT0 • If K = 1 => Parameter OUT1, etc.
IN	Input	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P, or constant	I, Q, M, D, L, P, or constant	Input value
OUT0	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	I, Q, M, D, L, P	First output
OUT1	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	I, Q, M, D, L, P	Second output

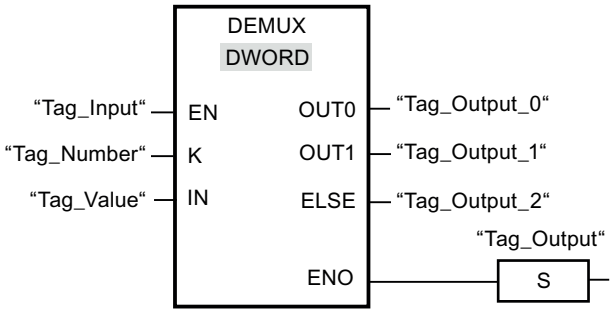
Parameter	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
OUTn	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	I, Q, M, D, L, P	Optional outputs
ELSE	Output	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, DATE	Binary numbers, integers, floating-point numbers, timers, CHAR, WCHAR, TOD, LTOD, DATE, LDT	I, Q, M, D, L, P	I, Q, M, D, L, P	Output to which the input value (IN) at K > n is copied.

You can select the data type of the instruction from the "<???>" drop-down list of the instruction box.

For additional information on available data types, refer to "See also".

Example

The following example shows how the instruction works:



The following tables show how the instruction works using specific operand values:

Table 11-16 Input values of the "Demultiplex" instruction before network execution

Parameter	Operand	Values	
K	Tag_Number	1	4
IN	Tag_Value	DW#16#FFFFFFFF	DW#16#003E4A7D

Table 11-17 Output values of the "Demultiplex" instruction after network execution

Parameter	Operand	Values	
OUT0	Tag_Output_0	Unchanged	Unchanged
OUT1	Tag_Output_1	DW#16#FFFFFFFF	Unchanged
ELSE	Tag_Output_2	Unchanged	DW#16#003E4A7D

If the "Tag_Input" input has the signal state"1", the "Demultiplex" instruction is executed. Depending on the value of the operand "Tag_Number", the value at input "IN" is copied to the corresponding output.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Shift and Rotate (S7-1200, S7-1500)

SHR: Shift right (S7-1200, S7-1500)

Description

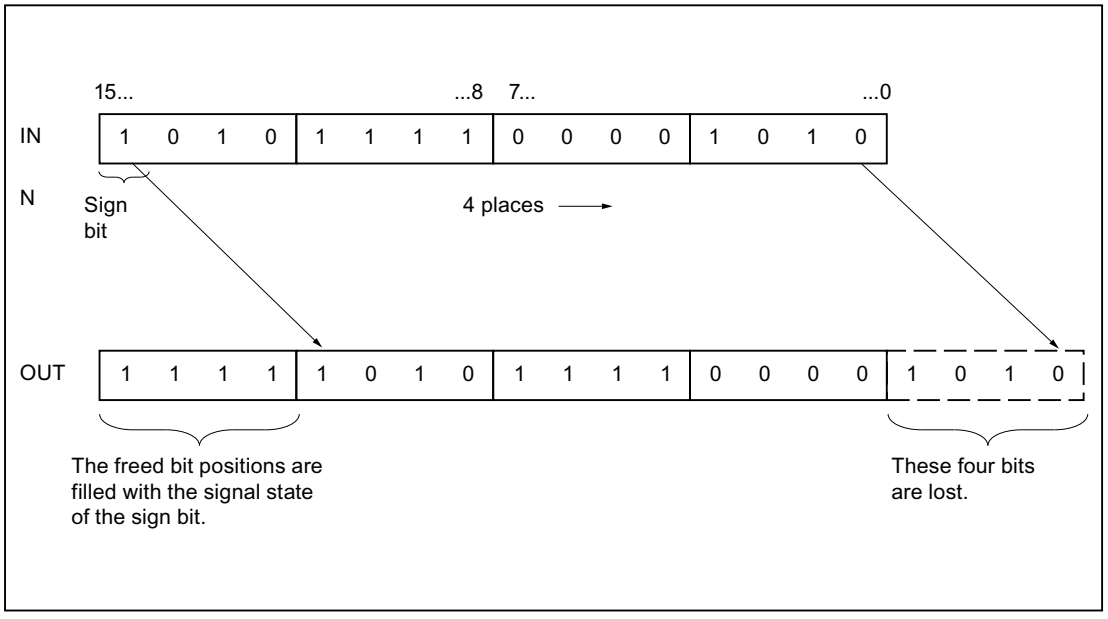
You can use the "Shift right" instruction to shift the content of the operand at the input IN bit-by-bit to the right and query the result at the OUT output. The input N is used to specify the number of bit positions by which the specified value should be moved.

If the value at the input N is "0", the value at input IN is copied unchanged to the operand at output OUT.

If the value at the input N is greater than the number of available bit positions, the operand value at input IN is shifted to the right by the available number of bit positions.

The freed bit positions in the left area of the operand are filled by zeroes when values without signs are shifted. If the specified value has a sign, the free bit positions are filled with the signal state of the sign bit.

The following figure show how the content of an operand of integer data type is shifted four bit positions to the right:



Parameters

The following table shows the parameters of the instruction "Shift right":

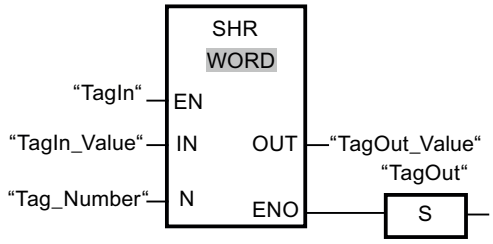
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Value to be shifted
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Number of bit positions by which the value is shifted
OUT	Output	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	I, Q, M, D, L	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	TagIn_Value	0011 1111 1010 1111
N	Tag_Number	3
OUT	TagOut_Value	0000 0111 1111 0101

If the "TagIn" operand has signal state "1", the instruction is executed. The content of the operand "TagIn_Value" is shifted three bit positions to the right. The result is output at the "TagOut_Value" output. If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SHL: Shift left (S7-1200, S7-1500)

Description

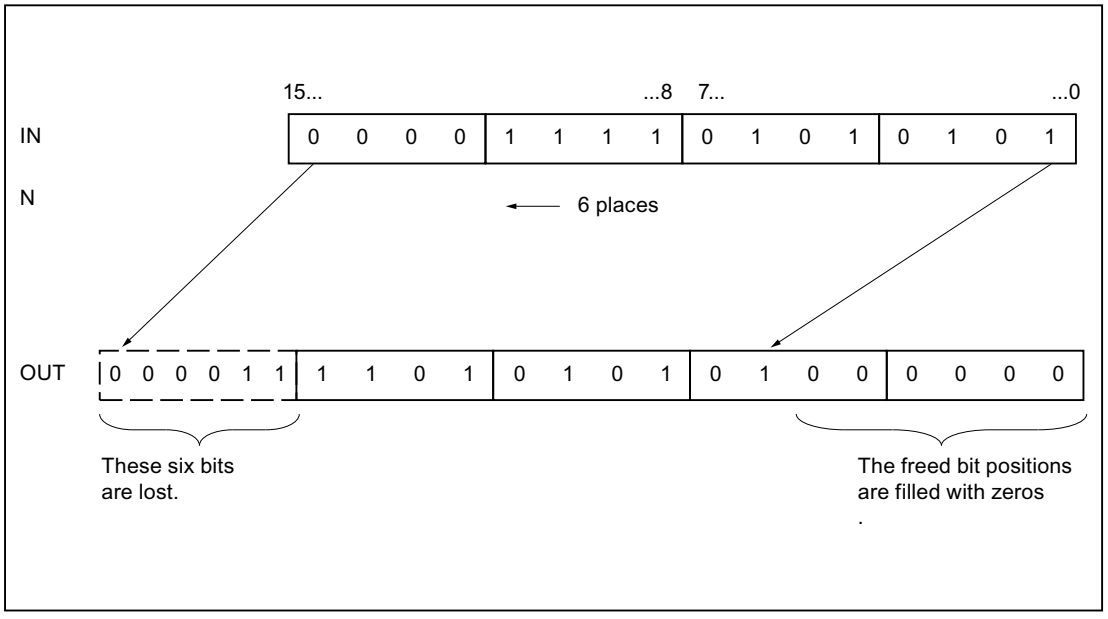
You can use the "Shift left" instruction to shift the content of the operand at the input IN bit-by-bit to the left and query the result at the OUT output. The input N is used to specify the number of bit positions by which the specified value should be moved.

If the value at the input N is "0", the value at input IN is copied unchanged to the operand at output OUT.

If the value at the input N is greater than the number of available bit positions, the operand value at input IN is shifted to the left by the available number of bit positions.

The bit positions in the right part of the operand freed by shifting are filled with zeros.

The following figure shows how the content of an operand of the WORD data type is shifted by six bit positions to the left:



Parameters

The following table shows the parameters of the instruction "Shift left":

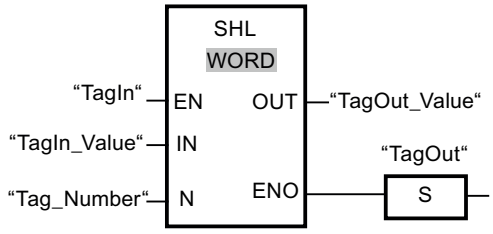
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Value to be shifted.
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Number of bit positions by which the value is shifted.
OUT	Output	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	I, Q, M, D, L	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	TagIn_Value	0011 1111 1010 1111
N	Tag_Number	4
OUT	TagOut_Value	1111 1010 1111 0000

If the "TagIn" operand has signal state "1", the instruction is executed. The content of the operand "TagIn_Value" is shifted four bit positions to the left. The result is output at the "TagOut_Value" output. If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

ROR: Rotate right (S7-1200, S7-1500)

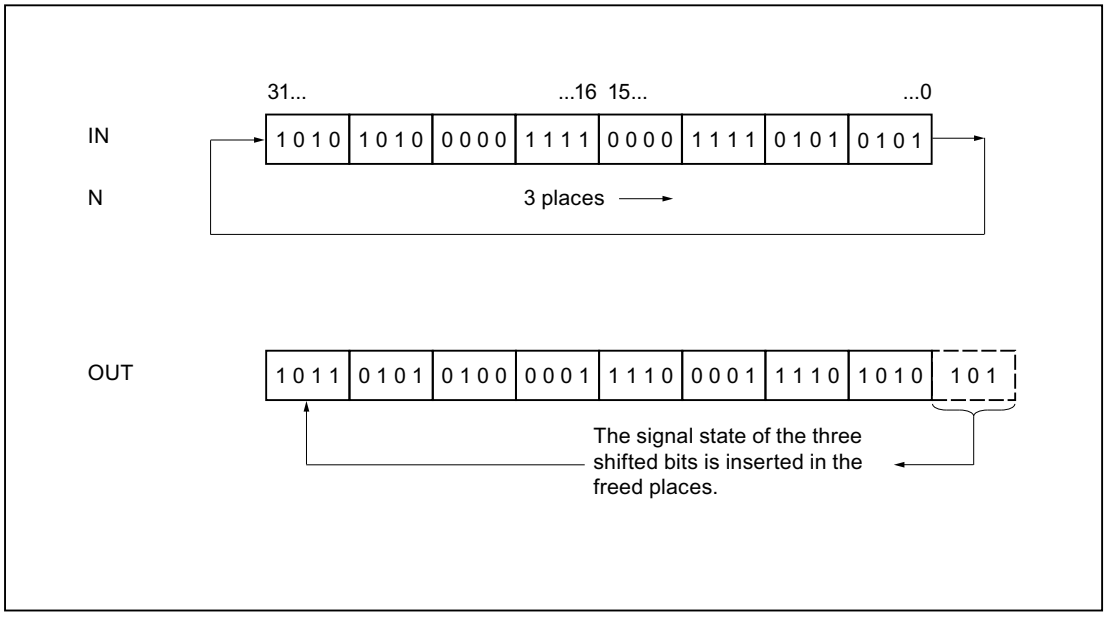
Description

You can use the "Rotate right" instruction to rotate the content of the operand at the input IN bit-by-bit to the right and query the result at the OUT output. The input N is used to specify the number of bit positions by which the specified value should be rotated. The bit positions freed by rotating on the left-hand side are filled true-to-position with the bit positions that are pushed out from the left-hand side.

If the value at the input N is "0", the value at input IN is copied unchanged to the operand at output OUT.

If the value of the N parameter is greater than the number of available bit positions, the operand value at the IN input is nevertheless rotated by the specified number of bit positions.

The following figure shows how the content of an operand of the DWORD data type is rotated three bit positions to the right:



Parameters

The following table shows the parameters of the instruction "Rotate right":

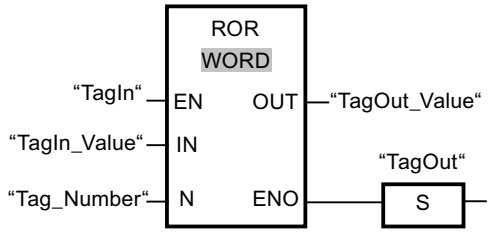
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Value to be rotated
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Number of bit positions by which the value is rotated
OUT	Output	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	I, Q, M, D, L	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	TagIn_Value	0000 1111 1001 0101
N	Tag_Number	5
OUT	TagOut_Value	1010 1000 0111 1100

If the "TagIn" operand has signal state "1", the instruction is executed. The content of the operand "TagIn_Value" is rotated five bit positions to the right. The result is output at the "TagOut_Value" output. If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

ROL: Rotate left (S7-1200, S7-1500)

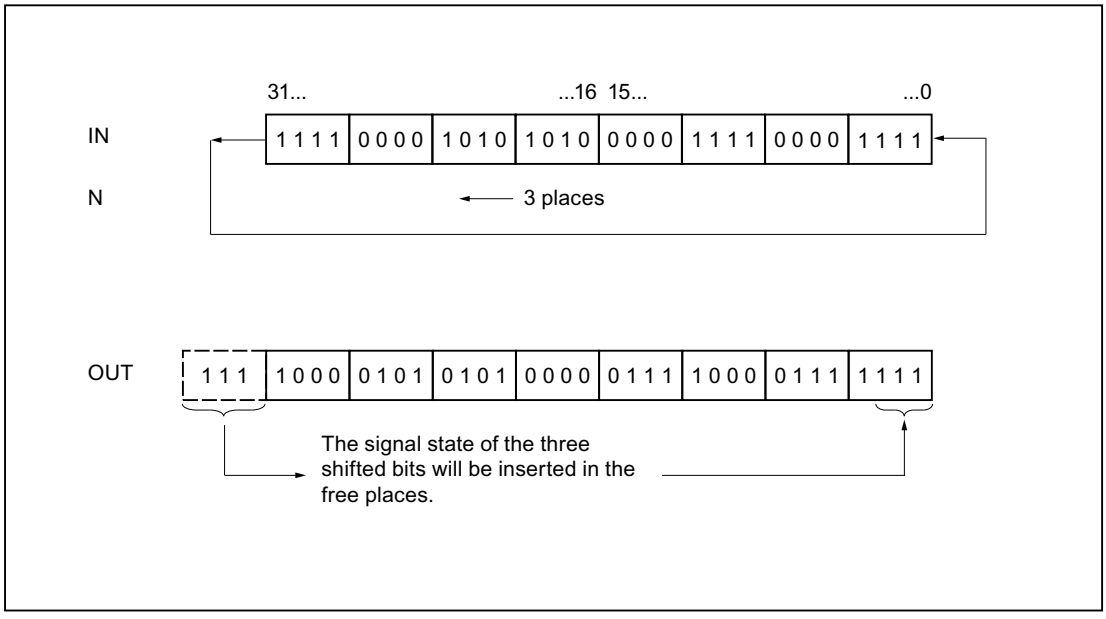
Description

You can use the "Rotate left" instruction to rotate the content of the operand at the input IN bit-by-bit to the left and query the result at the OUT output. The input N is used to specify the number of bit positions by which the specified value should be rotated. The bit positions freed by rotating on the right-hand side are filled true-to-position with the bit positions that are pushed out from the left-hand side.

If the value at the input N is "0", the value at input IN is copied to the operand at output OUT.

If the value of the N parameter is greater than the number of available bit positions, the operand value at the IN input is nevertheless rotated by the specified number of bit positions.

The following figure shows how the content of an operand of the DWORD data type is rotated three bit positions to the left:



Parameters

The following table shows the parameters of the instruction "Rotate left":

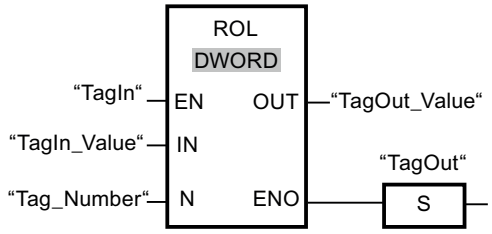
Parameters	Declaration	Data type		Memory area		Description
		S7-1200	S7-1500	S7-1200	S7-1500	
EN	Input	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	BOOL	I, Q, M, D, L	I, Q, M, D, L	Enable output
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Value to be rotated.
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Number of bit positions by which the value is rotated.
OUT	Output	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	I, Q, M, D, L	Result of the instruction

You can select the data type of the instruction from the "???" drop-down list of the instruction box.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:



The following table shows how the instruction works using concrete operand values:

Parameters	Operand	Value
IN	TagIn_Value	1010 1000 1111 0110
N	Tag_Number	5
OUT	TagOut_Value	0001 1110 1101 0101

If the "TagIn" has the signal state "1", the instruction is executed. The content of the operand "TagIn_Value" is rotated five bit positions to the left. The result is output at the "TagOut_Value" output. If the instruction is executed without errors, the ENO enable output has the signal state "1" and the "TagOut" output is set.

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

Legacy (S7-1500)

DRUM: Implement sequencer (S7-1500)

Description

You can use the "Implement sequencer" instruction to assign the programmed values of the OUT_VAL parameter of the corresponding step to the programmed output bits (OUT1 to OUT16) and the output word (OUT_WORD). The specific step must thereby satisfy the conditions of the programmed enable mask on the S_MASK parameter while the instruction remains at this step. The instruction advances to the next step if the event for the step is true and the programmed time for the current step elapses, or if the value at the JOG parameter changes from "0" to "1". The instruction is reset when the signal state of the RESET parameter changes to "1". The current step is hereby equated to the preset step (DSP).

The amount of time spent on a step is determined by the product of the preset timebase (DTBP) and the preset counter value (S_PRESET) for each step. At the start of a new step, this calculated value is loaded into the DCC parameter, which contains the time remaining for the current step. If, for example the value at the DTBP parameter is "2" and the preset value for the first step is "100" (100 ms), the DCC parameter has the value "200" (200 ms).

A step can be programmed with a timer value, an event, or both. Steps that have an event bit and the timer value "0" advance to the next step as soon as the signal state of the event bit is "1". Steps that are programmed only with a timer value start the time immediately. Steps that are programmed with an event bit and a timer value greater than "0" start the time when the signal state of the event bit is "1". The event bits are initialized with a signal state of "1".

When the sequencer is on the last programmed step (LST_STEP) and the time for this step has expired, the signal state on the Q parameter is set to "1"; otherwise it is set to "0". When the Q parameter is set, the instruction remains on the step until it is reset.

In the configurable mask (S_MASK) you can select the individual bits in the output word (OUT_WORD) and set or reset the output bits (OUT1 to OUT16) by means of the output values (OUT_VAL). When a bit of the configurable mask has a signal state of "1", the OUT_VAL value sets/resets the corresponding bit. If the signal state of a bit of the configurable mask is "0", the corresponding bit is left unchanged. All the bits of the configurable mask for all 16 steps are initialized with a signal state of "1".

The output bit at the OUT1 parameter corresponds to the least significant bit of the output word (OUT_WORD). The output bit at the OUT16 parameter corresponds to the most significant bit of the output word (OUT_WORD).

Parameters

The following table shows the parameters of the "Implement sequencer" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
RESET	Input	BOOL	I, Q, M, D, L or constant	The signal state "1" indicates a reset condition.
JOG	Input	BOOL	I, Q, M, D, L or constant	When the signal state changes from "0" to "1", the instruction advances to the next step.
DRUM_EN	Input	BOOL	I, Q, M, D, L or constant	The signal state "1" allows the sequencer to advance based on the event and time criteria.
LST_STEP	Input	BYTE	I, Q, M, D, L or constant	Maximum step number (for example: LST_STEP = 16#08; a maximum of 8 steps is possible.)
EVENT(i), 1 ≤ i ≤ 16	Input	BOOL	I, Q, M, D, L or constant	Event bit (i); Initial signal state is "1".
OUT(j), 1 ≤ j ≤ 16	Output	BOOL	I, Q, M, D, L	Output bit (j)
Q	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that the time for the last step has elapsed.
OUT_WORD	Output	WORD	I, Q, M, D, L, P	Word address to which the sequencer writes the output values.

Parameter	Declaration	Data type	Memory area	Description
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
JOG_HIS	Static	BOOL	I, Q, M, D, L or constant	JOG parameter history bit
EOD	Static	BOOL	I, Q, M, D, L or constant	The signal state "1" indicates that the time for the last step has elapsed.
DSP	Static	BYTE	I, Q, M, D, L, P or constant	Preset step of the sequencer
DSC	Static	BYTE	I, Q, M, D, L, P or constant	Current step of the sequencer
DCC	Static	DWORD	I, Q, M, D, L, P or constant	Current numerical value of the sequencer
DTBP	Static	WORD	I, Q, M, D, L, P or constant	Preset timebase of the sequencer
PrevTime	Static	TIME	I, Q, M, D, L or constant	Previous system time
S_PRESET	Static	ARRAY[1..16] of WORD	I, Q, M, D, L or constant	Preset counter value for each step [1 to 16] where one clock pulse = 1 ms.
OUT_VAL	Static	ARRAY[1..16, 0..15] of BOOL	I, Q, M, D, L or constant	Output values for each step [1 to 16, 0 to 15].
S_MASK	Static	ARRAY[1..16, 0..15] of BOOL	I, Q, M, D, L or constant	Configurable mask for each step [1 to 16, 0 to 15]. Initial signal states are "1".

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

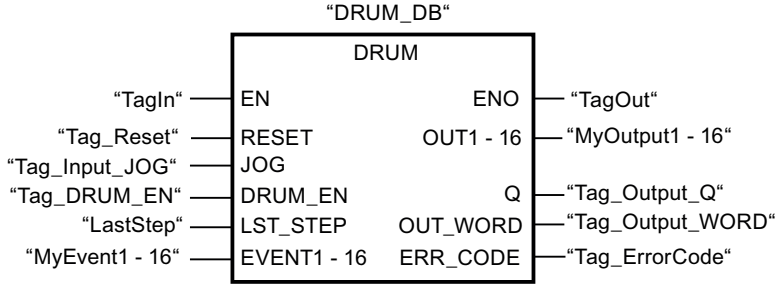
ERR_CODE*	Explanation
W#16#0000	No error
W#16#000B	The value at the LST_STEP parameter is less than 1 or greater than 16.
W#16#000C	The value at the DSC parameter is less than 1 or greater than the value of the LST_STEP parameter.
W#16#000D	The value at the DSP parameter is less than 1 or greater than the value of the LST_STEP parameter.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

In the following example the instruction advances from step 1 to step 2. The output bits (OUT1 to OUT16) and the output word (OUT_WORD) are set according to the mask configured for step 2 and the values of the OUT_VAL parameter.

Note

You can initialize static parameters in the data block.



The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for initializing the input parameters:

Parameters	Operand	Address	Value
RESET	Tag_Reset	M0.0	FALSE
JOG	Tag_Input_JOG	M0.1	FALSE
DRUM_EN	Tag_Input_DrumEN	M0.2	TRUE
LST_STEP	Tag_Number_LastStep	MB1	B#16#08
EVENT2	MyTag_Event_2	M20.0	FALSE
EVENT4	MyTag_Event_4	M20.1	FALSE
EVENT6	MyTag_Event_6	M20.2	FALSE
EVENT8	MyTag_Event_8	M20.3	FALSE
EVENT10	MyTag_Event_10	M20.4	FALSE
EVENT12	MyTag_Event_12	M20.5	FALSE
EVENT14	MyTag_Event_14	M20.6	FALSE
EVENT16	MyTag_Event_16	M20.7	FALSE

The following values are saved in the "DRUM_DB" instance data block of the instruction:

Parameter	Address	Value
JOG_HIS	DBX12.0	FALSE
EOD	DBX12.1	FALSE
DSP	DBB13	W#16#0001
DSC	DBB14	W#16#0001
DCC	DBD16	DW#16#0000000A

Parameter	Address	Value
DTBP	DBW20	W#16#0001
S_PRESET[1]	DBW26	W#16#0064
S_PRESET[2]	DBW28	W#16#00C8
OUT_VAL[1,0]	DBX58.0	TRUE
OUT_VAL[1,1]	DBX58.1	TRUE
OUT_VAL[1,2]	DBX58.2	TRUE
OUT_VAL[1,3]	DBX58.3	TRUE
OUT_VAL[1,4]	DBX58.4	TRUE
OUT_VAL[1,5]	DBX58.5	TRUE
OUT_VAL[1,6]	DBX58.6	TRUE
OUT_VAL[1,7]	DBX58.7	TRUE
OUT_VAL[1,8]	DBX59.0	TRUE
OUT_VAL[1,9]	DBX59.1	TRUE
OUT_VAL[1,10]	DBX59.2	TRUE
OUT_VAL[1,11]	DBX59.3	TRUE
OUT_VAL[1,12]	DBX59.4	TRUE
OUT_VAL[1,13]	DBX59.5	TRUE
OUT_VAL[1,14]	DBX59.6	TRUE
OUT_VAL[1,15]	DBX59.7	TRUE
OUT_VAL[2,0]	DBX60.0	FALSE
OUT_VAL[2,1]	DBX60.1	FALSE
OUT_VAL[2,2]	DBX60.2	FALSE
OUT_VAL[2,3]	DBX60.3	FALSE
OUT_VAL[2,4]	DBX60.4	FALSE
OUT_VAL[2,5]	DBX60.5	FALSE
OUT_VAL[2,6]	DBX60.6	FALSE
OUT_VAL[2,7]	DBX60.7	FALSE
OUT_VAL[2,8]	DBX61.0	FALSE
OUT_VAL[2,9]	DBX61.1	FALSE
OUT_VAL[2,10]	DBX61.2	FALSE
OUT_VAL[2,11]	DBX61.3	FALSE
OUT_VAL[2,12]	DBX61.4	FALSE
OUT_VAL[2,13]	DBX61.5	FALSE
OUT_VAL[2,14]	DBX61.6	FALSE
OUT_VAL[2,15]	DBX61.7	FALSE
S_MASK[2,0]	DBX92.0	FALSE
S_MASK[2,1]	DBX92.1	TRUE
S_MASK[2,2]	DBX92.2	TRUE
S_MASK[2,3]	DBX92.3	TRUE
S_MASK[2,4]	DBX92.4	TRUE
S_MASK[2,5]	DBX92.5	FALSE
S_MASK[2,6]	DBX92.6	TRUE
S_MASK[2,7]	DBX92.7	TRUE

Parameter	Address	Value
S_MASK[2,8]	DBX93.0	FALSE
S_MASK[2,9]	DBX93.1	FALSE
S_MASK[2,10]	DBX93.2	TRUE
S_MASK[2,11]	DBX93.3	TRUE
S_MASK[2,12]	DBX93.4	TRUE
S_MASK[2,13]	DBX93.5	TRUE
S_MASK[2,14]	DBX93.6	FALSE
S_MASK[2,15]	DBX93.7	TRUE

The output parameters are set to the following values before the execution of the instruction:

Parameter	Operand	Address	Value
Q	Tag_Output_Q	M6.0	FALSE
OUTWORD	Tag_OutputWord	MW8	W#16#FFFF
OUT1	MyTag_Output_1	M4.0	TRUE
OUT2	MyTag_Output_2	M4.1	TRUE
OUT3	MyTag_Output_3	M4.2	TRUE
OUT4	MyTag_Output_4	M4.3	TRUE
OUT5	MyTag_Output_5	M4.4	TRUE
OUT6	MyTag_Output_6	M4.5	TRUE
OUT7	MyTag_Output_7	M4.6	TRUE
OUT8	MyTag_Output_8	M4.7	TRUE
OUT9	MyTag_Output_9	M5.0	TRUE
OUT10	MyTag_Output_10	M5.1	TRUE
OUT11	MyTag_Output_11	M5.2	TRUE
OUT12	MyTag_Output_12	M5.3	TRUE
OUT13	MyTag_Output_13	M5.4	TRUE
OUT14	MyTag_Output_14	M5.5	TRUE
OUT15	MyTag_Output_15	M5.6	TRUE
OUT16	MyTag_Output_16	M5.7	TRUE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameter	Operand	Address	Value
OUT1	MyTag_Output_1	M4.0	TRUE
OUT2	MyTag_Output_2	M4.1	FALSE
OUT3	MyTag_Output_3	M4.2	FALSE
OUT4	MyTag_Output_4	M4.3	FALSE
OUT5	MyTag_Output_5	M4.4	FALSE
OUT6	MyTag_Output_6	M4.5	TRUE
OUT7	MyTag_Output_7	M4.6	FALSE

Parameter	Operand	Address	Value
OUT8	MyTag_Output_8	M4.7	FALSE
OUT9	MyTag_Output_9	M5.0	TRUE
OUT10	MyTag_Output_10	M5.1	TRUE
OUT11	MyTag_Output_11	M5.2	FALSE
OUT12	MyTag_Output_12	M5.3	FALSE
OUT13	MyTag_Output_13	M5.4	FALSE
OUT14	MyTag_Output_14	M5.5	FALSE
OUT15	MyTag_Output_15	M5.6	TRUE
OUT16	MyTag_Output_16	M5.7	FALSE
Q	Tag_Output_Q	M6.0	FALSE
OUTWORD	Tag_OutputWord	MW8	W#16#4321
ERR_CODE	Tag_ErrorCode	MW10	W#16#0000

The following values are changed in the instance data block "DRUM_DB" of the instruction after the execution of the instruction:

Parameter	Address	Value
JOG_HIS	DBX12.0	FALSE
EOD	DBX12.1	FALSE
DSC	DBB14	W#16#0002
DCC	DBD16	DW#16#000000C8

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

DCAT: Discrete control-timer alarm (S7-1500)

Description

You can use the "Discrete control-timer alarm" to count the time from the point at which the CMD parameter issues the command to open or close. The time is accumulated until the preset time (PT) is exceeded or the information is received that the device was opened or closed (O_FB or C_FB) within the specified time. If the preset time is exceeded before the information on the opening or closing of the device is received, the corresponding alarm is activated. If the signal state of the command input changes before the preset time, the time is restarted.

The "Discrete control-timer alarm" instruction has the following reactions to the input conditions:

- When the signal state of the CMD parameter changes from "0" to "1", the signal states of the parameters Q, CMD_HIS, ET (only if ET is < PT) OA and CA are influenced as follows:
  - The Q and CMD_HIS parameters are set to "1".
  - The ET, OA, and CA parameters are reset to "0".
- When the signal state of the parameter CMD changes from "1" to "0", the parameters Q, ET (only if ET < PT), OA, CA and CMD_HIS are reset to "0".
- When the signal state of the CMD and CMD_HIS parameters is "1" and the parameter O_FB is set to "0", the time difference (ms) since the last execution of the instruction is added to the value at the parameter ET. If the value of the ET parameter exceeds the value of the PT parameter, the signal state at the OA parameter is set to "1". If the value of the ET parameter does not exceed the value of the PT parameter, the signal state of the OA parameter is reset to "0". The value at the parameter CMD_HIS is reset to the value of the parameter CMD.
- If the signal state of the CMD, CMD_HIS and O_FB parameters are set to "1" and the parameter C_FB has the value "0", the signal state of the parameter OA is set to "0". The value of the ET parameter is set to the value of the PT parameter. If the signal state of the O_FB parameter changes to "0", the alarm is set the next time the instruction is executed. The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the CMD, CMD_HIS and C_FB parameters return signal state "0", the time difference (ms) since the last execution of the instruction is added to the value of the ET parameter. If the value of the ET parameter exceeds the value of the PT parameter, the signal state of the parameter CA is reset to "1". If the value at the parameter PT is not exceeded, the parameter CA has the signal state "0". The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the parameters CMD, CMD_HIS and O_FB have the signal state "0" and the parameter C_FB is set to "1", the parameter CA is set to "0". The value of the ET parameter is set to the value of the PT parameter. If the signal state of the C_FB parameter changes to "0", the alarm is set the next time the instruction is executed. The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the parameters O_FB and C_FB simultaneously have the signal state "1", the signal states of both alarm outputs are set to "1".

The "Discrete control-timer alarm" instruction has no error information.

Parameters

The following table shows the parameters of the "Discrete control-timer alarm" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output

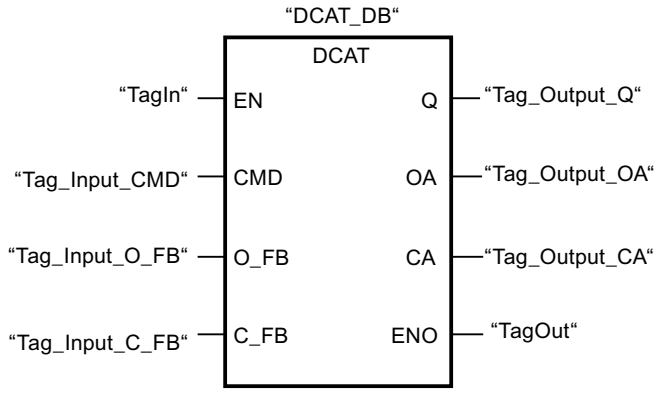
Parameter	Declaration	Data type	Memory area	Description
CMD	Input	BOOL	I, Q, M, D, L or constant	A signal state of "0" indicates a "close" command. A signal state of "1" indicates an "open" command.
O_FB	Input	BOOL	I, Q, M, D, L or constant	Feedback input when opening
C_FB	Input	BOOL	I, Q, M, D, L or constant	Feedback input when closing
Q	Output	BOOL	I, Q, M, D, L	Shows the status of the parameter CMD
OA	Output	BOOL	I, Q, M, D, L	Alarm output when opening
CA	Output	BOOL	I, Q, M, D, L	Alarm output when closing
ET	Static	DINT	D, L or constant	Currently elapsed time, where one count = 1 ms.
PT	Static	DINT	D, L or constant	Preset timer value, where one clock pulse = 1 ms.
PREV_TIME	Static	DWORD	D, L or constant	Previous system time
CMD_HIS	Static	BOOL	D, L or constant	CMD history bit

Example

In the following example the parameter CMD changes from "0" to "1". After the execution of the instruction the parameter Q is set to "1" and the two alarm outputs OA and CA have the signal state "0". The CMD_HIS parameter of the instance data block is set to the signal state "1" and the ET parameter is reset to "0".

Note

You can initialize static parameters in the data block.



The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for the input and output parameters:

Parameter	Operand	Value
CMD	Tag_Input_CMD	TRUE
O_FB	Tag_Input_O_FB	FALSE
C_FB	Tag_Input_C_FB	FALSE
Q	Tag_Output_Q	FALSE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE

The following values are saved in the instance data block "DCAT_DB" of the instruction:

Parameter	Address	Value
ET	DBD4	L#12
PT	DBD8	L#222
CMD_HIS	DBX16.0	FALSE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameter	Operand	Value
Q	Tag_Output_Q	TRUE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE

The following values are saved in the instance data block "DCAT_DB" of the instruction:

Parameter	Address	Value
ET	DBD4	L#0
CMD_HIS	DBX16.0	TRUE

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

MCAT: Motor control-timer alarm (S7-1500)

Description

The "Motor control-timer alarm" instruction is used to accumulate the time from the point at which one of the command inputs (opening or closing) is switched on. The time is accumulated until the preset time is exceeded or the relevant feedback input indicates that the device has executed the requested operation within the specified time. If the preset time is exceeded before the feedback is received, the corresponding alarm is triggered.

The "Motor control-timer alarm" instruction returns no error information.

Parameters

The following table shows the parameters of the "Motor control-timer alarm" instruction:

Parameters	Declaration	Data type	Memory space	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
O_CMD	Input	BOOL	I, Q, M, D, L or constant	"Open" command input
C_CMD	Input	BOOL	I, Q, M, D, L or constant	"Close" command input
S_CMD	Input	BOOL	I, Q, M, D, L or constant	"Stop" command input
O_FB	Input	BOOL	I, Q, M, D, L or constant	Feedback input when opening
C_FB	Input	BOOL	I, Q, M, D, L or constant	Feedback input when closing
OO	Output	BOOL	I, Q, M, D, L	"Open" output
CO	Output	BOOL	I, Q, M, D, L	"Close" output
OA	Output	BOOL	I, Q, M, D, L	Alarm output when opening
CA	Output	BOOL	I, Q, M, D, L	Alarm output when closing
Q	Output	BOOL	I, Q, M, D, L	A signal state of "0" indicates an error condition.
ET	Static	DINT	D, L or constant	Currently elapsed time, where one count = 1 ms
PT	Static	DINT	D, L or constant	Preset timer value, where one clock pulse = 1 ms
PREV_TIME	Static	DWORD	D, L or constant	Previous system time
O_HIS	Static	BOOL	D, L or constant	"Open" history bit
C_HIS	Static	BOOL	D, L or constant	"Close" history bit

Execution of the "Motor control-timer alarm" instruction

The following table shows the reactions of the "Motor control-timer alarm" instruction to the various input conditions:

Input parameters								Output parameters								
ET	O_HIS	C_HIS	O_CMD	C_CMD	S_CMD	O_FB	C_FB	OO	CO	OA	CA	ET	O_HIS	C_HIS	Q	Status
X	1	1	X	X	X	X	X	0	0	1	1	PT	0	0	0	Alarm
X	X	X	X	X	X	1	1	0	0	1	1	PT	0	0	0	Alarm
X	X	X	X	X	1	X	X	0	0	0	0	X	0	0	1	Stop
X	X	X	1	1	X	X	X	0	0	0	0	X	0	0	1	Stop
X	0	X	1	0	0	X	X	1	0	0	0	0	1	0	1	Start opening
<PT	1	0	X	0	0	0	X	1	0	0	0	INC	1	0	1	Open
X	1	0	X	0	0	1	0	0	0	0	0	PT	1	0	1	Opened
>=PT	1	0	X	0	0	0	X	0	0	1	0	PT	1	0	0	Opening alarm
X	X	0	0	1	0	X	X	0	1	0	0	0	0	1	1	Start closing
<PT	0	1	0	X	0	X	0	0	1	0	0	INC	0	1	1	Close
X	0	1	0	X	0	0	1	0	0	0	0	PT	0	1	1	Closed
>=PT	0	1	0	X	0	X	0	0	0	0	1	PT	0	1	0	Closing alarm
X	0	0	0	0	0	X	X	0	0	0	0	X	0	0	1	Stopped

Legend:	
INC	Add the time difference (ms) since the last processing of the FB to ET
PT	PT is set to the same value as ET
X	Cannot be used
<PT	ET < PT
>=PT	ET >= PT

If the input parameters O_HIS and C_HIS both have the signal state "1", they are immediately set to signal state "0". In this case, the last row in the table (X) mentioned above is valid. Because it is therefore no longer possible to check whether the input parameters O_HIS and C_HIS have the signal state "1", the output parameters are set as follows in this case:

OO = FALSE

CO = FALSE

OA = FALSE

CA = FALSE

ET = PT

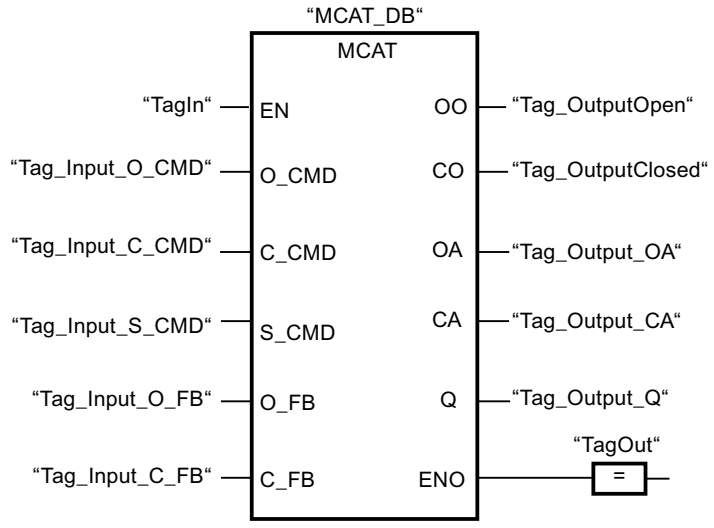
Q = TRUE

Example

The following example shows how the instruction works:

Note

You can initialize static parameters in the data block.



The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for the input and output parameters:

Parameters	Operand	Value
O_CMD	Tag_Input_O_CMD	TRUE
C_CMD	Tag_Input_C_CMD	FALSE
S_CMD	Tag_Input_S_CMD	FALSE
O_FB	Tag_Input_O_FB	FALSE
C_FB	Tag_Input_C_FB	FALSE
OO	Tag_OutputOpen	FALSE
CO	Tag_OutputClosed	FALSE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE
Q	Tag_Output_Q	FALSE

The following values are saved in the instance data block "MCAT_DB" of the instruction:

Parameters	Address	Value
ET	DBD4	L#2
PT	DBD8	L#22
O_HIS	DBX16.0	TRUE
C_HIS	DBX16.1	FALSE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameters	Operand	Value
OO	Tag_OutputOpen	TRUE
CO	Tag_OutputClosed	FALSE

Parameters	Operand	Value
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE
Q	Tag_Output_Q	TRUE

The following values are saved in the instance data block "MCAT_DB" of the instruction:

Parameters	Address	Value
ET	DBD4	L#0
O_HIS	DBX16.0	TRUE
CMD_HIS	DBX16.1	FALSE

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

IMC: Compare input bits with the bits of a mask (S7-1500)

Description

You can use the "Compare input bits with the bits of a mask" instruction to compare the signal state of up to 16 programmed input bits (IN_BIT0 to IN_BIT15) with the corresponding bits of a mask. Up to 16 steps with masks can be programmed. The value of the IN_BIT0 parameter is compared with the value of the mask CMP_VAL[x,0], with "x" indicating the step number. In the CMP_STEP parameter, you specify the step number of the mask that is used for the comparison. All programmed values are compared in the same manner. Unprogrammed input bits or unprogrammed bits of the mask have the default signal state FALSE.

If a match is found in the comparison, the signal state of the OUT parameter is set to "1". Otherwise the OUT parameter is set to "0".

If the value of the CMP_STEP parameter is greater than 15, the instruction is not executed. An error message is output at the ERR_CODE parameter.

Parameters

The following table shows the parameters of the "Compare input bits with the bits of a mask" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN_BIT0	Input	BOOL	I, Q, M, D, L or constant	Input bit 0 is compared with bit 0 of the mask.
IN_BIT1	Input	BOOL	I, Q, M, D, L or constant	Input bit 1 is compared with bit 1 of the mask.

Parameter	Declaration	Data type	Memory area	Description
IN_BIT2	Input	BOOL	I, Q, M, D, L or constant	Input bit 2 is compared with bit 2 of the mask.
IN_BIT3	Input	BOOL	I, Q, M, D, L or constant	Input bit 3 is compared with bit 3 of the mask.
IN_BIT4	Input	BOOL	I, Q, M, D, L or constant	Input bit 4 is compared with bit 4 of the mask.
IN_BIT5	Input	BOOL	I, Q, M, D, L or constant	Input bit 5 is compared with bit 5 of the mask.
IN_BIT6	Input	BOOL	I, Q, M, D, L or constant	Input bit 6 is compared with bit 6 of the mask.
IN_BIT7	Input	BOOL	I, Q, M, D, L or constant	Input bit 7 is compared with bit 7 of the mask.
IN_BIT8	Input	BOOL	I, Q, M, D, L or constant	Input bit 8 is compared with bit 8 of the mask.
IN_BIT9	Input	BOOL	I, Q, M, D, L or constant	Input bit 9 is compared with bit 9 of the mask.
IN_BIT10	Input	BOOL	I, Q, M, D, L or constant	Input bit 10 is compared with bit 10 of the mask.
IN_BIT11	Input	BOOL	I, Q, M, D, L or constant	Input bit 11 is compared with bit 11 of the mask.
IN_BIT12	Input	BOOL	I, Q, M, D, L or constant	Input bit 12 is compared with bit 12 of the mask.
IN_BIT13	Input	BOOL	I, Q, M, D, L or constant	Input bit 13 is compared with bit 13 of the mask.
IN_BIT14	Input	BOOL	I, Q, M, D, L or constant	Input bit 14 is compared with bit 14 of the mask.
IN_BIT15	Input	BOOL	I, Q, M, D, L or constant	Input bit 15 is compared with bit 15 of the mask.
CMP_STEP	Input	BYTE	I, Q, M, D, L, P or constant	The step number of the mask used for the comparison.
OUT	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that a match was found. A signal state of "0" indicates that no match was found.
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
CMP_VAL	Static	ARRAY OF WORD	I, Q, M, D, L or constant	Comparison masks [0 to 15, 0 to 15]: The first number of the index is the step number and the second number is the bit number of the mask.

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

Error code* (W#16#...)	Explanation
0000	No error
000A	The value at the CMP_STEP parameter is greater than 15.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

SMC: Compare scan matrix (S7-1500)

Description

You can use the "Compare scan matrix" instruction to compare the signal state of up to 16 programmed input bits (IN_BIT0 to IN_BIT15) with the corresponding bit of the comparison masks for each step. Processing starts at step 1 and is continued until the last programmed step (LAST) or until a match is found. The input bit of the IN_BIT0 parameter is compared with the value of the mask CMP_VAL[x,0], with "x" indicating the step number. All programmed values are compared in the same manner. If a match is found, the signal state of the OUT parameter is set to "1" and the step number with the matching mask is written to the OUT_STEP parameter. Unprogrammed input bits or unprogrammed bits of the mask have the default signal state FALSE. If more than one step has a matching mask, only the first one found is indicated in the OUT_STEP parameter. If no match is found, the signal state of the OUT parameter is set to "0". In this case the value at the OUT_STEP parameter is greater by "1" than the value at the LAST parameter.

Parameters

The following table shows the parameters of the "Compare scan matrix" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN_BIT0	Input	BOOL	I, Q, M, D, L or constant	Input bit 0 is compared with bit 0 of the mask.
IN_BIT1	Input	BOOL	I, Q, M, D, L or constant	Input bit 1 is compared with bit 1 of the mask.
IN_BIT2	Input	BOOL	I, Q, M, D, L or constant	Input bit 2 is compared with bit 2 of the mask.
IN_BIT3	Input	BOOL	I, Q, M, D, L or constant	Input bit 3 is compared with bit 3 of the mask.

Parameter	Declaration	Data type	Memory area	Description
IN_BIT4	Input	BOOL	I, Q, M, D, L or constant	Input bit 4 is compared with bit 4 of the mask.
IN_BIT5	Input	BOOL	I, Q, M, D, L or constant	Input bit 5 is compared with bit 5 of the mask.
IN_BIT6	Input	BOOL	I, Q, M, D, L or constant	Input bit 6 is compared with bit 6 of the mask.
IN_BIT7	Input	BOOL	I, Q, M, D, L or constant	Input bit 7 is compared with bit 7 of the mask.
IN_BIT8	Input	BOOL	I, Q, M, D, L or constant	Input bit 8 is compared with bit 8 of the mask.
IN_BIT9	Input	BOOL	I, Q, M, D, L or constant	Input bit 9 is compared with bit 9 of the mask.
IN_BIT10	Input	BOOL	I, Q, M, D, L or constant	Input bit 10 is compared with bit 10 of the mask.
IN_BIT11	Input	BOOL	I, Q, M, D, L or constant	Input bit 11 is compared with bit 11 of the mask.
IN_BIT12	Input	BOOL	I, Q, M, D, L or constant	Input bit 12 is compared with bit 12 of the mask.
IN_BIT13	Input	BOOL	I, Q, M, D, L or constant	Input bit 13 is compared with bit 13 of the mask.
IN_BIT14	Input	BOOL	I, Q, M, D, L or constant	Input bit 14 is compared with bit 14 of the mask.
IN_BIT15	Input	BOOL	I, Q, M, D, L or constant	Input bit 15 is compared with bit 15 of the mask.
OUT	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that a match was found. A signal state of "0" indicates that no match was found.
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
OUT_STEP	Output	BYTE	I, Q, M, D, L, P	Contains the step number with the matching mask, or the step number which is greater by "1" than the value at the LAST parameter, provided no match is found.
LAST	Static	BYTE	I, Q, M, D, L, P or constant	Specifies the step number of the last step to be scanned for a matching mask.
CMP_VAL	Static	ARRAY OF WORD	I, Q, M, D, L or constant	Comparison masks [0 to 15, 0 to 15]: The first number of the index is the step number and the second number is the bit number of the mask.

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

Error code* (W#16#...)	Explanation
0000	No error
000E	The value at the LAST parameter is greater than 15.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

LEAD_LAG: Lead and lag algorithm (S7-1500)

Description

You can use the "Lead and lag algorithm" instruction to process signals with an analog tag. The gain value at the GAIN parameter must be greater than zero. The result of the instruction "Lead and lag algorithm" is calculated using the following equation:

$$OUT = \left[\frac{LG_TIME}{LG_TIME + SAMPLE_T}\right] PREV_OUT + GAIN \left[\frac{LD_TIME + SAMPLE_T}{LG_TIME + SAMPLE_T}\right] IN - GAIN \left[\frac{LD_TIME}{LG_TIME + SAMPLE_T}\right] * PREV_IN$$

The "Lead and lag algorithm" instruction supplies plausible results only when processing is in fixed program cycles. The same units must be specified for the LD_TIME, LG_TIME, and SAMPLE_T parameters. At LG_TIME > 4 + SAMPLE_T, the instruction approaches the following function:

OUT = GAIN * ((1 + LD_TIME * s) / (1 + LG_TIME * s)) * IN

When the value of the GAIN parameter is less than or equal to zero, the calculation is not performed and an error information is output on the ERR_CODE parameter.

You can use the instruction "Lead and lag algorithm" in conjunction with loops as a compensator in dynamic feed-forward control. The instruction consists of two operations. The "Lead" operation shifts the phase of the OUT output so that the output leads the input. The "Lag" operation, on the other hand, shifts the output so that the output lags behind the input. Because the "Lag" operation is equivalent to an integration, it can be used as a noise suppressor or as a low-pass filter. The "Lead" operation is equivalent to a differentiation and can therefore be used as a high-pass filter. The two instructions together (Lead and Lag) result in the output phase lagging behind the the input at lower frequencies and leading it at higher frequencies. This means that the "Lead and lag algorithm" instruction can be used as a band pass filter.

Parameters

The following table shows the parameters of the "Lead and lag algorithm" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	REAL	I, Q, M, D, L, P or constant	The input value of the current sample time (cycle time) to be processed.
SAMPLE_T	Input	INT	I, Q, M, D, L, P or constant	Sample time
OUT	Output	REAL	I, Q, M, D, L, P	Result of the instruction
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
LD_TIME	Static	REAL	I, Q, M, D, L, P or constant	Lead time in the same unit as sample time.
LG_TIME	Static	REAL	I, Q, M, D, L, P or constant	Lag time in the same unit as sample time
GAIN	Static	REAL	I, Q, M, D, L, P or constant	Gain as % / % (the ratio of the change in output to a change in input as a steady state).
PREV_IN	Static	REAL	I, Q, M, D, L, P or constant	Previous input
PREV_OUT	Static	REAL	I, Q, M, D, L, P or constant	Previous output

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

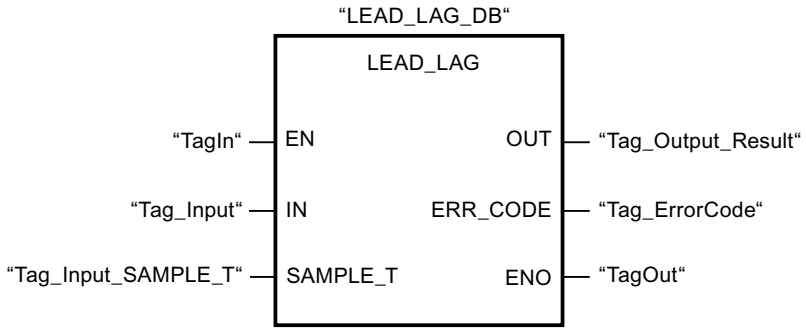
Error code* (W#16#...)	Explanation
0000	No error
0009	The value at the GAIN parameter is less than or equal to zero.
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".	

Example

The following example shows how the instruction works:

Note

You can initialize static parameters in the data block.



The following tables show how the instruction works using specific values.

Before processing

In this example the following values are used for the input parameters:

Parameters	Operand	Value
IN	Tag_Input	2.0
SAMPLE_T	Tag_InputSampleTime	10

The following values are saved in the "LEAD_LAG_DB" instance data block of the instruction:

Parameters	Address	Value
LD_TIME	DBD12	2.0
LG_TIME	DBD16	2.0
GAIN	DBD20	1.0
PREV_IN	DBD24	6.0
PREV_OUT	DBD28	6.0

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameters	Operand	Value
OUT	Tag_Output_Result	2.0

The following values are saved in the instance data block "LEAD_LAD_DB" of the instruction:

Parameters	Operand	Value
PREV_IN	DBD24	2.0
PREV_OUT	DBD28	2.0

See also

Overview of the valid data types (Page 2239)

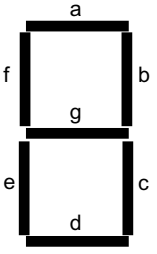
Basic information on FBD (Page 4937)

SEG: Create bit pattern for seven-segment display (S7-1500)

Description

The instruction "Create bit pattern for seven-segment display" is used to convert each of the four hexadecimal digits of the specified source word (IN) into an equivalent bit pattern for a seven-segment display. The result of the instruction is output in the double word on the OUT parameter.

The following relation exists between the hexadecimal digits and the assignment of the 7 segments (a, b, c, d, e, f, g):

Input digit (Binary)	**Assignment of the segments - g f e d c b a**	**Display** (Hexadecimal)	**Seven-segment display**
0000	00111111	0	
0001	00000110	1	
0010	01011011	2	
0011	01001111	3	
0100	01100110	4	
0101	01101101	5	
0110	01111101	6	
0111	00000111	7	
1000	01111111	8	
1001	01100111	9	
1010	01110111	A	
1011	01111100	B	
1100	00111001	C	
1101	01011110	D	
1110	01111001	E	
1111	01110001	F	

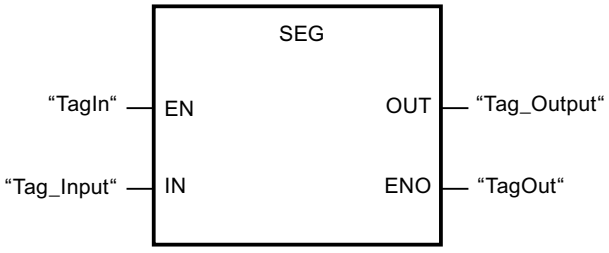
Parameters

The following table shows the parameters of the "Create bit pattern for seven-segment display" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output
IN	Input	WORD	I, Q, M, D, L, P or constant	Source word with four hexa-decimal digits
OUT	Output	DWORD	I, Q, M, D, L, P	Bit pattern for the seven-segment display

Example

The following example shows how the instruction works:



The following table shows how the instruction functions using specific values:

Parameters	Operand	Value	
Hexadecimal	Binary		
IN	Tag_Input	W#16#1234	0001 0010 0011 0100
OUT	Tag_Output	DW#16065B4F66	00000110 01011011 01001111 01100110 Display: 1234

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

BCDCPL: Create tens complement (S7-1500)

Description

You can use the "Create tens complement" instruction to create the tens complement of a seven-digit BCD number specified in the IN parameter. This instruction uses the following mathematical formula to calculate:

10000000 (as BCD)

– 7-digit BCD value

Tens complement (as BCD)

Parameters

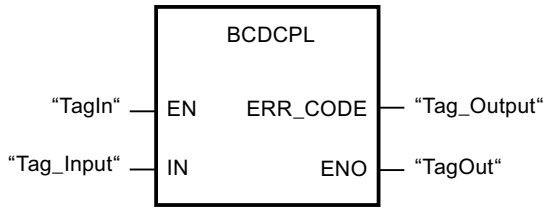
The following table shows the parameters of the "Create tens complement" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output

Parameter	Declaration	Data type	Memory area	Description
IN	Input	Bit strings	I, Q, M, D, L, P or constant	7-digit BCD number
ERR_CODE	Output	DWORD	I, Q, M, D, L, P	Result of the instruction

Example

The following example shows how the instruction works:



The following table shows how the instruction functions using specific values:

Parameter	Operand	Value*
IN	Tag_Input	DW#16#01234567
ERR_CODE	Tag_Output	DW#16#08765433
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".		

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

BITSUM: Count number of set bits (S7-1500)

Description

The "Count number of set bits" instruction is used to count the number of bits of an operand that is set to the signal state "1". The operand whose bits are to be counted is specified on the IN parameter. The result of the instruction is output at the RET_VAL parameter.

Parameters

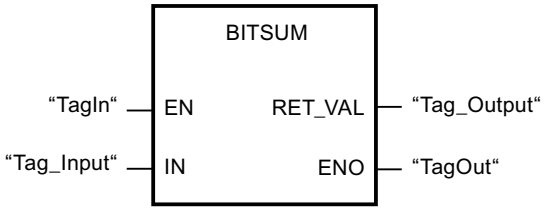
The following table shows the parameters of the "Count number of set bits" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN	Input	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	Output	BOOL	I, Q, M, D, L	Enable output

Parameter	Declaration	Data type	Memory area	Description
IN	Input	DWORD	I, Q, M, D, L, P or constant	Operand whose set bits are counted.
RET_VAL	Output	INT	I, Q, M, D, L, P	Number of bits to be set

Example

The following example shows how the instruction works:



The following table shows how the instruction functions using specific values:

Parameter	Operand	Value*
IN	Tag_Input	DW#16#12345678
RET_VAL	Tag_Output	W#16#000D (13 Bits)
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats under "See also".		

See also

Overview of the valid data types (Page 2239)

Basic information on FBD (Page 4937)

11.4.3.3 SCL (S7-1200, S7-1500)

Bit logic operations (S7-1200, S7-1500)

R_TRIG: Detect positive signal edge (S7-1200, S7-1500)

Description

With the "Detect positive signal edge" instruction, you can detect a state change from "0" to "1" at the CLK input. The instruction compares the current value at the CLK input with the state of the previous query (edge memory bit) that is saved in the specified instance. If the instruction detects a state change at the CLK input from "0" to "1", a positive signal edge is generated at the Q output, i.e., the output has the value TRUE or "1" for exactly one cycle.

In all other cases, the signal state at the output of the instruction is "0".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
CLK	Input	BOOL	I, Q, M, D, L	Incoming signal, the edge of which is to be queried
Q	Output	BOOL	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:

```
SCL
"R_TRIG_DB"(CLK := "TagIn",
            Q => "TagOut");
```

The previous state of the tag at the CLK input is stored in the "R_TRIG_DB" tag. If a change is the signal state from "0" to "1" is detected in the "TagIn_1" and "TagIn_2" operands or in the "TagIn_3" operand, the "TagOut_Q" output has signal state "1" for one cycle.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

F_TRIG: Detect negative signal edge (S7-1200, S7-1500)

Description

With the "Detect negative signal edge" instruction, you can detect a state change from "1" to "0" at the CLK input. The instruction compares the current value at the CLK input with the state of the previous query (edge memory bit) that is saved in the specified instance. If the instruction detects a state change at the CLK input from "1" to "0", a negative signal edge is generated at the Q output, i.e., the output has the value TRUE or "1" for exactly one cycle.

In all other cases, the signal state at the output of the instruction is "0".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
CLK	Input	BOOL	I, Q, M, D, L	Incoming signal, the edge of which is to be queried
Q	Output	BOOL	I, Q, M, D, L	Result of edge evaluation

Example

The following example shows how the instruction works:

SCL

```
"F_TRIG_DB"(CLK := "TagIn",
            Q => "TagOut");
```

The previous state of the tag at the CLK input is stored in the "F_TRIG_DB" tag. If a change in the signal state from "1" to "0" is detected in the "TagIn" operand, the "TagOut" output has signal state "1" for one cycle.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Timer operations (S7-1200, S7-1500)

Calling IEC timers (S7-1200, S7-1500)

Description

You have the option of declaring the IEC timer as single or multiple instance and calling it in the program code.

The following syntax options are available for the declaration of an IEC timer as multi-instance within a structure in the block interface:

IEC timer as ARRAY element

Declaration in the block interface:

	Static	
	MyARRAY	Array[0..1] of IEC_TIMER
	MyARRAY[0]	IEC_TIMER
	MyARRAY[1]	IEC_TIMER

Program code:

SCL

```
#MyARRAY[1].TOF(IN := <Operand>, PT := <Operand>)
```

Declaration in the block interface:

	Static	
	MyARRAY	Array[0..1] of TON_TIME
	MyARRAY[0]	TON_TIME
	MyARRAY[1]	TON_TIME

Program code:

SCL

```
#MyARRAY[#index](IN := <Operand>, PT := <Operand>)
```

IEC timer within an anonymous structure

Declaration in the block interface:

Static	
MyStruct	Struct
FirstTime	IEC_TIMER
PT	Time
ET	Time
IN	Bool
Q	Bool

Program code:

SCL

```
#MyStruct.FirstTime.TOF(IN := <Operand>, PT := <Operand>)
```

Declaration in the block interface:

Static	
MyStruct	Struct
FirstTime	TON_TIME
PT	Time
ET	Time
IN	Bool
Q	Bool

Program code:

SCL

```
#MyStruct.FirstTime(IN := <Operand>, PT := <Operand>)
```

IEC timer in the global data block

Declaration in the data block:

MyGlobalDB

	Name	Datentyp
1	Static	
2	Timer	"MyUDT"
3	FirstTime	IEC_TIMER
4	PT	Time
5	ET	Time
6	IN	Bool
7	Q	Bool

Program code:

SCL

```
"MyGlobalDB".Timer.FirstTime.TOF(IN := <Operand>, PT := <Operand>)
```

Declaration in the data block:

MyGlobalDB

	Name	Datentyp
1	Static	
2	Timer	"MyUDT"
3	FirstTime	IEC_TIMER
4	SecondTime	TON_TIME
5	PT	Time
6	ET	Time
7	IN	Bool
8	Q	Bool

Program code:

SCL

```
"MyGlobalDB".Timer.SecondTime(IN := <Operand>, PT := <Operand>)
```

IEC timer as elements in the block interface

Declaration in the block interface:

Static	
Timer	"MyUDT"
FirstTime	IEC_TIMER
PT	Time
ET	Time
IN	Bool
Q	Bool

Program code:

SCL

```
#Timer.FirstTime.TOF(IN := <Operand>, PT := <Operand>)
```

Declaration in the block interface:

Static	
Timer	"MyUDT"
FirstTime	IEC_TIMER
SecondTime	TON_TIME
PT	Time
ET	Time
IN	Bool
Q	Bool

Program code:

SCL

```
#Timer.SecondTime(IN := <Operand>, PT := <Operand>)
```

IEC timer within an ARRAY DB

Declaration in the ARRAY DB:

MyARRAYDB

		Name	Datentyp
1		MyARRAYDB	Array[0..1] of "MyUDT"
2		MyARRAYDB[0]	"MyUDT"
3		FirstTime	IEC_TIMER
4		PT	Time
5		ET	Time
6		IN	Bool
7		Q	Bool

Program code:

SCL

```
"MyARRAYDB"."THIS"[0].FirstTime.TOF(IN := <Operand>, PT := <Operand>)
```

Declaration in the ARRAY DB:

MyARRAYDB

		Name	Datentyp
1		MyARRAYDB	Array[0..1] of "MyUDT"
2		MyARRAYDB[0]	"MyUDT"
3		FirstTime	IEC_TIMER
4		SecondTime	TON_TIME
5		PT	Time
6		ET	Time
7		IN	Bool
8		Q	Bool

Program code:

SCL

```
"MyARRAYDB"."THIS"[0].SecondTime(IN := <Operand>, PT := <Operand>)
```

TP: Generate pulse (S7-1200, S7-1500)

Description

You can use the "Generate pulse" instruction to set the parameter Q for a programmed duration. The instruction is started when the result of logic operation (RLO) of the IN parameter changes from "0" to "1" (positive signal edge). The programmed time PT begins when the instruction starts. The Q parameter is set for the time PT, regardless of the subsequent changes in the input signal. Even when a new positive signal edge is detected, the signal state of the Q parameter is not affected as long as the time duration PT is active.

You can query the current time value set in the ET parameter. The timer value starts at T#0s and ends when the value of the time duration PT is reached. When the time duration PT is reached and the signal state at the IN parameter is "0", the ET parameter is reset.

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

Each call of the "Generate pulse" instruction must be assigned to an IEC timer in which the instruction data is stored.

The instruction data is updated in the following cases:

- When the instruction is called, if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

For information on calling IEC timers within structures (multi-instance), please refer to: Calling IEC timers (Page 3221)

For CPUs of the S7-1200 series

An IEC timer is a structure of the data type IEC_TIMER or TP_TIME that you can declare as follows:

- Declaration of an instance data block of system data type IEC_TIMER (for example, "MyIEC_TIMER_DB")
- Declaration as local tag of the data type TP_TIME in the "Static" section of a program block (for example, #MyIEC_TIMER_Instance)

For CPUs of the S7-1500 series

An IEC Timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TP_TIME or TP_LTIME that you can declare as follows:

- Declaration of an instance data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_LTIMER_DB")
- Declaration as local tag of the data type TP_TIME or TP_LTIME in the "Static" section of a program block (for example, #MyIEC_LTIMER_Instance)

IEC timer as instance data block of system data type IEC_<Timer> (Shared DB)

You can declare the IEC timer as a data block as follows:

<IEC_Timer_DB>.TP();

IEC timer as local tag from the block interface (multi-instance)

You can declare the IEC timer as a local tag as follows:

#myLocal_Timer();

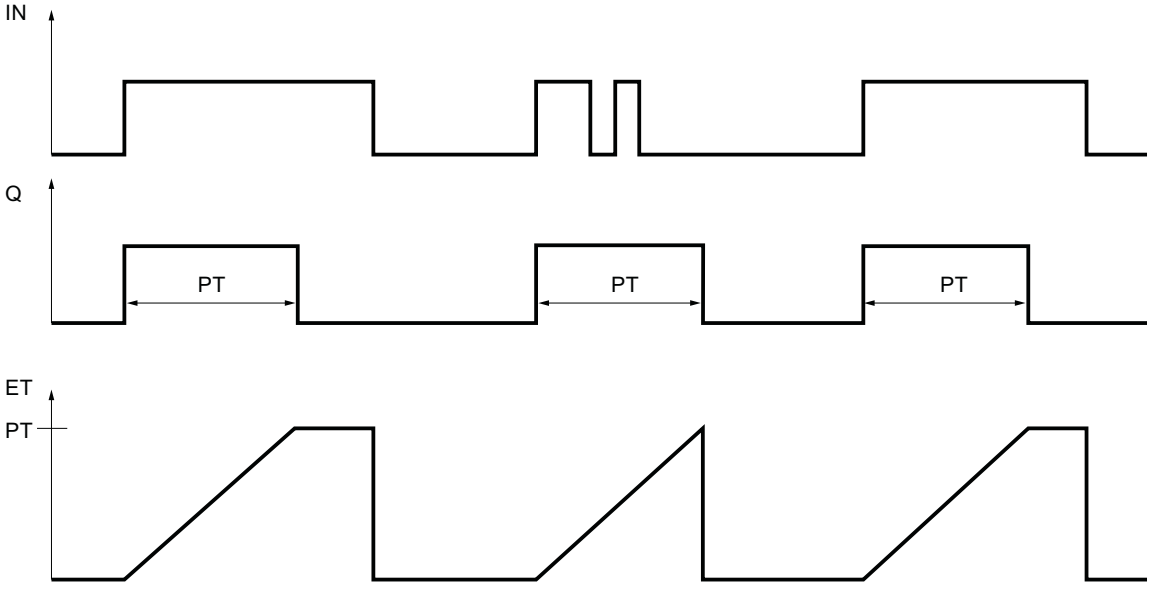
Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	BOOL	BOOL	I, Q, M, D, L, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L, P	Duration of the pulse. The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L, P	Operand that is set for the PT duration.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L, P	Current timer value

Pulse timing diagram

The following figure shows the pulse timing diagram of the "Generate pulse" instruction:



Example

The following example shows how the instruction works:

```
SCL
"TP_DB".TP(IN := "Tag_Start",
           PT := "Tag_PresetTime",
           Q => "Tag_Status",
           ET => "Tag_ElapsedTime");
```

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started and the "Tag_Status" operand is set to "1". The current time value is stored in the "Tag_ElapsedTime" operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

TON: Generate on-delay (S7-1200, S7-1500)

Description

You can use the "Generate on-delay" instruction to delay the setting of the Q parameter for the programmed duration PT. The instruction is started when the result of logic operation (RLO) of the IN parameter changes from "0" to "1" (positive signal edge). The programmed time PT begins when the instruction starts. When the PT duration has expired, the Q parameter returns signal state "1". The Q parameter remains set as long as the start input is still "1". If the signal state of the IN parameter changes from "1" to "0", the parameter Q will be reset. The timer function is restarted when a new positive signal edge is detected at the IN parameter.

The current time value can be queried in the ET parameter. The time value starts at T#0s and ends when the value of the time duration PT is reached. The ET parameter is reset as soon as the signal state of the IN parameter changes to "0".

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

Each call of the "Generate on-delay" instruction must be assigned to an IEC timer in which the instruction data is stored.

The instruction data is updated in the following cases:

- When the instruction is called, if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

For information on calling IEC timers within structures (multi-instance), please refer to: Calling IEC timers (Page 3221)

For CPUs of the S7-1200 series

An IEC timer is a structure of the data type IEC_TIMER or TON_TIME that you can declare as follows:

- Declaration of an instance data block of system data type IEC_TIMER (for example, "MyIEC_TIMER_DB")
- Declaration as local tag of the data type TON_TIME in the "Static" section of a program block (for example, #MyIEC_TIMER_Instance)

For CPUs of the S7-1500 series

An IEC Timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TON_TIME or TON_LTIME that you can declare as follows:

- Declaration of an instance data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER_DB")
- Declaration as local tag of the data type TON_TIME or TON_LTIME in the "Static" section of a program block (for example, #MyIEC_TIMER_Instance)

IEC timer as instance data block of system data type IEC_<Timer> (Shared DB)

You can declare the IEC timer as a data block as follows:

<IEC_Timer_DB>.TON();

IEC timer as local tag from the block interface (multi-instance)

You can declare the IEC timer as a local tag as follows:

#myLocal_Timer();

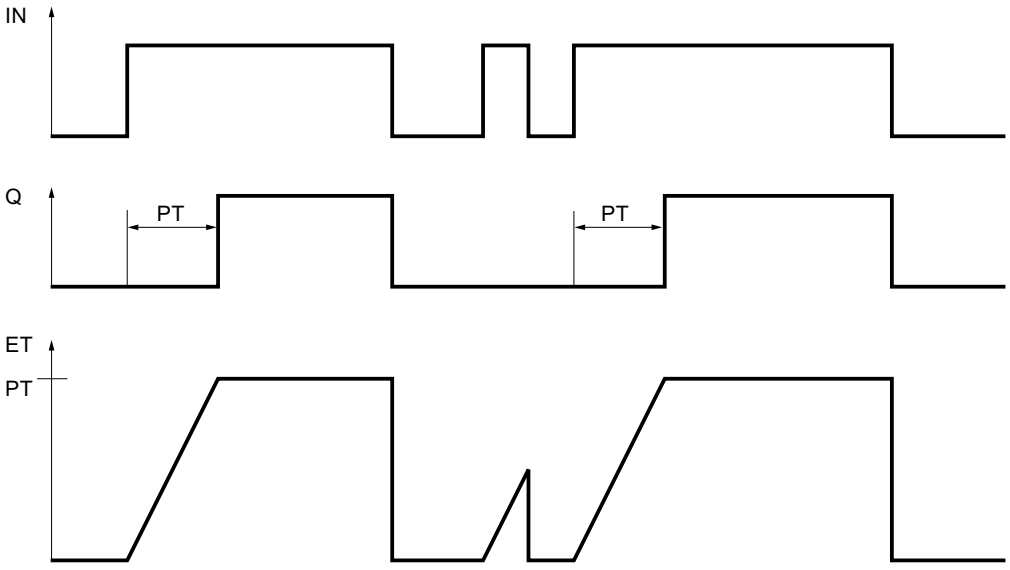
Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	BOOL	BOOL	I, Q, M, D, L, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L, P	Duration of the on delay. The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L, P	Operand that is set when the timer PT expires.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L, P	Current timer value

Pulse timing diagram

The following figure shows the pulse timing diagram of the "Generate on-delay" instruction:



Example

The following example shows how the instruction works:

```
SCL
"TON_DB".TON(IN := "Tag_Start",
             PT := "Tag_PresetTime",
             Q => "Tag_Status",
             ET => "Tag_ElapsedTime");
```

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started. When the time duration expires, the "Tag_Status" operand is set to the signal state "1". The Tag_Status operand remains set to "1" as long as the Tag_Start operand has signal state "1". The current time value is stored in the "Tag_ElapsedTime" operand. When the signal state at the operand "Tag_Start" changes from "1" to "0", the "Tag_Status" operand is reset.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

TOF: Generate off-delay (S7-1200, S7-1500)

Description

You can use the "Generate off-delay" instruction to delay the resetting of the Q parameter for the programmed duration PT. The Q parameter is set when the result of logic operation (RLO) of the IN parameter changes from "0" to "1" (positive signal edge). When the signal state at the IN parameter changes back to "0", the programmed time PT starts. The Q parameter remains set as long as time PT is running. When the time PT expires, the Q parameter is reset. If the signal state of the IN parameter changes to "1" before time PT expires, the timer is reset. The signal state of the Q parameter remains set to "1".

The current time value can be queried in the ET parameter. The timer value starts at T#0s and ends when the value of the time duration PT is reached. When the time duration PT expires, the ET parameter remains set to the current value until the IN parameter changes back to "1". If the IN parameter changes to "1" before the time PT has expired, the ET parameter is reset to the value T#0s.

Note

If the timer is not called in the program because it is skipped, for example, output ET returns a constant value as soon as the timer has expired.

Each call of the "Generate off-delay" instruction must be assigned to an IEC Timer in which the instruction data is stored.

The instruction data is updated in the following cases:

- When the instruction is called, if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

For information on calling IEC timers within structures (multi-instance), please refer to: Calling IEC timers (Page 3221)

For CPUs of the S7-1200 series

An IEC timer is a structure of the data type IEC_TIMER or TOF_TIME that you can declare as follows:

- Declaration of an instance data block of system data type IEC_TIMER (for example, "MyIEC_TIMER_DB")
- Declaration as local tag of the data type TOF_TIME in the "Static" section of a program block (for example, #MyIEC_TIMER_Instance)

For CPUs of the S7-1500 series

An IEC Timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TOF_TIME or TOF_LTIME that you can declare as follows:

- Declaration of an instance data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER_DB")
- Declaration as local tag of the data type TOF_TIME or TOF_LTIME in the "Static" section of a program block (for example, #MyIEC_TIMER_Instance)

IEC timer as instance data block of system data type IEC_<Timer> (Shared DB)

You can declare the IEC timer as a data block as follows:

<IEC_Timer_DB>.TOF();

IEC timer as local tag from the block interface (multi-instance)

You can declare the IEC timer as a local tag as follows:

#myLocal_Timer();

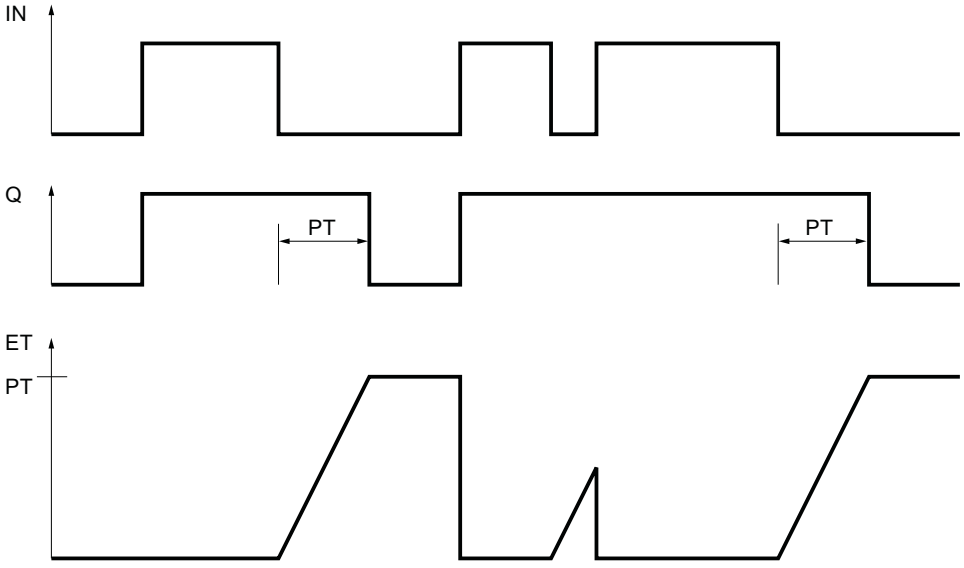
Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	BOOL	BOOL	I, Q, M, D, L, P	Start input
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L, P	Duration of the off delay. The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L, P	Operand that is reset when the time PT expires.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L, P	Current timer value

Pulse timing diagram

The following figure shows the pulse diagram of the "Generate off-delay" instruction:



Example

The following example shows how the instruction works:

```
SCL
"TOF_DB".TOF(IN := "Tag_Start",
             PT := "Tag_PresetTime",
             Q => "Tag_Status",
             ET => "Tag_ElapsedTime");
```

With a change in the signal state of the "Tag_Start" operand from "0" to "1", the "Tag_Status" operand is set. When the signal state of the "Tag_Start" operand changes from "1" to "0", the time programmed for the PT parameter is started. As long as the time is running, the "Tag_Status" operand remains set. When the time has expired, the "Tag_Status" operand is reset. The current time value is stored in the "Tag_ElapsedTime" operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

TONR: Time accumulator (S7-1200, S7-1500)

Description

The instruction "Time accumulator" is used to accumulate time values within a period set by the parameter PT. When the signal state of the IN parameter changes to "1", the instruction executes and the time duration PT starts. While the time duration PT is running, the time values that are recorded when the IN parameter has signal state "1" are accumulated. The accumulated time is output in the ET parameter and can be queried there. When the time duration PT is reached, the Q parameter has signal state "1". The Q parameter remains set to "1", even when the signal state at the IN parameter changes to "0".

The R parameter resets the ET and Q parameters regardless of the signal state at the IN parameter.

Each call of the "Time accumulator" instruction must be assigned an IEC timer in which the instruction data is stored.

The instruction data is updated in the following cases:

- When the instruction is called, if the ET or Q outputs are interconnected. If the outputs are not interconnected, the current time value at output ET is not updated.
- At an access to the Q or ET outputs.

For information on calling IEC timers within structures (multi-instance), please refer to: Calling IEC timers (Page 3221)

For CPUs of the S7-1200 series

An IEC Timer is a structure of the data type IEC_TIMER or TONR_TIME that you can declare as follows:

- Declaration of an instance data block of system data type IEC_TIMER (for example, "MyIEC_TIMER_DB")
- Declaration as local tag of the data type TONR_TIME in the "Static" section of a program block (for example, #MyIEC_TIMER_Instance)

For CPUs of the S7-1500 series

An IEC Timer is a structure of the data type IEC_TIMER, IEC_LTIMER, TONR_TIME or TONR_LTIME that you can declare as follows:

- Declaration of an instance data block of system data type IEC_TIMER or IEC_LTIMER (for example, "MyIEC_TIMER_DB")
- Declaration as local tag of the data type TONR_TIME or TONR_LTIME in the "Static" section of a program block (for example, #MyIEC_TIMER_Instance)

IEC timer as instance data block of system data type IEC_<Timer> (Shared DB)

You can declare the IEC timer as a data block as follows:

<IEC_Timer_DB>.TONR();

IEC timer as local tag from the block interface (multi-instance)

You can declare the IEC timer as a local tag as follows:

#myLocal_Timer();

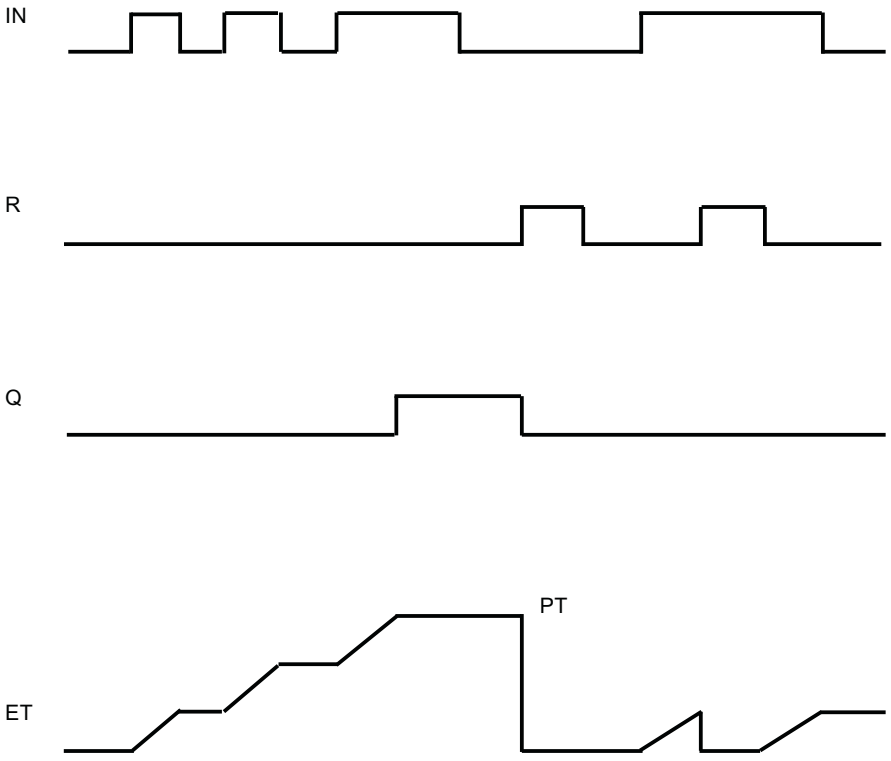
Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	BOOL	BOOL	I, Q, M, D, L, P	Start input
R	Input	BOOL	BOOL	I, Q, M, D, L, P	Reset of the ET and Q parameters
PT	Input	TIME	TIME, LTIME	I, Q, M, D, L, P	Maximum duration of time recording. The value of the PT parameter must be positive.
Q	Output	BOOL	BOOL	I, Q, M, D, L, P	Operand that remains set when the timer PT has expired.
ET	Output	TIME	TIME, LTIME	I, Q, M, D, L, P	Accumulated time

Pulse timing diagram

The following figure shows the pulse timing diagram of the "Time accumulator" instruction:



Example

The following example shows how the instruction works:

```
SCL
"TONR_DB".TONR(IN := "Tag_Start",
               R := "Tag_Reset",
               PT := "Tag_PresetTime",
               Q => "Tag_Status",
               ET => "Tag_Time");
```

When the signal state of the "Tag_Start" operand changes from "0" to "1", the time programmed for the PT parameter is started. While the time is running, the time values are accumulated that are recorded when the "Tag_Start" operand has signal state "1". The accumulated time is stored in the "Tag_Time" operand. When the time value specified for the PT parameter is reached, the "Tag_Status" operand is set to the signal state "1". The current time value is stored in the "Tag_Time" operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

RESET_TIMER: Reset timer (S7-1200, S7-1500)

Description

You can use the "Reset timer" instruction to reset an IEC timer to "0". The structure components of the timer in the specified data block are reset to "0".

The instruction does not influence the RLO. At the TIMER parameter, the "Reset timer" instruction is assigned an IEC timer declared in the program. The instruction must be programmed in an IF instruction. The instruction data is updated only when the instruction is called and not each time the assigned IEC timer is accessed. Querying the data is only identical from the call of the instruction to the next call of the instruction.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<IEC timer>	Output	IEC_TIMER, TP_TIME, TON_TIME, TOF_TIME, TONR_TIME	IEC_TIMER, IEC_LTIMER, TP_TIME, TP_LTIME, TON_TIME, TON_LTIME, TOF_TIME, TOF_LTIME, TONR_TIME, TONR_LTIME	D, L	IEC timer that is reset

Example

The following example shows how the instruction works:

SCL

```
IF #started = false THEN
"TON_DB".TON(IN := "Tag_Start",
             PT := "Tag_PresetTime",
             Q => "Tag_Status",
             ET => "Tag_ElapsedTime");
#started := true;
END_IF;

IF "TON_DB".ET < T#25s THEN
```

SCL

```
"RESET_TIMER".(TIMER := "TON_DB"),
#started := false;
END_IF;
```

When the tag #started has the signal state "0", the instruction "Generate on-delay" is executed when there is positive signal edge on the operand "Tag_Start". The IEC timer stored in the instance data block "TON_DB" is started with the time duration that is specified by the operand "Tag_PresetTime". The operand "Tag_Status" is set if the time duration specified by the operand "Tag_PresetTime" has expired. The parameter Q remains set as long as the operand "Tag_Start" still has the signal state "1". When the signal state of the start input changes from "1" to "0", the operand on the Q parameter is reset.

If the expired time of the IEC timer "TON_DB" is less than 25s, the "Reset timer" instruction is executed and the timer stored in the "TON_DB" instance data block is reset.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

PRESET_TIMER: Load time duration (S7-1200, S7-1500)

Description

You can use the "Load time duration" instruction to set the time for an IEC timer. The instruction is executed in every cycle when the result of logic operation (RLO) at the input of the instruction has the signal state "1". The instruction writes the specified time to the structure of the specified IEC timer.

Note

If the specified IEC timer is running while the instruction executes, the instruction overwrites the current time of the specified IEC timer. This can change the timer status of the IEC timer.

You assign an IEC timer declared in the program to the "Load time duration" instruction.

The instruction data is updated only when the instruction is called and each time the assigned IEC timer is accessed. The query on Q or ET (for example, "MyTimer".Q or "MyTimer".ET) updates the IEC_TIMER structure.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<Time duration>	Input	TIME	TIME, LTIME	I, Q, M, D, L	Duration with which the IEC timer runs.
<IEC timer>	Output	IEC_TIMER, TP_TIME, TON_TIME, TOF_TIME, TONR_TIME	IEC_TIMER, IEC_LTIMER, TP_TIME, TP_LTIME, TON_TIME, TON_LTIME, TOF_TIME, TOF_LTIME, TONR_TIME, TONR_LTIME	D, L	IEC timer whose duration of which is set.

Example

The following example shows how the instruction works:

SCL

```
IF #started = false THEN
"TON_DB".TON(IN := "Tag_Start",
             PT := "Tag_PresetTime",
             Q => "Tag_Status",
             ET => "Tag_ElapsedTime");
#started := true;
#preset = true
END_IF;

IF "TON_DB".ET < T#10s AND #preset = true THEN
PRESET_TIMER(PT := T#25s,
             TIMER := "TON_DB");
#preset := false;
END_IF;
```

When the tag #started has the signal state "0" and the operand "Tag_Start" has a positive signal edge, the instruction "Generate on-delay" is executed. The IEC timer stored in the instance data block "TON_DB" is started with the time duration that is specified by the operand "Tag_PresetTime". The operand "Tag_Status" is set if the time duration PT specified by the operand "Tag_PresetTime" has expired. The parameter Q remains set as long as the operand

"Tag_Start" still has the signal state "1". When the signal state of the start input changes from "1" to "0", the operand on the Q parameter is reset.

The instruction "Load time duration" is executed when the expired time of the IEC timer "TON_DB" is less than 10s and the tag #preset has the signal state "1". The instruction writes the time duration that is specified at the parameter PT in the instance data block "TON_DB", thereby overwriting the time value of the operand "Tag_PresetTime" within the instance data block. The signal state of the timer status may therefore change at the next query or when "TON_DB".Q or "TON_DB".ET are accessed.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Legacy (S7-1500)

S_PULSE: Assign pulse timer parameters and start (S7-1500)

Description

The "Assign pulse timer parameters and start" instruction starts the time programmed in the T_NO parameter when a change from "0" to "1" (positive signal edge) is detected in the result of logic operation (RLO) of the S parameter. The timer runs for the programmed time (TV) as long as the signal state of the S parameter is "1".

When the signal state of the S parameter changes to "0" before the programmed time has expired, the timer is stopped and the Q parameter is reset to "0".

Internally, the time is made up of a time value and a time base and is programmed in the TV parameter. When the instruction starts, the programmed time value counts down to zero. The time base specifies the time increment by which the time value changes. The current time value is provided at the parameter BI.

If the timer is running and the signal state at input R changes to "1" then the current time value and the time base are also set to zero. If the timer is not running, the signal state "1" at the R input has no effect.

Parameter Q returns signal state "1" as long as the timer is running and the signal state at parameter S is "1". When the signal state of the S parameter changes to "0" before the programmed time has expired, the Q parameter returns signal state "0". If the timer is reset by parameter R or if the timer has expired then parameter Q also returns signal state "0".

The instruction data is updated with each access. It is therefore possible that the query of the data at the start of the cycle returns different values from those at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

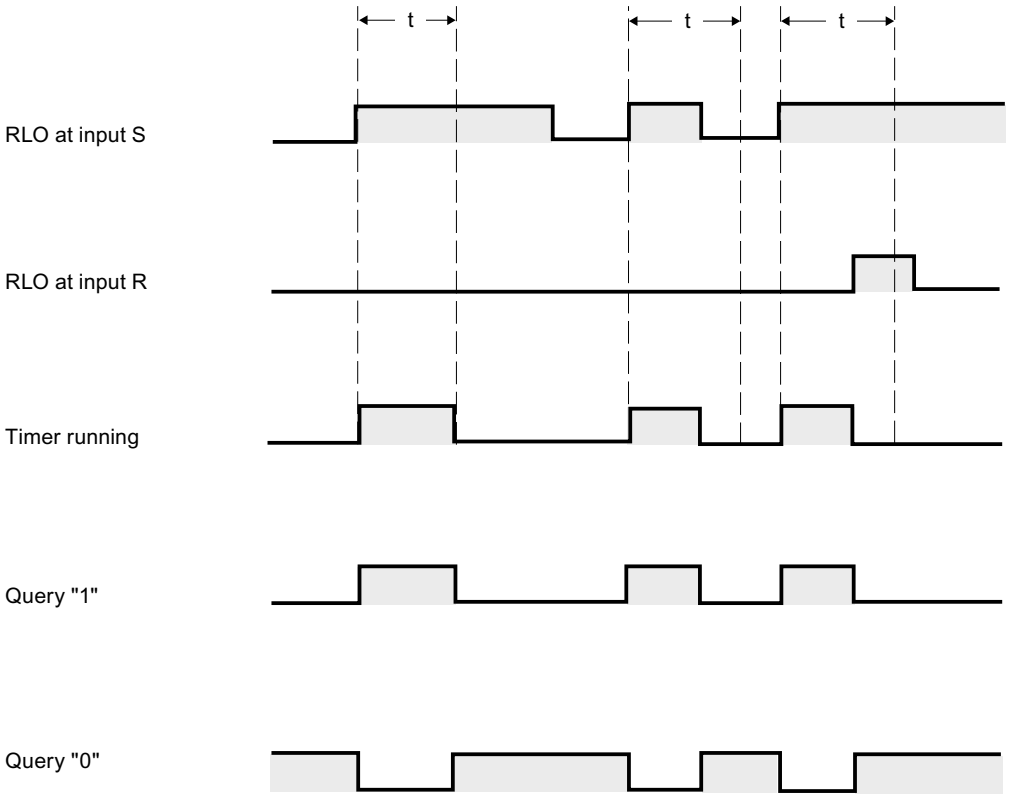
The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
T_NO	Input	TIMER, INT	T	The timer that is started. The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L	Preset time value
R	Input	BOOL	I, Q, M, D, L, P	Reset input
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer
BI	Output	WORD	I, Q, M, D, L, P	Current dual-coded timer value
Function value		S5TIME	I, Q, M, D, L	Current timer value

For additional information on valid data types, refer to "See also".

Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign pulse timer parameters and start" instruction:



t = programmed time

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := S_PULSE(T_NO := "Timer_1",
                        S := "Tag_1",
                        TV := "Tag_Number",
                        R := "Tag_Reset",
                        Q => "Tag_Status",
                        BI => "Tag_Value");
```

"Timer_1" starts when the signal state of the "Tag_1" operand changes from "0" to "1". The timer counts down with the time value of the operand "Tag_Number" until operand "Tag_1" returns signal state "1".

If the signal state at the S parameter changes to "0" before the programmed time has elapsed, the "Tag_Status" operand is reset to "0". If the timer is reset by the R parameter or if the timer has expired, the "Tag_Status" operand also returns signal state "0".

The current time value is stored both dual-coded at the "Tag_Value" operand and returned as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

S_PEXT: Assign extended pulse timer parameters and start (S7-1500)

Description

The "Assign extended pulse timer parameters and start" instruction starts a programmed timer when a positive signal edge is detected at the S parameter. The timer runs for the programmed time (TV) even if the signal state at the S parameter changes to "0". As long as the timer runs, parameter Q returns the signal state "1".

When the timer has expired, parameter Q is reset to "0". If the signal state at the S parameter changes from "0" to "1" while the timer is running, the timer is restarted with the time programmed in the TV parameter.

Internally, the time is made up of a time value and a time base and is programmed in the TV parameter. When the instruction starts, the programmed time value counts down to zero. The time base specifies the time increment by which the time value changes. The current time value is provided at the parameter BI.

If the timer is running and the signal state at parameter R changes to "1" then the current time value and the time base are also set to zero. If the timer is not running then signal state "1" at parameter R has no effect.

The instruction data is updated with each access. It is therefore possible that the query of the data at the start of the cycle returns different values from those at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

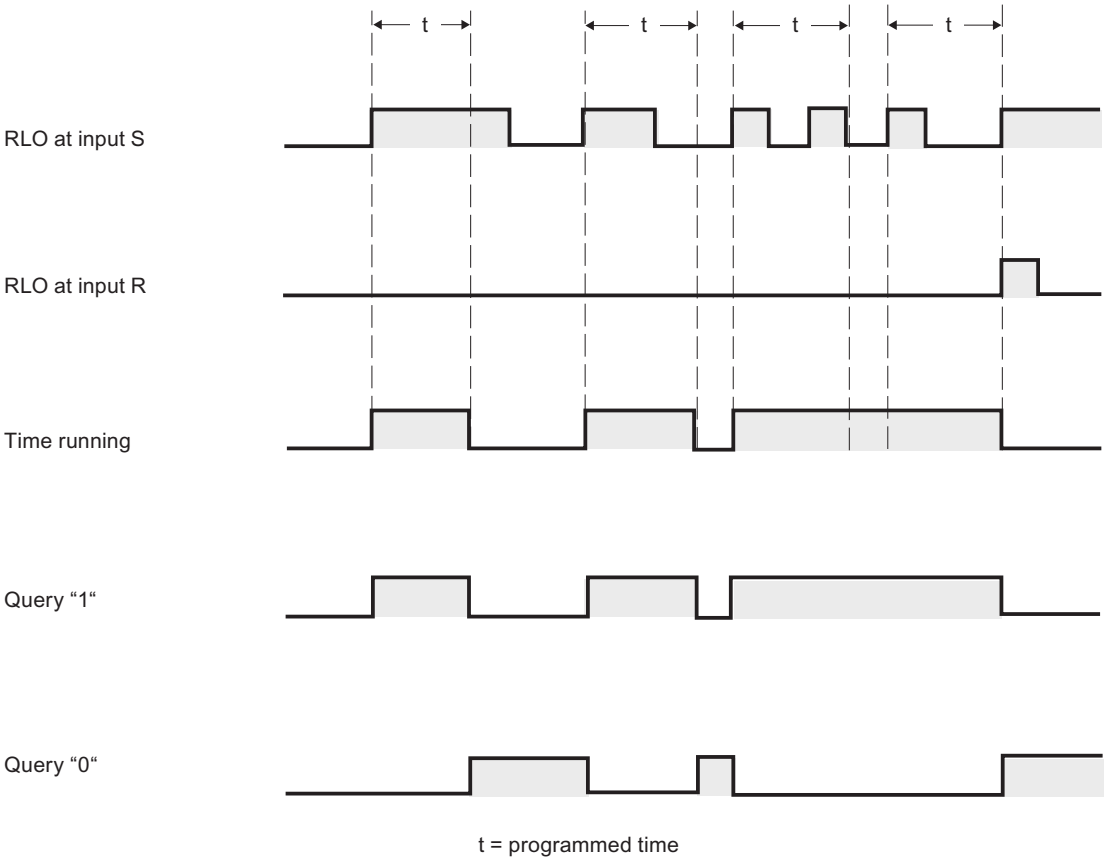
The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
T_NO	Input	TIMER, INT	T	The timer that is started. The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L	Preset time value
R	Input	BOOL	I, Q, M, D, L, P	Reset input
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer
BI	Output	WORD	I, Q, M, D, L, P	Current dual-coded timer value
Function value		S5TIME	I, Q, M, D, L	Current timer value

For additional information on valid data types, refer to "See also".

Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign extended pulse timer parameters and start" instruction:



t = programmed time

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := S_PEXT(T_NO := "Timer_1",
                       S := "Tag_1",
                       TV := "Tag_Number",
                       R := "Tag_Reset",
                       Q => "Tag_Status",
                       BI => "Tag_Value");
```

"Timer_1" starts when the signal state of the "Tag_1" operand changes from "0" to "1". As long as the timer runs, operand "Tag_Status" returns the signal state "1". When the timer has expired, operand "Tag_Status" is reset to "0". If the signal state at the S input changes from "0" to "1" while the timer is running, the timer is restarted with the time "Tag_Number".

The current time value is stored both dual-coded at the "Tag_Value" operand and returned as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

S_ODT: Assign on-delay timer parameters and start (S7-1500)

Description

The "Assign on-delay timer parameters and start" instruction starts a programmed timer as on-delay when a positive signal edge is detected at the S parameter. The timer runs for the programmed time (TV) as long as the signal state of the S parameter is "1".

If the timer expires correctly and parameter S still has signal state "1" then parameter Q returns signal state "1". If the signal state at the S parameter changes from "1" to "0" while the timer is running, the timer is stopped. In this case, output Q is reset to signal state "0".

Internally, the time is made up of a time value and a time base and is programmed in the TV parameter. When the instruction starts, the programmed time value counts down to zero. The time base specifies the time increment by which the time value changes. The current time value is provided at the parameter BI.

If the time is running and the signal state at input R changes from "0" to "1" then the current time value and the time base are also set to zero. In this case, the signal state at parameter Q is "0". The timer is reset if the signal state at the R parameter is "1", even if the timer is not running and the result of logic operation (RLO) at the S parameter is "1".

The instruction data is updated with each access. It is therefore possible that the query of the data at the start of the cycle returns different values from those at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

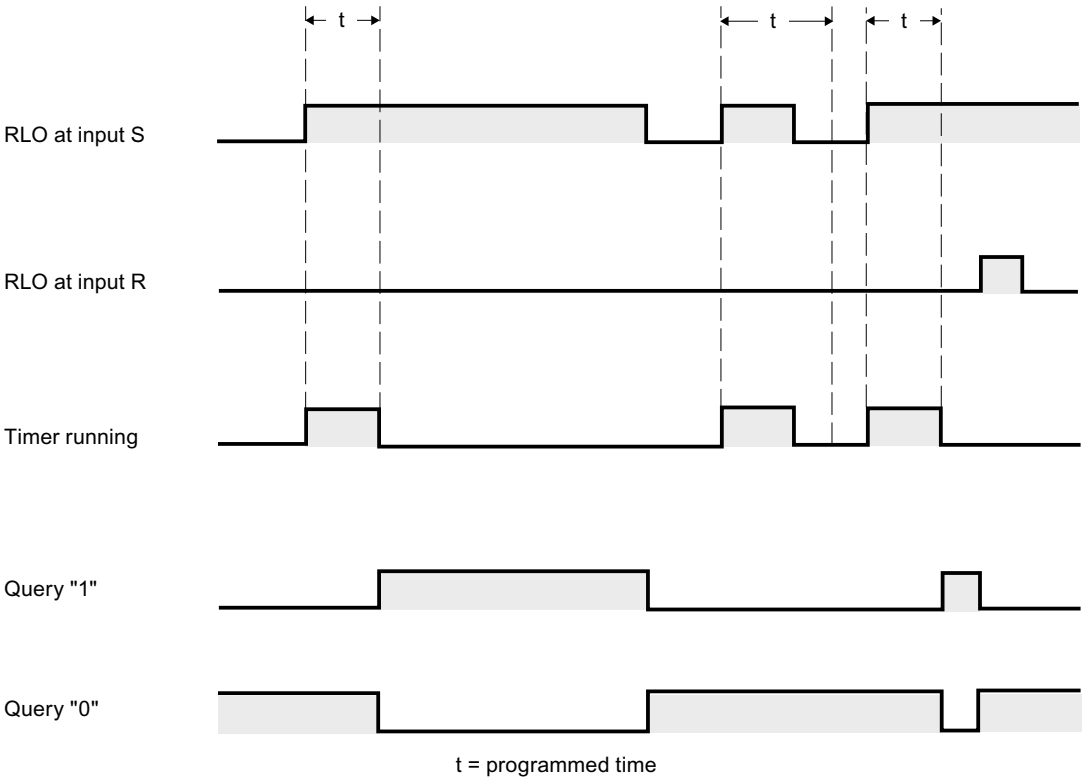
The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
T_NO	Input	TIMER, INT	T	The timer that is started. The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L	Preset time value
R	Input	BOOL	I, Q, M, D, L, P	Reset input
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer
BI	Output	WORD	I, Q, M, D, L, P	Current dual-coded timer value
Function value		S5TIME	I, Q, M, D, L	Current timer value

For additional information on valid data types, refer to "See also".

Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign on-delay timer parameters and start" instruction:



Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := S_ODT(T_NO := "Timer_1",
                      S := "Tag_1",
                      TV := "Tag_Number",
                      R := "Tag_Reset",
                      Q => "Tag_Status",
                      BI => "Tag_Value");
```

"Timer_1" starts when the signal state of the "Tag_1" operand changes from "0" to "1". The timer runs for the duration "Tag_Number" as long as the signal state of operand "Tag_1" is "1".

If the timer expires correctly and operand "Tag_Status" has signal state "1" then operand "Tag_Status" is reset to "1". If the signal state of the "Tag_1" operand changes from "1" to "0" while the timer is running, the timer is stopped. In this case, operand "Tag_Status" returns signal state "0".

The current time value is stored both dual-coded at the "Tag_Value" operand and returned as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

S_ODTS: Assign retentive on-delay timer parameters and start (S7-1500)

Description

The "Assign retentive on-delay timer parameters and start" instruction starts a programmed timer when a positive signal edge is detected at the S parameter. The timer runs for the programmed time (TV) even if the signal state at the S parameter changes to "0".

When the timer expires, the "Q" parameter returns signal state "1" regardless of the signal state of the "S" parameter. If the signal state at the S parameter changes from "0" to "1" while the timer is running, the timer is restarted with the programmed time TV.

Internally, the time is made up of a time value and a time base and is programmed in the TV parameter. When the instruction starts, the programmed time value counts down to zero. The time base specifies the time increment by which the time value changes. The current time value is provided at the parameter BI.

Signal state "1" at parameter R resets the current time value and time base to "0", independent of the signal state at parameter S. In this case, the signal state at parameter Q is "0".

The instruction data is updated with each access. It is therefore possible that the query of the data at the start of the cycle returns different values from those at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

The following table shows the parameters of the instruction:

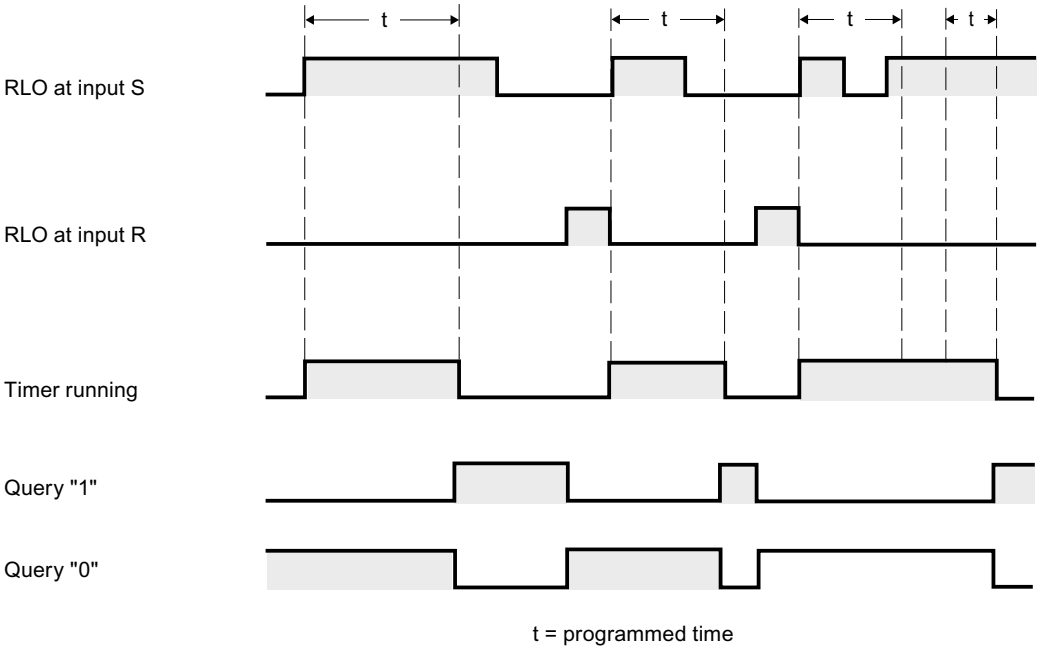
Parameters	Declaration	Data type	Memory area	Description
T_NO	Input	TIMER, INT	T	The timer that is started. The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L	Preset time value
R	Input	BOOL	I, Q, M, D, L, P	Reset input

Parameters	Declaration	Data type	Memory area	Description
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer
BI	Output	WORD	I, Q, M, D, L, P	Current dual-coded timer value
Function value		S5TIME	I, Q, M, D, L	Current timer value

For additional information on valid data types, refer to "See also".

Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign retentive on-delay timer parameters and start" instruction:



t = programmed time

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := S_ODTS(T_NO := "Timer_1",
                       S := "Tag_1",
                       TV := "Tag_Number",
                       R := "Tag_Reset",
                       Q => "Tag_Status",
                       BI => "Tag_Value");
```

"Timer_1" starts when the signal state of the "Tag_1" operand changes from "0" to "1". The timer runs for the duration "Tag_Number".

If the timer expires, operand "Tag_Status" returns signal state "1" independent of the signal state of operand "Tag_1". If the signal state of the "Tag_1" operand changes from "0" to "1" while the timer is running, the timer is restarted with the time "Tag_Number".

The current time value is stored both dual-coded at the "Tag_Value" operand and returned as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

S_OFFDT: Assign off-delay timer parameters and start (S7-1500)

Description

The "Assign off-delay timer parameters and start" instruction starts a programmed timer when a negative signal edge is detected at the S parameter. The timer runs for the programmed time (TV). As long as the timer is running or parameter S returns signal state "1", then parameter Q has signal state "1".

If the timer expires and the signal state is "0" then parameter Q is reset to signal state "0". If the signal state at parameter S changes from "0" to "1" while the timer is running, the timer is stopped. The timer is only restarted after a falling signal edge is detected at parameter S.

Internally, the time is made up of a time value and a time base and is programmed in the TV parameter. When the instruction starts, the programmed time value counts down to zero. The time base specifies the time increment by which the time value changes. The current time value is provided at the parameter BI.

Signal state "1" at parameter R resets the current time value and time base to "0". In this case, the signal state at parameter Q is "0".

The instruction data is updated with each access. It is therefore possible that the query of the data at the start of the cycle returns different values from those at the end of the cycle.

Note

In the time cell, the operating system reduces the time value in an interval specified by the time base by one unit until the value equals "0". The decrementation is performed asynchronously to the user program. The resulting timer is therefore at maximum up to one time interval shorter than the desired time base.

You can find an example of how a time cell can be formed under: See also "L: Load timer value".

Parameters

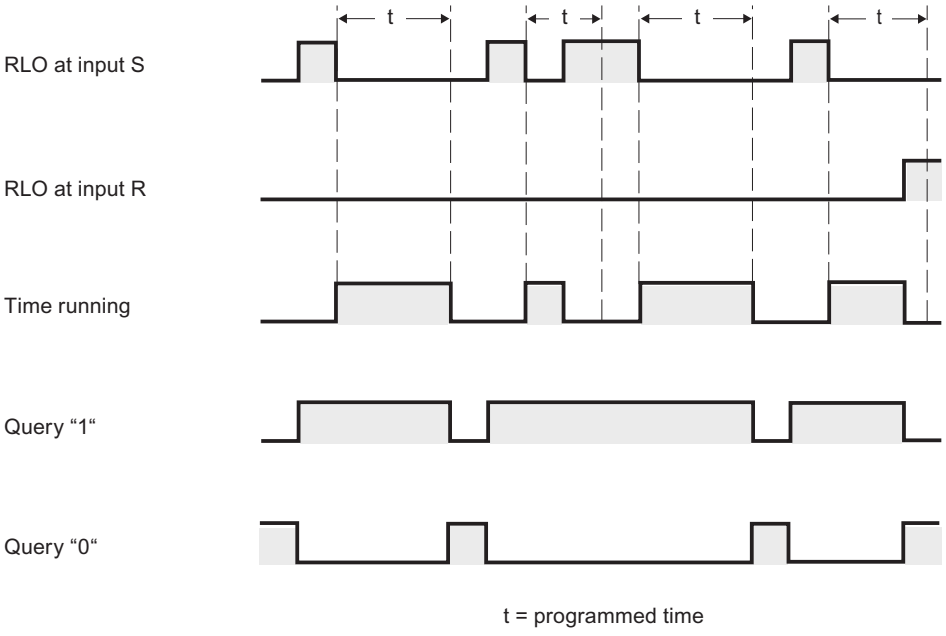
The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
T_NO	Input	TIMER, INT	T	The timer that is started. The number of timers depends on the CPU.
S	Input	BOOL	I, Q, M, D, L	Start input
TV	Input	S5TIME, WORD	I, Q, M, D, L	Preset time value
R	Input	BOOL	I, Q, M, D, L, P	Reset input
Q	Output	BOOL	I, Q, M, D, L, P	Status of the timer
BI	Output	WORD	I, Q, M, D, L, P	Current dual-coded timer value
Function value		S5TIME	I, Q, M, D, L	Current timer value

For additional information on valid data types, refer to "See also".

Pulse timing diagram

The following figure shows the pulse timing diagram of the "Assign off-delay timer parameters and start" instruction:



Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := S_OFFDT(T_NO := "Timer_1",
                        S := "Tag_1",
                        TV := "Tag_Number",
                        R := "Tag_Reset",
                        Q => "Tag_Status",
                        BI => "Tag_Value");
```

The timer "Timer_1" is started when the signal state of the operand "Tag_1" changes from "1" to "0". The timer runs for the duration "Tag_Number". As long as the timer is running or operand "Tag_1" returns signal state "1", then operand "Tag_Status" has signal state "1".

If the timer expires and the signal state of operand "Tag_1" is "0" then operand "Tag_Status" is reset to signal state "0". If the signal state of the "Tag_1" operand changes from "0" to "1" while the timer is running, the timer is reset. The timer is only restarted after a falling edge is detected at parameter S.

The current time value is stored both dual-coded at the "Tag_Value" operand and returned as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Counter operations (S7-1200, S7-1500)

Calling IEC counters (S7-1200, S7-1500)

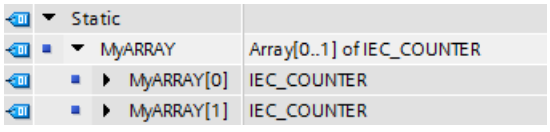
Description

You have the option of declaring the IEC counter as single or multiple instance and calling it in the program code.

The following syntax options are available for the declaration of an IEC counter as multi-instance within a structure in the block interface:

IEC counter as ARRAY element

Declaration in the block interface:



Program code:

SCL

```
#MyARRAY[1].CTU(CU := <Operand>, PV := <Operand>)
```

Declaration in the block interface:

	Static	
	MyARRAY	Array[0..1] of CTU_INT
	MyARRAY[0]	CTU_INT
	MyARRAY[1]	CTU_INT

Program code:

SCL

```
#MyARRAY[#index](CU := <Operand>, PV := <Operand>)
```

IEC counter within an anonymous structure

Declaration in the block interface:

	Static	
	MyStruct	Struct
	FirstTime	IEC_COUNTER
	CU	Bool
	CD	Bool
	R	Bool
	LD	Bool
	QU	Bool
	QD	Bool
	PV	Int
	CV	Int

Program code:

SCL

```
#MyStruct.FirstTime.CTU(CU := <Operand>, PV := <Operand>)
```

Declaration in the block interface:

	Static	
	MyStruct	Struct
	FirstTime	CTU_INT
	CU	Bool
	CD	Bool
	R	Bool
	LD	Bool
	QU	Bool
	QD	Bool
	PV	Int
	CV	Int

Program code:

SCL

```
#MyStruct.FirstTime(CU := <Operand>, PV := <Operand>)
```

IEC counter in the global data block

Declaration in the data block:

MyGlobalDB

	Name	Datentyp
1	Static	
2	Counter	"MyUDT"
3	FirstCount	IEC_COUNTER
4	CU	Bool
5	CD	Bool
6	R	Bool
7	LD	Bool
8	QU	Bool
9	QD	Bool
10	PV	Int
11	CV	Int

Program code:

SCL

```
"MyGlobalDB".Counter.FirstCount.CTU(CU := <Operand>, PV := <Operand>)
```

Declaration in the data block:

MyGlobalDB

	Name	Datentyp
1	Static	
2	Counter	"MyUDT"
3	FirstCount	IEC_COUNTER
4	SecondCount	CTU_INT
5	CU	Bool
6	CD	Bool
7	R	Bool
8	LD	Bool
9	QU	Bool
10	QD	Bool
11	PV	Int
12	CV	Int

Program code:

SCL

```
"MyGlobalDB".Counter.SecondCount(CU := <Operand>, PV := <Operand>)
```

IEC counter as element in the block interface

Declaration in the block interface:

Static		
Counter	"MyUDT"	
FirstCount	IEC_COUNTER	
CU	Bool	
CD	Bool	
R	Bool	
LD	Bool	
QU	Bool	
QD	Bool	
PV	Int	
CV	Int	

Program code:

SCL

```
#Counter.FirstCount.CTU(CU := <Operand>, PV := <Operand>)
```

Declaration in the block interface:

Static		
Counter	"MyUDT"	
FirstCount	IEC_COUNTER	
SecondCount	CTU_INT	
CU	Bool	
CD	Bool	
R	Bool	
LD	Bool	
QU	Bool	
QD	Bool	
PV	Int	
CV	Int	

Program code:

SCL

```
#Counter.SecondCount(CU := <Operand>, PV := <Operand>)
```

IEC counter within an ARRAY DB

Declaration in the ARRAY DB:

MyARRAYDB

	Name	Datentyp
1	MyARRAYDB	Array[0..1] of "MyUDT"
2	MyARRAYDB[0]	"MyUDT"
3	FirstCount	IEC_COUNTER
4	CU	Bool
5	CD	Bool
6	R	Bool
7	LD	Bool
8	QU	Bool
9	QD	Bool
10	PV	Int
11	CV	Int

Program code:

SCL

```
"MyARRAYDB"."THIS"[0].FirstCount.CTU(CU := <Operand>, PV := <Operand>)
```

Declaration in the ARRAY DB:

MyARRAYDB

	Name	Datentyp
1	MyARRAYDB	Array[0..1] of "MyUDT"
2	MyARRAYDB[0]	"MyUDT"
3	FirstCount	IEC_COUNTER
4	SecondCount	CTU_INT
5	CU	Bool
6	CD	Bool
7	R	Bool
8	LD	Bool
9	QU	Bool
10	QD	Bool
11	PV	Int
12	CV	Int

Program code:

SCL

```
"MyARRAYDB"."THIS"[0].SecondCount(CU := <Operand>, PV := <Operand>)
```

CTU: Count up (S7-1200, S7-1500)

Description

You can use the "Count up" instruction to increment the value at the CV parameter. When the signal state of the parameter CU changes from "0" to "1" (positive signal edge), the instruction is executed and the current counter value of the parameter CV is incremented by one. The counter value is incremented each time a positive signal edge is detected, until it reaches the high limit of the data type specified at the CV parameter. When the high limit is reached, the signal state of the CU parameter no longer has an effect on the instruction.

You can query the count status of the Q parameter. The signal state of the Q parameter is determined by the PV parameter. When the current counter value is greater than or equal to the value of the PV parameter, the Q parameter is set to signal state "1". In all other cases, the signal state of the Q parameter is "0". You can also specify a constant for the PV parameter.

The value of the CV parameter is reset to zero when the signal state at the R parameter changes to "1". As long as the signal state of the R parameter is "1", the signal state of the CU parameter has no effect on the instruction.

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count up" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For CPUs of the S7-1200 series

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER	• CTU_SINT / CTU_USINT
• IEC_COUNTER / IEC_UCOUNTER	• CTU_INT / CTU_UINT
• IEC_DCOUNTER / IEC_UDCOUNTER	• CTU_DINT / CTU_UDINT

For CPUs of the S7-1500 series

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER	• CTU_SINT / CTU_USINT
• IEC_COUNTER / IEC_UCOUNTER	• CTU_INT / CTU_UINT
• IEC_DCOUNTER / IEC_UDCOUNTER	• CTU_DINT / CTU_UDINT
• IEC_LCOUNTER / IEC_ULCOUNTER	• CTU_LINT / CTU_ULINT

You can declare an IEC counter as follows:

- Declaration of an instance data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER_DB")
- Declaration as local tag of the data type CTUD_<DatenType> in the "Static" section of a program block (for example, #MyIEC_COUNTER_Instance)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive.

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

IEC counter as data block of system data type IEC_<Counter> (Shared DB)

You can declare the IEC counter as a data block as follows:

<IEC_Counter_DB>.CTU();

IEC counter as local tag from the block interface (multi-instance)

You can declare the IEC counter as a local tag as follows:

#myLocal_Counter();

For information on calling IEC counters within structures (multi-instance), please refer to: Calling IEC counters (Page 3252)

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
CU	Input	BOOL	I, Q, M, D, L	Count input
R	Input	BOOL	I, Q, M, D, L, P	Reset input
PV	Input	Integers	I, Q, M, D, L, P	Value at which the Q output is set
Q	Output	BOOL	I, Q, M, D, L	Counter status
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	Current counter value

Example

The following example shows how the instruction works:

```
SCL
"IEC_COUNTER_DB".CTU(CU := "Tag_Start",
                     R := "Tag_Reset",
                     PV := "Tag_PresetValue",
                     Q => "Tag_Status",
                     CV => "Tag_CounterValue");
```

When the signal state of the "Tag_Start" operand changes from "0" to "1", the "Count up" instruction is executed and the current counter value of the "Tag_CounterValue" operand is incremented by one. With each additional positive signal edge, the counter value is incremented until the high limit of the specified data type (INT = 32767) is reached.

The "Tag_Status" output has signal state "1" as long as the current counter value is greater than or equal to the value of the "Tag_PresetValue" operand. In all other cases, the "Tag_Status" output has signal state "0". The current counter value is stored in the "Tag_CounterValue" operand.

See also

Overview of the valid data types (Page 2239)

Setting retentivity in an instance data block (Page 4809)

Basics of SCL (Page 4991)

CTD: Count down (S7-1200, S7-1500)

Description

The "Count down" instruction is used to decrement the value at the parameter CV. When the signal state of the CD parameter changes from "0" to "1" (positive signal edge), the instruction is executed and the current counter value of the CV parameter is decremented by one. Each time a positive signal edge is detected, the counter value is decremented until it reaches the low limit of the specified data type. When the low limit is reached, the signal state of the CD parameter no longer has an effect on the instruction.

You can query the count status of the Q parameter. If the current counter value is less than or equal to zero, the Q parameter is set to signal state "1". In all other cases, the signal state of the Q parameter is "0". You can also specify a constant for the PV parameter.

The value of the CV parameter is set to the value of the PV parameter when the signal state of the LD parameter changes to "1". As long as the signal state of the LD parameter is "1", the signal state of the CD parameter has no effect on the instruction.

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count down" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For CPUs of the S7-1200 series

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER	• CTD_SINT / CTD_USINT • CTD_INT / CTD_UINT • CTD_DINT / CTD_UDINT

For CPUs of the S7-1500 series

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER • IEC_COUNTER / IEC_UCOUNTER • IEC_DCOUNTER / IEC_UDCOUNTER • IEC_LCOUNTER / IEC_ULCOUNTER	• CTD_SINT / CTD_USINT • CTD_INT / CTD_UINT • CTD_DINT / CTD_UDINT • CTD_LINT / CTD_ULINT

You can declare an IEC counter as follows:

- Declaration of an instance data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER_DB")
- Declaration as local tag of the data type CTD_<DatenType> in the "Static" section of a program block (for example, #MyIEC_COUNTER_Instance)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive.

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

IEC counter as data block of system data type IEC_<Counter> (Shared DB)

You can declare the IEC counter as a data block as follows:

<IEC_Counter_DB>.CTD();

IEC counter as local tag from the block interface (multi-instance)

You can declare the IEC counter as a local tag as follows:

#myLocal_Counter();

For information on calling IEC counters within structures (multi-instance), please refer to: Calling IEC counters (Page 3252)

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
CD	Input	BOOL	I, Q, M, D, L	Count input
LD	Input	BOOL	I, Q, M, D, L, P	Load input
PV	Input	Integers	I, Q, M, D, L, P	Value to which the CV output is set with LD = 1.
Q	Output	BOOL	I, Q, M, D, L	Counter status
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	Current counter value

Example

The following example shows how the instruction works:

```
SCL
"IEC_SCOUNTER_DB".CTD(CD := "Tag_Start",
                      LD := "Tag_Load",
                      PV := "Tag_PresetValue",
                      Q => "Tag_Status",
                      CV => "Tag_CounterValue");
```

When the signal state of the "Tag_Start" operand changes from "0" to "1", the instruction is executed and the value of the "Tag_CounterValue" operand is decremented by one. With each additional positive signal edge, the counter value is decremented until it reaches the low limit of the specified data type (-128).

The operand "Tag_Status" has signal state "1" as long as the current counter value is less than or equal to zero. In all other cases, the "Tag_Status" output has signal state "0". The current counter value is stored in the "Tag_CounterValue" operand.

See also

Overview of the valid data types (Page 2239)

Setting retentivity in an instance data block (Page 4809)

Basics of SCL (Page 4991)

CTUD: Count up and down (S7-1200, S7-1500)

Description

Use the "Count up and down" instruction to increment or decrement the counter value at the CV parameter. When the signal state of the CU parameter changes from "0" to "1" (positive signal edge), the current counter value of the CV parameter is incremented by one. When the signal state of the CD parameter changes from "0" to "1" (positive signal edge), the counter value of the CV parameter is decremented by one. If there is a positive signal edge at the CU and CD inputs in a program cycle, the current counter value of the CV parameter remains unchanged.

The counter value can be incremented until it reaches the high limit of the data type specified at the CV parameter. When the high limit is reached, the counter value is no longer incremented on a positive signal edge. The counter value is no longer decremented once the low limit of the specified data type has been reached.

When the signal state of the LD parameter changes to "1", the counter value of the CV parameter is set to the value of the PV parameter. As long as the LD parameter has signal state "1", the signal state of the CU and CD parameters has no effect on the instruction.

The counter value is set to zero when the signal state of the R parameter changes to "1". As long as the R parameter has signal state "1", a change in the signal state of the CU, CD and LD parameters has no effect on the "Count up and down" instruction.

You can query the status of the up counter at the QU parameter. When the current counter value is greater than or equal to the value of the PV parameter, the QU parameter is set to

signal state "1". In all other cases, the signal state of the QU parameter is "0". You can also specify a constant for the PV parameter.

You can query the status of the down counter at the QD parameter. If the current counter value is less than or equal to zero, the QD parameter is set to signal state "1". In all other cases, the signal state of the QD parameter is "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Each call of the "Count up and down" instruction must be assigned an IEC counter in which the instruction data is stored. An IEC counter is a structure with one of the following data types:

For CPUs of the S7-1200 series

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER	• CTUD_SINT / CTUD_USINT
• IEC_COUNTER / IEC_UCOUNTER	• CTUD_INT / CTUD_UINT
• IEC_DCOUNTER / IEC_UDCOUNTER	• CTUD_DINT / CTUD_UDINT

For CPUs of the S7-1500 series

Data block of system data type IEC_<Counter> (Shared DB)	Local tag
• IEC_SCOUNTER / IEC_USCOUNTER	• CTUD_SINT / CTUD_USINT
• IEC_COUNTER / IEC_UCOUNTER	• CTUD_INT / CTUD_UINT
• IEC_DCOUNTER / IEC_UDCOUNTER	• CTUD_DINT / CTUD_UDINT
• IEC_LCOUNTER / IEC_ULCOUNTER	• CTUD_LINT / CTUD_ULINT

You can declare an IEC counter as follows:

- Declaration of an instance data block of system data type IEC_<Counter> (for example, "MyIEC_COUNTER_DB")
- Declaration as local tag of the data type CTUD_<DatenType> in the "Static" section of a program block (for example, #MyCTUD_COUNTER_Instance)

When you set up the IEC counter in a separate data block (single instance), the instance data block is created by default with "optimized block access" and the individual tags are defined as retentive.

When you set up the IEC counter as local tag (multi-instance) in a function block with "optimized block access", it is defined as retentive in the block interface.

IEC counter as data block of system data type IEC_<Counter> (Shared DB)

You can declare the IEC counter as a data block as follows:

<IEC_Counter_DB>.CTUD();

IEC counter as local tag from the block interface (multi-instance)

You can declare the IEC counter as a local tag as follows:

#myLocal_Counter();

For information on calling IEC counters within structures (multi-instance), please refer to: Calling IEC counters (Page 3252)

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
CU	Input	BOOL	I, Q, M, D, L	Count up input
CD	Input	BOOL	I, Q, M, D, L	Count down input
R	Input	BOOL	I, Q, M, D, L, P	Reset input
LD	Input	BOOL	I, Q, M, D, L, P	Load input
PV	Input	Integers	I, Q, M, D, L, P	Value at which the QU output is set / value to which the CV output is set when LD = 1.
QU	Output	BOOL	I, Q, M, D, L	Status of the up counter
QD	Output	BOOL	I, Q, M, D, L	Status of the down counter
CV	Output	Integers, CHAR, WCHAR, DATE	I, Q, M, D, L, P	Current counter value

Example

The following example shows how the instruction works:

```
SCL
"IEC_COUNTER_DB".CTUD(CU := "Tag_Start1",
                      CD := "Tag_Start2",
                      LD := "Tag_Load",
                      R := "Tag_Reset",
                      PV := "Tag_PresetValue",
                      QU => "Tag_CU_Status",
                      QD => "Tag_CD_Status",
                      CV => "Tag_CounterValue");
```

If the "Tag_Start1" operand has a positive signal edge in the signal state, the current counter value is incremented by one and stored in the "Tag_CounterValue" operand. If the "Tag_Start2" operand has a positive signal edge in the signal state, the counter value is decremented by one and is also stored in the "Tag_CounterValue" operand. The counter value is incremented on the positive signal edge of the CU parameter until it reaches the high limit of the specified data type (INT). If the CD parameter has a positive signal edge, the counter value is decremented until it reaches the low limit of the specified data type (INT).

The operand "Tag_CU_Status" has signal state "1" as long as the current counter value is greater than or equal to the value of the operand "Tag_PresetValue". In all other cases, the "Tag_CU_Status" output has signal state "0".

The operand "Tag_CD_Status" has signal state "1" as long as the current counter value is less than or equal to zero. In all other cases, the "Tag_CD_Status" output has signal state "0".

See also

Overview of the valid data types (Page 2239)

Setting retentivity in an instance data block (Page 4809)

Basics of SCL (Page 4991)

Legacy (S7-1500)

S_CU: Assign parameters and count up (S7-1500)

Description

You can use the "Assign parameters and count up" instruction to increment the value of a counter. When the signal state of the CU parameter changes from "0" to "1" (positive signal edge), the current counter value is incremented by one. The current counter value is provided at the parameter CV. The counter value is incremented until the limit of "999" is reached. When the limit value is reached, the counter value is no longer incremented on a positive signal edge.

When the signal state of the S parameter changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if the result of logic operation (RLO) at the CU input is "1", the counter counts once in the next cycle, even when no signal edge change is detected.

The counter value is set to zero when the signal state of the R parameter changes to "1". As long as the R parameter has the signal state "1", a change in the signal state of the parameters CU and S has no effect on the counter value.

The signal state at parameter Q is "1" if the counter value is greater than zero. When the counter value equals zero, parameter Q returns signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
C_NO	Input	COUNTER, INT	C	Counter operations The number of counters depends on the CPU.
CU	Input	BOOL	I, Q, M, D, L	Count up input
S	Input	BOOL	I, Q, M, D, L	Input for presetting the counter

Parameters	Declaration	Data type	Memory area	Description
PV	Input	WORD	I, Q, M, D, L	Preset counter value (C#0 to C#999) in BCD format
R	Input	BOOL	I, Q, M, D, L	Reset input
Q	Output	BOOL	I, Q, M, D, L	Status of the counter
CV	Output	WORD	I, Q, M, D, L	Current counter value
Function value		WORD	I, Q, M, D, L	Current counter value in BCD format

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := S_CU(C_NO := "Counter_1",
                     CU := "Tag_Start",
                     S := "Tag_1",
                     PV := "Tag_PresetValue",
                     R := "Tag_Reset",
                     Q => "Tag_Status",
                     CV => "Tag_Value");
```

If the signal state of the "Tag_Start" parameter changes from "0" to "1" (positive signal edge) and the current counter value is less than "999", the counter value is incremented by one. If the signal state of the "Tag_1" input changes from "0" to "1", the counter value in BCD format is set to the value of the "Tag_PresetValue" operand. The counter value is reset to "0" when the "Tag_Reset" operand has signal state "1".

The current counter value is stored in hexadecimal form in the operand "Tag_Value".

The "Tag_Status" output has the signal state "1" as long as the current counter value is not equal to "0". The current counter value is returned in the "Tag_Value" operand and as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

S_CD: Assign parameters and count down (S7-1500)

Description

You can use the "Assign parameters and count down" instruction to decrement the value of a counter. When the signal state of the CD parameter changes from "0" to "1" (positive signal edge), the current counter value is decreased by one. The current counter value is provided at the parameter CV. The counter value is decreased until the low limit of "0" is reached. When the low limit is reached, the counter value is no longer decreased on a positive signal edge.

When the signal state of the S parameter changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if the result of logic operation (RLO) at the CD parameter is "1", the counter counts once in the next cycle, even when no signal edge change is detected.

The counter value is set to zero when the signal state of the R parameter changes to "1". As long as the R parameter has the signal state "1", a change in the signal state of the parameters CD and S has no effect on the counter value.

The signal state at parameter Q is "1" if the counter value is greater than zero. When the counter value equals zero, parameter Q returns signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
C_NO	Input	COUNTER, INT	C	Counter operations The number of counters depends on the CPU.
CD	Input	BOOL	I, Q, M, D, L	Count down input
S	Input	BOOL	I, Q, M, D, L	Input for presetting the counter
PV	Input	WORD	I, Q, M, D, L	Preset counter value (C#0 to C#999) in BCD format
R	Input	BOOL	I, Q, M, D, L	Reset input
Q	Output	BOOL	I, Q, M, D, L	Status of the counter
CV	Output	WORD	I, Q, M, D, L	Current counter value
Function value		WORD	I, Q, M, D, L	Current counter value in BCD format

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := S_CD(C_NO := "Counter_1",
                     CD := "Tag_Start",
                     S := "Tag_1",
                     PV := "Tag_PresetValue",
                     R := "Tag_Reset",
                     Q => "Tag_Status",
                     CV => "Tag_Value");
```

When the signal state of the "Tag_Start" operand changes from "0" to "1" (positive signal edge) and the current counter value is greater than "0", the counter value is decreased by one. If the signal state of the "Tag_1" operand changes from "0" to "1", the counter value in BCD format is set to the value of the "Tag_PresetValue" operand. The counter value is reset to "0" when the "Tag_Reset" operand has signal state "1".

The current counter value is stored in the "Tag_Value" operand.

The "Tag_Status" operand returns signal state "1" as long as the current counter value is not equal to "0". The current counter value is returned in the "Tag_Value" operand and as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

S_CUD: Assign parameters and count up / down (S7-1500)

Description

You can use the "Assign parameters and count up / down" instruction to increment and decrement the value of a counter. When the signal state of the CU parameter changes from "0" to "1" (positive signal edge), the current counter value is incremented by one. When the signal state of the CD parameter changes from "0" to "1" (positive signal edge), the counter value is decreased by one. The current counter value is provided at the parameter CV. If there is a positive signal edge at the parameters CU and CD in one program cycle, the counter value remains unchanged.

The counter value is incremented until the high limit of "999" is reached. When the high limit is reached, the counter value is no longer incremented on a positive signal edge. When the low limit "0" is reached, the counter value is not decremented any further.

When the signal state of the S parameter changes from "0" to "1", the counter value is set to the value of the PV parameter. If the counter is set and if the result of logic operation (RLO) of the CU and CD parameters is "1", the counter will count once in the next cycle, even if no signal edge change was detected.

The counter value is set to zero when the signal state of the R parameter changes to "1". As long as the parameter R has the signal state "1", processing of the signal state of the parameters CU, CD and S has no effect on the counter value.

The signal state at parameter Q is "1" if the counter value is greater than zero. When the counter value equals zero, parameter Q returns signal state "0".

Note

Only use a counter at a single point in the program to avoid the risk of counting errors.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
C_NO	Input	COUNTER, INT	C	Counter operations The number of counters depends on the CPU.
CU	Input	BOOL	I, Q, M, D, L	Count up input
CD	Input	BOOL	I, Q, M, D, L	Count down input
S	Input	BOOL	I, Q, M, D, L	Input for presetting the counter
PV	Input	WORD	I, Q, M, D, L	Preset counter value (C#0 to C#999) in BCD format
R	Input	BOOL	I, Q, M, D, L	Reset input
Q	Output	BOOL	I, Q, M, D, L	Status of the counter
CV	Output	WORD	I, Q, M, D, L	Current counter value (hexadecimal)
Function value		WORD	I, Q, M, D, L	Current counter value in BCD format

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := S_CD(C_NO := "Counter_1",
                     CU := "Tag_CU",
                     CD := "Tag_CD",
                     S := "Tag_1",
                     PV := "Tag_PresetValue",
                     R := "Tag_Reset",
                     Q => "Tag_Status",
                     CV => "Tag_Value");
```

When a positive signal edge is detected in the signal state of the "Tag_CU" operand and the current counter value is less than "999", the counter value is incremented by one. When a positive signal edge is detected in the signal state of the "Tag_CD" operand and the current counter value is greater than "0", the counter value is decremented by one.

If the signal state of the "Tag_1" operand changes from "0" to "1", the counter value in BCD format is set to the value of the "Tag_PresetValue" operand. The counter value is reset to "0" when the "Tag_Reset" operand has signal state "1".

The current counter value is stored in the "Tag_Value" operand.

The "Tag_Status" operand returns signal state "1" as long as the current counter value is not equal to "0". The current counter value is returned in the "Tag_Value" operand and as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Comparator operations (S7-1200, S7-1500)

TypeOf: Check data type of a VARIANT tag (S7-1200, S7-1500)

Description

You can use the "Check data type of a VARIANT tag" instruction to query the data type of a tag to which a VARIANT tag is pointing. You can compare the data type of the that you have declared in the block interface either to the data type of another tag or directly with a data type to determine whether they are "Equal" or "Not equal".

The comparison operand can be an elementary data type or a PLC data type.

You can only use the "Check data type of a VARIANT tag" instruction within an IF or CASE instruction.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	Binary numbers, integers, floating-point numbers, times, date and time, character strings, VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand

For additional information on valid data types, refer to "See also".

Example

The following example shows the comparison with another tag:

SCL

```
IF TypeOf(#TagVARIANT) = TypeOf("TagBYTE") THEN
...;
END_IF;
```

The following example shows the comparison with a data type:

SCL

```
IF TypeOf(#TagVARIANT) = BYTE THEN
...;
END_IF;
```

If the operand to which the VARIANT #TagVARIANT is pointing has the BYTE data type, the comparison condition has been met.

See also

Overview of the valid data types (Page 2239)

Example of programming a queue (FIFO) (Page 2320)

Basic information on VARIANT (Page 2308)

Basics of SCL (Page 4991)

TypeOfElements: Check data type of an ARRAY element of a VARIANT tag (S7-1200, S7-1500)

Description

You can use the "Check data type of an ARRAY element of a VARIANT tag" instruction to query the data type of a tag to which a VARIANT tag is pointing. You compare the data type of a tag to the data type of the tag that you have declared in the block interface to determine whether they are "Equal" or "Not equal".

The operand must have the VARIANT data type. The comparison operand can be an elementary data type or a PLC data type.

If the data type of the VARIANT tag is an ARRAY, the data type of the ARRAY elements is compared.

You can only use the instruction within an IF instruction.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	First operand

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
IF TypeOfElements(#Tag_VARIANT) = TypeOf("GlobalDB".Product[1]) THEN
"Tag_Result" := "GlobalDB".Product[1] * 3;
END_IF;
```

The following table shows how the instruction works using specific operand values:

Operand	Value
#Tag_VARIANT	1.5
"GlobalDB".Product[1]	4.5

If the tag to which the VARIANT points and the "GlobalDB".Product[1] operand have the REAL data type, the "GlobalDB".Product[1] operand is then multiplied by 3 and the result is written to the "Tag_Result" operand.

See also

Overview of the valid data types (Page 2239)

Example of programming a queue (FIFO) (Page 2320)

Basic information on VARIANT (Page 2308)

Basics of SCL (Page 4991)

IS_ARRAY: Check for ARRAY (S7-1200, S7-1500)

Description

You can use the "Check for ARRAY" instruction to query whether the VARIANT points to a tag of the ARRAY data type.

You can only use the "Check for ARRAY" instruction within an IF instruction.

Parameters

The following table shows the parameters of the "Check for ARRAY" instruction:

Parameters	Declaration	Data type	Memory area	Description
<operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Operand that is queried for ARRAY
Function value		UDINT	I, Q, M, D, L	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
IF IS_ARRAY(#Tag_VARIANTToArray) THEN
"Tag_Result" := CountOfElements(#Tag_VARIANTToArray);
END_IF;
```

If the tag to which the VARIANT points is an ARRAY, the number of ARRAY elements is output.

See also

Overview of the valid data types (Page 2239)

Example of programming a queue (FIFO) (Page 2320)

Basic information on VARIANT (Page 2308)

Basics of SCL (Page 4991)

Math functions (S7-1200, S7-1500)

ABS: Form absolute value (S7-1200, S7-1500)

Description

Use the "Form absolute value" instruction to calculate the absolute value of an input value and to save the result in the specified operands.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<expression>	Input	SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P	Input value
Function value		SINT, INT, DINT, floating-point numbers	SINT, INT, DINT, LINT, floating-point numbers	I, Q, M, D, L, P	Absolute value of the input value

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result1" := ABS("Tag_Value");
"Tag_Result2" := ABS("Tag_Value1"*"Tag_Value2");
```

The absolute value of the input value is returned in the format of the input value as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	-2
Tag_Result1	2
Tag_Value1	4
Tag_Value2	-1
Tag_Result2	4

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

MIN: Get minimum (S7-1200, S7-1500)

Description

The "Get minimum" instruction compares the values of the available inputs and returns the lowest value as the result.

A minimum of two and a maximum of 32 inputs can be specified at the instruction.

The result is invalid if any of the following conditions are met:

- The implicit conversion of the data types fails during execution of the instruction.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Get minimum" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN1	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	First input value
IN2	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Second input value
INn	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Additionally inserted inputs whose values are to be compared
Function value		Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Result of the instruction
The data types TOD, LTOD, DATE, and LDT can only be used if the IEC test is not enabled.					

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := MIN(IN1 := "Tag_Value1",
                    IN2 := "Tag_Value2",
                    IN3 := "Tag_Value3");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN1	Tag_Value1	12222
IN2	Tag_Value2	14444
IN3	Tag_Value3	13333
Function value	Tag_Result	12222

The instruction compares the values of the available inputs and copies the lowest value ("Tag_Value1") to operand "Tag_Result".

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

MAX: Get maximum (S7-1200, S7-1500)

Description

The "Get maximum" instruction compares the values of the available inputs and returns the greatest value as the result.

A minimum of two and a maximum of 32 inputs can be specified at the instruction.

The result is invalid if any of the following conditions are met:

- The implicit conversion of the data types fails during execution of the instruction.
- A floating-point number has an invalid value.

Parameters

The following table shows the parameters of the "Get maximum" instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN1	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	First input value
IN2	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Second input value
INn	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Additionally inserted inputs whose values are to be compared
Function value		Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Result of the instruction
The data types TOD, LTOD, DATE, and LDT can only be used if the IEC test is not enabled.					

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := MAX(IN1 := "Tag_Value1",
                    IN2 := "Tag_Value2",
                    IN3 := "Tag_Value3");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN1	Tag_Value1	12 222
IN2	Tag_Value2	14 444
IN3	Tag_Value3	13 333
Function value	Tag_Result	14 444

The instruction compares the values of the specified operands and copies the greatest value ("Tag_Value2") to the operand "Tag_Result".

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

LIMIT: Set limit value (S7-1200, S7-1500)

Description

The "Set limit value" instruction limits the value of the parameter IN to the values of the parameters MN and MX. The value of the parameter MN may not be greater than the value of the parameter MX.

If the value of the IN parameter fulfills the MN condition <=IN <= MX, it is returned as the result of the instruction. If the condition is not fulfilled and the IN input value is less than the MN low limit, the value of the MN parameter is returned as the result. If the high limit MX is exceeded, the value of the MX parameter is returned as the result.

If the value at the MN input is greater than at the MX input, the result is the value specified at the IN parameter and the enable output ENO is "0".

The instruction is only executed if the operands of all parameters are of the same data type.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
MN	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Low limit
IN	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Input value
MX	Input	Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	High limit
Function value		Integers, floating-point numbers, TIME, TOD, DATE, DTL	Integers, floating-point numbers, TIME, LTIME, TOD, LTOD, DATE, LDT, DT, DTL	I, Q, M, D, L, P	Result of the instruction
The data types TOD, LTOD, DATE, and LDT can only be used if the IEC test is not enabled.					

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := LIMIT(MN := "Tag_Minimum",
                      IN := "Tag_Value",
                      MX := "Tag_Maximum");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
MN	Tag_Minimum	12 000
IN	Tag_Value	8 000
MX	Tag_Maximum	16 000
Function value	Tag_Result	12 000

The value of operand "Tag_Value" is compared with the values of operands "Tag_Minimum" and "Tag_Maximum". Because the value of the operand "Tag_Value" is less than the low limit value, the value of operand "Tag_Minimum" will be copied to operand "Tag_Result".

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SQR: Form square (S7-1200, S7-1500)

Description

Use the "Form square" instruction to square the input value and save the result in the specified operand.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value
Function value		Floating-point numbers	I, Q, M, D, L, P	Square of the input value

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := SQR("Tag_Value");
"Tag_Result2" := SQR((SQR("Tag_Value1"))*"Tag_Value2");
```

The square of the input value is returned in the operand "Tag_Resultxy" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	2.5
Tag_Result1	6.25
Tag_Value1	6.0
Tag_Value2	2.0
Tag_Result2	5184.0

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SQRT: Form square root (S7-1200, S7-1500)

Description

Use the "Form square root" instruction to calculate the square root of the input value and save the result in the specified operand. The instruction has a positive result if the input value is greater than zero. If input values are less than zero, the instruction returns an invalid floating-point number. If the input value is "0", the result is also "0".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value
Function value		Floating-point numbers	I, Q, M, D, L, P	Square root of the input value

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := SQRT("Tag_Value");
"Tag_Result2" := SQRT((SQR("Tag_Value1"))+"Tag_Value2");
```

The square root of the input value is returned in the operand "Tag_Resultxy" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	4.0
Tag_Result1	2.0
Tag_Value1	3.0
Tag_Value2	16.0
Tag_Result2	5.0

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

LN: Form natural logarithm (S7-1200, S7-1500)

Description

You can use the "Form natural logarithm" instruction to calculate the natural logarithm to base e (e=2.718282) from the input value. The instruction has a positive result if the input value is greater than zero. If input values are less than zero, the instruction returns an invalid floating-point number.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expres-sion>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value
Function value		Floating-point numbers	I, Q, M, D, L, P	Natural logarithm of the input value

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := LN("Tag_Value");
"Tag_Result2" := LN("Tag_Value1"+"Tag_Value2");
```

The result of the instruction is returned in the operand "Tag_Resultxy" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	2.5
Tag_Result1	0.916
Tag_Value1	1.5
Tag_Value2	3.2
Tag_Result2	1.548

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

EXP: Form exponential value (S7-1200, S7-1500)

Description

The "Form exponential value" instruction calculates the exponent from the base e (e = 2.718282) and the input value and saves the result in the specified operand.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value
Function value		Floating-point numbers	I, Q, M, D, L, P	Exponential value of the input value

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := EXP("Tag_Value");
"Tag_Result2" := EXP("Tag_Value1"/"Tag_Value2");
```

The result of the instruction is returned in the operand "Tag_Resultxy" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	20.5
Tag_Result1	799 902 200

Operand	Value
Tag_Value1	15.5
Tag_Value2	30.2
Tag_Result2	1.671

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SIN: Form sine value (S7-1200, S7-1500)

Description

Use the "Form sine value" instruction to calculate the sine of the input value. The input value must be given in radians.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value (size of an angle in radians)
Function value		Floating-point numbers	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := SIN("Tag_Value");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	+1.570796 (π/2)
Tag_Result	1.0

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

COS: Form cosine value (S7-1200, S7-1500)

Description

Use the "Form cosine value" instruction to calculate the cosine of the input value. The input value must be given in radians.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value (size of an angle in radians)
Function value		Floating-point numbers	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := COS("Tag_Value");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	+1.570796 (π/2)
Tag_Result	0

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

TAN: Form tangent value (S7-1200, S7-1500)

Description

Use the "Form tangent value" instruction to calculate the sine of the input value. The input value must be given in radians.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expres-sion>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value (size of an angle in radians)
Function value		Floating-point numbers	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := TAN("Tag_Value");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	+3.141593 (π)
Tag_Result	0

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ASIN: Form arcsine value (S7-1200, S7-1500)

Description

You can use the "Form arcsine value" instruction to calculate the size of the angle from a sine value, which corresponds to this value. Only valid floating-point numbers within the range -1 to +1 can be specified as input values. The calculated angle size is given in radians and can range in value from -π/2 to +π/2.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expres-sion>	Input	Floating-point numbers	I, Q, M, D, L, P	Sine value
Function value		Floating-point numbers	I, Q, M, D, L, P	Size of angle in radians

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := ASIN("Tag_Value");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	1.0
Tag_Result	+1.570796 (π/2)

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ACOS: Form arccosine value (S7-1200, S7-1500)

Description

You can use the "Form arccosine value" instruction to calculate the size of the angle from a cosine value, which corresponds to this value. Only valid floating-point numbers within the range -1 to +1 can be specified as input values. The calculated angle size is given in radians and can range in value from 0 to +π.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expres-sion>	Input	Floating-point numbers	I, Q, M, D, L, P	Cosine value
Function value		Floating-point numbers	I, Q, M, D, L, P	Size of angle in radians

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := ACOS("Tag_Value");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	0
Tag_Result	+1.570796 (π/2)

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ATAN: Form arctangent value (S7-1200, S7-1500)

Description

You can use the "Form arctangent value" instruction to calculate the size of the angle from a tangent value, which corresponds to this value. It is only permitted to specify valid floating-point numbers (or -NaN/+NaN) as input value. The calculated angle size is given in radians and can range in value from -π/2 to +π/2.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expres-sion>	Input	Floating-point numbers	I, Q, M, D, L, P	Tangent value
Function value		Floating-point numbers	I, Q, M, D, L, P	Size of angle in radians

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := ATAN("Tag_Value");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	1.0
Tag_Result	+0.785398 (π/4)

See also

Overview of the valid data types (Page 2239)

Invalid floating-point numbers (Page 2259)

Basics of SCL (Page 4991)

FRAC: Return fraction (S7-1200, S7-1500)

Description

The result of the instruction "Return fraction" returns the decimal places of a value. Input value 123.4567, for example, returns the value 0.4567.

Use the following syntax to change the data type of the instruction:

FRAC_<data type>();

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value
_<Data type>		Floating-point numbers Default: REAL	-	Data type of the function value: 1. You can specify the data type of the instruction explicitly using "_". 2. If you do not specify the data type explicitly, it will be determined by the utilized tags or type-coded constants. 3. If you neither specify the data type explicitly nor specify defined tags or type-coded constants, the default data type will be used.
Function value		Floating-point numbers	I, Q, M, D, L, P	Decimal places of the input value

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := FRAC("Tag_Value");
"Tag_Result2" := FRAC_LREAL("Tag_Value");
```

The following table shows how the instruction works using specific operand values:

Operand	Value	
Tag_Value	2.555	-1.4421
Tag_Result1	0.555	-0.4421

See also

Overview of the valid data types (Page 2239)

Invalid floating-point numbers (Page 2259)

Basics of SCL (Page 4991)

Move operations (S7-1200, S7-1500)

Deserialize: Deserialize (S7-1200, S7-1500)

Description

You can use the "Deserialize" instruction to convert back the sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type> and to fill its entire contents.

The memory area in which the sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type> is located must have the ARRAY of BYTE data type and be declared with standard access in version 1.0. Optimized memory areas also permitted as of version 2.0. The capacity of the standard memory area is 64 KB. Make sure that there is enough memory space prior to the conversion. If the memory area is filled using the instruction "Serialize" and it is possible that fill bytes were inserted, their conversion is not taken into account.

It is recommended to define the low limit of the ARRAY with "0", because then the index within ARRAY corresponds to the value of the POS parameter, e.g. ARRAY[0] = POS 0. The description and example below have been formed based on this.

The instruction enables you to convert multiple sequential representations of converted data back to their original state step-by-step.

If you only want to convert back a single sequential representation of a PLC data type (UDT), STRUCT or ARRAY of <data type>, you can also directly use the instruction "TRCV: Receive data via communication connection".

Size of the memory area

The alignment rules mean that simple structures in the optimized memory area do not contain filling bytes. This results in a structure in the optimized memory area being smaller than in the standard memory area. ARRAYs of structures and structures in which other structures are located contain filling bytes. There is no general rule as to the memory area in which a composed structure requires more space.

Applies to CPUs of the S7-1500 series:

For a block with the block property "Optimized block access", the BOOL has a length of 1 byte. A structure that consists largely of the data type BOOL can therefore be larger in the optimized memory area than in the standard memory area. Composed structures with a low proportion of BOOL data types are smaller in the optimized memory area than in the standard memory area.

Optimized memory area

To deserialize larger structures, you can declare the memory area for sequential representation with optimized access for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0. The content of the sequential representation remains unchanged, as for a standard memory area. Only symbolic access to the bytes in the ARRAY is possible.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
SRC_ARRAY	Input	VARIANT ARRAY[*] of BYTE [1]	D, block interface of an FB (The sections Input, Output, Static and Temp are possible.)	Tag of ARRAY of BYTE data type in which the generate data flow is stored
DEST_VARIABLE	InOut	VARIANT	D, block interface of an FB	Tag in which the reconverted PLC data type (UDT), STRUCT, ARRAY of <data type> is stored
POS	InOut	DINT	I, Q, M, D, L	The operand at the POS parameter stores the index of the first byte based on the number of bytes occupied by the converted customer data. The POS parameter is calculated zero-based.
Function value		INT	I, Q, M, D, L	Error information
[1] Possible for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.				

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B0	The memory areas for the SRC_ARRAY and DEST_VARIABLE parameters overlap.
8136	The tag at the SRC_ARRAY parameter is not in a block with standard access.
8150	The VARIANT data type at the SRC_ARRAY parameter contains a ZERO pointer.
8151	No valid reference at the SRC_ARRAY parameter
8153	There is not enough free memory available at the SRC_ARRAY parameter.
8154	Invalid data type at the SRC_ARRAY parameter
8250	The VARIANT data type at the DEST_VARIABLE parameter contains a ZERO pointer.
8251	No valid reference at the DEST_VARIABLE parameter
8382	The value at the POS parameter is outside the limits of the array.
9512	For a CPU of the S7-1200 module and a block with the block property "Optimized block access": The BOOL data type is used at the DEST_VARIABLE parameter. This makes the object smaller than a full byte.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

For a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0, the following error code has a different meaning:

Error code* (W#16#...)	Explanation
8136	Access to the memory area at the SRC_ARRAY parameter is invalid.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following table shows the declaration of the operands:

Operand	Data type	Declaration
DeliverPos	INT	In the "Input" section of the block interface of an FB or FC.
BufferPos	DINT	In the "Temp" section of the block interface of an FB or FC.
Error	INT	
Label	STRING[4]	

The following table shows the declarations of the PLC data types:

Name of the PLC data types	Name	Data type
Article	Number	DINT
	Declaration	STRING
	Colli	INT
Client	Title	INT
	First name	STRING[10]
	Surname	STRING[10]

The following table shows the declaration of the data blocks:

Name of the data blocks	Name	Data type
Target	Client	"Client" (PLC data type)
	Article	Array[0..10] of "Article" (PLC data type)
	Bill	Array[0..10] of INT
Buffer	Field	Array[0..294] of BYTE

The following example shows how the instruction works:

SCL

```
#Tag_RetVal := Deserialize(SRC_ARRAY := "Buffer".Field,
                           DEST_VARIABLE := "Target".Client,
                           POS := #BufferPos);

#Tag_RetVal := Deserialize(SRC_ARRAY := "Buffer".Field,
                           DEST_VARIABLE := #Label,
                           POS := #BufferPos);
```

SCL

```
IF #Label = 'arti' THEN
#Tag_RetVal := Deserialize(SRC_ARRAY := "Buffer".Field,
                           DEST_VARIABLE := "Target".Article[#DeliverPos],
                           POS := #BufferPos);
ELSIF #Label = 'Bill' THEN
#Tag_RetVal := Deserialize(SRC_ARRAY := "Buffer".Field,
                           DEST_VARIABLE := "Target".Bill[#DeliverPos],
                           POS := #BufferPos);
;
ELSE
;
END_IF;
```

The "Deserialize" instruction deserializes the sequential representation of the customer data from the "Buffer" tag and writes it to the "Target" tag. The #BufferPos operand stores the index of the first byte based on the number of bytes occupied by the converted customer data.

The "Deserialize" instruction deserializes the sequential representation of the separator sheet, which was stored in the sequential representation after the customer data, from the "Buffer" tag and writes the characters to the #Label operand. The characters are compared using comparison instructions "arti" and "Bill". If the comparison for "arti" = TRUE, these are article data that have been deserialized and written to the "Target" tag. If the comparison for "Bill" = TRUE, these are billing data that have been deserialized and written to the "Target" tag.

See also

Overview of the valid data types (Page 2239)

Basics of PLC data types (UDT) (Page 2277)

Structure of an ARRAY tag (Page 2290)

Structure of a STRUCT tag (Page 2284)

Structure of a STRING tag (Page 2272)

Basics of SCL (Page 4991)

Serialize: Serialize (S7-1200, S7-1500)

Description

You can use the "Serialize" instruction to convert several PLC data types (UDT), STRUCT or ARRAY of <data type> to a sequential representation without losing parts of their structure.

You use the instruction to temporarily save multiple structured data items from your program in a buffer, which should preferably be in a global data block, and send them to another CPU. The memory area in which the converted data are stored must have the ARRAY of BYTE data type and be declared with standard access in version 1.0. Optimized data are also permitted as of version 2.0. Fill bytes or fill bits of a data area (e.g. ARRAY, STRUCT or PLC data type

(UDT)) as well as the characters of a string currently not in use are undefined in the destination ARRAY.

The capacity of the standard memory area is 64 KB. Structures that are larger than 64 KB in accordance with the rules for standard memory areas cannot be serialized if the operand at the DEST_ARRAY parameter is located in a standard memory area.

It is recommended to define the low limit of the ARRAY with "0", because then the index within ARRAY corresponds to the value of the POS parameter, e.g. ARRAY[0] = POS 0. The description and example below have been formed based on this.

The operand at the POS parameter contains information about the number of bytes used by the converted data.

If you want to send a single PLC data type (UDT), STRUCT or ARRAY of <data type>, you can directly call the instruction "TSEND: Send data via communication connection".

Note

Comparison of structures

To compare structures, they have to be serialized first. Use a comparison expression instead.

You can find additional information here: Relational expressions (Page 4995)

Size of the memory area

The alignment rules mean that simple structures in the optimized memory area do not contain filling bytes. This results in a structure in the optimized memory area being smaller than in the standard memory area. ARRAYs of structures and structures in which other structures are located contain filling bytes. There is no general rule as to the memory area in which a composed structure requires more space.

Applies to CPUs of the S7-1500 series:

For a block with the block property "Optimized block access", the BOOL has a length of 1 byte. A structure that consists largely of the data type BOOL can therefore be larger in the optimized memory area than in the standard memory area. Composed structures with a low proportion of BOOL data types are smaller in the optimized memory area than in the standard memory area.

Optimized memory area

To serialize larger structures, you can declare the memory area with optimized access for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0. The sequential representation remains unchanged, as for a standard memory area.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
SRC_VARIABLE	Input	VARIANT	D, block interface of an FB	PLC data type (UDT), STRUCT or ARRAY of <data type> that is converted to sequential representation.
DEST_ARRAY	InOut	VARIANT	D, block interface of an FB (The sections Input, Output, Static and Temp are possible.)	Data block in which the generated data stream is stored.
POS	InOut	DINT	I, Q, M, D, L	The operand at the POS parameter stores the index of the first byte based on the total number of bytes that the converted customer data has occupied. The POS parameter is calculated zero-based.
Function value		INT	I, Q, M, D, L	Error information

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B0	The memory areas for the SRC_VARIABLE and DEST_ARRAY parameters overlap.
8150	The VARIANT data type at the SRC_VARIABLE parameter contains a ZERO pointer.
8151	No valid reference at the SRC_VARIABLE parameter
8236	The tag at the SRC_ARRAY parameter is not in a block with standard access.
8250	The VARIANT data type at the DEST_ARRAY parameter contains a ZERO pointer.
8251	No valid reference at the DEST_ARRAY parameter
8253	There is not enough free memory available at the DEST_ARRAY parameter.
8254	Invalid data type at the DEST_ARRAY parameter
8382	The value at the POS parameter is outside the limits of the array.
9512	For a CPU of the S7-1200 module and a block with the block property "Optimized block access": The BOOL data type is used at the DEST_VARIABLE parameter. This makes the object smaller than a full byte.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

For a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0, the following error code has a different meaning:

Error code* (W#16#...)	Explanation
8236	Access to the memory area at the DEST_ARRAY parameter is invalid.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
#Tag_RetVal := Serialize(SRC_VARIABLE := "Source".Client,
                         DEST_ARRAY := "Buffer".Field,
                         POS := #BufferPos);

#Label := STRING_TO_WSTRING('arti');

#Tag_RetVal := Serialize(SRC_VARIABLE := #Label,
                         DEST_ARRAY := "Buffer".Field,
                         POS := #BufferPos);

#Tag_RetVal := Serialize(SRC_VARIABLE := "Source".Article[#DeliverPos],
                         DEST_ARRAY := "Buffer".Field,
                         POS := #BufferPos);
```

The "Serialize" instruction serializes the customer data from the "Source" tag and writes the sequential representation to the "Buffer" tag. The index of the next unwritten byte of the "Buffer".Field operand is saved in the #BufferPos operand.

A kind of separator sheet is inserted now to make it easier to deserialize the sequential representation later. The "Move character string" instruction moves the "arti" characters to the #Label operand. The "Serialize" instruction writes these characters after the customer data to the "Buffer" tag. The value of the #BufferPos operand is incremented accordingly.

The "Serialize" instruction serializes the data of a specific article, which is calculated in runtime, from the "Source" tag and writes them in sequential representation to the "Buffer" tag after the "arti" characters.

The following table shows the declaration of the operands:

Operand	Data type	Declaration
DeliverPos	INT	In the "Input" section of the block interface
BufferPos	DINT	In the "Temp" section of the block interface

Operand	Data type	Declaration
Error	INT	In the "Temp" section of the block interface
Label	STRING[4]	In the "Temp" section of the block interface

The following table shows the declarations of the PLC data types:

Name of the PLC data types	Name	Data type
Article	Number	DINT
	Declaration	STRING
	Colli	INT
Client	Title	INT
	First name	STRING[10]
	Surname	STRING[10]

The following table shows the declaration of the data blocks:

Name of the data blocks	Name	Data type
Source	Client	"Client" (PLC data type)
	Article	Array[0..10] of "Article" (PLC data type)
Buffer	Field	Array[0..294] of BYTE

See also

Overview of the valid data types (Page 2239)

Basics of PLC data types (UDT) (Page 2277)

Structure of an ARRAY tag (Page 2290)

Structure of a STRUCT tag (Page 2284)

Structure of a STRING tag (Page 2272)

Basics of SCL (Page 4991)

MOVE_BLK: Move block (S7-1200, S7-1500)

Description

You can use the "Move block" instruction to move the content of a memory area (source range) to another memory area (target range). The number of elements to be moved to the target range is specified with the COUNT parameter. The width of the elements to be moved is defined by the width of the first element in the source area.

The instruction can only be executed if the source range and the target range have the same data type.

The value of the OUT output is invalid if the following condition is met:

- More data is moved than is made available at the IN parameter or OUT parameter.

Note

Use of ARRAYs

The instruction copies the contents from the defined element n elements (n = depending on the value at the parameter COUNT) from the source range to the target range, starting at the specified index.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN [1]	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	The first element of the source area that is being copied
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P	Number of elements to be copied from the source range to the target range
OUT [1]	Output	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	The first element of the target range to which the contents of the source range are being copied
[1] The specified data types can only be used as elements of an ARRAY structure.					

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
MOVE_BLK(IN := #a_array[2],
         COUNT := "Tag_Count",
         OUT => #b_array[1]);
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	a_array[2]	The data type of the "a_array" operand is ARRAY[0..5] of INT. It consists of six elements of data type INT.
COUNT	Tag_Count	3
OUT	b_array[1]	The data type of the "b_array" operand is ARRAY[0..6] of INT. It consists of seven elements of data type INT.

Starting from the third element, the instruction selects three INT elements from the #a_array tag and copies their contents to the #b_array output tag, beginning with the second element.

Note

You can find more information on the MOVE_BLK instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SCATTER: Parse the bit sequence into individual bits (S7-1200, S7-1500)

Description

The instruction "Parse the bit sequence into individual bits" disassembles a tag of the BYTE, WORD, DWORD or LWORD data type into individual bits and saves them in an ARRAY of BOOL.

Note

Length of the ARRAY

The ARRAY of BOOL must have exactly the number of elements that is specified by the bit sequence.

This means for the data type BYTE, for example, the ARRAY must have exactly 8 elements (WORD = 16, DWORD = 32 and LWORD = 64).

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

This way you can, for example, parse a status word, and read and change the status of the individual bits using the index. Using GATHER, you can merge the bits once again into a bit sequence.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- If the ARRAY does not provide enough BOOL elements.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	BYTE, WORD, DWORD, LWORD	I, Q, M, D, L	Bit sequence that is parsed
OUT	Output	ARRAY[*] of BOOL *: 8, 16, 32 or 64 elements	I, Q, M, D, L	ARRAY in which the individual bits are saved

You can find additional information on valid data types under "See also".

Example

Create the following tags in the block interface:

Tag	Section	Data type
SourceWord	Input	WORD
DestinationArray	Output	ARRAY[0..15] of BOOL

The following example shows how the instruction works:

SCL

```
SCATTER(IN := #SourceWord,
        OUT => #DestinationArray);
```

The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceWord	WORD (16 bits)
OUT	DestinationArray	The operand "DestinationArray" is of the data type ARRAY[0..15] of BOOL. It consists of 16 elements and is therefore just as large as the WORD that is to be parsed.

The operand #SourceWord of the data type WORD is parsed into its individual bits (16) and assigned to the individual elements of the #DestinationArray operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SCATTER_BLK: Parse elements of an ARRAY of bit sequence into individual bits (S7-1200, S7-1500)

Description

The instruction "Parse elements of an ARRAY of bit sequence into individual bits" parses one or more elements of an ARRAY of BYTE, WORD, DWORD or LWORD into individual bits and saves them in an ARRAY of BOOL. At the COUNT_IN parameter you specify how many elements of the source ARRAY are going to be disassembled. The source ARRAY at the IN parameter may have more elements than specified at the COUNT_IN parameter. The ARRAY of BOOL must have sufficient elements to save the bits of the disassembled bit strings. However, the destination ARRAY may also be larger.

Note

If the low ARRAY limit of the target ARRAY is not "0", note the following:

For performance reasons the index must always start at a BYTE, WORD, DWORD or LWORD limit. This means the index must be calculated starting at the low limit of the ARRAY. The following formula is used as basis for this calculation:

Valid indices = Low ARRAY limit + n(number of bit strings) × number of bits of the desired bit string

For an ARRAY[-2..45] of BOOL and the bit string WORD the calculation looks as follows:

- Valid index (-2) = -2 + 0 × 16
- Valid index (14) = -2 + 1 × 16
- Valid index (30) = -2 + 2 × 16

You can find an example described below.

Note
Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

This way you can, for example, parse status words, and read and change the status of the individual bits using the index. Using GATHER, you can combine the bits once again into a bit string.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- The source ARRAY has fewer elements than specified at the COUNT_IN parameter.
- The index of the destination ARRAY does not start at a BYTE, WORD, DWORD or LWORD limit. In this case, no result is written to the ARRAY of BOOL.
- The ARRAY[*] of BOOL does not provide the required number of elements. In this case as many bit strings as possible are parsed and written to the ARRAY of BOOL. The remaining bit strings are no longer taken into account.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	Element of an ARRAY[*] of <bit string>	I, Q, M, D, L	ARRAY of <bit string> that is disassembled
COUNT_IN	Input	UDINT	I, Q, M, D, L	Counter for the number of elements of the source ARRAY that are going to be disassembled
OUT	Output	Element of an ARRAY[*] of BOOL	I, Q, M, D, L	ARRAY in which the individual bits are saved

You can find additional information on valid data types under "See also".

Example of a destination ARRAY wit the low limit "0"

Create the following tags in the block interface:

Tag	Section	Data type
SourceArrayWord	Input	ARRAY[0..5] of WORD
CounterInput		UDINT
DestinationArrayBool	Output	ARRAY[0..95] of BOOL

The following example shows how the instruction works:

SCL

```
SCATTER_BLK(IN := #SourceArrayWord[2],
            COUNT_IN := #CounterInput,
```

SCL

```
OUT => #DestinationArrayBool[0]);
```

The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayWord[2]	ARRAY[0..5] of WORD (96 bits can be disassembled.)
COUNT_IN	CounterInput = 3	UDINT3 (3 WORDs or 48 bits are to be parsed. This means at least 48 bits must be available in the destination ARRAY.)
OUT	DestinationArrayBool[0]	The operand "DestinationArrayBool" is of the data type ARRAY[0..95] of BOOL. This means it provides 96 BOOL elements.

The 3rd, 4th and 5th WORD of the operand #SourceArrayWord is parsed into its individual bits (48) and as of the first element assigned to the individual elements of the #DestinationArrayBool operand.

Example of a destination ARRAY wit the low limit "-2"

Create the following tags in the block interface:

Tag	Section	Data type
SourceArrayWord	Input	ARRAY[0..5] of WORD
CounterInput		UDINT
DestinationArrayBool	Output	ARRAY[-2..93] of BOOL

The following example shows how the instruction works:

SCL

```
SCATTER_BLK(IN := #SourceArrayWord[2],
            COUNT_IN := #CounterInput,
            OUT => #DestinationArrayBool[14]);
```

The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayWord[2]	ARRAY[0..5] of WORD (96 bits can be disassembled.)
COUNT_IN	CounterInput = 3	UDINT3 (3 WORDs or 48 bits are to be parsed. This means at least 48 bits must be available in the destination ARRAY.)
OUT	DestinationArrayBool[14]	The operand "DestinationArrayBool" is of the data type ARRAY[-2..93] of BOOL. This means it provides 96 BOOL elements.

The 3rd, 4th and 5th WORD of the operand #SourceArrayWord is parsed into its individual bits (48) and as of the 16th element assigned to the individual elements of the #DestinationArrayBool operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

GATHER: Merge individual bits into a bit sequence (S7-1200, S7-1500)

Description

The instruction "Merge individual bits into a bit sequence" combines the bits from an ARRAY of BOOL into a bit sequence. The bit sequence is saved in a tag of the data type BYTE, WORD, DWORD or LWORD.

Note

Length of the ARRAY

The ARRAY of BOOL must have exactly the number of elements that is specified by the bit sequence.

This means for the data type BYTE, for example, the ARRAY must have exactly 8 elements (WORD = 16, DWORD = 32 and LWORD = 64).

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- If the ARRAY has fewer or more BOOL elements than specified by the bit sequence, they are not transferred and the ENO returns signal state "0". The ENO also returns "0" if fewer than the necessary number of bits is available.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	ARRAY[*] of BOOL *: 8, 16, 32 or 64 elements	I, Q, M, D, L	ARRAY whose bits are combined into a bit sequence
OUT	Output	BYTE, WORD, DWORD, LWORD	I, Q, M, D, L	Combined bit sequence, saved in a tag

You can find additional information on valid data types under "See also".

Example

Create the following tags in the block interface:

Tag	Section	Data type
SourceArray	Input	ARRAY[0..15] of BOOL
DestinationWord	Output	WORD

The following example shows how the instruction works:

SCL

```
GATHER(IN := #SourceArray,
       OUT => #DestinationWord);
```

The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArray	The operand "SourceArray" is of the data type ARRAY[0..15] of BOOL. It consists of 16 elements and is therefore just as large as the WORD in which the bits are to be combined.
OUT	DestinationWord	WORD (16 bits)

The bits of the operand #SourceArray are combined into a WORD.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

GATHER_BLK: Merge individual bits into multiple elements of an ARRAY of bit sequence (S7-1200, S7-1500)

Description

The instruction "Merge individual bits into multiple elements of an ARRAY of bit sequence" combines the bits from an ARRAY of BOOL into one or multiple elements of an ARRAY of <bit string>. At the COUNT_OUT parameter you specify how many elements of the destination ARRAY are going to be written. With this step you also implicitly specify how many elements of the ARRAY of BOOL are required. The destination ARRAY at the OUT parameter may have more elements than specified at the COUNT_OUT parameter. The ARRAY of <bit string> must have sufficient elements to save the bits that are going to be combined. However, the destination ARRAY may also be larger.

Note

If the low ARRAY limit of the target ARRAY is not "0", note the following:

For performance reasons the index must always start at a BYTE, WORD, DWORD or LWORD limit. This means the index must be calculated starting at the low limit of the ARRAY. The following formula is used as basis for this calculation:

Valid indices = Low ARRAY limit + n(number of bit strings) × number of bits of the desired bit string

For an ARRAY[-2..45] of BOOL and the bit string WORD the calculation looks as follows:

- Valid index (-2) = -2 + 0 × 16
- Valid index (14) = -2 + 1 × 16
- Valid index (30) = -2 + 2 × 16

You can find an example described below.

Note

Availability of instruction on a S7-1500 CPU

The instruction can be used with a CPU of the S7-1200 series as of firmware version 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.1.

The enable output ENO returns signal state "0" if one of the following conditions applies:

- Enable input EN has signal state "0".
- The index of the source ARRAY does not start at a BYTE, WORD, DWORD or LWORD limit. In this case, no result is written to the ARRAY of <bit string>.
- The ARRAY[*] of <bit string> does not provide the required number of elements. In this case as many bit strings as possible are combined and written to the ARRAY of <bit string>. The remaining bits are no longer taken into account.

Parameter

The following table shows the parameters of the instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	Element of an ARRAY[*] of BOOL	I, Q, M, D, L	ARRAY of BOOL whose bits are combined (source ARRAY)
COUNT_OUT	Input	UDINT	I, Q, M, D, L	Counter of the number of elements of the destination ARRAY that are going to be written
OUT	Output	Element of an ARRAY[*] of <bit string>	I, Q, M, D, L	ARRAY of <bit string> to which the bits are saved (destination ARRAY)

You can select the required bit string from the "???" drop-down list of the instruction box.

You can find additional information on valid data types under "See also".

Example of a source ARRAY wit the low limit "0"

Create the following tags in the block interface:

Tag	Section	Data type
SourceArrayBool	Input	ARRAY[0..95] of BOOL
CounterOutput		UDINT
DestinationArrayWord	Output	ARRAY[0..5] of WORD

The following example shows how the instruction works:

SCL

```
GATHER_BLK(IN := #SourceArrayBool[0],
           COUNT_OUT := #CounterOutput,
           OUT => #DestinationArrayWord[2]);
```

The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayBool[0]	The operand "SourceArrayBool" is of the data type ARRAY[0..95] of BOOL. This means it provides 96 BOOL elements that can combined into words once again.
COUNT_OUT	CounterOutput = 3	UDINT3 (3 words are to be written. This means 48 bits must be available in the source ARRAY.)
OUT	DestinationArrayWord[2]	The operand "DestinationArrayWord" is of the data type ARRAY[0..5] of WORD. This means 6 WORD elements are available.

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. Starting with the 1st element of the operand #SourceArrayBool, 48 bits are combined in the operand #DestinationArrayWord. The starting point in the destination ARRAY is the 3rd element. This means the first 16 bits are written to the 3rd word, the second 16 bits to the 4th word, and the third 16 bits to the 5th word of the destination ARRAY. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

Example of a source ARRAY wit the low limit "-2"

Create the following tags in the block interface:

Tag	Section	Data type
SourceArrayBool	Input	ARRAY[-2..93] of BOOL
CounterOutput		UDINT
DestinationArrayWord	Output	ARRAY[0..5] of WORD

The following example shows how the instruction works:

SCL

```
GATHER_BLK(IN := #SourceArrayBool[14],
           COUNT_OUT := #CounterOutput,
           OUT => #DestinationArrayWord[2]);
```

The following table shows how the instruction works using specific operand values:

Parameter	Operand	Data type
IN	SourceArrayBool[14]	The operand "SourceArrayBool" is of the data type ARRAY[-2..93] of BOOL. Because the starting point is the 14th element, only 80 BOOL elements that can be combined into words again are available.
COUNT_OUT	CounterOutput = 3	UDINT3 (3 words are to be written. This means 48 bits must be available in the source ARRAY.)
OUT	DestinationArrayWord[2]	The operand "DestinationArrayWord" is of the data type ARRAY[0..5] of WORD. This means 6 WORD elements are available.

If the operand #Enable returns the signal state "1" at the enable input EN, the instruction is executed. Starting with the 14th element of the operand #SourceArrayBool, 48 bits are combined in the operand #DestinationArrayWord. The starting point in the destination ARRAY is the 3rd element. This means the first 16 bits of the source ARRAY are ignored. The second 16 bits are written to the 3rd word, the third 16 bits to the 4th word, and the fourth 16 bits to the 5th word of the destination ARRAY. The remaining 64 bits of the source ARRAY are not taken into account either. If an error occurs during the execution of the instruction, the operand #EnableOut returns the signal state "0" at the enable output ENO.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

MOVE_BLK_VARIANT: Move block (S7-1200, S7-1500)

Description

You can use the "Move block" instruction to move the content of a memory area (source range) to another memory area (target range). You can copy a complete ARRAY or elements of an ARRAY to another ARRAY of the same data type. The size (number of elements) of source and destination ARRAY may be different. You can copy multiple or single elements within an ARRAY.

The number of elements that are to be copied must not exceed the selected source range or target range.

If you use the instruction at the time the block is being created, the ARRAY does not yet have to be known, as the source and the target are transferred per VARIANT.

Counting at the parameters SRC_INDEX and DEST_INDEX always begins with the low limit "0", regardless of the later declaration of the ARRAY.

The instruction is not executed if more data is copied than is made available.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
SRC	Input [2)]	VARIANT (which points to an ARRAY or an individual ARRAY element), ARRAY of <Data_type>	L (A declaration in the "Temp" section of the block interface is not possible.)	Source block from which to copy
COUNT	Input	UDINT	I, Q, M, D, L	Number of elements which are copied Assign the value "1" to the parameter COUNT, if no ARRAY is specified at parameter SRC or at parameter DEST.
SRC_INDEX	Input	DINT	I, Q, M, D, L	Defines the first element to be copied: • The SRC_INDEX parameter is calculated zero-based. If an ARRAY is specified at parameter SRC, the integer at parameter SRC_INDEX specifies the first element within the source area from which it is to be copied. Independent of the declared ARRAY limits. • If no ARRAY is specified at parameter SRC or only one single element of an ARRAY is specified, then assign the value "0" at parameter SRC_INDEX.

Parameters	Declaration	Data type	Memory area	Description
DEST_INDEX	Input	DINT	I, Q, M, D, L	Defines the start of the destination memory area: • The DEST_INDEX parameter is calculated zero-based. If an ARRAY is specified at parameter DEST, the integer at parameter DEST_INDEX specifies the first element within the target range that is to be copied into. Independent of the declared ARRAY limits. • If no ARRAY is specified at parameter DEST, then assign the value "0" at parameter DEST_INDEX.
DEST	Output [1]	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Destination area into which the contents of the source block are copied.
Function value (RET_VAL)		INT	I, Q, M, D, L	Error information
1) The DEST parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface. 2) With the SRC parameter, the data types BOOL and ARRAY of BOOL are not permitted.				

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The data types do not correspond. Instead of an ARRAY of <Struct>, use an ARRAY of <PLC data type (UDT)>.
8151	Access to the SRC parameter is not possible.
8152	The operand at the SRC parameter is not typed.
8153	Code generation error at the SRC parameter
8154	The operand at the SRC parameter has the data type BOOL.
8281	The COUNT parameter has an invalid value
8382	The value at the SRC_INDEX parameter is outside the limits of the VARIANT.
8383	The value at the SRC_INDEX is outside the high limit of the ARRAY.
8482	The value at the DEST_INDEX parameter is outside the limits of the VARIANT.
8483	The value at the DEST_INDEX is outside the high limit of the ARRAY.

Error code* (W#16#...)	Explanation
8534	The DEST parameter is write protected
8551	Access to the DEST parameter is not possible.
8552	The operand at the DEST parameter is not typed.
8553	Code generation error at the DEST parameter
8554	The operand at the DEST parameter has the data type BOOL.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := MOVE_BLK_VARIANT(SRC := #SrcField,
                                 COUNT := "Tag_Count",
                                 SRC_INDEX := "Tag_Src_Index",
                                 DEST_INDEX := "Tag_Dest_Index",
                                 DEST => #DestField);
```

The following table shows how the instruction works using specific operand values:

Parameters	Declaration in the block interface	Operand	Value
SRC	Input	#SrcField	The local operand #SrcField uses a PLC data type that was still unknown at the time when the block was programmed. (ARRAY[0..10] of "MOVE_UDT")
COUNT	Input	Tag_Count	2
SRC_INDEX	Input	Tag_Src_Index	3
DEST_INDEX	Input	Tag_Dest_Index	3
DEST	InOut	#DestField	The local operand #DestField uses a PLC data type that was still unknown at the time when the block was programmed. (ARRAY[10..20] of "MOVE_UDT")

Two elements are moved from the source range, starting from the fourth element of the ARRAY [0..10] of MOVE_UDT, to the target range. The copies are pasted in the ARRAY [10..20] of MOVE_UDT starting from the fourth element.

Note

You can find more information on the MOVE_BLK_VARIANT instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

VariantGet: Read out VARIANT tag value (Page 3349)

Basics of SCL (Page 4991)

UMOVE_BLK: Move block uninterruptible (S7-1200, S7-1500)

Description

You use the "Move block uninterruptible" instruction to move the content of a memory area (source range) to another memory area (target range). The instruction cannot be interrupted. The number of elements to be moved to the target range is specified with the COUNT parameter. The width of the elements to be moved is defined by the width of the first element in the source area.

The instruction can only be executed if the source range and the target range have the same data type.

Note

The move operation cannot be interrupted by other operating system activities. This is why the interrupt reaction times of the CPU increase during the execution of the "Move block uninterruptible" instruction.

The value of the OUT output is invalid if the following condition is met:

- More data is moved than is made available at the IN parameter or OUT parameter.

You can use the "Move block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Note

Use of ARRAYs

The instruction copies the contents from the defined element n elements (n = depending on the value at the parameter COUNT) from the source range to the target range, starting at the specified index.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN [1]	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	The first element of the source area that is being copied
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P	Number of elements to be copied from the source range to the target range
OUT [1]	Output	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	The first element of the target range to which the contents of the source range are being copied
[1] The specified data types can only be used as elements of an ARRAY structure.					

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
UMOVE_BLK(IN := #a_array[2],
          COUNT := "Tag_Count",
          OUT => #b_array[1]);
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	a_array[2]	The data type of the "a_array" operand is ARRAY[0..5] of INT. It consists of six elements of data type INT.
COUNT	Tag_Count	3
OUT	b_array[1]	The data type of the "b_array" operand is ARRAY[0..6] of INT. It consists of seven elements of data type INT.

Starting from the third element, the instruction selects three INT elements from the #a_array tag and copies their contents to the #b_array output tag, beginning with the second element. The move operation cannot be interrupted by other operating system activities.

Note

You can find more information on the UMOVE_BLK instruction in the following article in the Siemens Industry Online Support:

In STEP 7 (TIA Portal) how do you copy memory areas and structured data from one data block to another?

https://support.industry.siemens.com/cs/ww/en/view/42603881 (https://support.industry.siemens.com/cs/ww/en/view/42603881)

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

FILL_BLK: Fill block (S7-1200, S7-1500)

Description

You can use the "Fill block" instruction to fill a memory area (target range) with the value of the IN input. The target range is filled beginning with the address specified at the OUT output. The number of repeated copy operations is specified with the COUNT parameter. When the instruction is executed, the value at the input IN is moved to the target range as often as specified by the value of the COUNT parameter.

The instruction can only be executed if the source range and the target range have the same data type.

The maximum number of elements changed is the number of elements in the ARRAY or structure. If you copy more data than there are elements at the OUT output, you will get an unintended result.

Note

Use of ARRAYs

The instruction reads the content from the selected element from the source range and copies this content n-times (n = depending on the value at the parameter COUNT) to the destination, starting at the specified index.

Filling structures

As well as the elements of an ARRAY, you can also fill multiple elements of a structure (STRUCT, PLC data type) with the same value. The structure whose elements you want to fill must only contain individual elements of the same elementary data type. The structure itself can, however, be embedded in another structure.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Binary numbers, integers, floating-point numbers, timers, TOD, DATE, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	I, Q, M, D, L, P	Element used to fill the target range
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P	Number of repeated move operations
OUT	Output	Binary numbers, integers, floating-point numbers, timers, TOD, DATE, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	Address in target range from which filling starts

For additional information on valid data types, refer to "See also".

Example with an ARRAY

The following example shows how the instruction works when you want to fill an ARRAY:

SCL

```
FILL_BLK(IN := #FillValue,
         COUNT := "Tag_Count",
         OUT => #TargetArea[1]);
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	FillValue	The data type of the operand is INT.
COUNT	Tag_Count	3
OUT	TargetArea	The data type of the operand TargetArea is ARRAY[1..5] of INT. It consists of five elements of data type INT.

The instruction copies the value of the operand #FillValue to the #TargetArea output tag three times, starting with the first element.

Examples with a structure

The following examples show how the instruction works when you want to fill a structure:

Create a global data block with the following elements:

Data_block_1		Data type
MyStruct1		STRUCT
	Member_1	INT
	Member_2	INT
	Member_3	INT
	Member_4	INT
MyStruct2		STRUCT

Data_block_1					Data type
	SubArray				ARRAY[1..2] of STRUCT
		SubArray[1]			STRUCT
			NestedStruct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT
		SubArray[2]			STRUCT
			Nested-Struct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT

Generate the following program code to address the MyStruct1 tag:

SCL

```
FILL_BLK(IN := 10,
         COUNT := 2,
         OUT => "Data_block_1".MyStruct1.Member_2);
```

Generate the following program code to address the MyStruct2 tag:

SCL

```
FILL_BLK(IN := 10,
         COUNT := 2,
        OUT => "Data_block_1".MyStruct2.SubArray[1].NestedStruct.Member_2);
```

In each of the two examples, the value 10 of the IN parameter is copied twice to the operand at the OUT parameter, starting with the element Member_2. This means that the value 10 is copied to the elements Member_2 and Member_3. The other two elements, Member_1 and Member_4, are not changed.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

UFILL_BLK: Fill block uninterruptible (S7-1200, S7-1500)

Description

You can use the "Fill block uninterruptible" instruction to fill a memory area (target range) with the value of the IN input. The instruction cannot be interrupted. The target range is filled beginning with the address specified at the OUT output. The number of repeated copy operations is specified with the COUNT parameter. When the instruction is executed, the value at the input IN is moved to the target range as often as specified by the value of the COUNT parameter.

The instruction can only be executed if the source range and the target range have the same data type.

Note

The move operation cannot be interrupted by other operating system activities. This is why the alarm reaction times of the CPU increase during the execution of the "Fill block uninterruptible" instruction.

The maximum number of elements changed is the number of elements in the ARRAY or structure. If you copy more data than there are elements at the OUT output, you will get an unintended result.

Note

Use of ARRAYs

The instruction reads the content from the selected element from the source range and copies this content n-times (n = depending on the value at the parameter COUNT) to the destination, starting at the specified index.

You use the "Fill block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Filling structures

As well as the elements of an ARRAY, you can also fill multiple elements of a structure (STRUCT, PLC data type) with the same value. The structure whose elements you want to fill must only contain individual elements of the same elementary data type. The structure itself can, however, be embedded in another structure.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	I, Q, M, D, L, P	Element used to fill the target range
COUNT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L, P	Number of repeated move operations
OUT	Output	Binary numbers, integers, floating-point numbers, timers, TOD, DATE, CHAR, WCHAR	Binary numbers, integers, floating-point numbers, timers, DATE, CHAR, WCHAR, TOD, LTOD	D, L	Address in target range from which filling starts

For additional information on valid data types, refer to "See also".

Example with an ARRAY

The following example shows how the instruction works when you want to fill an ARRAY:

SCL

```
UFILL_BLK(IN := #FillValue,
          COUNT := "Tag_Count",
          OUT => #TargetArea[1]);
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	FillValue	The data type of the operand is INT.
COUNT	Tag_Count	3
OUT	TargetArea	The data type of the operand TargetArea is ARRAY[1..5] of INT. It consists of five elements of data type INT.

The instruction copies the value of the operand #FillValue to the #TargetArea output tag three times, starting with the first element. The move operation cannot be interrupted by other operating system activities.

Examples with a structure

The following examples show how the instruction works when you want to fill a structure:

Create a global data block with the following elements:

Data_block_1					Data type
MyStruct1					STRUCT
	Member_1				INT
	Member_2				INT
	Member_3				INT
	Member_4				INT
MyStruct2					STRUCT
	SubArray				ARRAY[1..2] of STRUCT
		SubArray[1]			STRUCT
			NestedStruct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT
		SubArray[2]			STRUCT
			Nested-Struct		STRUCT
				Member_1	INT
				Member_2	INT
				Member_3	INT
				Member_4	INT

Generate the following program code to address the MyStruct1 tag:

SCL

```
UFILL_BLK(IN := 10,
          COUNT := 2,
          OUT => "Data_block_1".MyStruct1.Member_2);
```

Generate the following program code to address the MyStruct2 tag:

SCL

```
UFILL_BLK(IN := 10,
          COUNT := 2,
          OUT => "Data_block_1".MyStruct2.SubArray[1].NestedStruct.Member_2);
```

In each of the two examples, the value 10 of the IN parameter is copied twice to the operand at the OUT parameter, starting with the element Member_2. This means that the value 10 is copied to the elements Member_2 and Member_3. The other two elements, Member_1 and Member_4, are not changed.

See also

Overview of the valid data types (Page 2239)

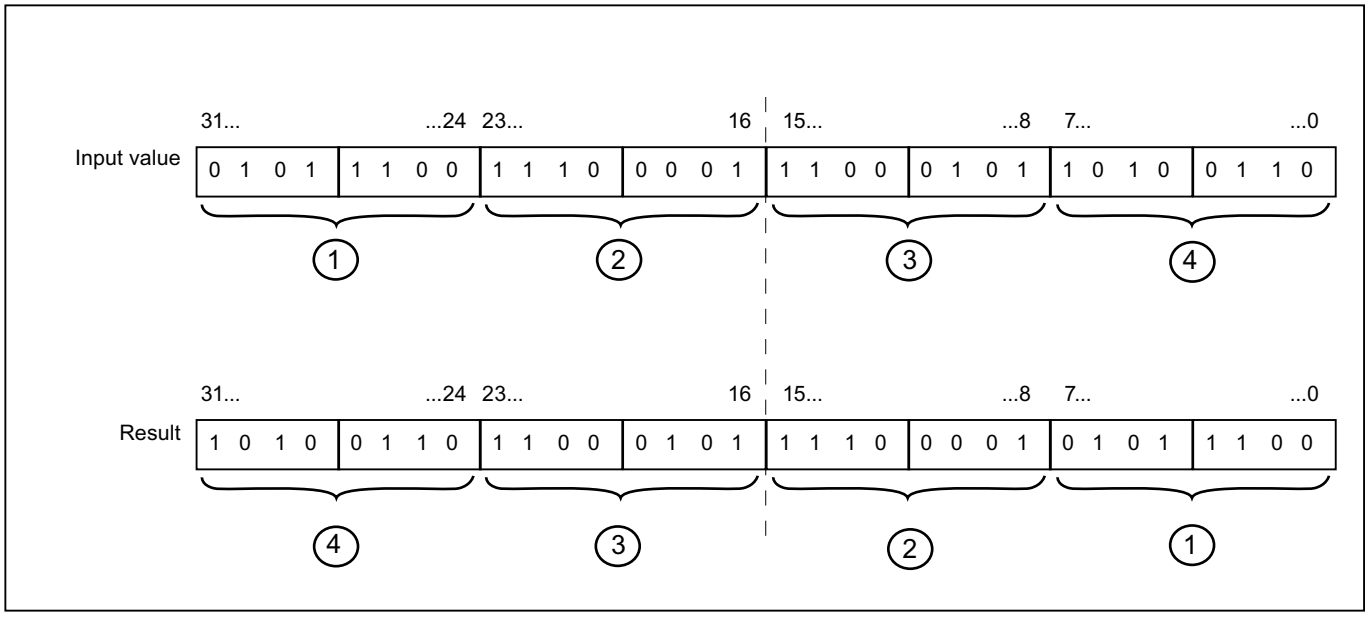
Basics of SCL (Page 4991)

SWAP: Swap (S7-1200, S7-1500)

Description

You can use the "Swap" instruction to change the arrangement of the bytes of an input value and save the result in the specified operand.

The following figure shows how the bytes of an operand of the DWORD data type are swapped using the "Swap" instruction:



Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
<Expression>	Input	WORD, DWORD	WORD, DWORD, LWORD	I, Q, M, D, L, P	Input value
Function value		WORD, DWORD	WORD, DWORD, LWORD	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := SWAP("Tag_Value");
```

The result of the instruction is returned as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value	0000 1111 0101 0101
Tag_Result	0101 0101 0000 1111

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ARRAY DB (S7-1500)

ReadFromArrayDB: Read from array data block (S7-1500)

Description

The instruction "Read from ARRAY data block" is used to read the element from a data block of the ARRAY DB block type to which the index references, and write the value of the element to the target range.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
DB	Input	DB_ANY	I, Q, M, D, L	Data block that is read
INDEX	Input	DINT	I, Q, M, D, L, P	Element in the DB that is read. The specification can be a constant, a global tag or an indexed value.
VALUE	Output [1)]	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Value that is read and output

Parameters	Declaration	Data type	Memory area	Description
Function value (RET_VAL)		INT	I, Q, M, D, L	Result of the instruction
1) The VALUE parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
80B5	The copying was interrupted.
8132	The data block does not exist, is too short, write protected, or is located in load memory.
8135	The ARRAY data block contains invalid values.
8154	The data block has the incorrect data type.
8282	The value at the INDEX parameter is outside the limits of the ARRAY.
8450	The data type VARIANT at parameter VALUE provides the value "0".
8452	Code generation error
8453	There are two possible causes of the error: • The size of the VALUE parameter does not match the element length in the ARRAY data block. • The two tags are not in a memory area with optimized access. For additional information on the memory area access types, refer to: AUTOHOTSPOT
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"TagResult" := ReadFromArrayDB(DB := "ArrayDB",
                               INDEX := 2,
                               VALUE => "TargetField");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type Array [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	TargetField	The "TargetField" operand is a global tag of the data type INT.

The 2nd element is read from "ArrayDB" and written to the "TargetField" operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

WriteToArrayDB: Write to array data block (S7-1500)

Description

The instruction "Write to ARRAY data block" is used to write the element to which the index references to a data block of the ARRAY DB block type.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
DB	Input	DB_ANY	I, Q, M, D, L	Data block to which data is written
INDEX	Input	DINT	I, Q, M, D, L, P	Element in the DB to which data is written. The specification can be a constant, a global tag or an indexed value.
VALUE	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Value to be written
Function value (RET_VAL)		INT	I, Q, M, D, L	Result of the instruction

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
80B5	The copying was interrupted.
8132	The data block does not exist, is too short, or is located in load memory.
8134	The data block is write protected.
8135	The data block is not an ARRAY data block.
8154	The data block has the incorrect data type.
8282	The value at the INDEX parameter is outside the limits of the ARRAY.
8350	The data type VARIANT at parameter VALUE provides the value "0".
8352	Code generation error
8353	There are two possible causes of the error: • The size of the VALUE parameter does not match the element length in the ARRAY data block. • The two tags are not in a memory area with optimized access. For additional information on the memory area access types, refer to: AUTOHOTSPOT
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"TagResult" := WriteToArrayDB(DB := "ArrayDB",
                              INDEX := 2,
                              VALUE := "SourceField");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type Array [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	SourceField	The "SourceField" operand is a global tag of the data type INT.

The value of the "SourceField" operand is written to the 2nd element of the ARRAY DB.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ReadFromArrayDBL: Read from array data block in load memory (S7-1500)

Description

The instruction "Read from ARRAY data block in load memory" is used to read the element ti which the index references from a data block of the ARRAY DB block type in the load memory and write it to the target range.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

If the ARRAY data block has been designated with the block attribute "Only store in load memory", it will only be stored in the load memory.

The instruction is executed when a positive signal edge is detected at the REQ parameter. The BUSY parameter then has the signal state "1". The instruction is terminated if a negative signal edge is detected at the BUSY parameter. The DONE parameter has the signal state "1" for one program cycle and the read value is output at the VALUE parameter within this cycle. With all other program cycles, the value at the VALUE parameter is not changed.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L	REQ = "1": Begin with the reading of the ARRAY DB
DB [1)]	Input	DB_ANY	I, Q, M, D, L	ARRAY data block that is read.
INDEX	Input	DINT	I, Q, M, D, L, P	Element in the DB that is read. The specification can be a constant, a global tag or an indexed value.
VALUE [1)]	InOut	VARIANT	D (element of a global data block) L (A declaration in the "Temp" section of the block interface is not possible.)	Pointer to the DB in the work memory that is to be read and the value of which is to be written. No local constants or tags from the TEMP section must be used.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = "1": The array DB is still being read
DONE	Output	BOOL	I, Q, M, D, L	DONE = "1": The instruction was executed successfully

Parameters	Declaration	Data type	Memory area	Description
ERROR	Output	INT	I, Q, M, D, L	Error information: If an error occurs during execution of the instruction, an error code is output at the ERROR parameter.
[1] The data blocks must be created with the "Optimized" block property.				

ERROR parameter

The following table shows the meaning of the values of the ERROR parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
8230	The data block number is incorrect.
8231	The data block does not exist.
8232	The data block is too short, or is not located in load memory.
8235	The data block is not an ARRAY DB.
8254	The data block has the incorrect data type.
8382	The value at the INDEX parameter is outside the limits of the ARRAY.
8750	The data type VARIANT at parameter VALUE provides the value "0".
8751	Code generation error
8752	Code generation error
8753	The size of the VALUE parameter does not match the element length in the ARRAY data block.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

You can find descriptions of the error codes triggered by the instructions "READ_DBL" and "WRIT_DBL" in the corresponding descriptions of the instructions.

Example

The following example shows how the instruction works:

SCL

```
"ReadFromArrayDBL_DB"(REQ := "TagReg",
                      DB := "ArrayDB",
                      INDEX := 2,
                      VALUE := "TargetField",
                      BUSY => "TagBusy",
                      DONE => "TagDone",
                      ERROR => "TagError");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
REQ	TagReq	BOOL
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type ARRAY [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	TargetField	The "TargetField" operand is a global tag of the data type INT.
BUSY	TagBusy	BOOL
DONE	TagDone	BOOL

The instruction is executed when a positive signal edge is detected at the "TagReq" operand. The 2nd element is read from "ArrayDB" and output at the VALUE parameter. As soon as a negative signal edge is detected at the "TagBusy" operand, the instruction is terminated and the value at the VALUE parameter is no longer changed. After the instruction has been processed, the "TagDone" operand has the signal state TRUE.

See also

Overview of the valid data types (Page 2239)

READ_DBL: Read from data block in the load memory (Page 4050)

WRIT_DBL: Write to data block in the load memory (Page 4054)

Difference between synchronous and asynchronous instructions (Page 2571)

Basics of SCL (Page 4991)

WriteToArrayDBL: Write to array data block in load memory (S7-1500)

Description

The instruction "Write to ARRAY data block in load memory" is used to write the element to which the index references to a data block of the ARRAY DB block type in load memory.

An ARRAY data block is a data block that consists of exactly one ARRAY of <Data type>. The elements of the ARRAY can be of PLC data type or any other elementary data type. Counting of the ARRAY always begins with the low limit "0".

If the ARRAY data block has been designated with the block attribute "Only store in load memory", it will only be stored in the load memory.

The instruction is executed when a positive signal edge is detected at the REQ parameter. The BUSY parameter then has the signal state "1". If a negative signal edge is detected at the BUSY parameter, the instruction is terminated and the value at the VALUE parameter is written to the data block. The DONE parameter then has the signal state "1" for one program cycle.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L	REQ = "1": Start writing to the array DB
DB [1]	Input	DB_ANY	I, Q, M, D, L	ARRAY data block to which data is written
INDEX	Input	DINT	I, Q, M, D, L, P	Element in the DB to which data is written. The specification can be a constant, a global tag or an indexed value.
VALUE [1]	Input	VARIANT	D (element of a global data block) L (A declaration in the "Temp" section of the block interface is not possible.)	Pointer to the DB in the work memory that is to be read and the value of which is to be written. No local constants or tags from the TEMP section must be used.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = "1": The array DB is still being written
DONE	Output	BOOL	I, Q, M, D, L	DONE = "1": The instruction was executed successfully
ERROR	Output	INT	I, Q, M, D, L	Error information: If an error occurs during execution of the instruction, an error code is output at the ERROR parameter.
[1] The data blocks must be created with the "Optimized" block property.				

ERROR parameter

The following table shows the meaning of the values of the ERROR parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
8230	The data block number is incorrect.
8231	The data block does not exist.
8232	The data block is too short, or is not located in load memory.
8234	The data block is write protected.
8235	The data block is not an ARRAY DB.
8254	The data block has the incorrect data type.
8382	The value at the INDEX parameter is outside the limits of the ARRAY.
8450	The data type VARIANT at parameter VALUE provides the value "0".
8751	Code generation error

Error code* (W#16#...)	Explanation
8752	Code generation error
8753	The size of the VALUE parameter does not match the element length in the ARRAY data block.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

You can find descriptions of the error codes triggered by the instructions "READ_DBL" and "WRIT_DBL" in the corresponding descriptions of the instructions.

Example

The following example shows how the instruction works:

SCL

```
"WriteToArrayDBL_DB"(REQ := "TagReg",
                     DB := "ArrayDB",
                     INDEX := 2,
                     VALUE := "SourceField",
                     BUSY => "TagBusy",
                     DONE => "TagDone",
                     ERROR => "TagError");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
REQ	TagReq	BOOL
DB	ArrayDB	The "ArrayDB" operand is an ARRAY DB of the data type ARRAY [0 to 10] of INT.
INDEX	2	2nd element of the "ArrayDB"
VALUE	SourceField	The "SourceField" operand is a global tag of the data type INT.
BUSY	TagBusy	BOOL
DONE	TagDone	BOOL

The instruction is executed when a positive signal edge is detected at the "TagReq" operand. As soon as a negative signal edge is detected at the "TagBusy" operand, the instruction is terminated and the value at the VALUE parameter is written to the 2nd element of the "ArrayDB". After the instruction has been processed, the "TagDone" operand has the signal state TRUE.

See also

Overview of the valid data types (Page 2239)

READ_DBL: Read from data block in the load memory (Page 4050)

WRIT_DBL: Write to data block in the load memory (Page 4054)

Difference between synchronous and asynchronous instructions (Page 2571)

Basics of SCL (Page 4991)

Read/write access (S7-1200, S7-1500)

PEEK: Read memory address (S7-1200, S7-1500)

Description

The "Read memory address" instruction is used to read a memory address from a standard memory area without specifying a data type.

Use the following syntax to change the data type of the instruction:

PEEK_<data type>();

Use of memory area 16#84: DB

If you use a data block as memory area and this is not yet known during the creation of the program code you have the option of using the EN/ENO mechanism to recognize possible access errors:

SCL

```
#Peeker := PEEK(AREA := 16#84,
                DBNUMBER := 1,
                BYTEOFFSET := 2,
                ENO => ENO);
IF NOT ENO THEN;
     #Peeker := 16#ffff;
END_IF;
```

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
AREA	Input	BYTE	I, Q, M, D	The following areas can be selected: • 16#81: Input • 16#82: Output • 16#83: Bit memory • 16#84: DB • 16#1: Peripheral input (S7-1500 only)
DBNUM-BER	Input	DINT, DB_ANY	D	Number of the data block if AREA = DB, otherwise "0"

Parameters	Declaration	Data type	Memory area	Description
BYTEOFF-SET	Input	DINT	I, Q, M, D	Address to read from Only the 16 least significant bits are used.
_<data type>		Bit strings default: BYTE	-	Data type of the function value: 1. You can specify the data type of the instruction explicitly using "_". 2. If you do not specify the data type explicitly, it will be determined by the utilized tags or type-coded constants. 3. If you neither specify the data type explicitly nor specify defined tags or type-coded constants, the default data type will be used.
Function value		Bit strings	I, Q, M, D	Result of the instruction

Note

If you read the memory address from the input, output or bit memory areas, you must assign the DBNUMBER parameter the value "0", as the instruction is invalid otherwise.

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := PEEK(AREA := "Tag_Area",
                      DBNUMBER := "Tag_DBNumber",
                      BYTEOFFSET := "Tag_Byte");
```

SCL

```
"Tag_Result2" := PEEK_WORD(AREA := "Tag_Area",
                           DBNUMBER := "Tag_DBNumber",
                           BYTEOFFSET := "Tag_Byte");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
AREA	Tag_Area	16#84
DBNUMBER	Tag_DBNumber	5
BYTEOFFSET	Tag_Byte	20
Function value	Tag_Result1	Byte value "20"
Function value	Tag_Result2	Word value "20"

The instruction reads the value of address "20" from the "Tag_Byte" operand at data block "5" and returns the result as a function value at the "Tag_Result" operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

PEEK_BOOL: Read memory bit (S7-1200, S7-1500)

Description

The "Read memory bit" instruction is used to read a memory bit from a standard memory area without specifying a data type.

Use of memory area 16#84: DB

If you use a data block as memory area and this is not yet known during the creation of the program code you have the option of using the EN/ENO mechanism to recognize possible access errors:

SCL

```
#Peeker_BOOL := PEEK_BOOL(AREA := 16#84,
                          DBNUMBER := 1,
                          BYTEOFFSET := 2,
                          BITOFFSET := 1,
                          ENO => ENO);
IF NOT ENO THEN;
     #Peeker_BOOL := 0;
END_IF;
```

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
AREA	Input	BYTE	I, Q, M, D	The following areas can be selected: • 16#81: Input • 16#82: Output • 16#83: Bit memory • 16#84: DB • 16#1: Peripheral input (S7-1500 only)
DBNUM-BER	Input	DINT, DB_ANY	D	Number of the data block if AREA = DB, otherwise "0"
BYTEOFF-SET	Input	DINT	I, Q, M, D	Address to read from Only the 16 least significant bits are used.

Parameters	Declaration	Data type	Memory area	Description
BITOFFSET	Input	INT	I, Q, M, D	Bit to be read from
Function value		BOOL	I, Q, M, D	Result of the instruction

Note

If you read the memory bit from the input, output or bit memory areas, you must assign the DBNUMBER parameter the value "0", as the instruction is invalid otherwise.

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := PEEK_BOOL(AREA := "Tag_Area",
                          DBNUMBER := "Tag_DBNumber",
                          BYTEOFFSET := "Tag_Byte",
                          BITOFFSET := "Tag_Bit");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
AREA	Tag_Area	16#84
DBNUMBER	Tag_DBNumber	5
BYTEOFFSET	Tag_Byte	20
BITOFFSET	Tag_Bit	3
Function value	Tag_Result	3

The instruction reads the value of memory bit "3" from the "Tag_Bit" operand at byte "20" of data block "5" and returns the result at the "Tag_Result" operand as function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

POKE: Write memory address (S7-1200, S7-1500)

Description

The "Write memory address" instruction is used to write a memory address to a standard memory area without specifying a data type.

Use of memory area 16#84: DB

If you use a data block as memory area and this is not yet known during the creation of the program code you have the option of using the EN/ENO mechanism to recognize possible access errors:

SCL

```
POKE(AREA := 16#84,
     DBNUMBER := 1,
     BYTEOFFSET := 2,
     VALUE := BYTE#7,
     ENO => ENO);
```

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
AREA	Input	BYTE	I, Q, M, D	The following areas can be selected: • 16#81: Input • 16#82: Output • 16#83: Bit memory • 16#84: DB • 16#2: Peripheral output (S7-1500 only)
DBNUM-BER	Input	DINT, DB_ANY	D	Number of the data block if AREA = DB, otherwise "0"
BYTEOFF-SET	Input	DINT	I, Q, M, D	Address to be written Only the 16 least significant bits are used.
VALUE	Input	Bit strings	I, Q, M, D	Value to be written

Note

If you write the memory address to the input, output or bit memory areas, you must assign the DBNUMBER parameter the value "0", as the instruction is invalid otherwise.

Example

The following example shows how the instruction works:

SCL

```
POKE(AREA := "Tag_Area",
     DBNUMBER := "Tag_DBNumber",
     BYTEOFFSET := "Tag_Byte");
     VALUE := "Tag_Value",
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
AREA	Tag_Area	16#84
DBNUMBER	Tag_DBNumber	5
BYTEOFFSET	Tag_Byte	20
VALUE	Tag_Value	16#11

The instruction overwrites the memory address "20" in the data block "5" with value "16#11".

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

POKE_BOOL: Write memory bit (S7-1200, S7-1500)

Description

The "Write memory bit" instruction is used to write a memory bit to a standard memory area without specifying a data type.

Use of memory area 16#84: DB

If you use a data block as memory area and this is not yet known during the creation of the program code you have the option of using the EN/ENO mechanism to recognize possible access errors:

SCL

```
POKE(AREA := 16#84,
     DBNUMBER := 1,
     BYTEOFFSET := 2,
     BITOFFSET := 4,
     VALUE := TRUE,
     ENO => ENO);
```

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
AREA	Input	BYTE	I, Q, M, D	The following areas can be selected: • 16#81: Input • 16#82: Output • 16#83: Bit memory • 16#84: DB • 16#2: Peripheral output (S7-1500 only)
DBNUM-BER	Input	DINT, DB_ANY	D	Number of the data block if AREA = DB, otherwise "0"
BYTEOFF-SET	Input	DINT	I, Q, M, D	Address to be written Only the 16 least significant bits are used.
BITOFFSET	Input	INT	I, Q, M, D	Bit to be written
VALUE	Input	BOOL	I, Q, M, D	Value to be written

Note

If you write the memory bit to the input, output or bit memory areas, you must assign the DBNUMBER parameter the value "0", as the instruction is invalid otherwise.

Example

The following example shows how the instruction works:

SCL

```
POKE_BOOL(AREA := "Tag_Area",
          DBNUMBER := "Tag_DBNumber",
          BYTEOFFSET := "Tag_Byte",
          BITOFFSET := "Tag_Bit",
          VALUE := "Tag_Value");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
AREA	Tag_Area	16#84
DBNUMBER	Tag_DBNumber	5
BYTEOFFSET	Tag_Byte	20
BITOFFSET	Tag_Bit	3
VALUE	Tag_Value	M0.0

The instruction overwrites the memory bit "3" in data block "5" in byte "20" with the value "M0.0".

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

POKE_BLK: Write memory area (S7-1200, S7-1500)

Description

The "Write memory area" instruction is used to write a memory area to a different standard memory area without specifying a data type.

Use of memory area 16#84: DB

If you use a data block as memory area and this is not yet known during the creation of the program code you have the option of using the EN/ENO mechanism to recognize possible access errors:

SCL

```
POKE_BLK(AREA_SRC := 16#84,
         DBNUMBER_SRC := 1,
         BYTEOFFSET_SRC := 2,
         AREA_DEST := 16#84,
         DBNUMBER_DEST := 11,
         BYTEOFFSET_DEST := 22,
         COUNT := 3,
         ENO => ENO);
```

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
AREA_SRC	Input	BYTE	I, Q, M, D	The following areas can be selected in the source memory area: • 16#81: Input • 16#82: Output • 16#83: Bit memory • 16#84: DB
DBNUMBER_SRC	Input	DINT, DB_ANY [1]	D	Number of the data block in the source memory area, if AREA = DB, otherwise "0"
BYTEOFF-SET_SRC	Input	DINT	I, Q, M, D	Address in the source memory area to be written Only the 16 least significant bits are used.

Parameters	Declaration	Data type	Memory area	Description
AREA_DEST	Input	BYTE	I, Q, M, D	The following areas can be selected in the destination memory area: • 16#81: Input • 16#82: Output • 16#83: Bit memory • 16#84: DB
DBNUMBER_DEST	Input	DINT, DB_ANY [1]	D	Number of the data block in the destination memory area, if AREA = DB, otherwise "0"
BYTEOFFSET_DEST	Input	DINT	I, Q, M, D	Address in the destination memory area to be written Only the 16 least significant bits are used.
COUNT	Input	DINT	I, Q, M, D	Number of bytes which are copied
[1] The data types of the parameters DBNUMBER_SRC and DBNUMBER_DEST must have the same data type. In other words, both tags must be either data type DINT or data type DB_ANY.				

Note

If you write the memory address to the input, output or bit memory areas, you must assign the DBNUMBER parameter the value "0", as the instruction is invalid otherwise.

Example

The following example shows how the instruction works:

SCL

```
POKE_BLK(AREA_SRC := "Tag_Source_Area",
         DBNUMBER_SRC := "Tag_Source_DBNumber",
         BYTEOFFSET_SRC := "Tag_Source_Byte");
         AREA_DEST := "Tag_Destination_Area",
         DBNUMBER_DEST := "Tag_Destination_DBNumber",
         BYTEOFFSET_DEST := "Tag_Destination_Byte",
         COUNT := "Tag_Count");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
AREA_SRC	Tag_Source_Area	16#84
DBNUMBER_SRC	Tag_Source_DBNumber	5
BYTEOFFSET_SRC	Tag_Source_Byte	20
AREA_DEST	Tag_Destination_Area	16#83
DBNUMBER_DEST	Tag_Destination_DBNumber	0
BYTEOFFSET_DEST	Tag_Destination_Byte	30
COUNT	Tag_Count	100

The instruction writes 100 byte from data block "5" starting with address "20" in the memory area of the bit memory starting at address "30".

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

READ_LITTLE: Read data in little endian format (S7-1200, S7-1500)

Description

The instruction "Read data in little endian format" is used to read data from a memory area and to write this to a single tag in the little endian byte sequence. With the little endian format, the byte with the least significant bits is saved first, which means at the lowest memory address.

The parameters SRC_ARRAY and DEST_VARIABLE are of data type VARIANT. However, there are a few restrictions regarding the data type with which the parameters can be interconnected. The VARIANT at parameter DEST_VARIABLE has to be an elementary data type. The VARIANT at the SRC_ARRAY parameter points to the memory area to be read from, and this must be an ARRAY of BYTE.

You can also use an actual parameter with variable ARRAY index for the parameters SRC_ARRAY and DEST_VARIABLE.

The operand at the POS parameter determines the position in the memory area at which reading starts.

Note

Reading tag of data type VARIANT or BOOL

If you want to read a tag to which a VARIANT points, use the "Serialize" or "Deserialize" instructions.

If you want to read a tag of the data type BOOL, use a "Slice access".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200 V4.0 and higher	S7-1500		
SRC_ARRAY	Input	ARRAY of BYTE	ARRAY of BYTE	I, Q, M, D, L	Memory area to be read from
DEST_VARIABLE	Output	Bit strings, integers, floating-point numbers, TOD, DATE, CHAR, WCHAR	Bit strings, integers, floating-point numbers, LDT, TOD, LTOD, DATE, CHAR, WCHAR	I, Q, M, D, L	Read value

Parameters	Declaration	Data type		Memory area	Description
		S7-1200 V4.0 and higher	S7-1500		
POS	InOut	DINT	DINT	I, Q, M, D, L	Determines the position at which the reading starts. The POS parameter is calculated zero-based.
Function value (RET_VAL)		INT	INT	I, Q, M, D, L	Error information

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The data type at the SRC_ARRAY parameter is not ARRAY of BYTE.
8382	The value at the POS parameter is outside the limits of the ARRAY.
8383	The value at the POS parameter is within the limits of the ARRAY but the size of the memory area exceeds the high limit of the ARRAY.

Example

The following example shows how the instruction works:

SCL

```
#TagResult := READ_LITTLE(SRC_ARRAY := #SourceField,
                          DEST_VARIABLE => #DINTVariable,
                          POS := #TagPos);
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
SRC_ARRAY	#SourceField	ARRAY[0..3] of BYTE := 16#1A, 16#2B, 16#3C, 16#4D
DEST_VARIABLE	#DINTVariable	1295788826 16#4D3C2B1A
POS	#TagPos	0 => 4

The instruction reads the integer 1_295_788_826 from the #SourceField memory area and writes it to the #DINTVariable operand in little endian format. The data type at the DEST_VARIABLE parameter specifies how many bytes are read. The quantity 4 is stored in the #TagPos operand.

See also

Overview of the valid data types (Page 2239)

Deserialize: Deserialize (Page 3290)

Serialize: Serialize (Page 3293)

Basics of SCL (Page 4991)

WRITE_LITTLE: Write data in little endian format (S7-1200, S7-1500)

Description

You use the "Write data in little endian format" instruction to write the data of a single tag in the little endian byte sequence to a memory area. With the little endian format, the byte with the least significant bits is saved first, which means at the lowest memory address.

The parameters SRC_VARIABLE and DEST_ARRAY are of data type VARIANT. However, there are a few restrictions regarding the data type with which the parameters can be interconnected. The VARIANT at the SRC_VARIABLE parameter must point to an elementary data type. The VARIANT at the DEST_ARRAY parameter points to the memory area to which the data is written, and this must be an ARRAY of BYTE.

You can also use an actual parameter with variable ARRAY index for the parameters SRC_ARRAY and DEST_VARIABLE.

The operand at the POS parameter determines the position in the memory area at which writing starts.

Note

Writing tag of data type VARIANT or BOOL

If you want to write a tag to which a VARIANT points, use the "Serialize" or "Deserialize" instructions.

If you want to write a tag of the data type BOOL, use a "Slice access".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200 V4.0 and higher	S7-1500		
SRC_VARIABLE	Input	Bit strings, integers, floating-point numbers, TOD, DATE, CHAR, WCHAR	Bit strings, integers, floating-point numbers, LDT, TOD, LTOD, DATE, CHAR, WCHAR	I, Q, M, D, L	Tag whose data are written
DEST_ARRAY	InOut	ARRAY of BYTE	ARRAY of BYTE	I, Q, M, D, L	Memory area to which the data are written
POS	InOut	DINT	DINT	I, Q, M, D, L	Determines the position at which the writing starts. The POS parameter is calculated zero-based.
Function value (RET_VAL)		INT	INT	I, Q, M, D, L	Error information

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The data type at the SRC_ARRAY parameter is not ARRAY of BYTE.
8382	The value at the POS parameter is outside the limits of the ARRAY.
8383	The value at the POS parameter is within the limits of the ARRAY but the size of the memory area exceeds the high limit of the ARRAY.

Example

The following example shows how the instruction works:

SCL

```
#TagResult := WRITE_LITTLE(SRC_VARIABLE := #DINTVariable,
                           DEST_ARRAY := #TargetField,
                           POS := #TagPos);
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
SRC_VARIABLE	#DINTVariable	1295788826 16#4D3C2B1A
DEST_ARRAY	#TargetField	ARRAY[0..10] of BYTE = 16#1A, 16#2B, 16#3C, 16#4D
POS	#TagPos	0 => 4

The instruction writes the integer 1_295_788_826 from the #DINTVariable operand to the #TargetField memory area in little endian format. The data type at the SRC_VARIABLE parameter specifies how many bytes are written. The quantity 4 is stored in the #TagPos operand.

See also

Overview of the valid data types (Page 2239)

Deserialize: Deserialize (Page 3290)

Serialize: Serialize (Page 3293)

Basics of SCL (Page 4991)

READ_BIG: Read data in big endian format (S7-1200, S7-1500)

Description

The instruction "Read data in little endian format" is used to read data from a memory area and to write this to a single tag in the big endian byte sequence. With the big endian format, the byte with the most significant bits is saved first, which means at the lowest memory address.

The parameters SRC_ARRAY and DEST_VARIABLE are of data type VARIANT. However, there are a few restrictions regarding the data type with which the parameters can be interconnected. The VARIANT at parameter DEST_VARIABLE has to be an elementary data type. The VARIANT at the SRC_ARRAY parameter points to the memory area to be read from, and this must be an ARRAY of BYTE.

You can also use an actual parameter with variable ARRAY index for the parameters SRC_ARRAY and DEST_VARIABLE.

The operand at the POS parameter determines the position in the memory area at which reading starts.

Note

Reading tag of data type VARIANT or BOOL

If you want to read a tag to which a VARIANT points, use the "Serialize" or "Deserialize" instructions.

If you want to read a tag of the data type BOOL, use a "Slice access".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200 V4.0 and higher	S7-1500		
SRC_ARRAY	Input	ARRAY of BYTE	ARRAY of BYTE	I, Q, M, D, L	Memory area to be read from
DEST_VARIABLE	Output	Bit strings, integers, floating-point numbers, TOD, DATE, CHAR, WCHAR	Bit strings, integers, floating-point numbers, LDT, TOD, LTOD, DATE, CHAR, WCHAR	I, Q, M, D, L	Read value
POS	InOut	DINT	DINT	I, Q, M, D, L	Determines the position at which the reading starts. The POS parameter is calculated zero-based.
Function value (RET_VAL)		INT	INT	I, Q, M, D, L	Error information

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The data type at the SRC_ARRAY parameter is not ARRAY of BYTE.
8382	The value at the POS parameter is outside the limits of the ARRAY.
8383	The value at the POS parameter is within the limits of the ARRAY but the size of the memory area exceeds the high limit of the ARRAY.

Example

The following example shows how the instruction works:

SCL

```
#TagResult := READ_BIG(SRC_ARRAY := #SourceField,
                       DEST_VARIABLE => #DINTVariable,
                       POS := #TagPos);
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
SRC_ARRAY	#SourceField	ARRAY[0..10] of BYTE := 16#1A, 16#2B, 16#3C, 16#4D
DEST_VARIABLE	#DINTVariable	439041101 16#1A2B3C4D
POS	#TagPos	0 => 4

The instruction reads the integer 439_041_101 from the #SourceField memory area and writes it to the #DINTVariable operand in big endian format. The data type at the DEST_VARIABLE parameter specifies how many bytes are read. The quantity 4 is stored in the #TagPos operand.

See also

Overview of the valid data types (Page 2239)

Deserialize: Deserialize (Page 3290)

Serialize: Serialize (Page 3293)

Basics of SCL (Page 4991)

WRITE_BIG: Write data in big endian format (S7-1200, S7-1500)

Description

You use the "Write data in big endian format" instruction to write the data of a single tag in the big endian byte sequence to a memory area. With the big endian format, the byte with the most significant bits is saved first, which means at the lowest memory address.

The parameters SRC_VARIABLE and DEST_ARRAY are of data type VARIANT. However, there are a few restrictions regarding the data type with which the parameters can be interconnected. The VARIANT at the SRC_VARIABLE parameter must point to an elementary data type. The VARIANT at the DEST_ARRAY parameter points to the memory area to which the data is written, and this must be an ARRAY of BYTE.

You can also use an actual parameter with variable ARRAY index for the parameters SRC_ARRAY and DEST_VARIABLE.

The operand at the POS parameter determines the position in the memory area at which writing starts.

Note

Writing tag of data type VARIANT or BOOL

If you want to write a tag to which a VARIANT points, use the "Serialize" or "Deserialize" instructions.

If you want to write a tag of the data type BOOL, use a "Slice access".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200 V4.0 and higher	S7-1500		
SRC_VARIABLE	Input	Bit strings, integers, floating-point numbers, TOD, DATE, CHAR, WCHAR	Bit strings, integers, floating-point numbers, LDT, TOD, LTOD, DATE, CHAR, WCHAR	I, Q, M, D, L	Tag whose data are written
DEST_ARRAY	InOut	ARRAY of BYTE	ARRAY of BYTE	I, Q, M, D, L	Memory area to which the data are written
POS	InOut	DINT	DINT	I, Q, M, D, L	Determines the position at which the writing starts. The POS parameter is calculated zero-based.
Function value (RET_VAL)		INT	INT	I, Q, M, D, L	Error information

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B4	The data type at the SRC_ARRAY parameter is not ARRAY of BYTE.
8382	The value at the POS parameter is outside the limits of the ARRAY.
8383	The value at the POS parameter is within the limits of the ARRAY but the size of the memory area exceeds the high limit of the ARRAY.

Example

The following example shows how the instruction works:

SCL

```
#TagResult := WRITE_BIG(SRC_VARIABLE := #DINTVariable,
                        DEST_ARRAY := #TargetField,
                        POS := #TagPos);
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
SRC_VARIABLE	#DINTVariable	439041101 16#1A2B3C4D
DEST_ARRAY	#TargetField	ARRAY[0..10] of BYTE = 16#1A, 16#2B, 16#3C, 16#4D
POS	#TagPos	0 => 4

The instruction writes the integer 439_041_101 from the #DINTVariable operand to the #TargetField memory area in big endian format. The data type at the SRC_VARIABLE parameter specifies how many bytes are written. The quantity 4 is stored in the #TagPos operand.

See also

Overview of the valid data types (Page 2239)

Deserialize: Deserialize (Page 3290)

Serialize: Serialize (Page 3293)

Basics of SCL (Page 4991)

VARIANT (S7-1200, S7-1500)

VariantGet: Read out VARIANT tag value (S7-1200, S7-1500)

Description

You can use the "Read out VARIANT tag value" instruction to read the value of the tag to which the VARIANT at the SRC parameter points and write it in the tag at the DST parameter.

The SRC parameter has the VARIANT data type. Any data type except for VARIANT can be specified at the DST parameter.

The data type of the tag specified at the DST parameter must match the data type to which the VARIANT points.

Note

To copy structures and ARRAYs, you can use the "MOVE_BLK_VARIANT instruction: Move block" instruction. For additional information, refer to "See also".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
SRC	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	Tag to be read
DST	Output	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY elements, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
VariantGet(SRC := #TagIn_Source,
           DST => "TagOut_Dest");
```

The value of the tag to which the VARIANT at the "#TagIn_Source" operand points, is read and written to the "TagOut_Dest" operand.

See also

Overview of the valid data types (Page 2239)

MOVE_BLK_VARIANT: Move block (Page 3309)

Basics of SCL (Page 4991)

VariantPut: Write VARIANT tag value (S7-1200, S7-1500)

Description

You can use the "Write VARIANT tag value" instruction to write the value of the tag at the SRC parameter to the memory at the DST parameter to which the VARIANT points.

The DST parameter has the VARIANT data type. Any data type except for VARIANT can be specified at the SRC parameter.

The data type of the tag at the SRC parameter must match the data type to which the VARIANT points.

Note

To copy structures and ARRAYs, you can use the "MOVE_BLK_VARIANT instruction: Move block" instruction. For additional information, refer to "See also".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
SRC	Input	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY elements, PLC data types	I, Q, M, D, L	I, Q, M, D, L, P	Tag to be read
DST	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	L (A declaration in the "Temp" section of the block interface is not possible.)	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
VariantPut(SRC := "TagIn_Source",
           DST := #TagIn_Dest);
```

The value of the "TagIn_Source" operand is written to the tag to which the VARIANT at the #TagIn_Dest operand points.

See also

Overview of the valid data types (Page 2239)

MOVE_BLK_VARIANT: Move block (Page 3309)

Basics of SCL (Page 4991)

CountOfElements: Get number of ARRAY elements (S7-1200, S7-1500)

Description

You use the "Get number of ARRAY elements" instruction to query how many ARRAY elements a tag to which VARIANT points has.

If it is a multi-dimensional ARRAY, the number of ARRAY elements is output as the result. (The difference between the high and low limit +1). If it is a multi-dimensional ARRAY, the number of all dimensions is output as the result.

Note

Instances

The VARIANT pointer can point to no instance and therefore also to no multi-instance or an ARRAY of multi-instances.

Note

ARRAY within a data block

If you want to query the number of the elements of an ARRAY which is located in a data block, the block attribute "Data block write-protected in the device" must not be activated in this block. Otherwise, the RET_VAL parameter returns the result "0", irrespective of how many elements the ARRAY contains.

The result is also "0" if the VARIANT tag is not an ARRAY.

If the VARIANT points to an ARRAY of BOOL, the fill elements are included in the count. (For example, 8 is returned for an ARRAY[0..1] of BOOL).

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Tag to be queried
Function value		UDINT	I, Q, M, D, L	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
IF IS_ARRAY(#Tag_VARIANTToArray) THEN
"Tag_Result" := CountOfElements(#Tag_VARIANTToArray);
```

SCL

```
END_IF;
```

If the tag to which the VARIANT points is an ARRAY, the number of ARRAY elements is output.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ARRAY[*] (S7-1200, S7-1500)

LOWER_BOUND: Read out low ARRAY limit (S7-1200, S7-1500)

Description

You can declare tags of the data type ARRAY[*] in the block interface of a function block or function. For these local tags, you can read out the limits of the ARRAY. You will need to specify the required dimension at the DIM parameter.

The "Read out ARRAY low limit" instruction is used to read out the variable low limit of the ARRAY.

Note

Availability of the instruction

The instruction is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
ARR	Input	ARRAY[*]	FB => Section In-Out FC => Sections Input and InOut	ARRAY of which the variable low limit is to be read.
DIM	Input	UDINT	I, Q, M, D, L, P or constant	Dimension of the ARRAY of which the variable low limit is to be read.
Function value		DINT	I, Q, M, D, L, P	Result

Example

The following example shows how the instruction works:

SCL

```
"Result" := LOWER_BOUND(ARR := #ARRAY_A,
                        DIM := 2);
```

The instruction reads out the variable low limit of the ARRAY #ARRAY_A from the second dimension. If the instruction is executed without errors, the result is written to operand "Result".

See also

Overview of the valid data types (Page 2239)

Basic information on ARRAY (Page 2286)

Basics of SCL (Page 4991)

UPPER_BOUND: Read out high ARRAY limit (S7-1200, S7-1500)

Description

You can declare tags of the data type ARRAY[*] in the block interface of a function block or function. For these local tags, you can read out the limits of the ARRAY. You will need to specify the required dimension at the DIM parameter.

The instruction "Read out ARRAY high limit" is used to read out the variable high limit of the ARRAY.

Note

Availability of the instruction

The instruction is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
ARR	Input	ARRAY[*]	FB => Section In-Out FC => Sections Input and InOut	ARRAY of which the variable high limit is to be read.
DIM	Input	UDINT	I, Q, M, D, L, P or constant	Dimension of the ARRAY of which the variable high limit is to be read.
Function value		DINT	I, Q, M, D, L, P	Result

Example

The following example shows how the instruction works:

SCL

```
"Result" := UPPER_BOUND(ARR := #ARRAY_A,
                        DIM := 2);
```

The instruction reads out the variable high limit of the ARRAY #ARRAY_A from the second dimension. If the instruction is executed without errors, the result is written to operand "Result".

See also

Overview of the valid data types (Page 2239)

Basic information on ARRAY (Page 2286)

Basics of SCL (Page 4991)

Legacy (S7-1500)

BLKMOV: Move block (S7-1500)

Description

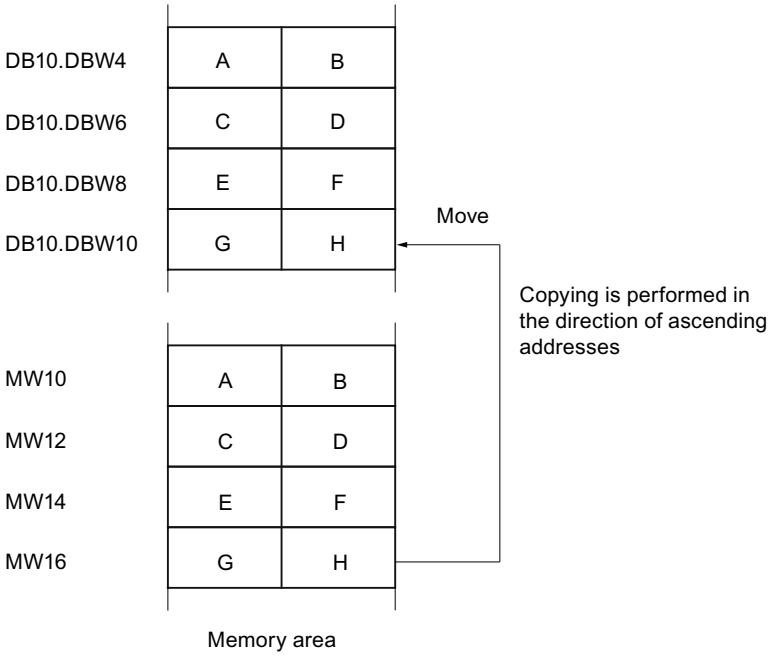
You can use the "Move block" instruction to move the content of a memory area (source area) to another memory area (destination area). The move operation takes place in the direction of ascending addresses. You use VARIANT to define the source and destination areas.

Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can also use this tag in memory areas "with optimized block access".

The following figure shows the principle of the move operation:



Consistency of source and destination data

Note that while the instruction "Move block" is being executed, the source data remains unchanged; otherwise the consistency of the destination data cannot be guaranteed.

Interruptibility

There is no limit to the nesting depth.

Memory areas

You can use the "Move block" instruction to move the following memory areas:

- Areas of a data block
- Bit memory
- Process image input
- Process image output

General rules for moving

The source and destination areas must not overlap. If the source and destination areas have different lengths, only the length of the smaller area will be moved.

If the source area is smaller than the destination area, the entire source area will be written to the destination area. The remaining bytes of the destination area remain unchanged.

If the destination area is smaller than the source area, the entire destination area will be written. The remaining bytes of the source area are ignored.

If a block of data type BOOL is moved, the tag must be addressed absolutely and the specified length of the area must be divisible by 8, otherwise the instruction cannot be executed.

Rules for moving character strings

You can use the "Move block" instruction to also move source and destination areas of the STRING data type. If only the source area is STRING data type, the characters will be moved that are actually contained in the character string. Information on the actual and maximum length is also written to the destination area. If the source and destination area are both STRING data type, the current length of the character string in the destination area is set to the number of characters actually moved.

If you want to move the information on the maximum and actual length of a character string, specify the areas in bytes in the SRCBLK and DSTBLK parameters.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
SRCBLK	Input	VARIANT	I, Q, M, D, L, P	Specifies the memory area to be moved (source area).
DSTBLK	Output 1)	VARIANT	I, Q, M, D, L, P	Specifies the memory area to which the block is to be moved (destination area).
Function value (RET_VAL)		INT	I, Q, M, D, L, P	Error information
1) The DSTBLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8092	The source or target area is only available in the load memory.
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported for the SRCBLK parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported for the DSTBLK parameter.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"Tag_RetVal" := BLKMOV(SRCBLK := P#M100.0 BYTE 10,
                       DSTBLK => P#DB1.DBX0.0 BYTE 10);
```

The instruction copies 10 bytes starting from MB100 and writes them to DB1. If an error occurs during the move operation, its error code is output in the "Tag_RetVal" tag.

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3409)

Basics of SCL (Page 4991)

UBLKMOV: Move block uninterruptible (S7-1500)

Description

You can use the "Move block uninterruptible" instruction to move the content of a memory area (source area) to another memory area (destination area). The move operation takes place in the direction of ascending addresses. You use VARIANT to define the source and destination areas.

The move operation cannot be interrupted by other operating system activities. As a result, the interrupt reaction time of the CPU can increase during the execution of the "Move block uninterruptible" instruction.

Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can also use this tag in memory areas "with optimized block access".

Memory areas

You can use the "Move block uninterruptible" instruction to move the following memory areas:

- Areas of a data block
- Bit memory
- Process image input
- Process image output

General rules for moving

The source and destination area must not overlap during the execution of the "Move block uninterruptible" instruction. If the source area is smaller than the destination area, the entire source area will be written to the destination area. The remaining bytes of the destination area remain unchanged.

If the destination area is smaller than the source area, the entire destination area will be written. The remaining bytes of the source area are ignored.

If a source or destination area defined as a formal parameter is smaller than a destination or source area specified in the SRCBLK or DSTBLK parameter, no data is transferred.

If a block of data type BOOL is moved, the tag must be addressed absolutely and the specified length of the area must be divisible by 8, otherwise the instruction cannot be executed.

You can use the "Move block uninterruptible" instruction to move a maximum of 16 KB. Note the CPU-specific restrictions for this.

Rules for moving character strings

You can use the "Move block uninterruptible" instruction to also move source and destination areas of the STRING data type. If only the source area is STRING data type, the characters will be moved that are actually contained in the character string. Information on the actual and maximum length are not written in the destination area. If the source and destination area are both STRING data type, the current length of the character string in the destination area is set to the number of characters actually moved. If areas of the STRING data type are moved, specify "1" as the area length.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
SRCBLK	Input	VARIANT	I, Q, M, D, L, P	Specifies the memory area to be moved (source area).
DSTBLK	Output [1)]	VARIANT	I, Q, M, D, L, P	Specifies the memory area to which the block is to be moved (destination area).
Function value (RET_VAL)		INT	I, Q, M, D, L, P	Error information
1) The DSTBLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code (W#16#...)	Explanation
0000	No error
8091	The source or target area is only available in the load memory.

Error code (W#16#...)	Explanation
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported for the SRCBLK parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported for the DSTBLK parameter.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"Tag_RetVal" := UBLKMOV(SRCBLK := P#M100.0 BYTE 10,
                        DSTBLK => P#DB1.DBX0.0 BYTE 10);
```

The instruction copies 10 bytes starting from MB100 and writes them to DB1. If an error occurs during the move operation, its error code is output in the "Tag_RetVal" tag.

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3409)

Basics of SCL (Page 4991)

FILL: Fill block (S7-1500)

Description

You can use the "Fill block" instruction to fill a memory area (destination area) with the content of another memory area (source area). The "Fill block" instruction moves the content of the source area to the destination area until the destination area is completely written. The move operation takes place in the direction of ascending addresses.

You use VARIANT to define the source and destination areas.

Note

You can also define the source and destination areas using the ANY data type

If you use the ANY data type, you must observe the following in connection with the STRING data type:

- In the case of an assignment of STRING (source area) using ANY after STRING (destination area), the content of the STRING is copied to the destination area repeatedly until it is full.
  Source area: 'STEP7-SCL-TIA-Portal'
  Destination area: 'STEP7-SCL-TIA-PortalSTEP7-SCL-TIA-PortalSTEP7-SCL'
- In the case of an assignment of WSTRING (source area) using ANY after WSTRING (destination area), the entire WSTRING including type information is copied only once to the destination area.
  Source area: WSTRING#'STEP7-SCL-TIA-Portal'
  Destination area: WSTRING#'STEP7-SCL-TIA-Portal'

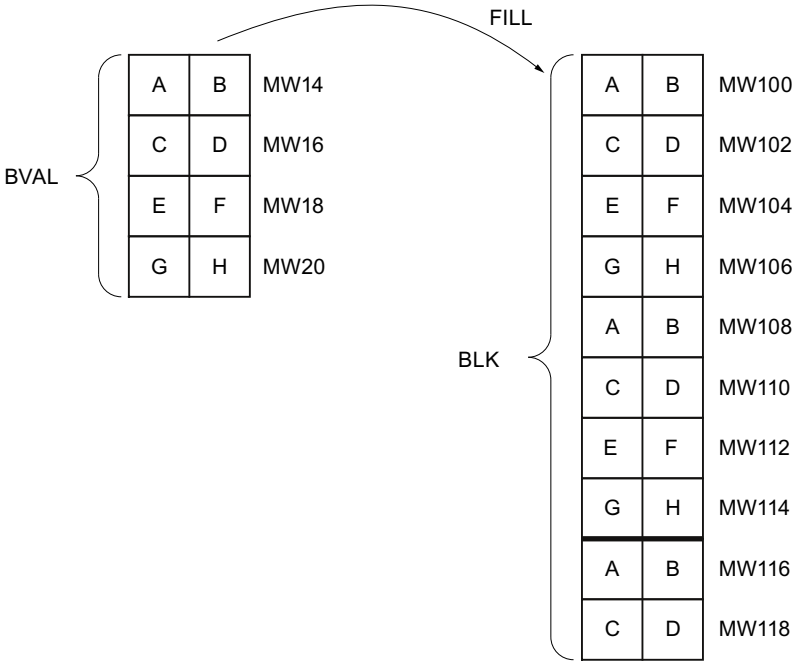
Note

The tags of the instruction can only be used within memory areas in which the "Optimized block access" attribute is not activated. This applies to data blocks (DBs), organization blocks (OBs), functions (FCs), bit memory (M), inputs (I), and outputs (Q).

If a tag of the instruction has been declared with the retentivity setting "Set in IDB", however, you can also use this tag in memory areas "with optimized block access".

For blocks with the "Optimized block access" attribute, you can use the "FILL_BLK instruction: Fill block" instruction.

The following figure shows the principle of the move operation:



Example: The contents of the range MW100 to MW118 are to be preassigned with the contents of the memory words MW14 to MW20.

Consistency of source and destination data

Note that while the instruction "Fill block" is being executed, the source data remains unchanged; otherwise the consistency of the destination data cannot be guaranteed.

Memory areas

You can use the "Fill block" instruction to move the following memory areas:

- Areas of a data block
- Bit memory
- Process image input
- Process image output

General rules for moving

The source and destination areas must not overlap. If the destination area to be preset is not an integer multiple of the length of the BVAL input parameter, the destination area is nevertheless written up to the last byte.

If the destination area to be preset is smaller than the source area, the function only copies as much data as can be written to the destination area.

If the destination or source area actually present is smaller than the assigned memory area for the source or destination area (BVAL, BLK parameters), no data is transferred.

If the ANY pointer (source or destination) is of the data type BOOL, it must be addressed absolutely and the length specified must be divisible by 8; otherwise the instruction is not executed.

If the destination area is STRING data type, the instruction writes the entire string including the administration information.

Rules for moving structures

When you transfer a structure as an input parameter you must bear in mind that the length of a structure is always oriented to an even number of bytes. The structure will need one byte of additional memory space if you declare a structure with an odd number of bytes.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
BVAL	Input	VARIANT	I, Q, M, D, L, P	Specification of the memory area (source area), the content of which is used to fill the destination area at the BLK parameter.
BLK	Output [1)]	VARIANT	I, Q, M, D, L, P	Specification of the memory area that will be filled with the content of the source area.
Function value (RET_VAL)		INT	I, Q, M, D, L, P	Error information
1) The BLK parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

For additional information on valid data types, refer to "See also".

BVAL parameter

When you transfer a structure as an input parameter, remember that the length of a structure is always based on an even number of bytes. The structure will need one byte of additional memory space if you declare a structure with an odd number of bytes.

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code (W#16#...)	Explanation
0000	No error
8092	The source or target area is only available in the load memory.
8152	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported for the BVAL parameter.
8352	The WSTRING, WCHAR, BOOL, and ARRAY of STRING data types are not supported at the BLK parameter.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"Tag_RetVal" := FILL(BVAL := P#M14.0 WORD 4,
                     BLK => P#M100.0 WORD 10);
```

The instruction copies the source area from MW14 to MW20 and fills the destination area from MW100 to MW118 with the content of the 4 words contained in the memory area in the BVAL parameter.

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3409)

Basics of SCL (Page 4991)

Conversion operations (S7-1200, S7-1500)

CONVERT: Convert value (S7-1200, S7-1500)

Description

Use the "Convert value" instruction to program explicit conversions. When the instruction is inserted, the "CONVERT" dialog opens. You specify the source data type and the destination data type of the conversion in this dialog. The source value is read and converted to the specified destination data type.

Conversion options for bit strings

BYTE and WORD bit strings cannot be selected in the instruction box. However, it is possible to specify an operand of data type DWORD or LWORD at a parameter of the instruction if the lengths of the input and output operands match. The operand is then interpreted by the data type of a bit string according to the data type of the input or output parameter and is implicitly converted. The data type DWORD is for example interpreted as DINT/UDINT and LWORD as LINT/ULINT. These conversion options are also available to you when "IEC check" is enabled.

Note

For CPUs of the S7-1500 series: The data types DWORD and LWORD can only be converted to or from data type REAL or LREAL.

During the conversion, the bit pattern of the source value is transferred unchanged, right-justified, to the target data type. If no errors occur during the conversion, the signal state of enable output ENO = 1; if an error occurs during processing, the signal state of the enable output ENO = 0.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Source type>	Input	Binary numbers, integers, floating-point numbers, times, date and time, character strings, BCD16, BCD32	I, Q, M, D, L, P or constant	Value to be converted.
<Destination type>	Output	Binary numbers, integers, floating-point numbers, times, date and time, character strings, BCD16, BCD32	I, Q, M, D, L, P or constant	Result of the conversion

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_INT" := REAL_TO_INT("Tag_REAL");
```

The following table shows how the instruction works using specific operand values:

Operand	Data type	Value
Tag_REAL	REAL	20.56
Tag_INT	INT	21

During the conversion, the value of the "Tag_REAL" operand will be rounded to the nearest integer and saved in the "Tag_INT" operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ROUND: Round numerical value (S7-1200, S7-1500)

Description

The "Round numerical value" instruction is used to round the value at input IN to the nearest integer. The instruction interprets the value at input IN as a floating-point number and converts it into an integer or floating-point number. If the input value is exactly between an even and odd number, the even number is selected.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value to be rounded.
Function value		Integers, floating-point numbers	I, Q, M, D, L, P	Result of the rounding

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := ROUND("Tag_Value");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value	
Tag_Value	1.50000000	-1.50000000
Tag_Result	2	-2

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

CEIL: Generate next higher integer from floating-point number (S7-1200, S7-1500)

Description

Use the "Generate next higher integer from floating-point number" instruction to round the value to the nearest integer. The instruction interprets the input value as floating-point number and converts it to the next higher integer. The function value can be greater than or equal to the input value.

Use the following syntax to change the data type of the instruction:

CEIL_<data type>();

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value
_<Data type>		Integers, floating-point numbers Default: DINT	-	Data type of the function value: 1. You can specify the data type of the instruction explicitly using "_". 2. If you do not specify the data type explicitly, it will be determined by the utilized tags or type-coded constants. 3. If you neither specify the data type explicitly nor specify defined tags or type-coded constants, the default data type will be used.
Function value		Integers, floating-point numbers	I, Q, M, D, L	Input value rounded up

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := CEIL("Tag_Value");
"Tag_Result2" := CEIL_REAL("Tag_Value");
```

The following table shows how the instruction works using specific operand values:

Operand	Value	
Tag_Value	0.5	-0.5
Tag_Result1	1	0
Tag_Result2	1.0	0.0

The result of the instruction is returned in the operand "Tag_Resultxy" as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

FLOOR: Generate next lower integer from floating-point number (S7-1200, S7-1500)

Description

Use the "Generate next lower integer from floating-point number" instruction to round the value of a floating point number to the next lower integer. The instruction interprets the input value as floating-point number and converts it to the next lower integer. The function value can be equal or less than the input value.

Use the following syntax to change the data type of the instruction:

FLOOR_<data type>();

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L, P	Input value
_<Data type>		Integers, floating-point numbers Default: DINT	-	Data type of the function value: 1. You can specify the data type of the instruction explicitly using "_". 2. If you do not specify the data type explicitly, it will be determined by the utilized tags or type-coded constants. 3. If you neither specify the data type explicitly nor specify defined tags or type-coded constants, the default data type will be used.
Function value		Integers, floating-point numbers	I, Q, M, D, L	Input value rounded

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := FLOOR("Tag_Value");
"Tag_Result2" := FLOOR_REAL("Tag_Value");
```

The following table shows how the instruction works using specific operand values:

Operand	Value	
Tag_Value	0.5	-0.5
Tag_Result1	0	-1
Tag_Result2	0.0	-1.0

The result of the instruction is returned in the operand "Tag_Resultxy" as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

TRUNC: Truncate numerical value (S7-1200, S7-1500)

Description

The "Truncate numerical value" instruction is used to generate an integer from the input value without rounding. The instruction selects only the integer part of the input value and returns this part without decimal places as the function value.

Use the following syntax to change the data type of the instruction:

TRUNC_<data type>();

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Expression>	Input	Floating-point numbers	I, Q, M, D, L	Input value
_<Data type>		Integers, floating-point numbers Default: DINT	-	Data type of the function value: 1. You can specify the data type of the instruction explicitly using "_". 2. If you do not specify the data type explicitly, it will be determined by the utilized tags or type-coded constants. 3. If you neither specify the data type explicitly nor specify defined tags or type-coded constants, the default data type will be used.
Function value		Integers, floating-point numbers	I, Q, M, D, L	Integer component of the input value

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result1" := TRUNC("Tag_Value1");
"Tag_Result2" := TRUNC("Tag_Value2"+"Tag_Value3");
"Tag_Result3" := TRUNC_SINT("Tag_Value4");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Value1	-1.5
Tag_Result1	-1
Tag_Value2	2.1
Tag_Value3	3.2
Tag_Result2	5
Tag_Result3	2
Tag_Value4	2.4

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SCALE_X: Scale (S7-1200, S7-1500)

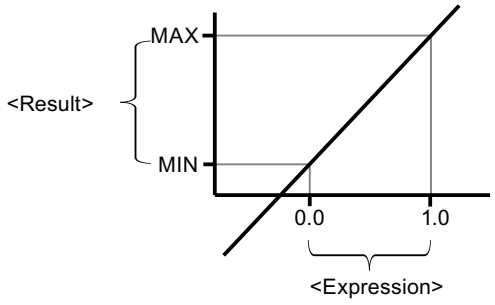
Description

Use the "Scale" instruction to scale a floating-point number by mapping it to a specific value range. You specify the value range with the MIN and MAX parameters. The result of the scaling is an integer.

Use the following syntax to change the data type of the instruction:

SCALE_X_<data type>();

The following figure shows an example of how values can be scaled:



The "Scale" instruction works with the following equation:

OUT = [VALUE * (MAX – MIN)] + MIN

Note

For more information on the conversion of analog values, refer to the respective manual.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
MIN	Input	Integers, floating-point numbers	I, Q, M, D, L	Low limit of the value range
VALUE	Input	Floating-point numbers	I, Q, M, D, L	Value to be scaled. If you enter a constant, you must declare it.
MAX	Input	Integers, floating-point numbers	I, Q, M, D, L	High limit of the value range
_<Data type>		Integers, floating-point numbers Default: INT	-	Data type of the function value: 1. You can specify the data type of the instruction explicitly using "_". 2. If you do not specify the data type explicitly, it will be determined by the utilized tags or type-coded constants. 3. If you neither specify the data type explicitly nor specify defined tags or type-coded constants, the default data type will be used.
Function value		Integers, floating-point numbers	-	Result of scaling

For additional information on valid data types, refer to "See also".

For additional information on declaring constants, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := SCALE_X(MIN := "Tag_Value1",
                         VALUE := "Tag_Real",
                         MAX := "Tag_Value2");
"Tag_Result2" := SCALE_X_REAL(MIN := "Tag_Value1",
                              VALUE := "Tag_Real",
                              MAX := "Tag_Value2");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_Real	0.5
Tag_Value1	10
Tag_Value2	30
Tag_Result1	20
Tag_Result2	20.0

See also

Overview of the valid data types (Page 2239)

Declaring global constants (Page 4854)

Basics of SCL (Page 4991)

NORM_X: Normalize (S7-1200, S7-1500)

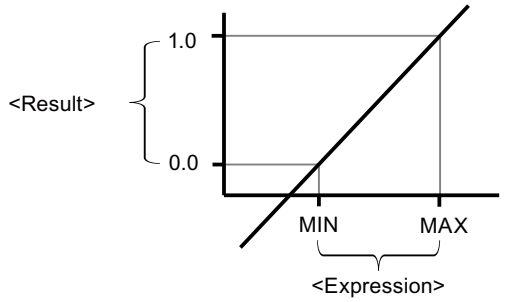
Description

You can use the instruction "Normalize" to normalize the value of the tag at the VALUE input by mapping it to a linear scale. You can use the MIN and MAX parameters to define the limits of a value range that is applied to the scale. The result at the OUT output is calculated and stored as a floating-point number depending on the location of the value to be normalized within this value range. If the value to be normalized equals the value at input MIN, the instruction returns the result "0.0". If the value to be normalized equals the value at input MAX, the instruction returns the result "1.0".

Use the following syntax to change the data type of the instruction:

NORM_X_<data type>();

The following figure shows an example of how values can be normalized:



The "Normalize" instruction works with the following equation:

OUT = (VALUE – MIN) / (MAX – MIN)

Note

For more information on the conversion of analog values, refer to the respective manual.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
MIN [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L	Low limit of the value range
VALUE [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L	Value to be normalized.
MAX [1)]	Input	Integers, floating-point numbers	I, Q, M, D, L	High limit of the value range

Parameters	Declaration	Data type	Memory area	Description
_<Data type>		Floating-point numbers Default: REAL	-	Data type of the function value: 1. You can specify the data type of the instruction explicitly using "_". 2. If you do not specify the data type explicitly, it will be determined by the utilized tags or type-coded constants. 3. If you neither specify the data type explicitly nor specify defined tags or type-coded constants, the default data type will be used.
Function value		Floating-point numbers	I, Q, M, D, L	Result of the normalization
[1] If you use constants in these three parameters, you only need to declare one of them.				

For additional information on valid data types, refer to "See also".

For additional information on declaring constants, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result1" := NORM_X(MIN := "Tag_Value1",
                        VALUE := "Tag_InputValue",
                        MAX := "Tag_Value2");
"Tag_Result2" := NORM_X_LREAL(MIN := "Tag_Value1",
                              VALUE := "Tag_InputValue",
                              MAX := "Tag_Value2");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value
Tag_InputValue	20
Tag_Value1	10
Tag_Value2	30
Tag_Result1	0.5
Tag_Result2	0.5

See also

Overview of the valid data types (Page 2239)

Declaring global constants (Page 4854)

Basics of SCL (Page 4991)

VARIANT (S7-1200, S7-1500)

VARIANT_TO_DB_ANY: Convert VARIANT to DB_ANY (S7-1200, S7-1500)

Description

You use the "Convert VARIANT to DB_ANY" instruction to query the data block number that the operand that is specified at the IN parameter addresses. This can be an instance data block or an ARRAY data block. The operand at the IN parameter has the data type VARIANT, which means you do not need to know the data type of the data block whose number is to be queried when the program is created. The data block number is read during runtime and written to the operand specified at the RET_VAL parameter.

Requirements

If the requirements are met, the instruction is executed. If the requirements are not met, "0" is output as data block number.

The output tag...	homed...	Conversion options
VARIANT	... a data block which is an instance data block of a PLC data type or a system data type (SDT).	The output tag can be converted to a data block number.
VARIANT	... a data block which is an ARRAY DB.	The output tag can be converted to a data block number.
VARIANT	... a tag with an elementary data type.	It is not possible to convert the output tag to a database number because a data block can never comprise only one elementary data type.
VARIANT	... a structure within a data block.	It is not possible to convert the output tags to a database number because it is only a part within the data block.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN	Input	VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Tag to be read. (The function value of the "DB_ANY_TO_VARIANT" instruction). You can use a local or global tag at the IN parameter.
ERR	Output	INT	I, Q, M, D, L	Error information
Function value (RET_VAL)		DB_ANY	I, Q, M, D, L	Result: Number of the DB

For additional information on valid data types, refer to "See also".

ERR parameter

The following table shows the meaning of the values of the ERR parameter:

Error code* (W#16#...)	Explanation
0000	No error
252C	The VARIANT data type at the IN parameter has the value "0" and the CPU changes to STOP mode.
80B4	The element data type stored in the ARRAY data block does not match the element data type transferred in the VARIANT.
8131	The data block does not exist, is too short, or is located in load memory.
8132	The data block is too short and not an ARRAY data block.
8150	The data type VARIANT at parameter IN provides the value "0". To receive this error message, the "Handle errors within block" block property must be activated. Otherwise the CPU changes to STOP mode and sends the error code 16#252C.
8153	The VARIANT data type at the IN parameter does not point to the start of an ARRAY data block or the length of the VARIANT does not match that of the data block.
8154	The data block has the incorrect data type.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"OutputDBNumber" := VARIANT_TO_DB_ANY(IN := #tempVARIANT,
                                      ERR := "Tag_Error");
```

The following table shows how the instruction works using specific operand values:

Parameters	Declaration in the block interface	Operand	Value
IN	Input	tempVARIANT	-
<function value>	Output	OutputDBNumber	11

The number of a data block that is specified at the tempVARIANT operand is read. Because the operand has the data type VARIANT, you do not need to know the data type of the tag during program creation. The number is written to the "OutputDBNumber" tag which has the data type DB_ANY.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

DB_ANY_TO_VARIANT: Convert DB_ANY to VARIANT (S7-1200, S7-1500)

Description

The "Convert DB_ANY to VARIANT" instruction is used to generate a VARIANT tag from a data block that meets the requirements listed below. The operand at the IN parameter has the data type DB_ANY, which means that the data block does not have to be known when the program is created. The data block number is read during runtime.

Requirements

If the requirements are met, the instruction is executed. If the requirements are not met or the data block does not exist, the value NULL is output at the RET_VAL parameter. All other accesses with the RET_VAL tag fail.

The input tag of data type ...	homed...	Conversion options
DB_ANY	...a data block which is an instance data block of a PLC data type or a system data type (SDT).	Conversion is possible
DB_ANY	...a data block which is an ARRAY DB.	Conversion is possible
DB_ANY	...a data block which is an instance data block of a function block or a global data block.	Conversion is not possible

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN	Input	DB_ANY	I, Q, M, D, L	Data block whose number is read. You can use a local or global tag at the IN parameter.
ERR	Output	INT	I, Q, M, D, L	Error information
Function value (RET_VAL) [1]		VARIANT	L (A declaration in the "Temp" section of the block interface is not possible.)	Number of the data block
1) The RET_VAL parameter is declared as Output, since the data flow into the tag. However, the tag itself must be declared as InOut in the block interface.				

For additional information on valid data types, refer to "See also".

ERR parameter

The following table shows the meaning of the values of the ERR parameter:

Error code* (W#16#...)	Explanation
0000	No error
8130	The number of the data block is "0"
8131	The data block does not exist, is too short, or is located in load memory.
8132	The data block is too short and not an ARRAY data block.
8134	The data block is write protected.
8154	The data block has the incorrect data type.
8155	The data block has an unknown data type. [1]
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

[1] The reason for the error code #8155 to be output is:

A PLC data type (UDT1) was declared and a data block (DB2) of the data type "UDT1" was created. The tag table contains a tag (3) of the data type DB_ANY. Next the instruction "DB_ANY_TO_VARIANT" was called in a program block (4) and supplied with the tag (3) at the IN parameter. During execution the instruction "DB_ANY_TO_VARIANT" returns the error code 16#8155.

Follow these steps to resolve the error code:

1. Create a function (FC5) and declare a tag of the data type VARIANT at the InOut interface.
2. Create an additional function (FC6) and call FC5 in it.
3. Create a tag (7) of the data type "UDT1" in FC6 in the Temp interface.

4. Supply the InOut interface of the FC5 with tag (7).
5. Compile and download the two blocks (FC5 and FC6) to your CPU. You do not need to call these blocks (FC5 und FC6) in the user program.

Result:

The error code 16#8155 is no longer output because the user program is now familiar with the data type.

This procedure is not necessary if you call one of the two instructions "VariantGet" or "VariantPut" after calling the instruction "DB_ANY_TO_VARIANT".

Example

The following example shows how the instruction works:

SCL

```
#tempVARIANT := DB_ANY_TO_VARIANT(IN := "InputDB",
                                  ERR := "Tag_Error");
```

The following table shows how the instruction works using specific operand values:

Parameters	Declaration in the block interface	Operand	Value
IN	Input	InputDB	11
<function value>	Temp	tempVARIANT	-

The number of any data block that is specified at the "InputDB" operand is used to generate a tag of data type VARIANT that addresses the data block. Because the operand at the IN parameter has the DB_ANY data type, the data block that will be used during runtime does not have to be known when the program is created (neither the name nor the number of the data block). Because the operand at the RET_VAL parameter has the data type VARIANT, you do not need to know the data type of the data block when the program is created.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Legacy (S7-1500)

SCALE: Scale (S7-1500)

Description

Use the "Scale" instruction to convert the integer at the IN parameter into a floating-point number that can be scaled in physical units between a low limit value and a high limit value. You can use the LO_LIM and HI_LIM parameters to specify the low limit and high limit of the value range to which the input value is scaled. The result of the instruction is output at the OUT parameter.

The "Scale" instruction works with the following equation:

OUT = [((FLOAT (IN) – K1)/(K2–K1)) * (HI_LIM–LO_LIM)] + LO_LIM

The values of the "K1" and "K2" constants are determined by the signal state on the BIPOLAR parameter. The following signal states are possible at the BIPOLAR parameter:

- Signal state "1": It is assumed that the value at the IN parameter is bipolar and in a value range between -27648 and 27648. In this case, the constant "K1" has the value -27648.0 and the constant "K2" the value +27648.0.
- Signal state "0": It is assumed that the value at the IN parameter is unipolar and in a value range between 0 and 27648. In this case, the constant "K1" has the value 0.0 and the constant "K2" the value +27648.0.

When the value at the IN parameter is greater than the value of the constant "K2", the result of the instruction is set to the value of the high limit (HI_LIM) and an error is output.

When the value at the IN parameter is less than the value of the constant "K1", the result of the instruction is set to the value of the low limit value (LO_LIM) and an error is output.

When the indicated low limit value is greater than the high limit value (LO_LIM > HI_LIM), the result is scaled inversely proportionate to the input value.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN	Input	INT	I, Q, M, D, L, P	Input value to be scaled.
HI_LIM	Input	REAL	I, Q, M, D, L, P	High limit
LO_LIM	Input	REAL	I, Q, M, D, L, P	Low limit
BIPOLAR	Input	BOOL	I, Q, M, D, L	Indicates whether the value at the IN parameter is to be interpreted as bipolar or unipolar. The parameter can assume the following values: 1: Bipolar 0: Unipolar
OUT	Output	REAL	I, Q, M, D, L, P	Result of the instruction
Function value (RET_VAL)		WORD	I, Q, M, D, L, P	Error information

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code (W#16#...)	Explanation
0000	No error
0008	The value of the parameter IN is greater than 27 648 or is less than 0 (unipolar) or -27 648 (bipolar).
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"Tag_ErrorCode" := SCALE(IN := "Tag_InputValue",
                         HI_LIM := "Tag_HighLimit",
                         LO_LIM := "Tag_LowLimit",
                         BIPOLAR := "Tag_Bipolar",
                         OUT => "Tag_Result");
```

The error information is returned in the operand "Tag_ErrorCode" as a function value.

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_InputValue	22
HI_LIM	Tag_HighLimit	100.0
LO_LIM	Tag_LowLimit	0.0
BIPOLAR	Tag_Bipolar	1
OUT	Tag_Result	50.03978588
RET_VAL	Tag_ErrorCode	W#16#0000

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3409)

Basics of SCL (Page 4991)

UNSCALE: Unscale (S7-1500)

Description

You use the "Unscale" instruction to unscale the floating-point number in the IN parameter into physical units between a low limit and a high limit and convert it into an integer. You use the LO_LIM and HI_LIM parameters to specify the low limit and high limit of the value range to which the input value is unscaled. The result of the instruction is output at the OUT parameter.

The "Unscale" instruction works with the following equation:

OUT = [((IN–LO_LIM)/(HI_LIM–LO_LIM)) * (K2–K1)] + K1

The values of the "K1" and "K2" constants are determined by the signal state at the BIPOLAR parameter. The following signal states are possible at the BIPOLAR parameter:

- Signal state "1": It is assumed that the value at the IN parameter is bipolar and in a value range between -27648 and 27648. In this case, the constant "K1" has the value -27648.0 and the constant "K2" the value +27648.0.
- Signal state "0": It is assumed that the value at the IN parameter is unipolar and in a value range between 0 and 27648. In this case, the constant "K1" has the value 0.0 and the constant "K2" the value +27648.0.

When the value at parameter IN is not within the limits defined by HI_LIM and LO_LIM, an error is output and the result is set to the nearest limit.

When the indicated low limit value is greater than the high limit value (LO_LIM > HI_LIM), the result is scaled in reverse proportion to the input value.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN	Input	REAL	I, Q, M, D, L, P	Input value to be unscaled to an integer value.
HI_LIM	Input	REAL	I, Q, M, D, L, P	High limit
LO_LIM	Input	REAL	I, Q, M, D, L, P	Low limit
BIPOLAR	Input	BOOL	I, Q, M, D, L	Indicates whether the value at the IN parameter is to be interpreted as bipolar or unipolar. The parameter can assume the following values: 1: Bipolar 0: Unipolar
OUT	Output	INT	I, Q, M, D, L, P	Result of the instruction
Function value (RET_VAL)		WORD	I, Q, M, D, L, P	Error information

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code (W#16#...)	Explanation
0000	No error
0008	The value of the IN parameter is greater than the value of the high limit (HI_LIM) or less than the value of the low limit (LO_LIM).
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. You can find information on switching the display formats here: AUTOHOTSPOT	

Example

The following example shows how the instruction works:

```
"Tag_ErrorCode" := UNSCALE(IN := "Tag_InputValue",
                           HI_LIM := "Tag_HighLimit",
                           LO_LIM := "Tag_LowLimit",
                           BIPOLAR := "Tag_Bipolar",
                           OUT => "Tag_Result");
```

The error information is returned in the operand "Tag_ErrorCode" as a function value.

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_InputValue	50.03978588
HI_LIM	Tag_HighLimit	100.0
LO_LIM	Tag_LowLimit	0.0
BIPOLAR	Tag_Bipolar	1
OUT	Tag_Result	22
RET_VAL	Tag_ErrorCode	W#16#0000

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3409)

Basics of SCL (Page 4991)

Program control operations (S7-1200, S7-1500)

IF: Run conditionally (S7-1200, S7-1500)

Description

The instruction "Run conditionally" branches the program flow depending on a condition. The condition is an expression with Boolean value (TRUE or FALSE). Logical expression or comparative expressions can be stated as conditions.

When the instruction is executed, the stated expressions are evaluated. If the value of an expression is TRUE, the condition is fulfilled; if the value is FALSE, it is not fulfilled.

Parameters

Depending on the type of branch, you can program the following forms of the instruction:

- Branch through IF:
  IF <condition> THEN <instructions>
  END_IF;

If the condition is satisfied, the instructions programmed after the THEN are executed. If the condition is not satisfied, the execution of the program continues with the next instruction after the END_IF.

- Branch through IF and ELSE:
  IF <condition> THEN <instructions1>
  ELSE <Instructions0>
  END_IF;

If the condition is satisfied, the instructions programmed after the THEN are executed. If the condition is not satisfied, the instructions programmed after the ELSE are executed. Then the execution of the program continues with the next instruction after the END_IF.

- Branch through IF, ELSIF and ELSE:
  IF <condition1> THEN <instructions1>
  ELSIF <condition2> THEN <instruction2>
  ELSE <Instructions0>
  END_IF;

If the first condition (<Condition1>) is satisfied, the instructions (<Instructions1>) after the THEN are executed. After execution of the instructions, the execution of the program continues after the END_IF.

If the first condition is not satisfied, the second condition (<Condition2>) is checked. If the second condition (<Condition2>) is fulfilled, the instructions (<Instructions2>) after the THEN are executed. After execution of the instructions, the execution of the program continues after the END_IF.

If none of the conditions are fulfilled, the instructions (<Instructions0>) after ELSE are executed followed by the execution of the program after END_IF.

You can nest as many combinations of ELSIF and THEN as you like within the IF instruction. The programming of an ELSE branch is optional.

The syntax of the IF instruction consists of the following parts:

Parameters	Data type	Memory area	Description
<Condition>	BOOL	I, Q, M, D, L	Expression to be evaluated
<Instructions>	-		Instructions to be executed with satisfied condition. An exception are instructions programmed after the ELSE. These are executed if no condition within the program loop is satisfied.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
IF "Tag_1" = 1
THEN "Tag_Value" := 10;
ELSIF "Tag_2" = 1
THEN "Tag_Value" := 20;
ELSIF "Tag_3" = 1
THEN "Tag_Value" := 30;
ELSE "Tag_Value" := 0;
END_IF;
```

The following table shows how the instruction works using specific operand values:

Operand	Value			
Tag_1	1	0	0	0
Tag_2	0	1	0	0
Tag_3	0	0	1	0
Tag_Value	10	20	30	0

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

CASE: Create multiway branch (S7-1200, S7-1500)

Description

The instruction "Create multiway branch" executes one of several instruction sequences depending on the value of a numerical expression.

The value of the expression must be an integer. When the instruction is executed, the value of the expression is compared with the values of several constants. If the value of the

expression agrees with the value of a constant, the instructions programmed directly after this constant are executed. The constants can assume the following values:

- An integer (for example, 5)
- A range of integers (for example 15...20)
- An enumeration consisting of integers and ranges (for example, 10, 11, 15...20)

You can declare the instruction as follows:

CASE <Tag> OF

<Constant1>: <Instructions1>;

<Constant2>: <Instructions2>;

<ConstantX>: <InstructionsX>; // X >= 3

ELSE <Instructions0>;

END_CASE;

Parameters

The following table shows the parameters of the instruction:

Parameters	Data type	Memory area	Description
<Tag>	Integers	I, Q, M, D, L	Value which is compared to the programmed constant values.
<Constant>	Integers	-	Constant values which form the condition for the execution of an instruction sequence. The constants can assume the following values: • An integer (for example, 5) • A range of integers (for example 15...20) • An enumeration consisting of integers and ranges (for example, 10, 11, 15...20)
<Instruction>	-	-	Any instructions which are executed if the value of the expression agrees with the value of a constant. An exception are instructions programmed after the ELSE. These instructions are executed if the values do not agree.

For additional information on valid data types, refer to "See also".

If the value of the expression agrees with the value of the first constant (<Constant1>), the instructions (<Instructions1>) which are programmed directly after the first constant are executed. Program execution subsequently resumes after the END_CASE.

If the value of the expression does not agree with the value of the first constant (<Constant1>), this value is compared to the value of the constant which is programmed next. In this way, the CASE instruction is executed until the values agree. If the value of the expression does not correspond to any of the programmed constant values, the instructions (<Instructions0>) which are programmed after the ELSE are executed. ELSE is an optional part of the syntax and can be omitted.

The CASE instruction can also be nested by replacing an instruction block with CASE. END_CASE represents the end of the CASE instruction.

Example

The following example shows how the instruction works:

```
SCL
CASE "Tag_Value" OF
     0 :
          "Tag_1" := 1;
     1,3,5 :
          "Tag_2" := 1;
     6...10 :
          "Tag_3" := 1;
     16,17,20...25 :
          "Tag_4" := 1;
     ELSE
          "Tag_5" := 1;
END_CASE;
```

The following table shows how the instruction works using specific operand values:

Operand	Values				
Tag_Value	0	1, 3 , 5	6, 7, 8, 9, 10	16, 17, 20, 21, 22, 23, 24, 25	2
Tag_1	1	-	-	-	-
Tag_2	-	1	-	-	-
Tag_3	-	-	1	-	-
Tag_4	-	-	-	1	-
Tag_5	-	-	-	-	1
1: The operand is set to the signal state "1". -: The signal state of the operand remains unaltered.					

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

FOR: Run in counting loop (S7-1200, S7-1500)

Description

The instruction "Run in counting loop" causes repeated execution of a program loop until a loop variable lies within a specified value range.

Program loops can also be nested. Within a program loop, you can program additional program loops with other loop variables.

The current continuous run of a program loop can be ended by the instruction "Recheck loop condition" (CONTINUE). The instruction "Exit loop immediately" (EXIT) ends the entire loop execution. For additional information on this topic, refer to "See also".

Note

Information on the number of runs and the run tag

The number of runs cannot be changed while the program is running.

For performance reasons, the run tag should be declared in the block interface in the "Temp" section. In addition, the run tag should not be changed while the program is running. Otherwise the instruction may return an undefined result depending on the implementation.

Limits for FOR statements

To program "safe" FOR statements that do not run endlessly, observe the following rules and limits:

FOR <Run_tag> := <Start_value> TO <End_value> BY <Increment> DO <Instructions>;

END_FOR;

If ...	... then	Note
Start value < End value	End value < (PMAX increment)	Run tag runs in positive direction
Start value > End value AND increment < O	End value > (NMAX increment)	Run tag runs in negative direction

Limits

Different limits apply to the possible data types:

Data type	PMAX	NMAX
Run tag of type SINT	127	-128
Run tag of type INT	32767	-32768
Run tag of type DINT	2147483647	-2147483648
Run tag of type LINT	9223372036854775807	-9223372036854775808

Parameters

The following table shows the parameters of the instruction:

Parameters	Data type		Memory area	Description
	S7-1200	S7-1500		
<Run tag>	SINT, INT, DINT	SINT, INT, DINT, LINT	I, Q, M, D, L	Operand whose value is evaluated with the loop execution. The data type of the loop variable determines the data type of the other parameters.
<Start value>	SINT, INT, DINT	SINT, INT, DINT, LINT	I, Q, M, D, L	Expression whose value is allocated at the start of the loop execution of the loop variables.

Parameters	Data type		Memory area	Description
	S7-1200	S7-1500		
<End value>	SINT, INT, DINT	SINT, INT, DINT, LINT	I, Q, M, D, L	Expression whose value defines the last run of the program loop. The value of the loop variable is checked before each loop: • End value not reached: The instructions according to DO are executed • End value is reached: The FOR loop is executed one last time • End value exceeded: The FOR loop is completed An alteration to the end value is not permitted during execution of the instruction.
<Increment>	SINT, INT, DINT	SINT, INT, DINT, LINT	I, Q, M, D, L	Expression by whose value the loop variable is increased (positive increment) or decreased (negative increment) after each loop. Specification of the increment is optional. If no increment is given, the value of the loop variable is increased by 1 after each loop. An alteration of the increment is not permitted during execution of the instruction.
<Instructions>	-	-		Instructions which are carried out with each loop, as long as the value of the loop variable lies within the value range. The value range is defined by the start and end values.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
FOR i := 2 TO 8 BY 2
    DO "a_array[i] := "Tag_Value"*"b_array[i]";
END_FOR;
```

The operand "Tag_Value" is multiplied with the elements (2, 4, 6, 8) of the ARRAY tag "b_array". The result is read in to the elements (2, 4, 6, 8) of the ARRAY tag "a_array".

See also

CONTINUE: Recheck loop condition (Page 3393)

EXIT: Exit loop immediately (Page 3394)

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

WHILE: Run if condition is met (S7-1200, S7-1500)

Description

The instruction "Run if condition is met" causes a program loop to be repeatedly executed until the implementation condition is satisfied. The condition is an expression with Boolean value (TRUE or FALSE). Logical expression or comparative expressions can be stated as conditions.

When the instruction is executed, the stated expressions are evaluated. If the value of an expression is TRUE, the condition is fulfilled; if the value is FALSE, it is not fulfilled.

Program loops can also be nested. Within a program loop, you can program additional program loops with other loop variables.

The current continuous run of a program loop can be ended by the instruction "Recheck loop condition" (CONTINUE). The instruction "Exit loop immediately" (EXIT) ends the entire loop execution. For additional information on this topic, refer to "See also".

You can declare the instruction as follows:

WHILE <Condition> DO <Instructions>;

END_WHILE;

Parameters

The following table shows the parameters of the instruction:

Parameters	Data type	Memory area	Description
<Condition>	BOOL	I, Q, M, D, L	Expression which is evaluated before each loop.
<Instruc-tions>	-		Instructions to be executed with satisfied condition. If the condition has not been satisfied, program execution continues after END_WHILE.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
WHILE
      "Tag_Value1" <> "Tag_Value2"
      DO "Tag_Result"
      := "Tag_Input";
END_WHILE;
```

As long as the values of the operands "Tag_Value1" and "Tag_Value2" do not match, the value of the operand "Tag_Input" is allocated to the operand "Tag_Result".

See also

EXIT: Exit loop immediately (Page 3394)

CONTINUE: Recheck loop condition (Page 3393)

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

REPEAT: Run if condition is not met (S7-1200, S7-1500)

Description

The instruction "Run if condition is not met" causes a program loop to be repeatedly executed until a termination condition is met. The condition is an expression with Boolean value (TRUE or FALSE). Logical expression or comparative expressions can be stated as conditions.

When the instruction is executed, the stated expressions are evaluated. If the value of an expression is TRUE, the condition is fulfilled; if the value is FALSE, it is not fulfilled.

The instructions are executed once, even if the termination condition is fulfilled.

Program loops can also be nested. Within a program loop, you can program additional program loops with other loop variables.

The current continuous run of a program loop can be ended by the instruction "Recheck loop condition" (CONTINUE). The instruction "Exit loop immediately" (EXIT) ends the entire loop execution. For additional information on this topic, refer to "See also".

You can declare the instruction as follows:

REPEAT <instructions>;

UNTIL <condition> END_REPEAT;

Parameters

The following table shows the parameters of the instruction:

Parameters	Data type	Memory area	Description
<Instruc-tions>	-		Instructions that are executed as long as the programmed condition has the value FALSE. The instructions are executed once, even if the termination condition is fulfilled.
<Condition>	BOOL	I, Q, M, D, L	Expression which is evaluated after each loop. If the expression has the value FALSE, the program loop is executed once again. If the expression has the value TRUE, the program loop continues after END_REPEAT.

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
REPEAT "Tag_Result"
        := "Tag_Value";
UNTIL "Tag_Error"
END_REPEAT;
```

As long as the value of the operand "Tag_Error" has the signal state "0", the value of the operand "Tag_Value" is allocated to the operand "Tag_Result".

See also

CONTINUE: Recheck loop condition (Page 3393)

EXIT: Exit loop immediately (Page 3394)

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

CONTINUE: Recheck loop condition (S7-1200, S7-1500)

Description

The "Recheck loop condition" instruction ends the current program run of a FOR, WHILE or REPEAT loop.

After execution of the instruction, the conditions for the continuation of the program loop are evaluated again. The instruction affects the program loop which directly contains the instruction.

Example

The following example shows how the instruction works:

SCL

```
FOR i
     := 1 TO 15 BY 2 DO
     IF (i < 5) THEN
     CONTINUE;
END_IF;
      "DB10".Test[i] := 1;
END_FOR;
```

For additional information on valid data types, refer to "See also".

If the condition i < 5 is satisfied, the subsequent value allocation ("DB10".Test[i] := 1) will not be executed. The run variable (i) is increased by the increment of "2" and checked to see whether its current value lies in the programmed value range. If the run variable lies in the value range, the IF condition is evaluated again.

If the condition i < 5 is not satisfied, the subsequent value allocation ("DB10".Test[i] := 1) will be executed and a new loop started. In this case, the run variable is also increased by the increment "2" and checked.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

EXIT: Exit loop immediately (S7-1200, S7-1500)

Description

The instruction "Exit loop immediately" cancels the execution of a FOR, WHILE or REPEAT loop at any point regardless of conditions. The execution of the program is continued after the end of the loop (END_FOR, END_WHILE, END_REPEAT).

The instruction affects the program loop which directly contains the instruction.

Example

The following example shows how the instruction works:

SCL

```
FOR i := 15 TO 1 BY -2 DO
IF (i < 5)
THEN EXIT;
END_IF;
"DB10".Test[i] := 1;
END_FOR;
```

For additional information on valid data types, refer to "See also".

If the condition i < 5 is satisfied, then the execution of the loop will be canceled. Program execution resumes after the END_FOR.

If the condition i < 5 is not satisfied, the subsequent value allocation ("DB10".Test[i] :=1) will be executed and a new loop started. The run tag (i) is decreased by the increment of "-2" and it is checked whether its current value lies in the programmed value range. If the (i) run variable lies within the value range, the IF condition is evaluated again.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

GOTO: Jump (S7-1200, S7-1500)

Description

Use the instruction "Jump" to resume the execution of a program at a given point marked with a jump label.

The jump labels and the instruction "Jump" must be in the same block. The name of a jump label can only be assigned once within a block. Each jump label can be the target of several jump instructions.

A jump from the "outside" into a program loop is not permitted, but a jump from a loop to the "outside" is possible.

Adhere to the following grammatical rules for jump labels:

- Letters (a to z, A to Z)
- A combination of letters and numbers. Check that the order is correct, i.e. first the letters, then the numbers (a - z, A - Z, 0 - 9).
- Special characters or a combination of letters and numbers must not be used in reverse order, i.e. first the numbers and then the letters (0-9, a - z, A - Z).

You can declare the instruction as follows:

GOTO <jump label>;

...

<Jump label>: <Instructions>

Parameters

The following table shows the parameters of the instruction:

Parameters	Data type	Description
<jump label>	-	Jump label to be jumped to
<instructions>	-	Instructions which are executed after the jump.

Example

The following example shows how the instruction works:

SCL

```
CASE "Tag_Value" OF
1 : GOTO MyLABEL1;
2 : GOTO MyLABEL2;
3 : GOTO MyLABEL3;
ELSE GOTO MyLABEL4;
END_CASE;
MyLABEL1: "Tag_1" := 1;
MyLABEL2: "Tag_2" := 1;
MyLABEL3: "Tag_3" := 1;
MyLABEL4: "Tag_4" := 1;
```

Depending on the value of the "Tag_Value" operand, the execution of the program will resume at the point identified by the corresponding jump label. If the operand "Tag_Value" has the value 2, for example, program execution will resume at the jump label "MyLABEL2". The program line identified by the jump label "MyLABEL1" will be skipped in this case.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

RETURN: Exit block (S7-1200, S7-1500)

Description

The instruction "Exit block" exits the program execution in the currently edited block and continues in the calling block.

The instruction can be omitted at the end of the block.

Example

The following example shows how the instruction works:

SCL

```
IF "Tag_Error" <>0 THEN RETURN;
END_IF;
```

If the signal state of the "Tag_Error" operand is not zero, execution of the program ends in the block currently being processed.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

(*...*): Insert a comment section (S7-1200, S7-1500)

Description

You can use the "Insert a comment section" instruction to add a comment section. The text inside the parentheses "(*...*)" is handled as a comment.

Example

The following example shows how the instruction works:

SCL

```
(*This is a comment section.*)
```

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

REGION: Structure program code (S7-1200, S7-1500)

Description

You can use the instruction "Structure program code" to structure the program code in an SCL block by dividing it into different areas.

For additional information on working with areas, please refer to:Working with regions (Page 5026)

You can declare the instruction as follows:

REGION <Name>

<Instructions>

END_REGION

Parameters

The following table shows the parameters of the instruction:

Parameters	Data type	Memory area	Description
REGION	-	-	Keyword with which the area starts
<Name>	-	-	Text designating the REGION
<Instructions>	-	-	Program code that is preceded and succeeded by the REGION
END_REGION	-	-	Keyword with which the area ends

Example

The following example shows how the instruction works:

SCL

```
REGION Feeder System
     // Structures the source code for the feeder system
   IF "Variable_1" = 0 THEN
      "Variable_2" := 10;
   END_IF;
END_REGION
```

The Feeder System area comprises the complete program code.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Runtime control (S7-1200, S7-1500)

ENDIS_PW: Limit and enable password legitimation (S7-1200, S7-1500)

Description

You use the "Limit and enable password legitimation" instruction to specify whether passwords configured for the CPU are legitimized or not. In this way, you can prevent legitimized connections, even if the correct password is known.

When you call the instruction and the REQ parameter has the signal state "0", the currently set state is displayed at the output parameters. If changes have been made to the input parameters, these are not transferred to the output parameters.

When the instruction is called and the REQ parameter has the signal state "1", the signal state is taken from the input parameters (F_PWD, FULL_PWD, R_PWD, HMI_PWD).

- If signal state "0" is present, authorization via password is not allowed.
- If signal state "1" is present, the password can be used.

Disable or enabling of the passwords can be allowed or prohibited individually. For example, all passwords can be prohibited except the fail-safe password. You can thus limit access options to a small user group. The output parameters (F_PWD_ON, FULL_PWD_ON, R_PWD_ON, HMI_PWD_ON) always show the current status of the password use, regardless of the REQ parameter.

Passwords that are not configured must have the signal state TRUE at the input and return the signal state TRUE at the output. The fail-safe password can only be configured for an F-CPU and must therefore always be interconnected with the signal state TRUE in a standard CPU. If the instruction returns an error, the call has no effect and the previous lock is still in effect.

Disabled passwords can be enabled again under the following conditions:

- The CPU is reset to its factory settings.
- The front panel of the S7-1500 CPU supports a dialog that allows you to navigate to the appropriate menu in which the passwords can be enabled again.
- When you call the "Limit and enable password legitimation" instruction, the input parameter of the desired password has the signal state "1".
- Set the mode selector to STOP. The restriction to password legitimation is re-established as soon as the switch is set back to RUN.
- Plugging an empty memory card (transfer card or program card) into an S7-1200 CPU.
- The transition from POWER OFF to POWER ON disables the protection in the S7-1200 CPU. The "Limit and enable password legitimation" instruction must be called once again in the program (e.g. in the startup OB).

Note

The "Limit and enable password legitimation" instruction blocks access from HMI systems if the HMI password has not been enabled.

Note

Existing legitimized connections retain their access rights and you cannot use the "Limit and enable password legitimation" instruction to restrict them.

Preventing accidental lockout of an S7-1500 CPU

The settings can be made on the front panel of the CPU and the CPU saves the most recent setting.

To prevent accidental lockout, the protection can be disabled by setting the mode switch to STOP with an S7-1500 CPU. The protection is automatically enabled again by setting mode selector to RUN without having to call the "Limit and enable password legitimation" instruction again or taking additional actions on the front panel.

Preventing accidental lockout of an S7-1200 CPU

Because the S7-1200 CPU does not have a mode selector, protection is disabled with POWER OFF - POWER ON. This means that it is possible and advisable to prevent accidental lockout with the help of specific program sequences within your program.

To do so, program a time control either by means of a cyclic interrupt OB or a timer in the Main OB (OB 1). This gives you the option of calling the "Limit and enable password legitimation" instruction in the respective OB (e.g., OB 1 or OB 35) relatively quickly after a transition from POWER OFF to POWER ON and the associated disabling of the protection. Call the instruction in the startup OB (OB 100) to keep the time window in which the instruction is inactive and there are no limitations for password legitimation relatively small. This procedure will give you the best possible protection from unauthorized access.

If there has been an accidental lockout, you can skip the call in the startup OB (by querying an input parameter, for example) and you have the set time (for example, 10 seconds up to 1 minute) to establish a connection to the CPU before the lock becomes active once again.

If there is no timer in your program code and a lockout has occurred, insert an empty transfer card or an empty program card into the CPU. The empty transfer card or program card deletes the internal load memory of the CPU. You must then download the user program once again from STEP 7 to the CPU.

Procedure for lost passwords with an S7-1200 CPU

If you have lost the password for a password-protected S7-1200 CPU, delete the password-protected program with an empty transfer card or program card. The empty transfer card or program card deletes the internal load memory of the CPU. You can then load a new user program from STEP 7 Basic in the CPU.

⚠ WARNING

Inserting an empty transfer card

When you insert a transfer card in a CPU during runtime, the CPU goes to STOP mode. If operating states are unstable, controllers may fail and thus cause the uncontrolled operation of the controlled devices. This leads to an unforeseen operation of the automation system which can cause deadly or serious injuries and/or damage to property.

Once you have pulled the transfer card, the content of the transfer card is available in the internal load memory. Make sure that the card does not include a program at this point.

⚠ WARNING

Inserting an empty program card

When you insert a program card in a CPU during runtime, the CPU goes to STOP mode. If operating states are unstable, controllers may fail and thus cause the uncontrolled operation of the controlled devices. This leads to an unforeseen operation of the automation system which can cause deadly or serious injuries and/or damage to property.

Make sure that the program card is empty. The internal load memory is copied to the empty program card. Once you have pulled the previously empty program card, the internal load memory is empty.

You must remove the transfer card or program card before you put the CPU into RUN mode.

Effects of password use on the operating modes

The following table shows what effects the password use by means of the "Limit and enable password legitimation" instruction has on the operating modes and the corresponding user actions.

Action	Password protection by means of the instruction
Basic state after • Operating mode switch to STOP • Reset memory manually (PG, switch, change of MC (Motion Control)) • Reset to factory setting	Not activated (no restrictions)
Basic state after POWER ON	• S7-1200 CPU: The lock is disabled and the instruction must be called once again in the program (e.g., in the startup OB). • S7-1500 CPU: Enabled (if a lock was activated before POWER OFF). The option of not allowing passwords is retentive.
Operating mode transition RUN/STARTUP/HOLD -> STOP (by terminating the instruction, an error or communication) or STOP -> STARTUP/RUN/HOLD	Activated Passwords still cannot be used.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	When the REQ parameter has the signal state "0", the currently set signal state of the passwords is queried.
F_PWD	Input	BOOL	I, Q, M, D, L or constant	Read/write access including fail-safe • F_PWD = "0": Do not allow password • F_PWD = "1": Allow password
FULL_PWD	Input	BOOL	I, Q, M, D, L or constant	Read/write access • FULL_PWD = "0": Do not allow password • FULL_PWD = "1": Allow password

Parameters	Declaration	Data type	Memory area	Description
R_PWD	Input	BOOL	I, Q, M, D, L or constant	Read access • R_PWD = "0": Do not allow password • R_PWD = "1": Allow password
HMI_PWD	Input	BOOL	I, Q, M, D, L or constant	HMI access • HMI_PWD = "0": Do not allow password • HMI_PWD = "1": Allow password
F_PWD_ON	Output	BOOL	I, Q, M, D, L	Read/write access including fail-safe • F_PWD_ON = "0": Password not allowed • F_PWD_ON = "1": Password allowed
FULL_PWD_ON	Output	BOOL	I, Q, M, D, L	Read/write access status • FULL_PWD_ON = "0": Password not allowed • FULL_PWD_ON = "1": Password allowed
R_PWD_ON	Output	BOOL	I, Q, M, D, L	Read-access status • R_PWD_ON = "0": Password not allowed • R_PWD_ON = "1": Password allowed
HMI_PWD_ON	Output	BOOL	I, Q, M, D, L	HMI-access status • HMI_PWD_ON = "0": Password not allowed • HMI_PWD_ON = "1": Password allowed
Function value (RET_VAL)		WORD	I, Q, M, D, L	Error information

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
8090	The "Limit and enable password legitimation" instruction is not supported
80D0	The password for fail-safe is not configured. The signal state must be TRUE for standard CPUs.
80D1	The read/write access is not configured

Error code* (W#16#...)	Explanation
80D2	The read access is not configured
80D3	The HMI access is not configured
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := ENDIS_PW(REQ := 0,
                         F_PWD := 0,
                         FULL_PWD := 0,
                         R_PWD := 1,
                         HMI_PWD := 0,
                         F_PWD_ON => "Status_F_PWD",
                         FULL_PWD_ON => "Status_FULL_PWD",
                         R_PWD_ON => "Status_R_PWD",
                         HMI_PWD_ON => "Status_HMI_PWD");
```

The following table shows how the instruction works using specific operand values:

Operand	Data type	Value
REQ	BOOL	0
F_PWD	BOOL	0
FULL_PWD	BOOL	0
R_PWD	BOOL	1
HMI_PWD	BOOL	0
Status_F_PWD	BOOL	0
Status_FULL_PWD	BOOL	0
Status_R_PWD	BOOL	1
Status_HMI_PWD	BOOL	0

The instruction is executed because the REQ operand has the signal state "1". The R_PWD operand has the signal state "1", which means that read access is allowed when the assigned password is entered. The R_PWD_ON status operand also has the signal state "1", thereby signaling that the R_PWD operand is activated.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

PC systems (S7-1200, S7-1500)

SHUT_DOWN: Shut down target system (S7-1200, S7-1500)

Description

You can use the instruction "SHUT_DOWN: Shutdown target system" to shut down the PC-based automation system and restart the S7 software controller CPU 150xS or Windows on the PC-based automation system.

The instruction is available in the "Instructions" task card under Basic instructions > Program control > Runtime control.

A restart is, for example, advisable in the following situations:

- An industrial UPS (uninterruptible power supply) reports a power failure via a digital input.
- Windows stops responding or shows a "blue screen".
- Too many error OBs are called in the user program.

Parameter

The following table shows the parameters of the "Shutdown target system" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	UINT	I, Q, M, D, L or constant	MODE = 1: Shut down CPU 150xS and Windows The CPU goes to STOP and saves the retentive data. Then the CPU and Windows are shut down. The system must be restarted manually. MODE = 2: Restart CPU 150xS The CPU goes to STOP and saves the retentive data. Then the CPU is shut down and restarted. MODE = 3: Restart Windows The CPU remains in RUN. Windows is restarted (as of TIA Portal V14, MODE 3 is only retained for backward compatibility. We recommend using MODE 4 or MODE 5). MODE = 4: Windows is shut down and restarted correctly. The CPU remains in RUN. MODE = 5: Restart Windows (Comparable with MODE 3; Exception: MODE 5 should only be used if Windows has crashed).
COM-MENT	Input	STRING	I, Q, M, D, L	With Mode = 1, 3 and 4, you can specify a reason for the restart. The reason is output in the Windows Event Log.
Ret_Val	Return	WORD	Q, M, D, L	Ret_Val = 0: No faults Ret_Val = 8090: The value passed to MODE is not supported. Ret_Val = 8091: Windows does not respond to the call of the Shut_Down instruction (only valid for Mode 3 and 4). Ret_Val = 8092: Please contact SIMATIC Customer Support if this error occurs (only valid for Mode 3 and 4).

RE_TRIGR: Restart cycle monitoring time (S7-1200, S7-1500)

Description

The "Restart cycle monitoring time" instruction is used to restart the cycle monitoring time of the CPU. The cycle monitoring time then restarts with the time you have set in the CPU configuration.

If the instruction is called in a block with a higher priority, such as a hardware interrupt or diagnostic error interrupt, the instruction is not executed and the enable output ENO returns signal state "0".

The instruction executes completely within a time span (10 times the maximum program cycle), regardless of the number of calls. Once this time has expired, the program cycle can no longer be prolonged.

Calling the instruction

The following call conditions apply:

- The following applies to a CPU of the S7-1200 series:
  In firmware versions <= 2.2, you can call the instruction only in a program cycle organization block of priority 1. This corresponds to the lowest priority of all organization blocks. If the instruction is called in an organization block with a higher priority, the instruction is not executed and the result (BR bit, enable output ENO) is always "0".
  In firmware versions >= 2.2, you can call the instruction regardless of the priority in all organization blocks.
- The following applies to a CPU of the S7-1500 series:
  You can call the instruction regardless of the priority in all organization blocks.

Parameters

The "Restart cycle monitoring time" instruction has no parameters and supplies no error information.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

STP: Exit program (S7-1200, S7-1500)

Description

You use the "Exit program" instruction to set the CPU to STOP mode and therefore to terminate the execution of the program. The effects of changing from RUN to STOP depend on the CPU configuration.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

GET_ERROR: Get error locally (S7-1200, S7-1500)

Description

The "Get error locally" instruction is used to query the occurrence of errors within a program block. This normally involves programming or access errors. If the system reports an error during the execution of the program block, detailed information is output at the <Operand> on the first error to occur during the execution of the block since the last execution of the instruction.

The error information can only be saved in operands of the "ErrorStruct" system data type. The "ErrorStruct" system data type specifies the exact structure in which the information about the error is stored. Using additional instructions, you can evaluate this structure and program an appropriate response. If several errors occur in the program block, the error information about the next error to occur is output by the instruction only after the first error that occurred has been remedied.

Note

<Operand>

The <Operand> is only changed if error information is present. To set the operand back to "0" after handling the error, you have the following options:

- Declare the operand in the "Temp" section of the block interface.
- Reset the operand to "0" before calling the instruction.

Note

Activation of the local error handling

As soon as you insert the instruction in the program code of a program block, the local error handling is activated and default system reactions are ignored when errors occur.

Options for error handling

You can find an overview of the options for error handling here: AUTOHOTSPOT

For a detailed example of local error handling that contains several error handling options, refer to: AUTOHOTSPOT

Parameters

The following table shows the parameters of the instruction:

Parameters	Data type	Memory area	Description
<Operand>	ErrorStruct	D, L	Information about errors that have occurred

Data type "ErrorStruct"

The "ErrorStruct" data type can be inserted in a global data block or in the block interface. You can insert the data type more then once if you assign a different name for the data structure each time. The data structure and the name of individual structure elements cannot be changed. If you save the error information in a global data block, it can also be read out by other program blocks.

The following table shows the structure of the "ErrorStruct" data type:

Structure component		Data type	Description
ERROR_ID		WORD	Error ID
FLAGS		BYTE	Shows whether the error occurred during a program block call. 16#01: Error during a program block call 16#00: No error during a program block call
REACTION		BYTE	Default reaction: 0: Ignore (write error) 1: Continue with substitute value "0" (read error) 2: Skip instruction (system error)
CODE_ADDRESS		CREF	Information about the address and type of the program block
	BLOCK_TYPE	BYTE	Program block type in which the error occurred: 1: Organization block (OB) 2: Function (FC) 3: Function block (FB)
	CB_NUMBER	UINT	Number of the code block
	OFFSET	UDINT	Reference to the internal memory
MODE		BYTE	Information about the address of an operand
OPERAND_NUMBER		UINT	Operand number of the machine command
POINTER_NUMBER_LOCATION		UINT	(A) Internal pointer
SLOT_NUMBER_SCOPE		UINT	(B) Storage area in internal memory
DATA_ADDRESS		NREF	Information about the address of an operand
	AREA	BYTE	(C) Memory area: L: 16#40...16#7F, 16#86, 16#87, 16#8E, 16#8F, 16#C0...16#FF I: 16#81 Q: 16#82 M: 16#83 DB: 16#40, 16#84, 16#85, 16#8A, 16#8B PI: 16#01 PQ: 16#02 Technological objects: 16#04

Structure component		Data type	Description
	DB_NUMBER	UINT	(D) Number of the data block
	OFFSET	UDINT	(E) Relative address of the operand

Structure component "ERROR_ID"

The following table shows the values that can be output at the structure component "ERROR_ID":

ID* (hexadecimal)	ID* (decimal)	Description
0	0	No error
2503	9475	Invalid pointer
2520	9504	Invalid STRING
2522	9506	Read error: Operand outside the valid range
2523	9507	Write error: Operand outside the valid range
2524	9508	Read error: Invalid operand
2525	9509	Write error: Invalid operand
2528	9512	Read error: Data alignment
2529	9513	Write error: Data alignment
252C	9516	Invalid pointer
2530	9520	Write error: Data block
2533	9523	Invalid reference used
2538	9528	Access error: DB does not exist
2539	9529	Access error: Wrong DB used
253A	9530	Global data block does not exist
253C	9532	Faulty information or the function does not exist
253D	9533	System function does not exist
253E	9534	Faulty information or the function block does not exist
253F	9535	System block does not exist
2550	9552	Access error: DB does not exist
2551	9553	Access error: Wrong DB used
2575	9589	Error in the program nesting depth
2576	9590	Error in the local data distribution
2577	9591	The block property "Parameter passing via registers" is not selected.
25A0	9632	Internal error in TP
25A1	9633	Tag is write-protected
25A2	9634	Invalid numerical value of tag
2942	10562	Read error: Input
2943	10563	Write error: Output
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".		

Example

The following example shows how the instruction works:

SCL

```
LABEL: #TagOut := #Field[#index] * REAL#40.5;

GET_ERROR(#Error);

IF #Error.REACTION = 1 THEN
   GOTO LABEL;
   ;
END_IF;
```

An error occurred accessing the #Field[#index] tag. The #TagOut operand returns the signal state "1" despite the read/access error and the multiplication is performed with the value "0.0". If this error scenario occurs, we recommend that you program the "Get error locally" instruction after the multiplication to get the error. The error information supplied by the "Get error locally" instruction is evaluated with a comparison. If the #Error.REACTION structure component has the value "1", there is a read/access error and the program execution begins again at the jump label LABEL.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

GET_ERR_ID: Get error ID locally (S7-1200, S7-1500)

Description

The "Get error ID locally" instruction is used to query the occurrence of errors within a block. This is usually to access error. If the system reports during the block processing errors within the block execution since the last execution of the instruction, the instruction outputs the error ID of the first error that occurred.

The error ID can only be saved in operands of the WORD data type. If several errors occur in the block, the error ID of the next error to occur in the instruction is output only after the first error that occurred has been remedied.

Note

The <Operand> is only changed if error information is present. To set the operand back to "0" after handling the error, you have the following options:

- Declare the operand in the "Temp" section of the block interface.
- Reset the operand to "0" before calling the instruction.

The output of the "Get error ID locally" instruction is only set if error information is present. If one of these conditions is not fulfilled, the remaining program execution is not affected by the "Get error ID locally" instruction.

You can find an example of how you can implement the instruction in combination with other troubleshooting options under "See also".

Note

The "Get error ID locally" instruction enables local error handling within a block. If the "Get error ID locally" instruction is inserted in the program code of a block, any predefined system reactions are ignored if errors occur.

Options for error handling

You can find an overview of the options for error handling here: AUTOHOTSPOT

For a detailed example of local error handling that contains several error handling options, refer to: AUTOHOTSPOT

Parameters

The following table shows the parameters of the instruction:

Parameters	Data type	Memory area	Description
Function value	WORD	D, L	Error ID

Error ID

The following table shows the values that can be output:

ID* (hexadecimal)	ID* (decimal)	Description
0	0	No error
2503	9475	Invalid pointer
2520	9504	Invalid STRING
2522	9506	Read error: Operand outside the valid range
2523	9507	Write error: Operand outside the valid range
2524	9508	Read error: Invalid operand
2525	9509	Write error: Invalid operand
2528	9512	Read error: Data alignment
2529	9513	Write error: Data alignment
252C	9516	Invalid pointer
2530	9520	Write error: Data block
2533	9523	Invalid reference used
2538	9528	Access error: DB does not exist
2539	9529	Access error: Wrong DB used

ID* (hexadecimal)	ID* (decimal)	Description
253A	9530	Global data block does not exist
253C	9532	Faulty information or the function does not exist
253D	9533	System function does not exist
253E	9534	Faulty information or the function block does not exist
253F	9535	System block does not exist
2550	9552	Access error: DB does not exist
2551	9553	Access error: Wrong DB used
2575	9589	Error in the program nesting depth
2576	9590	Error in the local data distribution
2577	9591	The block property "Parameter passing via registers" is not selected.
25A0	9632	Internal error in TP
25A1	9633	Tag is write-protected
25A2	9634	Invalid numerical value of tag
2942	10562	Read error: Input
2943	10563	Write error: Output
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".		

Example

The following example shows how the instruction works:

SCL

```
#TagOut := #Field[#index] * REAL#40.5;

#TagID := GET_ERR_ID();

IF #TagID = 16#2522 THEN
   MOVE_BLK(IN := #TagArrayIn[0],
            COUNT := 1,
            OUT => #TagArrayOut[1]);
END_IF;
```

An error occurred accessing the #Field[#index] tag. The #TagOut operand returns the signal state "1" despite the read/access error and the multiplication is performed with the value "0.0". If this error scenario occurs, we recommend that you program the "Get error ID locally" instruction after the multiplication to get the error. The error ID supplied by the "Get error ID locally" instruction is evaluated with a comparison. If the #TagID operand returns ID 16#2522, there is a read/access error and the value "100.0" of the #TagArrayIn[0] operand is written to the #TagArrayOut[1] operand.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

INIT_RD: Initialize all retain data (S7-1500)

Description

The "Initialize all retain data" instruction is used to reset the retentive data of all data blocks, bit memories and SIMATIC timers and counters at the same time. The instruction can only be executed within a startup OB because the execution exceeds the program cycle duration.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	BOOL	I, Q, M, D, L	If the input "REQ" has the signal state "1", all retentive data are reset.
Function value (RET_VAL)		INT	I, Q, M, D, L	Error information: If an error occurs during the execution of the instruction, an error code is output at the RET_VAL parameter.

For additional information on valid data types, refer to "See also".

RET_VAL parameter

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0000	No error
80B5	The instruction cannot be executed because it was not programmed within a startup OB.
General error information	See also: "GET_ERR_ID: Get error ID locally"
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := INIT_RD("Tag_REQ");
```

If the operand "Tag_REQ" has the signal state "1", the instruction is executed. All retentive data of all data blocks, bit memories and SIMATIC timers and counters are reset.

See also

Overview of the valid data types (Page 2239)

GET_ERR_ID: Get error ID locally (Page 3409)

Basics of SCL (Page 4991)

WAIT: Configure time delay (S7-1500)

Description

The "Configure time delay" instruction pauses the program execution for a specific period of time. You indicate the period of time in microseconds on the WT parameter of the instruction.

You can configure time delays from -32768 up to 32767 microseconds (µs). The smallest possible delay time depends on the CPU and corresponds to the execution time of the instruction.

The execution of the instruction can be interrupted by higher priority events and does not return any error information.

Note

Negative delay time

If you specify a negative delay time at parameter WT, the enable output ENO, or the RLO and the BR bit, will return the signal state FALSE. A negative delay time does not affect the CPU. The following instructions linked to enable output ENO are not executed in LAD or FBD.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
WT	Input	INT	I, Q, M, D, L, P or constant	Time delay in microseconds (µs)

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

RUNTIME: Measure program runtime (S7-1200, S7-1500)

Description

The "Measure program runtime" instruction is used to measure the runtime of the entire program, individual blocks or command sequences.

If you want to measure the runtime of your entire program, call the instruction "Measure program runtime" in OB1. Measurement of the runtime is started with the first call and the output RET_VAL returns the runtime of the program after the second call. The measured runtime includes all CPU processes that can occur during the program execution, for example, interruptions caused by higher-level events or communication. The instruction "Measure program runtime" reads an internal counter of the CPU and writes the value to the in/out parameter. The instruction calculates the current program runtime according to the internal counter frequency and writes it to output RET_VAL.

If you want to measure the runtime of individual blocks or individual command sequences, you need three separate networks. Call the instruction "Measure program runtime" in an individual network within your program. You set the starting point of the runtime measurement with this first call of the instruction. Then you call the required program block or the command sequence in the next network. In another network, call the "Measure program runtime" instruction a second time and assign the same memory to the in/out parameter as you did during the first call of the instruction. The "Measure program runtime" instruction in the third network reads an internal CPU counter and calculates the current runtime of the program block or the command sequence according to the internal counter frequency and writes it to the output RET_VAL.

The "Measure program runtime" instruction uses an internal high-frequency counter to calculate the time. If the counter overruns, the instruction returns values <= 0.0. This can occur up to once per minute for S7-1200 CPUs with firmware version <V4.2. Ignore these runtime values.

Note

The runtime of a command sequence cannot be determined exactly, because the sequence of instructions within a command sequence is changed during optimized compilation of the program.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	InOut	LREAL	I, Q, M, D, L	The content is for internal purposes only.
Function value		LREAL	I, Q, M, D, L	Returns the measured runtime in seconds

Example

The following example shows the how the instruction works based on the runtime calculation of a program block:

```
SCL
"Tag_Result" := RUNTIME("Tag_Memory");

"Best_before_date_DB" ();

"Tag_Result" := RUNTIME("Tag_Memory");
```

The starting point for the runtime measurement is set with the first call of the instruction and buffered as reference for the second call of the instruction in the "TagMemory" operand.

The "Best_before_date" program block FB1 is called.

When the program block FB1 has been processed, the instruction is executed a second time. The second call of the instruction calculates the runtime of the program block and writes the result to the output "Tag_Result".

You can find a more detailed example of how you can measure the total cycle time of a program in the Siemens Industry Online Support under the following entry ID: 87668055 (https://support.industry.siemens.com/cs/ww/en/view/87668055)

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Word logic operations (S7-1200, S7-1500)

DECO: Decode (S7-1200, S7-1500)

Description

The instruction "Decode" sets a bit specified by the input value in the output value.

The instruction "Decode" reads the value of the parameter IN and sets the bit in the output value, whose bit position corresponds to the read value. The other bits in the output value are filled with zeroes. If the value of the IN parameter is greater than 31, a modulo 32 instruction is executed.

Use the following syntax to change the data type of the instruction:

DECO_<data type>();

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN	Input	UINT	I, Q, M, D, L, P	Position of the bit in the output value which is set.
_<Data type>		Bit strings default: DWORD	-	Data type of the function value: 1. You can specify the data type of the instruction explicitly using "_". 2. If you do not specify the data type explicitly, it will be determined by the utilized tags or type-coded constants. 3. If you neither specify the data type explicitly nor specify defined tags or type-coded constants, the default data type will be used.
Function value		Bit strings	I, Q, M, D, L, P	Current output value

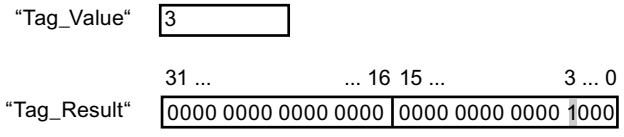
For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := DECO(IN := "Tag_Value");
"Tag_Result2" := DECO_BYTE(IN := "Tag_Value2");
```

The following figure shows how the instruction works using specific operand values:



The instruction reads the number "3" from the value of the operand "Tag_Value" and sets the third bit to the value of the operand "Tag_Result".

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ENCO: Encode (S7-1200, S7-1500)

Description

The instruction "Encode" reads the bit number of the lowest-value bit set in the input value and issues this as a result.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN	Input	Bit strings	I, Q, M, D, L, P	Input value
Function value		INT	I, Q, M, D, L, P	Bit number of the bit in the input value that is read out.

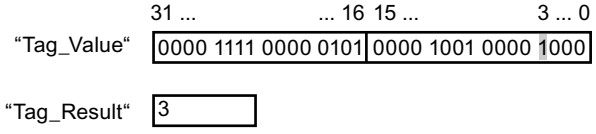
For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := ENCO(IN := "Tag_Value");
```

The following figure shows how the instruction works using specific operand values:



The instruction reads the lowest-value set bit of the operand "Tag_Value" and writes the bit position "3" in the operand "Tag_Result".

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SEL: Select (S7-1200, S7-1500)

Description

The instruction "Select" selects one of the parameters IN0 or IN1 depending on a switch (G parameter) and issues its content as a result. When the parameter G has the signal status "0", the value at parameter IN0 is moved. When the parameter G has the signal status "1", the value at parameter IN1 is moved and returned as a function value.

The instruction is only executed if the tags of all parameters have the same data type class.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
G	Input	BOOL	BOOL	I, Q, M, D, L	Switch
IN0	Input	Binary numbers, integers, floating-point numbers, timers, strings, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, timers, strings, DATE, TOD, LTOD, DT, LDT	I, Q, M, D, L, P	First input value
IN1	Input	Binary numbers, integers, floating-point numbers, timers, strings, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, timers, strings, DATE, TOD, LTOD, DT, LDT	I, Q, M, D, L, P	Second input value
Function value		Binary numbers, integers, floating-point numbers, timers, strings, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, timers, strings, DATE, TOD, LTOD, DT, LDT	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := SEL(G := "Tag_Value",
                    IN0 := "Tag_0",
                    IN1 := "Tag_1");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value	
Tag_Value	0	1
Tag_0	W#16#0000	W#16#4C
Tag_1	W#16#FFFF	D#16#5E
Tag_Result	W#16#0000	D#16#5E

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

MUX: Multiplex (S7-1200, S7-1500)

Description

The "Multiplex" instruction copies the value of a selected input parameter and issues it. You can use the parameter K to determine the number of the input parameter whose value will be moved. Numbering starts at IN0 and is incremented continuously with each new input. You can declare a maximum of 32 inputs.

Numerical data types and time data types are permitted at the inputs. All tags with assigned parameters must be of the same data type.

The function value is invalid if any of the following requirements are met:

- Errors occurred during execution of the instruction.
- If the input at the K parameter is located outside the available inputs and the INELSE input is not used, the function value is invalid.

Note

K parameter has a negative integer

If a tag with a valid data type is specified at the input parameters and the K parameter has a negative integer, the value of the tag is changed.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
K	Input	Integers	Integers	I, Q, M, D, L, P	Specifies the parameter whose content is to be transferred. • If K = 0 => Parameter IN0 • If K = 1 => Parameter IN1, etc.
IN0	Input	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, TOD, LTOD, DATE, timers, DT, LDT	I, Q, M, D, L, P	First input value
IN1	Input	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, TOD, LTOD, DATE, timers, DT, LDT	I, Q, M, D, L, P	Second input value
INn	Input	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, TOD, LTOD, DATE, timers, DT, LDT	I, Q, M, D, L, P	Optional input values

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
INELSE	Input	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, TOD, LTOD, DATE, timers, DT, LDT	I, Q, M, D, L, P	Specifies the value to be copied when K <> n.
Function value		Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, TOD, LTOD, DATE, timers, DT, LDT	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := MUX(K := "Tag_Number",
                    IN0 := "Tag_1",
                    IN1 := "Tag_2",
                    INELSE := "Tag_3");
```

The result of the instruction is returned in the operand "Tag_Result" as a function value.

The following table shows how the instruction works using specific operand values:

Operand	Value	
Tag_Number	1	4
Tag_1	DW#16#00000000	DW#16#00000000
Tag_2	DW#16#003E4A7D	DW#16#003E4A7D
Tag_3	DW#16#FFFF0000	DW#16#FFFF0000
Tag_Result	DW#16#003E4A7D	DW#16#FFFF0000

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

DEMUX: Demultiplex (S7-1200, S7-1500)

Description

The "Demultiplex" instruction transfers the value of the input parameter IN to a selected output parameter. The selection of the input parameter takes place independent of the parameter value K. The K parameter specifies the output parameter number to which the value of the input parameter IN is transferred. The other output parameters are not changed. Numbering starts at OUT0 and continues consecutively with each new output. You can declare a maximum of 32 output parameters.

If the value of the K parameter is greater than the number of output parameters, the content of the IN input parameter is copied to the OUTELSE output parameter and the signal state "0" is assigned to the enable output ENO.

The function value is invalid if any of the following requirements are met:

- The value of the K parameter is greater than the number of available outputs.
- Errors occurred during execution of the instruction.

Note

K parameter < 0 or K > available outputs

If you specify at the K parameter a value that is outside the available outputs (K < 0 or K > available outputs), the value of the IN input parameter is output at the OUTELSE output parameter.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
K	Input	Integers	Integers	I, Q, M, D, L, P	Specifies the output to which the input value (IN) will be copied. • If K = 0 => Parameter OUT0 • If K = 1 => Parameter OUT1, etc.
IN	Input	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, timers, TOD, LTOD, DATE, DT, LDT	I, Q, M, D, L, P	Input value
OUT0	Output	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, timers, TOD, LTOD, DATE, DT, LDT	I, Q, M, D, L, P	First output
OUT1	Output	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, timers, TOD, LTOD, DATE, DT, LDT	I, Q, M, D, L, P	Second output

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
OUTn	Output	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, timers, TOD, LTOD, DATE, DT, LDT	I, Q, M, D, L, P	Optional outputs
OUTELSE	Output	Binary numbers, integers, floating-point numbers, timers, STRING, CHAR, WCHAR, TOD, DATE, DT	Binary numbers, integers, floating-point numbers, strings, timers, TOD, LTOD, DATE, DT, LDT	I, Q, M, D, L, P	Output to which the value at input IN is copied if K > n.

For additional information on available data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
DEMUX(K := "Tag_Number",
      IN := "Tag_Value",
      OUT0 := "Tag_1",
      OUT1 := "Tag_2",
      OUTELSE := "Tag_3");
```

The following tables show how the instruction works using specific operand values:

Input values of the "Demultiplex" instruction before the network execution

Parameters	Operand	Values	
K	Tag_Number	2	4
IN	Tag_Value	DW#16#FFFFFFFF	DW#16#003E4A7D

Output values of the "Demultiplex" instruction after the network execution

Parameters	Operand	Values	
OUT0	Tag_1	Unchanged	Unchanged
OUT1	Tag_2	DW#16#FFFFFFFF	Unchanged
OUTELSE	Tag_3	Unchanged	DW#16#003E4A7D

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Shift and Rotate (S7-1200, S7-1500)

SHR: Shift right (S7-1200, S7-1500)

Description

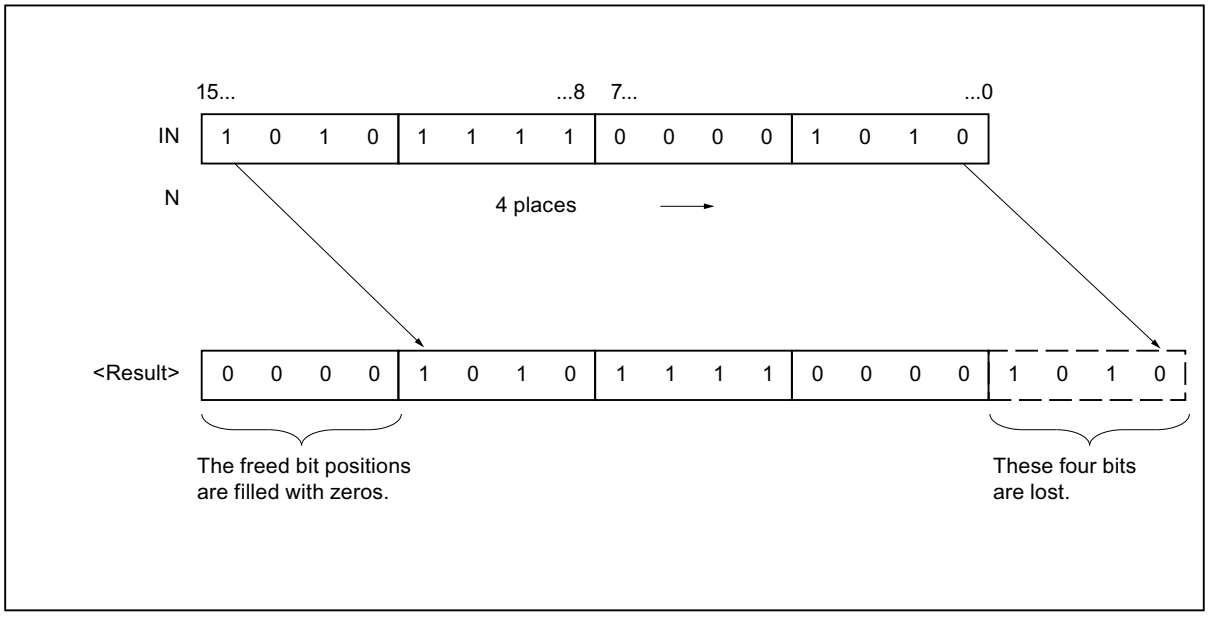
The "Shift right" instruction shifts the contents of the IN parameter bit-by-bit to the right and returns it as a function value. The parameter N is used to specify the number of bit positions by which the specified value should be shifted.

If the value of the N parameter is "0", the value of the IN parameter is given as a result.

If the value of the N parameter is greater than the number of available bit places, then the value of the IN parameter is shifted to the right by the available number of bit places.

The freed bit positions in the left area of the operand are filled by zeroes when values without signs are shifted. If the specified value has a sign, the free bit positions are filled with the signal state of the sign bit.

The following figure shows how the content of an integer data type operand is shifted by four bit positions to the right:



Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Value to be shifted
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L	Number of bits by which the value (IN) is shifted
Function value		Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := SHR(IN := "Tag_Value",
                    N := "Tag_Number");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	0011 1111 1010 1111
N	Tag_Number	3
Function value	Tag_Result	0000 0111 1111 0101

The content of the "Tag_Value" operand is shifted by three bit positions to the right. The result of the instruction is returned in the operand "Tag_Result" as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SHL: Shift left (S7-1200, S7-1500)

Description

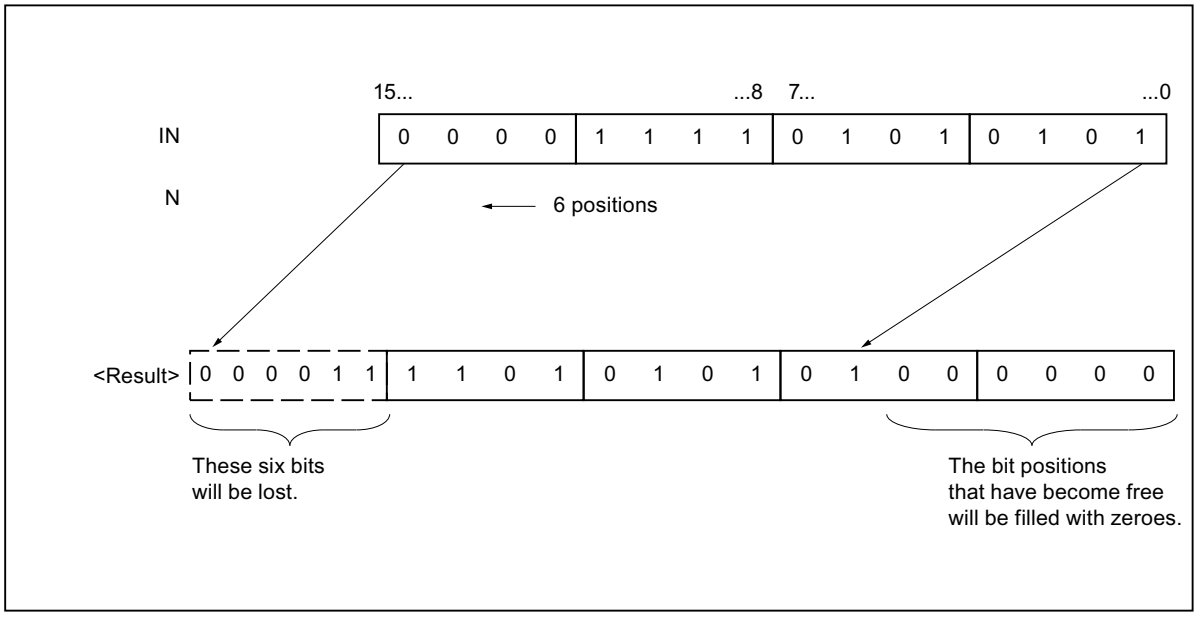
The "Shift left" instruction shifts the contents of the IN parameter bit-by-bit to the left and returns it as a function value. The parameter N is used to specify the number of bit positions by which the specified value should be shifted.

If the value of the N parameter is "0", the value of the IN parameter is given as a result.

If the value of the N parameter is greater than the number of bit places, the value of the IN parameter is shifted to the left by the available number of bit places.

The bit positions freed by the shift are filled with zeros in the result value.

The following figure shows how the content of an operand of the WORD data type is shifted six bit positions to the left:



Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Value to be shifted
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L	Number of bits by which the value (IN) is shifted
Function value		Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := SHL(IN := "Tag_Value",
                    N := "Tag_Number");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	0011 1111 1010 1111
N	Tag_Number	4
Function value	Tag_Result	1111 1010 1111 0000

The value of the "Tag_Value" operand is shifted by four bit places to the left. The result of the instruction is returned in the operand "Tag_Result" as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ROR: Rotate right (S7-1200, S7-1500)

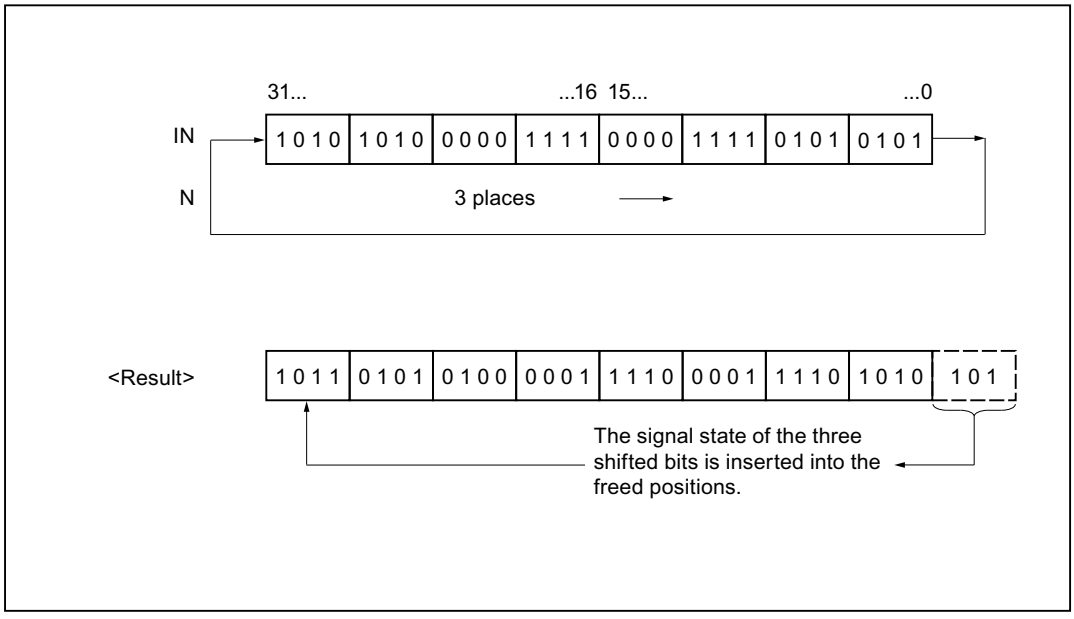
Description

The "Rotate right" instruction rotates the content of the IN parameter bit-by-bit to the right and assigns the result to the specified operand. The parameter N is used to specify the number of bit places by which the specified value should be rotated. The bit positions freed by rotating are filled with the bit positions that are pushed out.

If the value of the N parameter is "0", the value at input IN is given as a result.

If the value at the N parameter is greater than the number of available bit positions, the operand value at the IN input is still rotated by the specified number of bit positions.

The following figure shows how the content of an operand of the DWORD data type is rotated three bit positions to the right:



Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Value to be rotated
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L	Number of bit positions by which the value (IN) is rotated
Function value		Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

SCL

```
"Tag_Result" := ROR(IN := "Tag_Value",
                    N := "Tag_Number");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	0000 1111 1001 0101
N	Tag_Number	5
Function value	Tag_Result	1010 1000 0111 1100

The content of the "Tag_Value" operand is rotated by five bit positions to the right. The result of the instruction is returned in the operand "Tag_Result" as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

ROL: Rotate left (S7-1200, S7-1500)

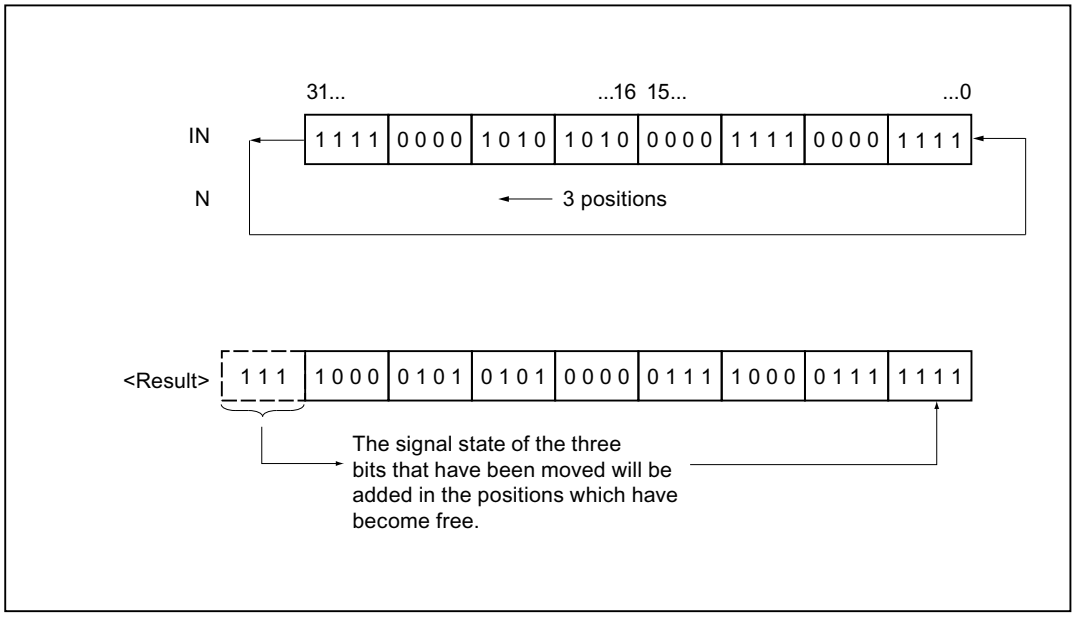
Description

The "Rotate left" instruction rotates the contents of the IN parameter bit-by-bit to the left and returns it as a function value. The parameter N is used to specify the number of bit places by which the specified value should be rotated. The bit positions freed by rotating are filled with the bit positions that are pushed out.

If the value of the N parameter is "0", the value at input IN is given as a result.

If the value at the N parameter is greater than the number of available bit positions, the operand value at the IN input is still rotated by the specified number of bit positions.

The following figure shows how the content of an operand of the DWORD data type is rotated three bit positions to the left:



Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Value to be rotated
N	Input	USINT, UINT, UDINT	USINT, UINT, UDINT, ULINT	I, Q, M, D, L	Number of bit positions by which the value (IN) is rotated
Function value		Bit strings, integers	Bit strings, integers	I, Q, M, D, L	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := ROL(IN := "Tag_Value",
                    N := "Tag_Number");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value
IN	Tag_Value	1010 1000 1111 0110
N	Tag_Number	5
Function value	Tag_Result	0001 1110 1101 0101

The content of the operand "Tag_Value" is rotated five bit places to the left. The result of the instruction is returned in the operand "Tag_Result" as a function value.

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

Legacy (S7-1500)

DRUM: Implement sequencer (S7-1500)

Description

You can use the "Implement sequencer" instruction to assign the programmed values of the OUT_VAL parameter of the corresponding step to the programmed output bits (OUT1 to OUT16) and the output word (OUT_WORD). The specific step must thereby satisfy the conditions of the programmed enable mask on the S_MASK parameter while the instruction remains at this step. The instruction advances to the next step if the event for the step is true and the programmed time for the current step elapses, or if the value at the JOG parameter changes from "0" to "1". The instruction is reset when the signal state of the RESET parameter changes to "1". The current step is hereby equated to the preset step (DSP).

The amount of time spent on a step is determined by the product of the preset timebase (DTBP) and the preset counter value (S_PRESET) for each step. At the start of a new step, this calculated value is loaded into the DCC parameter, which contains the time remaining for the current step. If, for example the value at the DTBP parameter is "2" and the preset value for the first step is "100" (100 ms), the DCC parameter has the value "200" (200 ms).

A step can be programmed with a timer value, an event, or both. Steps that have an event bit and the timer value "0" advance to the next step as soon as the signal state of the event bit is "1". Steps that are programmed only with a timer value start the time immediately. Steps that are programmed with an event bit and a time value greater than "0" start the time when the signal state of the event bit is "1". The event bits are initialized with a signal state of "1".

When the sequencer is on the last programmed step (LST_STEP) and the time for this step has expired, the signal state on the Q parameter is set to "1"; otherwise it is set to "0". When the parameter Q is set, the instruction remains on the step until it is reset.

In the configurable mask (S_MASK) you can select the individual bits in the output word (OUT_WORD) and set or reset the output bits (OUT1 to OUT16) by means of the output values (OUT_VAL). When a bit of the configurable mask has a signal state of "1", the OUT_VAL value sets/resets the corresponding bit. If the signal state of a bit of the configurable mask is "0", the corresponding bit is left unchanged. All the bits of the configurable mask for all 16 steps are initialized with a signal state of "1".

The output bit on the OUT1 parameter corresponds to the least significant bit of the output word (OUT_WORD). The output bit on the OUT16 parameter corresponds to the most significant bit of the output word (OUT_WORD).

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
RESET	Input	BOOL	I, Q, M, D, L	The signal state "1" indicates a reset condition.
JOG	Input	BOOL	I, Q, M, D, L	When the signal state changes from "0" to "1", the instruction advances to the next step.
DRUM_EN	Input	BOOL	I, Q, M, D, L	The signal state "1" allows the sequencer to advance based on the event and time criteria.
LST_STEP	Input	BYTE	I, Q, M, D, L	Maximum step number (for example: LST_STEP = 16#08; a maximum of 8 steps is possible.)
EVENT(i), 1 ≤ i ≤ 16	Input	BOOL	I, Q, M, D, L	Event bit (i); Initial signal state is "1".
OUT(j), 1 ≤ j ≤ 16	Output	BOOL	I, Q, M, D, L	Output bit (j)
Q	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that the time for the last step has elapsed.
OUT_WORD	Output	WORD	I, Q, M, D, L, P	Word address to which the sequencer writes the output values.
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
JOG_HIS	Static	BOOL	I, Q, M, D, L	JOG parameter history bit
EOD	Static	BOOL	I, Q, M, D, L	The signal state "1" indicates that the time for the last step has elapsed.
DSP	Static	BYTE	I, Q, M, D, L, P	Preset step of the sequencer
DSC	Static	BYTE	I, Q, M, D, L, P	Current step of the sequencer
DCC	Static	DWORD	I, Q, M, D, L, P	Current counter value of the sequencer
DTBP	Static	WORD	I, Q, M, D, L, P	Preset timebase of the sequencer
PrevTime	Static	TIME	I, Q, M, D, L	Previous system time
S_PRESET	Static	ARRAY[1..16] of WORD	I, Q, M, D, L	Preset counter value for each step [1 to 16] where one clock pulse = 1 ms.
OUT_VAL	Static	ARRAY[1..16, 0..15] of BOOL	I, Q, M, D, L	Output values for each step [1 to 16, 0 to 15].
S_MASK	Static	ARRAY[1..16, 0..15] of BOOL	I, Q, M, D, L	Configurable mask for each step [1 to 16, 0 to 15]. Initial signal states are "1".

For additional information on valid data types, refer to "See also".

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

ERR_CODE*	Explanation
W#16#0000	No error
W#16#000B	The value at the LST_STEP parameter is less than 1 or greater than 16.
W#16#000C	The value at the DSC parameter is less than 1 or greater than the value of the LST_STEP parameter.
W#16#000D	The value at the DSP parameter is less than 1 or greater than the value of the LST_STEP parameter.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

In the following example the instruction advances from step 1 to step 2. The output bits (OUT1 to OUT16) and the output word (OUT_WORD) are set according to the mask configured for step 2 and the values of the OUT_VAL parameter.

Note

You can initialize static parameters in the data block.

SCL

```
"DRUM_DB"(RESET := "Tag_Reset"
          JOG := "Tag_Input_Jog"
          DRUM_EN := "Tag_Input_DrumEN"
          LST_STEP := "Tag_Number_LastStep"
          EVENT1 := "MyTag_Event_1"
          EVENT2 := "MyTag_Event_2"
          EVENT3 := "MyTag_Event_3"
          EVENT4 := "MyTag_Event_4"
          EVENT5 := "MyTag_Event_5"
          EVENT6 := "MyTag_Event_6"
          EVENT7 := "MyTag_Event_7"
          EVENT8 := "MyTag_Event_8"
          EVENT9 := "MyTag_Event_9"
          EVENT10 := "MyTag_Event_10"
          EVENT11 := "MyTag_Event_11"
          EVENT12 := "MyTag_Event_12"
          EVENT13 := "MyTag_Event_13"
          EVENT14 := "MyTag_Event_14"
          EVENT15 := "MyTag_Event_15"
```

SCL

```
EVENT16 := "MyTag_Event_16"
OUT1 => "MyTag_Output_1"
OUT2 => "MyTag_Output_2"
OUT3 => "MyTag_Output_3"
OUT4 => "MyTag_Output_4"
OUT5 => "MyTag_Output_5"
OUT6 => "MyTag_Output_6"
OUT7 => "MyTag_Output_7"
OUT8 => "MyTag_Output_8"
OUT9 => "MyTag_Output_9"
OUT10 => "MyTag_Output_10"
OUT11 => "MyTag_Output_11"
OUT12 => "MyTag_Output_12"
OUT13 => "MyTag_Output_13"
OUT14 => "MyTag_Output_14"
OUT15 => "MyTag_Output_15"
OUT16 => "MyTag_Output_16"
Q => "Tag_Output_Q"
OUT_WORD => "Tag_OutputWord"
ERR_CODE => "Tag_ErrorCode");
```

The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for initializing the input parameters:

Parameters	Operand	Address	Value
RESET	Tag_Reset	M0.0	FALSE
JOG	Tag_Input_JOG	M0.1	FALSE
DRUM_EN	Tag_Input_DrumEN	M0.2	TRUE
LST_STEP	Tag_Number_LastStep	MB1	B#16#08
EVENT2	MyTag_Event_2	M20.0	FALSE
EVENT4	MyTag_Event_4	M20.1	FALSE
EVENT6	MyTag_Event_6	M20.2	FALSE
EVENT8	MyTag_Event_8	M20.3	FALSE
EVENT10	MyTag_Event_10	M20.4	FALSE
EVENT12	MyTag_Event_12	M20.5	FALSE
EVENT14	MyTag_Event_14	M20.6	FALSE
EVENT16	MyTag_Event_16	M20.7	FALSE

The following values are saved in the "DRUM_DB" instance data block of the instruction:

Parameters	Address	Value
JOG_HIS	DBX12.0	FALSE
EOD	DBX12.1	FALSE
DSP	DBB13	W#16#0001
DSC	DBB14	W#16#0001
DCC	DBD16	DW#16#0000000A
DTBP	DBW20	W#16#0001
S_PRESET[1]	DBW26	W#16#0064
S_PRESET[2]	DBW28	W#16#00C8
OUT_VAL[1,0]	DBX58.0	TRUE
OUT_VAL[1,1]	DBX58.1	TRUE
OUT_VAL[1,2]	DBX58.2	TRUE
OUT_VAL[1,3]	DBX58.3	TRUE
OUT_VAL[1,4]	DBX58.4	TRUE
OUT_VAL[1,5]	DBX58.5	TRUE
OUT_VAL[1,6]	DBX58.6	TRUE
OUT_VAL[1,7]	DBX58.7	TRUE
OUT_VAL[1,8]	DBX59.0	TRUE
OUT_VAL[1,9]	DBX59.1	TRUE
OUT_VAL[1,10]	DBX59.2	TRUE
OUT_VAL[1,11]	DBX59.3	TRUE
OUT_VAL[1,12]	DBX59.4	TRUE
OUT_VAL[1,13]	DBX59.5	TRUE
OUT_VAL[1,14]	DBX59.6	TRUE
OUT_VAL[1,15]	DBX59.7	TRUE
OUT_VAL[2,0]	DBX60.0	FALSE
OUT_VAL[2,1]	DBX60.1	FALSE
OUT_VAL[2,2]	DBX60.2	FALSE
OUT_VAL[2,3]	DBX60.3	FALSE
OUT_VAL[2,4]	DBX60.4	FALSE
OUT_VAL[2,5]	DBX60.5	FALSE
OUT_VAL[2,6]	DBX60.6	FALSE
OUT_VAL[2,7]	DBX60.7	FALSE
OUT_VAL[2,8]	DBX61.0	FALSE
OUT_VAL[2,9]	DBX61.1	FALSE
OUT_VAL[2,10]	DBX61.2	FALSE
OUT_VAL[2,11]	DBX61.3	FALSE
OUT_VAL[2,12]	DBX61.4	FALSE
OUT_VAL[2,13]	DBX61.5	FALSE
OUT_VAL[2,14]	DBX61.6	FALSE
OUT_VAL[2,15]	DBX61.7	FALSE
S_MASK[2,0]	DBX92.0	FALSE

Parameters	Address	Value
S_MASK[2,1]	DBX92.1	TRUE
S_MASK[2,2]	DBX92.2	TRUE
S_MASK[2,3]	DBX92.3	TRUE
S_MASK[2,4]	DBX92.4	TRUE
S_MASK[2,5]	DBX92.5	FALSE
S_MASK[2,6]	DBX92.6	TRUE
S_MASK[2,7]	DBX92.7	TRUE
S_MASK[2,8]	DBX93.0	FALSE
S_MASK[2,9]	DBX93.1	FALSE
S_MASK[2,10]	DBX93.2	TRUE
S_MASK[2,11]	DBX93.3	TRUE
S_MASK[2,12]	DBX93.4	TRUE
S_MASK[2,13]	DBX93.5	TRUE
S_MASK[2,14]	DBX93.6	FALSE
S_MASK[2,15]	DBX93.7	TRUE

The output parameters are set to the following values before the execution of the instruction:

Parameters	Operand	Address	Value
Q	Tag_Output_Q	M6.0	FALSE
OUTWORD	Tag_OutputWord	MW8	W#16#FFFF
OUT1	MyTag_Output_1	M4.0	TRUE
OUT2	MyTag_Output_2	M4.1	TRUE
OUT3	MyTag_Output_3	M4.2	TRUE
OUT4	MyTag_Output_4	M4.3	TRUE
OUT5	MyTag_Output_5	M4.4	TRUE
OUT6	MyTag_Output_6	M4.5	TRUE
OUT7	MyTag_Output_7	M4.6	TRUE
OUT8	MyTag_Output_8	M4.7	TRUE
OUT9	MyTag_Output_9	M5.0	TRUE
OUT10	MyTag_Output_10	M5.1	TRUE
OUT11	MyTag_Output_11	M5.2	TRUE
OUT12	MyTag_Output_12	M5.3	TRUE
OUT13	MyTag_Output_13	M5.4	TRUE
OUT14	MyTag_Output_14	M5.5	TRUE
OUT15	MyTag_Output_15	M5.6	TRUE
OUT16	MyTag_Output_16	M5.7	TRUE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameters	Operand	Address	Value
OUT1	MyTag_Output_1	M4.0	TRUE
OUT2	MyTag_Output_2	M4.1	FALSE
OUT3	MyTag_Output_3	M4.2	FALSE
OUT4	MyTag_Output_4	M4.3	FALSE
OUT5	MyTag_Output_5	M4.4	FALSE
OUT6	MyTag_Output_6	M4.5	TRUE
OUT7	MyTag_Output_7	M4.6	FALSE
OUT8	MyTag_Output_8	M4.7	FALSE
OUT9	MyTag_Output_9	M5.0	TRUE
OUT10	MyTag_Output_10	M5.1	TRUE
OUT11	MyTag_Output_11	M5.2	FALSE
OUT12	MyTag_Output_12	M5.3	FALSE
OUT13	MyTag_Output_13	M5.4	FALSE
OUT14	MyTag_Output_14	M5.5	FALSE
OUT15	MyTag_Output_15	M5.6	TRUE
OUT16	MyTag_Output_16	M5.7	FALSE
Q	Tag_Output_Q	M6.0	FALSE
OUTWORD	Tag_OutputWord	MW8	W#16#4321
ERR_CODE	Tag_ErrorCode	MW10	W#16#0000

The following values are changed in the instance data block "DRUM_DB" of the instruction after the execution of the instruction:

Parameters	Address	Value
JOG_HIS	DBX12.0	FALSE
EOD	DBX12.1	FALSE
DSC	DBB14	W#16#0002
DCC	DBD16	DW#16#000000C8

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

DCAT: Discrete control-timer alarm (S7-1500)

Description

You can use the "Discrete control-timer alarm" to count the time from the point at which the CMD parameter issues the command to open or close. The time is accumulated until the preset time (PT) is exceeded or the information is received that the device was opened or closed (O_FB or C_FB) within the specified time. If the preset time is exceeded before the information on the opening or closing of the device is received, the corresponding alarm is activated. If the signal state of the command input changes before the preset time, the time is restarted.

The "Discrete control-timer alarm" instruction has the following reactions to the input conditions:

- When the signal state of the CMD parameter changes from "0" to "1", the signal states of the parameters Q, CMD_HIS, ET (only if ET is < PT) OA and CA are influenced as follows:
  - The Q and CMD_HIS parameters are set to "1".
  - The ET, OA, and CA parameters are reset to "0".
- When the signal state on the parameter CMD changes from "1" to "0", the parameters Q, ET (only if ET < PT), OA, CA and CMD_HIS are reset to "0".
- When the signal state of the parameters CMD and CMD_HIS is "1" and the parameter O_FB is set to "0", the time difference (ms) since the last execution of the instruction is added to the value at the parameter ET. If the value of the ET parameter exceeds the value of the PT parameter, the signal state at the OA parameter is set to "1". If the value of the ET parameter does not exceed the value of the PT parameter, the signal state of the OA parameter is reset to "0". The value at the parameter CMD_HIS is reset to the value of the parameter CMD.
- When the signal state of the parameters CMD, CMD_HIS and O_FB are set to "1" and the parameter C_FB has the value "0", the signal state of the parameter OA is set to "0". The value of the ET parameter is set to the value of the PT parameter. If the signal state of the parameter O_FB changes to "0", the alarm is set the next time the instruction is executed. The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the parameters CMD, CMD_HIS and C_FB have the value "0", the time difference (ms) since the last execution of the instruction is added to the value of the parameter ET. If the value of the ET parameter exceeds the value of the PT parameter, the signal state of the parameter CA is reset to "1". If the value at the parameter PT is not exceeded, the parameter CA has the signal state "0". The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the parameters CMD, CMD_HIS and O_FB have the signal state "0" and the parameter C_FB is set to "1", the parameter CA is set to "0". The value of the ET parameter is set to the value of the PT parameter. If the signal state of the C_FB parameter changes to "0", the alarm is set the next time the instruction is executed. The value of the CMD_HIS parameter is set to the value of the CMD parameter.
- If the parameters O_FB and C_FB simultaneously have the signal state "1", the signal states of both alarm outputs are set to "1".

The "Discrete control-timer alarm" instruction has no error information.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
CMD	Input	BOOL	I, Q, M, D, L	A signal state of "0" indicates a "close" command. A signal state of "1" indicates an "open" command.
O_FB	Input	BOOL	I, Q, M, D, L	Feedback input when opening
C_FB	Input	BOOL	I, Q, M, D, L	Feedback input when closing
Q	Output	BOOL	I, Q, M, D, L	Shows the status of the parameter CMD
OA	Output	BOOL	I, Q, M, D, L	Alarm output when opening
CA	Output	BOOL	I, Q, M, D, L	Alarm output when closing
ET	Static	DINT	D, L	Currently elapsed time, where one count = 1 ms.
PT	Static	DINT	D, L	Preset timer value, where one clock pulse = 1 ms.
PREV_TIME	Static	DWORD	D, L	Previous system time
CMD_HIS	Static	BOOL	D, L	CMD history bit

The static parameters are not visible when calling the instruction in the program. These are saved in the instance of the instruction.

Example

In the following example the parameter CMD changes from "0" to "1". After the execution of the instruction the parameter Q is set to "1" and the two alarm outputs OA and CA have the signal state "0". The CMD_HIS parameter of the instance data block is set to the signal state "1" and the ET parameter is reset to "0".

Note

You can initialize static parameters in the data block.

SCL

```
"DCAT_DB"(CMD := "Tag_Input_CMD",
          O_FB := "Tag_Input_O_FB",
          C_FB := "Tag_Input_C_FB",
          Q => "Tag_Output_Q",
          OA => "Tag_Output_OA",
          CA => "Tag_Output_CA");
```

The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for the input and output parameters:

Parameters	Operand	Value
CMD	Tag_Input_CMD	TRUE
O_FB	Tag_Input_O_FB	FALSE
C_FB	Tag_Input_C_FB	FALSE
Q	Tag_Output_Q	FALSE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE

The following values are saved in the instance data block "DCAT_DB" of the instruction:

Parameters	Address	Value
ET	DBD4	L#12
PT	DBD8	L#222
CMD_HIS	DBX16.0	FALSE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameters	Operand	Value
Q	Tag_Output_Q	TRUE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE

The following values are saved in the instance data block "DCAT_DB" of the instruction:

Parameters	Address	Value
ET	DBD4	L#0
CMD_HIS	DBX16.0	TRUE

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

MCAT: Motor control-timer alarm (S7-1500)

Description

The "Motor control-timer alarm" instruction is used to accumulate the time from the point at which one of the command inputs (opening or closing) is switched on. The time is accumulated until the preset time is exceeded or the relevant feedback input indicates that the device has executed the requested operation within the specified time. If the preset time is exceeded before the feedback is received, the corresponding alarm is triggered.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory space	Description
O_CMD	Input	BOOL	I, Q, M, D, L	"Open" command input
C_CMD	Input	BOOL	I, Q, M, D, L	"Close" command input
S_CMD	Input	BOOL	I, Q, M, D, L	"Stop" command input
O_FB	Input	BOOL	I, Q, M, D, L	Feedback input when opening
C_FB	Input	BOOL	I, Q, M, D, L	Feedback input when closing
OO	Output	BOOL	I, Q, M, D, L	"Open" output
CO	Output	BOOL	I, Q, M, D, L	"Close" output
OA	Output	BOOL	I, Q, M, D, L	Alarm output when opening
CA	Output	BOOL	I, Q, M, D, L	Alarm output when closing
Q	Output	BOOL	I, Q, M, D, L	A signal state of "0" indicates an error condition.
ET	Static	DINT	D, L	Currently elapsed time, where one count = 1 ms
PT	Static	DINT	D, L	Preset timer value, where one clock pulse = 1 ms
PREV_TIME	Static	DWORD	D, L	Previous system time
O_HIS	Static	BOOL	D, L	"Open" history bit
C_HIS	Static	BOOL	D, L	"Close" history bit

The static parameters are not visible when calling the instruction in the program. These are saved in the instance of the instruction.

Execution of the "Motor control-timer alarm" instruction

The following table shows the reactions of the "Motor control-timer alarm" instruction to the various input conditions:

Input parameters								Output parameters								
ET	O_HIS	C_HIS	O_CMD	C_CMD	S_CMD	O_FB	C_FB	OO	CO	OA	CA	ET	O_HIS	C_HIS	Q	Status
X	1	1	X	X	X	X	X	0	0	1	1	PT	0	0	0	Alarm
X	X	X	X	X	X	1	1	0	0	1	1	PT	0	0	0	Alarm
X	X	X	X	X	1	X	X	0	0	0	0	X	0	0	1	Stop
X	X	X	1	1	X	X	X	0	0	0	0	X	0	0	1	Stop
X	0	X	1	0	0	X	X	1	0	0	0	0	1	0	1	Start opening
<PT	1	0	X	0	0	0	X	1	0	0	0	INC	1	0	1	Open
X	1	0	X	0	0	1	0	0	0	0	0	PT	1	0	1	Opened
>=PT	1	0	X	0	0	0	X	0	0	1	0	PT	1	0	0	Opening alarm
X	X	0	0	1	0	X	X	0	1	0	0	0	0	1	1	Start closing
<PT	0	1	0	X	0	X	0	0	1	0	0	INC	0	1	1	Close

Input parameters								Output parameters								
X	0	1	0	X	0	0	1	0	0	0	0	PT	0	1	1	Closed
>=PT	0	1	0	X	0	X	0	0	0	0	1	PT	0	1	0	Closing alarm
X	0	0	0	0	0	X	X	0	0	0	0	X	0	0	1	Stopped

Legend:	
INC	Add the time difference (ms) since the last processing of the FB to ET
PT	PT is set to the same value as ET
X	Cannot be used
<PT	ET < PT
>=PT	ET >= PT

If the input parameters O_HIS and C_HIS both have the signal state "1", they are immediately set to signal state "0". In this case, the last row in the table (X) mentioned above is valid. Because it is therefore not possible to check whether the input parameters O_HIS and C_HIS have the signal state "1", the output parameters are set as follows in this case:

OO = FALSE

CO = FALSE

OA = FALSE

CA = FALSE

ET = PT

Q = TRUE

Example

The following example shows how the instruction works:

Note

You can initialize static parameters in the data block.

SCL

```
"MCAT_DB"(O_CMD := "Tag_Iput_O_CMD",
          C_CMD := "Tag_Input_C_CMD",
          S_CMD := "Tag_Input_S_CMD",
          O_FB := "Tag_Input_O_FB",
          C_FB := "Tag_Input_C_FB",
          OO => "Tag_OutputOpen",
          CO => "Tag_OutputClosed",
          OA => "Tag_Output_OA",
          CA => "Tag_Output_CA",
          Q => "Tag_Output_Q");
```

The following tables show how the instruction works using specific values.

Before processing

In this example, the following values are used for the input and output parameters:

Parameters	Operand	Value
O_CMD	Tag_Input_O_CMD	TRUE
C_CMD	Tag_Input_C_CMD	FALSE
S_CMD	Tag_Input_S_CMD	FALSE
O_FB	Tag_Input_O_FB	FALSE
C_FB	Tag_Input_C_FB	FALSE
OO	Tag_OutputOpen	FALSE
CO	Tag_OutputClosed	FALSE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE
Q	Tag_Output_Q	FALSE

The following values are saved in the instance data block "MCAT_DB" of the instruction:

Parameters	Address	Value
ET	DBD4	L#2
PT	DBD8	L#22
O_HIS	DBX16.0	TRUE
C_HIS	DBX16.1	FALSE

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameters	Operand	Value
OO	Tag_OutputOpen	TRUE
CO	Tag_OutputClosed	FALSE
OA	Tag_Output_OA	FALSE
CA	Tag_Output_CA	FALSE
Q	Tag_Output_Q	TRUE

The following values are saved in the instance data block "MCAT_DB" of the instruction:

Parameters	Address	Value
ET	DBD4	L#0
O_HIS	DBX16.0	TRUE
CMD_HIS	DBX16.1	FALSE

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

IMC: Compare input bits with the bits of a mask (S7-1500)

Description

You can use the "Compare input bits with the bits of a mask" instruction to compare the signal state of up to 16 programmed input bits (IN_BIT0 to IN_BIT15) with the corresponding bits of a mask. Up to 16 steps with masks can be programmed. The value of the IN_BIT0 parameter is compared with the value of the mask CMP_VAL[x,0], with "x" indicating the step number. In the CMP_STEP parameter, you specify the step number of the mask that is used for the comparison. All programmed values are compared in the same manner. Unprogrammed input bits or unprogrammed bits of the mask have a default signal state FALSE.

If a match is found in the comparison, the signal state of the OUT parameter is set to "1". Otherwise the OUT parameter is set to "0".

If the value of the CMP_STEP parameter is greater than 15, the instruction is not executed. An error message is output at the ERR_CODE parameter.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN_BIT0	Input	BOOL	I, Q, M, D, L	Input bit 0 is compared with bit 0 of the mask.
IN_BIT1	Input	BOOL	I, Q, M, D, L	Input bit 1 is compared with bit 1 of the mask.
IN_BIT2	Input	BOOL	I, Q, M, D, L	Input bit 2 is compared with bit 2 of the mask.
IN_BIT3	Input	BOOL	I, Q, M, D, L	Input bit 3 is compared with bit 3 of the mask.
IN_BIT4	Input	BOOL	I, Q, M, D, L	Input bit 4 is compared with bit 4 of the mask.
IN_BIT5	Input	BOOL	I, Q, M, D, L	Input bit 5 is compared with bit 5 of the mask.
IN_BIT6	Input	BOOL	I, Q, M, D, L	Input bit 6 is compared with bit 6 of the mask.
IN_BIT7	Input	BOOL	I, Q, M, D, L	Input bit 7 is compared with bit 7 of the mask.
IN_BIT8	Input	BOOL	I, Q, M, D, L	Input bit 8 is compared with bit 8 of the mask.
IN_BIT9	Input	BOOL	I, Q, M, D, L	Input bit 9 is compared with bit 9 of the mask.
IN_BIT10	Input	BOOL	I, Q, M, D, L	Input bit 10 is compared with bit 10 of the mask.
IN_BIT11	Input	BOOL	I, Q, M, D, L	Input bit 11 is compared with bit 11 of the mask.
IN_BIT12	Input	BOOL	I, Q, M, D, L	Input bit 12 is compared with bit 12 of the mask.

Parameters	Declaration	Data type	Memory area	Description
IN_BIT13	Input	BOOL	I, Q, M, D, L	Input bit 13 is compared with bit 13 of the mask.
IN_BIT14	Input	BOOL	I, Q, M, D, L	Input bit 14 is compared with bit 14 of the mask.
IN_BIT15	Input	BOOL	I, Q, M, D, L	Input bit 15 is compared with bit 15 of the mask.
CMP_STEP	Input	BYTE	I, Q, M, D, L, P	The step number of the mask used for the comparison.
OUT	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that a match was found. A signal state of "0" indicates that no match was found.
ERR_COD E	Output	WORD	I, Q, M, D, L, P	Error information
CMP_VAL	Static	ARRAY OF WORD	I, Q, M, D, L	Comparison masks [0 to 15, 0 to 15]: The first number of the index is the step number and the second number is the bit number of the mask.

The static parameters are not visible when calling the instruction in the program. These are saved in the instance of the instruction.

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

Error code* (W#16#...)	Explanation
0000	No error
000A	The value at the CMP_STEP parameter is greater than 15.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

SMC: Compare scan matrix (S7-1500)

Description

You can use the "Compare scan matrix" instruction to compare the signal state of up to 16 programmed input bits (IN_BIT0 to IN_BIT15) with the corresponding bit of the comparison masks for each step. Processing starts at step 1 and is continued until the last programmed step (LAST) or until a match is found. The input bit of the IN_BIT0 parameter is compared with the value of the mask CMP_VAL[x,0], with "x" indicating the step number. All programmed values are compared in the same manner. If a match is found the signal state of the OUT parameter is set to "1" and the step number with the matching mask is written in the OUT_STEP parameter. Unprogrammed input bits or unprogrammed bits of the mask have a default signal state FALSE. If more than one step has a matching mask, only the first one found is indicated in the OUT_STEP parameter. If no match is found, the signal state of the OUT parameter is set to "0". In this case the value at the OUT_STEP parameter is greater by "1" than the value at the LAST parameter.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN_BIT0	Input	BOOL	I, Q, M, D, L	Input bit 0 is compared with bit 0 of the mask.
IN_BIT1	Input	BOOL	I, Q, M, D, L	Input bit 1 is compared with bit 1 of the mask.
IN_BIT2	Input	BOOL	I, Q, M, D, L	Input bit 2 is compared with bit 2 of the mask.
IN_BIT3	Input	BOOL	I, Q, M, D, L	Input bit 3 is compared with bit 3 of the mask.
IN_BIT4	Input	BOOL	I, Q, M, D, L	Input bit 4 is compared with bit 4 of the mask.
IN_BIT5	Input	BOOL	I, Q, M, D, L	Input bit 5 is compared with bit 5 of the mask.
IN_BIT6	Input	BOOL	I, Q, M, D, L	Input bit 6 is compared with bit 6 of the mask.
IN_BIT7	Input	BOOL	I, Q, M, D, L	Input bit 7 is compared with bit 7 of the mask.
IN_BIT8	Input	BOOL	I, Q, M, D, L	Input bit 8 is compared with bit 8 of the mask.
IN_BIT9	Input	BOOL	I, Q, M, D, L	Input bit 9 is compared with bit 9 of the mask.
IN_BIT10	Input	BOOL	I, Q, M, D, L	Input bit 10 is compared with bit 10 of the mask.
IN_BIT11	Input	BOOL	I, Q, M, D, L	Input bit 11 is compared with bit 11 of the mask.
IN_BIT12	Input	BOOL	I, Q, M, D, L	Input bit 12 is compared with bit 12 of the mask.

Parameters	Declaration	Data type	Memory area	Description
IN_BIT13	Input	BOOL	I, Q, M, D, L	Input bit 13 is compared with bit 13 of the mask.
IN_BIT14	Input	BOOL	I, Q, M, D, L	Input bit 14 is compared with bit 14 of the mask.
IN_BIT15	Input	BOOL	I, Q, M, D, L	Input bit 15 is compared with bit 15 of the mask.
OUT	Output	BOOL	I, Q, M, D, L	The signal state "1" indicates that a match was found. A signal state of "0" indicates that no match was found.
OUT_STEP	Output	BYTE	I, Q, M, D, L, P	Contains the step number with the matching mask or the step number which is greater by "1" than the value of the LAST parameter, provided no match is found.
ERR_CODE	Output	WORD	I, Q, M, D, L, P	Error information
LAST	Static	BYTE	I, Q, M, D, L, P	Specifies the step number of the last step to be scanned for a matching mask.
CMP_VAL	Static	ARRAY OF WORD	I, Q, M, D, L	Comparison masks [0 to 15, 0 to 15]: The first number of the index is the step number and the second number is the bit number of the mask.

The static parameters are not visible when calling the instruction in the program. These are saved in the instance of the instruction.

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

Error code* (W#16#...)	Explanation
0000	No error
000E	The value at the LAST parameter is greater than 15.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

LEAD_LAG: Lead and lag algorithm (S7-1500)

Description

You can use the "Lead and lag algorithm" instruction to process signals with an analog tag. The gain value at the GAIN parameter must be greater than zero. The result of the "Lead and lag algorithm" instruction is calculated using the following equation:

$$OUT = \left[\frac{LG_TIME}{LG_TIME + SAMPLE_T}\right] PREV_OUT + GAIN \left[\frac{LD_TIME + SAMPLE_T}{LG_TIME + SAMPLE_T}\right] IN - GAIN \left[\frac{LD_TIME}{LG_TIME + SAMPLE_T}\right] * PREV_IN$$

The "Lead and lag algorithm" instruction supplies plausible results only when processing is in fixed program cycles. The same units must be specified at the parameters LD_TIME, LG_TIME and SAMPLE_T. At LG_TIME > 4 + SAMPLE_T, the instruction approaches the following function:

OUT = GAIN * ((1 + LD_TIME * s) / (1 + LG_TIME * s)) * IN

When the value of the GAIN parameter is less than or equal to zero, the calculation is not performed and an error information is output on the ERR_CODE parameter.

You can use the instruction "Lead and lag algorithm" in conjunction with loops as a compensator in dynamic feed-forward control. The instruction consists of two operations. The "Lead" operation shifts the phase of output OUT so that the output leads the input. The "Lag" operation, on the other hand, shifts the output so that the output lags behind the input. Because the "Lag" operation is equivalent to an integration, it can be used as a noise suppressor or as a low-pass filter. The "Lead" operation is equivalent to a differentiation and can therefore be used as a high-pass filter. The two instructions together (Lead and Lag) result in the output phase lagging behind the input at lower frequencies and leading it at higher frequencies. This means that the "Lead and lag algorithm" instruction can be used as a band pass filter.

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN	Input	REAL	I, Q, M, D, L, P	The input value of the current sample time (cycle time) to be processed. Constants can also be specified at the IN parameter.
SAMPLE_T	Input	INT	I, Q, M, D, L, P	Sample time Constants can also be specified at the SAMPLE_T parameter.
OUT	Output	REAL	I, Q, M, D, L	Result of the instruction
ERR_CODE	Output	WORD	I, Q, M, D, L	Error information

Parameters	Declaration	Data type	Memory area	Description
LD_TIME	Static	REAL	I, Q, M, D, L, P	Lead time in the same unit as sample time.
LG_TIME	Static	REAL	I, Q, M, D, L, P	Lag time in the same unit as sample time
GAIN	Static	REAL	I, Q, M, D, L, P	Gain as % / % (the ratio of the change in output to a change in input as a steady state).
PREV_IN	Static	REAL	I, Q, M, D, L, P	Previous input
PREV_OUT	Static	REAL	I, Q, M, D, L, P	Previous output

The static parameters are not visible when calling the instruction in the program. These are saved in the instance of the instruction.

ERR_CODE parameter

The following table shows the meaning of the values of the ERR_CODE parameter:

Error code* (W#16#...)	Explanation
0000	No error
0009	The value at the GAIN parameter is less than or equal to zero.
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

Example

The following example shows how the instruction works:

Note

You can initialize static parameters in the data block.

SCL

```
"LEAD_LAG_DB"(IN := "Tag_Input",
              SAMPLE_T := "Tag_Input_SAMPLE_T",
              OUT => "Tag_Output_Result",
              ERR_CODE => "Tag_ErrorCode");
```

The following tables show how the instruction works using specific values.

Before processing

In this example the following values are used for the input parameters:

Parameters	Operand	Value
IN	Tag_Input	2.0
SAMPLE_T	Tag_Input_SAMPLE_T	10

The following values are saved in the "LEAD_LAG_DB" instance data block of the instruction:

Parameters	Address	Value
LD_TIME	DBD12	2.0
LG_TIME	DBD16	2.0
GAIN	DBD20	1.0
PREV_IN	DBD24	6.0
PREV_OUT	DBD28	6.0

After processing

The following values are written to the output parameters after the instruction has been executed:

Parameters	Operand	Value
OUT	Tag_Output_Result	2.0

The following values are saved in the instance data block "LEAD_LAD_DB" of the instruction:

Parameters	Operand	Value
PREV_IN	DBD24	2.0
PREV_OUT	DBD28	2.0

See also

Overview of the valid data types (Page 2239)

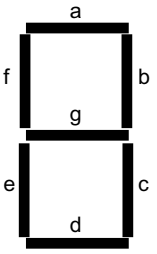
Basics of SCL (Page 4991)

SEG: Create bit pattern for seven-segment display (S7-1500)

Description

The "Create bit pattern for seven-segment display" instruction is used to convert each of the four hexadecimal digits of the specified source word (IN) into an equivalent bit pattern for a 7-segment display. The result of the instruction is output in the double word on the OUT parameter.

The following relation exists between the hexadecimal digits and the assignment of the 7 segments (a, b, c, d, e, f, g):

Input digit (Binary)	**Assignment of the segments - g f e d c b a**	**Display** (Hexadecimal)	**Seven-segment display**
0000	00111111	0	
0001	00000110	1	
0010	01011011	2	
0011	01001111	3	
0100	01100110	4	
0101	01101101	5	
0110	01111101	6	
0111	00000111	7	
1000	01111111	8	
1001	01100111	9	
1010	01110111	A	
1011	01111100	B	
1100	00111001	C	
1101	01011110	D	
1110	01111001	E	
1111	01110001	F	

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
IN	Input	WORD	I, Q, M, D, L, P	Source word with four hexadecimal digits
OUT	Output	DWORD	I, Q, M, D, L, P	Bit pattern for the seven-segment display
Function value		VOID		Empty function value

Example

The following example shows how the instruction works:

SCL

```
SEG(IN := "Tag_Input",
    OUT => "Tag_Output");
```

The following table shows how the instruction works using specific operand values:

Parameters	Operand	Value	
		Hexadecimal	Binary
IN	Tag_Input	W#16#1234	0001 0010 0011 0100
OUT	Tag_Output	DW16#065B4F66	00000110 01011011 01001111 01100110 Display: 1234

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

BCDCPL: Create tens complement (S7-1500)

Description

The "Create tens complement" instruction is used to create the tens complement of a seven-digit BCD number specified by the operand. This instruction uses the following mathematical formula to calculate:

10000000 (as BCD)

– 7-digit BCD value

Tens complement (as BCD)

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	Bit strings	I, Q, M, D, L, P	7-digit BCD number
Function value		DWORD	I, Q, M, D, L, P	Result of the instruction

For additional information on valid data types, refer to "See also".

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := BCDCPL("Tag_Input");
```

The following table shows how the instruction functions using specific values:

Operand	Value*
Tag_Input	DW#16#01234567
Tag_Result	DW#16#08765433
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

BITSUM: Count number of set bits (S7-1500)

Description

The "Count number of set bits" instruction is used to count the number of bits of an operand that are set to the signal state "1".

Parameters

The following table shows the parameters of the instruction:

Parameters	Declaration	Data type	Memory area	Description
<Operand>	Input	DWORD	I, Q, M, D, L, P	Operand whose set bits are counted
Function value		INT	I, Q, M, D, L, P	Result of the instruction

Example

The following example shows how the instruction works:

```
SCL
"Tag_Result" := BITSUM("Tag_Input");
```

The following table shows how the instruction functions using specific values:

Operand	Value*
Tag_Input	DW#16#12345678
Tag_Result	W#16#000D (13 bits)
*The error codes can be displayed as integer or hexadecimal value in the program editor. For information on toggling display formats, refer to "See also".	

See also

Overview of the valid data types (Page 2239)

Basics of SCL (Page 4991)

11.4.4 Extended instructions (S7-1200, S7-1500)

11.4.4.1 Date and time-of-day (S7-1200, S7-1500)

T_COMP: Compare time tags (S7-1500)

Description

This instruction is used to compare the contents of two tags of the "Timers" or "Date and time" data type.

The instruction supports the comparison of the following data types: DATE, TIME, LTIME, TOD (TIME_OF_DAY), LTOD (LTIME_OF_DAY), DT (DATE_AND_TIME), LDT (DATE_AND_LTIME), DTL, S5Time.

The data types must be the same length and format to carry out the comparison.

The comparison result is output at the OUT parameter as a return value. For this purpose, the parameter OUT is set to "1" if the condition used for the comparison is satisfied.

The following comparison options are available:

Symbol	Description
EQ	The return value has the signal state “1" if the time is the same at the parameters IN1 and IN2.
NE	The return value has the signal state “1" if the time at the parameter IN1 is not the same as that at the parameter IN2.
GE	The return value has the signal state "1" if the time at parameter IN1 is greater (more recent) than or equal to the time at parameter IN2 .
LE	The return value has the signal state "1" if the time at parameter IN1 is smaller (less recent) than or equal to the time at parameter IN2.
GT	The return value has the signal state "1" if the time at parameter IN1 is greater (more recent) than the time at parameter IN2.
LT	The return value has the signal state "1" if the time at parameter IN1 is smaller (older) than the time at parameter IN2 .

Parameters

The following table shows the parameters of the "T_COMP" instruction:

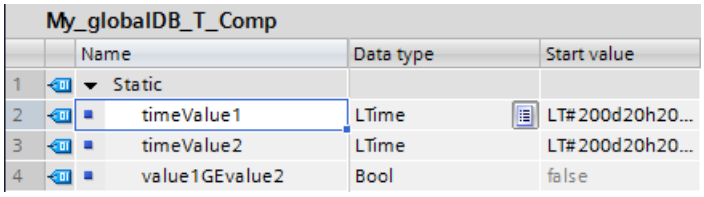
Parameter	Declaration	Data type	Memory area	Description
IN1	Input	DATE, TIME, LTIME, TOD, LTOD, DT, LDT, DTL, S5Time	I, Q, M, D, L, P or constant	First value to be compared.
IN2	Input	DATE, TIME, LTIME, TOD, LTOD, DT, LDT, DTL, S5Time	I, Q, M, D, L, P or constant	Second value to be compared.
OUT	Output	BOOL	I, Q, M, D, L, P	Return value

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

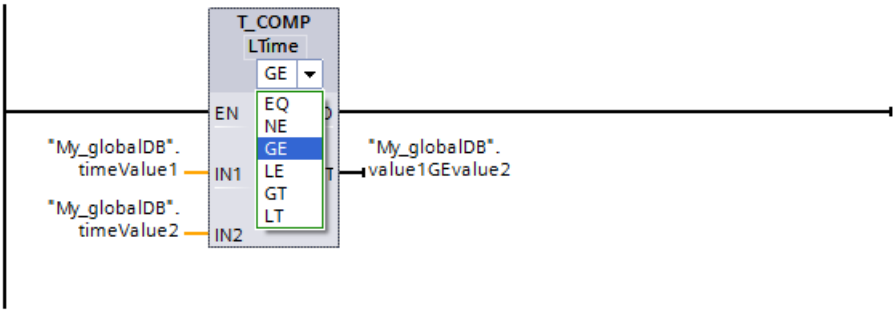
Example

In the following example, you use the "Greater or equal" comparison option to compare two times with the LTIME data type.

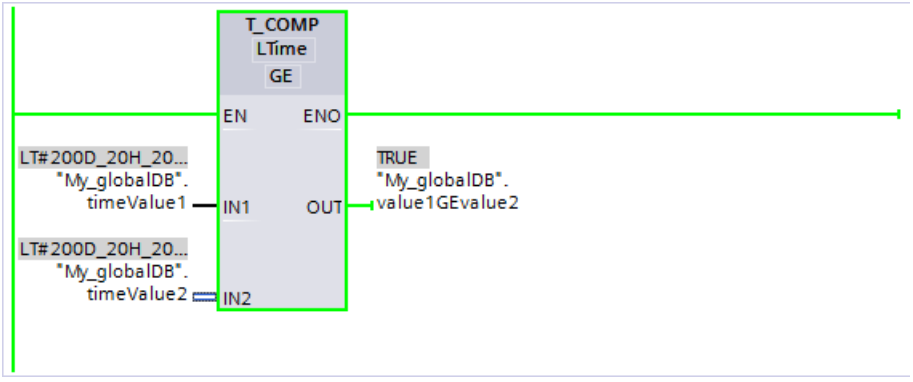
Create three tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows. Select the "GE" comparison option.



Since the time of the first value to be compared ("timeValue1") is greater than or equal to the second value ("timeValue2"), the return value ("value1GEvalue2") shows the signal state "TRUE".



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

T_CONV: Convert times and extract (S7-1200, S7-1500)

Description

You use the instruction "T_CONV" to convert the data type of the IN input parameter to the data type that is output at the OUT output. You select the data formats for the conversion from the instruction boxes of the input and output.

Parameters

The following table shows the parameters of the instruction "T_CONV". If an input and output parameter of the same data type is used, the instruction copies the corresponding value.

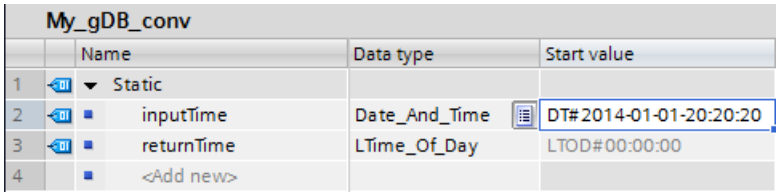
Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	Integers, TIME, date and time*	WORD, integers, timers, date and time*	I, Q, M, D, L, P or constant	Value to be converted
OUT	Return	Integers, TIME, date and time*	WORD, integers, timers, date and time*	I, Q, M, D, L, P	Result of the conversion
* The range of supported data types depends on the CPU. Please refer to the overviews of valid data types for information on the data types supported by the S7-1200 and S7-1500 modules.					

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

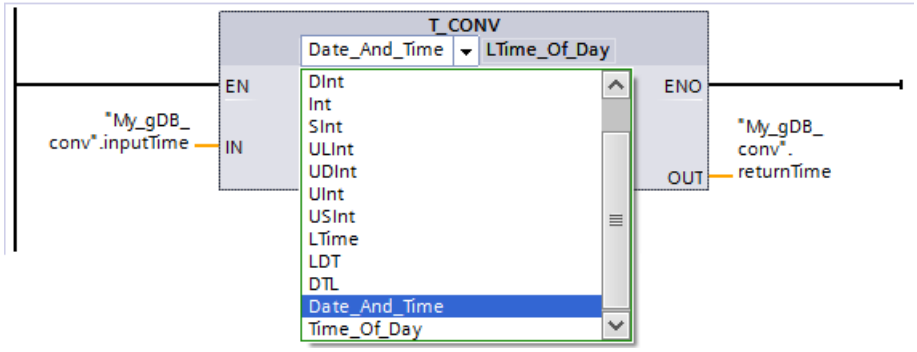
Example

In the following example, you convert a time with the DATE AND TIME data type to a time with the LTIME OF DAY data type.

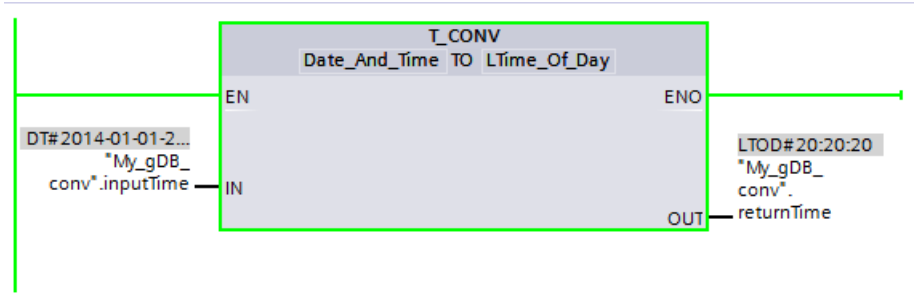
Create two tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows and select the data types.



The value to be converted ("inputTime") is output at the output parameter as a new value ("returnTime"). The information for the date is lost.



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

T_ADD: Add times (S7-1200, S7-1500)

Description

You use this instruction to add the time information in the IN1 input to the time information in the IN2 input. You can query the result in the OUToutput parameter. You can add the following formats:

- Addition of a time period to another time period.
  Example: Addition of a TIME data type to another TIME data type.
- Addition of a time period to a time.
  Example: Addition of a TIME data type to the DTL data type.

The data type for the value at input parameter IN1 and output parameter OUT is defined by the selection in the instruction boxes of the input and output. You can only specify time information in TIME format in the IN2 input parameter (for S7-1500 modules, also LTIME).

Note

Overflow/underflow

The largest possible value of the OUT output parameter may be exceeded or the smallest possible value undershot during the calculation. Evaluate the enable output ENO to detect this.

Parameters

The following tables show the parameters of the "T_ADD" instruction, according to the possible conversions:

Table 11-18 Addition of a time period to another time period

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN1	Input	TIME	TIME, LTIME	I, Q, M, D, L, P or constant	First number to be added
IN2	Input	TIME	TIME, LTIME	I, Q, M, D, L, P or constant	Second number to be added
OUT	Return	DINT, DWORD, TIME, TOD	TIME, LTIME,	I, Q, M, D, L, P	Result of addition The data type selection depends on the data types selected for the IN1 and IN2 input parameters.

Table 11-19 Addition of a time period to a time

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN1	Input	DTL, TOD	DT, TOD, LTOD, LDT, DTL	I, Q, M, D, L, P or constant	First number to be added For LTIME at parameter IN2, only LTOD, LDT or DTL can be used.
IN2	Input	TIME	TIME, LTIME	I, Q, M, D, L, P or constant	Second number to be added
OUT	Return	DINT, DWORD, TIME, TOD, UDINT, DTL	DT, DTL, LDT, TOD, LTOD	I, Q, M, D, L, P	Result of addition The data type selection depends on the data types selected for the IN1 and IN2 input parameters.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

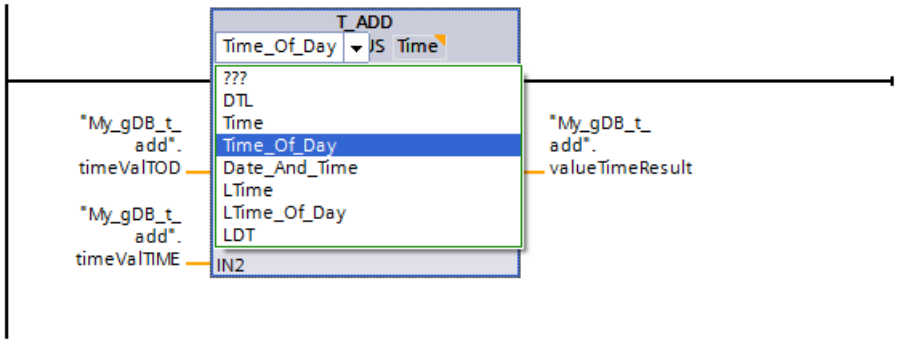
In the following example, you add a time duration with the TIME data type to a time of day with the TOD data type.

Create three tags in a global data block for storing the data.

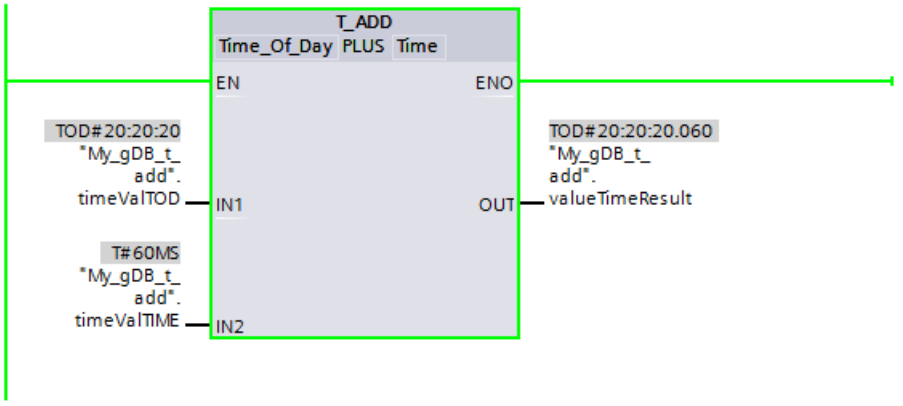
My_gDB_t_add

		Name	Data type	Start value
1		Static		
2		timeValTOD	Time_Of_Day	TOD#20:20:20
3		timeValTIME	Time	T#60ms
4		valueTimeResult	Time_Of_Day	TOD#00:00:00

Interconnect the parameters of the instruction as follows. You select the data types for time of day ("timeValTOD") and time period ("timeValTIME").



The time of day ("timeValTOD") and the time period ("timeValTIME") are added and the result is displayed as a time of day at output parameter OUT ("valueTimeResult").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

T_SUB: Subtract times (S7-1200, S7-1500)

Description

You use this instruction to subtract the time information in the IN2 input parameter from the time information in the IN1 input parameter. You can query the difference at the OUT output parameter. You can subtract the following formats:

- Subtraction of a time period from another time period
  Example: Subtraction of a time period of the data type TIME from another time period of the data type TIME. The result can be output to a tag with the TIME format.
- Subtraction of a time period from a time
  Example: Subtraction of a time period of the data type TIME from a time of the data type DTL. The result can be output to a tag with the DTL format.

You decide the formats of the values in the IN1 input parameter and the OUT output parameter by selecting the data types for the input and output parameters of the instruction.

Note

Overflow/underflow

The largest possible value of the OUT output parameter may be exceeded or the smallest possible value undershot during the calculation. Evaluate the enable output ENO to detect this.

Parameters

The following tables show the parameters of the "T_SUB" instruction, according to the possible conversions:

Table 11-20 Subtraction of a time period from another time period

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN1	Input	TIME	TIME, LTIME	I, Q, M, D, L, P or constant	Minuend
IN2	Input	TIME	TIME, LTIME	I, Q, M, D, L, P or constant	Subtrahend
OUT	Return	DINT, DWORD, TIME, TOD, UDINT	TIME, LTIME	I, Q, M, D, L, P	Result of subtraction

Table 11-21 Subtraction of a time period from a time

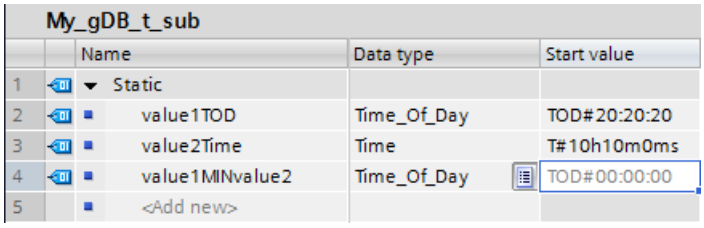
Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN1	Input	DTL, TOD	TOD, LTOD, DTL, DT, LDT	I, Q, M, D, L, P or constant	Minuend For LTIME at parameter IN2, only LTOD, LDT or DTL are used.
IN2	Input	TIME	TIME, LTIME	I, Q, M, D, L, P or constant	Subtrahend
OUT	Return	DTL, DINT, DWORD, TIME, TOD, UDINT	TOD, LTOD, DTL, DT, LDT	I, Q, M, D, L, P	Result of subtraction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

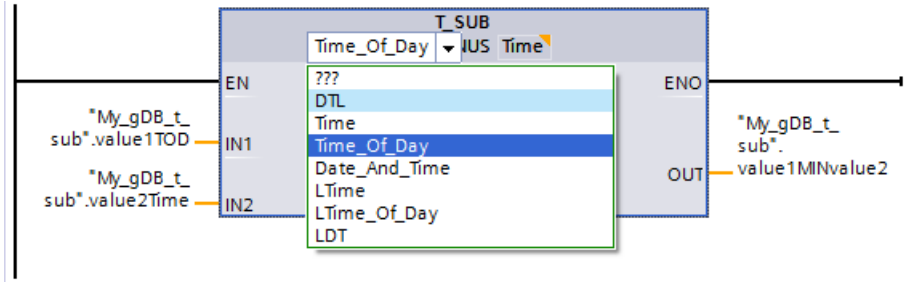
Example

In the following example, you subtract a time period with the TIME data type from a time of day with the TOD data type.

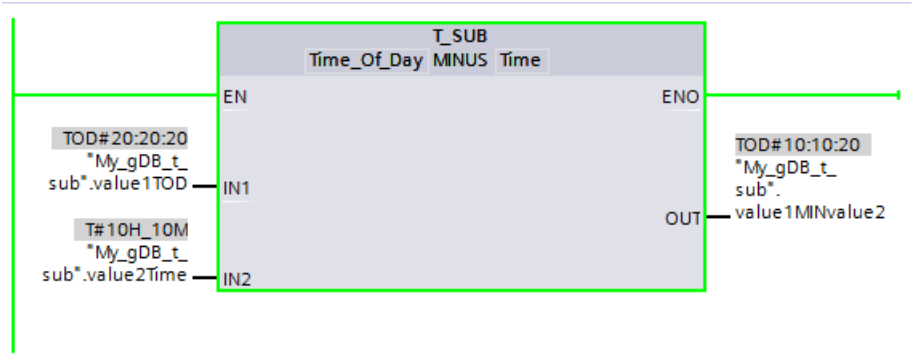
Create three tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows. You select the data type for the time of day ("value1TOD") and the data type for the time period ("value2Time").



The time of day ("value1TOD") and the time period ("value2Time") are subtracted and the result is displayed as a time of day at output parameter OUT ("value1MINvalue2").



You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

T_DIFF: Time difference (S7-1200, S7-1500)

Description

You use this instruction to subtract the time information in the IN2 input parameter from the time information in the IN1 input parameter. The result is sent at output parameter OUT.

- If the time information at the IN2 input parameter is greater than the time information at the IN1 input parameter, the result is output as a negative value at the OUT output parameter.
- If the result of the subtraction is outside the TIME number range, the enable output ENO is = "0". Depending on the data types used, you get either a truncated result or "0" (0:00).
  - If you select the minuend and subtrahend parameters of the DTL data type, the result has the TIME data type. It may not be greater than 24 days, otherwise the enable output ENO = "0" and the result is "0".
  - If you select the minuend and subtrahend parameters of the LDT data type, this restriction is avoided.

Parameters

The following table shows the parameters of the "T_DIFF" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN1	Input	DTL, DATE, TOD	DTL, DATE, DT, TOD, LTOD, LDT	I, Q, M, D, L, P or constant	Minuend
IN2	Input	DTL, DATE, TOD	DTL, DATE, DT, TOD, LTOD, LDT	I, Q, M, D, L, P or constant	Subtrahend
OUT	Return	TIME, INT	TIME, LTIME, INT	I, Q, M, D, L, P	Difference of input parameters

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

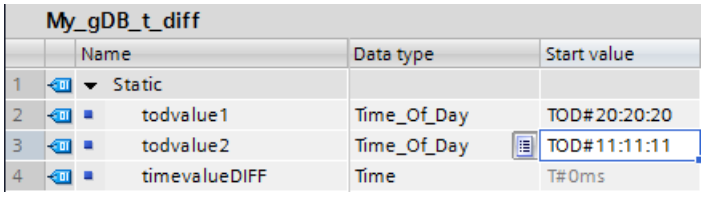
Use of the "T_DIFF instruction in SCL

In SCL, the data type TIME is preset for the OUT parameter (see also: Basic information on the data types of SCL instructions (Page 5034) If you want to use the data type INT for OUT, you need to extend the instruction name T_DIFF by adding "_INT" (see also: Changing the data type of an SCL instruction (Page 5035)).

Example

In the following example, you calculate the difference between two times of day with the TOD data type. You specify the difference in the "TIME" data type.

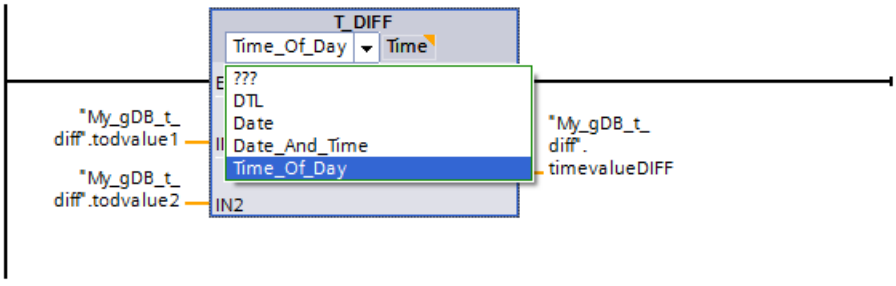
Create three tags in a global data block for storing the data.



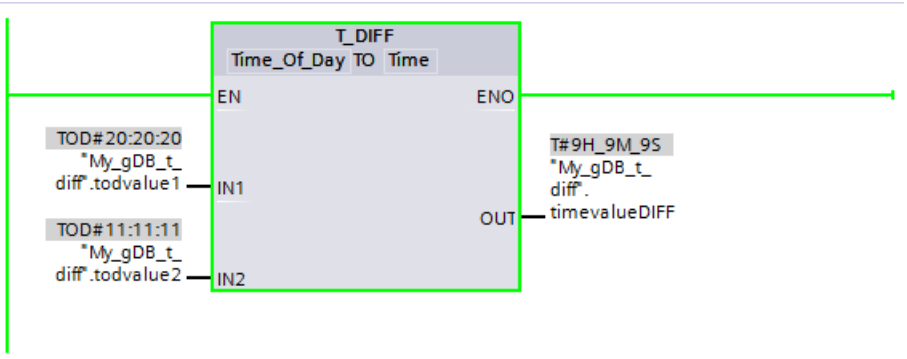
My_gDB_t_diff

	Name	Data type	Start value
1	Static		
2	todvalue1	Time_Of_Day	TOD#20:20:20
3	todvalue2	Time_Of_Day	TOD#11:11:11
4	timevalueDIFF	Time	T#0ms

Interconnect the parameters of the instruction as follows and select the data types. With the first selection option, you specify the data type of the points in time. With the second selection option, you specify the data type of the difference.



The first time of day ("todvalue1") and the second time of day ("todvalue2") are subtracted and the difference is displayed as a time period at output parameter OUT ("timevalueDIFF").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

T_COMBINE: Combine times (S7-1200, S7-1500)

Description

The instruction combines the value of a date with the value of a time, resulting in a combined date and time value.

- The date is output at the input parameter IN1. Use a value between DATE and 1990-01-01 for the data type 2089-12-31 (this is not checked).
- The time is input at the IN2 input value (TOD/LTOD data type).
- The combined data type for the date and time value is output at the OUT output value.

Parameters

The following table shows the parameters of the "T_COMBINE" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN1	Input	DATE	DATE	I, Q, M, D, L, P or constant	Input tag of the date
IN2	Input	TOD	TOD, LTOD	I, Q, M, D, L, P or constant	Input tag of the time
OUT	Return	DTL	DT, DTL, LDT	I, Q, M, D, L, P	Return value of the date and time

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

Use of T_COMBINE in SCL programs

When you drag the instruction "T_COMBINE" from the "Instructions" task card to the SCL programming window, the internal instruction CONCAT_DATE_TOD or CONCAT_DATE_LTOD is displayed there.

Example

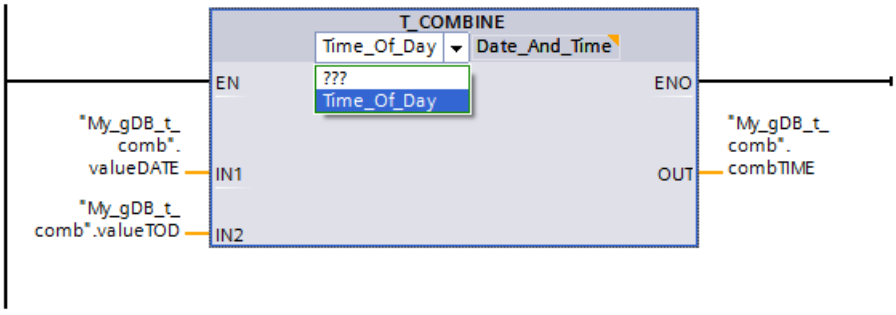
In the following example, you combine a time of day with the TOD data type and a date with the DATE data type. You specify the return value in the DT data type.

Create three tags in a global data block for storing the data.

My_gDB_t_comb

	Name	Data type	Start value
1	Static		
2	valueDATE	Date	D#2014-01-01
3	valueTOD	Time_Of_Day	TOD#20:22:20
4	combTIME	Date_And_Time	DT#1990-01-01-00:00:00

Interconnect the parameters of the instruction as follows and select the data types. With the first selection option, you specify the data type of the time of day ("valueTOD"). With the second selection option, you specify the data type of the return value ("combTIME").



The date ("valueDATE") is expanded by the specification of the time of day ("valueTOD") and the return value is displayed at output parameter OUT ("combTIME").

My_gDB_t_comb

	Name	Data type	Start value	Monitor value
1	Static			
2	valueDATE	Date	D#2014-01-01	D#2014-1-1
3	valueTOD	Time_Of_Day	TOD#20:22:20	TOD#20:22:20
4	combTIME	Date_And_Time	DT#1990-01-01-0	DT#2014-01-01-20:22:20

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

Time-of-day functions (S7-1200, S7-1500)

WR_SYS_T: Set time-of-day (S7-1200, S7-1500)

Description

You use this instruction to set the date and time-of-day (module time) of the CPU clock. Enter the date and time-of-day at the input parameter IN . The value must be in the following range:

- With DT: min. DT#1990-01-01-0:0:0, max. DT#2089-12-31-23:59:59.999
- With LDT: min. LDT#1970-01-01-0:0:0.000000000, max. LDT#2200-12-31-23:59:59.999999999
- With DTL: min. DTL#1970-01-01-00:00:00.0, max. DTL#2200-12-31-23:59:59.999999999

You can query whether errors have occurred during execution of the instruction in the RET_VAL output parameter.

The "WR_SYS_T" instruction is not used to pass information about the local time zone or daylight saving time.

Note

Module time of CPU clock

The CPU clock interprets the module time as the coordinated universal time (UTC). Accordingly, the module time is always stored without the factors "local time zone" or "summer time" in the CPU clock. The CPU clock then calculates the local time of the CPU clock based on the module time.

The module time of the CPU clock is used as template for all time processes starting from the CPU. Examples of use:

- Calculation of local time of CPU clock based on module time
- Representation of the module time in local time under "Online & Diagnostics"
- Block properties
- Entries in the diagnostics buffer of the CPU

Parameter

The following table shows the parameters of the "WR_SYS_T" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	DTL	DT **, DTL, LDT	I, Q, M, D, L, P or constant *	Date and time
RET_VAL	Return	INT	INT	I, Q, M, D, L, P	Status of the instruction
* The data types DT and DTL are not used for the following memory areas: input, output and bit memory.					
** When using the data type DT the milliseconds are not transferred to the CPU.					

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

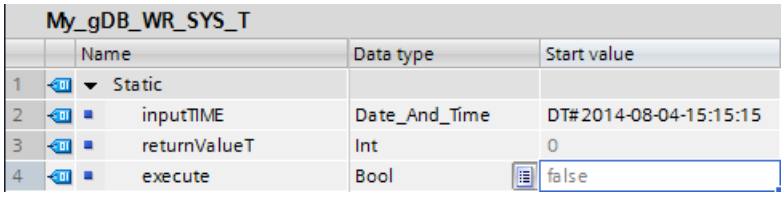
Parameter RET_VAL

Error code* (W#16#....)	Description
0000	No error
8080	Error in date
8081	Error in time
8082**	Invalid value specified for the month (byte 2 in DTL format).
8083**	Invalid value specified for the day (byte 3 in DTL format).
8084**	Invalid value specified for the hour (byte 5 in DTL format).
8085**	Invalid value specified for the minute (byte 6 in DTL format).
8086**	Invalid value specified for the second (byte 7 in DTL format).
8087**	Invalid value specified for the nanosecond (bytes 8 to 11 in DTL format).
80B0	The real-time clock has failed
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	
** Only if the date and time are specified at the parameter IN in DTL format	

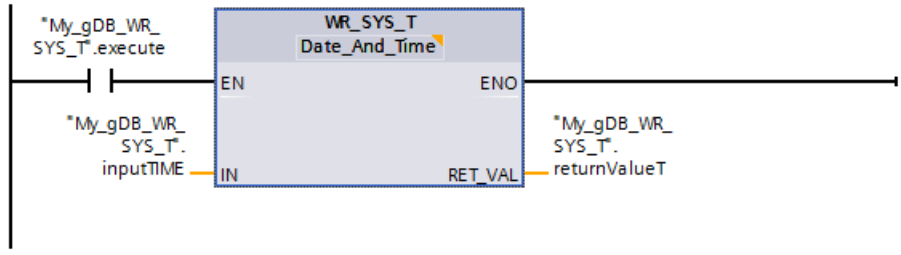
Example

In the following example, you set the date and time of the CPU clock. The data type used is DATE AND TIME.

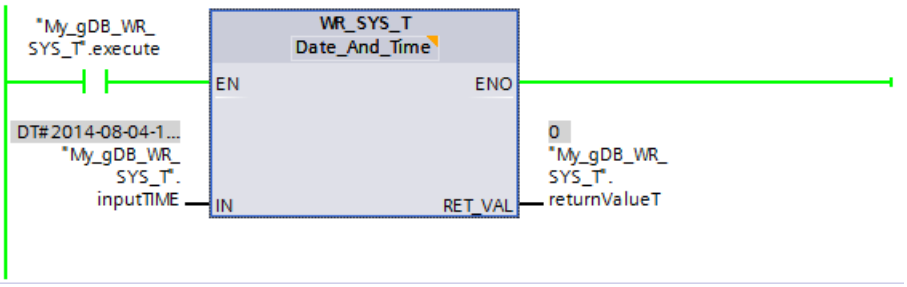
Create three tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows. Select the DATE AND TIME data type.



If the normally open contact ("execute") supplies the signal state "TRUE", the "WR_SYS_T" instruction is executed. The module time of the CPU clock is overwritten with the time to be set ("inputTIME"). The output parameter RET_VAL ("returnValueT") indicates that processing took place without errors.



You can establish whether the new module time ("inputTIME") has been correctly received by the CPU clock as follows:

- Using the display of an S7-1500 CPU: Navigate on the CPU display to "Settings > Date & Time > General".
- Using TIA Portal: Read out the module time of the CPU clock using the "RD_SYS_T (Page 3469)" instruction.
- Using TIA Portal: Navigate to the "Online & Diagnostics" entry of the CPU, and open the "Functions > Set time of day" tab.
  The Coordinated Universal Time (UTC) is set for the module time of the CPU clock. Central European Time is set as local time in the TIA Portal in this example. Accordingly, an hour is added to the time to be set ("inputTIME") in the "Online & Diagnostics" entry in the TIA Portal. Another hour is added when daylight saving time is used. The calculated local time is output in 12-hour clock format.



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

RD_SYS_T: Read time-of-day (S7-1200, S7-1500)

Description

You use this instruction to read the current date and current time-of-day (module time) of the CPU clock.

The read dates are output at the OUT output parameter of the instruction. The provided value does not include information about the local time zone or daylight saving time.

Note

Module time of CPU clock

The CPU clock interprets the module time as the coordinated universal time (UTC). Accordingly, the module time is always stored without the factors "local time zone" or "summer time" in the CPU clock. The CPU clock then calculates the local time of the CPU clock based on the module time.

The module time of the CPU clock is used as template for all time processes starting from the CPU. Examples of use:

- Calculation of local time of CPU clock based on module time
- Representation of the module time in local time under "Online & Diagnostics"
- Block properties
- Entries in the diagnostics buffer of the CPU

You can query whether errors have occurred during execution of the instruction in the RET_VALoutput.

Parameters

The following table shows the parameters of the "RD_SYS_T" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
RET_VAL	Return	INT	INT	I, Q, M, D, L, P	Status of the instruction
OUT	Output	DTL	DT, DTL, LDT	I, Q, M, D, L, P *	Date and time of CPU
* The data types DT and DTL cannot be used for the memory areas: input, output and bit memory.					

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#....)	Description
0000	No error
8081	Time value specified at the OUT parameter is outside the valid value range: • With DT: min. DT#1990-01-01-0:0:0, max. DT#2089-12-31-23:59:59.999 • With LDT: min. LDT#1970-01-01-0:0:0.000000000, max. LDT#2262-04-11-23:47:16.854775807 • With DTL: min. DTL#1970-01-01-00:00:00.0, max. DTL#2262-04-11-23:47:16.854775807
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Example

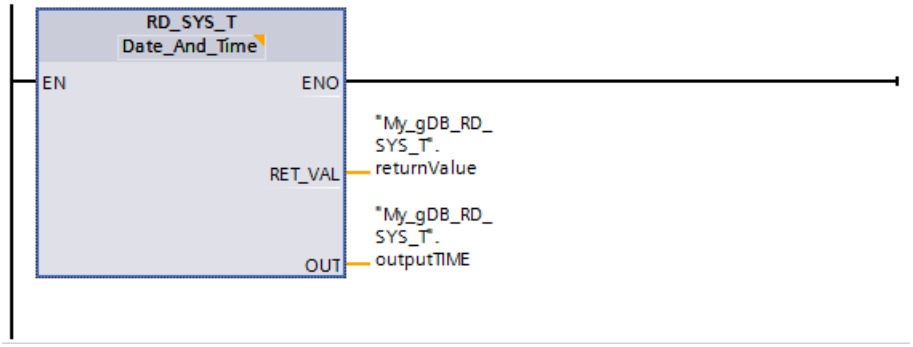
In the following example, you read out the module time of the CPU clock. The data type used is DATE AND TIME.

Create two tags in a global data block for storing the data.

My_gDB_RD_SYS_T

	Name	Data type	Start value
1	Static		
2	outputTIME	Date_And_Time	DT#1990-01-01-00:00:00
3	returnValue	Int	0

Interconnect the parameters of the instruction as follows. Select the DATE AND TIME data type.



The module time of the CPU clock is read out and displayed at output parameter OUT ("outputTIME"). The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.

My_gDB_RD_SYS_T

	Name	Data type	Start value	Monitor value
1	Static			
2	outputTIME	Date_And_Time	DT#1990-01-01-00:00:00	DT#2014-08-04-15:15:15
3	returnValue	Int	0	0

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

RD_LOC_T: Read local time (S7-1200, S7-1500)

Description

You use this instruction to read the current local time from the CPU clock and output this at the OUT output. Information on the time zone and the start of daylight saving time and standard time, which you have set in the configuration of the CPU clock, is used to output the local time.

Parameters

The following table shows the parameters of the "RD_LOC_T" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
RET_VAL	Return	INT	INT	I, Q, M, D, L, P	Status of the instruction
OUT	Output	DTL	DT, LDT, DTL	I, Q, M, D, L, P *	Local time
* The data types DT and DTL cannot be used for the memory areas: input, output and bit memory.					

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#....)	Description
0000	No error
0001	No error. Local time is output as daylight saving time.
8080	Local time cannot be read.
8081	Time value specified at the OUT parameter is outside the valid value range: • With DT: min. DT#1990-01-01-0:0:0, max. DT#2089-12-31-23:59:59.999 • With LDT: min. LDT#1970-01-01-0:0:0.000000000, max. LDT#2262-04-11-23:47:16.854775807 • With DTL: min. DTL#1970-01-01-00:00:00.0, max. DTL#2262-04-11-23:47:16.854775807
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Example

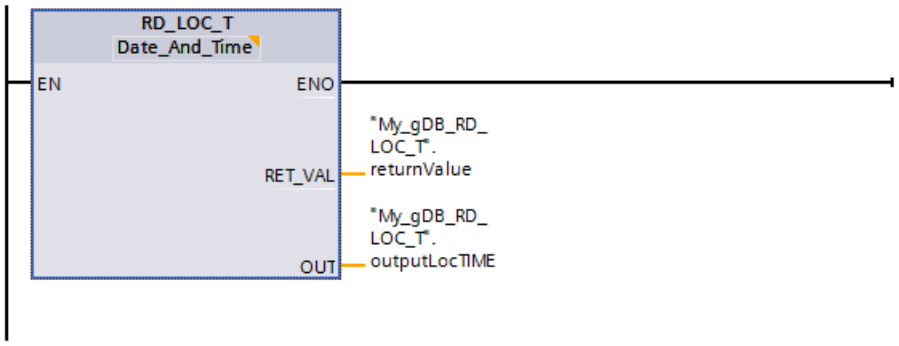
In the following example, you read out the local time of the CPU clock. The data type used is DATE AND TIME.

Create two tags in a global data block for storing the data.

My_gDB_RD_LOC_T

	Name	Data type	Start value
1	Static		
2	outputLocTIME	Date_And_Time	DT#1990-01-01-00:00:00
3	returnValue	Int	0

Interconnect the parameters of the instruction as follows. Select the DATE AND TIME data type.



The local time of the CPU clock is read out and displayed at output parameter OUT ("outputLocTIME"). Output parameter RET_VAL ("returnValue") indicates that processing took place without errors and the local time is output as daylight saving time with this call.

My_gDB_RD_LOC_T

		Name	Data type	Start value	Monitor value
1		Static			
2		outputLocTIME	Date_And_Time	DT#1990-01-01-00:00:00	DT#2014-08-04-16:15:15.001
3		returnValue	Int	0	1

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

WR_LOC_T: Write local time (S7-1200, S7-1500)

Description

The instruction "WR_LOC_T" is used to set the date and time of the CPU clock. Enter the date and time-of-day as local time at the input parameter LOCTIME.

The value must be in the following range:

- With DTL: min. DTL#1970-01-01-00:00:00.0, max. DTL#2200-12-31 23:59:59.999999999
- With LDT: min. LDT#1970-01-01-0:0:0.000000000, max. LDT#2200-12-31 23:59:59.999999999

The granularity of the time information for local time and system time is product-specific and is at least one millisecond. Input values at the LOCTIME parameter which are less than those supported by the CPU are rounded up during system time calculation.

You can query whether errors have occurred during execution of the instruction in the RET_VAL output parameter.

Parameters

The following table shows the parameters of the "WR_LOC_T" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
LOCTIME	Input	DTL	DTL, LDT	I, Q, M, D, L, P or constant *	Local time
DST	Input	BOOL	BOOL	I, Q, M, D, L, P, T, C or constant	**D**aylight **S**aving **T**ime Is only evaluated during the "double hour" at changeover to standard time. • TRUE = daylight saving time (first hour) • FALSE = standard time (second hour)
RET_VAL	Return	INT	INT	I, Q, M, D, L, P	Error message (see "RET_VAL parameter")
* The data type DTL cannot be used for the memory areas: input, output and bit memory.					

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter DST (Daylight Saving Time)

The parameter DST is evaluated only during the changeover from daylight saving to standard time:

- Use of DST during changeover from daylight saving to standard time
  The clock is set back by (1) one hour at 3 a.m. This means there are two hours between 02:00:00:000000000 AM and 02:59:59:999999999 AM.
  For all times for LOCTIME that are between 02:00:00:000000000 AM and 02:59:59:999999999 AM, you therefore need to declare whether the time is before or after the time changeover. The DST parameter is used for this:
  – With DST=TRUE, the time is in the first of the two hours, i.e. still in daylight saving time.
  – With DST=FALSE, the time is in the second of the two hours, i.e. in standard time.

  For all times for LOCTIME that are outside the double hour, DST is irrelevant.
- Changeover from standard to daylight saving time
  The parameter DST is not relevant for the changeover from standard to daylight saving time.
  The clock is set forward by (1) one hour at 2 a.m. This means there is no hour between 02:00:00:000000000 AM and 02:59:59:999999999 AM.
  If you specify a time within this hour at LOCTIME, the error code W#16#8089 is output.

(1)The description here assumes Central European Time (CET). The changeover time for CET is 2:00 AM (start of daylight saving), or 3:00 AM (end of daylight saving). Different changeover times may apply correspondingly to other time zones.

Parameter RET_VAL

Error code* (W#16#....)	Description
0000	No error.
8080	The LOCTIME parameter has an invalid value.
8081	Time value specified at the LOCTIME parameter is outside the valid value range: • With DTL: min. DTL#1970-01-01-00:00:00.0, max. DTL#2200-12-31 23:59:59.999999999 • With LDT: min. LDT#1970-1-1-0:0:0.000000000, max. LDT#2200-12-31 23:59:59.999999999
8082**	Invalid value specified for the month (byte 2 in DTL format).
8083**	Invalid value specified for the day (byte 3 in DTL format).
8084**	Invalid value specified for the hour (byte 5 in DTL format).
8085**	Invalid value specified for the minute (byte 6 in format DTL).
8086**	Invalid value specified for the second (byte 7 in DTL format).
8087**	Invalid value specified for the nanosecond (byte 8 to 11 in DTL format).
8089	Time value does not exist (hour already passed upon changeover to daylight saving time).
80B0	The real-time clock has failed.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	
** Only for local time information at the LOCTIME parameter in DTL format.	

Example

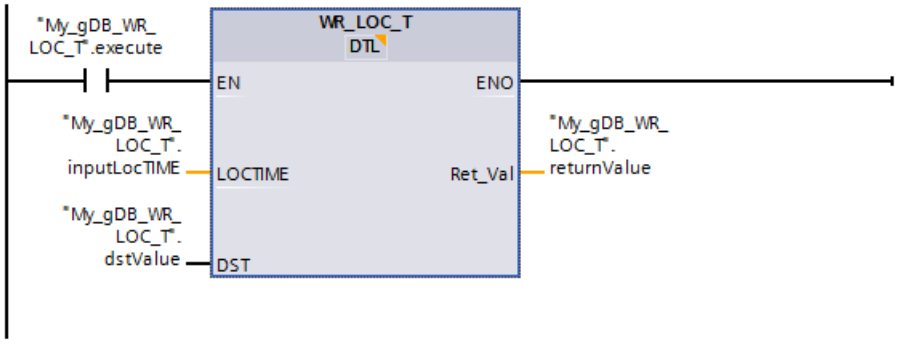
In the following example, you set the local time of the CPU clock. The data type used is DTL.

Create four tags in a global data block for storing the data.

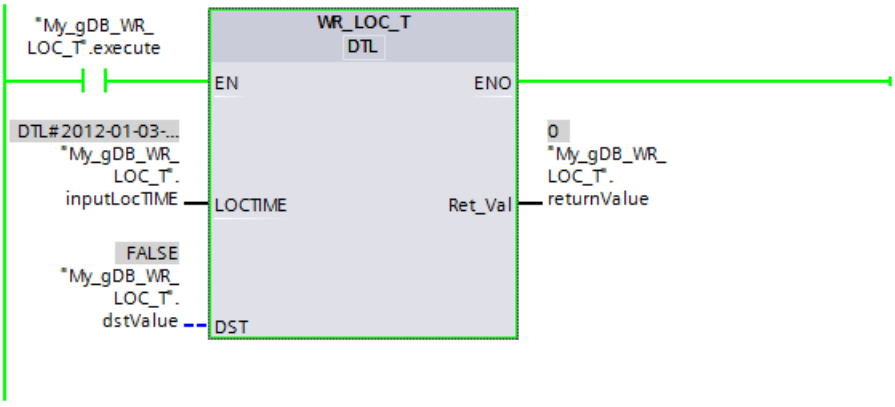
My_gDB_WR_LOC_T

	Name	Data type	Start value
1	Static		
2	inputLocTIME	DTL	DTL#2012-01-03-12:12:12
3	dstValue	Bool	false
4	returnValue	Int	0
5	execute	Bool	false

Interconnect the parameters of the instruction as follows. Select the DTL data type.



If the normally open contact ("execute") supplies the signal state "TRUE", the "WR_LOC_T" instruction is executed. The local time of the CPU clock is overwritten with the time to be set ("inputLocTIME"). The output parameter RET_VAL ("returnValue") indicates that the process took place without errors. Input parameter DST ("dstValue") specifies that the time information refers to standard time. This parameter is only relevant for the "double hour" (see description "Parameter DST").



You can establish whether the new local time ("inputLocTIME") has been correctly received by the CPU clock as follows:

- Using the display of an S7-1500 CPU: Navigate on the CPU display to "Settings > Date & Time > General".
- Using TIA Portal: Read out the local time of the CPU clock using the "RD_LOC_T (Page 3471)" instruction.
- Using TIA Portal: Navigate to the "Online & Diagnostics" entry of the CPU, and open the "Functions > Set time of day" tab.
  The local time is output in a 12-hour clock format.



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

SET_TIMEZONE: Set time zone (S7-1200, S7-1500)

Description

Use the instruction "SET_TIMEZONE" to set the parameter for the local time zone and the daylight saving / standard time changeover.

The settings which are carried out with the instruction "SET_TIMEZONE" correspond with the settings for the time-of-day in the properties of the CPU. Define the corresponding parameters in the system data type TimeTransformationRule for execution of the instruction "SET_TIMEZONE".

The local time is calculated based on the system time using the settings for the time zone and the daylight saving / standard time changeover. The system time of the CPU is the UTC time. The system time is used exclusively for communication within the system.

Note

Calling the "SET_TIMEZONE" instruction

The instruction "SET_TIMEZONE" writes data internally to the load memory of the CPU. Accordingly, the change in time zone also applies after a power failure. Calling the instruction again is unnecessary.

Call the instruction once every time the time zone has to be changed. For example, it is advisable to call "SET_TIMEZONE" in a startup OB.

Note

Use with CPUs of the S7-1500 series

The "SET_TIMEZONE" instruction can only be used with a CPU of the S7-1500 series starting from firmware version V1.7.

Parameters

The following table shows the parameters of the "SET_TIMEZONE" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, P or constant	Setting of parameters that are stored at parameter TimeZone.
TimeZone	Input	TimeTransformationRule	D, L	You interconnect the TimeTransformationRule system data type (see below) at parameter TimeZone.
DONE	Output	BOOL	I, Q, M, D, L, P	Status parameter: • 0: Job not yet started or still in progress • 1: Job completed without error
BUSY	Output	BOOL	I, Q, M, D, L, P	Status parameter: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started.
ERROR	Output	BOOL	I, Q, M, D, L, P	Status parameter: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L, P	Detailed error and status information is output at the parameter STATUS. The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter TimeZone

You store the parameters for the local time zone and the daylight saving/standard time changeover in the TimeTransformationRule system data type

You create the TimeTransformationRule by entering TimeTransformationRule as the data type in a data block or the local interface of a function block.

The structure of the TimeTransformationRule is as follows:

Name	Data type	Description
TimeTransformationRule	STRUCT	

Name	Data type	Description
Bias	INT	Time difference between the local time and the system time (UTC) in minutes. The value must be between -720 and +780 minutes (-12 to +13 hours). The value (UTC -12 to +13 hours) corresponds to the time zones that you specify in the properties of the CPU.
DaylightBias	INT	Time difference between standard and daylight saving time in minutes. The value must be between 0 and 120 minutes. • The value "0" deactivates the changeover between daylight saving time and standard time. The values for "DaylightStart..." and "StandardStart..." are set to "0". Only the value of Bias is evaluated (time difference local time/ system time). • If the value is not "0", all tags of the TimeTransformationRule structure are evaluated. If an entry is invalid, error code 808F is output at parameter STATUS.
Specification of the time of day for changeover to daylight saving time. The following times always refer to the local time.		
DaylightStartMonth	USINT	Month in which change is made to daylight saving time: 1 = January 2 = February 3 = March ... 12 = December
DaylightStartWeek	USINT	Week in which change is made to daylight saving time. 1 = First occurrence of the weekday in the month ... 5 = Last occurrence of the weekday in the month
DaylightStartWeek-day	USINT	Weekday on which change is made to daylight saving time: 1 = Sunday ... 7 = Saturday
DaylightStartHour	USINT	Hour in which change is made to daylight saving time
DaylightStartMinute	USINT	Minute in which change is made to daylight saving time
Specification of the time of day for changeover to standard time. The following times always refer to the local time.		
StandardStartMonth	USINT	Month in which change is made to standard time: 1 = January 2 = February 3 = March ... 12 = December
StandardStartWeek	USINT	Week in which change is made to standard time: 1 = First occurrence of the weekday in the month ...

Name		Data type	Description
			5 = Last occurrence of the weekday in the month
	StandardStartWeek-day	USINT	Weekday on which change is made to standard time: 1 = Sunday ... 7 = Saturday
	StandardStartHour	USINT	Hour in which change is made to standard time
	StandardStartMinute	USINT	Minute in which change is made to standard time
	TimeZoneName	STRING[80]	Not used: The configured character string is ignored and not written to the CPU-internal data.

Parameter STATUS

Error code* (W#16#....)	Description
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
808F	The structure, content or data types of the TimeTransformationRule at the TimeZone parameter are invalid or inconsistent.
80C3	Temporary resource error: The CPU is currently processing the maximum possible number of simultaneous block calls. "SET_TIMEZONE" cannot be executed until at least one of the block calls is finished.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Example

In the following example, you set the parameters for the local time zone and the daylight saving/standard time changeover. You are advised to call the example in a startup OB.

Create the following for storing the data in a global data block: The "timezone" structure (with the TimeTransformationRule data type) and five additional tags.

Note

If you want to call the example in a startup OB: Set the start value of the tag "execute" to "TRUE".

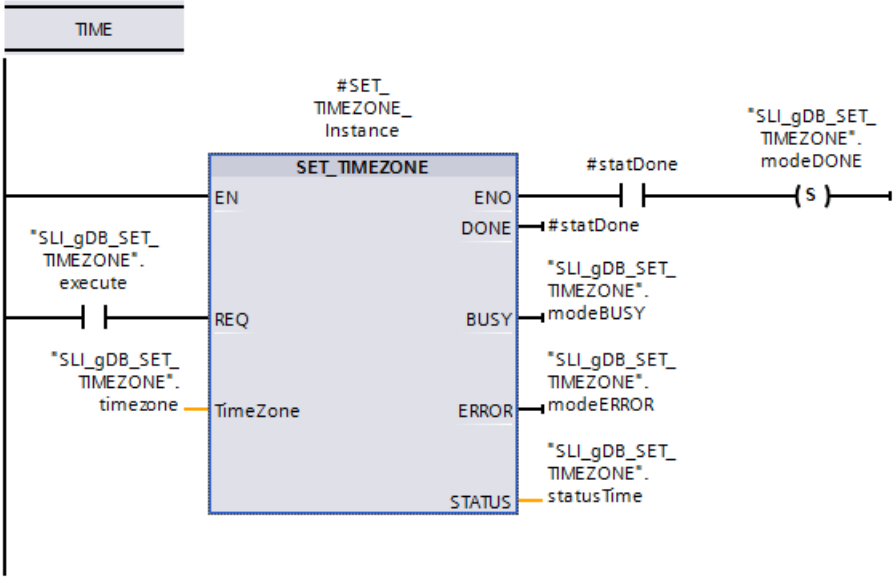
SLI_gDB_SET_TIMEZONE

	Name	Data type	Start value
1	Static		
2	execute	Bool	false
3	timezone	TimeTransformatio...	
4	Bias	Int	120
5	DaylightBias	Int	60
6	DaylightStartMonth	USInt	3
7	DaylightStartWeek	USInt	5
8	DaylightStartWeekday	USInt	7
9	DaylightStartHour	USInt	2
10	DaylightStartMinute	USInt	0
11	StandardStartMonth	USInt	10
12	StandardStartWeek	USInt	5
13	StandardStartWeekday	USInt	1
14	StandardStartHour	USInt	3
15	StandardStartMinute	USInt	0
16	TimeZoneName	String[80]	'My_GMT+'
17	modeDONE	Bool	false
18	modeBUSY	Bool	false
19	modeERROR	Bool	false
20	statusTime	Word	16#0
21	memErrStatus	Word	16#0

Create a local tag for the interconnection of the output parameter DONE.

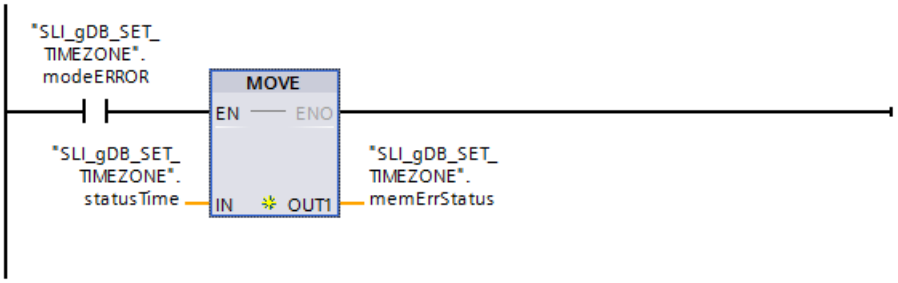
9	statDone	Bool	false	Non-retain

Network 1: Interconnect the parameters of the "SET_TIMEZONE" instruction as follows.

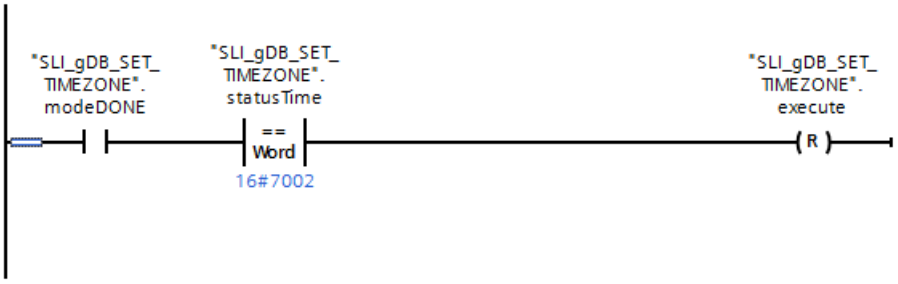


Also, create the following additional interconnections:

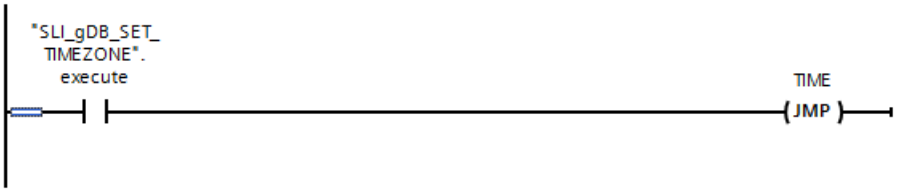
Network 2: In case of error, you save the status in the "memErrStatus" tag.



Network 3: After successful processing, have the input parameter REQ ("execute") automatically reset as follows.



Network 4: To ensure that the instruction "SET_TIMEZONE" is fully processed when called in a startup OB, use the instructions JMP (in network 4) and Label (see network 1).



Note

If you want to call the example in a startup OB, the CPU for processing the example must be in startup. Set the CPU to STOP mode before you load the example to the CPU. Then set the CPU to RUN mode.

If the input parameter REQ ("execute) returns the signal state "TRUE", the data for the time zone of the CPU clock is overwritten by the data to be set ("timezone"). That also means that:

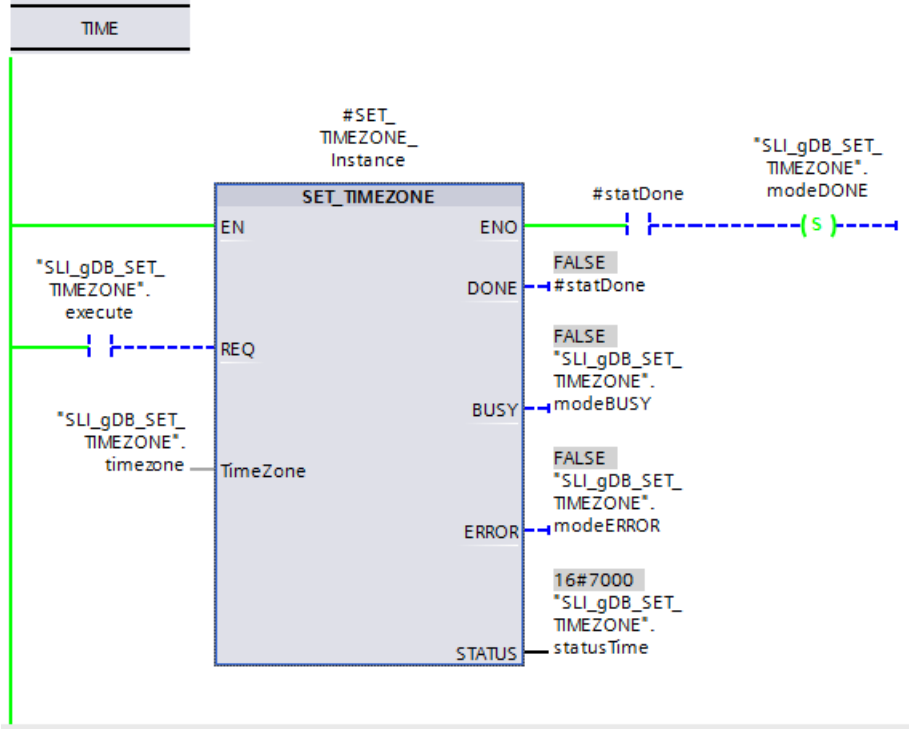
- Output parameter BUSY ("modeBUSY") goes to signal state "TRUE". After processing, output parameter BUSY shows the value "FALSE" and output parameter DONE ("#statDone") shows the value "TRUE". The success status is saved to the "modeDONE" tag.
- The output parameter STATUS ("statusTime") indicates how processing is running*. (*The job processing begins (value "7001") and the instruction is then shown as already active (value "7002").)
- The output parameter ERROR ("modeERROR") indicates that processing is running without errors (signal state is "FALSE").

SLI_gDB_SET_TIMEZONE

	Name	Data type	Start value	Monitor value
1	Static			
2	execute	Bool	false	FALSE
3	timezone	TimeTransformatio...		
4	Bias	Int	120	120
5	DaylightBias	Int	60	60
6	DaylightStartMonth	USInt	3	3
7	DaylightStartWeek	USInt	5	5
8	DaylightStartWeekday	USInt	7	7
9	DaylightStartHour	USInt	2	2
10	DaylightStartMinute	USInt	0	0
11	StandardStartMonth	USInt	10	10
12	StandardStartWeek	USInt	5	5
13	StandardStartWeekday	USInt	1	1
14	StandardStartHour	USInt	3	3
15	StandardStartMinute	USInt	0	0
16	TimeZoneName	String[80]	'My_GMT+'	'My_GMT+'
17	modeDONE	Bool	false	TRUE
18	modeBUSY	Bool	false	FALSE
19	modeERROR	Bool	false	FALSE
20	statusTime	Word	16#0	16#7000
21	memErrStatus	Word	16#0	16#0000

Note: The "SET_TIMEZONE" instruction is level-triggered. The instruction is executed only when the input parameter REQ ("execute") returns the signal state "TRUE".

After successful processing of SET_TIMEZONE ("modeDONE" is "TRUE"), the input parameter REQ ("execute") is automatically reset to the value "FALSE". The value "16#7002" is queried for "statusTime" for this purpose. After the reset, the status value "16#7000" is output.



You can establish whether the data to be set ("timezone") has been correctly received by the CPU clock as follows:

- Using the display of an S7-1500 CPU: Navigate on the CPU display to "Settings > Date & Time > Daylight Saving Time".
- Using TIA Portal: Read out the local time of the CPU clock using the "RD_LOC_T (Page 3471)" instruction.
- Using TIA Portal: Read out the module time of the CPU clock using the "RD_SYS_T (Page 3469)" instruction.

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

SNC_RTCB: Synchronize slave clocks (S7-1500)

Definition: Synchronization of slave clocks

The synchronization of clock slaves refers to the transfer of the date and time-of-day from the clock master of a bus segment to all clock slaves of this bus segment.

Description

You use this instruction to synchronize all slave clocks present locally on the CPU or locally on a bus segment independent of the set synchronization interval (the external interface of a CP or CM belongs to a different bus segment). Successful synchronization is only possible if "SNC_RTCB" is called on a CPU whose real-time clock was assigned as the master clock for at least one bus segment.

Parameters

The following table shows the parameters of the "SNC_RTCB" instruction:

Parameter	Declaration	Data type	Memory area	Description
RET_VAL	Output	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred during synchronization.
0001	The existing clock was not assigned the master clock function for any of the bus segments.

Error code* (W#16#...)	Explanation
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

TIME_TCK: Read system time (S7-1500)

Description

With the "TIME_TCK" instruction, you read the system time of the CPU. The system time is a time counter that counts from 0 to a maximum of 2147483647 ms. In case of an overflow, the system time is counted again starting with "0". The time scale and the accuracy of the system time is 1 ms. The system time is only influenced by the operating modes of the CPU. You can use the system time, for example, to measure the duration of processes by comparing the results of two "TIME_TCK" calls. The instruction does not provide any error information.

The following table provides an overview of how the system time changes depending on the operating modes of the CPU.

Operating mode	System time ...
Startup	... is constantly updated.
RUN	
STOP	... is stopped and retains the current value.
Warm restart	... is deleted and restarts with "0".

Parameters

The following table shows the parameters of the "TIME_TCK" instruction:

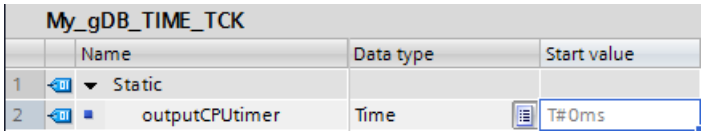
Parameter	Declaration	Data type	Memory area	Description
RET_VAL	Return	TIME	I, Q, M, D, L	The RET_VAL parameter contains the read system time in the range from 0 to 2^{31} -1 ms.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

In the following example, you read out the system time of the CPU. You specify the return value in the TIME data type.

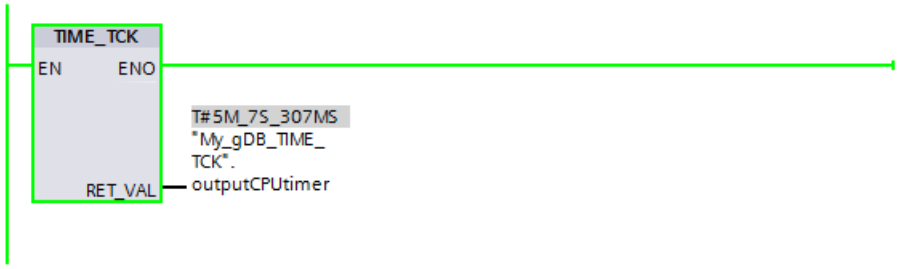
Create one tag for storing the data in a global data block.



Interconnect the parameters of the instruction as follows.



The system time of the CPU is read out and displayed at output parameter RET_VAL ("outputCPUtimer").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

RTM: Runtime meters (S7-1200, S7-1500)

Description

You can use this instruction to set, start, stop, and read out a 32-bit operating hours counter of your CPU.

Ensure that the operating hours counter can also be stopped or restarted during execution of the user program, which may render the saved values incorrect.

Parameters

The following table shows the parameters of the "RTM" instruction:

Parameter	Declaration	Data type	Memory area	Description
NR	Input	RTM	I, Q, M, D, L or constant	Number of the operating hours counter Numbering starts with 0. For information on the number of operating hours counters of your CPU, refer to the technical specifications.
MODE	Input	BYTE	I, Q, M, D, L or constant	Job ID: • 0: Read out (the status is then written to CQ and the current value to CV). After the operating hours counter has reached (2^31) - 1 hours, it stops at the highest value that can be displayed and outputs an "Overflow" error message. • 1: start (at the last counter value) • 2: stop • 4: set to the value specified in the parameter PV • 5: set to the value specified in the parameter PV and start • 6: set to the value specified in the parameter PV and stop
PV	Input	DINT	I, Q, M, D, L or constant	New value for the operating hours counter
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.
CQ	Output	BOOL	I, Q, M, D, L	Status of the operating hours counter (1: running)
CV	Output	DINT	I, Q, M, D, L	Current value of the operating hours counter

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code (W#16#...)	Explanation
0000	No error
8080	Wrong number for the operating hours counter
8081	A negative value was passed to the PV parameter.
8082	Overflow of the operating hours counter
8091	The MODE input parameter contains an invalid value.
General error information	See also: Evaluating errors with GET_ERR_ID (Page 3409)

Example

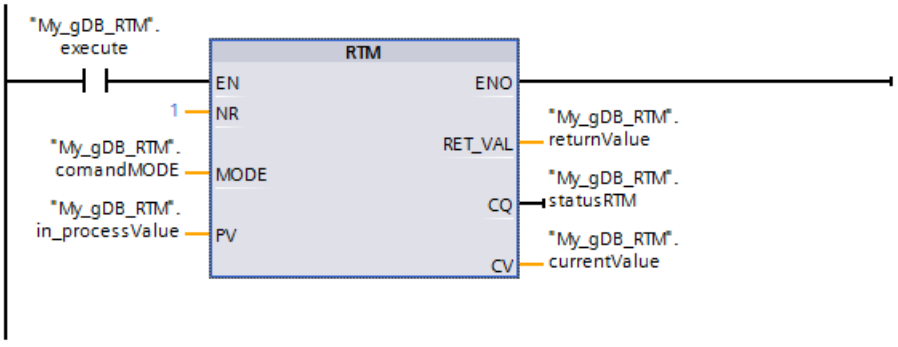
In the following example, you set the operating hours counter of the CPU and read out the value after an hour.

Create six tags in a global data block for storing the data.

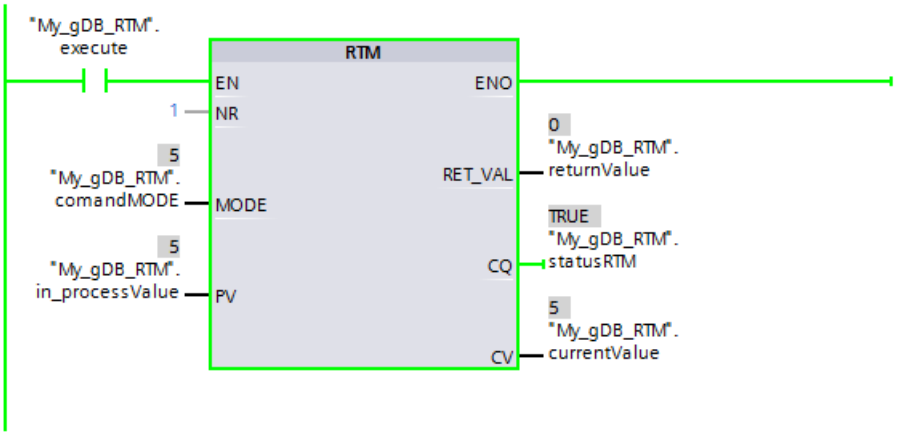
My_gDB_RTM

	Name	Data type	Start value
1	Static		
2	execute	Bool	false
3	in_processValue	DInt	5
4	returnValue	Int	0
5	statusRTM	Bool	false
6	currentValue	DInt	0
7	comandMODE	Byte	5

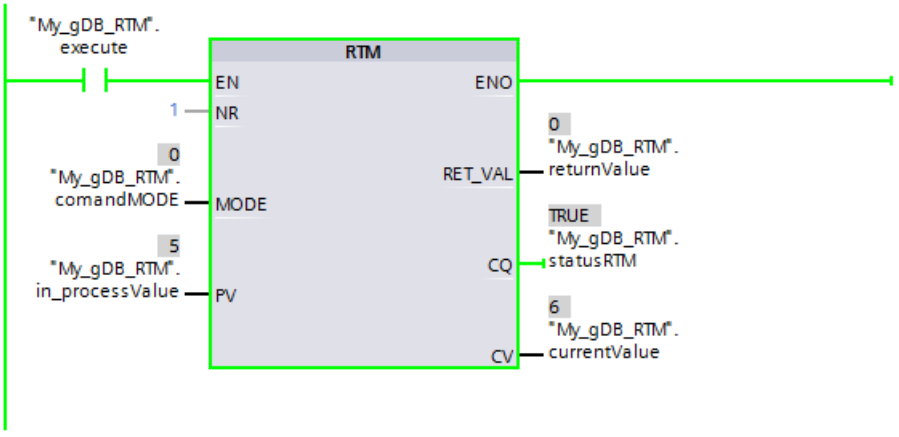
Interconnect the parameters of the instruction as follows. At input parameter NR, specify the number of the operating hours counter of the CPU.



If the normally open contact ("execute") supplies the signal state "TRUE", the "RTM" instruction is executed. The operating hours counter of the CPU is set to the value to be set ("in_processValue") and started. After the start of the operating hours counter, set the value of input parameter MODE ("comandMODE") to "0". (To do so, click the parameter and select "Modify operand > Set to 0".) As a result, the "RTM" instruction only reads the current value ("currentValue") of the operating hours counter without changing it. Output parameter CQ ("statusRTM") indicates after the start of the operating hours counter that the operating hours counter is running (value is "TRUE"). The output parameter RET_VAL ("returnValue") indicates that processing is running without errors.



After an hour, output parameter CV ("currentValue") indicates the value "6".



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

11.4.4.2 String + Char (S7-1200, S7-1500)

S_MOVE: Move character string (S7-1200, S7-1500)

Description

You can use this instruction to write the content of a character string (W)STRING from parameter IN to the data area that you specify at parameter OUT.

To copy tags of data type ARRAY, use the "MOVE_BLK" and "UMOVE_BLK" instructions.

Parameters

The following table shows the parameters of the "S_MOVE" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	STRING, WSTRING	D, L or constant	Source string
OUT	Output	STRING, WSTRING	D, L	Destination string

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

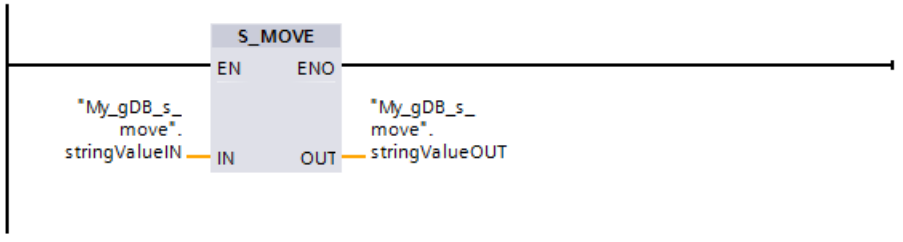
In the following example, you copy the content of a character string from input parameter IN to another character string at output parameter OUT. The data type used is STRING.

Create two tags in a global data block for storing the data.

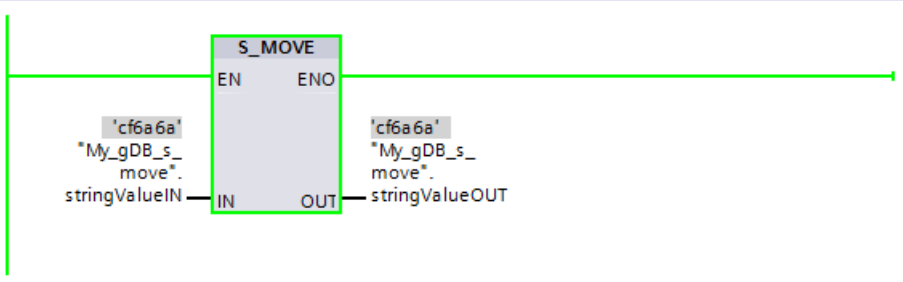
My_gDB_s_move

	Name	Data type	Start value
1	Static		
2	stringValueIN	String	'cf6a6a'
3	stringValueOUT	String	''

Interconnect the parameters of the instruction as follows.



The result of the character string to be copied ("stringValueIN") is output at output parameter OUT ("stringValueOUT").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

S_COMP: Compare character strings (S7-1500)

Description

The instruction compares the contents of two tags in the (W)STRING format and outputs the result of the comparison as a return value. The tags that are to be compared will be interconnected at the IN1 and IN2 inputs. You can only assign a symbolically defined tag for the input parameters.

Use the instruction box to select the comparison condition. If the comparison condition (for example, greater than or equal to) is satisfied, the signal state is set to "1" at the output parameter OUT .

Select one of the following comparison conditions:

Symbol	Description
EQ	The return value has the signal state "1" if the string at the parameter IN1 is equal to the string at the parameter IN2 .
NE	The return value has the signal state "1" if the string at the parameter IN1 is not equal to the string at the parameter IN2 .
GT (1)	The return value has the signal state "1" if the string at the parameter IN1 is greater than the string at the parameter IN2 .
LT (1)	The return value has the signal state "1" if the string at the parameter IN1 is less than the string at the parameter IN2 .
GE (1)	The return value has the signal state "1" if the string at the parameter IN1 is greater than or equal to the string at the parameter IN2 .
LE (1)	The return value has the signal state "1" if the string at the parameter IN1 is less than or equal to the string at the parameter IN2 .
(1) The characters are compared by their ASCII code (for example, 'a' is greater than 'A'), starting from the left. The first character to be different decides the result of the comparison. If the left part of the longer string is identical to the shorter string, the longer string is considered greater.	

Parameters

The following table shows the parameters of the "S_COMP" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN1	Input	STRING, WSTRING*	D, L or constant	Input tag in the STRING / WSTRING format.
IN2	Input	STRING, WSTRING*	D, L or constant	Input tag in the STRING / WSTRING format.
OUT	Output	BOOL	I, Q, M, D, L	Result of comparison
* Define the maximum length of the character string if you use the data type STRING / WSTRING in the interface declaration for a temporary tag (you will find further information in the description of the data type).				

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

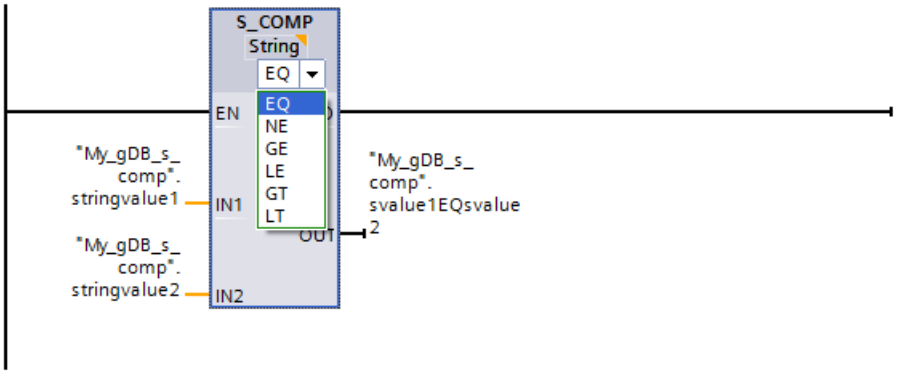
In the following example, you use the "Equal" comparison option to compare two character strings with the "STRING" data type.

Create three tags in a global data block for storing the data.

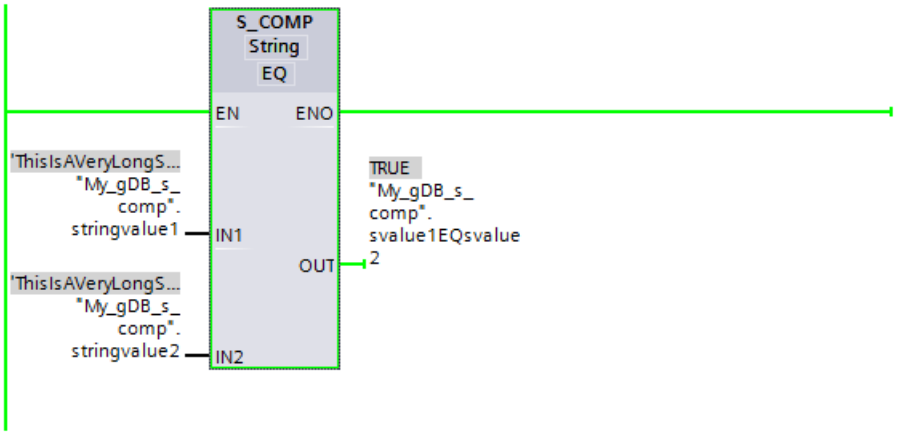
My_gDB_s_comp

	Name	Data type	Start value
1	Static		
2	stringvalue1	String	'ThisIsAVeryLongStringWithSomeNumbers646'
3	stringvalue2	String	'ThisIsAVeryLongStringWithSomeNumbers646'
4	svalue1EQsvalue2	Bool	false

Interconnect the parameters of the instruction as follows. Select the STRING data type and the EQ comparison option.



Since the value of the first character string to be compared ("stringvalue1") is equal to the second character string ("stringvalue2"), the comparison result ("svalue1EQsvalue2") shows the signal state "TRUE".



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

S_CONV: Convert character string (S7-1200, S7-1500)

Description

You use this instruction to convert the value at the IN input to the data format you have specified in the OUT output. You decide the output format of the conversion by selecting a data type for the OUT output parameter.

The following conversions are possible:

- Conversion of a string
- Conversion of a numerical value or character to a string
- Conversion of a character to a character

Note

Explicit conversions

You can find more information on explicit conversions here:

- For an S7-1200-CPU: Explicit conversions (Page 2487)
- For an S7-1500-CPU: Explicit conversions (Page 2382)

Conversion of a string

- To a numerical value (integer or floating point):
  The conversion is performed for all characters of the character string specified in the IN input parameter. Permitted characters are the digits "0" to "9", the decimal point, and the plus and minus signs. The first character of the string may be a valid number or a sign. Leading spaces and exponential notations are ignored.
- To a character:
  When a character string is converted to a character, the first character of the character string is transferred to parameter OUT.
- To a string

Conversion of a numerical value or character to a string

- You decide the format of the numeric value to be converted by selecting a data type for the IN input. Specify a valid tag of the OUT data type at the (W)STRING output. The length of the character string after conversion depends on the value at the IN input.
- The conversion result is saved as a string starting at the third byte. The first byte of the string records the maximum length and the second byte the actual length of the character string. Positive numeric value are output without a sign.
- If the numeric value 0, which exists as an INT or UINT data type, is converted to a string (e.g. INT_TO_STRING(0)), the string in the result has a length of 6 characters.

- When a numeric value is converted to a string, the first characters of the string are padded with spaces. The number of spaces depends on the length of the numerical value.

  Note

  STL/SCL-specific information

  When a numerical value is converted to a string, the first characters of the string are **not** padded with spaces.

  If you want the first character of the string to be padded with spaces, use the "VAL_STRG" instruction.

- When a (W)CHAR character is converted, the character is written at the first position of the character string.

Note

Exponential notation during conversion from floating-point numbers

Do not use exponential notation ("e" or "E") for the conversion from floating-point numbers with the instruction "S_CONV". Instead, use the "STRG_VAL" instruction for the conversion of floating-point numbers with exponential notation. Select the input format of the exponential notation via the FORMAT parameter of the instruction.

Parameters

The following tables show the parameters of the "S_CONV" instruction, according to the possible conversions:

Table 11-22 Parameters for converting a character string to a numeric value:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	STRING, WSTRING	D, L or constant	Value to be converted
OUT	Output	CHAR, WCHAR, USINT, UINT, UDINT, ULINT, SINT, INT, DINT, LINT, REAL, LREAL	I, Q, M, D, L	Result of the conversion

Table 11-23 Parameters for converting a character string to a character string:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	STRING, WSTRING	D, L or constant	Value to be converted
OUT	Output	STRING, WSTRING	D, L	Result of the conversion (possible conversions: STRING to WSTRING and vice-versa)

Table 11-24 Parameters for converting a numerical value or character to a character string:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	CHAR, WCHAR, USINT, UINT, UDINT, ULINT, SINT, INT, DINT, LINT, REAL, LREAL	I, Q, M, D, L or constant	Value to be converted
OUT	Output	STRING, WSTRING	D, L	Result of the conversion

Table 11-25 Parameters for converting a character to a character:

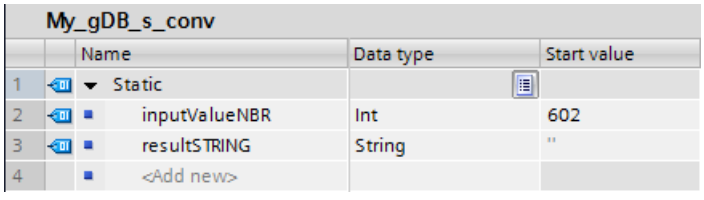
Parameter	Declaration	Data type	Memory area	Description
IN	Input	CHAR, WCHAR	I, Q, M, D, L or constant	Value to be converted
OUT	Output	CHAR, WCHAR	I, Q, M, D, L	Result of the conversion (possible conversions: CHAR to WCHAR and vice-versa)

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

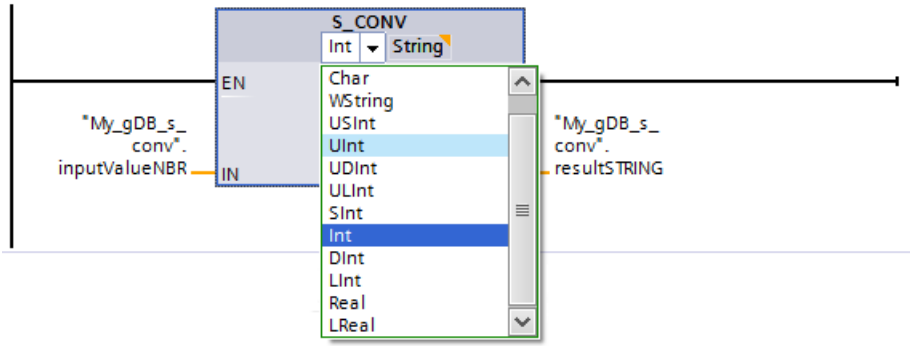
Example

In the following example, convert a number with the INT data type to a character string with the STRING data type.

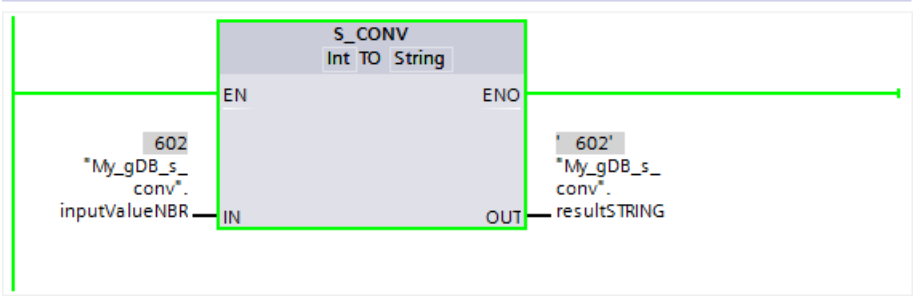
Create two tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows. In doing so, select the data types. With the first selection option, you specify the data type of the value to be converted ("inputValueNBR"). With the second selection option, you specify the data type of the character string to be generated ("resultSTRING").



The value to be converted ("inputValueNBR") is converted to the output format. Blank spaces are written to the empty places at the start of the character string. The result of the conversion is output at output parameter OUT ("resultSTRING") as a string.



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

See also

STRG_VAL: Convert character string to numerical value (Page 3497)

STRG_VAL: Convert character string to numerical value (S7-1200, S7-1500)

Description

The "STRG_VAL" instruction converts a character string to an integer or a floating-point number:

- You specify the character string to be converted in the IN input parameter.
- You define the format of the output value by selecting a data type for the OUT output parameter.

Permitted characters for the conversion are the digits "0" to "9", the decimal point, the decimal comma, notations "E" and "e", and the plus and minus characters. The conversion is canceled if a invalid character is found.

Parameters

The following table shows the parameters of the "STRG_VAL" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	STRING, WSTRING	STRING, WSTRING	D, L or constant	Number string to be converted
FORMAT	Input	WORD	WORD	I, Q, M, D, L, P or constant	Input format of the characters
P	Input	UINT	UINT	I, Q, M, D, L, P or constant	Reference to the first character to be converted (first character = 1, the value "0" or a value > length of the string is invalid)
OUT	Output	USINT, SINT, UINT, INT, UDINT, DINT, REAL, LREAL	USINT, SINT, UINT, INT, UDINT, DINT, ULINT, LINT, REAL, LREAL	I, Q, M, D, L, P	Result of the conversion

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter FORMAT

You use the FORMAT parameter to specify how the characters of a character string are to be interpreted. Exponential values are also converted and represented with the "STRG_VAL" instruction.

The following table shows the possible values of the FORMAT parameter and their meaning:

Value (W#16#....)	Notation	Decimal representation
0000	Decimal fraction	"."
0001		","
0002	Exponential	"."
0003		","
0004 to FFFF	Invalid values	

Parameter P

The conversion starts at the character whose position you specified in the P parameter. If, for example, the value "1" is specified in the P parameter, the conversion starts at the first character of the specified character string.

Example

The following table shows examples of the conversion of a character string to a numeric value:

IN (STRING)	FORMAT (W#16#....)	OUT (data type)	OUT (value)	ENO status
'123'	0000	INT/DINT	123	1
'-00456'	0000	INT/DINT	-456	1
'123.45'	0000	INT/DINT	123	1
'+2345'	0000	INT/DINT	2345	1
'00123AB'	0000	INT/DINT	123	1
'123'	0000	REAL	123.0	1
'-00456'	0001	REAL	-456.0	1
'+00456'	0001	REAL	456.0	1
'123.45'	0000	REAL	123.45	1
'123.45'	0001	REAL	12345.0	1
'123,45'	0000	REAL	12345.0	1
'123,45'	0001	REAL	123.45	1
'.00123AB'	0001	REAL	123.0	1
'1.23e-4'	0000	REAL	1.23	1
'1.23E-4'	0000	REAL	1.23	1
'1.23E-4'	0002	REAL	1.23E-4	1
'12,345.67'	0000	REAL	12345.67	1
'12,345.67'	0001	REAL	12.345	1
'3.4e39'	0002	REAL	W#16#7F800000	0
'-3.4e39'	0002	REAL	W#16#FF800000	0
'1.1754943e-38'	0002	REAL	0.0	1
'12345'	-/-	SINT	0	0
'A123'	-/-	-/-	0	0
''	-/-	-/-	0	0
'++123'	-/-	-/-	0	0
'+-123'	-/-	-/-	0	0

Example

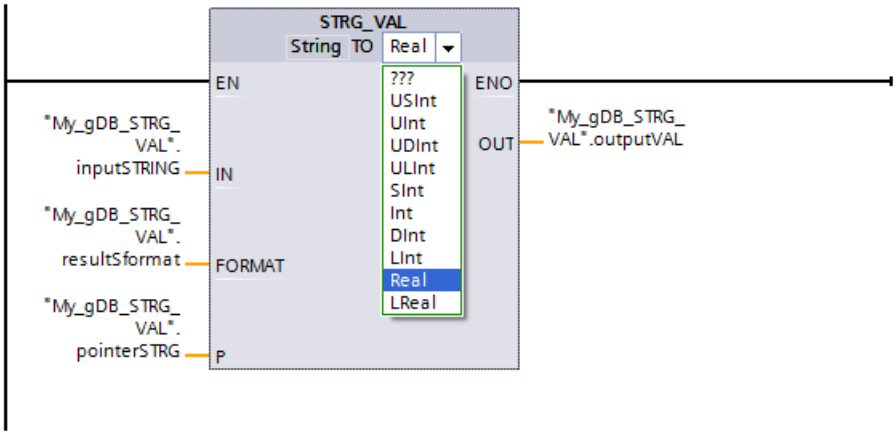
In the following example, you convert a number string with the STRING data type to a floating-point number with the REAL data type. The result will have a length of 32 bits and may have a sign due to the REAL data type.

Create four tags in a global data block for storing the data.

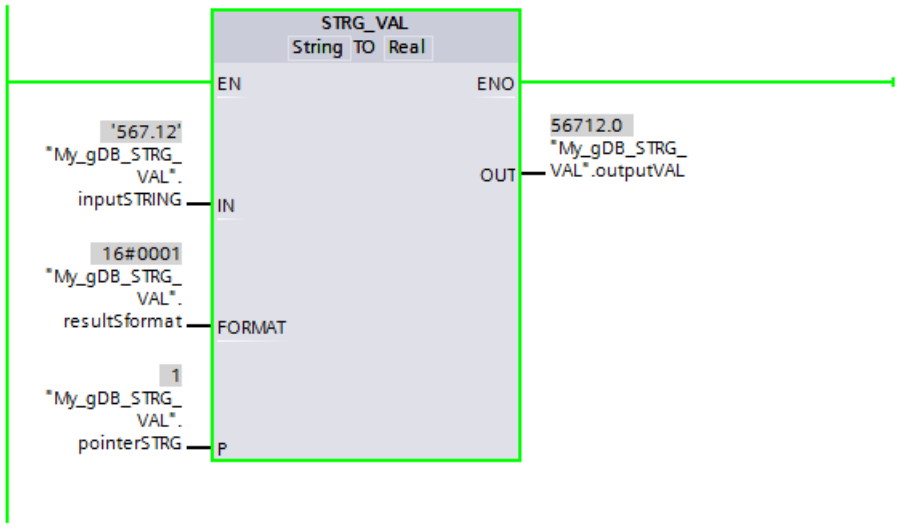
My_gDB_STRG_VAL

	Name	Data type	Start value
1	Static		
2	inputSTRING	String	'567.12'
3	resultSformat	Word	16#0001
4	pointerSTRG	UInt	1
5	outputVAL	Real	0.0

Interconnect the parameters of the instruction as follows. Using the left selection option, select the data type for the character string. Using the right selection option, select the data type for the floating-point number.



The number string is converted starting from the first character according to the value "1" of parameter P ("pointerSTRG"). The period in the number string is interpreted as a thousands delimiter due to the value "0001" of parameter FORMAT ("resultSformat"). (The comma is the decimal separator for the value "0001".) The value to be converted ("inputSTRING") is output at output parameter OUT ("outputVAL") as a floating point number.



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

VAL_STRG: Convert numerical value to character string (S7-1200, S7-1500)

Description

The "VAL_STRG" instruction is used to convert a numerical value into a character string.

- You specify the value to be converted in the IN input parameter. You decide the format of the numeric value by selecting a data type.
- You query the result of the conversion in the OUT output parameter.

Permitted characters for the conversion are the digits "0" to "9", the decimal point, the decimal comma, notations "E" and "e", and the plus and minus characters. The conversion is interrupted by invalid characters.

Parameter

The following table shows the parameters of the "VAL_STRG" instruction:

Parameter	Declaration	Data type		Memory area	Description
		S7-1200	S7-1500		
IN	Input	USINT, SINT, UINT, INT, UDINT, DINT, REAL, LREAL	USINT, SINT, UINT, INT, UDINT, DINT, ULINT, LINT, REAL, LREAL	I, Q, M, D, L, P or constant	Value to be converted
SIZE	Input	USINT	USINT	I, Q, M, D, L, P or constant	Number of character positions
PREC	Input	USINT	USINT	I, Q, M, D, L, P or constant	Number of decimal places
FORMAT	Input	WORD	WORD	I, Q, M, D, L, P or constant	Output format of the characters
P	InOut	UINT	UINT	I, Q, M, D, L, P or constant	Character starting at which the result is written.
OUT	Output	STRING, WSTRING	STRING, WSTRING	D, L	Result of the conversion

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter P

With the P parameter, you specify the character in the string starting at which the result is written. If, for example, the value "2" is specified in the P parameter, the converted value is saved starting at the second character of the string.

SIZE and P parameters

Use the SIZE parameter to specify how many characters of the character string will be written. This is counted starting from the character specified in the P parameter. If the output value is shorter than the specified length, the result is written to the character string right-justified. The empty character positions are filled with blanks.

Parameter FORMAT

Use the FORMAT parameter to specify how the numerical value is interpreted during conversion and written to the character string. You can specify only tags of the FORMAT data type in the USINT parameter.

The following table shows the possible values of the FORMAT parameter and their meaning:

Value (W#16#....)	Notation	Sign	Decimal representation
0000	Decimal fraction	"-"	"."
0001			","
0002	Exponential		"."
0003			","
0004	Decimal fraction	"+" and "-"	"."
0005			","
0006	Exponential		"."
0007			","
0008 to FFFF	Invalid values		

Note

Relevance of values 2, 3, 6 and 7 of the FORMAT parameter

The values 2, 3, 6 and 7 of the FORMAT parameter are only relevant if the parameter IN has data type REAL or LREAL.

Parameter PREC

Use the PREC parameter to define the number of decimal places when converting floating-point numbers. A maximum precision of seven numbers is supported for numerical values of the REAL data type. If the value to be converted is an integer, you use the PREC parameter to specify the position where the decimal point will be placed.

Example

The following table shows examples of the conversion of numeric values to a character string if the parameters OUT contains an empty string before the call of VAL_STRG.

IN (value)	IN (data type)	P	SIZE	FORMAT (W#16#....)	PREC	OUT (STRING)	ENO status
123	UINT	16	10	0000	0	`xxxxxxx123 C`	1
0	UINT	16	10	0000	2	`xxxxxx0.00 C`	1

IN (value)	IN (data type)	P	SIZE	FORMAT (W#16#....)	PREC	OUT (STRING)	ENO status
12345678	UDINT	16	10	0000	3	x12345.678 C	1
12345678	UDINT	16	10	0001	3	x12345.678 C	1
123	INT	16	10	0004	0	xxxxxx+123 C	1
-123	INT	16	10	0004	0	xxxxxx-123 C	1
-0.00123	REAL	16	10	0004	4	xxx-0.0012 C	1
-0.00123	REAL	16	10	0006	4	-1.2300E-3 C	1
-Inf [1]	REAL	16	10	-/-	4	xxxxxx-INF C	0
+Inf [2]	REAL	16	10	-/-	4	xxxxxx+INF C	0
NaN [3]	REAL	16	10	-/-	4	xxxxxxxNaN C	0
12345678	UDINT	16	6	-/-	3	xxxxxxxxxx C	0
"x" represents blanks [1]-Inf: Floating-point number representing a negative infinite value. [2]+Inf: Floating-point number representing a positive infinite value. [3]NaN: Value returned as the result of invalid math operations.							

Example

The following table shows examples of the conversion of numeric values to a character string when the parameter OUT contains an empty string prior to calling VAL_STRG.

In all cases, the parameters IN, SIZE, PREC, FORMAT and P have the same values: IN=123, SIZE=16, PREC=0, FORMAT=4, P=10. Only the default of OUT is different.

- OUT contains a string that is shorter than P.
  The CPU fills OUT after the existing string with spaces up to the (P-1)th position. As of position P OUT It is written taking into account leading spaces.

OUT before the call of VAL_STRG:	ABCDEFxxxxxxxxxx xxxxxxxxxxxxxxx
OUT after the call of VAL_STRG:	ABCDEFxxxxxxxxxx xxxxxx123xxxxxxx
"x" represents a space	

- OUT contains a string that is longer than P+SIZE
  As of position P the CPU fills OUT with SIZE characters taking into account the leading spaces.

OUT before the call of VAL_STRG:	ABCDEFGHIJKLMNOP QRSTUVWXYZABCDEF
OUT after the call of VAL_STRG:	ABCDEFGHIxxxxxxx xxxxxx123ZABCDEF
"x" represents a space	

- OUT contains a string that is longer than P but shorter than P+SIZE
  As of position P the CPU fills OUT with SIZE characters taking into account the leading spaces.

OUT before the call of VAL_STRG:	ABCDEFGHIJKLMNOP xxxxxxxxxxxxxxx
OUT after the call of VAL_STRG:	ABCDEFGHIxxxxxxx xxxxxx123xxxxxxx
"x" represents a space	

Example

In the following example, you convert a floating-point number with the REAL data type to a character string with the STRING data type.

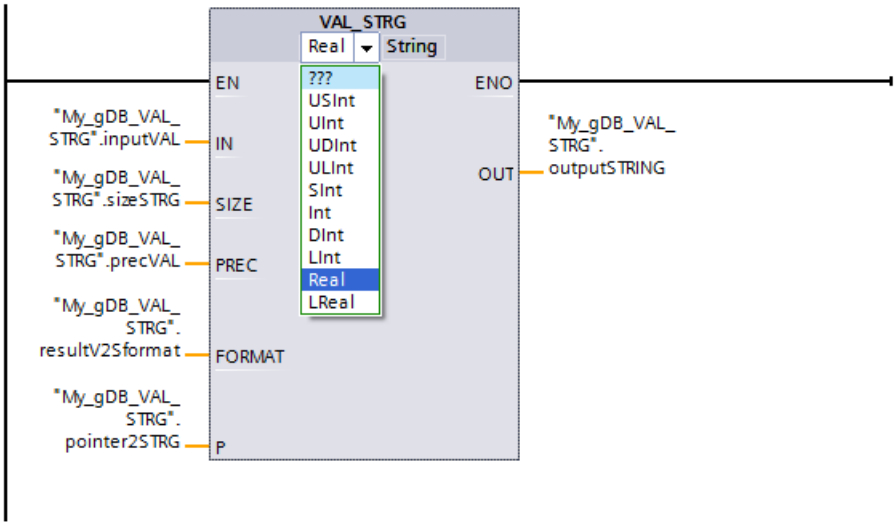
Create six tags in a global data block for storing the data.

My_gDB_VAL_STRG

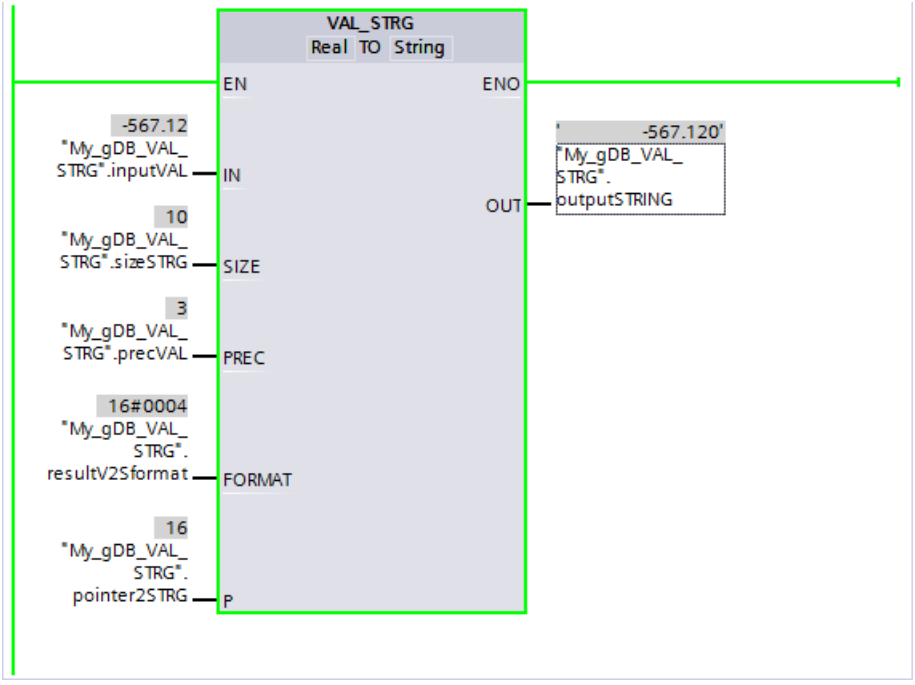
	Name	Data type	Start value
1	Static		
2	inputVAL	Real	-567.12
3	pointer2STRG	UInt	16
4	sizeSTRG	USInt	10
5	precVAL	USInt	3
6	outputSTRING	String	''
7	resultV2Sformat	Word	16#0004

Interconnect the parameters of the instruction as follows. Select the data types:

- The left selection option is for the value to be converted.
- The right selection option is for the character string to be generated.



The character string is written starting from the 16th character according to the value "16" of parameter P" ("pointer2STRG") . Starting from this point, the character string is 10 characters long, according to the "10" value of the SIZE parameter ("sizeSTRG"). The point of the value to be converted ("inputVAL") is interpreted as a decimal separator due to the value "0004" of parameter FORMAT ("resultV2Sformat"). Three decimal places are written to the string according to the value "3" of parameter PREC ("precVAL"). The sign of the value to be converted is stored as a character in the string and prefixes the numbers. The remaining 10 characters of the string are written as spaces in front of the sign. The character string is output at output parameter OUT ("outputSTRING").



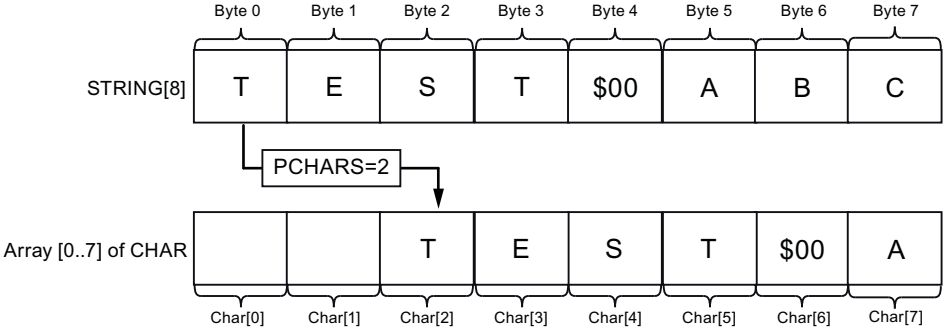
You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

Strg_TO_Chars: Convert character string to Array of CHAR (S7-1200, S7-1500)

Description

With the "Strg_TO_Chars" instruction, you copy a character string of the data type STRING to an Array of CHAR or Array of BYTE or a character string of the data type WSTRING to an Array of WCHAR or Array of WORD. Only ASCII characters are valid for the copy process.

- Specify the character string at the input parameter STRG.
- The characters are written at the parameter CHARS to a Array of CHAR / BYTE / WCHAR / WORD data type.
  - The number of characters in the destination field must be at least the same as the number copied from the source string.
  - If the destination field contains fewer characters than the source string, the characters are written up to the maximum length of the destination field.
  - If the character string contains a "$00" or W#16#0000 character, this has no effect on the copy operation (see graphic).
  - The number of copied characters is output at parameter CNT.
- Use the parameter PCHARS to indicate from which position writing will begin in the destination field.
  - Example: If writing should take place starting from the third position, use the value "2" at parameter PCHARS:



  - The default for PCHARS is "0". When PCHARS = 0, the low index limit of the array is used (e.g. CHAR[0] for an Array [0..5] of CHAR). This also applies if the low limit of the array is negative (e.g. CHAR[-5] for an Array [-5..5] of CHAR).

Note

Use of the instruction with S7-1200 V2.0

S7-1200 up to Version 2.0 only supports Array [0 .. n] of CHAR / BYTE. Negative index limits (e.g. Array [-3..2] of CHAR) are not permissible. This restriction is not checked by the software.

Parameters

The following table shows the parameters of the "Strg_TO_Chars" instruction:

Parameter	Declaration	Data type	Memory area	Description
STRG	Input	STRING, WSTRING	D, L or constant	Source of the copy operation
PCHARS	Input	DINT	I, Q, M, D, L, P or constant	Position in the structure Array of (W)CHAR / BYTE / WORD starting from which the characters of the character string are written.
CHARS	InOut	VARIANT	D, L	Destination of the copy operation Copy the characters to a structure of the Array of (W)CHAR/ BYTE/ WORD data type.
CNT	Output	UINT	I, Q, M, D, L, P	Number of moved characters

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

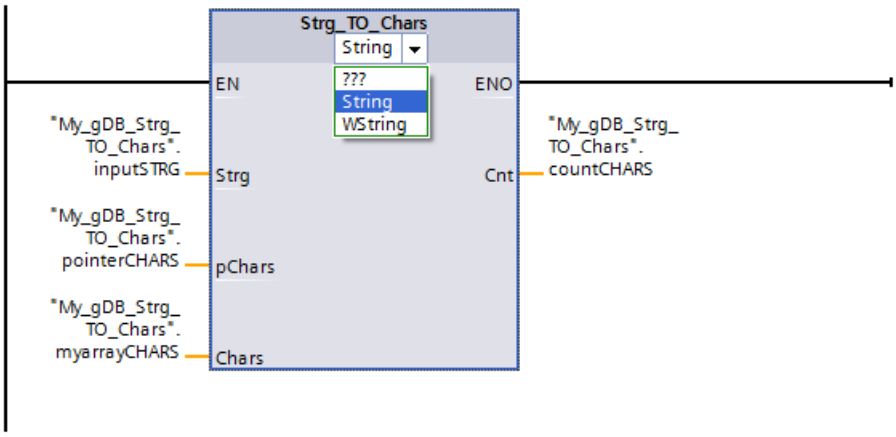
In the following example, you copy characters of a character string with the STRING data type to a structure of the Array of CHAR data type.

Create four tags in a global data block for storing the data.

My_gDB_Strg_TO_Chars

	Name	Data type	Start value
1	Static		
2	inputSTRG	String[10]	'cf7a7a#'
3	pointerCHARS	DInt	2
4	myarrayCHARS	Array[0..9] of Char	
5	myarrayCHARS[0]	Char	' '
6	myarrayCHARS[1]	Char	' '
7	myarrayCHARS[2]	Char	' '
8	myarrayCHARS[3]	Char	' '
9	myarrayCHARS[4]	Char	' '
10	myarrayCHARS[5]	Char	' '
11	myarrayCHARS[6]	Char	' '
12	myarrayCHARS[7]	Char	' '
13	myarrayCHARS[8]	Char	' '
14	myarrayCHARS[9]	Char	' '
15	countCHARS	UInt	0

Interconnect the parameters of the instruction as follows and select the data type of the character string.



A structure made up of individual characters is created due to the Array of CHAR data type. The CHARS structure ("myarrayCHARS") is ten characters long (Array ... [0..9]). According to the value "2" of parameter PCHARS ("pointerCHARS"), writing begins starting from the third character in the structure ("0" and "1" are empty, "2" contains the first character of the character string ("inputSTRG")). Once the characters of the string ("inputSTRG") are written to the structure ("myarrayCHARS"), the last character of the structure to be created is written empty. The number of characters of the character string that are moved is output at output parameter CNT ("countCHARS").

My_gDB_Strg_TO_Chars

	Name	Data type	Start value	Monitor value
1	Static			
2	inputSTRG	String[10]	'cf7a7a#'	'cf7a7a#'
3	pointerCHARS	DInt	2	2
4	myarrayCHARS	Array[0..9] of Char		
5	myarrayCHARS[0]	Char	' '	' '
6	myarrayCHARS[1]	Char	' '	' '
7	myarrayCHARS[2]	Char	' '	'c'
8	myarrayCHARS[3]	Char	' '	'f'
9	myarrayCHARS[4]	Char	' '	'7'
10	myarrayCHARS[5]	Char	' '	'a'
11	myarrayCHARS[6]	Char	' '	'7'
12	myarrayCHARS[7]	Char	' '	'a'
13	myarrayCHARS[8]	Char	' '	'#'
14	myarrayCHARS[9]	Char	' '	' '
15	countCHARS	UInt	0	7

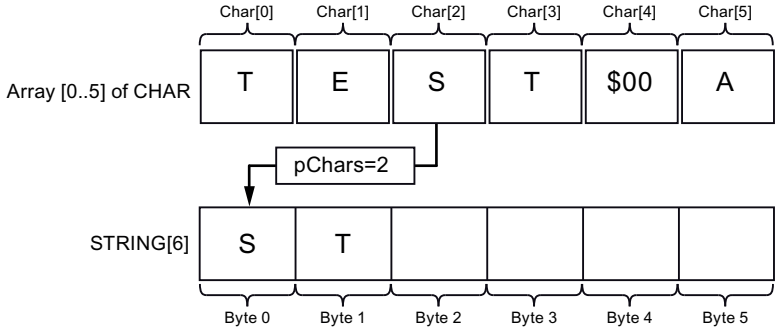
You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

Chars_TO_Strg: Convert Array of CHAR to character string (S7-1200, S7-1500)

Description

With the "Chars_TO_Strg" instruction you copy characters from an Array of CHAR or Array of BYTE to a character string of the data type STRING or from an ARRAY of WCHAR or Array of WORD to a character string of the data type WSTRING. For the copy operation only ASCII characters are valid.

- Specify the characters of the Array of (W)CHAR / BYTE / WORD that will be copied to a character string at the input parameter CHARS.
- The characters are written at the parameter STRG to a (W)STRING data type.
  - The number of characters in the string is at least the same number as were copied from the source field.
  - If the character string is shorter than the number of characters in the source field, the characters are written up to the maximum length of the string.
  - If the Array of CHAR / BYTE contains a "$00" character or if the Array of WCHAR / WORD contains a W#16#0000 character, the copy operation is only carried out up to the corresponding position (see graphic).
- Using the PCHARS parameter, specify the position of the source field as of which the characters will be copied. PCHARS = 0 is the default value and always specifies the lower index limit of the array even if this is negative.
  - Example: if copying is to start with the third character of the source field, use the value "2" at parameter PCHARS:



  - If an index is specified at parameter PCHARS and the index is not contained in the copy source (e.g. "7" at Array [0..5] of CHAR) then the instruction is not executed.

Note

Use of the instruction with S7-1200 V2.0

S7-1200 up to Version 2.0 only supports Array [0 .. n] of CHAR / BYTE. Negative index limits (e.g. Array [-3..2] of CHAR) are not permissible. This restriction is not checked by the software.

Parameters

The following table shows the parameters of the "Chars_TO_Strg" instruction:

Parameter	Declaration	Data type	Memory area	Description
CHARS	Input	VARIANT	D, L	Source of the copy operation Array of (W)CHAR / BYTE / WORD from which the characters are copied.
PCHARS	Input	DINT	I, Q, M, D, L, P or constant	Position in the Array of (W)CHAR / BYTE / WORD starting at which the characters are copied.
CNT	Input	UINT	I, Q, M, D, L, P or constant	Number of characters to be copied. Use "0" to copy all characters.
STRG	Output	STRING, WSTRING	D, L	Destination of the copy operation Character string with the (W)STRING data type. Observe the maximum length of the data types: • STRING: 254 characters • WSTRING: 254 characters (default) / 16382 characters (maximum) When using WSTRING, note that you must define a length > 254 characters explicitly with square brackets (e.g., WSTRING[16382]).

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

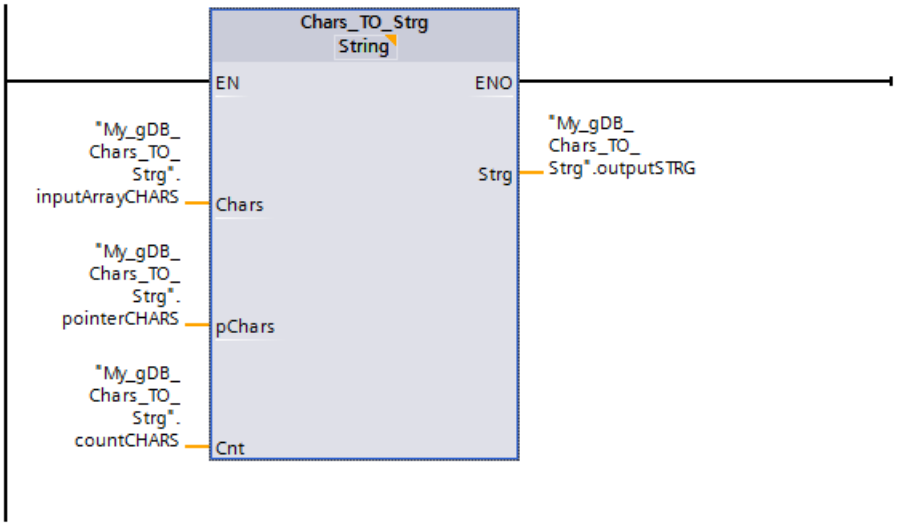
In the following example, you copy characters of a structure of the Array of CHAR data type to a character string with the STRING data type.

Create four tags in a global data block for storing the data.

My_gDB_Chars_TO_Strg

	Name	Data type	Start value
1	Static		
2	inputArrayCHARS	Array[0..9] of Char	
3	inputArrayCHARS[0]	Char	'M'
4	inputArrayCHARS[1]	Char	'y'
5	inputArrayCHARS[2]	Char	'S'
6	inputArrayCHARS[3]	Char	'7'
7	inputArrayCHARS[4]	Char	'P'
8	inputArrayCHARS[5]	Char	'L'
9	inputArrayCHARS[6]	Char	'C'
10	inputArrayCHARS[7]	Char	' '
11	inputArrayCHARS[8]	Char	' '
12	inputArrayCHARS[9]	Char	' '
13	pointerCHARS	DInt	2
14	outputSTRG	String	''
15	countCHARS	UInt	0

Interconnect the parameters of the instruction as follows and select the data type of the character string.



The CHARS structure ("inputArrayCHARS") is ten characters long (Array ... [0..9]). According to the value "2" of parameter PCHARS ("pointerCHARS"), characters are copied starting from the third position of the structure to the character string ("outputSTRG"). Starting from position "2", all characters of the structure ("inputArrayCHARS") are copied to the character string ("outputSTRG"), since parameter CNT ("countCHARS") has the value "0".

My_gDB_Chars_TO_Strg

	Name	Data type	Start value	Monitor value
1	Static			
2	inputArrayCHARS	Array[0..9] of Char		
3	inputArrayCHARS[0]	Char	'M'	'M'
4	inputArrayCHARS[1]	Char	'y'	'y'
5	inputArrayCHARS[2]	Char	'S'	'S'
6	inputArrayCHARS[3]	Char	'7'	'7'
7	inputArrayCHARS[4]	Char	'P'	'P'
8	inputArrayCHARS[5]	Char	'L'	'L'
9	inputArrayCHARS[6]	Char	'C'	'C'
10	inputArrayCHARS[7]	Char	' '	' '
11	inputArrayCHARS[8]	Char	' '	' '
12	inputArrayCHARS[9]	Char	' '	' '
13	pointerCHARS	DInt	2	2
14	outputSTRG	String	''	'S7PLC '
15	countCHARS	UInt	0	0

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

MAX_LEN: Determine the length of a character string (S7-1200, S7-1500)

Description

A tag of the (W)STRING data type contains two lengths: the maximum length and the current length (this is the number of currently valid characters).

- The maximum length of the character string is specified for each tag in the STRING keyword in square brackets. The number of bytes occupied by a string is 2 greater than the maximum length.
- The maximum length of the character string is specified for each tag in the WSTRING keyword in square brackets. The number of words occupied by a character string is 2 greater than the maximum length.
- The current length represents the number of the character places actually used. The current length must be less than or equal to the maximum length.

You use the "MAX_LEN" instruction to determine the maximum length of the character string specified at the IN input parameter This is output as numerical value at the output par OUT.

Note

Checking the character string

The "MAX_LEN" instruction does not check the character string at the parameter "IN".

If errors occur during processing of the instruction, the value "0" is output at the OUT parameter.

Note

Reading out the current length

You use the "LEN (Page 3531)" instruction to read out the current length of a string.

Parameter

The following table shows the parameters of the "MAX_LEN" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	• STRING • WSTRING	D, L or constant	Character string
OUT	Return	• INT • DINT	I, Q, M, D, L, P	Maximum number of characters

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

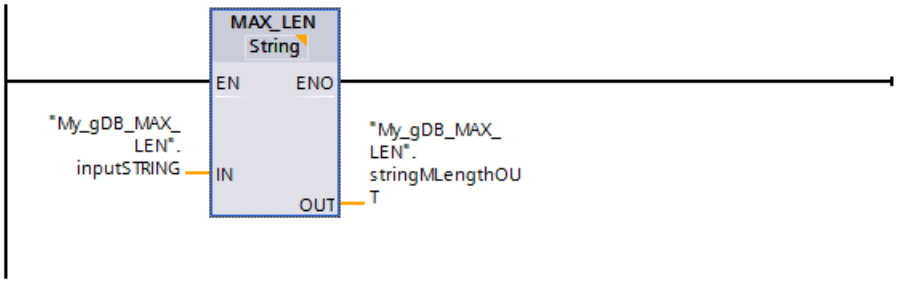
Example

In the following example, you determine the maximum length of a character string with the STRING data type.

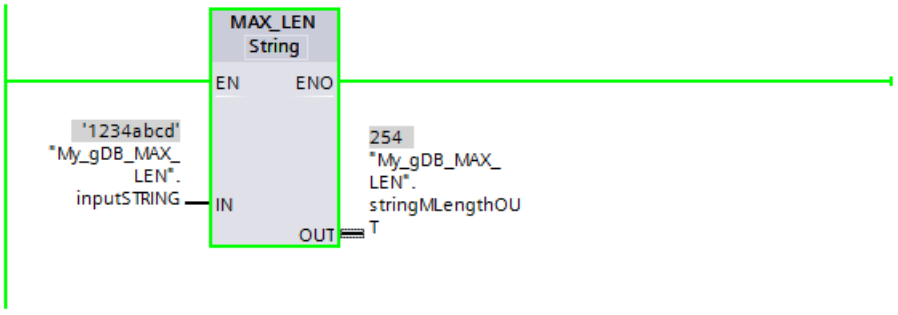
Create two tags in a global data block for storing the data.

My_gDB_MAX_LEN			
	Name	Data type	Start value
1	Static		
2	inputSTRING	String	'1234abcd'
3	stringMLengthOUT	Int	0

Interconnect the parameters of the instruction as follows.



The maximum length of the specified character string ("inputSTRING") is determined and returned at output parameter OUT ("stringMLengthOUT") as a numerical value.



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

JOIN: Join multiple strings (S7-1500)

Description

The "JOIN" instruction connects multiple strings into an array.

To convert multiple strings into a single one, the instruction provides the following functions:

- **Format selection**
  You can use the first bit at the Mode parameter to specify whether the sequence of the source strings for an array should be in the format CSV or FSR.
  In the following example, two source strings are specified by the two columns of the table. The maximum number of characters amounts to 4 characters for the first source string, 13 characters for the second, 10 characters for the third and 14 characters for the fourth.

1963	1974
Miller	Jackson
John	Peter
Roadname	VeryLongRoadna

  – With CSV (Comma Separated Values), the contents of the source strings are written consecutively into the destination array and separated with a delimiter. (See examples below.)
  – With FSR (Fixed Size Records), a certain number of characters is defined in the destination array for each source string. If the characters of a source string do not need the space reserved for them in the destination array, the corresponding array elements are filled with delimiters. However, if the number of characters in a source string is greater than the space reserved for them, the corresponding array elements are filled from the front, and the excess characters of the source string are truncated (see example below).

- **Selection of the delimiter for the source strings**
  The RecSeparator parameter is used to select the delimiter that will be used for each string. Select a character based on the content of the input strings at the SrcStruct parameter. If the input strings contain a comma within a string, for example, do not use the comma as a delimiter. The data type that you use for the delimiter must correspond to the destination array at the DstArray parameter As a result, the delimiters can also be written to the array.

- **Selection of the delimiter for the end of all source strings**
  You can use the third bit of the Mode parameter to specify whether an additional character will be written as a delimiter at the end of the copied characters in the destination array (DstArray parameter). At the EndSeparator parameter, specify the character that will be used as a delimiter. Make sure to use a character other than the one set for the RecSeparator parameter (delimiter for individual strings). If the two delimiters are indistinguishable, you will obtain unexpected results if you reverse the conversion using the "SPLIT" instruction.

- **Selection of the source strings**
  You specify the source strings at the SrcStruct parameter. For the data type, you can use Array of STRING or Array of WSTRING or structures which only contain the STRING or WSTRING data type. This also applies to user data types or nested structures. You can use these as long as they exclusively contain the data type STRING or WSTRING respectively.

- **Specification of the number of joined strings**
  If you use an Array of STRING or Array of WSTRING at the SrcStruct parameter (source string) (no nested structures), you can use the Count parameter to specify the number of source strings that are joined to form a single string. If you use a data type other than Array of (W)STRING at the SrcStruct parameter, the Count parameter is ignored. It is only possible to connect a larger part of an array in this way.

- **Selection of a destination area for writing the array**
  You use the Array of (W)CHAR data type at the DstArray parameter. In this case, it is not possible to use the STRING or WSTRING data types, because the length is limited to 254 characters or 256 bytes for STRING.
- **Index of the position in the array (DestArray target parameter)**
  The conversion starts at this position. The instruction reads the Position parameter to determine the position at which the conversion is completed. This enables follow-up calls of the instruction for filling the array.

Parameter

The following table shows the parameters of the "JOIN" instruction:

Parameter	Declaration	Data type	Memory area	Description
Mode	Input	DWORD	I, Q, M, D, L or constant	Specifies how the merger to string is performed (see Mode" parameter).
RecSeparator	Input	VARIANT	I, Q, M, D, L	Delimiter for the source strings • With CSV: Character that is used as a delimiter for the individual strings. • With FSR: Character that is used as a fill character for the individual strings.
EndSeparator	Input	VARIANT	I, Q, M, D, L	Delimiter for the end of the conversion Delimiter that is written at the end of the characters if bit 3 = 1 has been set for the Mode parameter.
SrcStruct	Input	VARIANT	I, Q, M, D, L	Pointer to the source strings.
Count	Input	UDINT	I, Q, M, D, L or constant	Number of strings that were joined. The Count parameter can only be used when an Array of (W)STRING has been used at the SrcStruct parameter.
DestArray	InOut	VARIANT	I, Q, M, D, L	Area in which the characters are written after the conversion. Use the Array [0 .. x] of CHAR/WCHAR data type at the DestArray parameter. You can set the length (x) of the array based on the length of the source strings at the SrcStruct parameter.
Position	InOut	UDINT	I, Q, M, D, L	Index the position in the total string
Ret_Val	Return	INT	I, Q, M, D, L	Status of instruction (see table "RET_VAL parameter").

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter Mode

Bit	Bit value "0"	Bit value "1"	Description
0	CSV format (Comma Separated Values)	FSR format (Fixed Size Record)	Selection of the format: • With CSV, the source strings are separated by a delimiter in the destination array. • With FSR, the source strings are written into the destination array with the fill characters defined at the RecSeparator parameter.
1	-	-	Irrelevant for the "JOIN" instruction.
2	-	-	Reserved (irrelevant for bit value)
3	Write no additional delimiters.	Write character defined by the EndSeparator parameter at the end of the read characters.	Select whether an additional character will be written as a delimiter at the end of the characters in the array (DestArray parameter).
4	-	-	Irrelevant for the "JOIN" instruction.

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error.
8190	Selection at the Mode parameter is not supported.
8x20	Source strings invalid.
8x53	VARIANT points to data structure that is too short.
8x54	Invalid data type
8082	The value at parameter Count is greater than the number of source character strings in SrcStruct.
8xB4	Different data types at parameters SrcStruct (source) and DestArray (destination) or at separators (parameters RecSeparator and EndSeparator).
80B5	Buffer overflow with the instruction. The characters are not completely output at parameter DestArray, or the value at parameter Position is outside of DestArray.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)

* Note the following for the error codes:

- The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".
- The "x" in the second position of the listed error codes stands for the parameter that caused the error.
  Example: Error code 8352 hex = error occurred at 3rd Parameter (EndSeparator), see Parameter table.
- If the error cannot be clearly assigned to a specific parameter, "0" is output.
  Example: The CHAR data type is used for the delimiter (RecSeparator parameter). WCHAR is used as data type for the array at the DestArray parameter. Error code 80B4 is output in this case.

Two examples for the JOIN instruction when the destination ARRAY should have the format CSV

- First example:
  The following source strings are available:
  - `1963`
  - `Miller`
  - `John`
  - `Roadname`

  If you select "," as the delimiter, calling JOIN provides the following destination array:
  `1963,Miller,John,Roadname`
- Second example:
  The following source strings are available:
  - `1974`
  - `Jackson`
  - `Peter`
  - `VeryLongRoadname`

  If you select "," as the delimiter, calling JOIN provides the following destination array:
  `1974,Jackson,Peter,VeryLongRoadname`

Two examples for the JOIN instruction when the destination ARRAY should have the format FSR

- First example:
  The following source strings are available:
  - `1963`
  - `Miller`
  - `John`
  - `Roadname`

  The number of reserved characters in the destination array amounts to 4 characters for the first source string, 13 characters for the second, 10 characters for the third and 14 characters for the fourth.
  If you select "," as the fill characters, calling JOIN provides the following destination array:
  `1963Miller,,,,,,,John,,,,,,Roadname,,,,,,`
- Second example:
  The following source strings are available:
  - `1974`
  - `Jackson`
  - `Peter`
  - `VeryLongRoadname`

  The number of reserved characters in the destination array amounts to 4 characters for the first source string, 13 characters for the second, 10 characters for the third and 14 characters for the fourth.
  If you select "," as the fill characters, calling JOIN provides the following destination array:
  `1974Jackson,,,,,,Peter,,,,,VeryLongRoadna`

SPLIT: Splitting an array of characters into multiple strings (S7-1500)

Description

The "SPLIT" instruction converts an array (Array of CHAR / WCHAR) into several separate strings (Array of STRING / WSTRING or structure).

To convert the array into several strings, specify the following information:

- **Selection of array to be read**
  You specify the array to be read at the SrcArray parameter. Make sure that the data types used for the input and output parameters match any other parameters used. If you use an array with the CHAR data type at the SrcArray parameter, for example, you must use the CHAR data type for the delimiter (Rec/EndSeparator) and the structure must only contain strings with the STRING data type at the DestStruct parameter.
- **Format selection for the source array**
  You use the first bit of the Mode parameter to specify whether the array to be read has the CSV or FSR format.
  - With CSV (Comma Separated Values), characters in the source array that belong together are separated by a delimiter from the next characters that belong together. Two examples of a source array:
    `1963,Miller,John,CitynameA,Roadname`
    `1974,Jackson,Peter,CitynameB,VeryLongRoadname`
  - With FSR (Fixed Size Records), a certain number of characters is defined for each item of logical information of the source array. Each item of information should fit into the space defined for it. When information does not need the space defined for it, it is filled with delimiters.
    Two examples of source arrays, whereby the length of the information is 4 characters for the first information item (year of birth), 13 characters for the second (last name), 10 characters for the third (first name), 9 characters for the fourth (city) and 16 characters for the fifth (street):
    `1963Miller,,,,,,,John,,,,,,CitynameARoadname,,,,,,,,`
    `1974Jackson,,,,,,Peter,,,,,CitynameBVeryLongRoadname`
- **Delimiter used for the array to be read**
  - If the array to be read has CSV format, use the RecSeparator parameter to specify which delimiter was used.
  - If the array to be read has FSR format, use the RecSeparator parameter to specify which fill character was used.
- **Selection of the delimiter for the end of the total string**
  Specify the delimiter after which the reading of the array should stop at the EndSeparator parameter. The "SPLIT" instruction stops at this position and outputs the string that has been found. Note that the EndSeparator delimiter is evaluated with higher priority than the RecSeparator delimiter. If the delimiter at the EndSeparator parameter was used in a string to be read (between two RecSeparator delimiters), all content after the EndSeparator delimiter is ignored.
- **Specification of the position starting at which the array is read**
  The conversion starts at this position in the array and the instruction reads the Position parameter to determine the position at which the conversion is completed. This enables follow-up calls of the instruction for filling the various strings at the DestStruct parameter.
- **Outputting the number of read strings**
  If you use an Array of STRING at the DestStruct parameter, you can use the Count parameter to output the number of read strings. Only strings with content are counted. If you use a data type other than Array of STRING at the DestStruct parameter, "0" is output at the Count parameter.

Parameters

The following table shows the parameters of the "SPLIT" instruction:

Parameter	Declaration	Data type	Memory area	Description
Mode	Input	DWord	I, Q, M, D, L or constant	Specifies how the split into multiple strings is performed (see "Mode" parameter).
RecSeparator	Input	Variant	I, Q, M, D, L	Delimiter or fill character • With CSV: Character that was used in the array to be read to identify the individual strings. • With FSR: Character that has been used as a fill character in the array to be read.
EndSeparator	Input	Variant	I, Q, M, D, L	Delimiter used to define the end of the total string in the array to be read.
SrcArray	Input	Variant	I, Q, M, D, L	Pointer to the array to be read (Array of CHAR/WCHAR)
DestStruct	InOut	Variant	I, Q, M, D, L	Structure which includes the converted strings (Array of STRING / WSTRING).
Position	InOut	UDInt	I, Q, M, D, L	Position from which the array is to be read at the SrcArray parameter.
Ret_Val	Return	Int	I, Q, M, D, L	Result of instruction execution / error code (see table "Ret_Val parameter")
Count	Output	UDInt	I, Q, M, D, L	Number of strings that were found.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter "Mode"

Bit	Bit value "0"	Bit value "1"	Description
0	CSV format (Comma Separated Values)	FSR format (Fixed Size Record)	Basic mode: Selection of CSV or FSR
1	• With CSV: Additional characters cause an error. • With FSR: Additional fill characters remain in string.	• With CSV: Additional characters are ignored. • With FSR: Additional fill characters are removed.	Use bit 1 to specify how you want to deal with additional characters: • With CSV: – If the bit is set, additional characters that do not fit in the destination string are ignored. Example: The instruction writes to a string with a length of 16 characters (STRING[16] data type). No delimiter is contained in the source after the first 16 characters. If bit 1 is set, the extra characters are ignored and the instruction continues to read the array. – If the bit is not set, the instruction stops in such a case and generates an error message at the Ret_Val parameter. • With FSR: – If the bit is set, the fill characters located to the right of characters containing information are not written to the destination character string when the source array is transferred to the destination character strings (see example). – If the bit is not set, the fill characters located to the right of characters containing information are written to the destination character string when the source array is transferred to the destination character strings (see example).
2	-	-	Reserved for use in future versions.
3	Separator is retained.	Separator is removed.	Specification of whether the delimiter is removed at the end of the total string.
4	Leave unwritten strings (STRING) at length.	Set unwritten strings (STRING) to length "0".	Specification of whether unused strings (STRING) at the DestStruct parameter are set to the length "0".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error.
0001	Additional characters are ignored.
8190	Selection at the Mode parameter is not supported.
8x20	Source strings invalid.
8x53	VARIANT points to data structure that is too short.
8x54	Invalid data type
8x84	Additional characters found.
8xB4	Different data types at parameters SrcArray (source) and DestStruct (destination) or at separators (parameters RecSeparator and EndSeparator).
80B5	Buffer overflow with the instruction. The character strings are not completely output at parameter DestStruct.

Error code* (W#16#...)	Explanation
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)

* Note the following for the error codes:

- The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".
- The "x" in the second position of the listed error codes stands for the parameter that caused the error.
  Example: Error code 8352 hex = error occurred at 3rd parameter (EndSeparator), see Parameter table.
- If the error cannot be clearly assigned to a specific parameter, "0" is output.
  Example: The CHAR data type is used for the delimiter (RecSeparator parameter). WCHAR is used as data type for the array at the DestArray parameter. Error code 80B4 is output in this case.

Two examples for the SPLIT instruction when the ARRAY to be read has the CSV format

In the first step, the source array is split into strings as specified by the delimiter (RecSeparator parameter, e.g. ",").

In the second step, the resulting strings after the split are stored in destination strings. You can use the DestStruct parameter to specify their length.

If the strings produced by the split have more characters than can fit in the destination string, the behavior of SPLIT after this point depends on the Mode parameter. With Mode=2#00, this results in an error: The associated error code is returned and no other destination strings are

filled. With Mode=2#10, the surplus characters are ignored and the next destination string is filled with the characters after the next delimiter.

- First example:
  In this example, there are these two source arrays:
  - `1963,Miller,John,Roadname`
  - `1974,Jackson,Peter,VeryLongRoadname`

  The length of the destination string amounts to 4 characters for the first destination string, 13 characters for the second, 10 characters for the third and 14 characters for the fourth.
  The first call for SPLIT produces the following destination strings with Mode=2#10:
  - `1963`
  - `Miller`
  - `John`
  - `Roadname`

  The second call for SPLIT produces the following destination strings with Mode=2#10:
  - `1974`
  - `Jackson`
  - `Peter`
  - `VeryLongRoadna`
- Second example:
  In this example, the logical information in the source array is not in the order expected. There are these two source arrays:
  - `1963,Miller,Roadname,John`
  - `1974,Jackson,VeryLongRoadname,Peter`

  As in the first example, the length of the destination string amounts to 4 characters for the first destination string, 13 characters for the second, 10 characters for the third and 14 characters for the fourth.
  The first call for SPLIT produces the following destination strings with Mode=2#10:
  - `1963`
  - `Miller`
  - `Roadname`
  - `John`

  The second call for SPLIT produces the following destination strings with Mode=2#10:
  - `1974`
  - `Jackson`
  - `VeryLongRo`
  - `Peter`

Two examples for the SPLIT instruction when the ARRAY to be read has format FSR

The source array is split into strings as you specified in the DestStruct parameter for the length of the destination string.

With regard to the fill characters (RecSeparator parameter e.g. ","), the behavior of SPLIT depends on the Mode parameter: With Mode=2#01, the fill characters are entered in the destination strings; with Mode=2#11, they are not entered.

- First example:
  In this example, there are these two source arrays:
  - `1963Miller,,,,,,,John,,,,,,Roadname,,,,,,`
  - `1974Jackson,,,,,,Peter,,,,,VeryLongRoadname`

  The length of the destination string amounts to 4 characters for the first destination string, 13 characters for the second, 10 characters for the third and 14 characters for the fourth. The first call for SPLIT produces the following destination strings with Mode=2#01:
  - `1963`
  - `Miller,,,,,,,`
  - `John,,,,,,`
  - `Roadname,,,,,,`

  The second call for SPLIT produces the following destination strings with Mode=2#01:
  - `1974`
  - `Jackson,,,,,,`
  - `Peter,,,,,`
  - `VeryLongRoadna`
- Second example:
  In this example, the logical information in the source array is not in the order expected. There are these two source arrays:
  - `1963Miller,,,,,,,Roadname,,,,,,John,,,,,,`
  - `1974Jackson,,,,,,VeryLongRoadnamePeter,,,,,`

  As in the first example, the length of the destination string amounts to 4 characters for the first destination string, 13 characters for the second, 10 characters for the third and 14 characters for the fourth.
  The first call for SPLIT produces the following destination strings with Mode=2#11:
  - `1963`
  - `Miller`
  - `Roadname`
  - `,,,,John`

  Explanation: The SPLIT instruction lays templates over the source array, so to speak. The length of these templates is determined by the length of the destination strings. During transfer to the destination strings, only those commas to the right of characters with information content are interpreted as fill characters. The fill characters are not entered with Mode=2#11. In the first call, the commas are to the right of the "John" fill characters. The commas to the left of "John", however, are not fill characters, because the relevant "John" characters follow before the end of the corresponding destination string is reached. Therefore, the commas to the left of "John" appear in the destination string.
  The second call for SPLIT produces the following destination strings with Mode=2#11:

- 1974
- Jackson
- VeryLongRo
- adnamePeter

See also

JOIN: Join multiple strings (Page 3513)

ATH: Convert ASCII string to hexadecimal number (S7-1200, S7-1500)

Description

You use the instruction "ATH" to convert the ASCII character string specified at the IN input parameter into a hexadecimal number. The result of the conversion is output to the OUT output parameter.

- You can reference the following data types using the pointer at the IN parameter (ASCII): STRING, WSTRING, CHAR, BYTE, Array of CHAR, Array of BYTE, WCHAR, Array of WCHAR, Array of WORD.
- You can reference the following data types using the pointer at the OUT parameter (hexadecimal): Array of CHAR, Array of BYTE, Array of WORD, STRING, BYTE, CHAR, WORD, INT, DWORD, DINT, SINT, USINT, UINT, UDINT. Only with S7-1500: Array of WCHAR, WSTRING, WCHAR, ULINT, LINT, LWORD

You specify the number of ASCII characters to be converted with the N parameter. A maximum of 32767 valid ASCII characters can be converted. Only digits "0" to "9", upper case letters "A" to "F", and lower case letters "a" to "f" can be interpreted. All other characters are converted to zeros.

Since 8 bits are required for the ASCII character and only 4 bits for the hexadecimal digit, the output word length is only half of the input word length. The ASCII characters are converted and positioned in the output in the same order as they are read in. If there is an odd number of ASCII characters, the hexadecimal number is padded with zeros in the nibble to the right of the last converted hexadecimal number.

Parameter

The following table shows the parameters of the "ATH" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	VARIANT	D, L	Pointer to ASCII character string
N	Input	INT	I, Q, M, D, L or constant	Number of ASCII characters to be converted
RET_VAL	Return	WORD	I, Q, M, D, L	Status of the instruction
OUT	Output	VARIANT	I, Q, M, D, L	Hexadecimal number

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code (W#16#....)*	Description
0000	No error
0007	Invalid character. Only the following ASCII characters may be used: Digits "0" to "9", upper case letters "A" to "F", lower case letters "a" to "f".
8101	Invalid pointer at the IN parameter, e.g., because a non-existing data block is referenced.
8182	Input buffer is too small for data in the N parameter.
8120	Invalid format in the IN parameter.
8151	Non-supported data type in the IN parameter.
8401	Invalid pointer at the OUT parameter, e.g., because a non-existing data block is referenced.
8482	Output buffer is too small for data in the N parameter.
8420	Invalid format in the OUT parameter.
8451	Non-supported data type in the OUT parameter.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

ASCII characters and hexadecimal values

The following table shows the ASCII characters and the corresponding hexadecimal values:

ASCII character	ASCII-coded hexadecimal value	Hexadecimal digit
"0"	30	0
"1"	31	1
"2"	32	2
"3"	33	3
"4"	34	4
"5"	35	5
"6"	36	6
"7"	37	7
"8"	38	8
"9"	39	9
"A"	41	A
"B"	42	B
"C"	43	C
"D"	44	D
"E"	45	E
"F"	46	F

Example

The following table shows examples of the conversion of ASCII character strings to hexadecimal numbers:

Data type of IN	IN	N	OUT	ENO status
STRING, WSTRING	'0a23'	4	• WORD#16#0A23 • UINT#10#2595	1
STRING, WSTRING	‘123aFx1a23’	10	• LWORD#16#123AF01A_23000000 • ULINT#10#1 313 626 236 378 939 392	0
STRING, WSTRING	‘a23’	3	WORD#16#A230	1
STRING, WSTRING	‘0a23’	4	• WORD#[16#0A23, 16#0000] • BYTE#[16#0A, 16#23]	1
STRING, WSTRING	‘aFbE’	1	• CHAR#‘A’ • WCHAR#‘A’	1
STRING, WSTRING	‘0a23’	4	• CHAR#[‘0’, ‘A’, ‘2’, ‘3’] • WCHAR#[‘0’, ‘A’, ‘2’, ‘3’]	1
STRING, WSTRING	‘123aFC1a23’	10	• STRING#‘123AFC1A23’ • WSTRING#‘123AFC1A23’	1
ArrayOfCHAR, ArrayOfWCHAR	[‘a’, ‘B’, ‘E’, ‘3’]	4	WORD#16#ABE3	1
ArrayOfBYTE	[16#39, 16#32, 16#35]	3	WORD#16#9250	1
ArrayOfWORD	[16#1234, 16#0032, 16#0031, 16#0000]	3	WORD#16#0210	0
ArrayOfWORD	[16#0031, 16#0032, 16#0033, 16#0034]	4	WORD#16#1234	1
ArrayOfWORD	[16#AB31, 16#32AF, 16#4333, 16#0034]	4	WORD#16#0004	0
ArrayOfWORD	[16#0031, 16#0032, 16#0033, 16#0034]	4	[16#1234, 16#0000, 16#0000, 16#0000]	1
ArrayOfWORD	[16#0034, 16#0035, 16#0036, 16#0037, 16#0041, 16#0042, 16#0043, 16#0044]	8	DWORD#16#4567_ABCD	1

HTA: Convert hexadecimal number to ASCII string (S7-1200, S7-1500)

Description

You use the instruction "HTA" to convert the hexadecimal number specified at the IN input into an ASCII character string. The result of the conversion is stored at the address specified in the OUT parameter.

- You can reference the following data types using the pointer in the IN parameter (hexadecimal): Array of CHAR, Array of BYTE, STRING, BYTE, CHAR, WORD, Array of WORD, INT, DWORD, Array of DWORD, DINT, SINT, USINT, UINT, UDINT. Only with S7-1500: Array of WCHAR, WSTRING, WCHAR, ULINT, LINT, LWORD, Array of LWORD
- You can reference the following data types using the pointer in the OUT parameter (ASCII): STRING, WSTRING, Array of CHAR, Array of WCHAR, Array of BYTE, Array of WORD

You specify the number of hexadecimal bytes to be converted with the N parameter. Since 8 bits are required for the ASCII character and only 4 bits for the hexadecimal digit, the output value is twice as long as the input value. Each nibble of the hexadecimal number is converted to a character while maintaining the original order.

A maximum of 32767 characters are written to the ASCII character string. The result of the conversion is represented by the digits "0" to "9" and upper-case letters "A" to "F".

If the complete result of the conversion cannot be displayed in the OUT parameter, the result is will only be partially written to the parameter.

Parameter

The following table shows the parameters of the "HTA" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	VARIANT	I, Q, M, D, L	Start address of the hexadecimal digits
N	Input	UINT	I, Q, M, D, L or constant	Number of hexadecimal bytes to be converted
RET_VAL	Return	WORD	I, Q, M, D, L	Error message
OUT	Output	VARIANT	D, L	Address at which the result is stored.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#....)	Description
0000	No error
8101	Invalid pointer at the IN parameter, e.g., because a non-existing data block is referenced.
8182	Input buffer is too small for data in the N parameter.
8120	Invalid format in the IN parameter.

Error code* (W#16#....)	Description
8151	Non-supported data type in the IN parameter.
8401	Invalid pointer at the OUT parameter, e.g., because a non-existing data block is referenced.
8482	Output buffer is too small for data in the N parameter.
8420	Invalid format in the OUT parameter.
8451	Non-supported data type in the OUT parameter.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

ASCII characters and hexadecimal values

The following table shows the ASCII characters and the corresponding hexadecimal values:

Hexadecimal digit	ASCII-coded hexadecimal value	ASCII character
0	30	"0"
1	31	"1"
2	32	"2"
3	33	"3"
4	34	"4"
5	35	"5"
6	36	"6"
7	37	"7"
8	38	"8"
9	39	"9"
A	41	"A"
B	42	"B"
C	43	"C"
D	44	"D"
E	45	"E"
F	46	"F"

Example

The following table shows examples of the conversion of hexadecimal numbers to ASCII character strings:

IN	N	OUT	Data type of OUT	ENO status
• WORD#16#0123 • UINT#10#291	2	'0123'	STRING, WSTRING	1
• DWORD#16#123AF012 • INT#10#305852434	4	'123AF012'	STRING, WSTRING	1
WORD#16#ABE3	2	CHAR#['A', 'B', 'E', '3']	ArrayOfCHAR, ArrayOfW-CHAR	1
WORD#16#9250	2	[16#39, 16#32, 16#35, 16#30]	ArrayOfBYTE	1

IN	N	OUT	Data type of OUT	ENO status
ArrayOfWORD#[16#123A, 16#0000, 16#ABCD, 16#0000]	2	[16#0031, 16#0032, 16#0033, 16#0041]	ArrayOfWORD	1
WORD#16#1234	2	[16#0031, 16#0032, 16#0033, 16#0034]	ArrayOfWORD	1
DWORD#16#4567ABCD	4	[16#0034, 16#0035, 16#0036, 16#0037, 16#0041, 16#0042, 16#0043, 16#0044]	ArrayOfWORD	1
ArrayOfD-WORD#[16#A800_0037, 16#0000_0041]	1	'A8'	STRING, WSTRING	1
• CHAR#['0', 'A', '2', '3'] • WCHAR#['0', 'A', '2', '3']	2	'3041'	STRING, WSTRING	1
• STRING#'123AFC1A23' • WSTRING#'123AFC1A23'	5	'3132334146'	STRING, WSTRING	1
• Char#'A' • Byte#16#41	1	'41'	STRING, WSTRING	1

LEN: Determine the length of a character string (S7-1200, S7-1500)

Description

A tag of the (W)STRING data type contains two lengths: the maximum length and the current length (this is the number of currently valid characters).

- The maximum length of the character string is specified for each tag in the STRING keyword in square brackets. The number of bytes occupied by a string is 2 greater than the maximum length.
- The maximum length of the character string is specified for each tag in the WSTRING keyword in square brackets. The number of words occupied by a character string is 2 greater than the maximum length.
- The current length represents the number of the character places actually used. The current length must be less than or equal to the maximum length.

You use the instruction "LEN" to query the current length of the character string specified at the IN input parameter. This is output as numerical value at the output par OUT. An empty string ('') has the length zero.

If errors occur during processing of the instruction, the value "0" is output at the OUT parameter.

Note

Reading out the maximum length

You use the "MAX_LEN (Page 3512)" instruction to read out the maximum length of a string.

Parameter

The following table shows the parameters of the "LEN" instruction:

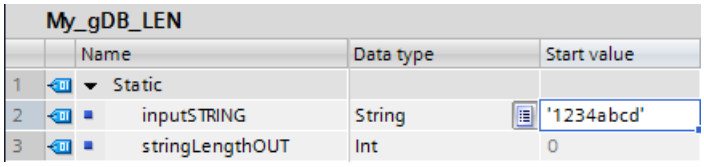
Parameter	Declaration	Data type	Memory area	Description
IN	Input	STRING, WSTRING	D, L or constant	Character string
OUT	Return	INT	I, Q, M, D, L	Number of valid characters

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

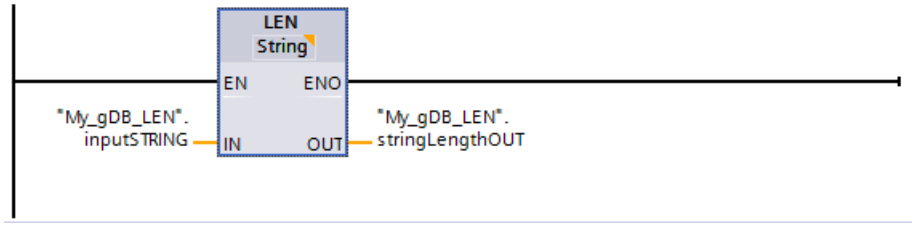
Example

In the following example, you determine the length of a character string with the STRING data type.

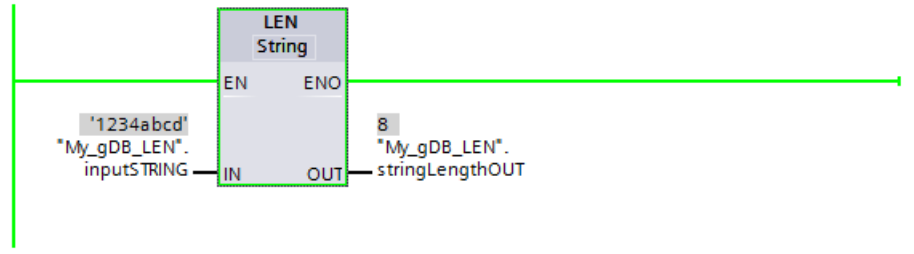
Create two tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows.



The number of actually occupied characters of the character string ("inputSTRING") is determined and returned as a numerical value at output parameter "OUT" ("stringLengthOUT").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

CONCAT: Combine character strings (S7-1200, S7-1500)

Description

You use the instruction "CONCAT" to combine the character string at the IN1 input parameter with the character string at the IN2 input parameter. The result is output at the OUT output parameter in (W)STRING format. If the resulting character string is longer than the tag specified in the OUT output parameter, then the resulting character string will be limited to the available length.

If errors occur during processing of the instruction and it is possible to write the OUT output parameter, an empty string will be output.

Parameters

The following table shows the parameters of the "CONCAT" instruction:

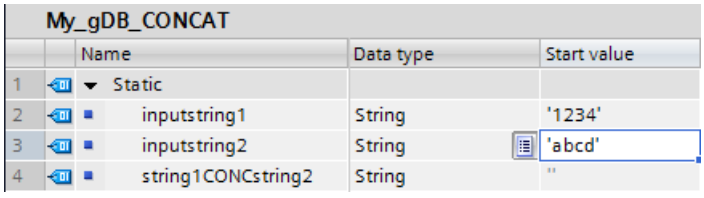
Parameter	Declaration	Data type	Memory area	Description
IN1	Input	STRING, WSTRING	D, L or constant	Character string
IN2	Input	STRING, WSTRING	D, L or constant	Character string
OUT	Return	STRING, WSTRING	D, L	Resulting character string

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

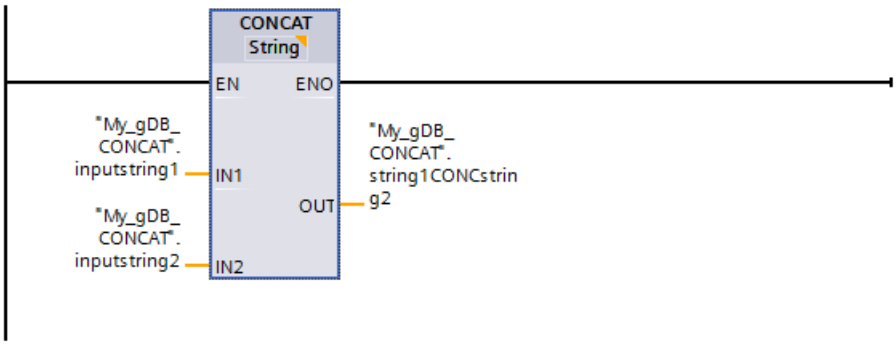
Example

In the following example, you connect two character strings with the STRING data type.

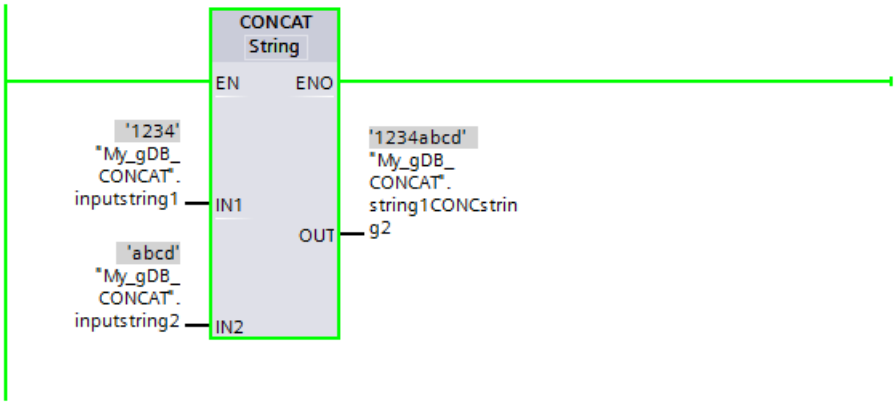
Create three tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows.



The characters of the second character string ("inputstring2") are appended to the first character string ("inputstring1") and the result is output at output parameter OUT ("string1CONCstring2").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

LEFT: Read the left character of a character string (S7-1200, S7-1500)

Description

You use the instruction "LEFT" to extract a partial string beginning with the first character of the string at the IN input parameter. You specify the number of characters to be extracted in the L parameter. The extracted characters are output at the OUT output parameter in (W)STRING format.

If the number of characters to be extracted is greater than the current length of the character string, the OUT output parameter returns the input character string as a result. If the L parameter contains the value "0" or the input value is an empty string, an empty string will be returned. If the value in the L parameter is negative, an empty string will be output.

If errors occur during processing of the instruction and it is possible to write the OUT output parameter, an empty string will be output.

Parameters

The following table shows the parameters of the "LEFT" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	STRING, WSTRING	D, L or constant	Character string
L	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Number of characters to be extracted
OUT	Return	STRING, WSTRING	D, L	Extracted partial string

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

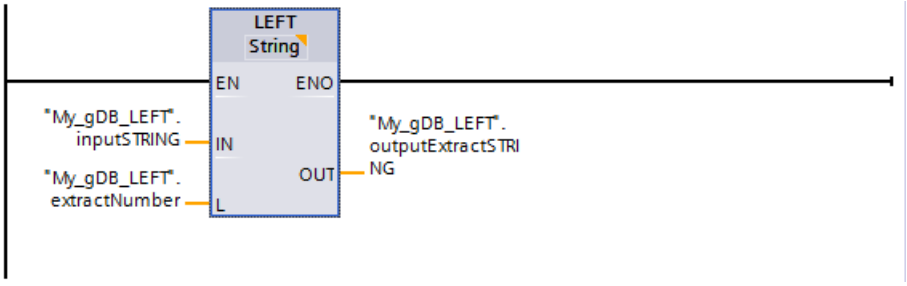
In the following example, you extract a partial character string with the STRING data type from a character string starting from the first character.

Create three tags in a global data block for storing the data.

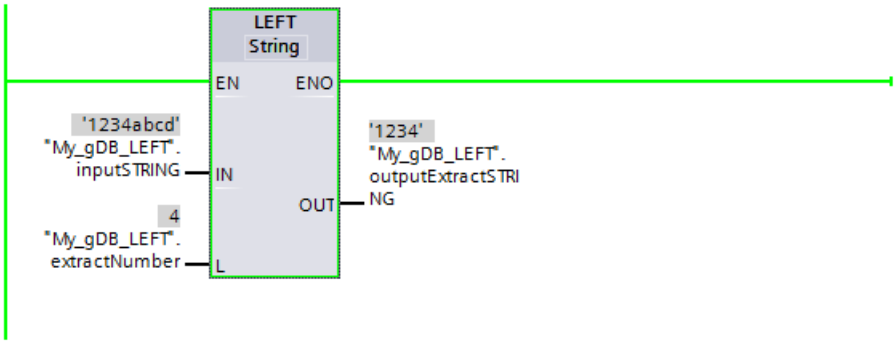
My_gDB_LEFT

	Name	Data type	Start value
1	Static		
2	inputSTRING	String	'1234abcd'
3	extractNumber	Int	4
4	outputExtractSTRING	String	''

Interconnect the parameters of the instruction as follows.



According to the value "4" of parameter L ("extractNumber"), you extract a four-character partial character string from the character string ("inputSTRING") starting from the first character from the left. The extracted partial character string is output at output parameter OUT ("outputExtractSTRING").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

RIGHT: Read the right characters of a character string (S7-1200, S7-1500)

Description

You use this instruction to extract the last L character in a character string in the input parameter IN. You specify the number of characters to be extracted in the L parameter. The extracted characters are output at the OUT output parameter in (W)STRING format.

If the number of characters to be extracted is greater than the current length of the character string, the OUT output parameter returns the input character string as a result. If the L parameter contains the value "0" or the input value is an empty string, an empty string will be returned. If the value in the L parameter is negative, an empty string will be output.

If errors occur during processing of the instruction and it is possible to write the OUT output parameter, an empty string is output.

Parameters

The following table shows the parameters of the "RIGHT" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	STRING, WSTRING	D, L or constant	Character string
L	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Number of characters to be extracted
OUT	Return	STRING, WSTRING	D, L	Extracted partial string

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

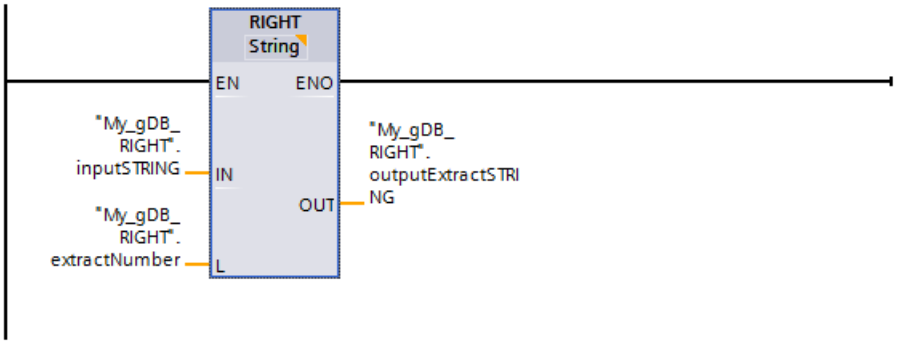
In the following example, you extract a partial character string with the STRING data type from a character string starting from the last character.

Create three tags in a global data block for storing the data.

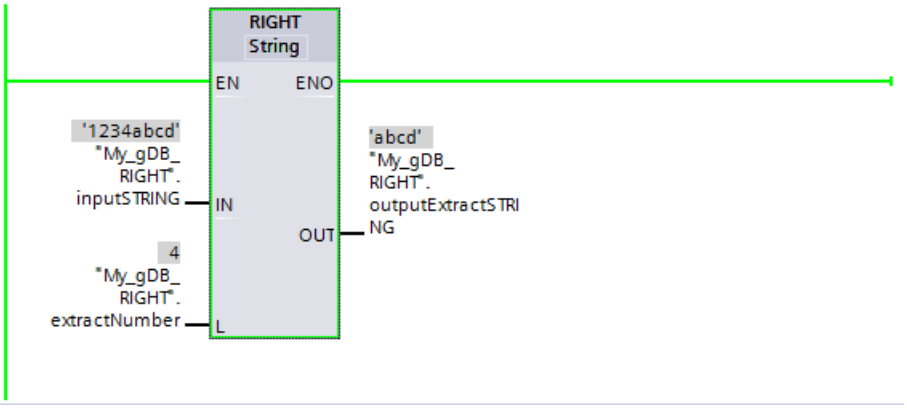
My_gDB_RIGHT

	Name	Data type	Start value
1	Static		
2	inputSTRING	String	'1234abcd'
3	extractNumber	Int	4
4	outputExtractSTRING	String	''

Interconnect the parameters of the instruction as follows.



According to the value "4" of input parameter L ("extractNumber"), you extract a four-character partial character string from the character string ("inputSTRING") starting from the first character from the right. The extracted partial character string is output at output parameter OUT ("outputExtractSTRING").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

MID: Read middle characters of a character string (S7-1200, S7-1500)

Description

You use this instruction to extract a portion of the character string in the IN input parameter. With the P parameter, you specify the position of the first character to be extracted. With the L parameter, you define the length of the character string to be extracted. The extracted partial character string is output to the OUT output parameter.

Note the following rules when executing the instruction:

- If the number of characters to be extracted exceeds the current length of the character string in the IN input parameter, a partial character string will be output. The partial character string starts at the P character string and is continued until the end of the character string.
- If the character position specified in the P parameter falls outside the current character string length in the IN input parameter, an empty character string will be output in the OUT output parameter.
- If the value of the P or L parameter equals zero or is negative, an empty character string will be output in the OUT output parameter.

If errors occur during processing of the instruction and it is possible to write the OUT output parameter, an empty string will be output.

Parameters

The following table shows the parameters of the "MID" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	STRING, WSTRING	D, L or constant	Character string
L	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Length of the string to be extracted
P	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Position of the first character to be extracted (first character = 1)
OUT	Return	STRING, WSTRING	D, L	Extracted partial string

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

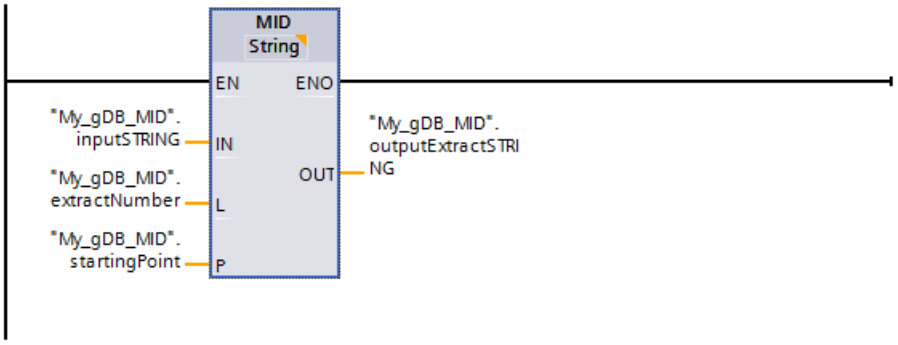
In the following example, you extract a partial character string with the STRING data type from the middle of a character string.

Create four tags in a global data block for storing the data.

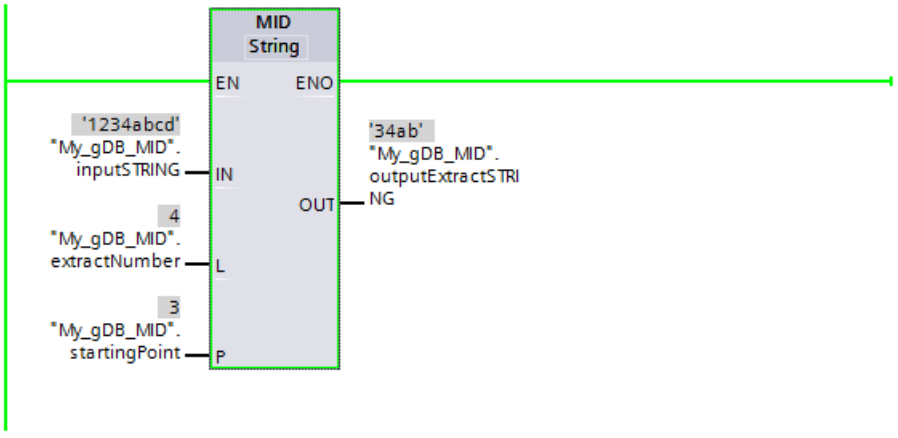
My_gDB_MID

	Name	Data type	Start value
1	Static		
2	inputSTRING	String	'1234abcd'
3	startingPoint	Int	3
4	extractNumber	Int	4
5	outputExtractSTRING	String	''

Interconnect the parameters of the instruction as follows.



According to the value "4" of input parameter L ("extractNumber"), you extract a four-character partial character string from the character string ("inputSTRING"). The extraction starts from the third character ("startingPoint" has the value "3") of the character string ("inputSTRING"). The extracted partial character string is output at output parameter OUT ("outputExtractSTRING").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

DELETE: Delete characters in a character string (S7-1200, S7-1500)

Description

You use this instruction to delete a portion of the character string in the IN input parameter. With the P parameter, you specify the position of the first character to be deleted. You specify the number of characters to be deleted in the L parameter. The remaining partial character string is output at the OUT output parameter in (W)STRING format.

Note the following rules when executing the instruction:

- If the value in the P parameter is less than or equals zero, an empty character string will be output in the OUT output parameter.
- If the value in the P parameter is greater than the current length of the character string in the IN input, the input character string will be returned in the OUT output parameter.

- If the value in the L parameter equals zero, the input character string will be returned in the OUT output parameter.
- If the number of characters to be deleted at the L parameter is greater than the length of the character string at the IN input parameter, the characters starting at the position specified by the P parameter are deleted. The string resulting from this is output.
- If the value in the L parameter is negative, an empty character string will be output.

If errors occur during processing of the instruction and it is possible to write the OUT output parameter, an empty string will be output.

Parameters

The following table shows the parameters of the "DELETE" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN	Input	STRING, WSTRING	D, L or constant	Character string
L	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Number of characters to be deleted
P	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Position of first character to be deleted
OUT	Return	STRING, WSTRING	D, L	Resulting character string

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

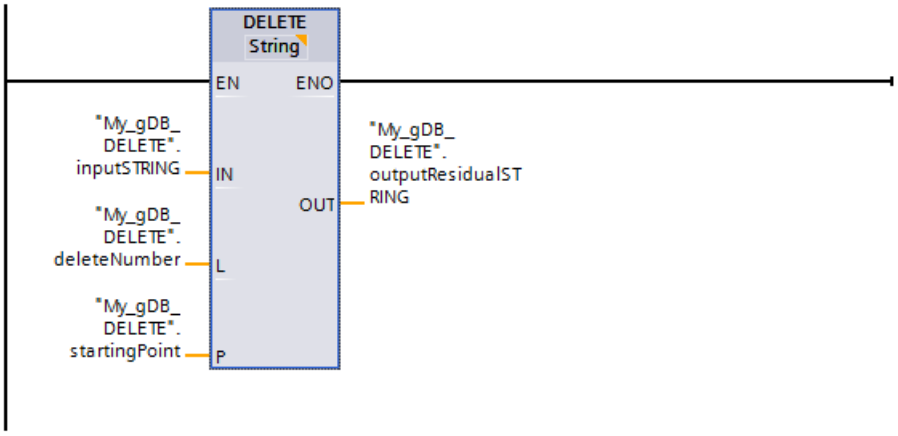
In the following example, you delete characters from a character string with the STRING data type.

Create four tags in a global data block for storing the data.

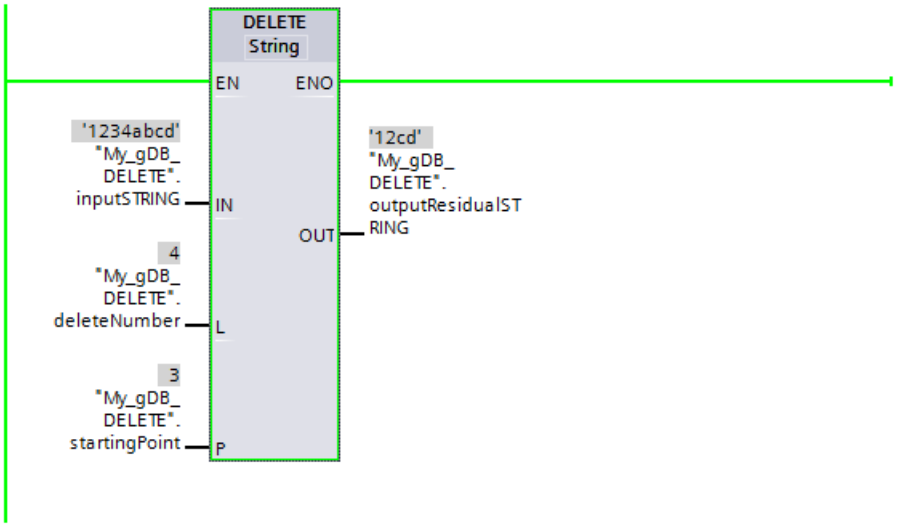
My_gDB_DELETE

	Name	Data type	Start value
1	Static		
2	inputSTRING	String	'1234abcd'
3	startingPoint	Int	3
4	deleteNumber	Int	4
5	outputResidualSTRING	String	''

Interconnect the parameters of the instruction as follows.



According to the value "4" of input parameter L ("deleteNumber"), you delete four characters from the character string ("inputSTRING") starting from the third character ("startingPoint" has the value "3"). The remaining character string is output at output parameter OUT ("outputResidualSTRING").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

INSERT: Insert characters in a character string (S7-1200, S7-1500)

Description

You use this instruction to insert the character string in the IN2 input parameter to the character string in the IN1 input parameter. With the P parameter, you specify the position of the character starting at which the characters are inserted. The result is output at the OUT output parameter in (W)STRING format.

Note the following rules when executing the instruction:

- If the value in the P parameter exceeds the current length of the character string in the IN1 input parameter, the character string of the IN2 input parameter will be appended to the character string of the IN1 input parameter.
- If the value at the P parameter is zero, the character string at the OUT parameter followed by the character string at the IN2 parameter will be output in the IN1 parameter.
- If the value in the P parameter is negative, an empty character string will be output in the OUT output parameter.
- If the resulting character string is longer than the tag specified in the OUT output parameter, the resulting character string will be limited to the available length.

Parameters

The following table shows the parameters of the "INSERT" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN1	Input	STRING, WSTRING	D, L or constant	Character string
IN2	Input	STRING, WSTRING	D, L or constant	String to insert
P	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Insert position
OUT	Return	STRING, WSTRING	D, L	Resulting character string

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

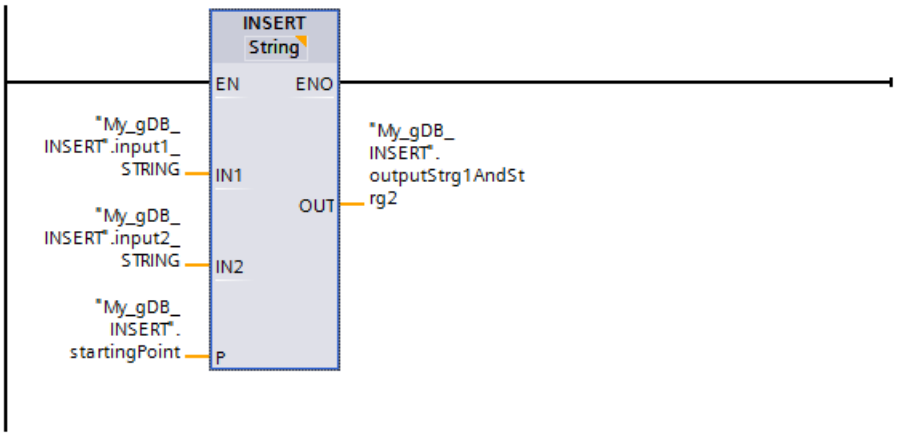
In the following example, you insert a character string in another character string. The data type used is STRING.

Create four tags in a global data block for storing the data.

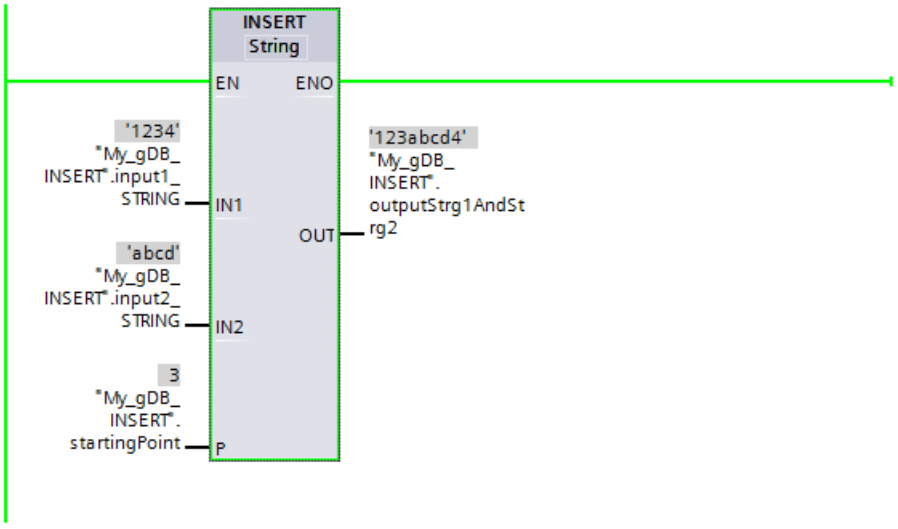
My_gDB_INSERT

	Name	Data type	Start value
1	Static		
2	input1_STRING	String	'1234'
3	input2_STRING	String	'abcd'
4	startingPoint	Int	3
5	outputStrg1AndStrg2	String	''

Interconnect the parameters of the instruction as follows.



The characters of the second character string ("input2_STRING") are inserted in the first character string ("input1_STRING"). According to the value "3" of parameter P ("startingPoint"), characters are inserted after the third character of the first character string ("input1_STRING"). The result is output at output parameter OUT ("outputStrg1AndStrg2").



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

REPLACE: Replace characters in a character string (S7-1200, S7-1500)

Description

You use this instruction to replace a portion of the character string in the IN1 input with the character string in the IN2 input parameter. You specify the position of the first character to be replaced in the P parameter. You specify the number of characters to be replaced in the L parameter. The result is output at the OUT output parameter in (W)STRING format.

Note the following rules when executing the instruction:

- If the value in the P parameter is less than or equals zero, an empty character string will be output in the OUT output parameter.
- If the value in the L parameter is less than zero, an empty character string will be output in the OUT output parameter.
- If P equals one, the character string in the IN1 input will be replaced beginning with (and including) the first character.
- If the value in the P parameter exceeds the current length of the character string in the IN1 input parameter, the character string of the IN2 input parameter will be appended to the character string of the IN1 input parameter.
- If the resulting character string is longer than the tag specified in the OUT output parameter, the resulting character string will be limited to the available length.
- If the value at parameter L is zero, characters are not replaced but inserted. Similar conditions apply as with the instruction INSERT. See also: INSERT: Insert characters in a character string (Page 3541)

Parameters

The following table shows the parameters of the "REPLACE" instruction:

Parameter	Declaration	Data type	Memory area	Description
IN1	Input	STRING, WSTRING	D, L or constant	String with characters to be replaced.
IN2	Input	STRING, WSTRING	D, L or constant	String with characters to be inserted.
L	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Number of characters to be replaced
P	Input	BYTE, INT, SINT, USINT	I, Q, M, D, L or constant	Position of first character to be replaced
OUT	Return	STRING, WSTRING	D, L	Resulting character string

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

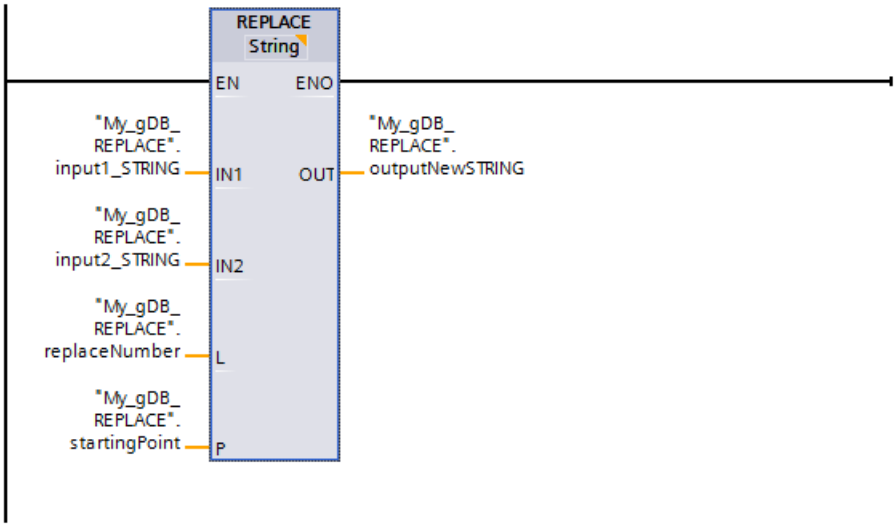
In the following example, you replace part of a character string with another character string. The data type used is STRING.

Create five tags in a global data block for storing the data.

My_gDB_REPLACE

	Name	Data type	Start value
1	Static		
2	input1_STRING	String	'1234'
3	input2_STRING	String	'abcd'
4	replaceNumber	Int	2
5	startingPoint	Int	3
6	outputNewSTRING	String	''

Interconnect the parameters of the instruction as follows.



The second character string ("input2_STRING") is added to the first character string ("input1_STRING") starting from the third character ("startingPoint" has the value "3"). According to the value "2" of parameter L ("replaceNumber"), the third and fourth characters of the first character string ("input1_STRING") are replaced in the process. The result is output at output parameter OUT ("outputNewSTRING").

My_gDB_REPLACE

	Name	Data type	Start value	Monitor value
1	Static			
2	input1_STRING	String	'1234'	'1234'
3	input2_STRING	String	'abcd'	'abcd'
4	replaceNumber	Int	2	2
5	startingPoint	Int	3	3
6	outputNewSTRING	String	''	'12abcd'

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

FIND: Find characters in a character string (S7-1200, S7-1500)

Description

You use this instruction to search through the character string at the IN1 input parameter for a specific character string.

- You specify the value to be searched for in the IN2 input parameter. The search is made from left to right.
- The position of the first occurrence is output in the OUT output parameter. If the search returns no match, the value "0" will be output in the OUT output parameter.

If an invalid character is specified at the IN2 parameter or if an error occurs during processing, the value "0" is output at the OUT parameter.

Parameters

The following table shows the parameters of the "FIND" instruction:

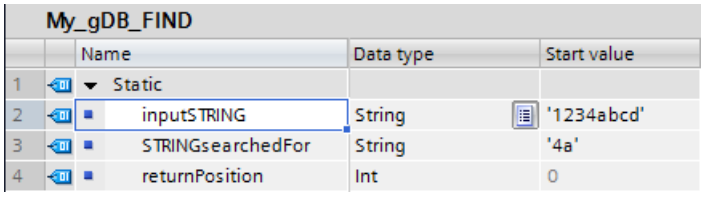
Parameter	Declaration	Data type	Memory area	Description
IN1	Input	STRING, WSTRING	D, L or constant	String searched through
IN2	Input	STRING, WSTRING	D, L or constant	Character string that is searched for
OUT	Return	INT	I, Q, M, D, L	Character position

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

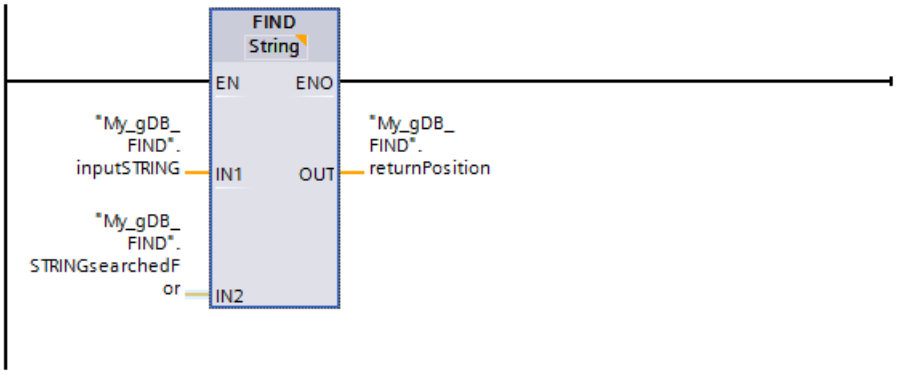
Example

In the following example, you search for a character string in another character string. The data type used is STRING.

Create three tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows.



The first character string ("inputSTRING") searches for the value "4a" of the second character string ("STRINGsearchedFor"). The character position ("4") at which the searched character string begins is output at output parameter OUT ("returnPosition").

My_gDB_FIND

	Name	Data type	Start value	Monitor value
1	Static			
2	inputSTRING	String	'1234abcd'	'1234abcd'
3	STRINGsearchedFor	String	'4a'	'4a'
4	returnPosition	Int	0	4

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

Runtime information (S7-1200, S7-1500)

GetSymbolName: Read out a tag on the input parameter (S7-1200, S7-1500)

Description

With the "GetSymbolName" instruction, you use the input parameter of a block to read out the name of a tag interconnected at the parameter.

Note

Increased code memory requirement

The use of GetSymbolName leads to an increased code memory requirement. The more frequently the instruction is called and the longer the actual symbols are, the greater the memory requirement.

If a block is used multiple times in the project and called each time by a different tag, you can use the "GetSymbolName" instruction to evaluate the name of the calling tag. The process value of the tag is irrelevant in this case.

- You specify the input parameters of the block interface at the VARIABLE parameter. Use only an interface parameter for this parameter and no PLC or data block tags.
- To limit the length of the read tag name, use the SIZE parameter. If the name has been truncated, this is indicated by the characters "..." (Unicode character 16#2026) appears at the end of the name. Note that this character has the length 1.
  An example to illustrate the meaning of the SIZE parameter. The following tag name is read out via the block interface: "MyPLCTag" (The double quotes at the start and end belong to the name.)

SIZE	GetSymbolName returns	Explanation
1	'...'	• First character of WSTRING:' • Identifier that the name was truncated: ... • Last character of WSTRING:'
2	'"...'	• First character of WSTRING:' • The first character of the name and identifier that the name was truncated:"... • Last character of WSTRING:'
3	'"M...'	• First character of WSTRING:' • The first two characters of the name and identifier that the name was truncated:"... "M... • Last character of WSTRING:'
6	'"MyPL...'	• First character of WSTRING:' • The first five characters of the name and identifier that the name was truncated: "MyPL... • Last character of WSTRING:'
0	'"MyPLCTag"'	• First character of WSTRING:' • All characters of the name: "MyPLCTag" • Last character of WSTRING:'

- The name read is output at the OUT parameter.

Note

Truncating the tag name in SCL

If SIZE = 0 or SIZE is greater than the maximum length of WSTRING, the name is then truncated, without the character "..." at the end of the string, in SCL in cases where the tag name does not fit in OUT.

Parameter

The following table shows the parameters of the "GetSymbolName" instruction:

Parameter	Declaration	Data type	Memory area	Description
VARIABLE	Input	PARAMETER	Parameter sections Input, Output, InOut	Selection of the local interface to which you want to read out the name of the input parameter supply.
SIZE	Input	DINT	I, Q, M, D, L	Limits the number of characters output at the OUT parameter. • SIZE > 0: GetSymbolName returns the first SIZE characters of the name. • SIZE = 0: GetSymbolName returns the entire name. • SIZE < 0: GetSymbolName returns the last SIZE characters of the name.
OUT	Return	WSTRING	D, L	Output of the tag name of the input parameters supply

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

In the following example, you read out the name of a tag that is interconnected via the input parameter of a block.

Create two tags in a global data block for storing the data.

SLI_gDB_SymName

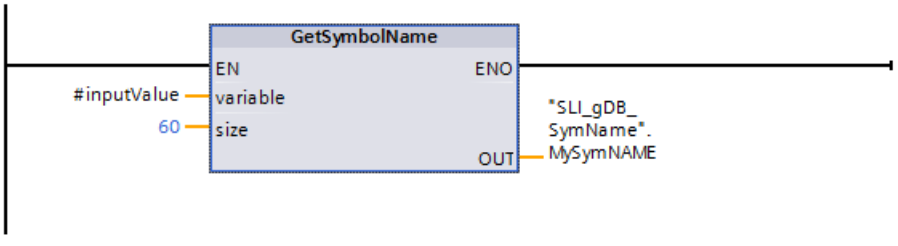
	Name	Data type	Start value
1	Static		
2	MySymNAME	WString	WSTRING#''
3	symbolVALUE	Byte	16#42

Create an FC "SLI_FC_GetSymbolName". Create the following local tag in the block.

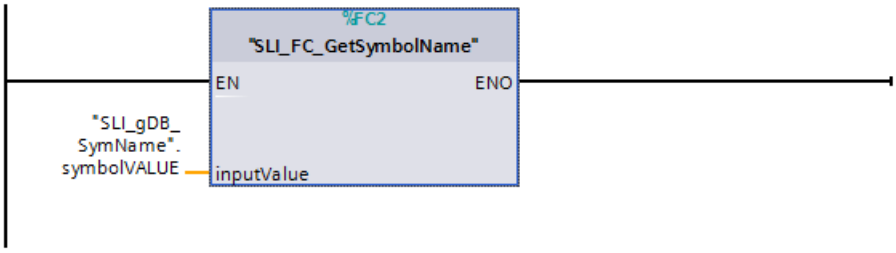
SLI_FC_GetSymbolName

	Name	Data type
1	Input	
2	inputValue	Byte

Call the instruction "GetSymbolName" in the FC and interconnect "GetSymbolName" as follows.



Create an FC "SLI_FC_FCCall_GetSymbolName". In the FC, call the FC "SLI_FC_GetSymbolName" and interconnect this as follows:



The "GetSymbolName" instruction is executed in the "SLI_FC_GetSymbolName" block. Input parameter inputValue of the "SLI_FB_GetSymbolName" block is examined for its interconnection using input parameter VARIABLE of the instruction. In doing so, the "symbolVALUE" tag is read out and output as a character string at output parameter OUT ("MySymNAME"). According to the value of input parameter SIZE, the length of the character string is limited to 60 characters.

SLI_gDB_SymName

		Name	Data type	Start value	Monitor value
1		Static			
2		MySymNAME	WString	WSTRING#''	WSTRING#'"SLI_gDB_SymName".symbolVALUE'
3		symbolVALUE	Byte	16#42	16#42

Program code

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

Querying GetSymbolPath: actual parameter at beginning of a call path (S7-1200, S7-1500)

Description

Use the instruction "GetSymbolPath" to read the names of the parameters transferred at the beginning of a call path and passed via multiple block calls.

Note

Increased code memory requirement

The use of GetSymbolPath leads to an increased code memory requirement. The more frequently the instruction is called and the longer the actual symbols are, the greater the memory requirement.

- At the parameter VARIABLE of the instruction, you specify the formal parameter names from which the name is to be read when the interconnected actual parameter is called during runtime.
  - If the tag of a data block is used to supply the input parameters, this tag - consisting of the name of the DB and access path to the structured tag, is output.
  - If a PLC tag is used to supply the input parameters, the name of the PLC tag is output.
  - If a constant is used to supply the input parameters, this constant is output.
- To limit the length of the read tag name, use the SIZE parameter. If the name has been truncated, this is indicated by the character "..." (Unicode character 16#2026) at the end of the name. Note that this character has the length 1.
  An example to illustrate the meaning of the SIZE parameter. The following tag name is read out by the block interface: "MyPLCTag" (The double quotes at the start and end belong to the name.)

SIZE	GetSymbolPath returns	Explanation
1	'...'	• First character of WSTRING:' • Identifier that the name was truncated: ... • Last character of WSTRING:'
2	'"...'	• First character of WSTRING:' • The first character of the name and identifier that the name was truncated:"... • Last character of WSTRING:'
3	'"M...'	• First character of WSTRING:' • The first two characters of the name and identifier that the name was truncated:"... "M... • Last character of WSTRING:'
6	'"MyPL...'	• First character of WSTRING:' • The first five characters of the name and identifier that the name was truncated: "MyPL... • Last character of WSTRING:'
0	'"MyPLCTag"'	• First character of WSTRING:' • All characters of the name: "MyPLCTag" • Last character of WSTRING:'

Note

Truncating the tag name in SCL

If SIZE = 0 or SIZE is greater than the maximum length of WSTRING, the name is then truncated, without the character "..." at the end of the string, in SCL in cases where the tag name does not fit in OUT.

Parameter

The following table shows the parameters of the "GetSymbolPath" instruction:

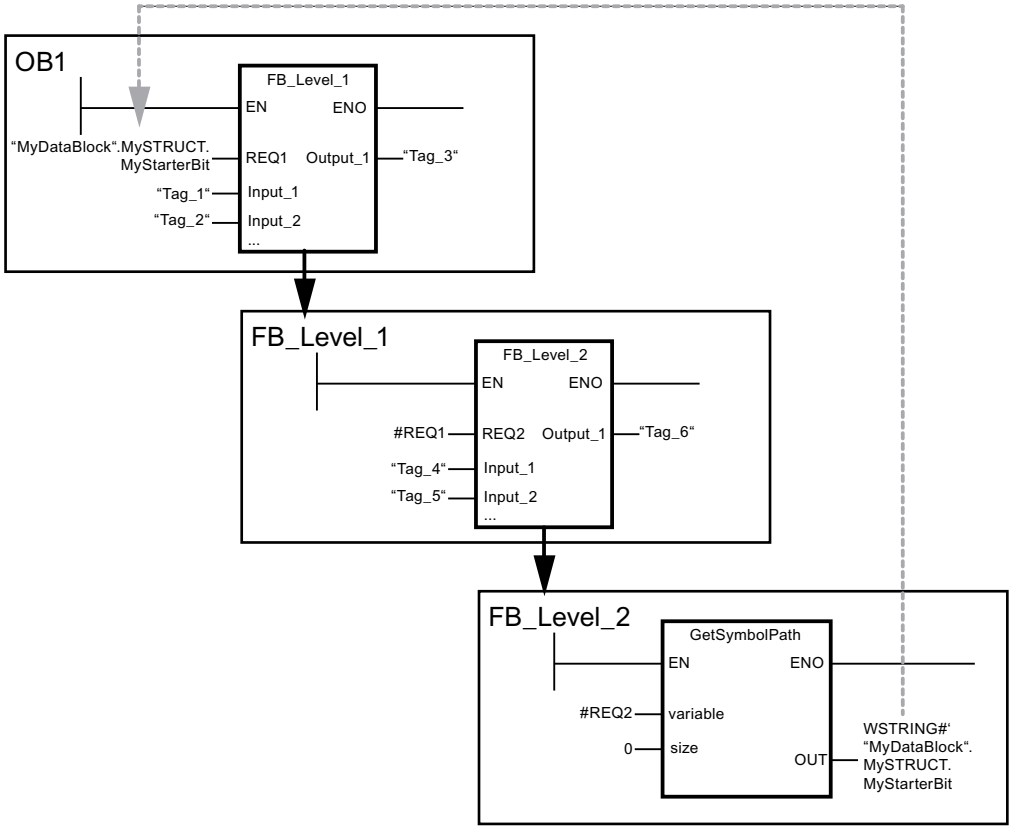
Parameter	Declaration	Data type	Memory area	Description
VARIABLE	Input	PARAMETER	Parameter sections Input, Output, InOut	Selection of the formal parameter for which you want to read out the name of the corresponding actual parameter at the beginning of the call path.
SIZE	Input	DINT	I, Q, M, D, L or constant	Limits the number of characters output at the OUT parameter. • SIZE > 0: GetSymbolPath returns the first SIZE characters of the name. • SIZE = 0: GetSymbolPath returns the entire name. • SIZE < 0: GetSymbolPath returns the last SIZE characters of the name.
OUT	Output	WSTRING	D, L	Output of the tag name of the input parameters supply.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

The following example shows the use of GetSymbolPath over several call levels:

- The FB_Level_1 block is called from the organization block OB1. In turn, the FB_Level_2 block is called from this.
- GetSymbolPath is executed in the FB_Level_2 block to read the name of the parameter whose value is passed at the beginning of the call path to REQ2.
- At the REQ1 parameter, the MyStarterBit tag is interconnected as actual parameter. The bit is located in the MySTRUCT structure in the MyDatablock data block.
  This information is read by GetSymbolPath and output at the OUT parameter.



You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

GetInstanceName: Read out name of the read instance (S7-1200, S7-1500)

Description

You can use the "GetInstanceName" instruction to read the name of the instance data block within a function block.

- To limit the length of the read instance name, use the SIZE parameter. If the name has been truncated, this is indicated by the character "..." (Unicode character 16#2026) at the end of the name. Note that this character has the length 1.
  An example to illustrate the meaning of the SIZE parameter. The following instance name is read out via the block interface: "Level1_DB" (The double quotes at the start and end belong to the name.)

SIZE	GetSymbolPath returns	Explanation
1	'...'	• First character of WSTRING:' • Identifier that the name was truncated: ... • Last character of WSTRING:'
2	'"...'	• First character of WSTRING:' • The first character of the name and identifier that the name was truncated:"... • Last character of WSTRING:'
3	'"L...'	• First character of WSTRING:' • The first two characters of the name and identifier that the name was truncated:"... "L... • Last character of WSTRING:'
6	'"Leve...'	• First character of WSTRING:' • The first five characters of the name and identifier that the name was truncated: "Leve... • Last character of WSTRING:'
0	'"Level1_DB"'	• First character of WSTRING:' • All characters of the name: "Level1_DB" • Last character of WSTRING:'

- The name of the instance data block is written to the OUT parameter. If the name of the instance data block is longer than the maximum length of WSTRING, the name is truncated.

Note

Truncating the instance DB name in SCL

If SIZE = 0 or SIZE is greater than the maximum length of WSTRING, the name is then truncated, without the character "..." at the end of the string, in SCL in cases where the instance DB name does not fit in OUT.

Parameter

The following table shows the parameters of the "GetInstanceName" instruction:

Parameter	Declaration	Data type	Memory area	Description
SIZE	Input	DINT	I, Q, M, D, L or constant	Limits the number of characters output at the OUT parameter. • SIZE > 0: GetInstanceName returns the first SIZE characters of the name. • SIZE = 0: GetInstanceName returns the entire name. • SIZE < 0: GetInstanceName returns the last SIZE characters of the name.
OUT	Output	WSTRING	D, L	Read name of the instance data block

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

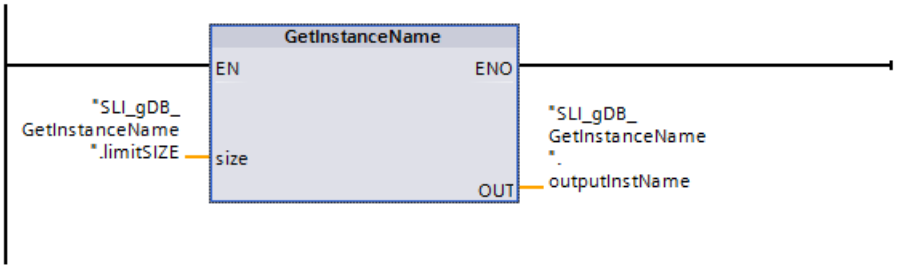
In the following example, you read out the name of an instance data block.

Create two tags in a global data block for storing the data.

SLI_gDB_GetInstanceName

	Name	Data type	Start value
1	Static		
2	limitSIZE	DInt	0
3	outputInstName	WString	WSTRING#''

Create an FB "SLI_FB_GetInstanceName". Interconnect the parameters of the "GetInstanceName" instruction as follows.



The "GetInstanceName" instruction is executed in the "SLI_FB_GetInstanceName" block. By means of the "GetInstanceName" instruction, the associated instance data block of the "SLI_FB_GetInstanceName" block is determined and output as a character string at output parameter OUT ("outputInstName"). According to the value "0" of parameter SIZE ("limitSIZE"), the length of the character string is unlimited.

SLI_gDB_GetInstanceName

	Name	Data type	Start value	Monitor value
1	Static			
2	limitSIZE	DInt	0	0
3	outputInstName	WString	WSTRING#''	WSTRING#'"SLI_FB_GetInstanceName_DB"'

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

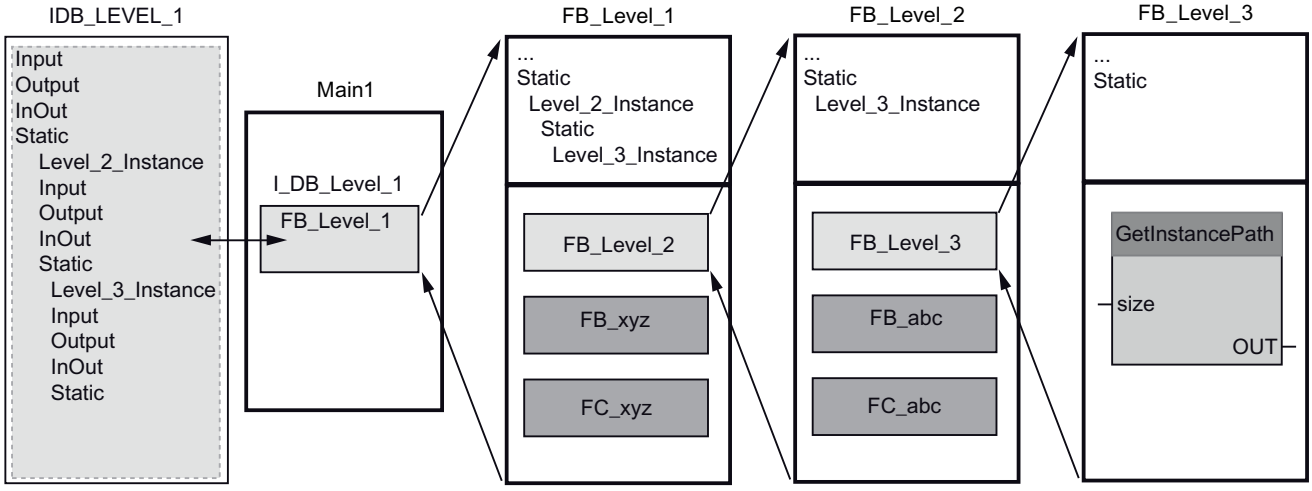
GetInstancePath: Query composite global name of the block instance (S7-1200, S7-1500)

Description

You use the "GetInstancePath" instruction to read the composed global name of the block instance within a function block. The composed global name of the block instance is the path of the complete call hierarchy when multi-instances are used.

In the following example, the "GetInstancePath" instruction is called in the FB_Level_3 function block.

- The FB_Level_3 function block stores its data in the calling FB_Level_2 function block.
- The FB_Level_2 function block in turn stores its data in the calling FB_Level_1 function block.
- The FB_Level_1 function block in turn stores its data in its instance data block IDB_LEVEL_1. Through the use of multi-instances, the instance data block of FB_Level_1 contains all data of the three function blocks.



The "GetInstancePath" instruction returns the following values for this example:

SIZE	GetInstancePath returns	Explanation
1	'...'	• First character of WSTRING:' • Identifier that the name was truncated: ... • Last character of WSTRING:'
2	'"...'	• First character of WSTRING:' • The first character of the name and identifier that the name was truncated:"... • Last character of WSTRING:'

SIZE	GetInstancePath returns	Explanation
3	'"I...'	• First character of WSTRING:' • The first two characters of the name and identifier that the name was truncated:"... "I... • Last character of WSTRING:'
6	'"IDB_...'	• First character of WSTRING:' • The first five characters of the name and identifier that the name was truncated: "IDB_... • Last character of WSTRING:'
0	'"IDB_LEVEL_1".Level_2_Instance.Level_3_Instance'	• First character of WSTRING:' • All characters of the name: "IDB_LEVEL_1".Level_2_Instance.Level_3_Instance • Last character of WSTRING:'

Note

Use of "GetInstancePath" in function blocks with single instance

If the function block in which you call "GetInstancePath" saves data in its own instance data block, the name of the single instance is output as the global name. The result at parameter OUT corresponds in this case to the ""GetInstanceName (Page 3554)" instruction.

Note

Truncating the global name of the block instance in SCL

If SIZE = 0 or SIZE is greater than the maximum length of WSTRING, the name is then truncated, without showing the character "..." at the end of the string, in SCL in cases where the block instance name does not fit in OUT.

Parameter

The following table shows the parameters of the "GetInstancePath" instruction:

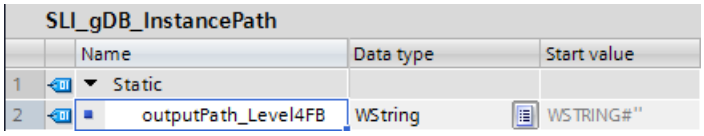
Parameter	Declaration	Data type	Memory area	Description
SIZE	Input	DINT	I, Q, M, D, L or constant	Limits the number of characters output at the OUT parameter. • SIZE > 0: GetInstancePath returns the first SIZE characters of the name. • SIZE = 0: GetInstancePath returns the entire name. • SIZE < 0: GetInstancePath returns the last SIZE characters of the name.
OUT	Output	WSTRING	D, L	Read global name of the block instance. If the global name of the block instance is longer than the maximum length of WSTRING (254 characters), the name is truncated.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

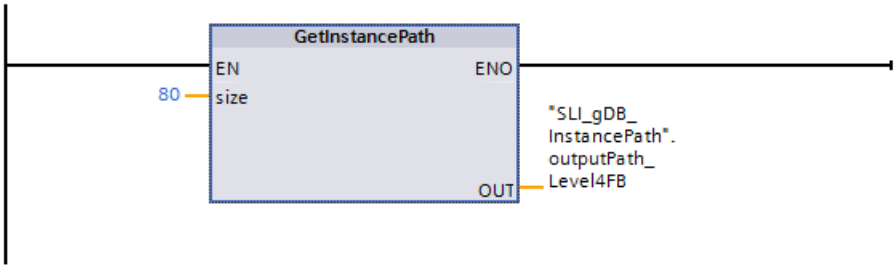
Example

In the following you read out the assembled global names of the block instance.

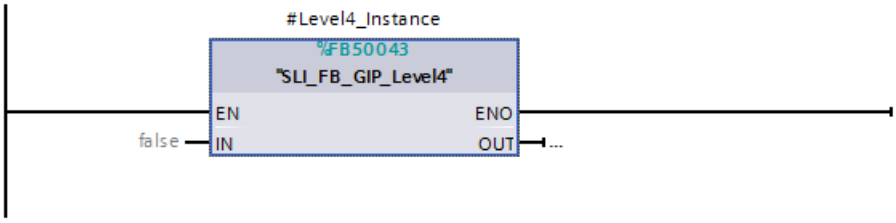
Create one tag for storing the data in a global data block.



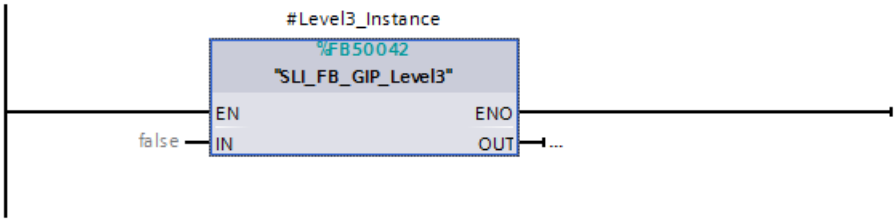
Create an FB "SLI_FB_GIP_Level4" and call the instruction "GetInstancePath" in it.



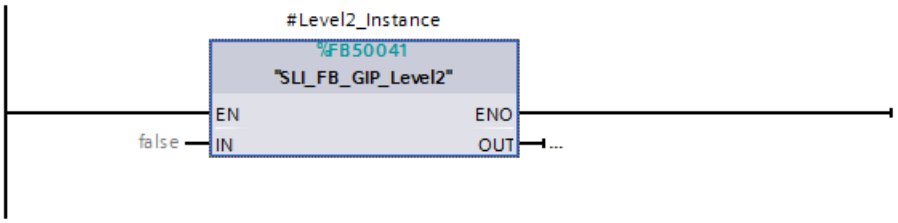
Create an FB "SLI_FB_GIP_Level3" and call the instruction "SLI_FB_GIP_Level4" in it.



Create an FB "SLI_FB_GIP_Level2" and call the instruction "SLI_FB_GIP_Level3" in it.



Create an FB "SLI_FB_GIP_Level1" and call the instruction "SLI_FB_GIP_Level2" in it.

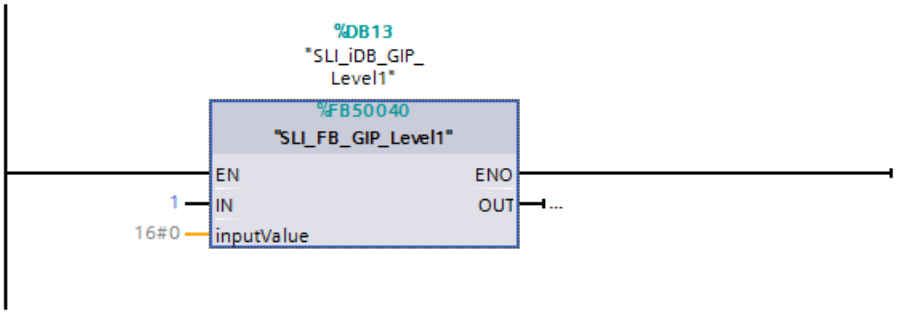


Create the following local tags in the function block "SLI_FB_GIP_Level1".

SLI_FB_GIP_Level1

		Name	Data type	Default value
1		Input		
2		IN	Bool	false
3		inputValue	Byte	16#0
4		Output		
5		OUT	Bool	false
6		InOut		
7		<Add new>		
8		Static		
9		Level2_Instance	"SLI_FB_GIP_Level2"	

Create a cycle OB (OB123" and call the instruction "SLI_FB_GIP_Level1" in it. Interconnect the function block as follows.



The instruction "GetInstancePath" is executed in the OB123. The instruction "GetInstancePath" reads out the instances of all function blocks, from call of instruction in the FB "SLI_FB_GIP_Level4" to the call in the OB123. The names of the instances are output as strings at the output parameter OUT ("outputPath_Level4FB") of the instruction "GetInstancePath". The length of the character string is limited to 80 characters, according to the value of input parameter SIZE of the operation "GetInstancePath".

SLI_gDB_InstancePath

		Name	Data t..	St...	Monitor value
1		Static			
2		outputPath_Level4FB	W...	WSTR	WSTRING#'"SLI_iDB_GIP_Level1".Level2_Instance.Level3_Instance.Level4_Instance'

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

GetBlockName: Read out name of the block (S7-1200, S7-1500)

Description

You can use the "GetBlockName" instruction to read the name of the block in which the instruction is called.

- If the length of the block name is to be limited to a certain number of characters, specify the max. length at the SIZE parameter. If the name has been truncated, this is indicated by the character "..." (Unicode character 16#2026) at the end of the name. Note that this character has the length 1.
  An example to illustrate the meaning of the SIZE parameter. The following block name is read out: "Level1_gbn" (The double quotes at the start and end belong to the name.)

SIZE	GetBlockName returns	Explanation
1	'...'	• First character of WSTRING:' • Identifier that the name was truncated: ... • Last character of WSTRING:'
2	'"...'	• First character of WSTRING:' • The first character of the name and identifier that the name was truncated:"... • Last character of WSTRING:'
3	'"L...'	• First character of WSTRING:' • The first two characters of the name and identifier that the name was truncated:"... "L... • Last character of WSTRING:'
6	'"Leve...'	• First character of WSTRING:' • The first five characters of the name and identifier that the name was truncated: "Leve... • Last character of WSTRING:'
0	'"Level1_gbn"'	• First character of WSTRING:' • All characters of the name: "Level1_gbn" • Last character of WSTRING:'

- The name of the block is written at the RET_VAL parameter. If the name of the block is longer than the maximum length of WSTRING, the name is truncated.

Note

Truncating the block name in SCL

If SIZE = 0 or SIZE is greater than the maximum length of WSTRING, the name is then truncated, without the character "..." at the end of the string, in SCL in cases where the block name does not fit in RET_VAL.

Parameter

The following table shows the parameters of the "GetBlockName" instruction:

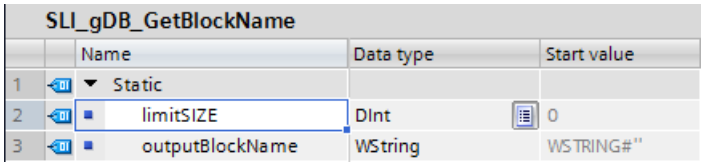
Parameter	Declaration	Data type	Memory area	Description
SIZE	Input	UINT	I, Q, M, D, L or constant	Limits the number of characters output at the RET_VAL parameter. • SIZE > 0: GetBlockName returns the first SIZE characters of the name. • SIZE = 0: GetBlockName returns the entire name. • SIZE < 0: GetBlockName returns the last SIZE characters of the name.
RET_VAL	Output	WSTRING	D, L	Read name of the instance data block

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

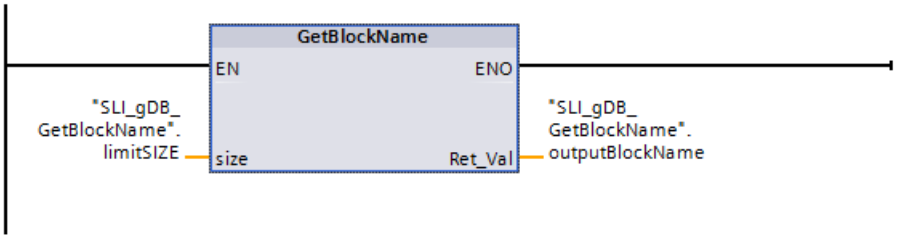
Example

In the following example, you read out the block name.

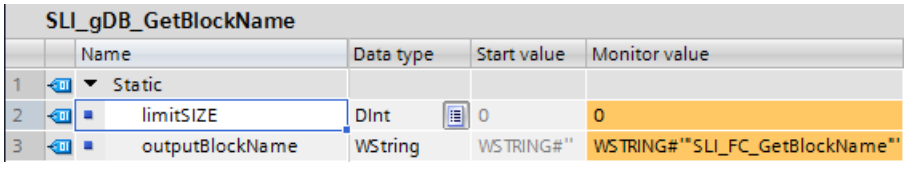
Create two tags in a global data block for storing the data.



Interconnect the parameters of the instruction as follows.



The "GetBlockName" instruction is executed in the "SLI_FC_GetBlockName" block. By means of the "GetBlockName" instruction, the name of the "SLI_FC_GetBlockName" block is read out and output as a character string at output parameter RET_VAL ("outputBlockName"). According to the value "0" of parameter SIZE ("limitSIZE"), the length of the character string is unlimited.



Program code

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

11.4.4.3 Process image (S7-1500)

UPDAT_PI: Update the process image inputs (S7-1500)

Description

With the instruction, you update a partial image partition of the inputs defined in the configuration.

If you have configured repeated signaling of I/O access errors for the system-side process image update, then the selected process image will be updated constantly.

Otherwise, the update can only be performed when the selected process image partition is not updated by the system; in other words if you have not assigned this process image partition to an interrupt OB.

Note

Each logical address that you have assigned to a process image partition of the inputs per configuration no longer belongs to the OB 1 process image of the inputs.

When you update a process image partition with "UPDAT_PI", you must not perform a simultaneous update with the "SYNC_PI (Page 3568)" instruction.

System-side updating of the OB 1 process image of the inputs and the process image partitions of inputs that you have assigned to an interrupt OB takes place independently of "UPDAT_PI" calls.

Parameter

The following table shows the parameters of the "UPDAT_PI" instruction:

Parameter	Declaration	Data type	Memory area	Description
PART	Input	PIP	I, Q, M, D, L or constant	Number of the process image partition of the inputs to be updated. Maximum value range (depending on the CPU): 1 <= n <= 31 (means process image partition n).
RET_VAL	Return	INT	I, Q, M, D, L, P	Error information
FLADDR	Output	WORD	I, Q, M, D, L, P	Address of the first byte to cause an error if an access error has occurred.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8090	Illegal value for PART parameter
8091	The specified process image partition was not yet defined.
8092	The process image partition is being updated by the system and must not be used for the PART parameter.
8093	The process image partition update is being processed in another OB.
80A0	An access error was detected when accessing the I/O.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Note

If you use the instruction for process image partitions of DP standard slaves, the error codes of the "DPRD_DAT (Page 3700)" instruction are also possible. This is the case if you have defined consistency areas greater than 32 bytes.

Example

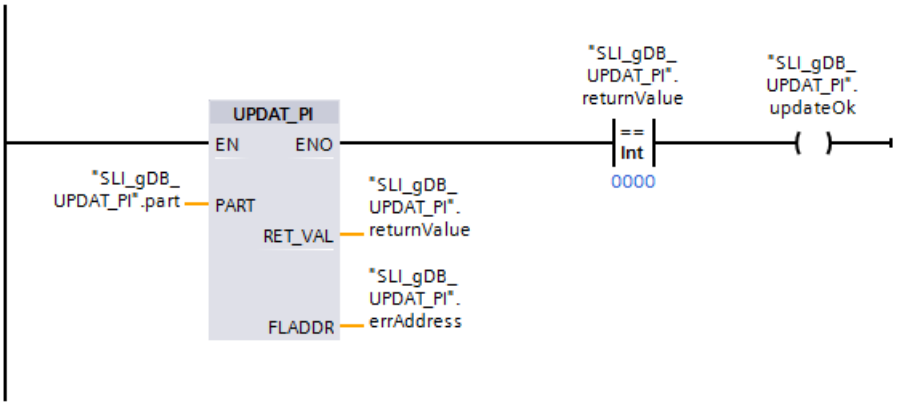
In the following example, you update the process image partition "1" of the inputs defined in the configuration.

Create four tags in a global data block for storing the data.

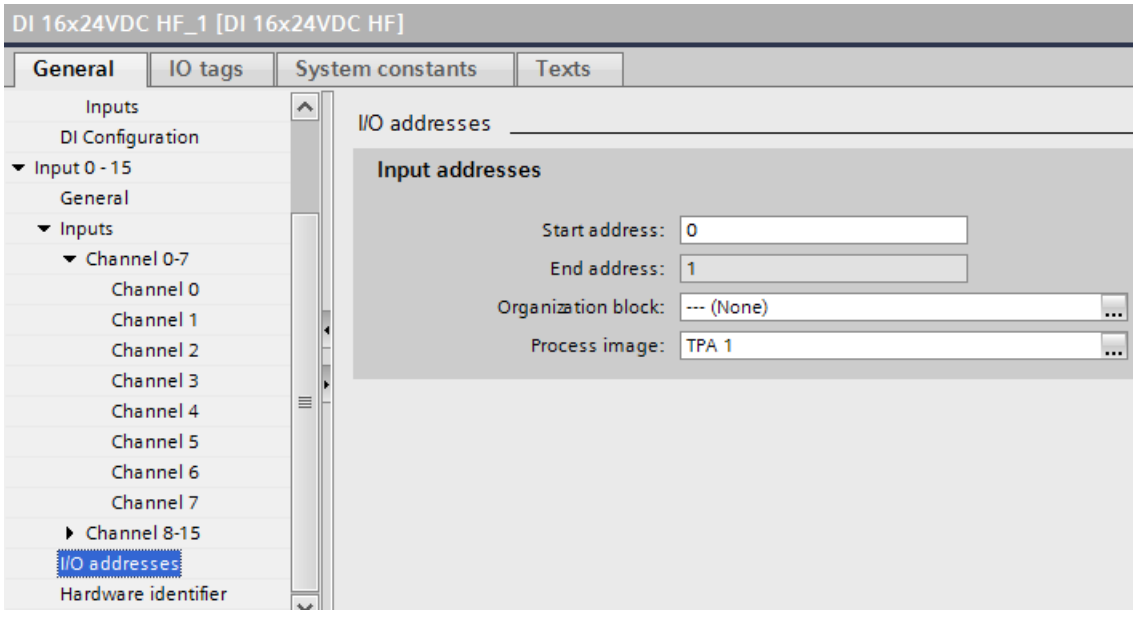
SLI_gDB_UPDAT_PI

	Name	Data type	Start value
1	Static		
2	part	PIP	1
3	returnValue	Int	0
4	errAddress	Word	16#0
5	updateOk	Bool	false

Interconnect the parameters of the instruction as follows.



You create an input module (DI) for your CPU, and configure the process image partition "1" under "Properties > Input 0 - 15 > I/O addresses". You also do not assign the process image partition to any OB.



The number of the process image partition that is to be updated is stored in the input parameter PART ("part") of the instruction "UPDAT_PI". The specified process image partition is updated when the instruction "UPDAT_PI" is called.

The output parameter RET_VAL ("returnValue") indicates that processing took place without errors. Processing without errors sets the coil ("updateOk") to "TRUE". Because no access error occurs, the error address ("errAddress") of the instruction "UPDAT_PI" shows the value "16#0000".

SLI_gDB_UPDAT_PI

	Name	Data type	Start value	Monitor value
1	Static			
2	part	PIP	1	1
3	returnValue	Int	0	0
4	errAddress	Word	16#0	16#0000
5	updateOk	Bool	false	TRUE

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

UPDAT_PO: Update the process image outputs (S7-1500)

Description

You use the instruction to transfer the signal states of a process image partition of the outputs defined by configuration to the output modules.

If you have specified a consistency range for the selected process image partition, corresponding data is transferred as consistent data to the respective I/O module.

Note

Each logical address you have assigned to a process image partition of the outputs during configuration no longer belongs to the OB 1 process image of the outputs.

Outputs that you update with "UPDAT_PO" must not be updated simultaneously with the "SYNC_PO (Page 3569)" instruction.

The OB 1 process image of the outputs and the process image partitions of the outputs that you have assigned to an interrupt OB are transferred by the system to the output modules independently of "UPDAT_PO" calls.

Parameter

The following table shows the parameters of the "UPDAT_PO" instruction:

Parameter	Declaration	Data type	Memory area	Description
PART	Input	PIP	I, Q, M, D, L or constant	Number of the process image partition of the outputs to be transferred. Maximum value range (depending on the CPU): 1 <= n <= 31 (means process image partition n)
RET_VAL	Return	INT	I, Q, M, D, L, P	Error information
FLADDR	Output	WORD	I, Q, M, D, L, P	Address of the first byte to cause an error if an access error has occurred.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8090	Illegal value for PART parameter
8091	The specified process image partition was not yet defined.
8092	The process image partition is being updated by the system and must not be used for the PART parameter.
8093	The process image partition update is being processed in another OB.
80A0	An access error was detected when accessing the I/O.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Note

If you use the instruction for process image partitions of DP standard slaves, the error codes of the "DPWR_DAT (Page 3702)" instruction are also possible. This is the case if you have defined consistency areas greater than 32 bytes.

Example

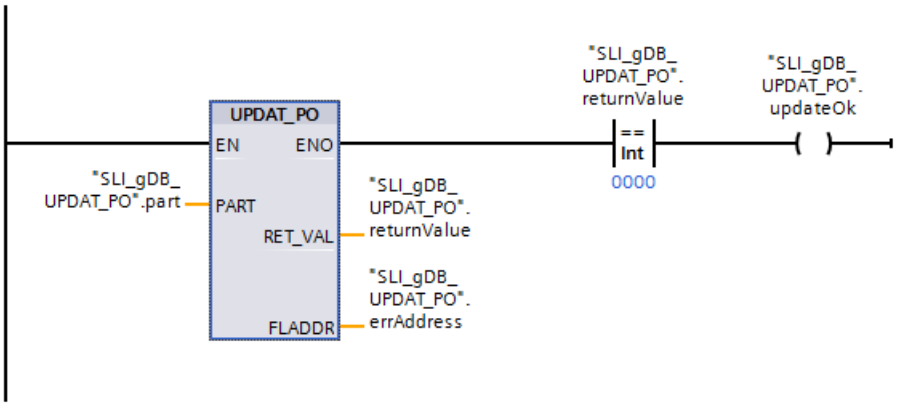
In the following example, you update the process image partition "1" of the outputs defined in the configuration.

Create four tags in a global data block for storing the data.

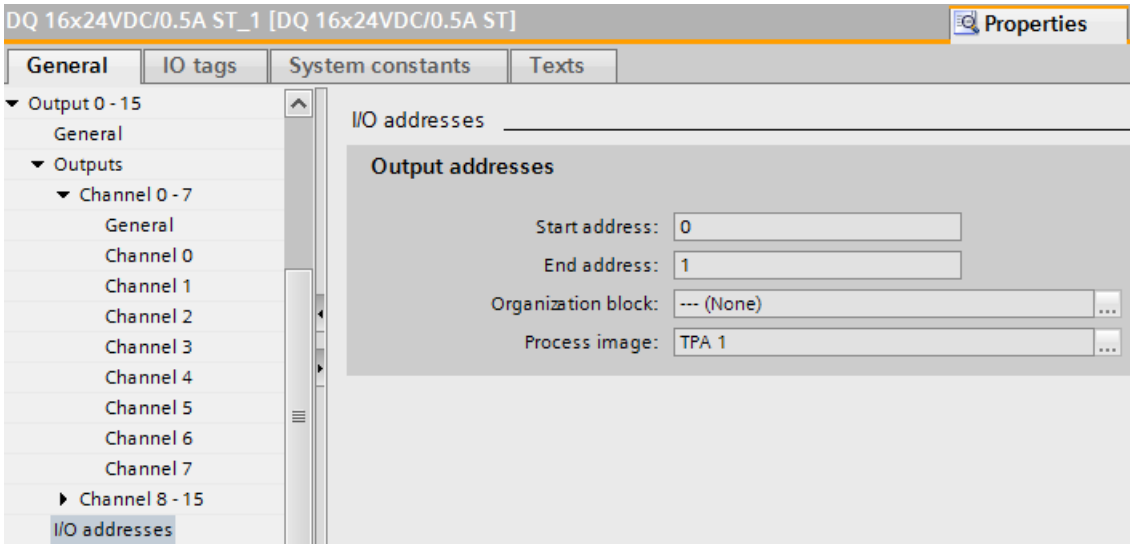
SLI_gDB_UPDAT_PO

	Name	Data type	Start value
1	Static		
2	part	PIP	1
3	returnValue	Int	0
4	errAddress	Word	16#0
5	updateOk	Bool	false

Interconnect the parameters of the instruction as follows.



You create an output module (DQ) for your CPU, and configure the process image partition "1" under "Properties > Output 0 - 15 > I/O addresses". You also do not assign the process image partition to any OB.



The number of the process image partition that is to be updated is stored in the input parameter PART ("part") of the instruction "UPDAT_PO". The specified process image partition is updated when the instruction "UPDAT_PO" is called.

The output parameter RET_VAL ("returnValue") indicates that processing took place without errors. Processing without errors sets the coil ("updateOk") to "TRUE". Because no access error occurs, the error address ("errAddress") of the instruction "UPDAT_PO" shows the value "16#0000".

SLI_gDB_UPDAT_PO

	Name	Data type	Start value	Monitor value
1	Static			
2	part	PIP	1	1
3	returnValue	Int	0	0
4	errAddress	Word	16#0	16#0000
5	updateOk	Bool	false	TRUE

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

SYNC_PI: Synchronize the process image inputs (S7-1500)

Description

The "SYNC_PI" is used to update the process image partition of inputs in isochronous mode. A user program linked to a DP cycle or PN send cycle uses this instruction to update input data acquired in a process image partition of the inputs isochronously and consistently.

Call

"SYNC_PI" is interruptible and is only be called in OBs 61, 62, 63 and 64.

Note

The call of the "SYNC_PI" instruction is only permitted in OBs 61 to 64 if you have assigned the affected process image partition to the associated OB in the HW configuration. A process image partition that you update with "SYNC_PI" must not be updated simultaneously with the "UPDAT_PI (Page 3562)" instruction.

Parameter

The following table shows the parameters of the "SYNC_PI" instruction:

Parameter	Declaration	Data type	Memory area	Value range	Description
PART	Input	PIP	I, Q, M, D, L or constant	1 to 31	Number of the process image partition of the inputs to be updated isochronously.
RET_VAL	Return	INT	I, Q, M, D, L, P	-	Error information
FLADDR	Output	WORD	I, Q, M, D, L, P		Address of the first byte to cause an error if an access error has occurred.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code (W#16#...)*	Explanation
0000	No error occurred.
0001	Consistency warning. The update of the process image partition was distributed over two DP or PN cycles. However, the data in one submodule were consistently transferred.
8090	Illegal value at PART parameter or updating the specified process image partition of the inputs is not permitted in this OB. The specified process image partition was possibly not yet defined. The process image partition of the inputs was not updated.
8092	The process image partition is being updated by the system and must not be used for the PART parameter.

Error code (W#16#...)*	Explanation
8093	The process image partition update is being processed in another OB.
80A0	During updating an access error was detected. The affected inputs were set to "0".
80A1	The update time is after the permitted access window. The process image partition of the inputs was not updated. The DP or PN cycle is too short to ensure enough time for processing the instruction. You must therefore increase the TDP (also known as T_DC), Ti and To timers.
80A2	Access error with consistency warning In the case of update of the specified process image partition an access error with simultaneous consistency warning was detected. • The data of the incorrect inputs was not read by the I/O. In the process image partition of the inputs, the affected inputs are set to "0". • The update of the process image partition of the not involved by an access error input data was distributed over two DP and PN cycles.
80C1	The update time is before the permitted access window. The process image partition of the inputs was not updated.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Note

If you use the "SYNC_PI" instruction for process image partitions of DP standard slaves, error codes of the "DPRD_DAT" instruction are also possible. This is the case if you have defined consistency areas greater than 32 bytes.

See also

DPRD_DAT: Read consistent data of a DP standard slave (Page 3700)

SYNC_PO: Synchronize the process image outputs (S7-1500)

Description

The instruction "SYNC_PO" is used to update a process image partition of outputs in isochronous mode. An application program linked to a DP cycle or a PN send cycle can use this instruction for the consistent isochronous transfer of the computed output data of a process image partition from outputs to the I/O devices.

Call

"SYNC_PO" is interruptible and should only be called in OBs 61, 62, 63 and 64.

Note

The call of the "SYNC_PO" instruction in OBs 61 to 64 is only permitted if you have assigned the affected process image partition to the associated OB in the HW configuration. A process image partition that you update with "SYNC_PO" must not be updated simultaneously with the "UPDAT_PO (Page 3565)" instruction.

Parameter

The following table shows the parameters of the "SYNC_PO" instruction:

Parameter	Declaration	Data type	Memory area	Value range	Description
PART	Input	PIP	I, Q, M, D, L or constant	1 to 31	Number of the process image partition of the outputs to be updated isochronously.
RET_VAL	Return	INT	I, Q, M, D, P	-	If an error occurs while the instruction is being executed, the return value contains an error code.
FLADDR	Output	WORD	I, Q, M, D, L, P	-	Address of the first byte that has caused the error.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code (W#16#...)*	Explanation
0000	No error occurred.
0001	Consistency warning. The update of the process image partition was distributed over two DP or PN cycles. However, the data in one submodule were consistently transferred.
8090	Illegal value at PART parameter or updating of the specified process image partition of the outputs is not permitted in this OB. The specified process image partition was possibly not yet defined. Outputs were not transferred to the I/O. The process image partition of the outputs remains unchanged.
8092	The process image partition is being updated by the system and is not used for the PART parameter.
8093	The process image partition update is being processed in another OB.
80A0	In the case of update of the specified process image partition of the outputs an access error was detected. Incorrect outputs were not transferred to the I/O. During process image partition of the outputs, these outputs remain unchanged.
80A1	Access error with consistency warning In the case of update of the specified process image partition of the outputs an access error with simultaneous consistency warning was detected. • The data of the incorrect outputs were not transferred to the I/O. During process image partition of the outputs, the involved outputs remain unchanged. • The update of the process image partition of the not involved by an access error input data was distributed over two DP and PN cycles.

Error code (W#16#...)*	Explanation
80A2	The update time is after the permitted access window. Outputs were not transferred to the I/O. The process image partition of the outputs remains unchanged.
80C1	The update time is before the permitted access window. Outputs were not transferred to the I/O. The process image partition of the outputs remains unchanged.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Note

If you use the "SYNC_PO" instruction for process image partitions of DP standard slaves, error codes of the "DPWR_DAT" instruction are also possible. This is the case if you have defined consistency areas greater than 32 bytes.

See also

DPWR_DAT: Write consistent data of a DP standard slave (Page 3702)

11.4.4.4 Distributed I/O (S7-1200, S7-1500)

Overview of the types of data records (S7-1200, S7-1500)

Introduction

The data record as property of the hardware corresponds to the summary of data that reflects certain properties of the hardware. Each hardware can have numerous data records as properties. The data records can hereby be distinguished based on their use and their scope of information.

Below you can familiarize yourself with the types of data records and you will be guided through working with data records by assigning modules, systems and instructions.

The following information applies to the S7-1200 and S7-1500 automation systems as well as the ET 200MP and ET 200SP distributed I/O systems.

Types of data records

The following types of data records exist:

Types		Uses
Parameter data records	a) for input modules	Parameter data records contain the properties of the input/output function of the relevant I/O module. With the data record, you, for example, read out a wire break or change the input delay. Parameter data records can be read and written.
	b) for technology modules or modules with the use of the technology function	
	a) for output modules	
Control data records	a) to configure a central or distributed setup	The control data record contains the hardware configuration of the system or setup and allows reconfiguration of a module or an IO system via the user program. The control data record can be read and written.
	b) to reconfigure an IO system	
I&M data records	a) for central setup or distributed setup via PROFINET IO	The I&M data records contain the identification and maintenance data of a module (I&M = Identification & Maintenance). The data differs as follows: • Identification data: – depends on vendor data – read-only – You can, for example, read out the serial number. • Maintenance data – depends on the system and the use of the device – can be read and written. – can be used to change the installation location of the module, for example.
	b) for central setup or distributed setup via PROFIBUS DP	
Diagnostics data records and configuration data records	a) for distributed setup via PROFINET IO	Diagnostics data records and configuration data records each have the same structure and vary only in content and size. The data records can be distinguished as follows: • Diagnostics data records: – The diagnostics data records contain the diagnostics and monitoring data of a module. – Channel diagnostics data records are generated if a channel is in an error state and / or has triggered an interrupt. A diagnostics data record of length 0 is returned if there is no fault/error. – The structure and size of vendor-specific diagnostics data records depend on the vendor. – Diagnostics data records are read-only. • Configuration data records: – The configuration data records contain the data of the actual configuration or preset configuration of a module. – Configuration data records are readable (actual configuration, preset configuration) and writable (preset configuration).
	b) for distributed setup via PROFIBUS DP	

Note

The recipe data records used in WinCC are not properties of the module and have nothing to do with the data records being discussed here.

Use of a data record in the proect

Before you use a data record in you project, please read the relevant manual for the data record structure and any requirements. The "Additional information" section references the respective manual.

Parameter data records

Note

Reading out channels

The data records 0..X and 64..64+X contain channel-specific parameters. Accordingly for example the data record 0 contains the information of channel 0.

The data record 128 contains the information of all channels of the relevant module from the hardware configuration.

The RD_DPAR instruction can only be executed on an S7-1500 CPU or an ET 200 distributed I/O.

Type:	**Parameter data records (a)**				
Relates to the module:	Input module				
Location:	Central setup (S7-1500) or ET 200MP		S7-1200 or ET 200SP (*)		
Instructions:	RDREC, WRREC	RD_DPAR	RDREC, WRREC		RD_DPAR (not for S7-1200)
Data record (DR):	DR 0.X (DR = channel; X = the last channel of the input module)	DR 128	DR 128	*For ET 200SP Namur: DR 0.X (DR = channel; X = the last channel of the input module) or DR 128	DR 128
HW ID:	Use HW ID of the input module				
Example:	For example, S7-1500 DI16x24VDC HF --> channels 0..15; (e.g., use DR 13). See also "Use the program example of the parameter data record with WRREC & RDREC (Page 3598)".	For example, S7-1500 DI16x24VDC HF --> DR 128. See also "Program example of RD_DPAR (Page 3783)".	1st example: Use input module of an S7-1212C CPU --> HW ID of CPU, DR 128 and RDREC / WRREC. 2nd example: Use ET 200SP DI8x24VDC HF --> HW ID of DI module, DR 128 and RDREC / WRREC or RD_DPAR. See also "Program example of RD_DPAR (Page 3783)".		

Type:	**Parameter data records (b)**	
Relates to the module:	Technology modules or module with the use of the technology function	
Location:	Central setup (S7-1500 or S7-1200) or distributed setup (ET 200MP or ET 200SP)	
Instructions:	RDREC, WRREC	RD_DPAR (not for S7-1200)

Type:	**Parameter data records (b)**		
Data record (DR):	DR 128	Alternatively, for DI modules of S7-1500: DR 0.X (DR = channel; X = the last channel of the input module)	DR 128
HW ID:	Use HW ID of the affected module / I/O module.		
Example:	1) Example: CPU S7-1511 PN; high speed counter of the S7-1500 DI16x24VDC HF --> for example, DR 1; use HW ID of the DI module (HSC). Use WRREC/RDREC. 2) Example: TM Count 2x24V --> DR 128; use HW ID of the TM. Use WRREC / RDREC or RD_DPAR. 3) Example: High speed counter of an S7-1200 CPU --> DR 128; use HW ID of CPU and WRREC / RDREC.		

Type:	**Parameter data records (c)**				
Relates to the module:	Output module				
Location:	Central setup (S7-1500) or ET 200MP		S7-1200 or ET 200SP (*)		
Instructions:	RDREC, WRREC	RD_DPAR	RDREC, WRREC		RD_DPAR (not for S7-1200)
Data record (DR):	DR 64.64 + X (DR 64 = channel 0; X = the last channel of the output module)	DR 128	DR 128	*For ET 200SP Namur: DS 64..64 + X (DR 64 = channel 0; X = the last channel of the output module) or DR 128	DR 128
HW ID:	Use HW ID of the output module.				
Example:	For example, DQ16x24VDC/0.5A ST --> channels 0..15; (e.g. for channel 13 use the DR 77).	For example, DQ16x24VDC/0.5A ST --> DR 128.	For example, ET 200SP DQ8x24VDC/0.5A ST --> DR 128.		

Note

Further data records

There may be more data records. E.g. for the input modules:

- With the ET 200SP AI RTD there is the DR 237 for deactivating the wire break check.
- With the ET 200SP AI Energy Meter ST there is the DR 142 for reading the measured data.

Note

Interface module with PROFIBUS DP

If you use an interface module with PROFIBUS DP, you cannot read out parameter data records "0" and "1" from the downstream input/output module. Instead read out the content of the diagnostics data records "0" and "1".

For example with output modules the "parameter data record 0" is the "DR 64" and the "parameter data record 1"is the "DR 65".

Additional information

You can find additional information on the parameter data record in the manual for the relevant I/O module (e.g. S7-1500 DI16x24VDC HF (https://support.industry.siemens.com/cs/ww/en/view/59193001)).

Control data records

Type:	Control data records			
Relates to the module:	All modules of the setup that are to be used			IO system and required IO devices
Location:	Central setup (S7-1200, S7-1500)	Distributed setup (e.g. ET 200MP, ET 200SP)		PROFINET IO
Instructions:	WRREC, RDREC	WRREC, RDREC	RDREC	ReconfigIOSystem
Data record (DR):	DR 196	DR 196	DR 197 (feedback data record)	DR "CTRLREC" (parameter of ReconfigIOSystem)
HW ID:	Use HW ID 33.	Use HW ID of the head [Head] of the IM of the distributed I/O.		Use HW ID of the PROFINET interface (IO controller). Specify the corresponding HW IDs via CTRLREC according to the connections and IO devices that are to be set up.
Example:	For example, for CPU S7-1511-1 PN with DI16x24VDC HF. See also "Use the program example of the control data record with WRREC & RDREC (Page 3607)".	For example, DI16x24VDC HF, ET 200MP IM 155-5 PN ST --> use the ID of "<IO-device_V2.0[Head]>". The real setup contains an additional DI16x24VDC HF, which is not to be considered during the configuration.		For example, connect and activate an S7-1500 CPU (IO controller) with an ET 200MP (IO device) and an ET 200SP (as optional IO device). See also "Program example for ReconfigIOSystem (Page 3683)".

Additional information

For more information on the control data record 196, refer to the system manual of the relevant automation system (e.g. "S7-1500 automation system (https://support.industry.siemens.com/cs/ww/en/view/59191792)").

For more information on the control data record 197, refer to the system manual of the relevant interface module (e.g. "ET 200MP IM 155-5 PN ST (https://support.industry.siemens.com/cs/ww/en/view/59193106)").

You can find more information on the control data record "CTRLREC" with the instruction "ReconfigIOSystem: Reconfigure IO system (Page 3677)".

I&M data records

Note the following use of the I&M data records:

- The data record "I&M0" ("16#AFF0") is used to transfer general information about the module or device.
- The data record "I&M1" ("16#AFF1") is used to transfer the function and installation location of the module or device.
- The data record "I&M2" ("16#AFF2") is used to transfer the installation date of the module or device.
- The data record "I&M3" ("16#AFF3") is used to transfer individual additional information about the module or device.

Note

The data record "I&M0" ("16#AFF0") is supported by every module. All other I&M data records are optional.

Type:	**I&M data records**				
Relates to the module:	Device / module				
Location:	Central setup or distributed setup via PROFINET IO or PROFIBUS DP				
Instructions:	RDREC, WRREC				
Data record (DR):	DR 16#AFF0	DR 16#AFF1	DR 16#AFF2	DR 16#AFF3	Alternatively, for distributed setup via PROFIBUS DP: DR 255
HW ID:	Use HW ID of the module.				
Example:	1) Example: For CPU S7-1511-1 PN --> DR 16#AFF0; use the HW ID of the CPU. See also "Program example of I&M data record with read RDREC (Page 3580)".				2) Example: For ET 200SP with IM 155-6 DP ST --> DR 255; use the HW ID of the IM.

Additional information

You can find additional information on the relevant I&M data record:

- in the system manual of the relevant automation system (e.g. "S7-1500 automation system (https://support.industry.siemens.com/cs/ww/en/view/59191792)").
- In the programming manual "SIMATIC NET PROFINET IO-Base user programming interface (https://support.industry.siemens.com/cs/ww/en/view/26435491)".

Diagnostics data records and configuration data records

Type:	**Diagnostics data records and configuration data records**
Relates to the module:	Device/module or submodule in the IO system
Location:	Distributed setup (PROFINET IO, PROFIBUS DP)

Type:	**Diagnostics data records and configuration data records**			
Instructions:	RDREC, WRREC	RDREC		
Data record (DR):	Configuration (actual/preset):	Channel diagnostics:	Manufacturer-specific diagnostics via interface module:	
	Use DR for the relevant module. See also the addressing level and DR for "PROFINET IO (https://support.industry.siemens.com/cs/ww/en/view/19289930)".		For PROFINET IO: See Interface module	For PROFIBUS DP: DR 0, 1
HW ID:	Use hardware ID of the relevant module (HW ID with "Hw_SubModule" data type).		Use hardware ID of the relevant module (HW ID with "Hw_SubModule" data type).	
Example:	For example, IO controller S7-1513-1 PN; use HW ID of the module on the slot level for the IO device (ET 200MP IM 155-5 PN HF). Read out the preset configuration with DR 16#C001.	For example, IO device (ET 200MP IM 155-5 PN HF) with DQ module causes a wire break. Use the hardware ID of the header module of the IO device. IO controller is S7-1513-1 PN. Use DR 16#E00A for channel diagnostics. See also "Read program example for diagnostic data record with RDREC (Page 3584)".	For example, PROFIBUS DP: ET 200MP with IM 155-5 DP ST; Read DI module --> HW ID ("Hw_SubModule" data type) of the DI module and use DR 1 for diagnostics.	

Additional information

The addressing level, the setup and the important data records for diagnostics and configuration of PROFINET IO are available at:

- in the programming manual "From PROFIBUS DP to PROFINET IO (https://support.industry.siemens.com/cs/ww/en/view/19289930)".

You can find additional and general information on channel diagnostics data records:

- in the online help of STEP 7 in "Diagnostics of PROFINET devices".
- in the system manual for PROFINET (e.g. "PROFINET with STEP 7 as of V13 (https://support.industry.siemens.com/cs/ww/en/view/49948856)").

You can find more information on manufacturer-specific diagnostics data records:

- in the manual for the relevant interface module (for example, ET 200MP IM 155-5 DP ST (https://support.industry.siemens.com/cs/ww/en/view/78324181)).

See also

Sample Library for Instructions (Page 2568)

RDREC: Read data record (Page 3578)

WRREC: Write data record (Page 3596)

RDREC: Read data record (S7-1200, S7-1500)

RDREC: Read data record (S7-1200, S7-1500)

Description

You use the instruction "RDREC" to read the data record with the number INDEX from the module addressed using the ID. This may be a module in a central rack or a distributed module (PROFIBUS DP or PROFINET IO).

- Use the parameter ID to select from which module (DP/PROFINET IO) a data record is to be read. Use only the hardware identifier (HW ID) of the module for the ID parameter. You can find the hardware identifier of the module in "PLC tags > System constants". For example, for an S7-1500 DI module: This is shown as <Name of DI module> with the data type "Hw_SubModule".
- Use the INDEX parameter to select which data record is read from the module. Which data records can be read with which data record number depends on the module. You can find additional information on this in the manual for the relevant module.
- The data records readable via "RDREC" have a different length. Use MLEN to specify the maximum number of data record bytes you want to read. If length "0" is selected at the parameter MLEN, the complete data record is written at the parameter RECORD.
- Select the target range RECORD at least MLEN bytes long. If you read out the full data record with MLEN=0, use the maximum length of the data record for RECORD. The structure (configuration, data types and length) that you use at the parameter RECORD, also depends on which data record is read out by which module.
- The value TRUE for the output parameter VALID indicates that the data record was successfully transferred to the target range RECORD. In this case, the LEN output parameter contains the length of the read data in bytes.
- If an error has occurred during transfer of the data record, this is indicated by the output parameter ERROR. In this case, the output parameter STATUS contains the error information.

Note

The interface of the "RDREC" instruction is identical to that in the "RDREC" FB defined in the standard "PROFIBUS and PROFINET Guideline Communication Function Blocks on PROFIBUS DP and PROFINET IO".

Functional description

"RDREC" is an instruction that works asynchronously, which means its execution extends over multiple calls. You start the data record transfer by calling "RDREC" with REQ= 1.

The job status is displayed via the output parameter BUSY and the two central bytes of output parameter STATUS. The two central bytes of STATUS correspond to the RET_VAL output parameter of the instructions that operate asynchronously.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

The transfer of the data record is complete when the output parameter BUSY has the value FALSE .

Parameter

The following table shows the parameters of the "RDREC" instruction:

Parameter	Declaration	Data type*	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, T**, C** or constant	REQ = 1: Transfer data record
ID	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the hardware module (DP/PROFINET IO) The number is assigned automatically and is stored in the properties of the module or of the interface in the hardware configuration.
INDEX	Input	DINT	I, Q, M, D, L or constant	Data record number
MLEN	Input	UINT	I, Q, M, D, L or constant	Maximum length in bytes of the data record information to be read
VALID	Output	BOOL	I, Q, M, D, L	New data record was received and is valid.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The reading process is not yet complete.
ERROR	Output	BOOL	I, Q, M, D, L	ERROR = 1: An error occurred during the reading process.
STATUS	Output	DWORD	I, Q, M, D, L	Block status or error information
LEN	Output	UINT	I, Q, M, D, L	Length of the read data record information
RECORD	InOut	VARIANT	I, Q, M, D, L	Target range for the data record read.
* There is no implicit conversion in STL, which is why the range of valid data types may be limited. During programming in STL, note the permissible data types in each case in the tooltip of the parameter ** For S7-1500 only.				

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

If you use "RDREC" to read a data record for PROFINET IO, then negative values in the INDEX, MLEN, and LEN parameters will be interpreted as an unsigned 16-bit integer.

Parameter STATUS

For interpretation of the STATUS parameter, see Parameter STATUS (Page 3638).

Diagnostics for Windows applications

It may be possible that Windows can no longer send a signal to the CPU in case of a bluescreen. The OB82 is not called in this case and no diagnostic entry is created.

For additional information, call the instruction "RDREC" (SFB52) with a cyclical OB (e.g. OB1).

To determine the Windows status with the "RDREC" instruction, a new data record index (0x7FFE) is defined for all input/output interfaces.

The "RDREC" instruction is called with the following input parameters:

Input	Type	Value
(HIC) ID	HW_IO	59 (Host Interface RID)
INDEX	DINT	0x7FFE
MLEN	UINT	9
RECORD	VARIANT	Address for each area of 9 bytes

The data record for the Windows status is output by the eighth byte and can have the following content:

- 0x01: Windows running
- 0x02: Windows not running
- 0x03: Bluescreen

Examples

You can find an example here: Program example of I&M data record with read RDREC (Page 3580).

You can find another example here: Read program example for diagnostic data record with RDREC (Page 3584).

You can find additional information and the program code for the examples here: Sample Library for Instructions (Page 2568).

See also

Overview of the types of data records (Page 3571)

Program example of I&M data record with read RDREC (S7-1200, S7-1500)

Introduction

In the following example, you read out the I&M0 data record of a CPU S7-1500.

For more information on data records, refer to the Overview of the types of data records (Page 3571).

Requirement

Create nine tags and a structure in a global data block for storing the data.

You read the hardware identifier of the CPU in “PLC tags > System constants. The CPU should be displayed there as <Local> with the data type “Hw_SubModule“. You store the hardware identifier via the input parameter ID ("id").

SLI_gDB_RDREC_I-M

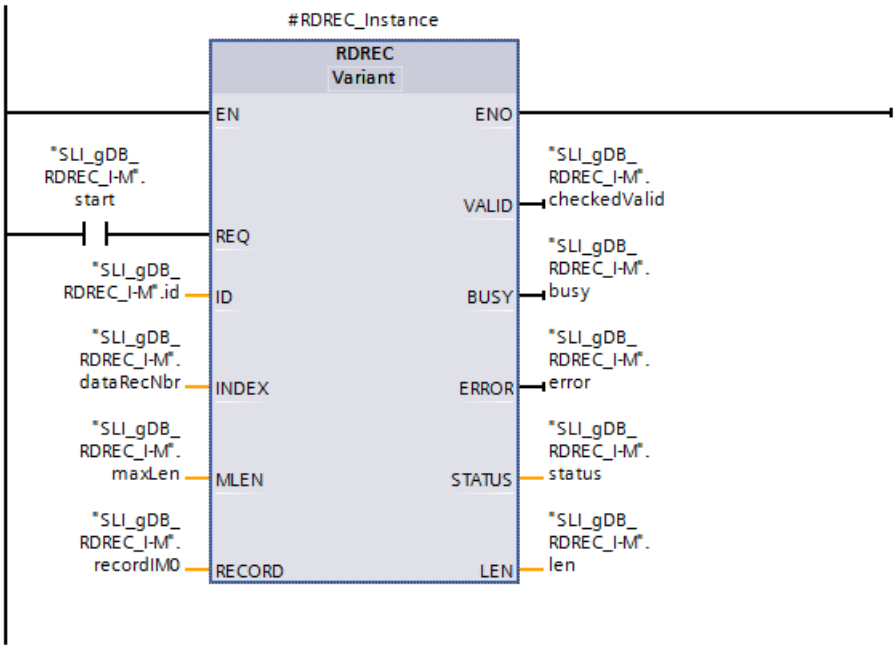
	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	id	HW_IO	49
4	dataRecNbr	DInt	16#AFF0
5	maxLen	UInt	0
6	checkedValid	Bool	false
7	busy	Bool	false
8	error	Bool	false
9	status	DWord	16#0
10	len	UInt	0
11	recordIM0	Struct	
12	BlockType	Word	16#0
13	BlockLength	UInt	0
14	BlockVersion_HL	Word	16#0
15	VendorID_HL	Word	16#0
16	OrderID	Array[0..19] of Byte	
17	IM_SerialNbr	Array[0..15] of Byte	
18	IM_HWrev	UInt	0
19	IM_SWrev	Struct	
20	IM_RevCounter	UInt	0
21	IM_ProfileID	UInt	0
22	IM_ProfileSpecType	Word	16#0
23	IM_Version	Word	16#0
24	IM_Supported	Word	16#0

Create the following local tags in an FB.

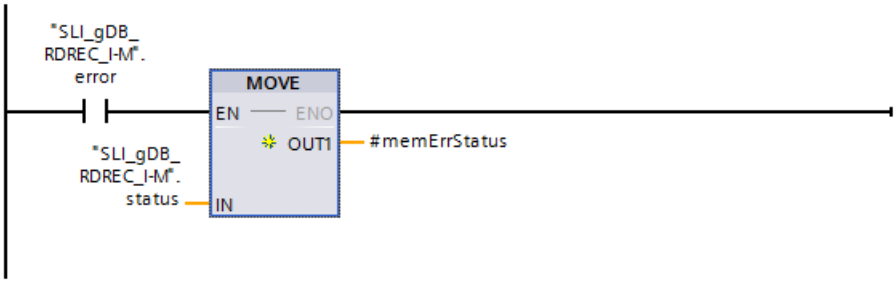
memErrStatus	DWord	16#0	Set in IDB

Interconnect parameters

Network 1: Interconnect the parameters of the "RDREC" instruction as follows.



Network 2: To save the status if an error occurs, interconnect the “status“ variable as follows



Result

When the input parameter REQ ("start") returns the signal state "TRUE", the "RDREC" instruction is started. Via the input parameter ID ("id") the instruction "RDREC" calls the module. With the input parameter INDEX ("dataRecNbr") the I&M0 data record of the CPU is called.

Over several calls the "RDREC" instruction reads out the I&M0 data record and saves the data record using the parameter RECORD ("recordIM0").

Among other things the I&M0 data record contains the following information.

- The vendor name ("VendorID_HL")
- The article (“OrderID“) & device number (“IM_SerialNbr“)

- Hardware ("IM_HWrev") & software information ("IMSWrev")
- IM version ("IM_Version") and the bandwidth supported I&Mdata records ("IM_Supported")

recordIM0	Struct		
BlockType	Word	16#0	16#0020
BlockLength	UInt	0	56
BlockVersion_HL	Word	16#0	16#0100
VendorID_HL	Word	16#0	16#002A
OrderID	Array[0..19] of Byte		
IM_SerialNbr	Array[0..15] of Byte		
IM_HWrev	UInt	0	3
IM_SWrev	Struct		
Refix	Byte	16#0	16#56
FunctEnh	Byte	16#0	16#01
BugFix	Byte	16#0	16#05
InternalChange	Byte	16#0	16#00
IM_RevCounter	UInt	0	0
IM_ProfileID	UInt	0	0
IM_ProfileSpecType	Word	16#0	16#0000
IM_Version	Word	16#0	16#0101
IM_Supported	Word	16#0	16#001E

The length of the read data record is stored using the output parameter LEN ("len"). According to the value of the input parameter MLEN ("maxLen"), has the value “0“ the data record information that is to be read is unlimited.

During processing the output parameter BUSY ("busy") is set to "TRUE" and VALID ("checkedValid") to "FALSE". A successful reading result ("checkedValid" to "TRUE") is not output until processing is complete. The block status is displayed at the output parameter STATUS ("status"). Processing takes place without errors.

SLI_gDB_RDREC_I-M

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	id	HW_IO	49	16#0031
4	dataRecNbr	DInt	16#AFF0	45040
5	maxLen	UInt	0	0
6	checkedValid	Bool	false	TRUE
7	busy	Bool	false	FALSE
8	error	Bool	false	FALSE
9	status	DWord	16#0	16#0000_0000
10	len	UInt	0	60
11	recordIM0	Struct		

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

RDREC: Read data record (Page 3578)

Read program example for diagnostic data record with RDREC (S7-1200, S7-1500)

Introduction

In the following example, you use RDREC to read the diagnostic data record "16#E00A" of an IO device. When a wire break is triggered, the error information is output as a diagnostic data record.

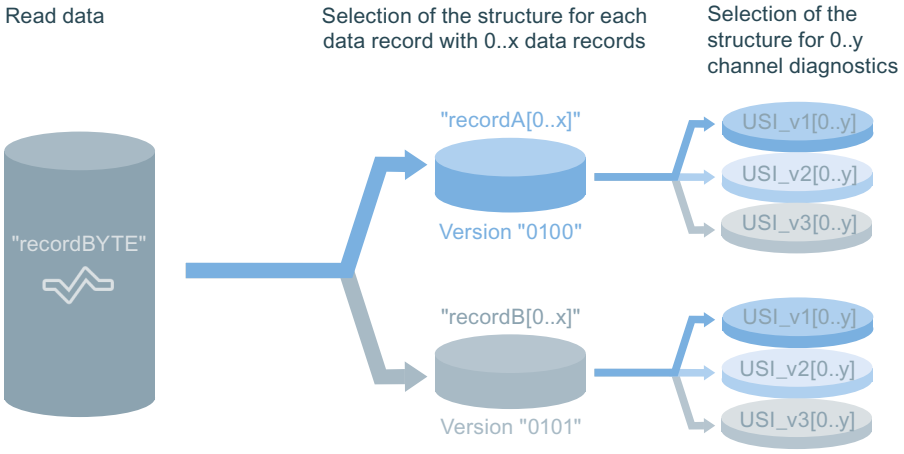
For more information on data records, refer to the Overview of the types of data records (Page 3571).

Note

Number of diagnostic data records

Each slot of a module triggering an error generates its own data record for the IO device. Diagnostics can thus contain multiple records.

The example is intended to read multiple hardware errors (channel errors) of multiple modules and copy them to the appropriate structure of a memory array. The memory arrays ("recordA", "recordB") can hold a maximum of six data records with a maximum of 16 channels each based on a specified setting.



Requirements

Hardware installed

The following hardware is required:

- An IO controller (e.g. S7-1513-1 PN)
- An IO device (e.g. ET 200MP IM 155-5 PN HF)
- One or more digital output modules of the HF type (for example, DQ 4x24VDC HF)

Note

Trigger wire break

Make sure that the channels of the DQ module that are to trigger the wire break are not wired.

Note

Procedure for configuration of the hardware

Configure the hardware as follows:

1. Connect the CPU and IO device over PROFINET and assign the CPU to the IO device as an IO controller.
2. Open the path "Properties > Output 0-x > Outputs > Channel 0-y" for the respective DQ module.
3. Select each channel to be configured and select the "Wire break" option in the ""Diagnostics" area.

PLC data types created

You need the following PLC data types for the transfer and storage of data records:

- For the body content of the "A"/"0100" structure:

Rd_0100body

	Name	Data type	Default value
1	BlockType	Word	16#0
2	BlockLength	UInt	0
3	BlockVersion_HL	Word	16#0
4	SlotNumber	UInt	0
5	SubSlotNumber	UInt	0
6	ChannelLocation	Word	16#0
7	ChannelProperties	Array[0..15] of Bool	
8	UserStructureIdentifier	Word	16#0

- For the body content of the "B"/"0101" structure:

Rd_0101body

	Name	Data type	Default value
1	BlockType	Word	16#0
2	BlockLength	UInt	0
3	BlockVersion_HL	Word	16#0
4	API	DWord	16#0
5	SlotNumber	UInt	0
6	SubSlotNumber	UInt	0
7	ChannelLocation	Word	16#0
8	ChannelProperties	Array[0..15] of Bool	
9	UserStructureIdentifier	Word	16#0

- For the structure of type 1 in accordance with the value from "UserStructureIdentifier": Type 1 of the structure is used for basic channel diagnostics.

USI_v1

	Name	Data type	Default value
1	ChannelNumber	UInt	0
2	ChannelProperties	Array[0..15] of Bool	
3	ChannelErrorType	Word	16#0

- For the structure of type 2 in accordance with the value from "UserStructureIdentifier": Type 2 of the structure is used for extended channel diagnostics.

USI_v2

	Name	Data type	Default value
1	ChannelNumber	UInt	0
2	ChannelProperties	Array[0..15] of Bool	
3	ChannelErrorType	Word	16#0
4	ExtChErrorType	Word	16#0
5	ExtChAddValue	DWord	16#0

- For the structure of type 3 in accordance with the value from "UserStructureIdentifier": Type 3 of the structure is used for manufacturer-specific channel diagnostics. The configuration of the structure is not defined in this example. For this example, a length of 28 bytes is assumed for type 3.

USI_v3

	Name	Data type	Default value
1	ChannelDiagData3	Array[0..27] of Byte	

- For the summary of the "A"/"0100" structure:

RECdiag_0100

	Name	Data type
1	Body0100	"Rd_0100body"
2	USI_DetailsType1	Array[0..15] of "USI_v1"
3	USI_DetailsType2	Array[0..15] of "USI_v2"
4	USI_DetailsType3	Array[0..15] of "USI_v3"

- For the summary of the "B"/"0101" structure:

RECdiag_0101

	Name	Data type
1	Body0101	"Rd_0101body"
2	USI_DetailsType1	Array[0..15] of "USI_v1"
3	USI_DetailsType2	Array[0..15] of "USI_v2"
4	USI_DetailsType3	Array[0..15] of "USI_v3"

Note

Size of the arrays

If you want to change the array sizes (for read memory, data record or channels), you need to do the following:

- For channels: Adapt the size of the arrays ("Array[0..y] of USI_v..") in the PLC data types ("RECdiag_0100", "RECdiag_0101"). Adapt the "maxChannelPos" value in the gDB.
- For data records: Adapt the size of the arrays ("recordA", "recordB") in the gDB. Adapt the "maxRecordPos" value in the gDB.
- For the read memory ("recordBYTE"): Adapt the size of the array ("recordBYTE") and the value of "maxPosition" in the gDB. Note that the diagnostics data record "16#E00A" can have a maximum total of 4176 bytes.

Watch table created

To trigger a wire break, use the "Force" function of a watch table.

If you force the value "16#FFFF" for the "%QW0" output, for example, all channels with configured wire break using the output trigger a hardware error.

Storage of data

Create the following tags and structures for storing data in a global data block (gDB):

SLI_gDB_RDREC_Diag

	Name	Data type	Start value
1	Static		
2	id	HW_IO	262
3	dataRecNbr	DInt	16#E00A
4	startRead	Bool	false
5	maxLength	UInt	0
6	maxPosition	UInt	4175
7	recordBYTE	Array[0..4175] of Byte	
8	readLength	UInt	0
9	checkedValid	Bool	false
10	busy	Bool	false
11	error	Bool	false
12	memErrStatus	DWord	16#0
13	countCalls	UInt	0
14	maxChannelPos	UInt	15
15	maxRecordPos	UInt	5
16	recordA	Array[0..5] of "RECdiag_0100"	
17	recordB	Array[0..5] of "RECdiag_0101"	
18	data_sorting_done	Bool	false
19	outputOverflow	Struct	
20	Too_many_bytes	Bool	false
21	Too_many_records	Bool	false
22	Too_many_channels	Bool	false
23	different_structs_used	Bool	false
24	countRecs	UInt	0
25	chosenStruct	UInt	0
26	posForRec	UInt	0
27	posForChannel	UInt	0
28	resetOps	Struct	
29	reset	Bool	false

You read the hardware identifier of the head of the IO device and store the hardware identifier using the "id" tag. You can find the hardware identifier for the header module with "PLC tags > System constants" and with an entry such as <Head~IO_Device>. The data type used is "HW_SubModule".

FC "SLI_FC_start_RDiag": Interconnect parameters

To start the "RDREC" instruction, based on a hardware fault, create the FC "SLI_FC_start_RDiag".

Create the following interconnection in the FC.



You call the FC "SLI_FC_start_RDiag" in a diagnostic interrupt OB (OB82).

FC "SLI_FC_reset_RDiag": Interconnect parameters

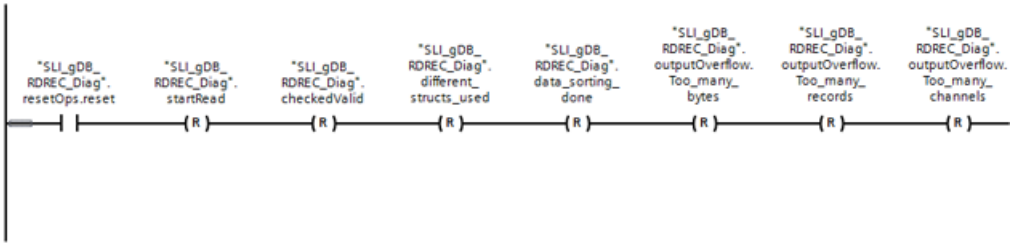
To reset process values after an error, create the FC "SLI_FC_reset_RDiag".

Create the following local tag in the FC.



Create the following interconnections in the FC.

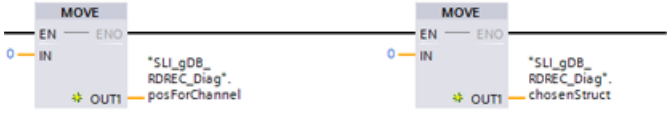
Network 1: To reset the process values, create the following interconnections. Part 1:



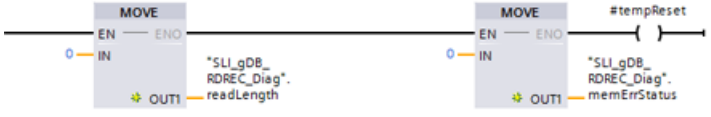
Part 2:



Part 3:



Part 4:



Network 2: To reset the "reset" tag, create the following interconnections.



FB "SLI_FB_chooseStruct_RDiag": Interconnect parameters

You create the FB "SLI_FB_chooseStruct_RDiag" to select the appropriate structure for the data record and to copy the data to the appropriate location in the array.

You can find additional information in the "Result" section ("Network 6") or in the comment field of the networks (see "Program code").

FB "SLI_FB_RDREC_Diag": Interconnect parameters

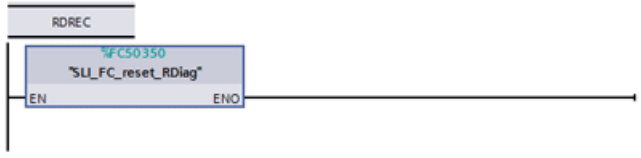
You create the FB "SLI_FC_start_RDiag" to call and process the "RDREC" instruction and all other subsequent processes.

Create the following local tags in the FB.

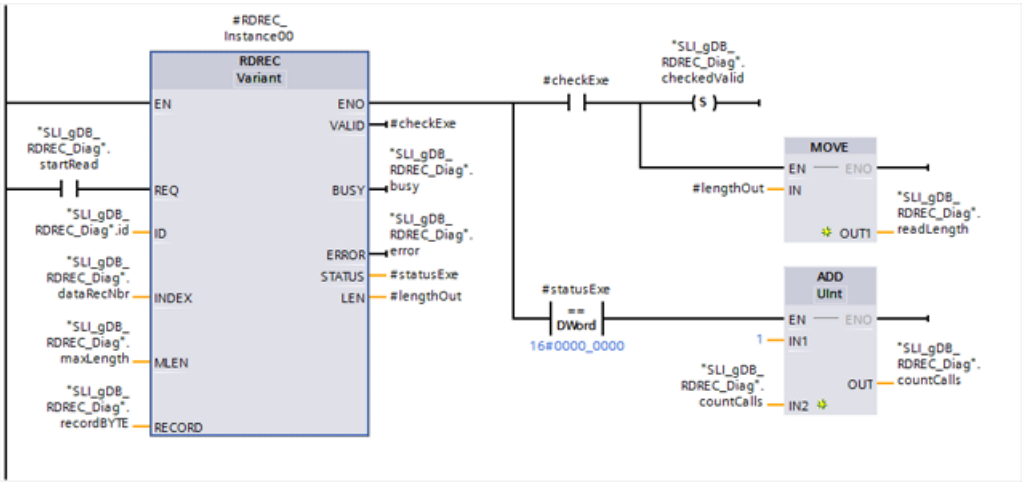
	Static			
	statusExe	DWord	16#0	Nicht re...
	checkExe	Bool	false	Nicht rema...
	lengthOut	UInt	0	Nicht rema...
	RDREC_Instance00	RDREC		

Create the following interconnections in the FB:

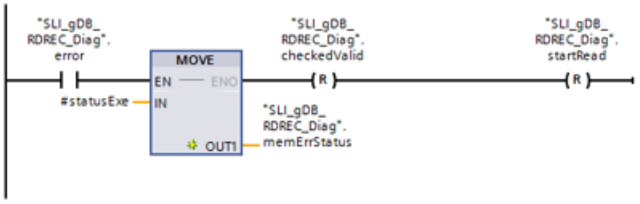
Network 1: Call the FC "SLI_FC_reset_RDiag" and create the "Label" instruction with the name "RDREC".



Network 2: Interconnect the parameters of the "RDREC" instruction as follows.



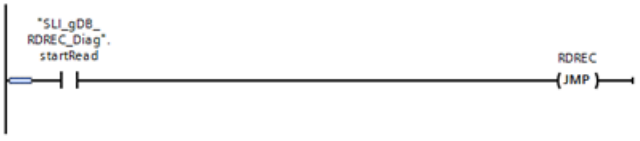
Network 3: To save the status if a RDREC error occurs, interconnect the parameter as follows



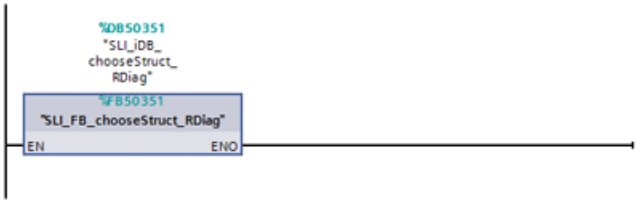
Network 4: To end the processing of RDREC, create the following interconnections.



Network 5: To jump back to network 1, create the following interconnections.



Network 6: Call the FB "SLI_FB_chooseStruct_RDiag" as follows.



You call the FB "SLI_FB_RDREC_Diag" in a cyclic OB (OB1).

Event FB "SLI_FB_RDREC_Diag"

Network 1:

If the NO contact ("reset") has the "TRUE" signal state, the process values of the FB "SLI_FB_chooseStruct_RDiag" and the process values of the global data block are reset.

Network 2:

When the input parameter REQ ("startRead") returns the signal state "TRUE", the "RDREC" instruction is started.

Note

Calling "startRead"

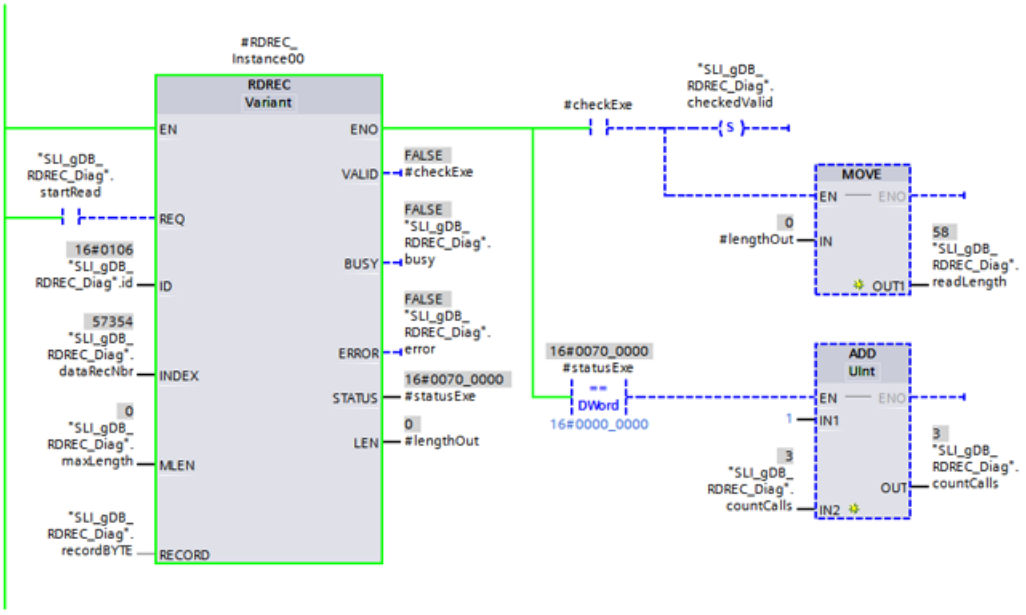
In accordance with the call by OB82, "startRead" is set to "TRUE" every time a hardware error is detected. The successful calls of RDREC can be counted using the "countCalls" tag. After three triggered channel errors, the "countCalls" value is "3".

The "RDREC" instruction calls the header module of the IO device using the input parameter ID ("id"). The input parameter INDEX ("dataRecNbr") is used to call the "16#E00A" diagnostics data record of the IO device.

The "RDREC" instruction reads the diagnostic data over several calls and saves the data read (x data records) with the RECORD parameter ("recordBYTE"). The length of the read data is recorded in the LEN output parameter ("#lengthOut") and saved for further use with the "readLength" tag. In accordance with the value of the input parameter MLEN ("maxLength" has the value "0"), the length of the data to be read is not restricted.

During processing the output parameter BUSY ("busy") is set to "TRUE" and VALID ("checkedValid") to "FALSE". A successful reading result ("checkedValid" set to "TRUE") is not output until processing is complete; it is used by the FB "SLI_FB_chooseStruct_RDiag" as a start condition. To end processing of the FB "SLI_FB_chooseStruct_RDiag", the "checkedValid" tag is set to "FALSE".

The block status is displayed at the output parameter STATUS ("#statusExe"). In the example, processing takes place without errors.



Network 3:

When the NO contact ("error") supplies the signal state "TRUE", the status ("#statusExe") is saved if an error occurs ("memErrStatus"). The "checkedValid" and "startRead" tags are also reset. The processing of RDREC stops. The processing of the FB "SLI_FB_chooseStruct_RDiag" cannot be started.

Network 4:

When RDREC has been fully processed ("#statusExe" has the value "16#0070_0200"), "startRead" is set to "FALSE". The processing of RDREC stops.

Network 5:

If the NO contact ("startRead") has the signal state "TRUE", processing jumps to network 1 (with the "JMP" instruction) in accordance with the "RDREC" label.

Network 6:

If the "checkedValid" tag has the signal state "TRUE" and the "reset" tag has the signal state "FALSE", the sorting process is started in the FB "SLI_FB_chooseStruct_RDiag".

In the FB "SLI_FB_chooseStruct_RDiag", the structure type of the respective data record and the structure type corresponding to the UserStructureIdentifier are selected. The data of the data record is copied to the correct location in the appropriate array. In addition, processing may be repeated several times according to the size of the respective data record. The information of one or more channels is read in this case.

If the "length of the data record + preceding lengths " ("#allRecLEN") is less than the length of all read data ("readLength") at the end of processing the data record content, processing of the FB starts again from the top. Another data record is read, its contents compared and its data is copied to the correct location of the appropriate array.

The sorting process is completed by setting the local tags ("#readComplete", "#dbComplete", "#limitReach") and the global tag "checkedValid" to "FALSE".

The sorting process is documented based in the following tags:

- "data_sorting_done": This tag indicates that processing of the FB "SLI_FB_chooseStruct_RDiag" is complete and has ended. The "checkedValid" tag is reset, which means processing of the FB cannot be started again.
- "outputOverflow": This structure contains three tags, each of which records a different violation of array limits. For example, more data records might have been read than the corresponding array ("recordA" or "recordB") is capable of storing.
- "different_structs_used": If both "recordA" and "recordB" are used to store the diagnostic data records, the "different_structs_used" tag returns the value "TRUE".
- "countRecs": This tag counts the copied data records.
- "chooseStruct": This tag indicates the structure used last. The value "4" indicates that a data record was last stored in the "recordB" structure with the "USI Type 1" form.
- "posForRec": This tag stores the most recently used location in the array of the data records ("recordA" or "recordB"). The value "1" indicates that a data record was last stored in "recordB[1]" (see "chooseStruct").
- "posForChannel": This tag stores the most recently used location in the array of the channels ("USI_DetailsType[x]"). The value "0" indicates that a channel was stored in "recordB[1]/USI_DetailsType[0]" (see "chooseStruct" and "posForRec").

SLI_gDB_RDREC_Diag

	Name	Data type	Start value	Monitor value
1	Static			
2	id	HW_IO	262	16#0106
3	dataRecNbr	DInt	16#E00A	57354
4	startRead	Bool	false	FALSE
5	maxLength	UInt	0	0
6	maxPosition	UInt	4175	4175
7	recordBYTE	Array[0..4175] of B...		
8	readLength	UInt	0	58
9	checkedValid	Bool	false	FALSE
10	busy	Bool	false	FALSE
11	error	Bool	false	FALSE
12	memErrStatus	DWord	16#0	16#0000_0000
13	countCalls	UInt	0	3
14	maxChannelPos	UInt	15	15
15	maxRecordPos	UInt	5	5
16	recordA	Array[0..5] of "RECd...		
17	recordB	Array[0..5] of "RECd...		
18	data_sorting_done	Bool	false	TRUE
19	outputOverflow	Struct		
20	different_structs_used	Bool	false	FALSE
21	countRecs	UInt	0	2
22	chosenStruct	UInt	0	4
23	posForRec	UInt	0	1
24	posForChannel	UInt	0	0
25	resetOps	Struct		
26	reset	Bool	false	FALSE

Below you can see the data record with the structure "B"/"0101" at position "0" ("recordB[0]") with two read channels.

The value "16#8000" from "ChannelLocation" shows diagnostics on the sub-module level. According to the value "16#0006" of "ChannelErrorType", there is a wire break at the channels 0 and 1 (see "ChannelNumber") of submodule 1 (see "SubSlotNumber"), of the module in slot 2 (see "SlotNumber").

17	recordB	Array[0..5] of "RECd...		
18	recordB[0]	"RECdiag_0101"		
19	Body0101	"Rd_0101body"		
20	BlockType	Word	16#0	16#0010
21	BlockLength	UInt	0	28
22	BlockVersion_HL	Word	16#0	16#0101
23	API	DWord	16#0	16#0000_0000
24	SlotNumber	UInt	0	2
25	SubSlotNumber	UInt	0	1
26	ChannelLocation	Word	16#0	16#8000
27	ChannelProperties	Array[0..15] of Bool		
28	UserStructureIdentifier	Word	16#0	16#8000
29	USI_DetailsType1	Array[0..15] of "USI...		
30	USI_DetailsType1[0]	"USI_v1"		
31	ChannelNumber	UInt	0	0
32	ChannelProperties	Array[0..15] of Bool		
33	ChannelErrorType	Word	16#0	16#0006
34	USI_DetailsType1[1]	"USI_v1"		
35	ChannelNumber	UInt	0	1
36	ChannelProperties	Array[0..15] of Bool		
37	ChannelProperties[0]	Bool	false	FALSE
38	ChannelProperties[1]	Bool	false	FALSE
39	ChannelProperties[2]	Bool	false	FALSE
40	ChannelProperties[3]	Bool	false	TRUE
41	ChannelProperties[4]	Bool	false	FALSE
42	ChannelProperties[5]	Bool	false	FALSE
43	ChannelProperties[6]	Bool	false	TRUE
44	ChannelProperties[7]	Bool	false	FALSE
45	ChannelProperties[8]	Bool	false	TRUE

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

RDREC: Read data record (Page 3578)

WRREC: Write data record (S7-1200, S7-1500)

WRREC: Write data record (S7-1200, S7-1500)

Description

The instruction "WRREC" is used to transfer the RECORD data record to the component addressed using ID. This may be a module in a central rack or a distributed component (PROFIBUS DP or PROFINET IO).

- Use the ID parameter to select the module for which a data record is to be written. Use only the hardware identifier (HW ID) of the module for the ID parameter. You can find the hardware identifier of the module in "PLC tags > System constants". For example, for an S7-1500 DI module: This is shown as <Name of DI module> with the data type "Hw_SubModule".
- Use the INDEX parameter to select which data record is written. The module determines which data records can be written with which data record number. You can find additional information on this in the manual for the relevant module.
- Use LEN to specify the length of the data record to be transmitted in bytes. Select the source area RECORD at least LEN bytes long.
- The value TRUE at output parameter DONE indicates that the data record has been successfully transferred.
- If an error has occurred during transfer of the data record, this is indicated by the output parameter ERROR. In this case, the output parameter STATUS contains the error information.

Note

The interface of the "WRREC" instruction is identical to that in the "WRREC" FB defined in the standard "PROFIBUS and PROFINET Guideline Communication Function Blocks on PROFIBUS DP and PROFINET IO".

Functional description

"WRREC" is an asynchronous instruction. Processing takes place across several calls. You start the data record transfer by calling "WRREC" with REQ = 1.

The job status is displayed via the output parameter BUSY and the two central bytes of output parameter STATUS. The two central bytes of STATUS correspond to the RET_VAL output parameter of the instructions that operate asynchronously.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

Note that you must assign the same value to the actual parameter of RECORD for all "WRREC" calls that belong to one and the same job. The same applies to the actual parameters of LEN.

The transfer of the data record is complete when the output parameter BUSY has the value FALSE.

Parameter

The following table shows the parameters of the "WRREC" instruction:

Parameter	Declaration	Data type**	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, T*, C* or constant	REQ = 1: Transfer data record
ID	Input	HW_IO	I, Q, M, D, L or constant	ID number of the hardware component (DP/PROFINET IO) The number is assigned automatically and is stored in the properties of the component or of the interface in the hardware configuration.
INDEX	Input	DINT	I, Q, M, D, L or constant	Data record number
LEN	Input	UINT	I, Q, M, D, L or constant	(hidden) Maximum length of the data record to be transferred in bytes
DONE	Output	BOOL	I, Q, M, D, L	Data record was transferred
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The writing process is not yet complete.
ERROR	Output	BOOL	I, Q, M, D, L	ERROR = 1: An error occurred during the writing process.
STATUS	Output	DWORD	I, Q, M, D, L	Block status or error information For interpretation of the STATUS parameter, see Parameter STATUS (Page 3638).
RECORD	InOut	VARIANT	I, Q, M, D, L	Data record

* For S7-1500 only.

** There is no implicit conversion in STL, which is why the range of valid data types may be limited. During programming in STL, please note the valid data types. These are displayed in the parameter tooltip.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

If you use "WRREC" to write a data record for PROFINET IO, negative values in the INDEX and LEN parameters will be interpreted as an unsigned 16-bit integer.

Parameter STATUS

For interpretation of the STATUS parameter, see Parameter STATUS (Page 3638).

Examples

You can find an example here: Use the program example of the parameter data record with WRREC & RDREC (Page 3598).

You can find another example here: Use the program example of the control data record with WRREC & RDREC (Page 3607).

You can find additional information and the program code for the examples here: Sample Library for Instructions (Page 2568).

See also

Overview of the types of data records (Page 3571)

Use the program example of the parameter data record with WRREC & RDREC (S7-1200, S7-1500)

Introduction

In the following example you read the parameter data record of a digital input module via RDREC and rewrite the data record via WRREC. When doing this you change the input delay.

You can find additional information on the data records in the Overview of the types of data records (Page 3571).

Requirement

Create the following tags for storing the data in a global data block:

- two tags for general use and seven tags for WRREC:

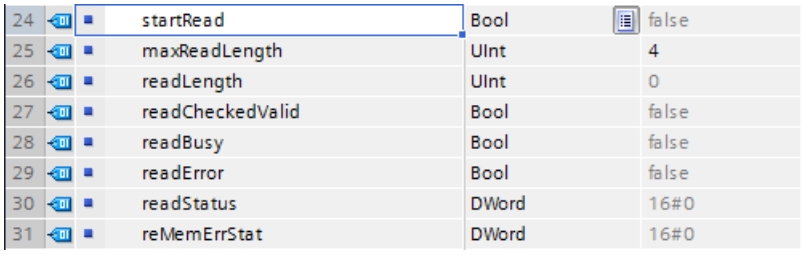
SLI_gDB_WRREC_Para

	Name	Data type	Start value
1	Static		
2	id	HW_IO	257
3	dataRecNbr	DInt	3
4	startWrite	Bool	false
5	maxWriteLength	UInt	4
6	writeCheckedValid	Bool	false
7	writeBusy	Bool	false
8	writeError	Bool	false
9	writeStatus	DWord	16#0
10	wrMemErrStat	DWord	16#0

- a structure for the data record at WRREC:

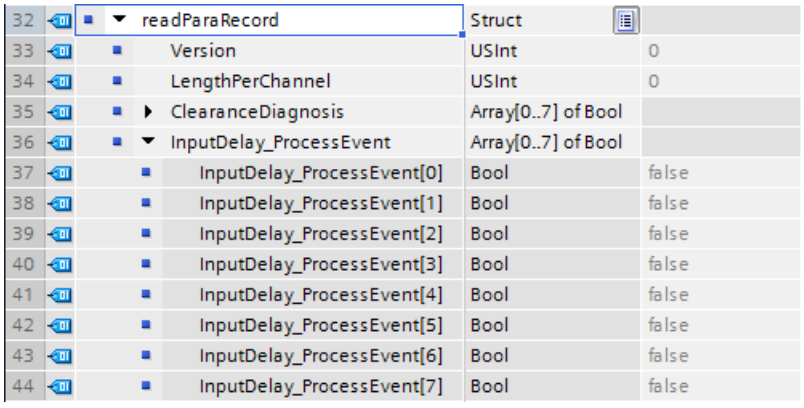
11	writeParaRecord	Struct	
12	Version	USInt	16
13	LengthPerChannel	USInt	2
14	ClearanceDiagnosis	Array[0..7] of Bool	
15	InputDelay_ProcessEvent	Array[0..7] of Bool	
16	InputDelay_ProcessEvent[0]	Bool	1
17	InputDelay_ProcessEvent[1]	Bool	false
18	InputDelay_ProcessEvent[2]	Bool	false
19	InputDelay_ProcessEvent[3]	Bool	false
20	InputDelay_ProcessEvent[4]	Bool	false
21	InputDelay_ProcessEvent[5]	Bool	false
22	InputDelay_ProcessEvent[6]	Bool	false
23	InputDelay_ProcessEvent[7]	Bool	false

- eight tags for RDREC:



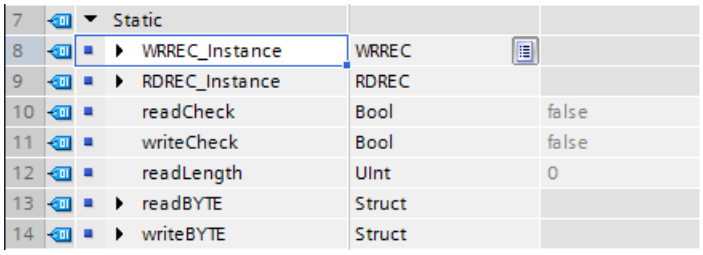
24	startRead	Bool	false
25	maxReadLength	UInt	4
26	readLength	UInt	0
27	readCheckedValid	Bool	false
28	readBusy	Bool	false
29	readError	Bool	false
30	readStatus	DWord	16#0
31	reMemErrStat	DWord	16#0

- a structure for the data record at RDREC:



32	readParaRecord	Struct	
33	Version	USInt	0
34	LengthPerChannel	USInt	0
35	ClearanceDiagnosis	Array[0..7] of Bool	
36	InputDelay_ProcessEvent	Array[0..7] of Bool	
37	InputDelay_ProcessEvent[0]	Bool	false
38	InputDelay_ProcessEvent[1]	Bool	false
39	InputDelay_ProcessEvent[2]	Bool	false
40	InputDelay_ProcessEvent[3]	Bool	false
41	InputDelay_ProcessEvent[4]	Bool	false
42	InputDelay_ProcessEvent[5]	Bool	false
43	InputDelay_ProcessEvent[6]	Bool	false
44	InputDelay_ProcessEvent[7]	Bool	false

Create the following local tags in an "SLI_FB_WRREC_Para" FB.



7	Static		
8	WRREC_Instance	WRREC	
9	RDREC_Instance	RDREC	
10	readCheck	Bool	false
11	writeCheck	Bool	false
12	readLength	UInt	0
13	readBYTE	Struct	
14	writeBYTE	Struct	

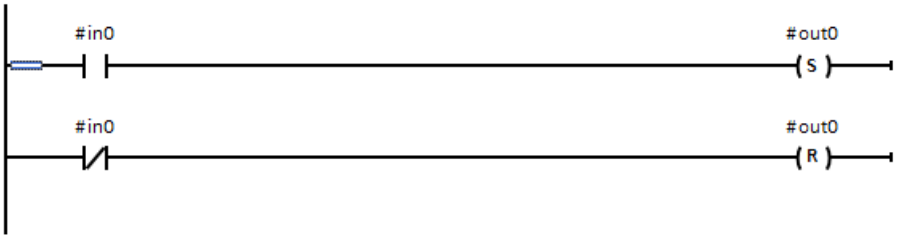
Interconnect parameters: "SLI_itemisePara_WRREC_Para" FC

To transfer the values of a bit sequence (0..7) create the FC "SLI_itemisePara_WRREC_Para".

Create the following local tags in it.

SLI_itemisePara_WRREC_Para		
	Name	Data type
1	Input	
2	in0	Bool
3	in1	Bool
4	in2	Bool
5	in3	Bool
6	in4	Bool
7	in5	Bool
8	in6	Bool
9	in7	Bool
10	Output	
11	out0	Bool
12	out1	Bool
13	out2	Bool
14	out3	Bool
15	out4	Bool
16	out5	Bool
17	out6	Bool
18	out7	Bool

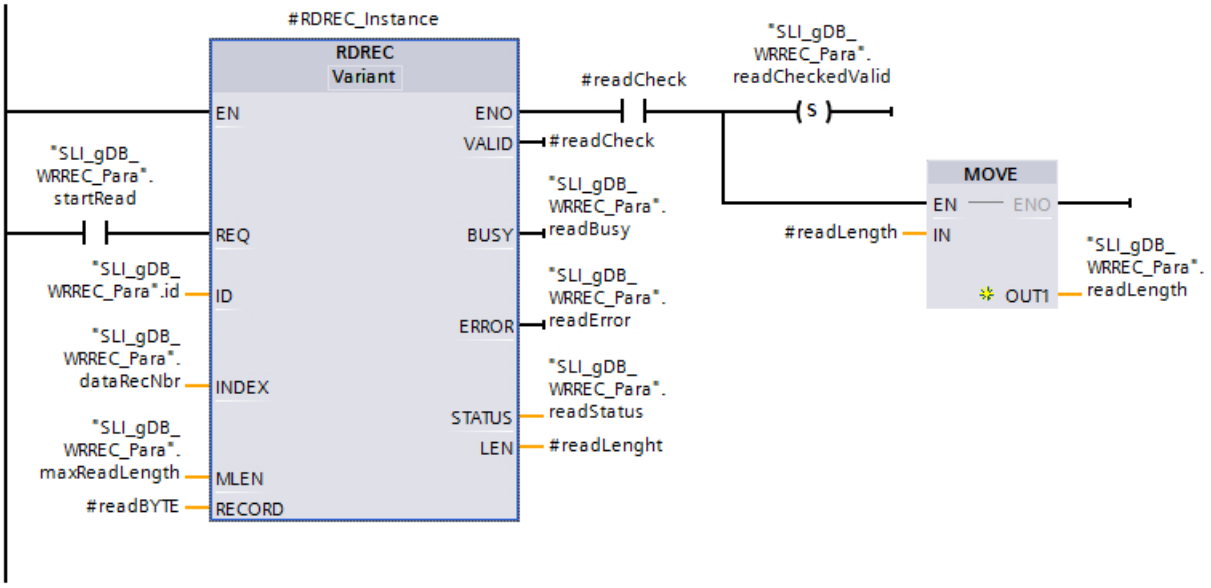
Network 1: According to the example of network 1, create interconnections for all bits (0..7).



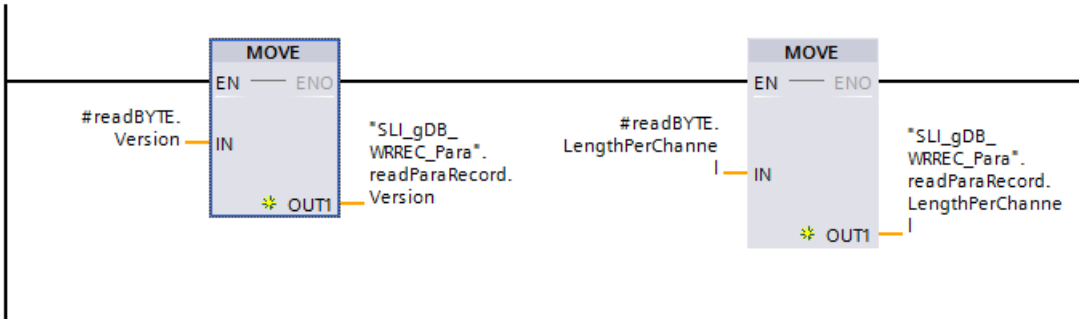
Interconnect parameters: "SLI_FB_WRREC_Para" FB

Create the following interconnections in the "SLI_FB_WRREC_Para" FB.

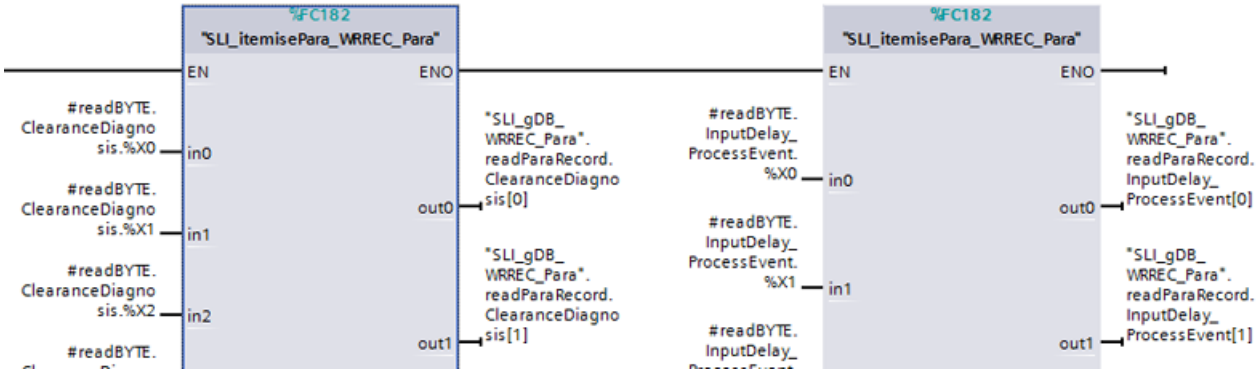
Network 1: Interconnect the parameters of the "RDREC" instruction as follows.



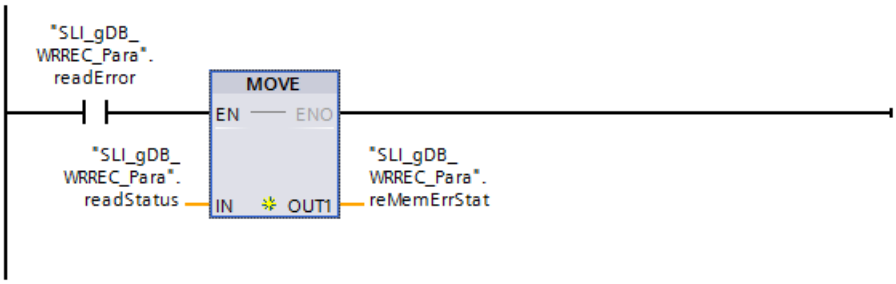
Network 2: Below interconnect all tags of the structure “#readBYTE“ with the structure “readParaRecord“. Part 1 of the network:



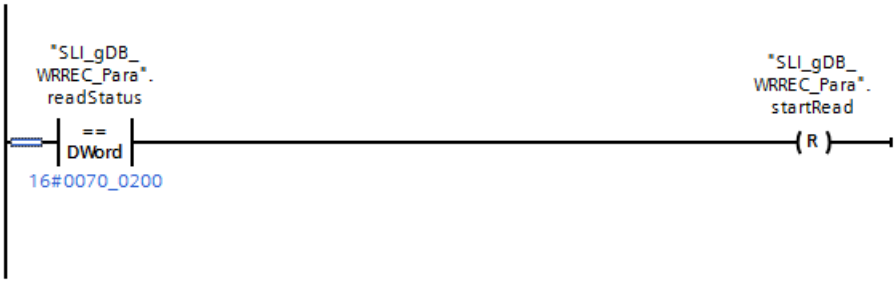
Part 2 of the network: Interconnect the tags "ClearanceDiagnosis" and "InputDelay_ProcessEvent" according to the example below.



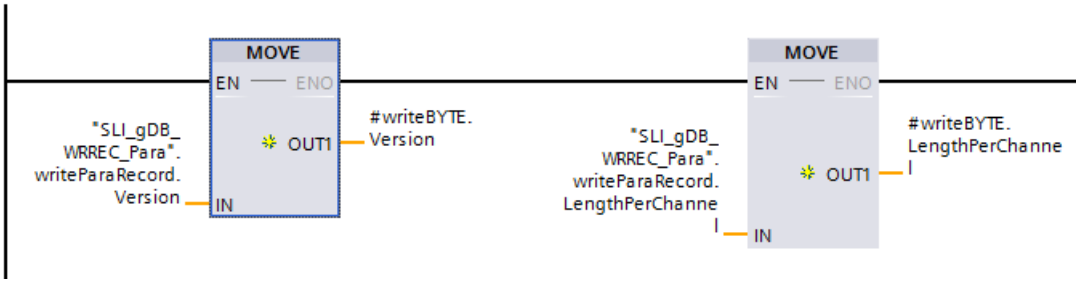
Network 3: Then interconnect the tags for the error situation of RDREC.



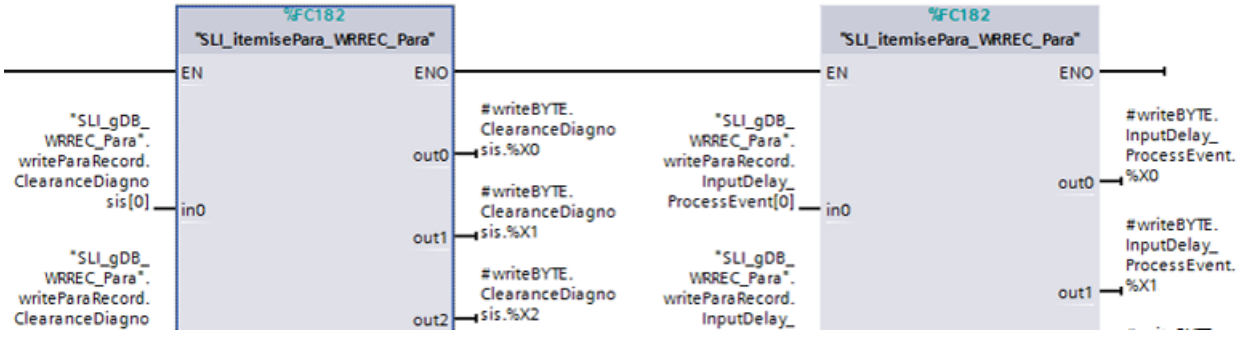
Network 4: Then interconnect the tags for ending RDREC.



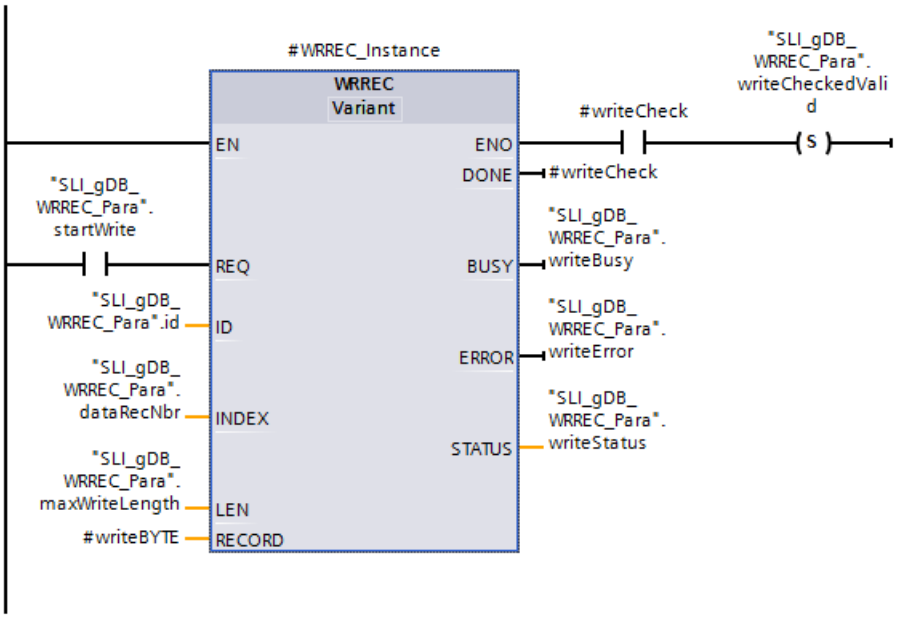
Network 5: Below interconnect all tags of the structure “writeParaRecord“ with the structure “#writeBYTE“. Part 1 of the network:



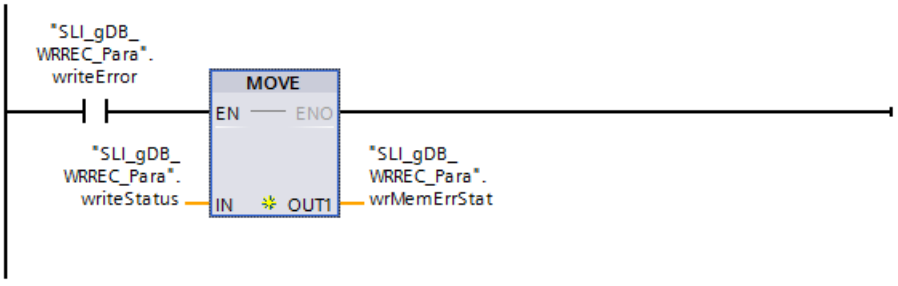
Part 2 of the network:



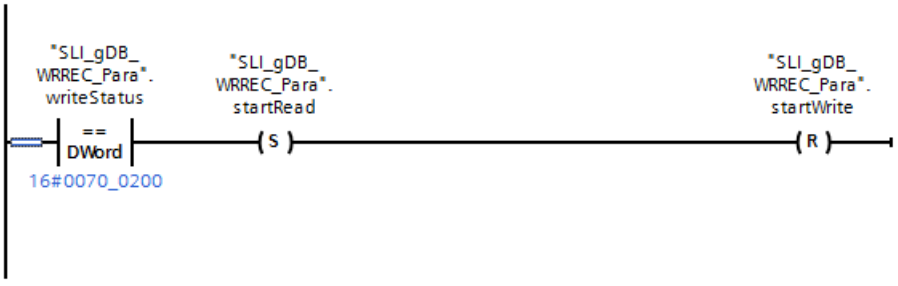
Network 6: Interconnect the parameters of the "WRREC" instruction as follows.



Network 7: Then interconnect the tags for the error situation of WRREC.



Network 8: Then interconnect the tags for ending WRREC and calling RDREC again.



Result of network 1:

You create an input module and read the hardware ID in the properties of the input module. You store the hardware identifier via the input parameter ID ("id"). Using the input parameter INDEX ("dataRecNbr") the data record with the number "3" of the module is called. The data record corresponds to the channel with the same name.

SLI_gDB_WRREC_Para

	Name	Data type	Start value	Monitor value
1	Static			
2	id	HW_IO	257	16#0101
3	dataRecNbr	DInt	3	3

When the input parameter REQ ("startRead") returns the signal state "TRUE", the "RDREC" instruction is started. Via the input parameter ID ("id") the instruction "RDREC" calls the input module.

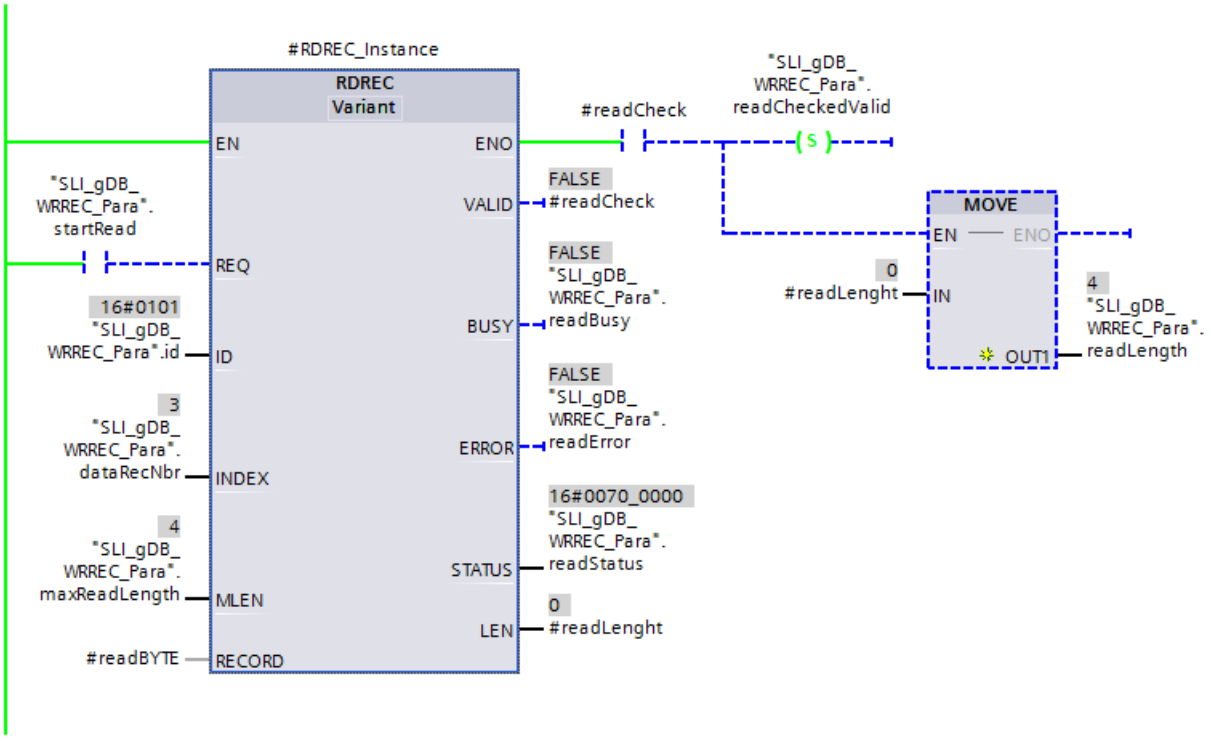
Over several calls the "RDREC" instruction reads out the data record and saves the data record using the parameter RECORD ("#readBYTE"). The length of the read data record is stored using the output parameter LEN ("#readLength"). According to the value of the input parameter MLEN ("maxReadLength"), the data record information that is to be read is limited to 4 bytes.

During processing the output parameter BUSY ("readBusy") is set to "TRUE" and VALID ("#readCheck") to "FALSE".

A successful reading result is output as follows when processing is complete.

- The output parameter VALID ("#readCheck") is set to "TRUE" and sets the tag "readCheckedValid" permanently to "TRUE".
- In addition to this the value of the local tag "#readLength" is stored permanently in the tag "readLength".

The block status is displayed at the output parameter STATUS ("readStatus"). Processing takes place without errors.



Result of network 2:

Using MOVE instructions and the FC SLI_itemisePara_WRREC_Para, the structure "#readBYTE" transfers the values of the tags to the tags of the structure "readParaRecord" with the same names. To improve understanding, the tags of the “readParaRecord“ structure are represented by descriptive names. The structure is described in the device manual of the module. For example "InputDelay_ProcessEvent" indicates an input delay of 3.2 ms. For the status of the channel (“ClearanceDiagnosis“) no disruption is output (bits 0 to 8 are “FALSE“.

32		readParaRecord	Struct		
33		Version	USInt	0	16
34		LengthPerChannel	USInt	0	2
35		ClearanceDiagnosis	Array[0..7] of Bool		
36		InputDelay_ProcessEvent	Array[0..7] of Bool		
37		InputDelay_ProcessEvent[0]	Bool	false	FALSE
38		InputDelay_ProcessEvent[1]	Bool	false	TRUE
39		InputDelay_ProcessEvent[2]	Bool	false	TRUE
40		InputDelay_ProcessEvent[3]	Bool	false	FALSE
41		InputDelay_ProcessEvent[4]	Bool	false	FALSE
42		InputDelay_ProcessEvent[5]	Bool	false	FALSE
43		InputDelay_ProcessEvent[6]	Bool	false	FALSE
44		InputDelay_ProcessEvent[7]	Bool	false	FALSE

Result of network 3:

When “readError“ supplies the signal status “TRUE“, the value of the tag “readStatus“ is stored in the tag “reMemErrStat“.

Result of network 4:

When the instruction “RDREC“ has been executed, “readStatus“ supplies the value “16#0070_0200“. After this a resetter sets the tag “startRead“ to “FALSE“

Result of network 5:

Using MOVE instructions and the FC SLI_itemisePara_WRREC_Para, the structure "writeParaRecord" transfers the values of the tags to the tags of the structure "#writeBYTE" with the same names. To improve understanding, the tags of the “writeParaRecord“ structure are represented by descriptive names.

Result of network 6:

When the input parameter REQ ("startWrite") returns the signal state "TRUE", the "WRREC" instruction is started.

The new data for the data record is stored at the RECORD (“#writeBYTE“) parameter. Over several calls, the instruction “WRREC“ writes the new data to the data record "3". According to the value of the input parameter LEN ("maxWriteLength"), the data record information that is to be written is limited to 4 bytes.

During processing the output parameter BUSY ("writeBusy") is set to "TRUE" and DONE ("#writeCheck") to "FALSE".

A successful writing result is output as follows when processing is complete.

- The output parameter DONE ("#writeCheck") is set to "TRUE".
- When "#writeCheck" supplies the signal state "TRUE" the tag "writeCheckedValid" is set permanently to "TRUE".

The block status is displayed at the output parameter STATUS ("writeStatus"). Processing takes place without errors.

4		startWrite	Bool	false	FALSE
5		maxWriteLength	UInt	4	4
6		writeCheckedValid	Bool	false	TRUE
7		writeBusy	Bool	false	FALSE
8		writeError	Bool	false	FALSE
9		writeStatus	DWord	16#0	16#0070_0000
10		wrMemErrStat	DWord	16#0	16#0000_0000

Result of network 7:

When “writeError“ supplies the signal status “TRUE“, the value of the tag “writeStatus“ is stored in the tag “wrMemErrStat“.

Result of network 8:

When the instruction “WRREC“ has been executed, “writeStatus“ supplies the value “16#0070_0200“. After this the resetter sets the tag “startWrite“ to “FALSE“ The tag “startRead“ is set to “TRUE“ after which the instruction “RDREC“ is executed again. The structure “readParaRecord“ shows the newly written values of the data record. Since in “InputDelay_ProcessEvent“ the bit 0 supplies the signal state “TRUE“, the channel 3 now has an input delay off 0.1 ms.

32	readParaRecord	Struct		
33	Version	USInt	0	16
34	LengthPerChannel	USInt	0	2
35	ClearanceDiagnosis	Array[0..7] of Bool		
36	InputDelay_ProcessEvent	Array[0..7] of Bool		
37	InputDelay_ProcessEvent[0]	Bool	false	TRUE
38	InputDelay_ProcessEvent[1]	Bool	false	FALSE
39	InputDelay_ProcessEvent[2]	Bool	false	FALSE
40	InputDelay_ProcessEvent[3]	Bool	false	FALSE
41	InputDelay_ProcessEvent[4]	Bool	false	FALSE
42	InputDelay_ProcessEvent[5]	Bool	false	FALSE
43	InputDelay_ProcessEvent[6]	Bool	false	FALSE
44	InputDelay_ProcessEvent[7]	Bool	false	FALSE

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

WRREC: Write data record (Page 3596)

RDREC: Read data record (Page 3578)

Use the program example of the control data record with WRREC & RDREC (S7-1200, S7-1500)

Introduction

In the following example, you write the control data record 196 using WRREC and check this by reading the data record out with RDREC. For the configuration select the central setup of an S7-1500

You can find additional information on the data records in Overview of the types of data records (Page 3571).

Requirement

Note

Enabling configuration control

To be able to write the control data record, the configuration control (option handling) of the CPU must be enabled. You will find these in “CPU properties > Configuration control”.

Create the following tags for storing the data in a global data block:

- three tags for general use and six for WRREC:

SLI_gDB_WRREC_Control

	Name	Data type	Start value
1	Static		
2	id	HW_IO	33
3	dataRecNbr	DInt	196
4	maxLength	UInt	0
5	configDone	Bool	false
6	checkedValid_WR	Bool	false
7	busyWR	Bool	false
8	errorWR	Bool	false
9	statusWR	DWord	16#0
10	memErrStatusWR	DWord	16#0

- a structure for the data at WRREC (data record 196):

11	recConfigCtrl_WR	Struct	
12	BlockLength	USInt	8
13	BlockID	USInt	196
14	Version	USInt	4
15	Subversion	USInt	0
16	Slot_0	USInt	0
17	Slot_1	USInt	1
18	Slot_2	USInt	2
19	Slot_3	USInt	3

The data record is structured as follows:

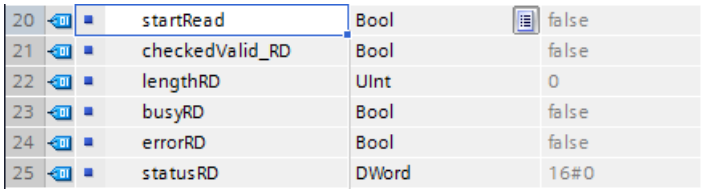
– "BlockLength" indicates the length of the data record in Byte.

– "BlockID" indicates the data record number.

– "Version" indicates the type of structure. A central structure is used for the example. In a central structure "Subversion" has the value "0".

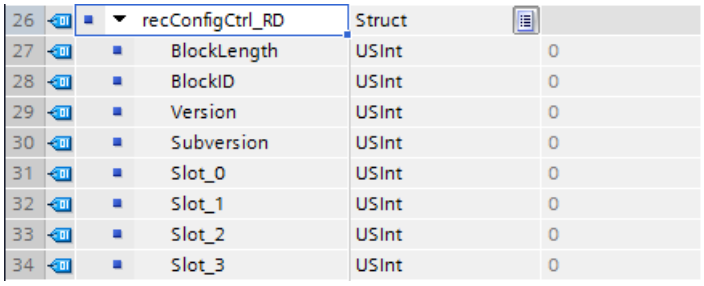
– "Slot_0" to "Slot_.." correspond to the slots used. The values of the slots indicate the numbering to be used.

- six tags for RDREC:



20		startRead	Bool	false
21		checkedValid_RD	Bool	false
22		lengthRD	UInt	0
23		busyRD	Bool	false
24		errorRD	Bool	false
25		statusRD	DWord	16#0

- a structure for the data at RDREC:



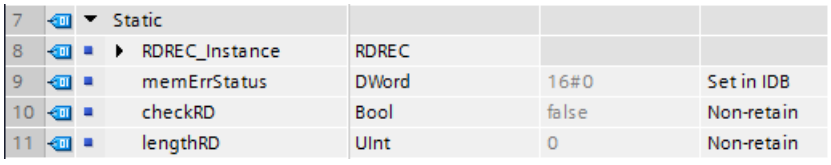
26		recConfigCtrl_RD	Struct	
27		BlockLength	USInt	0
28		BlockID	USInt	0
29		Version	USInt	0
30		Subversion	USInt	0
31		Slot_0	USInt	0
32		Slot_1	USInt	0
33		Slot_2	USInt	0
34		Slot_3	USInt	0

The instruction “WRREC“ for writing the control data recording must be called in a startup OB. In the startup OB create the following local tags for passing on the data.



4		Temp	
5		checkWR	Bool
6		statusWR	DWord

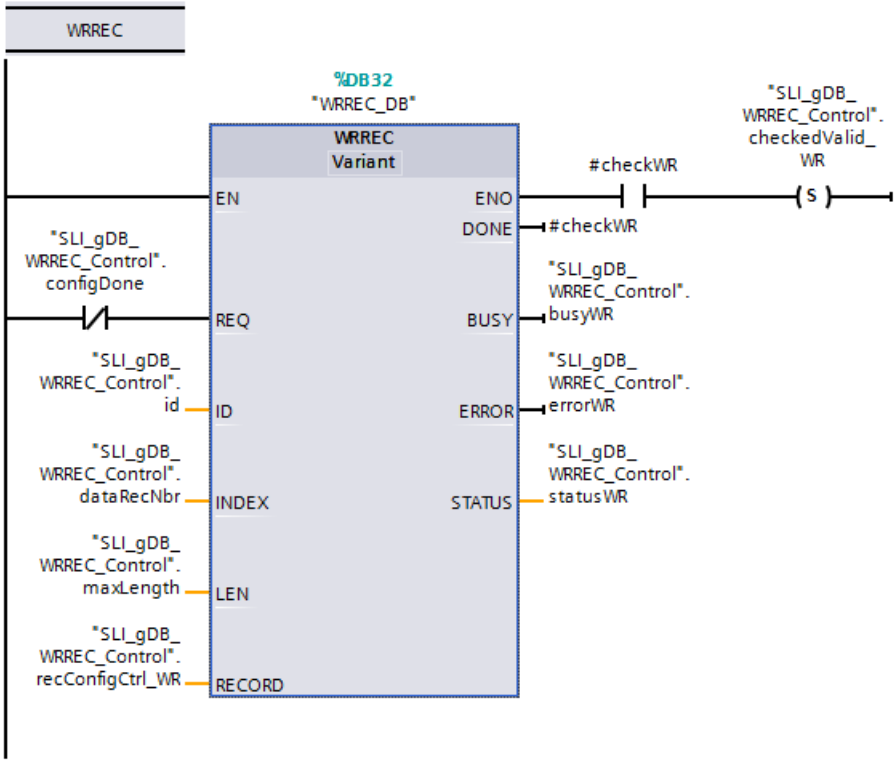
Create the following local tags in an FB. Call the FB in a program cycle OB (OB1).



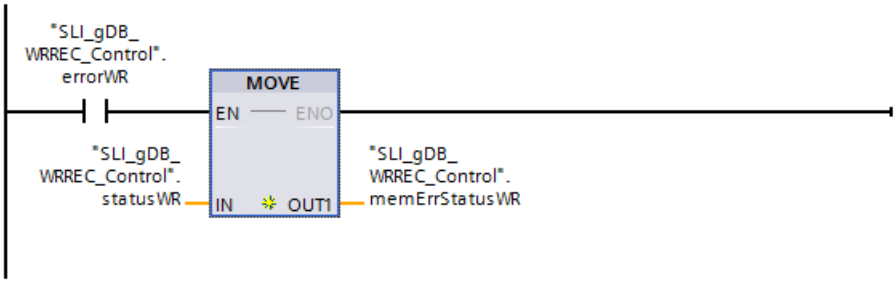
7		Static			
8		RDREC_Instance	RDREC		
9		memErrStatus	DWord	16#0	Set in IDB
10		checkRD	Bool	false	Non-retain
11		lengthRD	UInt	0	Non-retain

Interconnecting parameters in the startup OB

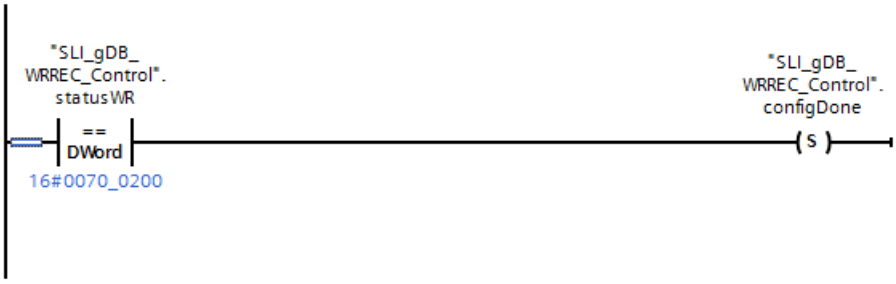
Network 1: Interconnect the parameters of the "WRREC" instruction as follows.



Network 2: Then interconnect the parameters for the error situation of WRREC.



Network 3: Then interconnect the parameters for ending WRREC.

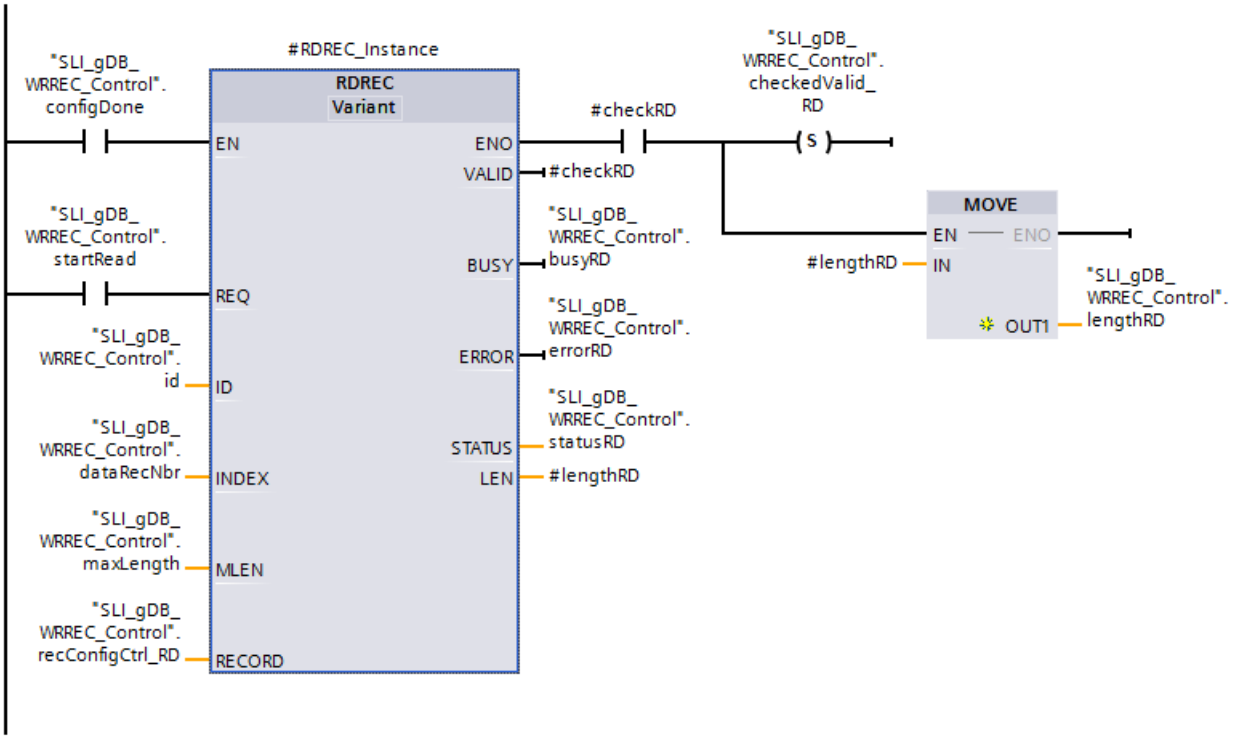


Network 4: This is followed by interconnecting the parameters for requesting the repeated call of WRREC.

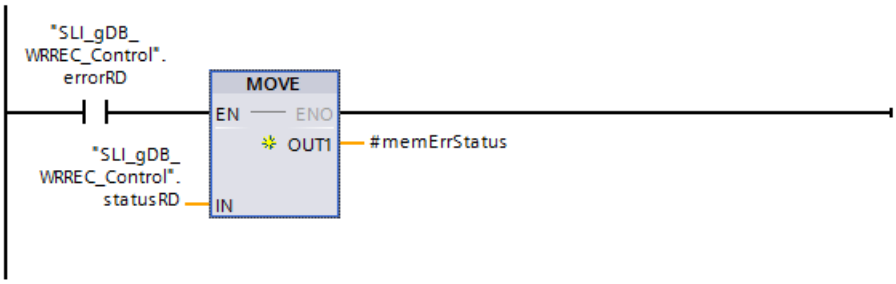


Interconnect parameters in the FB

Network 1: Interconnect the parameters of the "RDREC" instruction as follows.



Network 2: Then interconnect the parameters for the error situation of RDREC.



Network 3: Then interconnect the parameters for ending RDREC.



General parameters

You store the hardware identifier of the configuration of the CPU (“33“) via the input parameter ID (“id“). You store the control data record ("196") using the input parameter INDEX ("dataRecNbr"). According to the value of the tag maxLength" (with the value "0") the amount of data record information to be taken into account is unlimited.

Note

Loading in the CPU and processing of the program in startup.

1. To load the program, set the CPU to "STOP" mode.
2. Set the project to "Online" mode.
3. Set the CPU to "RUN" mode. The "Startup" mode is worked through.

In "Startup” mode the CPU works through the program sections located in the startup OB.

Startup OB: Result of network 1

Since the input parameter REQ supplies the signal state "TRUE", the "WRREC" instruction is started. The normally closed contact (“configDone“) has the signal state "FALSE".

The data for the control data record is stored at the RECORD (“recConfigCtrl_WR“) parameter. To improve understanding, the tags of the “recConfigCtrl_WR“ structure are represented by clear names and have the same structure as described in the system manual..

11	recConfigCtrl_WR	Struct		
12	BlockLength	USInt	8	8
13	BlockID	USInt	196	196
14	Version	USInt	4	4
15	Subversion	USInt	0	0
16	Slot_0	USInt	0	0
17	Slot_1	USInt	1	1
18	Slot_2	USInt	2	2
19	Slot_3	USInt	3	3

Over several calls, the instruction “WRREC“ writes the new data to the control data record. During processing of WRREC the output parameter BUSY ("busyWR") is set to "TRUE" and DONE ("#checkWR") to "FALSE".

A successful writing result is output as follows when processing is complete.

- The output parameter DONE ("#checkWR") is set to "TRUE" and sets the tag "checkedValid_WR" permanently to "TRUE".

The block status is displayed at the output parameter STATUS ("statusWR"). Processing takes place without errors.

SLI_gDB_WRREC_Control

	Name	Data type	Start value	Monitor value
1	Static			
2	id	HW_IO	33	16#0021
3	dataRecNbr	DInt	196	196
4	maxLength	UInt	0	0
5	configDone	Bool	false	TRUE
6	checkedValid_WR	Bool	false	TRUE
7	busyWR	Bool	false	FALSE
8	errorWR	Bool	false	FALSE
9	statusWR	DWord	16#0	16#0000_0000
10	memErrStatusWR	DWord	16#0	16#0000_0000

Startup OB: Result of network 2

If an error occurs when processing WRREC:

When “errorWR“ supplies the signal status “TRUE“, the value of the tag “statusWR“ is stored in the tag “memErrStatusWR“.

Startup OB: Result of network 3

When the instruction “WRREC“ has been executed, “statusWR“ supplies the value “16#0070_0200“. The tag "configDone" is then set to "TRUE". This ends the processing of WRREC.

Startup OB: Result of network 4

As long as the processing of the "WRREC" instruction has not yet ended ("configDone" has the value"FALSE" and "busyWR" has the value "TRUE") processing returns to network 1.

FB: Result of network 1

After processing the program sections located in the startup OB the CPU is in the “RUN“ mode.

When the input parameter EN ("configDone") supplies the signal state "TRUE", the "RDREC" instruction is released for processing. When the input parameter REQ ("startRead") returns the signal state "TRUE", the "RDREC" instruction is started.

Over several calls the "RDREC" instruction reads out the control data record and saves the data record using the parameter RECORD ("recConfigCtrl_RD").

26		recConfigCtrl_RD	Struct		
27		BlockLength	USInt	0	8
28		BlockID	USInt	0	196
29		Version	USInt	0	4
30		Subversion	USInt	0	0
31		Slot_0	USInt	0	0
32		Slot_1	USInt	0	1
33		Slot_2	USInt	0	2
34		Slot_3	USInt	0	3

The length of the read data record is stored using the output parameter LEN ("#readLength"). During processing of RDREC the output parameter BUSY ("busyRD") is set to "TRUE" and VALID ("#checkRD") to "FALSE".

A successful reading result is output as follows when processing is complete.

- The output parameter VALID ("#checkRD") is set to "TRUE". #checkRD then sets the tag "checkedValid_RD" permanently to "TRUE".
- In addition to this the value of the local tag "#lengthRD" is stored permanently in the tag "lengthRD".

The block status is displayed at the output parameter STATUS ("statusRD"). Processing takes place without errors.

20		startRead	Bool	false	FALSE
21		checkedValid_RD	Bool	false	TRUE
22		lengthRD	UInt	0	8
23		busyRD	Bool	false	FALSE
24		errorRD	Bool	false	FALSE
25		statusRD	DWord	16#0	16#0070_0000

FB: Result of network 2

If an error occurs when processing RDREC:

When “errorRD“ supplies the signal status “TRUE“, the value of the tag “statusRD“ is stored in the local tag “#memErrStatus“.

FB: Result of network 3

When the instruction “RDREC“ has been executed and has checked the data, “#checkRD“ supplies the value “TRUE“. The tag "startRead" is then set to "FALSE". Execution of the “RDREC“ instruction is then ended.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

WRREC: Write data record (Page 3596)

RDREC: Read data record (Page 3578)

GETIO: Read process image (S7-1200, S7-1500)

Description

You use the instruction "GETIO" to consistently read out all inputs of a DP standard slave/ PROFINET IO device. The instruction "GETIO" calls the instruction "DPRD_DAT (Page 3700)". If there was no error during the data transmission, the data that has been read is entered in the destination area indicated by INPUTS .

The destination area must have the same length that you configured for the selected component.

If you read from a DP standard slave with a modular configuration or with several DP identifiers, you only access the data of one component/DP identifier at the configured start address with a "GETIO" call.

Parameter

The following table shows the parameters of the "GETIO" instruction:

Parameter	Declaration	Data type	Memory area	Description
ID	Input	HW_SUB-MODULE	I, Q, M, D, L or constant	Hardware ID of the DP standard slave / PROFINET IO device.
STATUS	Output	DWORD	I, Q, M, D, L	Contains the error information of "DPRD_DAT (Page 3700)" in the form DW#16#40xxxx00.
LEN	Output	INT	I, Q, M, D, L	Amount of data read in bytes
INPUTS	InOut	VARIANT	I, Q, M, D, L	Destination area for the read data. It must have the same length as the range that you configured for the selected DP standard slave / PROFINET IO device. Only the data type BYTE is valid.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

See also: DPRD_DAT: Read consistent data of a DP standard slave (Page 3700)

Example

You will find the example here: Program example for GETIO & SETIO (Page 3617).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

SETIO: Transfer process image (S7-1200, S7-1500)

Description

Transfer the data using the SETIO instruction. The source area of the transfer is defined using the OUTPUTS parameter. The data is transferred in a consistent state to the addressed DP standard slave
/ PROFINET IO device. If you have configured the relevant address area of the DP standard slave / PROFINET IO device as a consistent range in a process image, the data is transferred to the process image. "SETIO" calls the "DPWR_DAT (Page 3702)" instruction during this transfer.

The source range must have the same length that you configured for the selected component.

In the case of a DP standard slave / PROFINET IO device with modular configuration or with several DP identifiers, you can only access one DP identifier / component per "SETIO" call.

Parameter

The following table shows the parameters of the "SETIO" instruction:

Parameter	Declaration	Data type	Memory area	Description
ID	Input	HW_SUBMOD-ULE	I, Q, M, D, L or con-stant	Hardware ID of the DP standard slave / PROFINET IO device.
STATUS	Output	DWORD	I, Q, M, D, L	Contains the error information of "DPWR_DAT (Page 3702)" in the form DW#16#40xxxx00.
OUTPUTS	InOut	VARIANT	I, Q, M, D, L	Source range for the data to be written. It must have the same length as the range that you configured for the selected DP standard slave / PROFINET IO device. Only the data type BYTE is valid.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

See also: DPWR_DAT: Write consistent data of a DP standard slave (Page 3702)

Example

You will find the example here: Program example for GETIO & SETIO (Page 3617).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

Program example for GETIO & SETIO (S7-1200, S7-1500)

Introduction

In the following example, you read out the channels of a digital input module via GETIO and write new values to the channels of a digital output module via SETIO. You then read out the changes via GETIO and via the digital input module.

Requirements

- An IO device/DP slave is created with a digital input module and a digital output module.

  Note
  Change with SETIO, read out changes with GETIO

  The digital input module must be connected to the digital output module in order that the written value can be read via SETIO using GETIO

- To store the data, create nine tags and the two structures "readinputs" and "writeOutputs" in a global data block.
  Use the tag "id_Read" to store the hardware identifier of the digital input module. Use the tag "id_Write" to store the hardware identifier of the digital output module.

  Note
  Find hardware identifier

  Use "PLC tags > System constants" to find the hardware identifier. For example, for the digital input module, search for "Local~<Name of the DI module>", with the data type "HW_Submodule". The associated cell "value" contains the hardware identifier.

SLI_gDB_GETIO

	Name	Data type	Start value
1	Static		
2	id_Read	HW_SUBMODULE	257
3	executeRead	Bool	false
4	readInputs	Struct	
5	Channel_0-7	Array[0..7] of Bool	
6	Channel_8-15	Array[0..7] of Bool	
7	Channel_16-23	Array[0..7] of Bool	
8	Channel_24-31	Array[0..7] of Bool	
9	statusRead	DWord	16#0
10	lengthRead	Int	0
11	doneRead	Bool	false
12	id_Write	HW_SUBMODULE	258
13	executeWrite	Bool	false
14	writeOutputs	Struct	
15	Channel_0-7	Array[0..7] of Bool	
16	Channel_8-15	Array[0..7] of Bool	
17	Channel_16-23	Array[0..7] of Bool	
18	Channel_24-31	Array[0..7] of Bool	
19	statusWrite	DWord	16#0
20	doneWrite	Bool	false

- To forward the data, create the following local tags in an "SLI_FB_GETIO_SETIO" FB.

7	Static			
8	GETIO_Instance	GETIO		
9	SETIO_Instance	SETIO		
10	readBYTE	Struct		Non-retain
11	Channel_0-7	Byte	16#0	Non-retain
12	Channel_8-15	Byte	16#0	Non-retain
13	Channel_16-23	Byte	16#0	Non-retain
14	Channel_24-31	Byte	16#0	Non-retain
15	writeBYTE	Struct		Non-retain
16	Channel_0-7	Byte	16#0	Non-retain
17	Channel_8-15	Byte	16#0	Non-retain
18	Channel_16-23	Byte	16#0	Non-retain
19	Channel_24-31	Byte	16#0	Non-retain

- To change the channel states, set the following bits (channel 1 and 2) in the global data block to "TRUE":

14	writeOutputs	Struct	
15	Channel_0-7	Array[0..7] of Bool	
16	Channel_0-7[0]	Bool	false
17	Channel_0-7[1]	Bool	1
18	Channel_0-7[2]	Bool	1
19	Channel_0-7[3]	Bool	false
20	Channel_0-7[4]	Bool	false
21	Channel_0-7[5]	Bool	false
22	Channel_0-7[6]	Bool	false
23	Channel_0-7[7]	Bool	false

Interconnect parameters: "SLI_itemiseByte_gsIO" FC

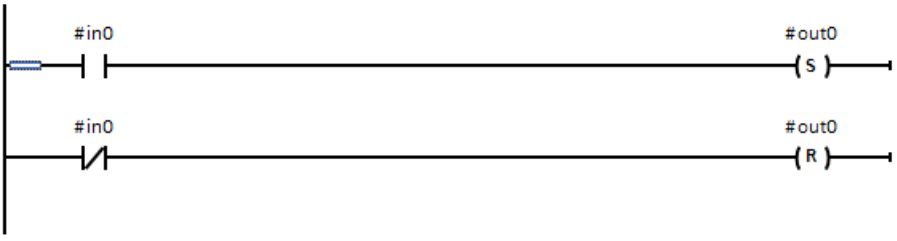
To transfer the values of a bit sequence (0..7) create the FC "SLI_itemiseByte_gsIO".

Create the following local tags in it.

SLI_itemiseByte_gsIO

		Name	Data type
1		Input	
2		in0	Bool
3		in1	Bool
4		in2	Bool
5		in3	Bool
6		in4	Bool
7		in5	Bool
8		in6	Bool
9		in7	Bool
10		Output	
11		out0	Bool
12		out1	Bool
13		out2	Bool
14		out3	Bool
15		out4	Bool
16		out5	Bool
17		out6	Bool
18		out7	Bool

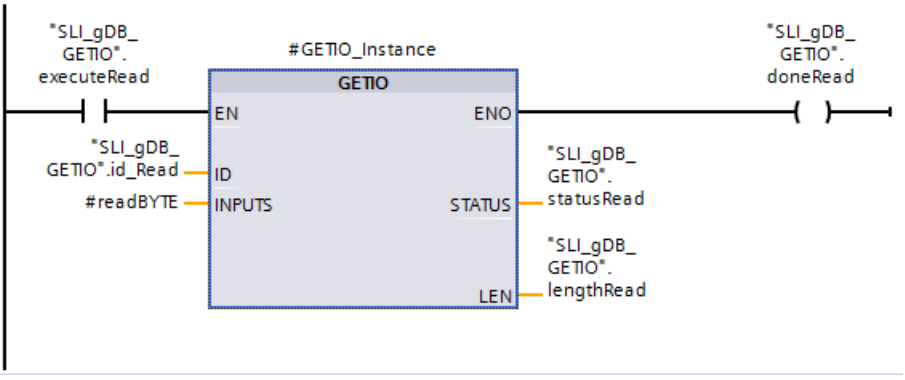
Network 1: According to the example of network 1, create interconnections for all bits (0..7).



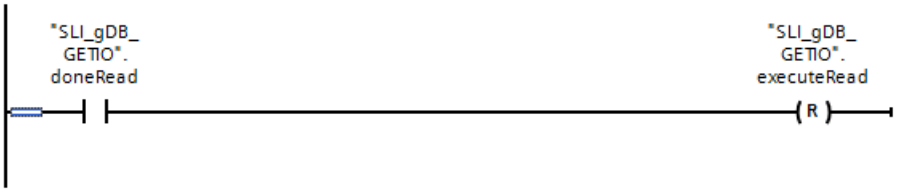
Interconnect parameters: "SLI_FB_GETIO_SETIO" FB

Create the following interconnections in the "SLI_FB_GETIO_SETIO" FB. On the IO device/ DP slave, then call the FB "SLI_FB_GETIO_SETIO" in a program cycle OB (OB1).

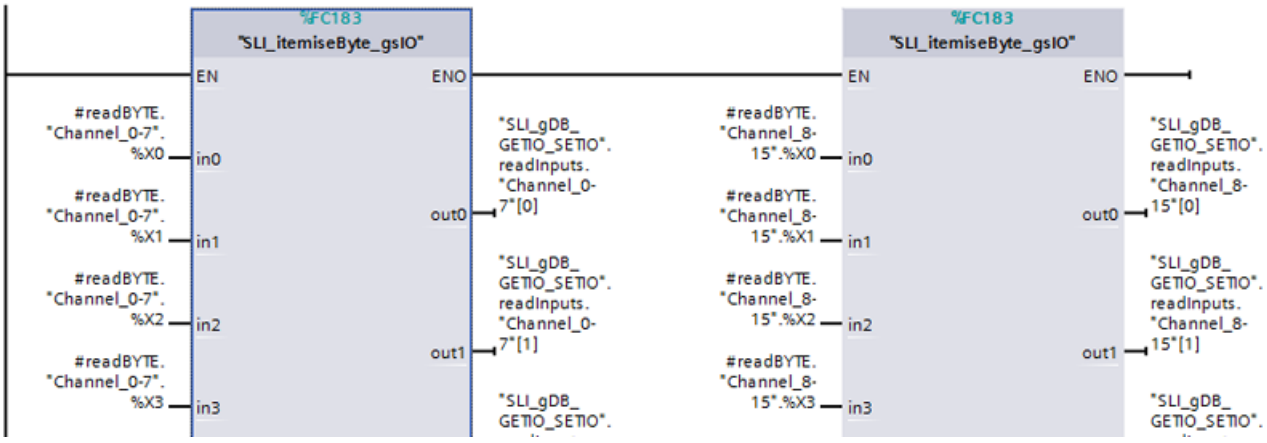
Network 1: Interconnect the parameters of the "GETIO" instruction as follows.



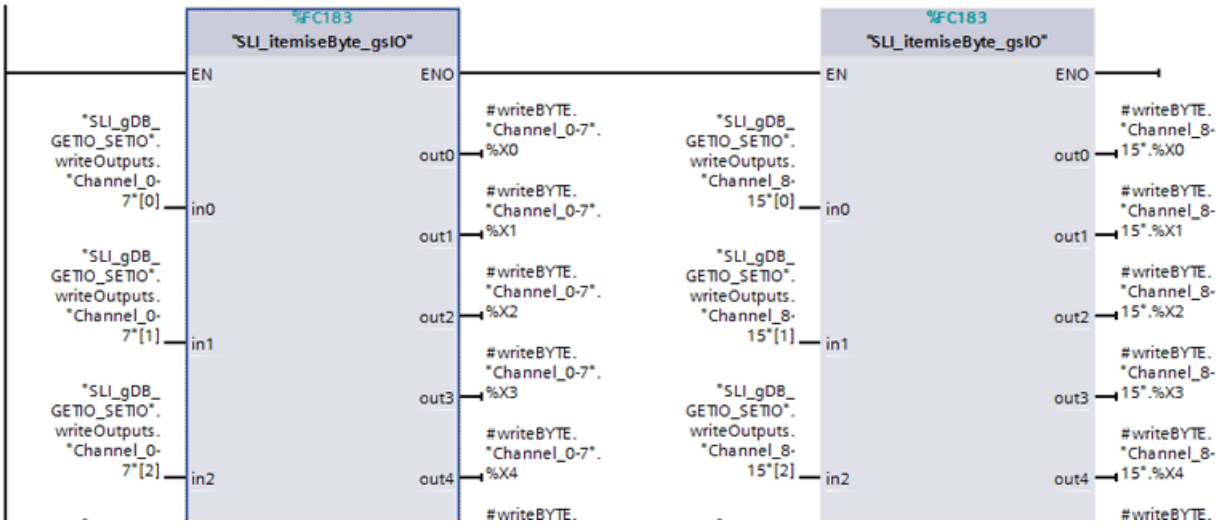
Network 2: Close the instruction "GETIO" as follows.



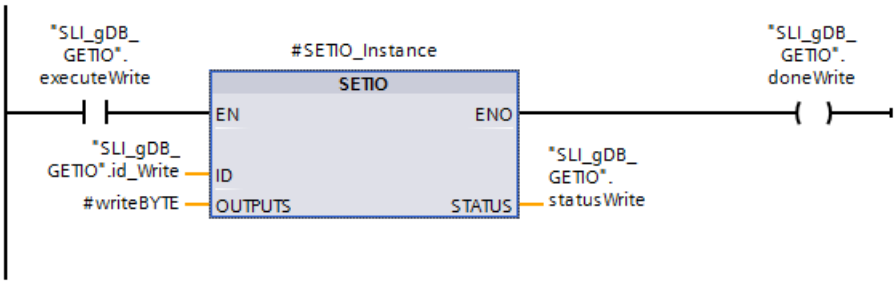
Network 3: Create the interconnections according to the following example to convert the channel values of GETIO. All inputs and outputs (0..7) of the "SLI_itemiseByte_gsIO" FC are interconnected in this case.



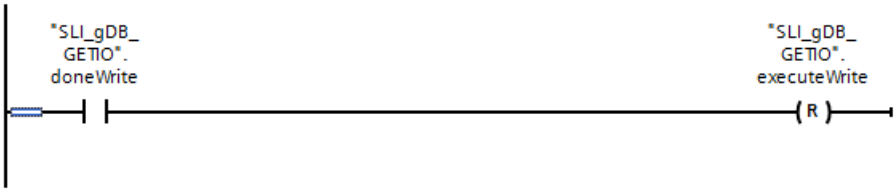
Network 4: Create the interconnections according to the following example to convert the channel values of SETIO. All inputs and outputs (0..7) of the "SLI_itemiseByte_gsIO" FC are interconnected in this case.



Network 5: Interconnect the parameters of the "SETIO" instruction as follows.



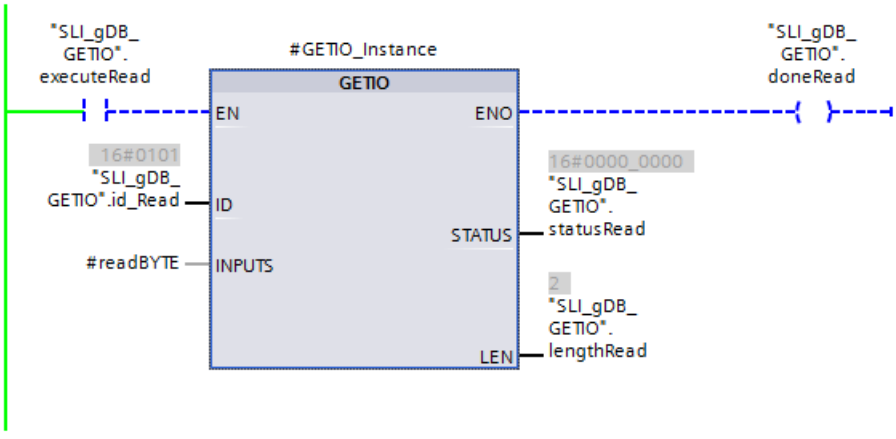
Network 6: Close the instruction "SETIO" as follows.



Result of network 1 (GETIO)

When the input parameter EN ("executeRead") returns the signal state "TRUE", the "GETIO" instruction is started. The instruction "GETIO" calls the input module via the input parameter ID ("id_Read"). The channels of the input module are read out via the input parameter INPUTS ("#readBYTE"). The length of the read channels in BYTE is recorded at the output parameter LEN ("lengthRead"). The value "2" references 16 input channels.

The block status is displayed at the output parameter STATUS ("statusRead"). Processing takes place without errors.



Result of network 2 (GETIO)

When the instruction "GETIO" has been executed, the output parameter ENO ("doneRead") returns the signal state "TRUE". After this, the resetter sets the tag "executeRead" to "FALSE" and ends the processing of "GETIO".

Result of network 3 (GETIO)

Using MOVE instructions and the FC SLI_itemiseByte_gsIO, the structure "#readBYTE" transfers the values of the tags to the tags of the structure "readInputs". The values of the channels 0 to 7 and 8 to 15 are hereby converted from BYTE to array of BOOL.

To improve understanding, the tags of the "readInputs" structure are represented by descriptive names.

Result of network 4 (SETIO)

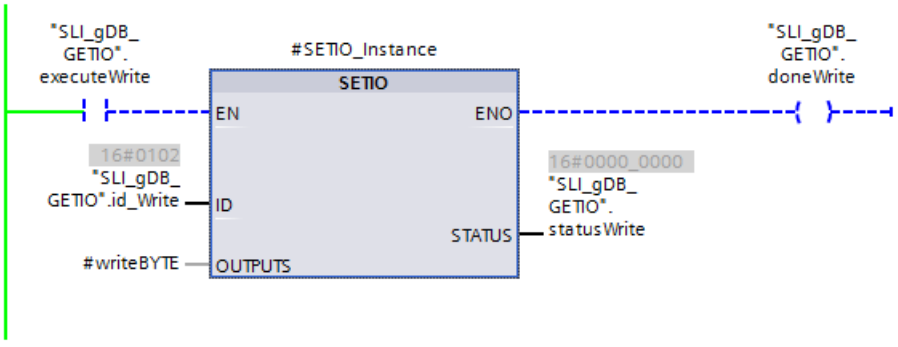
Using MOVE instructions and the FC SLI_itemiseByte_gsIO, the structure "writeOutputs" transfers the values of the tags to the tags of the structure "#writeBYTE". The values of the channels 0 to 7 and 8 to 15 are converted from array of BOOL to BYTE.

To improve understanding, the tags of the "writeOutputs" structure are represented by descriptive names.

Result of network 5 (SETIO)

When the input parameter EN ("executeWrite") returns the signal state "TRUE", the "SETIO" instruction is started. The instruction "SETIO" calls the output module via the input parameter ID ("id_Write"). New values are written to the channels of the output module via the input parameter OUTPUTS ("#writeBYTE").

The block status is displayed at the output parameter STATUS ("statusWrite"). Processing takes place without errors.



Result of network 6 (SETIO)

When the instruction "SETIO" has been executed, the output parameter ENO ("doneWrite") returns the signal state "TRUE". After this, the resetter sets the tag "executeWrite" to "FALSE" and ends the processing of "SETIO".

Result of reprocessing of GETIO

If the tag "executeRead" is reset to "TRUE", the structure "readInputs" shows the new values on the channels of the input module after the execution of GETIO. The channels 1 and 2 ("Channel_0-7[1]..[2]") now return the signal state "TRUE".

4	readInputs	Struct		
5	Channel_0-7	Array[0..7] of Bool		
6	Channel_0-7[0]	Bool	false	FALSE
7	Channel_0-7[1]	Bool	false	TRUE
8	Channel_0-7[2]	Bool	false	TRUE
9	Channel_0-7[3]	Bool	false	FALSE
10	Channel_0-7[4]	Bool	false	FALSE
11	Channel_0-7[5]	Bool	false	FALSE
12	Channel_0-7[6]	Bool	false	FALSE
13	Channel_0-7[7]	Bool	false	FALSE
14	Channel_8-15	Array[0..7] of Bool		
15	Channel_16-23	Array[0..7] of Bool		
16	Channel_24-31	Array[0..7] of Bool		

Note

Changing the signal state of a channel

If a value is already assigned to a channel on which a signal state is to be changed (for example by a PLC tag), this pre-assignment can make a change by SETIO invalid. Make sure that the signal state of a channel to be changed is not already determined by other factors.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

GETIO: Read process image (Page 3615)

SETIO: Transfer process image (Page 3616)

GETIO_PART: Read process image area (S7-1200, S7-1500)

Description

You use the instruction "GETIO_PART" to consistently read out a related part of the inputs of an IO module. "GETIO_PART" calls the instruction "DPRD_DAT (Page 3700)".

Use the ID input parameter to select the IO module by means of the hardware ID.

You use the OFFSET and LEN parameters to specify the portion of the process image area to be read. If the input area spanned by OFFSET and LEN is not completely covered by the module, the block returns the error code DW#16#40**80B7**00.

The length of the destination area must be larger than or equal to the amount of bytes to be read:

- If there was no error during the data transmission, ERROR receives the value FALSE. The data that is read is written to the destination area defined at the INPUTS parameter.
- If there was an error during the data transmission, ERROR receives the value TRUE. The STATUS parameter receives the error information from "DPRD_DAT".
- If the destination area is greater than LEN, then the first LEN bytes of the destination area will be written. ERROR receives the value FALSE.

Parameter

The following table shows the parameters of the "GETIO_PART" instruction:

Parameter	Declaration	Data type	Memory area	Description
ID	Input	HW_SUBMOD-ULE	I, Q, M, D, L or constant	Hardware identifier of the module
OFFSET	Input	INT	I, Q, M, D, L or constant	Number of the first byte to be read in the process image for the component (smallest possible value: 0).
LEN	Input	INT	I, Q, M, D, L or constant	Number of bytes to be read.
STATUS	Output	DWORD	I, Q, M, D, L	Contains the error information of "DPRD_DAT" in the form DW#16#40xxxx00, if ERROR = TRUE.
ERROR	Output	BOOL	I, Q, M, D, L	Error display: ERROR = TRUE if an error occurs when "DPRD_DAT" is called.
INPUTS	InOut	VARIANT	I, Q, M, D, L	Destination area for read data: If the destination area is greater than LEN, the first LEN bytes of the destination area are written.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

See "RET_VAL" parameter of the "DPRD_DAT (Page 3700)" instruction.

Example

You will find the example here: Program example for GETIO_PART & SETIO_PART (Page 3627).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

SETIO_PART: Transfer process image area (S7-1200, S7-1500)

Description

You can use the "SETIO_PART" instruction to consistently write data from the source area spanned by OUTPUTS to the outputs of an IO module. "SETIO_PART" calls the instruction "DPWR_DAT (Page 3702)".

With the input parameter ID you select the I/O module based on the hardware identified. With the parameters OFFSET and LEN you specify the portion of the process image area to be written for the component addressed by ID. If the output area spanned by OFFSET and LEN is not completely covered by the module, the block returns the error code DW#16#40**80B7**00.

The length of the destination area must be larger than or equal to the amount of bytes to be read:

- If there was no error during the data transmission, ERROR receives the value FALSE.
- If there was an error during the data transmission, ERROR receives the value TRUE, and STATUS receives the error information of "DPWR_DAT".
- If the source area is greater than LEN, the first LEN bytes from OUTPUTS are transferred. ERROR receives the value FALSE.

Parameters

The following table shows the parameters of the "SETIO_PART" instruction:

Parameter	Declaration	Data type	Memory area	Description
ID	Input	HW_SUB-MODULE	I, Q, M, D, L or constant	Hardware identifier of the IO module.
OFFSET	Input	INT	I, Q, M, D, L or constant	Number of the first byte to be written in the process image for the component (smallest possible value: 0).
LEN	Input	INT	I, Q, M, D, L or constant	Number of bytes to be written.
STATUS	Output	DWORD	I, Q, M, D, L	Contains the error information of "DPWR_DAT" in the form DW#16#40xxxx00, if ERROR = TRUE.
ERROR	Output	BOOL	I, Q, M, D, L	Error display: ERROR = TRUE if an error occurs when "DPWR_DAT" is called.
OUTPUTS	InOut	VARIANT	I, Q, M, D, L	Source range for the data to be written: If the source area is greater than LEN, the first LEN bytes are transferred from OUT-PUTS.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

ERROR and STATUS parameters

See instruction "DPWR_DAT (Page 3702)".

Example

You will find the example here: Program example for GETIO_PART & SETIO_PART (Page 3627).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

Program example for GETIO_PART & SETIO_PART (S7-1200, S7-1500)

Introduction

In the following example, you read out the channels of a digital input module via GETIO_PART and write new values to some of the channels of a digital output module via SETIO_PART. You then read out the changes via GETIO_PART and via the digital input module.

Requirements

- An IO device/DP slave is created with a digital input module and a digital output module.

  Note
  Change with SETIO_PART, read out changes with GETIO_PART

  The digital input module must be connected to the digital output module in order that the written value can be read via SETIO_PART using GETIO_PART

- To store the data, create 14 tags and the two structures "readinputs" and "writeOutputs" in a global data block.
  Use the tag "id_Read" to store the hardware identifier of the digital input module. Use the tag "id_Write" to store the hardware identifier of the digital output module.

  Note
  Find hardware identifier

  Use "PLC tags > System constants" to find the hardware identifier. For example, for the digital input module, search for "Local~<Name of the DI module>", with the data type "HW_Submodule". The associated cell "value" contains the hardware identifier.

SLI_gDB_GETIO_SETIO_PART

	Name	Data type	Start value
1	Static		
2	id_Read	HW_SUBMODULE	257
3	executeRead	Bool	false
4	firstByteRead	Int	0
5	maxLengthRead	Int	2
6	readInputs	Struct	
7	Channel_0-7	Array[0..7] of Bool	
8	Channel_8-15	Array[0..7] of Bool	
9	Channel_16-23	Array[0..7] of Bool	
10	Channel_24-31	Array[0..7] of Bool	
11	errorRead	Bool	false
12	statusRead	DWord	16#0
13	doneRead	Bool	false
14	id_Write	HW_SUBMODULE	258
15	executeWrite	Bool	false
16	firstByteWrite	Int	0
17	maxLengthWrite	Int	1
18	writeOutputs	Struct	
19	Channel_0-7	Array[0..7] of Bool	
20	Channel_8-15	Array[0..7] of Bool	
21	Channel_16-23	Array[0..7] of Bool	
22	Channel_24-31	Array[0..7] of Bool	
23	errorWrite	Bool	false
24	statusWrite	DWord	16#0
25	doneWrite	Bool	false

- To change the channel states, set the following bits (for BYTE 0 and BYTE 1) to "TRUE" in the global data block:

14	writeOutputs	Struct	
15	Channel_0-7	Array[0..7] of Bool	
16	Channel_0-7[0]	Bool	false
17	Channel_0-7[1]	Bool	1
18	Channel_0-7[2]	Bool	1
19	Channel_0-7[3]	Bool	false
20	Channel_0-7[4]	Bool	false
21	Channel_0-7[5]	Bool	false
22	Channel_0-7[6]	Bool	false
23	Channel_0-7[7]	Bool	false
24	Channel_8-15	Array[0..7] of Bool	
25	Channel_8-15[0]	Bool	false
26	Channel_8-15[1]	Bool	1

- To forward the data, create the following local tags in an "SLI_FB_GETIO_SETIO_PART" FB.

7	Static			
8	GETIO_PART_Instance	GETIO_PART		
9	SETIO_PART_Instance	SETIO_PART		
10	readBYTE	Struct		Non-retain
11	writeBYTE	Struct		Non-retain

Interconnect parameters: "SLI_itemiseByte_gpspIO" FC

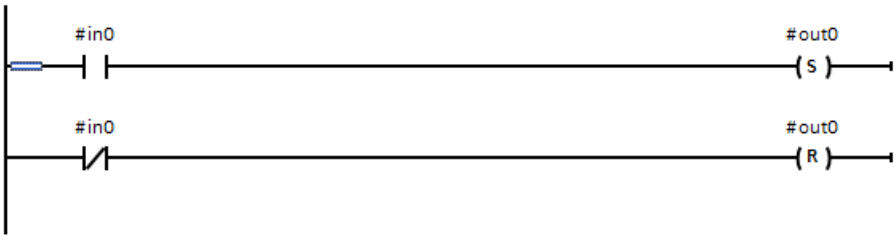
To transfer the values of a bit sequence (0..7) create the FC "SLI_itemiseByte_gpspIO".

Create the following local tags in it.

SLI_itemiseByte_gpspIO

	Name	Data type
1	Input	
2	in0	Bool
3	in1	Bool
4	in2	Bool
5	in3	Bool
6	in4	Bool
7	in5	Bool
8	in6	Bool
9	in7	Bool
10	Output	
11	out0	Bool
12	out1	Bool
13	out2	Bool
14	out3	Bool
15	out4	Bool
16	out5	Bool
17	out6	Bool
18	out7	Bool

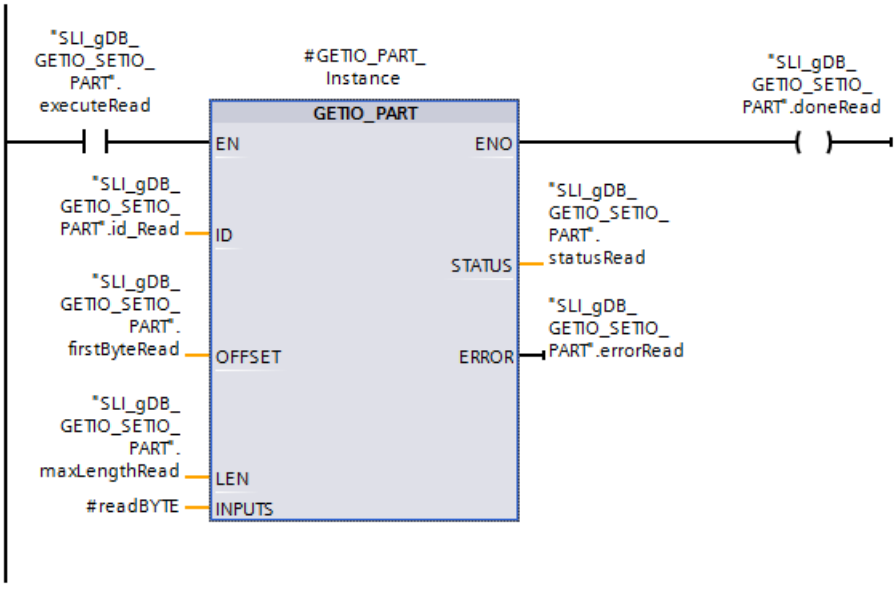
Network 1: According to the example of network 1, create interconnections for all bits (0..7).



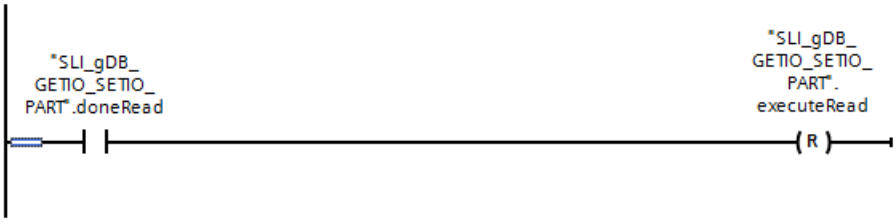
Interconnect parameters: "SLI_FB_GETIO_SETIO_PART" FB

Create the following interconnections in the "SLI_FB_GETIO_SETIO_PART" FB. On the IO device/DP slave, then call the FB "SLI_FB_GETIO_SETIO_PART" in a program cycle OB (OB1).

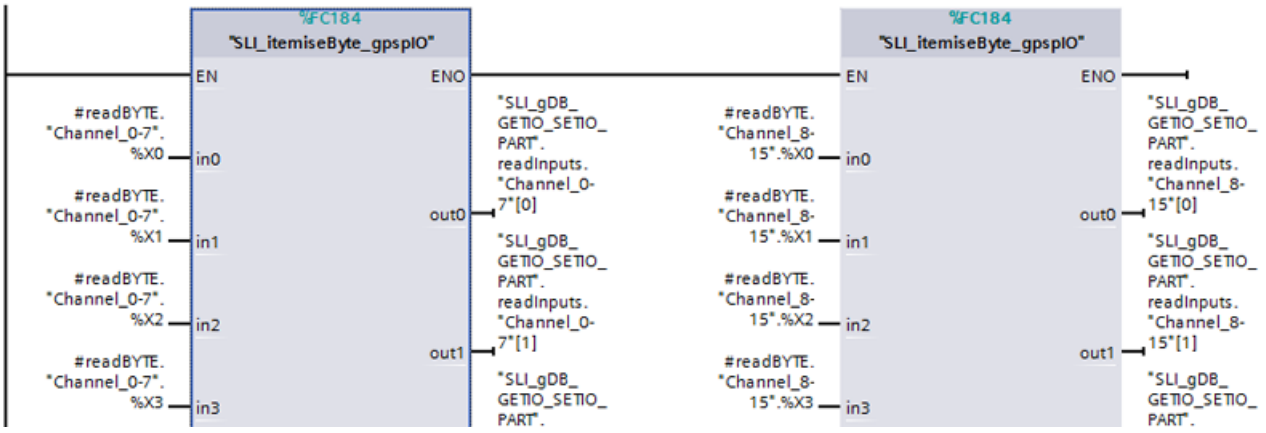
Network 1: Interconnect the parameters of the "GETIO_PART" instruction as follows.



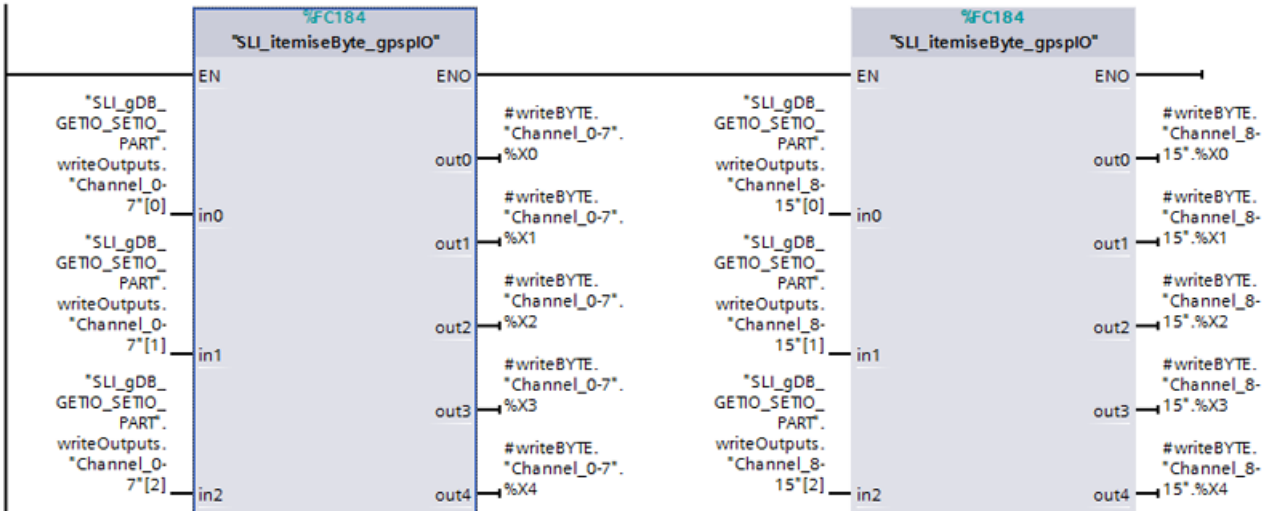
Network 2: Close the instruction "GETIO_PART" as follows.



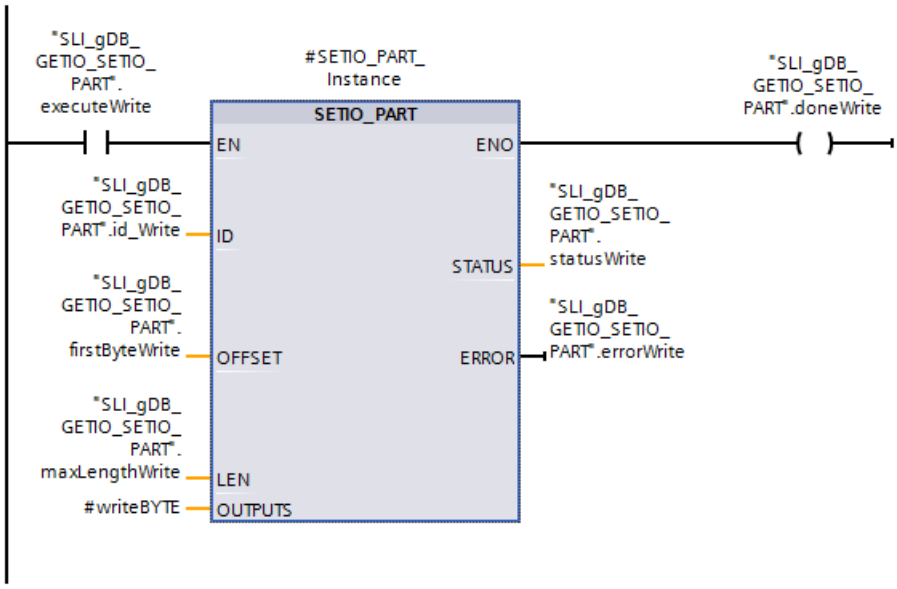
Network 3: Create the interconnections according to the following example to convert the channel values of GETIO_PART. All inputs and outputs (0..7) of the "SLI_itemiseByte_gpspIO" FC are interconnected in this case.



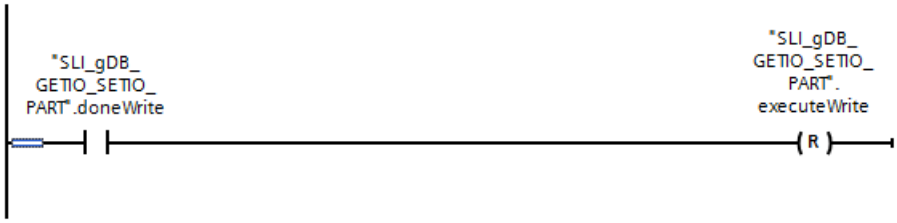
Network 4: Create the interconnections according to the following example to convert the channel values of SETIO_PART. All inputs and outputs (0..7) of the "SLI_itemiseByte_gpspIO" FC are interconnected in this case.



Network 5: Interconnect the parameters of the "SETIO_PART" instruction as follows.



Network 6: Close the instruction "SETIO_PART" as follows.

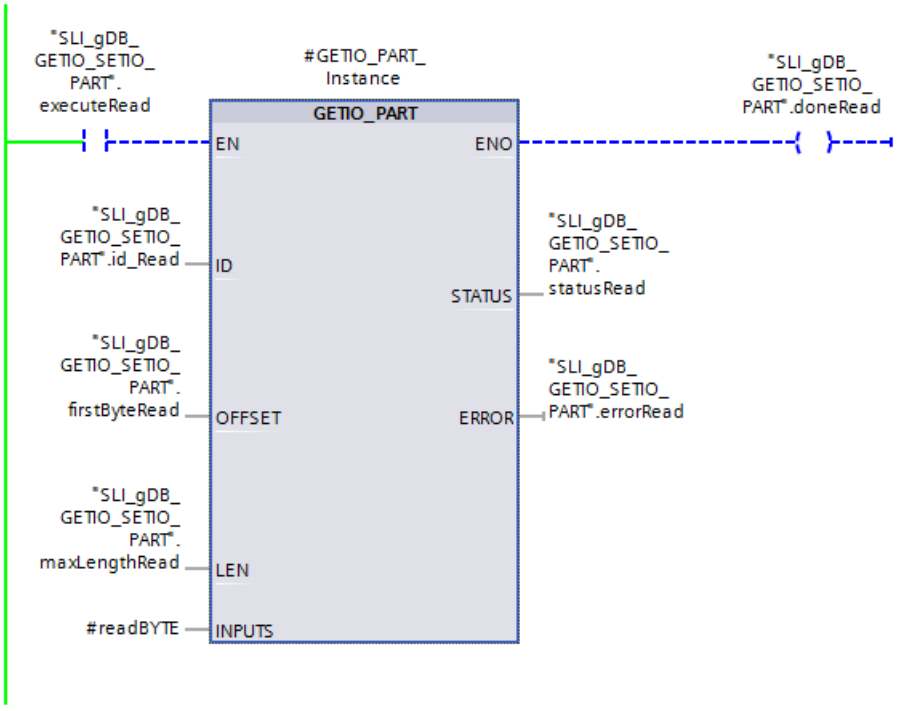


Result of network 1 (GETIO_PART)

When the input parameter EN ("executeRead") returns the signal state "TRUE", the "GETIO_PART" instruction is started. The instruction "GETIO_PART" calls the input module via the input parameter ID ("id_Read"). The channels of the input module are read out via the input parameter INPUTS ("#readBYTE").

Through the input parameter OFFSET ("firstByteRead") with the value "0", the instruction "GETIO_PART" starts to read channels of the input module starting from the "BYTE 0". Two BYTE (16 channels) are read out according to the value "2" of the input parameter LEN ("maxLengthRead"). In this example, this value corresponds to the channels 0 to 15.

The block status is displayed at the output parameter STATUS ("statusRead") and ERROR ("errorRead"). Processing takes place without errors.



Result of network 2 (GETIO_PART)

When the instruction "GETIO_PART" has been executed, the output parameter ENO ("doneRead") returns the signal state "TRUE". After this, the resetter sets the tag "executeRead" to "FALSE" and ends the processing of "GETIO_PART".

SLI_gDB_GETIO_SETIO_PART

	Name	Data type	Start value	Monitor value
1	Static			
2	id_Read	HW_SUBMODULE	257	16#0101
3	executeRead	Bool	false	FALSE
4	firstByteRead	Int	0	0
5	maxLengthRead	Int	2	2
6	readInputs	Struct		
7	errorRead	Bool	false	FALSE
8	statusRead	DWord	16#0	16#0000_0000
9	doneRead	Bool	false	FALSE

Result of network 3 (GETIO_PART)

Using MOVE instructions and the FC SLI_itemiseByte_gpspIO, the structure "#readBYTE" transfers the values of the tags to the tags of the structure "readInputs". The values of the channels 0 to 7 and 8 to 15 are hereby converted from BYTE to array of BOOL.

To improve understanding, the tags of the "readInputs" structure are represented by descriptive names.

Result of network 4 (SETIO_PART)

Using MOVE instructions and the FC SLI_itemiseByte_gpspIO, the structure "writeOutputs" transfers the values of the tags to the tags of the structure "#writeBYTE". The values of the channels 0 to 7 and 8 to 15 are converted from array of BOOL to BYTE.

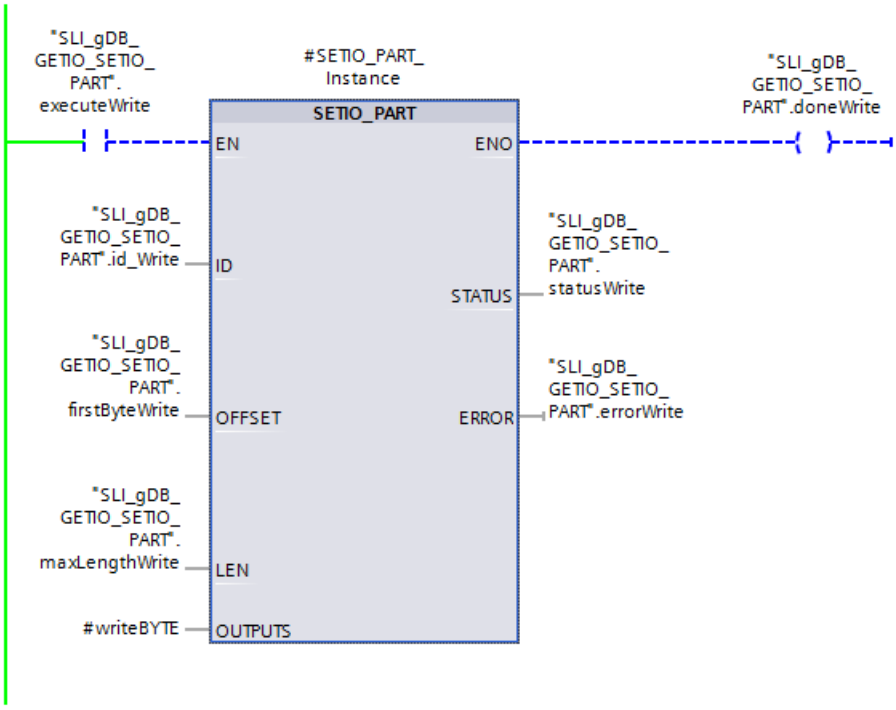
To improve understanding, the tags of the "writeOutputs" structure are represented by descriptive names.

Result of network 5 (SETIO_PART)

When the input parameter EN ("executeWrite") returns the signal state "TRUE", the "SETIO_PART" instruction is started. The instruction "SETIO_PART" calls the output module via the input parameter ID ("id_Write"). New values are written to the channels of the output module via the input parameter OUTPUTS ("#writeBYTE").

Through the input parameter OFFSET ("firstByteWrite") with the value "0", the instruction "SETIO_PART" starts to write to channels of the output module starting from the "BYTE 0". Only one BYTE (8 channels) is written according to the value "1" of the input parameter LEN ("maxLengthWrite"). In this example, this value corresponds to the channels 0 to 7. The channel 9 set to the value "TRUE" (see "writeOutputs", "Channel_8-15[1]") is thus not taken into consideration during writing to the channels of the output module.

The block status is displayed at the output parameter STATUS ("statusWrite") and ERROR ("errorWrite"). Processing takes place without errors.



Result of network 6 (SETIO_PART)

When the instruction "SETIO_PART" has been executed, the output parameter ENO ("doneWrite") returns the signal state "TRUE". After this, the resetter sets the tag "executeWrite" to "FALSE" and ends the processing of "SETIO_PART".

10	id_Write	HW_SUBMODULE	258	16#0102
11	executeWrite	Bool	false	FALSE
12	firstByteWrite	Int	0	0
13	maxLengthWrite	Int	1	1
14	writeOutputs	Struct		
15	errorWrite	Bool	false	FALSE
16	statusWrite	DWord	16#0	16#0000_0000
17	doneWrite	Bool	false	FALSE

Result of reprocessing of GETIO_PART

If the tag "executeRead" is reset to "TRUE", the structure "readInputs" shows the new values on the channels of the input module after the execution of GETIO_PART. A change affects only the channels 0 to 7 according to the values of the tags "firstByteWrite" and "maxLengthWrite". The channels 1 and 2 ("Channel_0-7[1]..[2]") now return the signal state "TRUE". The signal state of channel 9 ("Channel_8-15[1]") is unchanged.

6	readInputs	Struct		
7	Channel_0-7	Array[0..7] of Bool		
8	Channel_0-7[0]	Bool	false	FALSE
9	Channel_0-7[1]	Bool	false	TRUE
10	Channel_0-7[2]	Bool	false	TRUE
11	Channel_0-7[3]	Bool	false	FALSE
12	Channel_0-7[4]	Bool	false	FALSE
13	Channel_0-7[5]	Bool	false	FALSE
14	Channel_0-7[6]	Bool	false	FALSE
15	Channel_0-7[7]	Bool	false	FALSE
16	Channel_8-15	Array[0..7] of Bool		
17	Channel_8-15[0]	Bool	false	FALSE
18	Channel_8-15[1]	Bool	false	FALSE

Note

Changing the signal state of a channel

If a value is already assigned to a channel on which a signal state is to be changed (for example by a PLC tag), this pre-assignment can make a change by SETIO_PART invalid. Make sure that the signal state of a channel to be changed is not already determined by other factors.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

GETIO_PART: Read process image area (Page 3624)

SETIO_PART: Transfer process image area (Page 3626)

RALRM: Receive interrupt (S7-1200, S7-1500)

Description RALRM (S7-1200, S7-1500)

Description

The instruction receives an interrupt with all corresponding information from an I/O module (centralized configuration) or from a module of a DP slave or PROFINET IO device. This information is made available at its output parameters.

The information in the output parameters contains the start information of the called OB as well as information of the interrupt source.

The configuration of the data structure of the target range AINFO corresponds to that for a central configuration of the PROFINET IO data structure.

Call "RALRM" only within the interrupt OB started by the CPU operating system as a result of the I/O interrupt that is to be examined.

Note

If you call "RALRM" in an OB whose start event is not an I/O interrupt, the instruction will provide correspondingly reduced information in its outputs.
Make sure to use different instance DBs when you call "RALRM" in different OBs. If you evaluate data resulting from an "RALRM" call outside of the associated interrupt OB, use a separate instance DB per OB start event.

Note

The interface of the "RALRM" instruction is identical to that in the "RALRM" FB defined in the standard "PROFIBUS and PROFINET Guideline Communication Function Blocks on PROFIBUS DP and PROFINET IO".

Calling RALRM

RALRM" can be called in three operating modes (MODE parameter). These are explained in the table below.

MODE	RALRM ...
0	... shows the component that triggered the interrupt in the output parameter ID and sets the output parameter NEW to TRUE.
1	... writes all output parameters, regardless of the component triggering the interrupt.
2	... checks whether the component specified in the F_ID input parameter has triggered the interrupt. • If not, NEW = FALSE • If yes, NEW = TRUE and all other output parameters are written.

Parameters

The following table shows the parameters of the "RALRM" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	INT	I, Q, M, D, L or constant	Mode
F_ID	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the module. The number is assigned automatically and is stored in the hardware configuration properties of the component or interface.
MLEN	Input	UINT	I, Q, M, D, L or constant	Maximum length of the interrupt information to be received in bytes. With MLEN = 0, all data specified at the AINFO parameter are read.
NEW	Output	BOOL	I, Q, M, D, L	A new interrupt was received.
STATUS (Page 3638)	Output	DWORD	I, Q, M, D, L	Error code
ID	Output	HW_IO	I, Q, M, D, L	Hardware identifier of the module from which the interrupt was received.
LEN	Output	UINT	I, Q, M, D, L	Length of the received interrupt information
TINFO (Page 3642)	InOut	VARIANT	D, L	Target range for OB start and management information
AINFO (Page 3659)	InOut	VARIANT	I, Q, M, D, L	Target range for header information and additional interrupt information For AINFO , provide a length of at least MLEN bytes.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

If you select a target range (Page 3667)TINFO or AINFO that is too short, RALRM does not enter the full information.

Diagnostics for Windows applications

It may be possible that Windows can no longer send a signal to the CPU in case of a bluescreen.

For additional information, call the instruction "RALRM" (SFB54) with the diagnostic interrupt (OB82).

The "RALRM" instruction is called with the following input parameters:

Input	Type	Value
MODE	INT	2
F_ID	HW_IO	59 (Host Interface RID)
MLEN	UINT	0
TINFO	VARIANT	Address for each area of 54 bytes
AINFO	VARIANT	Address for each area of 54 bytes

Bytes 50 and 51 can be used by the "AINFO" parameter to receive additional information:

- 0xA1: Bluescreen
- 0xA2: Windows running
- 0xA3: Windows not running

Example

You can find the example here: Program example for RALRM (Page 3668).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

Parameter STATUS (S7-1200, S7-1500)

Description

The STATUS output parameter contains error information. If it is interpreted as ARRAY[1...4] of BYTE, the error information has the following structure:

Field element	Name	Meaning
STATUS[1]	Function_Num	• B#16#00, if no error • Function ID from DPV1-PDU: In the event of an error, B#16#80 is output (in the event of an error reading a data record B#16#DE and writing a data record B#16#DF). If no DPV1 protocol element is used, then B#16#C0 will be output.
STATUS[2]	Error_Decode	Location of the error ID
STATUS[3]	Error_Code_1	Error ID
STATUS[4]	Error_Code_2	Manufacturer-specific extension of the error ID

Field element STATUS[2]

STATUS[2] can have the following values:

Error_Decode (B#16#...)	Source	Meaning
00 to 7F	CPU	No error or no warning
80	DPV1	Error according to IEC 61158-6
81 to 8F	CPU	B#16#8x shows an error in the xth call parameter of the instruction.
FE, FF	DP profile	Profile-specific error

Field element STATUS[3]

STATUS[3] can have the following values:

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
00	00		No error, no warning
70	00	reserved, reject	Initial call; no active data record transfer
	01	reserved, reject	Initial call; data record transfer has started
	02	reserved, reject	Intermediate call; data record transfer already active
80	81		The system data type at parameter TINFO does not match the call environment of the instruction. The used system data type must match the organization block in the user program (example: for a time-delay interrupt OB you need the system data type TI_Delay).
	90	reserved, pass	Invalid logical start address
	92	reserved, pass	Illegal type for VARIANT pointer
	93	reserved, pass	The DP component addressed via ID or F_ID is not configured.
	96		The "RALRM (Page 3636)" does not supply the OB start information, management information, header information, or additional interrupt information. For OBs 4x, 55, 56, 57, 82 and 83, you use the "DPNRM_DG (Page 3715)" instruction to read the current diagnostic frame of the relevant DP slave asynchronously (address information from OB start information).
	A0	read error	Negative acknowledgment when reading the module
	A1	write error	Negative acknowledgment when writing to the module
	A2	module failure	DP protocol error at layer 2 (e.g., slave failure or bus problems)
	A3	reserved, pass	• PROFIBUS DP: DP protocol error with Direct-Data-Link-Mapper or User-Interface/User • PROFINET IO: General CM error
	A4	reserved, pass	Communication on PBUS+ disrupted

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
	A5	reserved, pass	–
	A7	reserved, pass	DP slave or module is occupied (temporary error
	A8	version conflict	DP slave or module reports non-compatible versions
	A9	feature not supported	Function is not supported by DP slave or module
	AA to AF	user specific	DP slave or module reports a manufacturer-specific error in its application. Please check the documentation from the manufacturer of the DP slave or module.
	B0	invalid index	Data record not known in module Illegal data record number ≥ 256
	B1	write length error	Error at length specification: • With "RALRM (Page 3636)": length error in AINFO (Page 3659) • With "RDREC (Page 3578)": length error in MLEN • With "WRREC (Page 3596)": length error in LEN
	B2	invalid slot	The configured slot is not occupied.
	B3	type conflict	Actual module type does not match specified module type
	B4	invalid area	DP slave or module reports access to an invalid area
	B5	state conflict	DP slave or module not ready
	B6	access denied	DP slave or module denies access
	B7	invalid range	DP slave or module reports an invalid range for a parameter or value
	B8	invalid parameter	DP slave or module reports an invalid parameter
	B9	invalid type	DP slave or module reports an invalid type With "RDREC (Page 3578)": buffer too small (subsets cannot be read) With "WRREC (Page 3596)": buffer too small (subsets cannot be written)
	BA to BF	user specific	DP slave or module reports a manufacturer-specific error when accessing. Please check the documentation from the manufacturer of the DP slave or module.
	C0	read constrain conflict	With "WRREC (Page 3596)": The data is only written when the CPU is in STOP mode. Note: this means that writing by the user program is not possible. You only write the data online with PG/PC. With "RDREC (Page 3578)": the module routes the data record, but either no data is present or the data can only be read when the CPU is in STOP mode. Note: if data can only be read when the CPU is in STOP mode, then an evaluation by the user program is not possible. Therefore, it is only possible to read the data online with PG/PC.

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
	C1	write constrain conflict	The data of the previous write job on the module for the same data record has not yet been processed by the module.
	C2	resource busy	The module is currently processing the maximum possible number of jobs for a CPU.
	C3	resource unavailable	The required operating resources are currently occupied.
	C4		Internal temporary error. Job could not be executed. Repeat the job. If this error occurs often, check your installation for sources of electrical interference.
	C5		DP slave or module not available.
	C6		Data record transfer was canceled due to priority class cancellation
	C7		Job aborted due to warm or cold restart on the DP master
	C8 to CF		DP slave or module reports a manufacturer-specific resource error. Please check the documentation from the manufacturer of the DP slave or module.
	Dx	user specific	DP slave specific. Refer to the description of the DP slave.
	Ex	user specific	Manufacturer-specific. Please check the documentation of the module.
81	00 to FF		Error in the initial call parameter (with "RALRM (Page 3636)": MODE
	00		Illegal operating mode
82	00 to FF		Error in the second call parameter
:	:		:
88	00 to FF		Error in the eighth call parameter (with "RALRM (Page 3636)": TINFO (Page 3642))
	01		Wrong syntax ID
	23		Configuration limits exceeded or destination area too small
	24		Wrong range ID
	32		DB/DI no. out of user range
	3A		DB/DI no. is NULL for area ID DB/DI or specified DB/DI does not exist
89	00 to FF		Error in the ninth call parameter (with "RALRM (Page 3636)": AINFO (Page 3659))
	01		Wrong syntax ID
	23		Configuration limits exceeded or destination area too small
	24		Wrong range ID
	32		DB/DI no. out of user range
	3A		DB/DI no. is NULL for area ID DB/DI or specified DB/DI does not exist
8A	00 to FF		Error in the 10. Call parameters

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
:	:		:
8F	00 to FF		Error in the 15. Call parameters
FE, FF	00 to FF		Profile-specific error

Parameter TINFO (S7-1200, S7-1500)

Data structure of the destination area TINFO

The data structure of the destination area TINFO contains the start information of the organization block in which "RALRM" is currently being called.

The TINFO destination area can contain the start information with standard access or optimized access. The format of the start information in the TINFO destination area must always match the start information of the corresponding organization block.

- The start information of an OB with standard access is always stored in the first 20 bytes of the "Temp" section in the block interface. Use the "TI_Classic" data structure for this.
- The start information of an OB with optimized access is always written to the "Input" section. The start information is stored in data structures specific to the organization block. Use the data structure specific to the organization block.

Changing the block access (standard/optimized) also changes the block interface.

The following table provides an overview of the data structures that are used depending on the organization block at the TINFO parameter.

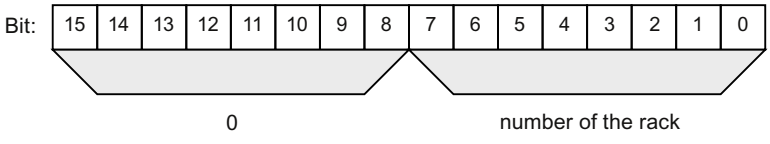
Name of the data structure		S7-1200 CPUs as of version	S7-1500 CPUs as of version	Data structure used for:
Data structure for organization blocks with standard access				
	TI_Classic	-	V1	Organization blocks without optimized block access.
Data structure for organization blocks with optimized block access				

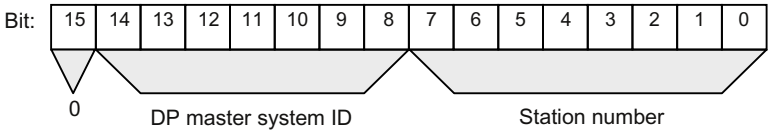
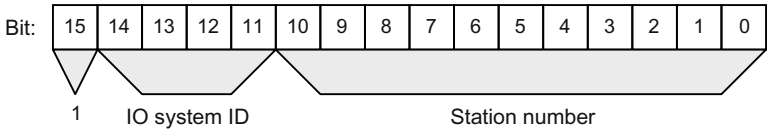
Name of the data structure	S7-1200 CPUs as of version	S7-1500 CPUs as of version	Data structure used for:
TI_ProgramCycle	V2	V1	Cycle OB (program cycle)
TI_Startup	V2	V1	Startup OB (startup)
TI_Delay	V2	V1	Time delay interrupt
TI_Cyclic	V2	V1	Cyclic interrupt OB
TI_HWInterrupt	V2	V1	Hardware interrupt OB
TI_HWInterrupt_Extended	-	V2.0	Hardware interrupt OB
TI_TimeError	V2	V1	Time error OB
TI_DiagnosticInterrupt	V2	V1	Diagnostic error interrupt OB
TI_PlugPullModule	V2	V1	Pull/plug of modules OB
TI_StationFailure	V2	V1	Rack or station failure OB
TI_ProgIOAccessError	V2	V1	• Programming error OB • I/O access error OB
TI_TimeOfDay	V2	V1	Time-of-day interrupt OB
TI_SynchCycle	-	V1	Synchronous cycle interrupt OB
TI_Submodule	V2	V1	• Status interrupt OB • Update interrupt OB • OB for manufacturer or profile-specific interrupt

Data structure for organization blocks with standard access

The following shows the TI_Classic data structure:

Parameter	Data type	Byte	Description
TI_Classic			Data structure for organization blocks without optimized block access.

Parameter	Data type	Byte	Description
Bytes 0 to 19: Start information of the OB in which "RALRM" is currently being called.*			
EV_CLASS	BYTE	0	Event class Example OB1: • Bits 0-3: ID of the event (1 = incoming event) • Bits 4-7: Event class (1 = Event class 1)
EV_NUM	BYTE	1	Event number (depending on OB type) Example OB1 (SCAN_1): • SCAN_1 = 1 for first call • SCAN_1 = 3 for all additional calls
PRIORITY	BYTE	2	Priority class
NUM	BYTE	3	OB number
TYP2_3	BYTE	4	Additional information Different information is stored in the BYTES "TYP2_3" and "TYPE 1" depending on the OB type used. What these are is described in the documentation for the organization blocks. Example (OB1): • TYP2_3: OB1_RESERVED_1 (reserved) • TYP1: OB1_RESERVED_2 (reserved)
TYP1	BYTE	5	
ZI1	WORD	6 to 7	Additional information Different information is stored in "ZI1" depending on the OB type used. What these are is described in the documentation for the organization blocks. Example (OB1): • ZI1: OB1_PREV_CYCLE (runtime of previous cycle in ms)
ZI2_3	DWORD	8 to 11	Additional information Different information is stored in "ZI2_3" depending on the OB type used. What these are is described in the documentation for the organization blocks. Example (OB1): • ZI2: OB1_MIN_CYCLE (minimum cycle time (ms) since the last startup) • ZI3: OB1_MAX_CYCLE (maximum cycle time (ms) since the last startup)
OB_DATE_TIME	DATE_AND_TIME (DT)	12 to 19	Date and time-of-day when the OB was called.
Bytes 20 and 21: Address information			
address	WORD	20 and 21	Address information like S7-300/400 CPUs: • In a central configuration, the rack number (0-31):  • For a distributed configuration with PROFIBUS DP: – DP master system ID (1-31) – Station number (0-127).

Parameter	Data type	Byte	Description
			 • In a distributed configuration with PROFINET IO: – The last two positions in the PROFINET IO system ID (0-15). To obtain the complete PROFINET IO system ID, you must add 100 (decimal) to it. – Station number (0-2047). 
Bytes 22 to 31: Management information			
slv_prfl	BYTE	22	Slave profile like S7-300/400 CPUs: • In a central configuration: 0 (data record 0 or data record 1) • In a distributed configuration: – Bit 0 to 3: Slave type - 0000: DP (structure data record 0) - 0001: DPS7 (structure data record 0 or data record 1) - 0010: DPS7 V1 (structure data record 0 or data record 1) - 0011: DPV1 (structure according to PROFIBUS DP standard) - 0100 – 0111: Reserved - 1000: PROFINET IO (structure according to PROFINET IO standard) - from 1001: Reserved – Bit 4 to 7: Profile type (reserved)
intr_type	BYTE	23	Interrupt info type like S7-300/400 CPUs: • In a central configuration: 0 • In a distributed configuration: – Bit 0 to 3: Interrupt info type - 0000: Interrupt comes from a configured remote module - 0001: Interrupt comes from a non-DPV1 slav / non IO device or a slot that is not configured - 0010: Interrupt generated in the CPU - from 0011: Reserved – Bit 4 to 7: Structure version - 0000: Initial - from 0001: Reserved
flags1	BYTE	24	Flags of the PROFIBUS DP master interface module / PROFINET IO controller interface module like S7-300/400 CPUs: • In a central configuration: 0 • In a distributed configuration: – Bit 0 = 0: Interrupt originating from an integrated interface module (PROFINET IO or PROFIBUS DP)

Parameter	Data type	Byte	Description
			– Bit 0 = 1: Interrupt originating from an external interface module (PROFINET IO or PROFIBUS DP) – Bit 1 to 7: Reserved
flags2	BYTE	25	Flags of the PROFIBUS DP master interface module / PROFINET IO controller interface module like S7-300/400 CPUs: • In a central configuration: 0 • For a distributed configuration with PROFIBUS DP: – Bit 0: EXT_DIAG_FLAG from the diagnostics message frame, or 0 if this bit does not exist in the interrupt The bit is 1 if the DP slave is faulty. – Bit 1 to 7: Reserved • In a distributed configuration with PROFINET IO: – Bit 0: ARDiagnosisState or 0 if there is no information in the interrupt. The bit is 1 if the IO device is faulty. – Bit 1 to 7: Reserved
id	UINT	26 and 27	Management information • In a central configuration: 0 • For a distributed configuration with PROFIBUS DP: PROFIBUS ID number as unique identifier of the PROFIBUS DP slave • In a distributed configuration with PROFINET IO: PROFINET IO device ID number as unique identifier of the PROFINET IO device
manufacturer	UINT	28 and 29	Manufacturer ID (only in a distributed configuration with PROFINET IO).
instance	UINT	30 and 31	Ident number of the instance (only in a distributed configuration with PROFINET IO).
*The start information depends on the OB used. You can find the start information of each OB type at the interface or in the documentation of the OB.			

Data structures for organization blocks with optimized block access

The data structures for organization blocks with optimized block access have the following format:

- Bytes 0 to 3: Format of the start information, class and number of the called OB (same structure for all data structures).
- Bytes 4 to 19: Optimized start information (structure depends on the OB type). The data in bytes 4 to 19 correspond to the structure and content of the corresponding OB interface.
- Bytes 20 to 31: In addition, the address and management information for certain OBs. The data in bytes 20 to 31 correspond to the data of 20 to 31 bytes of the TI_Classic data structure.

The format of the data structures is described in the following tables.

Table 11-26 **Cycle OB: Data structure TI_ProgramCycle**

Parameter		Data type	Byte	Description
TI_ProgramCycle				Data structure for cycle OB
	SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
	OB_Class	USINT	1	OB class (=1)
	OB_Nr	UINT	2	OB number (1...32767).
	Initial_Call	BOOL	4	= TRUE in the first call of this OB: • Transition from STOP or HOLD to RUN • After reloading
	Remanence	BOOL	5	= TRUE, if retentive data are available.

Table 11-27 **Startup OB: Data structure TI_Startup**

Parameter		Data type	Byte	Description
TI_Startup				Data structure for startup OB
	SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
	OB_Class	USINT	1	OB class (=100)
	OB_Nr	UINT	2	OB number (1...32767).
	LostRetentive	BOOL	4	=TRUE, if the contents of retentive data areas have been lost.
	LostRTC	BOOL	5	=TRUE, if the time-of-day of the real-time clock has been lost.

Table 11-28 **Time delay interrupt OB: Data structure TI_Delay**

Parameter		Data type	Byte	Description
TI_Delay				Data structure for time delay interrupt OB
	SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
	OB_Class	USINT	1	OB class (=20)
	OB_Nr	UINT	2	OB number (1...32767).
	Sign	WORD	4	User ID: Input parameter SIGN from the call of the "SRT_DINT (Page 3830)" instruction.

Table 11-29 **Cyclic interrupt OB: Data structure TI_Cyclic**

Parameter	Data type	Byte	Description
TI_Cyclic			Data structure for cyclic interrupt OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=30)
OB_Nr	UINT	2	OB number (1...32767).
Initial_Call	BOOL	4	= TRUE in the first call of this OB • At the transition from STOP or HOLD to RUN • After reloading
Event_Count	INT	6	Number of discarded start events since the last start of this OB.

Table 11-30 **Hardware interrupt OB: Data structure TI_HWInterrupt**

Parameter	Data type	Byte	Description
TI_HWInterrupt			Data structure for hardware interrupt OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=40)
OB_Nr	UINT	2	OB number (1...32767).
LADDR	HW_IO	4	Hardware identifier of the module that triggers the hardware interrupt.
USI	WORD	6	Identifier for future extensions (not user-relevant).
IChannel	USINT	8	Number of the channel that triggered the hardware interrupt.
EventType	BYTE	9	Identifier for the event type associated with the event triggering the interrupt (e.g., rising edge). You can find this ID in the description of the respective module.
address	WORD	20	See "address" parameter for the TI_Cassic. data structure
slv_prfl	BYTE	22	See "slv_prfl" parameter for the TI_Cassic data structure.
intr_type	BYTE	23	See "intr_type" parameter for the TI_Cassic data structure.
flags1	BYTE	24	See "flags1" parameter for the TI_Cassic data structure.
flags2	BYTE	25	See "flags2" parameter for the TI_Cassic data structure.
id	UNIT	26	See "id" parameter for the TI_Cassic data structure.
manufacturer	UNIT	28	See "manufacturer" parameter for the TI_Cassic data structure.
instance	UNIT	30	See "instance" parameter for the TI_Cassic data structure.

Table 11-31 **Hardware interrupt OB: Data structure TI_HWInterrupt_Extended**

Parameter	Data type	Byte	Description
TI_HWInterrupt_Extended			Data structure for hardware interrupt OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=40)
OB_Nr	UINT	2	OB number (1...32767).
LADDR	HW_IO	4	Hardware identifier of the module that triggers the hardware interrupt.
USI	WORD	6	Identifier for future extensions (not user-relevant).
IChannel	USINT	8	Number of the channel that triggered the hardware interrupt.
EventType	BYTE	9	Identifier for the event type associated with the event triggering the interrupt (e.g., rising edge). You can find this ID in the description of the respective module.
PointAddr	DWORD	12	• In the case of digital modules: Bit array containing the inputs on the module that triggered the hardware interrupt • In the case of analog modules: Bit array with the information as to which channel has exceeded which limit • In the case of CPs or IMs: Interrupt status of the module (not user-relevant)
address	WORD	20	See "address" parameter for the TI_Cassic. data structure
slv_prfl	BYTE	22	See "slv_prfl" parameter for the TI_Cassic data structure.
intr_type	BYTE	23	See "intr_type" parameter for the TI_Cassic data structure.
flags1	BYTE	24	See "flags1" parameter for the TI_Cassic data structure.
flags2	BYTE	25	See "flags2" parameter for the TI_Cassic data structure.
id	UNIT	26	See "id" parameter for the TI_Cassic data structure.
manufacturer	UNIT	28	See "manufacturer" parameter for the TI_Cassic data structure.
instance	UNIT	30	See "instance" parameter for the TI_Cassic data structure.

Table 11-32 **Time error OB: Data structure TI_TimeError**

Parameter	Data type	Byte	Description
TI_TimeError			Data structure for time error OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=80)
OB_Nr	UINT	2	OB number (1...32767).
Csg_OBnr	OB_ANY	4	Number of the OB that was being executed when the time error occurred.
Fault_ID	BYTE	7	Error code. Possible values: • B#16#01: Cycle time exceeded • B#16#02: The requested OB is still being processed. • B#16#05: Expired time-of-day interrupt due to time jump. • B#16#06: Expired time-of-day interrupt upon re-entry into RUN after HOLD. • B#16#07: Overflow of the OB request buffer for the current priority class. • B#16#08: Isochronous mode interrupt - time error. • B#16#09: Interrupt loss due to high interrupt load • B#16#0B: Technology synchronization interrupt - time error
Csg_Prio	UNIT	8	Priority of the OB that was being executed when the time error occurred.

Table 11-33 **Diagnostic interrupt OB: Data structure TI_DiagnosticInterrupt**

Parameter	Data type	Byte	Description
TI_DiagnosticInterrupt			Data structure for diagnostic interrupt OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=82)
OB_Nr	UINT	2	OB number (1...32767).
LADDR	HW_ANY	4	Hardware identifier of the hardware object that triggered the diagnostic interrupt.
IO_State	WORD	6	Status of the hardware object: • Bit 0: Good • Bit 1: Disabled • Bit 2: Maintenance required • Bit 3: Maintenance demanded • Bit 4: Error • Bit 5: Not accessible • Bit 6: Qualified • Bit 7: Not available
Channel	UINT	8	Channel number
MultiError	BOOL	10	=TRUE, if there is more than one error.
address	WORD	20	See "address" parameter for the TI_Cassic. data structure
slv_prfl	BYTE	22	See "slv_prfl" parameter for the TI_Cassic data structure.
intr_type	BYTE	23	See "intr_type" parameter for the TI_Cassic data structure.
flags1	BYTE	24	See "flags1" parameter for the TI_Cassic data structure.
flags2	BYTE	25	See "flags2" parameter for the TI_Cassic data structure.
id	UINT	26	See "id" parameter for the TI_Cassic data structure.
manufacturer	UINT	28	See "manufacturer" parameter for the TI_Cassic data structure.
instance	UINT	30	See "instance" parameter for the TI_Cassic data structure.

Table 11-34 **Pull/plug interrupt OB: Data structure** TI_PlugPullModule

Parameter	Data type	Byte	Description
TI_PlugPullModule			Data structure for pull/plug interrupt OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=83)
OB_Nr	UINT	2	OB number (1...32767).
LADDR	HW_IO	4	Hardware identifier of the affected module or submodule
Event_Class	BYTE	6	• B#16#38: (Sub)module plugged • B#16#39: (Sub)module pulled or not responding
Fault_ID	BYTE	7	Error code The following table shows the events that cause the start of the pull/plug interrupt OB. • With Event_Class = B#16#38 - module/submodule plugged: – B#16#54: Submodule plugged and matches configured submodule – B#16#55: Submodule plugged, but does not match configured submodule – B#16#56: Submodule plugged, but error in module parameter assignment – B#16#57: Submodule or module plugged, but with a fault or maintenance – B#16#58: Submodule access error corrected – B#16#61: Module inserted • With Event_Class = B#16#39 - module/submodule pulled or not responding: – B#16#51: Module pulled – B#16#54: Submodule pulled – B#16#61: Module pulled or not responding
address	WORD	20	See "address" parameter for the TI_Cassic. data structure
slv_prfl	BYTE	22	See "slv_prfl" parameter for the TI_Cassic data structure.
intr_type	BYTE	23	See "intr_type" parameter for the TI_Cassic data structure.
flags1	BYTE	24	See "flags1" parameter for the TI_Cassic data structure.
flags2	BYTE	25	See "flags2" parameter for the TI_Cassic data structure.
id	UINT	26	See "id" parameter for the TI_Cassic data structure.
manufacturer	UINT	28	See "manufacturer" parameter for the TI_Cassic data structure.
instance	UINT	30	See "instance" parameter for the TI_Cassic data structure.

Table 11-35 **Rack error OB: Data structure TI_StationFailure**

Parameter	Data type	Byte	Description
TI_StationFailure			Data structure for rack error OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=86)
OB_Nr	UINT	2	OB number (1...32767).
LADDR	HW_Device	4	Hardware identifier of the defective hardware object.
Event_Class	BYTE	6	• B#16#32: Activation of a station with the "D_ACT_DP" instruction • B#16#33: Deactivation of a station with the "D_ACT_DP" instruction • B#16#38: Outgoing event • B#16#39: Incoming event
Fault_ID	BYTE	7	Error code The error code is used to output the event that caused the start of the rack error OB. • Event_Class = B#16#39, FAULT_ID = C3: Distributed I/O: Failure of a DP master system • Event_Class = B#16#38, FAULT_ID = C4: Failure of a DP station • Event_Class = B#16#39, FAULT_ID = CA: PROFINET IO system failure • Event_Class = B#16#39/38, FAULT_ID = CB: PROFINET IO station failure/ station return • Event_Class = B#16#38, FAULT_ID = CC: PROFINET IO station return with problem or maintenance • Event_Class = B#16#38, FAULT_ID = CD: PROFINET IO station return, expected configuration different from actual configuration • Event_Class = B#16#38, FAULT_ID = CE: PROFINET IO station return, error in module parameter assignment • Event_Class = B#16#32/33, FAULT_ID = CF: Activation/deactivation of a PROFINET IO device with the "D_ACT_DP" instruction • Event_Class = B#16#39/38, FAULT_ID = F8: Failure/return of some of the submodules of a PROFINET I-device • Event_Class = B#16#38, FAULT_ID = F9: Return of some of the submodules of a PROFINET I-device with a device configuration difference
address	WORD	20	See "address" parameter for the TI_Cassic. data structure
slv_prfl	BYTE	22	See "slv_prfl" parameter for the TI_Cassic data structure.
intr_type	BYTE	23	See "intr_type" parameter for the TI_Cassic data structure.
flags1	BYTE	24	See "flags1" parameter for the TI_Cassic data structure.
flags2	BYTE	25	See "flags2" parameter for the TI_Cassic data structure.
id	UINT	26	See "id" parameter for the TI_Cassic data structure.
manufacturer	UINT	28	See "manufacturer" parameter for the TI_Cassic data structure.
instance	UINT	30	See "instance" parameter for the TI_Cassic data structure.

Table 11-36 **Programming error OB / I/O access error OB: Data structure TI_ProgIOAccessError**

Parameter	Data type	Byte	Description*
TI_ProgIOAccessError			Data structure for programming error OB and I/O access error OB

Parameter	Data type	Byte	Description*
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class: • =121 for programming error OB • =122 for I/O access error OB
OB_Nr	UINT	2	OB number (1...32767).
BlockNr	UINT	4	Number of the block in which the programming error occurred.
Reaction	USINT	6	• 0: Ignore error • 1: Substitute bad value • 2: Skip command
Fault_ID	BYTE	7	Error code • B#16#21: BCD conversion error • B#16#22: Area length error when reading • B#16#23: Area length error when writing • B#16#24: Area error when reading • B#16#25: Area error when writing • B#16#26: Error in timer no. • B#16#27: Error in counter no. • B#16#28: Read access to a byte, word or double word with a pointer, the bit address of which is not 0 • B#16#29: Write access to a byte, word or double word with a pointer, the bit address of which is not 0 • B#16#:30: Write access to a write-protected global DB • B#16#31: Write access to a write-protected instance DB • B#16#32: DB number error when accessing a global DB • B#16#33: DB number error when accessing an instance DB • B#16#34: Number error during FC call • B#16#35: Number error during FB call • B#16#42: I/O access error, reading • B#16#43: I/O access error, writing • B#16#3A: Access to a DB that is not loaded; the DB number is located in the permitted area • B#16#3C: Access to an FC that is not loaded; the FC number is within the permissible range • B#16#3D: Access to an instruction (SFC) that is not loaded; the SFC number is within the permissible range • B#16#3E: Access to an FB that is not loaded; the FB number is within the permissible range • B#16#3F: Access to an SFB that is not available; the SFB number is within the permissible range
BlockType	USINT	8	Type of block in which the error has occurred:

Parameter	Data type	Byte	Description*
			• OB: B#16#88 • FC: B#16#8C • FB: B#16#8E
Area	USINT	9	Memory area in which the incorrect access occurred: • Local data: B#16#40 to 4E, 86, 87, 8E, 8F, C0 to CE • Process image input: B#16#01 • Process image output: B#16#02 • Technology DB: B#16#04 • I: B#16#81 • Q: B#16#82 • M: B#16#83 • DB: B#16#84, 85, 8A, 8B
DBNr	DB_ANY	10	DB no. if AREA = DB (global DB or instance DB) Only relevant for programming error OB.
Csg_OBNr	OB_ANY	12	OB number: • 121: Programming error OB • 122: I/O access error OB
Csg_Prio	USINT	14	OB priority
Width	USINT	15	Type of access during which the error occurred: • Bit: B#16#00 • Byte: B#16#01 • Word: B#16#02 • DWord: B#16#03 • LWord: B#16#04
* Only certain output values are possible depending on the OB (I/O access error OB or programming error OB) from which information is read.			

Table 11-37 **Time-of-day interrupt OB: Data structure TI_TimeOfDay**

Parameter	Data type	Byte	Description
TI_TimeOfDay			Data structure for time-of-day interrupt OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=10)
OB_Nr	UINT	2	OB number (1...32767).
CaughtUp	BOOL	4	=TRUE if an OB call was executed subsequently because time was adjusted ahead
SecondTime	BOOL	5	=TRUE if an OB was called a second time because time was adjusted back. Note: SecondTime is set only once in situations in which the time is set back.

Table 11-38 **Isochronous mode interrupt OB: Data structure TI_SynchCycle**

Parameter	Data type	Byte	Description
TI_SynchCycle			Data structure for isochronous mode interrupt OB
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class (=61)
OB_Nr	UINT	2	OB number (1...32767).
Initial_Call	BOOL	4	= TRUE in the first call of this OB: • Transition from STOP or HOLD to RUN • After reloading
IO _System	USINT	5	Number of the distributed I/O system triggering the interrupt
Event_Count	INT	6	• = n: Number of lost cycles • = -1: An unknown number of cycles has been lost (e.g., because cycle has changed).
PIP_Input	BOOL	10	=TRUE: The associated process image of the inputs is up-to-date
PIP_Output	BOOL	11	=TRUE: The associated process image of the outputs was transferred to the outputs in good time after the last cycle
SyncCycle-Time	LTIME	16	Calculated cycle time

Table 11-39 **Status interrupt OB / update interrupt OB for manufacturer or profile-specific interrupt: Data structure TI_Submodule**

Parameter	Data type	Byte	Description
TI_Submodule			Data structure for status interrupt OB / update interrupt OB for manufacturer or profile-specific interrupt
SI_Format	USINT	0	Format of the start information: • 16#FF: None • 16#FE: Optimized start information
OB_Class	USINT	1	OB class • =55 for status interrupt OB • =56 for update interrupt OB • =57 for manufacturer or profile-specific interrupt OB
OB_Nr	UINT	2	OB number (1...32767).
LADDR	HW_IO	4	Hardware address of the component triggering the interrupt
Slot	UINT	6	Slot number of the component triggering the interrupt
Specifier	WORD	8	Interrupt specifier from the interrupt frame
address	WORD	20	See "address" parameter for the TI_Cassic. data structure
slv_prfl	BYTE	22	See "slv_prfl" parameter for the TI_Cassic data structure.
intr_type	BYTE	23	See "intr_type" parameter for the TI_Cassic data structure.
flags1	BYTE	24	See "flags1" parameter for the TI_Cassic data structure.
flags2	BYTE	25	See "flags2" parameter for the TI_Cassic data structure.
id	UINT	26	See "id" parameter for the TI_Cassic data structure.
manufacturer	UINT	28	See "manufacturer" parameter for the TI_Cassic data structure.
instance	UINT	30	See "instance" parameter for the TI_Cassic data structure.

Parameter AINFO (S7-1200, S7-1500)

Data structure of the destination area AINFO with interrupts from PROFIBUS DP

Byte	Meaning	
0 to 3	Header information, for exact description, see below	
4 to 63	Additional interrupt information: data for the respective interrupt:	
	Distributed:	ARRAY[0] to ARRAY[59]

Structure of the header information with interrupts from PROFIBUS DP

Byte	Data type	Meaning	
0	BYTE	Length of the received interrupt information in bytes	
		Central:	4 to 224
		Distributed:	4 to 63

Byte	Data type	Meaning		
1	BYTE	Central:	Reserved	
		Distributed:	ID for the interrupt type	
			1: 2: 3: 4: 5: 6: 31	Diagnostics interrupt Process interrupt Pull interrupt Plug interrupt Status interrupt Update interrupt Failure of an expansion device, DP master system or DP station
			32 to 126:	Manufacturer-specific interrupt
2	BYTE	Slot number of the component that triggered the interrupt		
3	BYTE	Central:	Reserved	
		Distributed:	Specifier	
			Bits 0 and 1:	0: No further information; 1: Incoming event, fault at slot 2: Outgoing event, fault at slot cleared 3: Outgoing event, slot still faulty
			Bit 2:	Add_Ack
			Bits 3 to 7:	Sequence number

Data structure of the destination area AINFO with interrupts from PROFINET IO or central I/O devices

Byte	Meaning
0 to 25	Header information, for exact description, see below
26 to 1431	Additional interrupt information: Standardized diagnostics data for the respective interrupt: ARRAY[0] to ARRAY[1405] Note: The additional interrupt information may also be omitted.

Structure of the header information with interrupts from PROFINET IO or central I/O devices

Byte	Data type	Meaning
0 and 1	WORD	• Bits 0 to 7: Block type • Bits 8 to 15: Reserved
2 and 3	WORD	Block length
4 and 5	WORD	Version: • Bits 0 to 7: low byte • Bits 8 to 15: high byte

Byte	Data type	Meaning
6 and 7	WORD	ID for interrupt type: • 1: Diagnostics interrupt (incoming or outgoing) • 2: Process interrupt • 3: Remove module interrupt • 4: Insert module interrupt • 5: Status interrupt • 6: Update interrupt • 7: Redundancy interrupt • 8: Controlled by supervisor • 9: Released by supervisor • 10: Configured module not inserted • 11: Return of the sub-module • 12: Diagnostics interrupt (outgoing) • 13: Slave-to-slave connection alarm • 14: Neighborhood change alarm • 15: Clock synchronization message (bus end) • 16: Clock synchronization alarm (device end) • 17: Network component alarm • 18: Time synchronization alarm (on bus side) • 19 to 31: Reserved • 32 to 127: Manufacturer-specific interrupt • 128 to 65535: Reserved
8 to 11	DWORD	API (Application Process Identifier)
12 to 13	WORD	Slot number of the component triggering the interrupt (value range 0 to 65535)
14 to 15	WORD	Submodule slot number of the component triggering the interrupt (value range 0 to 65535)
16 to 19	DWORD	Module identification; specific information on the source of the interrupt

Byte	Data type	Meaning
20 to 23	DWORD	Submodule identification; specific information on the source of the interrupt
24 to 25	WORD	Interrupt specifier: • Bits 0 to 10: Sequence number (value range 0 to 2047) • Bit 11: Channel diagnostics: 0: No channel diagnostics available 1: Channel diagnostic information exists • Bit 12: Status of manufacturer-specific diagnostics: 0: No manufacturer-specific status information available 1: Manufacturer-specific status information available • Bit 13: Status of diagnostics for sub-module: 0: No status information available, all errors have been corrected 1: At least one item of channel diagnostics and/or status information is available • Bit 14: Reserved • Bit 15: Application relationship diagnosis state: – 0: None of the modules configured within this AR are reporting diagnostic information – 1: At least one of the modules configured in this AR is reporting diagnostic information

Structure of the additional interrupt information for interrupts from PROFINET IO or central I/O

The additional interrupt information for PROFINET IO depends on the format identifier. It can comprise multiple data blocks with the same or different format identifier. The following format identifiers are available:

- W#16#0000 to W#16#7FFF: Manufacturer-specific diagnostics

Byte	Data type	Meaning
0 to 1	WORD	Format identifier for the structure of the following data serving as additional interrupt information W#16#0000 to W#16#7FFF: Manufacturer-specific diagnostics
2 to n	BYTE	See manufacturer's manual.

- W#16#8000: Channel diagnostics
  Channel diagnostics is output in blocks of 6 bytes each. The additional interrupt information (without format identifier) is only output for disrupted channels.

Byte	Data type	Meaning	
0 to 1	WORD	Format identifier for the structure of the following data serving as additional interrupt information W#16#8000: Channel diagnostics	
2 to 3	WORD	Channel number of the component triggering the interrupt (value range: 0 to 65535): • W#16#0000 to W#16#7FFF: Channel number of the interface module/sub-module • W#16#8000: The generic substitute for the whole sub-module • W#16#8001 to W#16#FFFF: Reserved	
4	BYTE	Bits 0 to 2:	Reserved
		Bits 3 to 4:	Type of error: • 0: Reserved • 1: Incoming error • 2: Outgoing error • 3: Outgoing error, other errors present
		Bits 5 to 7:	Type of channel: • 0: Reserved • 1: Input channel • 2: Output channel • 3: Input/output channel

Byte	Data type	Meaning
5	BYTE	Data format: • B#16#00: Free data format • B#16#01: Bit • B#16#02: 2-bit • B#16#03: 4-bit • B#16#04: Byte • B#16#05: Word • B#16#06: Double word • B#16#07: 2 double words • B#16#08 to B#16#FF: Reserved

Byte	Data type	Meaning
6 to 7	WORD	Type of error: • W#16#0000: Reserved • W#16#0001: Short-circuit • W#16#0002: Undervoltage • W#16#0003: Overvoltage • W#16#0004: Overload • W#16#0005: Overtemperature • W#16#0006: Wire break • W#16#0007: High limit exceeded • W#16#0008: Low limit value exceeded • W#16#0009: Error • W#16#000A: Simulation active • W#16#000B to W#16#000E: Reserved • W#16#000F to W#16#001F: Manufacturer-specific • W#16#0020 to W#16#00FF: Reserved • W#16#0100 to w#16#7FFF: Manufacturer-specific • W#16#8000: Data transfer not possible • W#16#8001: Error at partner • W#16#8002: Media redundancy error • W#16#8003: Synchronization loss • W#16#8004: Isochronous mode error • W#16#8005: Multicast CR error • W#16#8006: Reserved • W#16#8007: Fiber-optic error • W#16#8008: Function failure of network component • W#16#8009: Time error • W#16#800A: DFP problem (DFP: dynamic frame packing) • W#16#800B: MRPD problem (MRPD: media redundancy with planned duplication) • W#16#800C: System redundancy error • W#16#800D to W#16#FFFF: Reserved Not all channels support every error type. For detailed information, refer to the description of the diagnostic data for the specific device.

Note

The section from "channel number" to "error type" can occur from 0 to n times.

- W#16#8001: MULTIPLE (different types of diagnostics information are transmitted). In this case, the additional interrupt information is transmitted as blocks of variable length.

Byte	Data type	Meaning
0 to 1	WORD	Format identifier for the structure of the following data serving as additional interrupt information W#16#8001: Manufacturer-specific diagnostics and/or channel diagnostics
2 to 3	WORD	Block type
4 to 5	WORD	Block length
6	BYTE	Version: high byte
7	BYTE	Version: low byte
8 to 11	DWORD	API (only if low byte of version = 1)
12 to 13	WORD	Slot number
14 to 15	WORD	Subslot number
16 to 17	WORD	Channel number
18 to 19	WORD	Channel properties
20 to 21	WORD	Format identifier: • W#16#0000 to W#16#7FFF: Manufacturer-specific diagnostics • W#16#8000: Channel diagnostics • W#16#8002: Extended channel diagnostics • W#16#8003: Stepped extended channel diagnostics • W#16#8004 to W#16#80FF: Reserved
22 to n	BYTE	Data depend on the format identifier

Note

The section starting from "block type" can occur from 1 to n times.

- W#16#8002: Extended channel diagnostics

Byte	Meaning
0 to 1	Format identifier W#16#8002
2 to 3	Channel number
4 to 5	Channel properties
6 to 7	Error type
8 to 9	Additional error value
10 to 13	Additional error information

- W#16#8003: Stepped extended channel diagnostics

Byte	Meaning
0 to 1	Format identifier W#16#8003
2 to 3	Channel number
4 to 5	Channel properties
6 to 7	Error type
8 to 9	Additional error value
10 to 13	Additional error information
14 to 17	Qualified channel qualifier

- W#16#8100: Maintenance information

Byte	Meaning
0 to 1	Format identifier W#16#8100
2 to 3	Block type
4 to 5	Block length
6 to 7	Block version
8 to 9	Reserved Note: These two bytes are missing with central modules. These exist with distributed modules (according to the PROFINET standard).
10 to 13	Maintenance status

Note

You can find more detailed information about the structure of the additional alarm information in the *Programming Manual SIMATIC PROFINET IO from PROFIBUS DP to PROFINET IO* and the current version of IEC 61158-6-10-1.

Destination area TINFO and AINFO (S7-1200, S7-1500)

Destination area TINFO and AINFO

Depending on the respective OB in which "RALRM (Page 3636)" is called, the destination areas TINFO and AINFO are only partially written. Refer to the table below to find out which information is entered respectively.

Interrupt type	OB class	TINFO OB start information	TINFO Management information	AINFO Header information	AINFO Additional interrupt information	
Process interrupt	4x	Yes	Yes	Yes	Central:	No
					Distributed:	As supplied by PROFIBUS DP slave/PROFINET IO device
Status interrupt	55	Yes	Yes	Yes	Yes	Yes
Update interrupt	56	Yes	Yes	Yes	Yes	Yes
Manufacturer-specific interrupt	57	Yes	Yes	Yes	Yes	Yes

Interrupt type	OB class	TINFO OB start information	TINFO Management information	AINFO Header information	AINFO Additional interrupt information	
I/O redundancy error	70	Yes	Yes	No	No	No
Diagnostics interrupt	82	Yes	Yes	Yes	Central:	Structure according to PROFINET IO standard
					Distributed:	As supplied by PROFIBUS DP slave/PROFINET IO device
Insert/remove interrupt	83	Yes	Yes	Yes	Central:	No
					Distributed:	As supplied by PROFIBUS DP slave/PROFINET IO device
Special form of the remove module interrupt: Controlled by supervisor	83	Yes	Yes	Yes	PROFINET IO only	
Special form of the insert module interrupt: Enabled by supervisor	83	Yes	Yes	Yes	PROFINET IO only	
Unconfigured module inserted	83	Yes	Yes	Yes	PROFINET IO only	
Rack failure/ station failure	86	Yes	Yes	No	No	
... all other OBs		Yes	No	No	No	

Program example for RALRM (S7-1200, S7-1500)

Introduction

In the following example, you generate a wire break via an input module and receive the interrupt, including associated information. The example applies to PROFINET IO and/or an S7-1500 CPU.

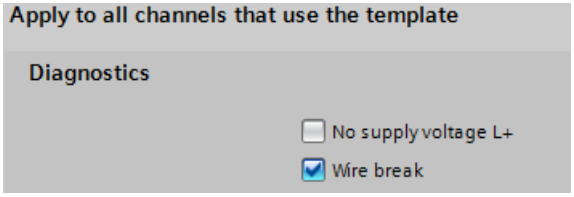
Note

Place the example

The example must be stored in the setup in which the input module is to generate an interrupt (wire break).

Requirements

- An input module is created and the hardware identifier is read out in the properties of the input module. You store the hardware identifier via the input parameter F_ID ("alarmCheckID").
- To generate an interrupt, select the option "Wire break" for the input channels. This option is available under "Module parameters > Channel template > Inputs".



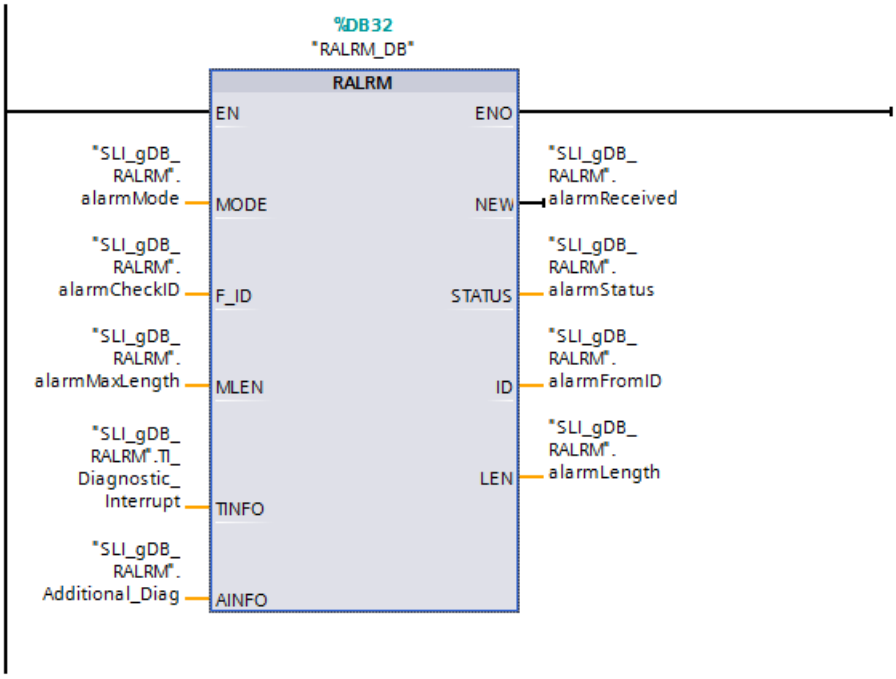
- To store the data, create seven tags and the structures "TI_Diagnostic_Interrupt" and "Additional_Diag" in a global data block. Create the "TI_Diagnostic_Interrupt" structure with the "TI_DiagnosticInterrupt" data type.

SLI_gDB_RALRM

	Name	Data type	Start value
1	Static		
2	alarmMode	Int	2
3	alarmCheckID	HW_IO	257
4	alarmStatus	DWord	16#0
5	alarmFromID	HW_IO	16#0
6	alarmReceived	Bool	false
7	alarmMaxLength	UInt	0
8	alarmLength	UInt	0
9	TI_Diagnostic_Interrupt	TI_DiagnosticInterrupt	
10	Additional_Diag	Struct	
11	header	Struct	
12	BlockType	Word	16#0
13	BlockLength	UInt	0
14	Version	Word	16#0
15	InterruptType	UInt	0
16	API	DWord	16#0
17	SlotNumber	UInt	0
18	SubSlotNumber	UInt	0
19	ModulSpecInfo	DWord	16#0
20	SubmodulSpecInfo	DWord	16#0
21	InterruptSpecifier	Word	16#0
22	addInfo	Struct	
23	FormatIdentifier	Word	16#0
24	ChannelNumber	Word	16#0
25	ChannelErrorType	Byte	16#0
26	DataFormat	Byte	16#0
27	ErrorCode	Word	16#0

Interconnect parameters

Call the instruction "RALRM" in a diagnostic interrupt OB (OB82). Interconnect the parameters of the instruction as follows.



Result of RALRM

Note

Trigger wire break

To trigger the interrupt, pull one of the power cables of the input module's inputs and then reconnect the cable. Alternatively, you can also pull a potential bridge of the input module and reconnect it.

When you generate a wire break, the input module generates an interrupt. The diagnostic interrupt OB is then called and the instruction "RALRM" started.

The instruction "RALRM" calls the input module via the input parameter F_ID ("alarmCheckID"). The instruction "RALRM" checks whether the interrupt originates from the input module according to the value "2" of the input parameter MODE ("alarmMode"). If the interrupt originates from the input module ("alarmCheckID" and "alarmFromID" are identical), the output parameter NEW ("alarmReceived") is set to "TRUE" and the interrupt information is stored via the parameters TINFO ("TI_Diagnostivc_Interrupt") and AINFO ("Additional_Diag"). The hardware identifier of the module generating the interrupt (the input module) is displayed at the output parameter ID ("alarmFromID").

The length of the received interrupt information is recorded at the output parameter LEN ("alarmLength"). According to the value of the input parameter MLEN ("alarmMaxLength" has the value “0"), the information that is to be read for AINFO ("Additional_Diag") is unlimited.

The block status is displayed at the output parameter STATUS ("alarmStatus"). Processing takes place without errors.

SLI_gDB_RALRM

	Name	Data type	Start value	Monitor value
1	Static			
2	alarmMode	Int	2	2
3	alarmCheckID	HW_IO	257	16#0101
4	alarmStatus	DWord	16#0	16#0000_0000
5	alarmFromID	HW_IO	16#0	16#0101
6	alarmReceived	Bool	false	TRUE
7	alarmMaxLength	UInt	0	0
8	alarmLength	UInt	0	76

The following sections contain the evaluation for TINFO and AINFO.

Details about parameter TINFO ("TI_Diagnostivc_Interrupt")

The OB start information and management information are stored at the parameter TINFO ("TI_Diagnostivc_Interrupt").

The generated interrupt includes the following information:

- OB start information: The OB82 uses optimized start information ("SI_Format" with the value "254").
- Management information:
  - The input module (see "LADDR") with the status "Error" (see the value of "IO_State") is read out.
  - An interrupt was recently recorded for the channel "15" (see the value of "Channel").
  - Multiple errors were recorded ("MutliError" with the value "TRUE").
  - An interrupt was recently recorded for station "0" (see the value of "address").
  - A structure was recorded according to the diagnostics data record 0 (see the value of "slv_prfl").
  - The interrupt was triggered by a configured, distributed module (see the value of "intr_type").

9	TI_Diagnostic_Interrupt	TI_DiagnosticInter...		
10	SI_Format	USInt	0	254
11	OB_Class	USInt	82	82
12	OB_Nr	UInt	0	82
13	LADDR	HW_ANY	16#0	16#0101
14	IO_State	Word	16#0	16#0010
15	Channel	UInt	0	15
16	MultiError	Bool	false	TRUE
17	address	Word	16#0	16#0000
18	slv_prfl	Byte	16#0	16#00
19	intr_type	Byte	16#0	16#00
20	flags1	Byte	16#0	16#00
21	flags2	Byte	16#0	16#00
22	id	UInt	0	0
23	manufacturer	UInt	0	0
24	instance	UInt	0	0

Details about parameter AINFO ("Additional_Diag")

The header information and additional information about the interrupt is stored at the parameter AINFO ("Additional_Diag").

Note

Structure and scope of the interrupt information

This example is intended to show reception of a single hardware fault (wire break). Accordingly, only the interrupt information about one of the relevant channels is displayed when a multierror occurs.

- To call the additional information for each channel individually, use a multiple call of RALRM: For each call, query the channel ("TI_Diagnostivc_Interrupt.Channel") and store the information of the tags "ChannelErrorType" to "ErrorCode" for each channel. See also "FAQ for channel diagnostics with RALRM (https://support.industry.siemens.com/cs/ww/en/view/109480387)".
- Depending on the application, note the structure to be stored at the parameter AINFO (Page 3659).

The generated interrupt includes the following information:

- The OB used is a diagnostic interrupt OB ("InterruptType" with the value "1").
- The interrupt is reported for a module at the slot "2" ("SlotNumber") and for the channels 8 to 15 (BYTE 1, "SubSlotNumber").
- At least one item of channel diagnostics and/or status information is available for the submodule ("InterruptSpecifier" with the value "16#2000").
- In this example, the structure of the channel diagnostics is used ("FormatIdentifier" with the value "16#8000").
- The channel number of the component that triggered the interrupt is assigned to the submodule ("ChannelNumber" with the value "16#0008").
- An incoming error for an input channel ("ChannelErrorType" with the value "16#28") is diagnosed.

- The data format used is a bit ("DataFormat" with the value "16#01").
- A wire break was diagnosed ("ErrorCode" with the value "16#0006").

25	Additional_Diag	Struct		
26	header	Struct		
27	BlockType	Word	16#0	16#0002
28	BlockLength	UInt	0	72
29	Version	Word	16#0	16#0100
30	InterruptType	UInt	0	1
31	API	DWord	16#0	16#0000_0000
32	SlotNumber	UInt	0	2
33	SubSlotNumber	UInt	0	1
34	ModulSpecInfo	DWord	16#0	16#0000_0000
35	SubmodulSpecInfo	DWord	16#0	16#0000_0000
36	InterruptSpecifier	Word	16#0	16#2000
37	addInfo	Struct		
38	FormatIdentifier	Word	16#0	16#8000
39	ChannelNumber	Word	16#0	16#0008
40	ChannelErrorType	Byte	16#0	16#28
41	DataFormat	Byte	16#0	16#01
42	ErrorCode	Word	16#0	16#0006

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Additional example

A more complex example is available here: "FAQ for channel diagnostics with RALRM (https://support.industry.siemens.com/cs/ww/en/view/109480387)".

D_ACT_DP: Enable/disable DP slaves (S7-1200, S7-1500)

Description

With the instruction D_ACT_DP", you can disable and enable configured DP slaves/ PROFINET IO devices in a targeted manner. In addition, you can determine whether each assigned DP slave or PROFINET IO device is currently activated or deactivated.

You cannot disable/enable an IE/PB Link PN IO type of gateway using the instruction "D_ACT_DP". If you nevertheless use "D_ACT_DP" on the gateway named, the CPU returns the value W#16#8093 (there is no hardware object that can be activated or deactivated for the address specified in LADDR).

As of firmware version V1.8 of the S7-1500 CPUs the following applies: You can use the "D_ACT_DP" instruction to disable/enable DP slaves that are connected to an IE/PB Link PN IO type of gateway.

The instruction cannot be used on PROFIBUS PA field devices that are connected by a DP/PA link to a DP master system.

Note

Several runs through the cycle control point are required to perform the disabling or enabling job. Therefore, you do not wait for the end of such a job in a programmed loop.

Functional description

"D_ACT_DP" is an asynchronous instruction. Processing takes place across several calls. You start the job by calling "D_ACT_DP" with REQ = 1.

The output parameters RET_VAL and BUSY indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571).

Application

If you configure DP slaves/PROFINET IO devices in a CPU which are not actually present or not currently required, the CPU will nevertheless continue to access these DP slaves/PROFINET IO devices at regular intervals. After the slaves are deactivated, further CPU accessing will stop. With PROFIBUS DP, the fastest possible DP bus cycle can be achieved in this manner. The corresponding error events no longer occur.

Examples

From a machine OEM's point of view, there are numerous device options possible in series production of machines. However, each delivered machine includes only one combination of selected options.

Every one of these possible machine options is configured as a DP slave/PROFINET IO device by the manufacturer in order to create and maintain a common user program having all possible options. Use "D_ACT_DP" to deactivate all DP slaves/PROFINET IO devices not present at machine startup.

A similar situation exists for machine tools having numerous tooling options available but actually using only a few of them at any given time. These tools are implemented as DP slaves/PROFINET IO devices. With "D_ACT_DP", the user program activates the tools currently needed and deactivates those required later.

Identification of a job

If you have started a deactivation or activation job and you call "D_ACT_DP" again before the job is complete, the behavior of the instruction depends on whether or not the new call involves the same job. If the input parameter LADDR matches, the call will be interpreted as follow-on call.

Deactivating DP slaves/PROFINET IO devices

When you deactivate a DP slave/PROFINET IO device with "D_ACT_DP", its process outputs are set to the configured substitute values or to 0 (safe state). The assigned DP master/ PROFINET IO controller does not continue to address this component. Deactivated DP slaves/ PROFINET IO devices are not identified as faulty or missing by the error LEDs on the DP master/PROFINET IO controller or CPU.

The process image inputs of deactivated DP slaves/PROFINET IO devices is updated with 0. Therefore, it is treated just like failed DP slaves/PROFINET IO devices.

If you directly access the user data of a previously deactivated DP slave/PROFINET IO device from your program, the I/O access error OB is called and the corresponding start event is entered in the diagnostics buffer. If you attempt to access a deactivated DP slave/PROFINET IO device via an instruction (such as "RD_REC (Page 3694)"), you will receive the same error information in RET_VAL as for an unavailable DP slave/PROFINET IO device.

Deactivating a DP slave/PROFINET IO device does not start the program error OB, even if its inputs or outputs belong to the system-side process image to be updated.

If a DP station/PNIO station fails after you have deactivated it with "D_ACT_DP", the operating system does not detect the failure.

Applies to PROFIBUS DP: If you wish to deactivate DP slaves functioning as transmitters in slave-to-slave communication, we recommend that you first deactivate the receivers (listeners) that detect which input data the transmitter is transferring to its DP master. Deactivate the transmitter only after you have performed this step.

Activating DP slaves/PROFINET IO devices

When you reactivate a DP slave/PROFINET IO device with "D_ACT_DP", this component is configured and assigned parameters by the associated DP master/PROFINET IO controller (as with the return of a failed DP station/PROFINET IO station). This activation is complete when the component is able to transfer user data.

Activating a DP slave/PROFINET IO device does not start the program error OB, even if its inputs or outputs belong to the system-side process image to be updated.

If you try to activate a DP slave/PROFINET IO device with "D_ACT_DP" that cannot be accessed (for example, because it was physically separated from the bus), the instruction returns the error code W#16#80A7 after expiration of the configured parameter assignment time for distributed I/O. The DP slave/PROFINET IO device is activated and the fact that the activated DP slave/PROFINET IO device cannot be accessed results in a corresponding display in the system diagnostics.

If the DP slave/PROFINET IO device is accessible again afterwards, this results in standard system behavior (for example, a call of the OB configured for this purpose).

Note

Activating a DP slave/PROFINET IO device may be time-consuming. If you want to cancel a currently running activation job, start "D_ACT_DP" with the same value for LADDR and MODE =2. You repeat the call for "D_ACT_DP" with MODE = 2 until the successful cancellation of the activation job is displayed with RET_VAL =0.

If you wish to activate DP slaves which take part in the slave-to-slave communication, we recommend that you first activate the transmitters and then the receivers (listeners).

Parameters

The following table shows the parameters of the "D_ACT_DP" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Level-triggered control parameter REQ=1: Run activation or deactivation
MODE	Input	USINT	I, Q, M, D, L or constant	Job identifier Possible values: • 0: Request information on whether the addressed component is activated or deactivated (output via RET_VAL parameter) • 1: Activate the DP slave/PROFINET IO device • 2: Deactivate the DP slave/PROFINET IO device
LADDR	Input	HW_DEVICE	I, Q, M, D, L or constant	Hardware identifier of the DP slave (HW_DPSlave)/PROFINET IO device (HW_Device) The number can be taken from the properties of the DP slave / PROFINET IO device in the Network view or from the "System constants" tab of the standard tag table. If both the identifier for the device diagnostics as well as the ID for operating state transitions are both specified there, you must use the code for the device diagnostics.
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.
BUSY	Output	BOOL	I, Q, M, D, L	Active code: • BUSY=1: The job is still active. • BUSY=0: The job was terminated.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	Job completed without error.
0001	The DP slave/PROFINET IO device is active (this error code is possible only with MODE = 0.)
0002	The DP slave/PROFINET IO device is deactivated (this error code is possible only with MODE = 0.)

Error code* (W#16#...)	Explanation
7000	First call with REQ= 0: The job specified with LADDR is not active; BUSY has the value "0".
7001	First call with REQ= 1. The job specified with LADDRwas triggered. BUSY has the value "1".
7002	Intermediate call (REQ irrelevant). The activated job is still active; BUSY has the value "1".
8090	• You have not configured a module with the address specified in LADDR. • You operate your CPU as I-slave / I-device and you have specified an address of this I-slave/I-device in LADDR.
8092	The deactivation of the currently addressed DP slave/PROFINET IO device (MODE=2) cannot be canceled by being activated (MODE=1). Activate the component at a later time.
8093	The address specified in LADDR does not belong to any DP slave/PROFINET IO device that can be activated or deactivated or the MODE parameter is unknown.
8094	You have attempted to activate a device which is potential partner for a tool change port. However, another device is already activated on this tool change port at this time. The activated device will remain activated.
80A0	Error during the communication between the CPU and the IO controller.
80A3	The DP master/PROFINET IO controller concerned does not support this function.
80A4	The CPU does not support this function for external DP masters/PROFINET IO controller.
80A7	A timeout occurred during activation: The remote device is unreachable or you have set the parameter assignment time for central and distributed I/O too short. The status of the remote device is "activated", but it is not reachable.
80AA	Activation with errors in the DP slave/PROFINET IO device: Differences in the configuration
80AB	Activation with errors in the DP slave/PROFINET IO device: Parameter assignment error
80AC	Activation with errors in the DP slave/PROFINET IO device: Maintenance required
80C3	• Temporary resource error: The CPU is currently processing the maximum possible activation and deactivation jobs (8). • The CPU is busy receiving a modified configuration. Currently you can not enable/disable DP slaves/ PROFINET IO devices.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

ReconfigIOSystem: Reconfigure IO system (S7-1200, S7-1500)

ReconfigIOSystem: Reconfigure IO system (S7-1500)

Description

You can use the "ReconfigIOSystem" instruction to transfer a data record to the PROFINET interface of the CPU. This data record ("CTRLREC" parameter) contains the following information for controlling the configuration:

- A list of optional IO devices to be activated
- A list of the partner ports to be specified, if you have set the "Partner is set in the user program" option in the port properties of IO devices

Note

The "ReconfigIOSystem" instruction uses the "D_ACT_DP" instruction internally in MODE 1 and MODE 3 to enable/disable IO devices. Therefore, observe the rules and notes in the description of this instruction.

See also "D_ACT_DP: Disable/enable distributed I/O devices (Page 3673)".

Dependencies between instruction versions

Version V1.1 of instruction "ReconfigIOSystem" requires version V1.1 of the instruction "WRREC".

Version V1.0 of instruction "ReconfigIOSystem" requires version V1.1 of instruction "WRREC".

Functional description

"ReconfigIOSystem" is an asynchronous instruction. Processing takes place across several calls. You start the job by calling "ReconfigIOSystem" with REQ=1.

The output parameters STATUS and BUSY indicate the status of the job.

Parameters

The following table shows the parameters of the "ReconfigIOSystem" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Edge-triggered control parameter REQ=1: Perform data transfer
MODE	Input	UINT	I, Q, M, D, L or constant	You control how the instruction works with the MODE parameter. You can find a detailed functional description in the table below. Possible values: • 1: Disable all IO devices of the IO system for the conversion phase • 2: Reconfigure the IO system according to the data record settings (CTRLREC) • 3: Enable all IO devices of the IO system again after reconfiguration
LADDR	Input	HW_INTERFACE	I, Q, M, D, L or constant	Hardware identifier of the PROFINET interface (IO controller)
CTRLREC	Input	VARIANT	I, Q, M, D, L	Data record for controlling the actual configuration of the IO system
DONE	Output	BOOL	I, Q, M, D, L	0: Instruction not yet completed 1: Instruction completed
BUSY	Output	BOOL	I, Q, M, D, L	Active code: 0: Instruction active 1: Instruction completed

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	0: Instruction completed without error 1: Instruction completed with error
STATUS	Output	DWORD	I, Q, M, D, L	Result/error code
ERR_INFO	Output	WORD	I, Q, M, D, L	HW ID of the most recently determined IO device that caused an error.

You can find additional information on valid data types under "Overview of valid data types (Page 2239)".

Parameter MODE

The following values are possible for the MODE input parameter:

MODE	Description
1	All IO devices of the IO system can be disabled by calling the instruction with mode 1. The "ReconfigIOSystem" instruction uses the "D_ACT_DP" instruction internally. "ReconfigIOSystem" returns errors that are detected by D_ACT_DP in the following output parameters: • STATUS (error code) • ERR_INFO (hardware identifier of the IO device causing the error). In STATUS and ERR_INFO, the CPU enters the last determined error/HW identifier and in so doing overwrites an existing error code. For this reason, additional errors can be present besides the entered error.
2	For controlling the actual configuration of the IO system, the instruction transfers the data record to the PROFINET interface, which is addressed with LADDR. See the following section for information on the structure of the data record. See the following section about the STATUS parameter for error analysis.
3	All non-optional IO devices in the IO system and optional IO devices that are listed in the control data record CTRLREC are enabled. The optional IO devices that are **not** listed in the CTRLREC data record remain disabled. If IO devices that are part of docking units are listed in the CTRLREC control data record, the PN IO system reacts as follows: • IO devices of the docking units remain disabled when ReconfigIOSystem is called with Mode 3. This reaction corresponds to the reaction of a configuration without configuration-controlled IO devices. IO devices of docking units are disabled by default and must be enabled in the user program.

Structure of the control data record

You can use the control data record ("CTRLREC" parameter) to inform the PROFINET interface of the CPU which optionally configured IO devices are available in the real IO system configuration and which port interconnections are to be set.

This requires a configuration that allows IO system configuration to be adapted:

- The IO devices listed in the control data record must be enabled as "Optional IO device" (IO device properties: PROFINET interface [X1] > Advanced options > Interface options).
- The port interconnections listed in the control data record are only possible when the partner port is set to "Partner is set by the user program" for the appropriate ports.

The data type of parameter CTRLREC is "VARIANT". The "CTRLREC" control data record must have the following structure:

- Until version V1.1 of ReconfigIOSystem: Array of elements of the Word data type (for numerical addressing)
- As of version V1.2 of ReconfigIOSystem: Array of elements of the UInt data type (for symbolic or numerical addressing) or array of elements of the Word data type (for numerical addressing)

Below, the basic structural composition of "CTRLREC" is described with the Word element data type.

You read the required hardware identifiers directly in the "System constants" tab in the network view or in the device view. To do this, select the object in the network view (IO device) or in the device view (PROFINET interface).

Recommendation: Use the symbolic addressing of the HW identifiers by using the names of the HW identifiers.

Name	Data type	Comment
Version_High, Version_Low	Word	Version of the control data record(High byte: 01 Low byte: 00
Number_of_opt_Devices_used	Word	Number of optional IO devices that are used in the real IO system configuration. Optional IO devices that are not listed below remain disabled.
Activate_opt_Device_1	Word / Hw_Device	The optional IO devices that are present in the real configuration must be listed with their hardware identifiers. Use the system constant of the IO device object. Example: The IO device object has the name "IO-Device_4~IO-Device" and its type is "Hw_Device".
Activate_opt_Device_2	Word / Hw_Device	2nd optional IO device (e.g. with value 262)
...	...	...
Activate_opt_Device_n	Word / Hw_Device	nth optional IO device (e.g. with value 282)
Number_of_Port_Interconnections_used	Word	Number of port interconnections listed below. If you do not specify port interconnections, enter "0". For all the ports for which you have configured "Partner is set by the user program" and that are not listed below, the CPU uses the "Any partner" setting.
Port_Interconnection_1_Local	Word / Hw_Interface	1st port interconnection, HW identifier of local port Use the system constant of the port object. Example: The port object has the name "IO-Device_2~PROFINET_interface~Port_2" and its type is "Hw_Interface".
Port_Interconnection_1_Remote	Word / Hw_Interface	1st port interconnection, HW identifier of partner port
Port_Interconnection_2_Local	Word / Hw_Interface	2nd port interconnection, HW identifier of local port

Name	Data type	Comment
Port_Interconnection_2_Remote	Word / Hw_Interface	2nd port interconnection, HW identifier of partner port
...	...	...
Port_Interconnection_n_Local	Word / Hw_Interface	nth Port interconnection, hardware identifier local port
Port_Interconnectio_n_Remote	Word / Hw_Interface	nth Port interconnection, hardware identifier local port

Parameter STATUS

The STATUS output parameter contains error information. You can find a detailed list of possible error codes in the next section. If STATUS is interpreted as ARRAY[1...4] of BYTE, the error information has the following structure:

Field element	Name	Meaning
STATUS[1]	Function_Num	B#16#DF: Error writing data record, otherwise B#16#00.
STATUS[2]	Error_Decode	B#16#80 is output for an error (corresponding to IEC 61158-6 in the context of reading and writing data records).
STATUS[3]	Error_Code_1	B#16#AA in the case of an error in the structure of the data record
STATUS[4]	Error_Code_2	Manufacturer-specific error ID extension: • B#16#00 Error in the data record (e.g. incorrect padding byte values (not equal to 0) • B#16#01 Reserved • B#16#02 At least one station number for an IO device in the data record is invalid (not configured, points to a non-optional IO device or has the value 0 for an IO controller) • B#16#03 At least one partner port specified in the data record is invalid: Examples: – Subslot address of a partner port is not available – Partner port is configured incorrectly (the correct setting is: "Partner is set by the user program") – Partner port belongs to an IO device that is disabled • B#16#10 Version of the CTRLREC control data record invalid (specified version is not supported) • B#16#11 The number of optional IO devices to be activated in the CTRLREC control data record is not supported • B#16#12 The number of specified interconnections ("Partner set in the user program") in the CTRLREC control data record is not supported • B#16#13 Internal conversion of the hardware identifier to device number failed. The ERR_INFO output parameter contains the hardware identifier of the device causing the error. • B#16#14 Consistency error: The length of the CTRLREC control data record does not match the information in the control data record. Example: 20 optional IO devices are specified, but the control data record only has a length of 10 bytes.

Error codes (STATUS parameter)

Error code	Description
16#0000_0000	Job completed without error
16#0070_0000	No job active
16#0070_0100	First call of the instruction
16#0070_0200	Following call of the instruction (instruction is still running, BUSY = 1)
16#0080_8000	MODE is not supported
16#0084_5100	Incorrect data type of CRTLREC data record. Use an array of UInt or an array of Word.
16#0080_9100	The LADDR parameter does not address any PROFINET interface (does not exist or wrong type, e.g. PROFIBUS interface). The PROFINET interface does not support configuration control of IO systems.
16#0080_Cx00	Temporary error, e.g., due to temporary lack of resources.
16#DF80_AAxy	Error in the structure of the data record (MODE 2). For the meaning of "xy", see the definition of STATUS[4] Error_Code_2 above.
16#DF80_B6xy	Configuration control not possible because either no optional IO device was configured or no port was assigned with "Partner is set in the user program". This configuration is a prerequisite for calling the instruction. "xy" is irrelevant.
16#0080_9400 16#0080_A000 16#0080_A700 16#0080_AA00 16#0080_AB00 16#0080_AC00	Forwarded error codes of the internally called D_ACT_DP instruction. For the meaning of these error codes, see "D_ACT_DP (Page 3673)". The hardware identifiers of IO devices causing errors are entered in ERR_INFO (entry is consecutively overwritten by subsequent errors). If more than one IO device is involved, online diagnostics with STEP 7 is recommended. With IRT configuration: The device numbers of the IO devices should follow from the IO controller in ascending order of their topological interconnection, see here.

Example

You will find the example here: Program example for ReconfigIOSystem (Page 3683).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

Program example for ReconfigIOSystem (S7-1200, S7-1500)

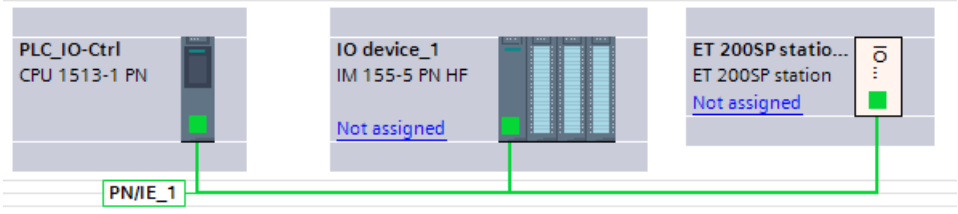
Introduction

In the following example, you configure an IO system via the program using the control data record "CTRLREC". The target configuration should consist of an S7-1500 CPU, an IO device (for example ET 200MP) and an optional IO device (for example, ET 200SP). For this purpose, disable an existing IO system, transfer the configuration values to the PROFINET interface of the CPU and enable the IO devices.

You can find additional information on the data records in Overview of the types of data records (Page 3571).

Requirements

- An S7-1500 CPU has been created.
- Two ET 200 have been created.
- A PROFINET connection has been created and the hardware settings have been made:
  - IO device (for example, ET 200MP IM 155-5 PN HF)
  - optional IO device (for example, ET 200SP IM 155-6 PN ST
  - IO system is set up by the program



Note

Hardware settings for an IO system to be set up by the program

Proceed as follows:

1. Connect the two ET 200 via PROFINET with the CPU.
2. Assign the two ET 200 as IO controllers to the CPU. Then open the CPU properties.
3. Open the entry "PROFINET interface [X1] > Advanced options > Port X1 P1 > Port interconnection > Partner port". Select the option "Set Partner by user program". Do the same for "Port X1 P2".
4. Open the properties of ET 200MP (...> Port X1 P1) and select the option "Set Partner by user program" (see step 3).
5. Open the properties of ET 200SP (...> Port X1 P1) and perform the following settings:
   - Select the option "Set Partner by user program" (see step 3).
   - Open "PROFINET interface [X1] > Advanced options > Interface options" and select the option "Optional IO device".

Storage of data

To store the data, create the two structures "operateMode" and "ctrlRec" and 13 additional tags in a global data block.

The structure "ctrlRec" contains the control data record "CTRLREC".

SLI_gDB_ReconfigIOSystem

	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	startDone	Bool	false
4	reset	Bool	false
5	getExecute	Bool	false
6	operateMode	Struct	
7	value	UInt	1
8	Disable_all_IO_devices_done	Bool	false
9	Reconfigure_IO_system_done	Bool	false
10	Enable_all_IO_devices_done	Bool	false
11	laddr_interface	HW_INTERFACE	64
12	ctrlRec	Struct	
13	Version	Word	16#0100
14	Number_of_opt_Devices_used	Word	1
15	Activate_opt_Device_1	HW_DEVICE	264
16	Number_of_Port_Interconnections_used	Word	2
17	Port_Interconnection_1_Local	HW_INTERFACE	65
18	Port_Interconnection_1_Remote	HW_INTERFACE	259
19	Port_Interconnection_2_Local	HW_INTERFACE	66
20	Port_Interconnection_2_Remote	HW_INTERFACE	273
21	status	DWord	16#0
22	busy	Bool	false
23	done	Bool	false
24	error	Bool	false
25	errorDevice	Word	16#0
26	memErr	Bool	false
27	memErrDevice	Word	16#0
28	memErrStatus	DWord	16#0

Note

You have to specify the values of the hardware identifiers ("laddr_interface", "ctrlRec") according to your system.

For example, determine the hardware identifier of the interface of the CPU ("laddr_interface") as follows:

In the CPU, open the entry "PLC tags > Show all tags > System constants" and search for the name <Local~PROFINET-Interface_1> with the data type "Hw_Interface". The associated cell "value" contains the hardware identifier.

Note

Pay attention to the interfaces used.

- The Port X1 P1 of the IO device (ET 200MP/"<IO_device_1>") is connected with the port X1 P1 of the CPU ("<Local>").
- The Port X1 P1 of the optional IO device (ET 200SP/"<IO_device_2>") is connected with the port X1 P2 of the CPU ("<Local>"). You also require the hardware identifier of the device (data type "Hw_Device") from the optional IO device.

35	Local	Hw_SubModule	257	
36	Local~Common	Hw_SubModule	50	
37	Local(1)	Hw_SubModule	49	
38	OB_Main	OB_PCYCLE	1	
39	Local~Display	Hw_SubModule	54	
40	Local~MC	Hw_SubModule	51	
41	Local~Exec	Hw_SubModule	52	
42	Local~PROFINET_interface_1	Hw_Interface	64	-->laddr
43	Local~PROFINET_interface_1~Port_1	Hw_Interface	65	-->Port 1 local
44	Local~PROFINET_interface_1~Port_2	Hw_Interface	66	-->Port 2 local
45	Local~PROFINET_IO-System	Hw_IoSystem	258	
46	IO_device_1~Proxy	Hw_SubModule	267	
47	IO_device_1~IODevice	Hw_Device	265	
48	IO_device_1~PROFINET_interface	Hw_Interface	268	
49	IO_device_1~PROFINET_interface~Port_1	Hw_Interface	259	-->Port 1 remote
50	IO_device_1~PROFINET_interface~Port_2	Hw_Interface	260	
51	IO_device_1~Head	Hw_SubModule	261	
52	IO_device_1~DI_16x24VDC_HF_1	Hw_SubModule	262	
53	IO_device_1~DQ_8x24VDC_2A_HF_1_1	Hw_SubModule	263	
54	IO_device_2~Proxy	Hw_SubModule	270	
55	IO_device_2~IODevice	Hw_Device	264	-->Used as optional
56	IO_device_2~Head	Hw_SubModule	271	IO device
57	IO_device_2~PROFINET_interface	Hw_Interface	272	
58	IO_device_2~PROFINET_interface~Port_1	Hw_Interface	273	-->Port 2 remote
59	IO_device_2~PROFINET_interface~Port_2	Hw_Interface	274	
60	IO_device_2~DQ_8x24VDC_0_5A_HF_1	Hw_SubModule	276	

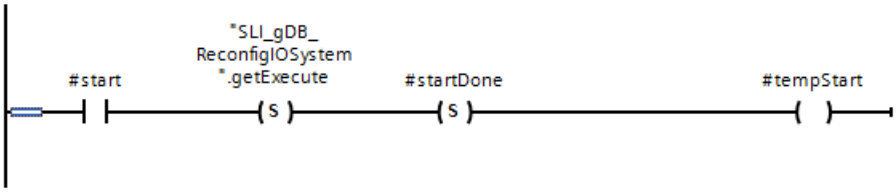
Interconnecting parameters in the FC "SLI_FC_start_reset_ReconfigIOS"

To start or reset the instruction "ReconfigIOSystem", create the FC "SLI_FC_start_reset_ReconfigIOS". To pass on the data, create the following local tags in the FC.

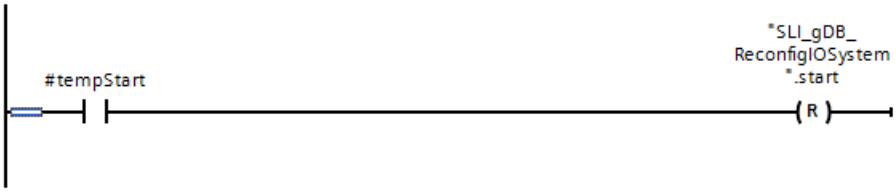
SLI_FC_start_reset_ReconfigIOS

	Name	Data type
1	Input	
2	start	Bool
3	reset	Bool
4	Output	
5	<Add new>	
6	InOut	
7	startDone	Bool
8	Temp	
9	tempStart	Bool
10	resetDone	Bool

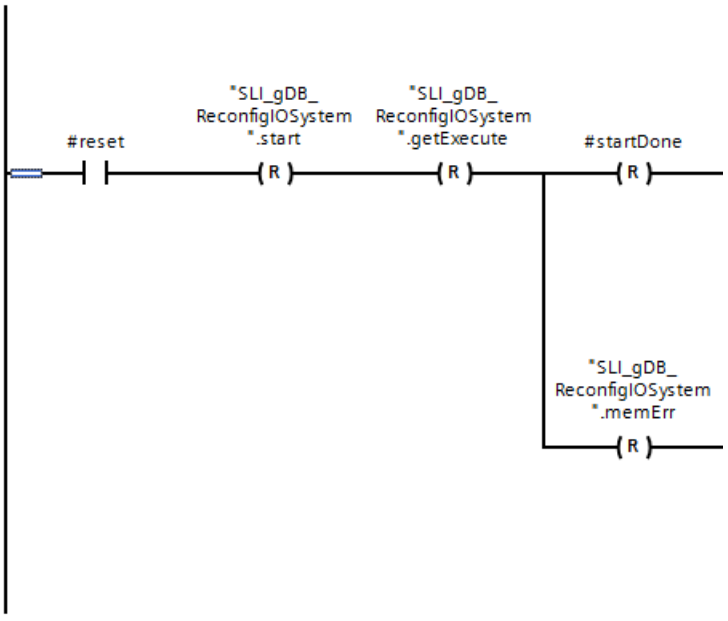
Network 1: To set the input parameter REQ ("getExecute"), interconnect the tag "start".



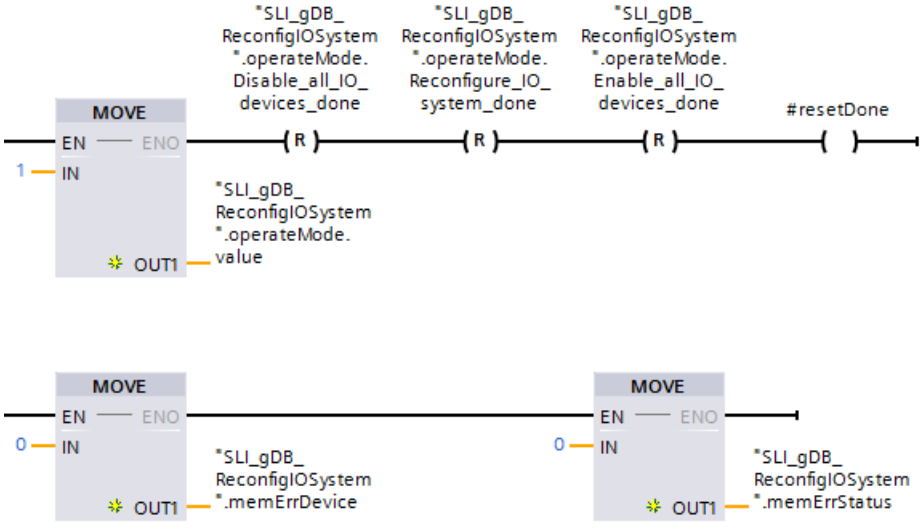
Network 2: To reset "start", create the following interconnections.



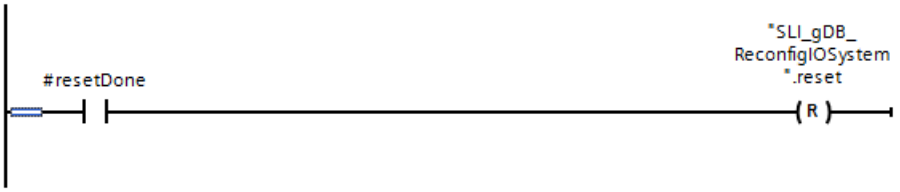
Network 3: To reset all set process values, create the following interconnections. Part 1 of the network:



Part 2 of the network:



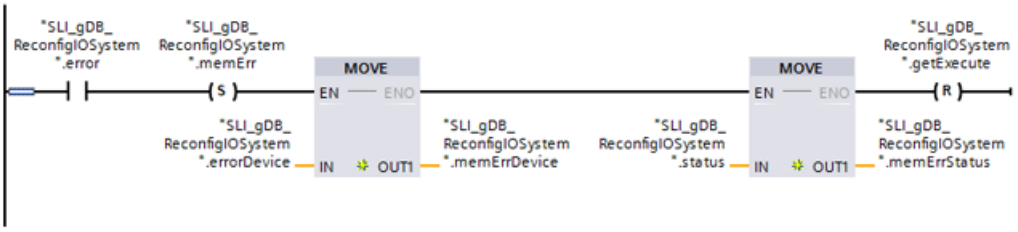
Network 4: To reset "reset", create the following interconnections.



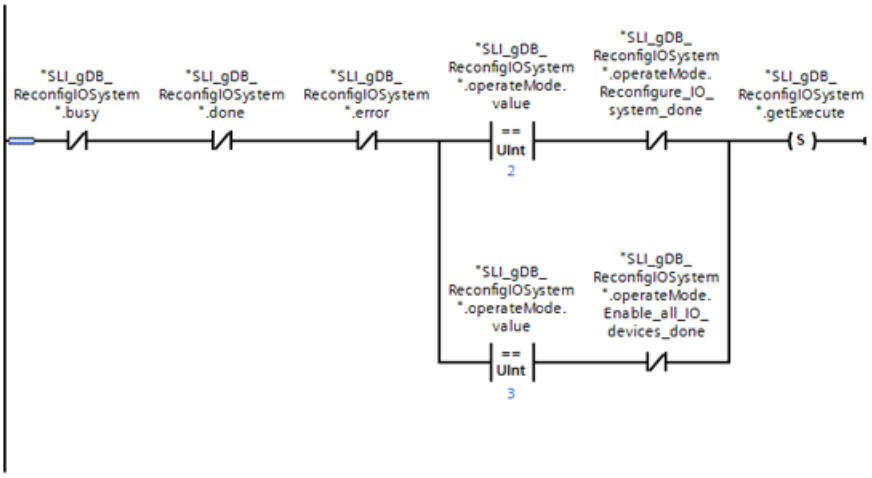
Interconnecting parameters in the FC "SLI_FC_switchMode_ReconfigIOS"

To safely reconfigure the IO system, run through all three modes of "ReconfigIOSystem". You need to set an automatism for mode switchover in the FC "SLI_FC_switchMode_ReconfigIOS" so that you do not have to switch the modes of the "ReconfigIOSystem" instruction yourself.

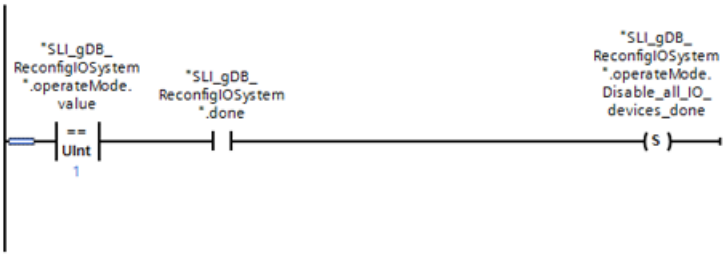
Network 1: To save the information in the case of error, create the following interconnections.



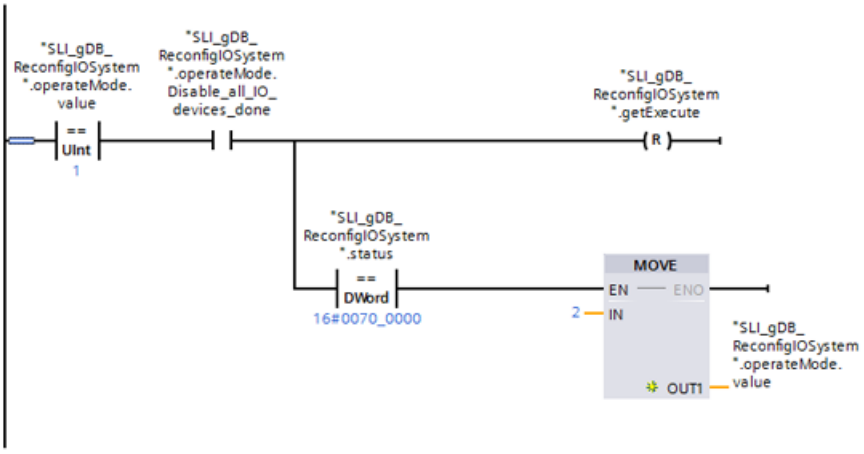
Network 2: To automatically start the instruction "ReconfigIOSystem" in mode 2 or mode 3, create the following interconnections.



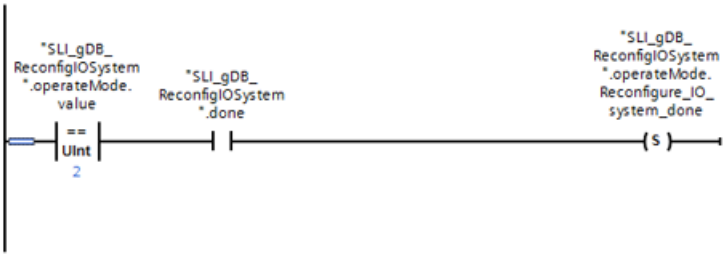
Network 3: Record the success of the processing of ReconfigIOSystem in mode 1 as follows.



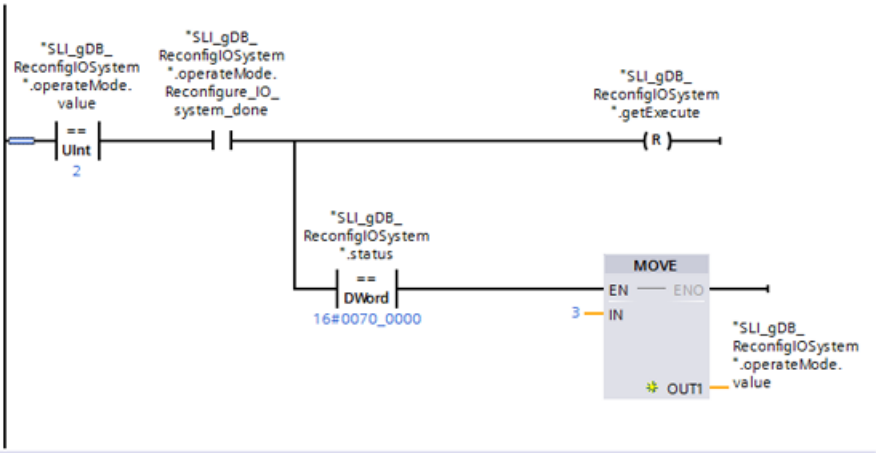
Network 4: To safely switch from mode 1 to mode 2, create the following interconnections.



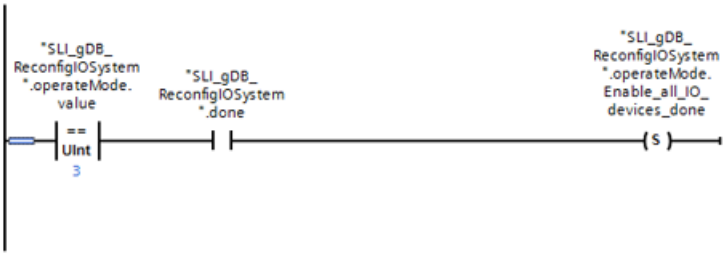
Network 5: Record the success of the processing of ReconfigIOSystem in mode 2 as follows.



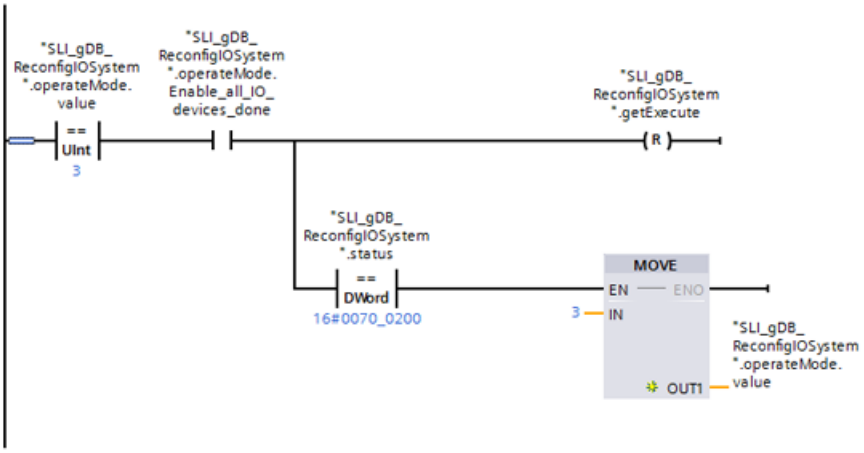
Network 6: To safely switch from mode 2 to mode 3, create the following interconnections.



Network 7: Record the success of the processing of ReconfigIOSystem in mode 3 as follows.



Network 8: To end the processing of ReconfigIOSystem, create the following interconnections.



Note

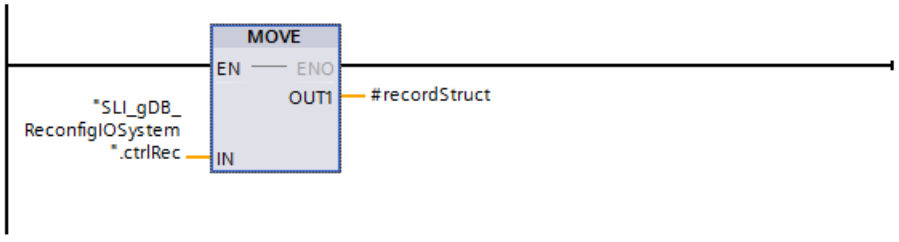
Alternatively, you can switch from mode 3 to mode 1 after processing. For this purpose, enter the value "1" for the instruction "MOVE".

Interconnecting parameters in the FB "SLI_FB_ReconfigIOSystem"

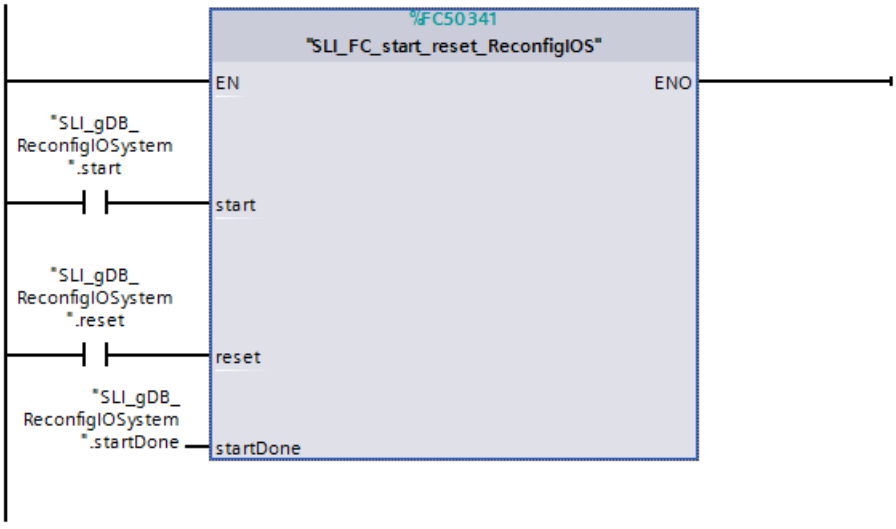
Call the instruction "ReconfigIOSystem" in an FB ("SLI_FB_ReconfigIOSystem"). To pass on the data, create the following local tags in the FB. Create the tags "#recordAoWord" and "#recordStruct" as AT constructs. The tags must be the same size (as "ctrlRec").

7		Static			
8		ReconfigIOSystem_Instance_1	ReconfigIOSyste...		
9		recordAoWord	Array[0..7] of Word		Set in IDB
10		recordStruct AT "recordAoWord"	Struct		Set in IDB
11		Version	Word		Set in IDB
12		Number_of_opt_Devices_used	Word		Set in IDB
13		Activate_opt_Device_1	HW_DEVICE		Set in IDB
14		Number_of_Port_Interconnections_...	Word		Set in IDB
15		Port_Interconnection_1_Local	HW_INTERFACE		Set in IDB
16		Port_Interconnection_1_Remote	HW_INTERFACE		Set in IDB
17		Port_Interconnection_2_Local	HW_INTERFACE		Set in IDB
18		Port_Interconnection_2_Remote	HW_INTERFACE		Set in IDB

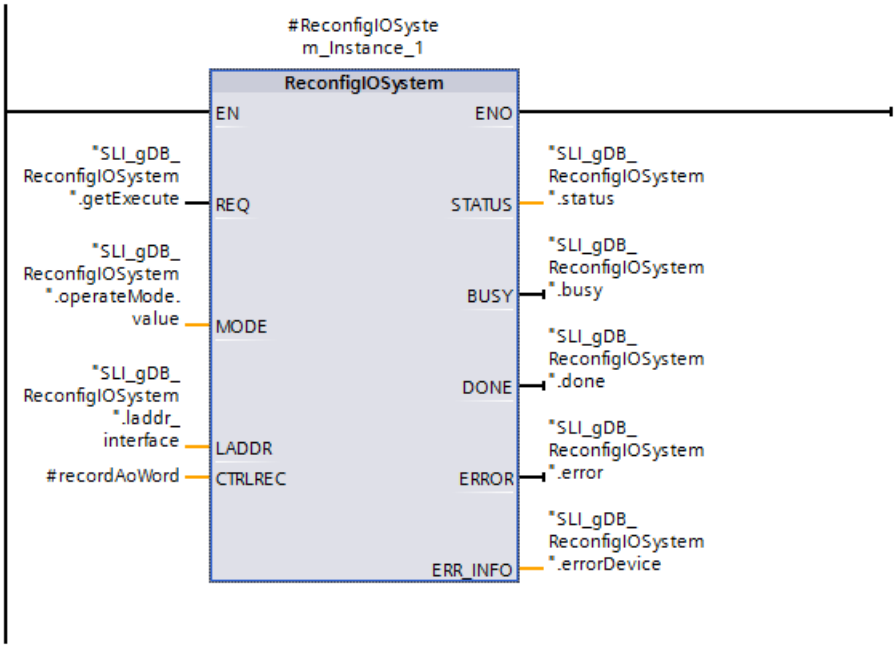
Network 1: To transfer the data record ("ctrlRec"), create the following interconnection.



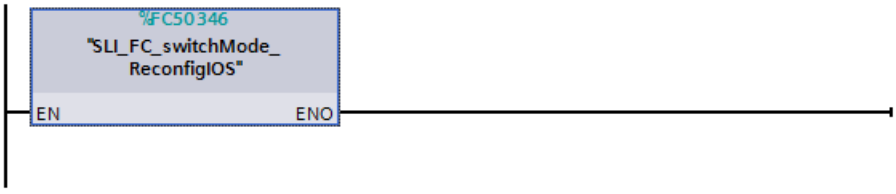
Network 2: Interconnect the FC "SLI_FC_start_reset_ReconfigIOS" as follows.



Network 3: Interconnect the instruction "ReconfigIOSystem" as follows.



Network 4: Interconnect the FC "SLI_FC_switchMode_ReconfigIOS" as follows.



Call the FB ("SLI_FB_ReconfigIOSystem") in an OB1.

Result (FB "SLI_FB_ReconfigIOSystem")

FB Network 1:

The values of the data record ("ctrlRec") are transferred to an Array of WORD ("#recordAoWord") via a MOVE instruction. To improve understanding, the tags of the "ctrlRec" and "#recordStruct" structures are represented by descriptive names.

FB Network 2:

When the input parameter start ("start") returns the signal state "TRUE", the tag "getExecute" and the output parameter startDone ("startDone") are set to "TRUE". The processing of the instruction "ReconfigIOSystem" is started by the tag "getExecute" at "TRUE".

When the input parameter reset ("reset") returns the signal state "TRUE", the start command of "ReconfigIOSystem", the process values of the processing and the information of the fault scenario are reset.

FB Network 3:

When the input parameter REQ ("getExecute") returns the signal state "TRUE", the "ReconfigIOSystem" instruction is started. The instruction "ReconfigIOSystem" calls the interface of the CPU via the input parameter LADDR ("laddr_interface").

The mode for the operating mode is transferred to the instruction "ReconfigIOSystem" via the input parameter MODE ("operateMode.value"). ReconfigIOSystem disables all existing IO systems of the CPU according to the start value "1". The block status is displayed at the output parameter STATUS ("status"). In the example, processing takes place without errors.

The "ReconfigIOSystem" instruction only calls the data record at the input parameter CTRLREC ("#recordAoWord") in mode 2. The configuration values for the IO system of ReconfigIOSystem to be set up are communicated to the CPU (IO controller).

In mode 3, the "ReconfigIOSystem" instruction enables the IO system. The switchover of the input parameters REQ ("getExecute") and MODE ("operateMode.value") is implemented in network 4 ("SLI_FC_switchMode_ReconfigIOS").

SLI_gDB_ReconfigIOSystem

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	FALSE
3	startDone	Bool	false	TRUE
4	reset	Bool	false	FALSE
5	getExecute	Bool	false	FALSE
6	operateMode	Struct		
7	value	UInt	1	3
8	Disable_all_IO_devices_done	Bool	false	TRUE
9	Reconfigure_IO_system_done	Bool	false	TRUE
10	Enable_all_IO_devices_done	Bool	false	TRUE
11	laddr_interface	HW_INTERFACE	64	16#0040
12	ctrlRec	Struct		
13	Version	Word	16#0100	16#0100
14	Number_of_opt_Devices_used	Word	1	16#0001
15	Activate_opt_Device_1	HW_DEVICE	264	16#0108
16	Number_of_Port_Interconnections_used	Word	2	16#0002
17	Port_Interconnection_1_Local	HW_INTERFACE	65	16#0041
18	Port_Interconnection_1_Remote	HW_INTERFACE	259	16#0103
19	Port_Interconnection_2_Local	HW_INTERFACE	66	16#0042
20	Port_Interconnection_2_Remote	HW_INTERFACE	273	16#0111
21	status	DWord	16#0	16#0070_0000
22	busy	Bool	false	FALSE
23	done	Bool	false	FALSE
24	error	Bool	false	FALSE
25	errorDevice	Word	16#0	16#0000
26	memErr	Bool	false	FALSE
27	memErrDevice	Word	16#0	16#0000
28	memErrStatus	DWord	16#0	16#0000_0000

FB Network 4:

If the instruction "ReconfigIOSystem" was successfully processed in mode 1 (the tag "done" has the value "TRUE"), the tag "operateMode.Disable_all_IO_devices_done" is set to "TRUE". As a result, the input parameter REQ ("getExecute") is then reset to "FALSE". As soon as ReconfigIOSystem is inactive ("status" has the value "16#0070_0000"), the value "2" is passed to the tag "operateMode.value". ReconfigIOSystem is then in mode 2.

In mode 2, the input parameter REQ ("getExecute") is automatically set to "TRUE" if the following applies to the instruction "ReconfigIOSystem":

- ReconfigIOSystem is inactive ("busy" and "done" is "FALSE").
- No error occurs ("error" is "FALSE").
- The success state of mode 2 is not reached ("operateMode.Reconfigure_IO_system_done" is "FALSE").

If the instruction "ReconfigIOSystem" was successfully processed in mode 2 (the tag "done" has the value "TRUE"), the tag "operateMode.Reconfigure_IO_system_done" is set to "TRUE". Afterwards, the scenario is similar to with mode 1.

- The "getExecute" tag is reset to "FALSE".
- ReconfigIOSystem ("operateMode.value") is set to mode 3.

In mode 3 the input parameter REQ ("getExecute") is automatically set to "TRUE" if the same applies to the "ReconfigIOSystem" instruction as for mode 2 - with one exception:

- The success state of mode 3 is not reached ("operateMode.Enable_all_IO_devices_done" is "FALSE").

The success status in mode 3 (the tag "done" has the value "TRUE") is queried in the same way as for mode 2. After the successful processing ReconfigIOSystem remains in mode 3 ("operateMode.value" retains the value).

After the completed processing of the three modes, the "ReconfigIOSystem" instruction is inactive ("status" has the value "16#0070_0000"). The IO system is set up: An S7-1500 CPU with an IO device and an optional IO device. Both IO devices are activated.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

ReconfigIOSystem: Reconfigure IO system (Page 3677)

Others (S7-1200, S7-1500)

RD_REC: Read data record from I/O (S7-1500)

Description

Use the instruction to read the data record with the number RECNUM from the addressed module. You start the read process by assigning the value "1" to the input parameter REQ during the call. If the read process is executed immediately, the instruction returns the value "0" at the output parameter BUSY . If BUSY has the value "1", reading is not yet complete.

See also: Difference between synchronous and asynchronous instructions (Page 2571). The data record read is entered in the target range spanned by the RECORD parameter, providing the data transfer was free of errors.

Note

When you read out a data record with a number greater than one from an FM or a CP you have purchased prior to February 1997 (below referred to as "old modules"), "RD_REC" responds differently than it does in new modules. This special situation is covered in the section "Using old S7-300 FMs and CPs with data record numbers >1" (see below).

If a DPV1 slave is configured via GSD file (GSD rev. 3 and higher) and the DP interface of the DP master is set to "S7 compatible", then you may not read any data records from the I/O modules in the user program with "RD_REC". In this case, the DP master addresses the wrong slot (configured slot + 3).

Remedy: Set the interface of the DP master to "DPV1".

Parameters

The following table shows the parameters of the "RD_REC" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = 1: Read request
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the module.
RECNUM	Input	BYTE	I, Q, M, D, L or constant	Data record number (permitted values: 0 to 240)
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code. In addition: Length of the actually transferred data record in bytes (possible values: +1 to +240), if the target range is greater than the transferred data record and if no error occurred during the transfer.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The reading process is not yet complete.
RECORD	Output	ANY	I, Q, M, D, L	Target range for the data record read. With asynchronous execution of "RD_REC", make sure that the actual parameters of RECORD have the same length information for all calls. Only the BYTE data type is permitted. Note: Note that for S7-300 CPUs, the parameter RECORD always requires the full specification of the DB parameters (for example, P#DB13.DBX0.0 byte 100). The omission of an explicit DB number is illegal for S7-300 CPUs and leads to an error message in the user program.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RECORD

Note

If you want to ensure that the entire data record is always read, select a target range with a length of 241 bytes. If the data transfer is without error, RET_VAL contains the actual data record length.

Using old S7-300 FMs and CPs with data record numbers > 1

If you want to use the instruction "RD_REC" to read out a data record with a number greater than one from an old S7-300 FM or old S7-300 CP, remember the following points:

- If the target range is greater than the actual length of the required data record, no data is entered in RECORD . RET_VAL is written with W#16#80B1.
- If the target range is less than the actual length of the required data record, the CPU reads as many bytes beginning at the start of the record as are specified in the length information of RECORD and enters this number of bytes in RECORD . RET_VAL is written with "0".
- If the length specified in RECORD is the same as the actual length of the required data record, the CPU reads the data record and enters it in RECORD . RET_VAL is written with "0".

Parameter RET_VAL

- If an error occurred while the function was being executed, the return value contains an error code.
- If no error occurred during the transfer, RET_VAL contains the following:
  - 0, if the entire target range was filled with data from the selected data record (the data record can also be incomplete).
  - The length of the actually transferred data record in bytes (possible values: +1 to +240), if the target range is greater than the transferred data record.

Note

If the general error W#16#8745 occurs, this only indicates that access to at least one byte of the process image was blocked. The data record was read by the module correctly and written to the I/O memory area.

When looking at the "real" error information (error codes W#16#8xyz) in the following table, two different cases are described:

- Temporary errors (error codes W#16#80A2 to 80A3, 80Cx):
  It is possible that this type of error is eliminated without user action. For this reason, it is advisable to call the instruction again (multiple calls, if necessary).
  Example of a temporary error: Resources required are currently in use (W#16#80C3).
- Permanent errors (error codes W#16#809x, 80A0, 80A1, 80Bx):
  This type of error does not correct itself. Once you have eliminated the error, the instruction needs to be called again. Example of a permanent error: Wrong length specification in RECORD (W#16#80B1).

Note

If you transfer data records to a DPV1 slave with "WR_REC (Page 3699)" or if you read data records from a DPV1 slave with RD_REC and if this DPV1 slave operates in DPV1 mode, the DP master evaluates the error information it received from the slave as follows:

If the error information lies within the range from W#16#8000 to W#16#80FF or W#16#F000 to W#16#FFFF, the DP master passes the error information to the instruction. If the error information is outside this range, the DP master passes the value W#16#80A2 to the instruction and suspends the slave.

For a description of the error information originating from DPV1 slaves, refer to STATUS[3] Parameter STATUS (Page 3638).

Parameter RET_VAL for WR_REC and RD_REC

Error code* (W#16#...)	Explanation	Restriction
0000	No error	-
7000	First call with REQ=0: No data transfer active; BUSY has the value 0.	-
7001	First call with REQ=1: Data transfer started; BUSY has the value 1.	Distributed I/O
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value 1.	Distributed I/O
8090	Address specified at parameter ADDR is invalid.	-
8092	A type other than BYTE is specified in the ANY reference.	-
8093	This instruction is not permitted for the module selected by means of LADDR and IOID .	-
80A0	Negative acknowledgment when reading from the module: Module was removed during the reading process or is defective	With "RD_REC" only
80A1	Negative acknowledgment when writing to the module: Module was removed during the writing process or is defective	With "WR_REC (Page 3699)" only
80A2	• DP protocol error at layer 2 (for example, slave failure or bus problems) • For ET 200S, data record cannot be read in DPV0 mode.	Distributed I/O
80A3	DP protocol error with user interface/user	Distributed I/O
80A4	Communication on PROFIBUS disrupted.	-

Error code* (W#16#...)	Explanation	Restriction
80B0	• Instruction not possible for module type. • The module does not recognize the data record. • Data record number 241 not permitted. • With "WR_REC (Page 3699)", data records 0 and 1 are not permitted.	-
80B1	The length specified in parameter RECORD is incorrect.	• With "WR_REC (Page 3699)": Length incorrect • With "RD_REC" (only when using old S7-300 FMs and S7-300 CPs): specified length > data record length • With DPNRM_DG: specified length < data record length
80B2	The configured slot is not occupied.	-
80B3	Actual module type does not match specified module type	-
80B5	DP slave or module is not ready.	-
80B7	DP slave or module reports an invalid range for a parameter or value.	With "RD_REC" only
80C0	With "WR_REC (Page 3699)": The data is only written when the CPU is in STOP mode. Note: this means that writing by the user program is not possible. You can only write the data online with PG/PC. With "RD_REC": the module routes the data record, but either no data is present or the data can only be read when the CPU is in STOP mode. Note: if data can only be read when the CPU is in STOP mode, then an evaluation by the user program is not possible. In this case, you can only read the data online with PG/PC. With "DPNRM_DG (Page 3715)": No diagnostics data available.	With "WR_REC (Page 3699)" or "RD_REC" or "DPNRM_DG (Page 3715)"
80C1	The data of the previous write job on the module for the same data record has not yet been processed by the module.	-
80C2	The module is currently processing the maximum possible number of jobs for a CPU.	-
80C3	The required resources (memory, etc.) are currently occupied.	-
80C4	Internal temporary error. Job could not be executed. Repeat the job. If this error occurs often, check your installation for sources of electrical interference.	-
80C5	Distributed I/O not available.	Distributed I/O
80C6	Data record transfer stopped due to priority class abort (restart or background)	Distributed I/O
80Ex	Manufacturer-specific. Please check the documentation of the module.	-

Error code* (W#16#...)	Explanation	Restriction
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)	-
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

WR_REC: Write data record to I/O (S7-1500)

Description

Use the instruction "WR_REC" to transfer the data record RECORD to the addressed module.

The data to be transferred are read from the RECORD parameter during the first call. If the transfer of the data record takes longer than the duration of one call, the contents of the RECORD parameter are no longer relevant for the subsequent instruction calls (for the same job).

You start the writing process by assigning the value "1" to the input parameter REQ during the call. If the writing process could be executed immediately, the instruction returns the value "0" at the output parameter BUSY. If BUSY has the value "1", writing is not yet complete.

Parameters

The following table shows the parameters of the instruction "WR_REC":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, T, C or constant	REQ = 1: Write request
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the module.
RECNUM	Input	BYTE	I, Q, M, D, L or constant	Data record number
RECORD	Input	VARIANT	I, Q, M, D, L	Data record
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The writing process is not yet complete.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

See also: RD_REC: Read data record from I/O (Page 3694)

Note

If the general error W#16#8544 occurs, it only indicates that access to at least one byte of the I/O memory area containing the data record was denied. The data transfer was continued.

DPRD_DAT: Read consistent data of a DP standard slave (S7-1200, S7-1500)

Description

Use the instruction "DPRD_DAT" to read out consistent data from an I/O module.

The instruction can be used for the central modules as well as for the DP standard slaves and PROFINET IO devices.

You require "DPRD_DAT" because you can only read out a maximum of four continuous bytes using the load commands that access the I/O or the process image input table. If required, you can also read consistent data via the process image of the inputs. Refer to the related documentation to find out if your CPU supports this functionality. For additional information on consistent data of a DP standard slave/PROFINET IO device, refer to Section "Data consistency (Page 4383)".

If necessary, use the instruction "DPRD_DAT" also for a data area of 1 byte or larger. For information on the maximum length of the data, refer to the documentation of your CPU (e.g. 64 bytes for an S7-1214).

- Use the parameter LADDR to select the module of the DP normslave/PROFINET IO device. If an access error occurs, the error code W#16#8090 is output.
- Use the RECORD parameter to define the target range for the read data:
  - The target range has to be at least as long as the inputs of the selected module. If the target range is longer than the inputs of the module, it is written from the front with the values of the inputs; with S7-1200 CPUs the remaining bytes of the target range remain unchanged, with S7-1500 CPUs the remaining bytes are filled with zeros. If you read from a DP standard slave with a modular configuration or with several DP identifiers, you can only access the data of a module of the configured hardware identifier per "DPRD_DAT" call. If the selected target range is too small, the error code RET_VAL is output at the 80B1 parameter.
  - The following data types can be used: BOOL, BYTE, CHAR, WCHAR, WORD, LWORD, DWORD, INT, UINT, USINT, SINT, LINT, ULINT, DINT, UDINT, REAL, LREAL. The use of these data types in a data structure of the type ARRAY or STRUCT is also permissible.
  - The STRING and WSTRING data types are not supported.
  - PLC data types (UDT) and system data types are supported.
- If there was no error during the data transmission, the data that has been read is entered in the target range defined at the RECORD parameter.

Parameters

The following table shows the parameters of the instruction "DPRD_DAT":

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_IO	I, Q, M, L or constant	Hardware ID of the module from which the data is to be read. The hardware ID can be found in the properties of the module in the device view or system constants.
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.
RECORD	Output	VARIANT	I, Q, M, D, L	Target range for the user data that were read.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8090	• You have not configured a module for the specified hardware identifier or • You have ignored the restriction concerning the length of consistent data, or • you have not specified a hardware identifier as an address at parameter LADDR .
8092	A data type other than (Array of) bit string or integer was specified at the RECORD parameter.
8093	No DP module/PROFINET IO device from which you can read consistent data exists for the hardware identifier specified in LADDR . This error code also occurs if the module addressed by means of LADDR does not have inputs.
80A0	An access error was detected when accessing the I/O.
80B1	The length of the specified target range at parameter RECORD is shorter than the configured user data length.
80C0	The data have not been read yet.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

See also

Parameter STATUS (Page 3638)

DPWR_DAT: Write consistent data of a DP standard slave (S7-1200, S7-1500)

Description

Use the instruction "DPWR_DAT" to transfer the data at the parameters RECORD consistently to the addressed module of the central module or of the DP standard slave/PROFINET IO device and if necessary to the process image (i.e. if the affected address range of the DP standard slaves is projected as consistency range in a process image).

You require "DPWR_DAT" because you can only write a maximum of four continuous bytes using the transfer commands that access the I/O or process image output. If required, also write consistent data via the process image output. Refer to the related documentation to find out if your CPU supports this functionality. Do not use both possibilities concurrently when writing consistent data: Either use "DPWR_DAT" or write via the process image output table. For additional information on consistent data of a DP standard slave/PROFINET IO device, refer to Section "Data consistency (Page 4383)". If the DP standard slave has a modular design, you only access one module of the DP slave.

⚠ CAUTION

I/O access

When using "DPWR_DAT", avoid accessing I/O areas that have process image partitions with OB6x connections (isochronous mode interrupts) assigned to them.

If necessary, the instruction "DPRD_DAT" can also be used for a data area of 1 byte or larger. For information on the maximum length of the data, refer to the documentation of your CPU (e.g. 64 bytes for an S7-1214).

- Use the LADDR parameter to select the DP standard slave / PROFINET IO device. If an access error occurs on the addressed module, the error code 8090 is output.
- Use the RECORD parameter to define the source range of the data to be written:
  - The source range has to be at least as long as the outputs of the selected module. Only the outputs are transferred, the other bytes are not considered. If the source area at parameter RECORD is longer than the outputs of the configured module, only the data up to the maximum length of the outputs is transferred. If the source area at parameter RECORD is shorter than the outputs of the configured module, the error code 80B1 is output.
  - The following data types can be used: BOOL, BYTE, CHAR, WCHAR, WORD, LWORD, DWORD, INT, UINT, USINT, SINT, LINT, ULINT, DINT, UDINT, REAL, LREAL. The use of these data types in a data structure of the type ARRAY or STRUCT is also permissible.
  - The STRING and WSTRING data types are not supported.
  - PLC data types (UDT) and system data types are supported.

The data transfer is synchronous. Once the instruction is finished, the write process is completed.

Parameters

The following table shows the parameters of the instruction "DPWR_DAT":

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_IO	I, Q, M, L or constant	Hardware ID of the module to which the data is to be written. The hardware ID can be found in the properties of the module in the device view or system constants.
RECORD	Input	VARIANT	I, Q, M, D, L	Source area for the user data to be written.
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8090	• You have not configured a module for the specified hardware identifier or • You have ignored the restriction concerning the length of consistent data, or • you have not specified a hardware identifier at parameter LADDR .
8092	A data type other than (Array of) bit string or integer was specified at the RECORD parameter.
8093	No DP module /PROFINET IO device to which you can write consistent data exists at the HW ID specified in LADDR . This error code also occurs if the DP standard slave / PROFINET IO device addressed via LADDR does not have outputs.
80A1	Access error detected while I/O devices were being accessed.
80B1	The length of the specified source range at the RECORD parameter is shorter than the outputs of the configured DP standard slave / PROFINET IO device.
80C1	The data of the previous write job have not yet been processed by the DP standard slave / PROFINET IO device.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

See also

Parameter STATUS (Page 3638)

iDevice / iSlave (S7-1200, S7-1500)

RCVREC: Receive data record (S7-1200, S7-1500)

Description

An I-device can receive a data record from a superordinate controller. The receipt takes place in the user program with the instruction "RCVREC" (receive record).

The instruction has the following operating modes:

- Check whether the I-device has a request for a data record receipt.
- Make the data record available to the output parameters.
- Sending an answer to the superordinate controller.

You can determine the operating mode executed by the instruction using the input parameter MODE (see below).

The I-device must be in the RUN or STARTUP mode.

With MLEN, you specify the maximum number of bytes you want to receive. Select the destination area RECORD at least MLEN bytes long.

If a data record was received (MODE=1 or MODE=2), the output parameter NEW will indicate that the data record was stored in RECORD. Note that RECORD has a sufficient length. The output parameter LEN contains the actual length of the data record received in bytes.

Set CODE1 and CODE2 to zero for the positive answer to the superordinate controller. If the received data record is rejected, enter Error Code 1 in CODE1 and Error Code 2 in CODE2 of the negative answer at the superordinate controller.

Note

If the I-device has received a request for a data record receipt, you must recognize the delivery of this request within a certain duration. After recognition, you must send an answer to the superordinate controller within this time period. Otherwise the I-device will experience a timeout error which causes the operating system of the I-device to send a negative answer to the superordinate controller. For information on the value for the time period, refer to the specifications of your CPU.

The output parameter STATUS receives the error information after the occurrence of an error.

Operating modes

You can determine the operating mode of the instruction "RCVREC" with the input parameter MODE. This step will be explained in the following table.

MODE	Meaning
0	Check whether a request for a data record receipt exists If a data record from a higher-level controller exists on the I-device, the instruction will only write the output parameters NEW, SLOT, SUBSLOT, INDEX and LEN. If you call the instruction several times with MODE=0, then the output parameter will only refer to one and the same request.
1	Receiving a data record for any subslot of the I-device If a data record from a superordinate controller exists on the I-device for any subslot of the I-device, the instruction will write the output parameter and transfer the data record to the parameter RECORD.
2	Receiving a data record for a specific subslot of the I-device If a data record from a superordinate controller exists on the I-device for a specific subslot of the I-device, the instruction will write the output parameter and transfer the data record to the parameter RECORD.
3	Sending a positive answer to the superordinate controller The instruction checks the request of the superordinate controller to receive a data record, accepts the existing data record and sends a positive acknowledgment to the superordinate controller.
4	Sending a negative answer to the superordinate controller The instruction checks the request of the superordinate controller to receive a data record, rejects the existing data record and sends a negative acknowledgment to the superordinate controller. Enter the reason for the rejection in the input parameters CODE1 and CODE2.

Note

After the receipt of a data record (NEW=1) you must call "RCVREC" twice to ensure complete processing. You must do this in the following order:

- First call with MODE=1 or MODE=2
- Second call with MODE=3 or MODE=4

Parameters

The following table shows the parameters of the "RCVREC" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	INT	I, Q, M, D, L or constant	Mode
F_ID	Input	HW_SUBMODULE	I, Q, M, D, L or constant	Subslot in the transfer area of the I-device for the data record to be received (only relevant for MODE=2). The high word is always set to zero.
MLEN	Input	INT*	I, Q, M, D, L or constant	Maximum length of the data record to be received in bytes
CODE1	Input	BYTE	I, Q, M, D, L or constant	"0" (for MODE=3) and/or Error Code 1 (for MODE=4)
CODE2	Input	BYTE	I, Q, M, D, L or constant	"0" (for MODE=3) and/or Error Code 2 (for MODE=4)

Parameter	Declaration	Data type	Memory area	Description
NEW	Output	BOOL	I, Q, M, D, L	• MODE=0: New data record was received • MODE=1 or 2: Data record was transferred to RECORD
STATUS	Output	DWORD	I, Q, M, D, L	Error information
SLOT	Output	HW_SUB-MODULE	I, Q, M, D, L	Identical to F_ID
SUBSLOT	Output	HW_SUB-MODULE	I, Q, M, D, L	Identical to F_ID
INDEX	Output	UINT	I, Q, M, D, L	Number of the data record received
LEN	Output	UINT	I, Q, M, D, L	Length of the data record received
RECORD	InOut	VARIANT	I, Q, M, D, L	Destination area for the data record received. Note: Note that for S7-300 CPUs, the parameter RECORD always requires the full specification of the DB parameters (for example, P#DB13.DBX0.0 Byte 100) The omission of an explicit DB number is illegal for S7-300 CPUs and leads to an error message in the user program.
* Use the data type UINT in the STL programming language.				

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

For interpretation of the STATUS parameter, refer to section: Parameter STATUS (Page 3638)

PRVREC: Make data record available (S7-1200, S7-1500)

Description

An I-device can receive a request from a superordinate controller to make a data record available. The I-device makes the data record available in the user program with the instruction "PRVREC" (provide record).

The instruction has the following operating modes:

- Check whether the I-device has a request for making a data record available.
- Transfer the requested data record to the superordinate controller.
- Sending an answer to the superordinate controller.

You can determine the operating mode executed by the instruction using the input parameter MODE (see below).

The I-device must be in the RUN or STARTUP mode.

Enter the maximum number of bytes the data record to be sent should have with LEN. Select the destination area RECORD at least LEN bytes long.

If a request to make a data record available exists, (MODE=0), the output parameter NEW will be set to TRUE.

If the request for making a data record available is accepted, write RECORD for the positive answer to the superordinate controller with the requested data record and write zero for CODE1 and CODE2. If the request for making a data record available is to be rejected, enter the negative answer to the superordinate controller in Error Code 1 of the CODE1 and in Error Code 2 of theCODE2.

Note

If the I-device has received a request for making a data record available, you must recognize the delivery of this request within a certain time period. After recognition, you must send an answer to the superordinate controller within this time period. Otherwise the I-device will experience a timeout error which causes the operating system of the I-device to send a negative answer to the superordinate controller. For information on the value for the time period, refer to the specifications of your CPU.

The output parameter STATUS receives the error information after the occurrence of an error.

Operating modes

You can determine the operating mode of the instruction "PRVREC" with the input parameter MODE. This step will be explained in the following table.

MODE	Meaning
0	Check whether a request for making a data record available exists If a request from a higher-level controller for making a data record available exists on the I-device, the instruction will only write the output parameters NEW, SLOT, SUBSLOT, INDEX and RLEN. If you call the instruction several times with MODE=0, then the output parameter will only refer to one and the same request.
1	Receiving a request for making a data record available for any subslot of the I-device If such a request from a superordinate controller for any subslot of the I-device exists on the I-device, the instruction will write the output parameter.
2	Receiving a request for making a data record available for a specific subslot of the I-device If such a request from a superordinate controller for a specific subslot of the I-device exists on the I-device, the instruction will write the output parameter.
3	Make the data record available and send a positive answer to the superordinate controller The instruction checks the request of the superordinate controller to make a data record available, makes the request data record available to RECORD and sends a positive acknowledgement to the superordinate controller.
4	Sending a negative answer to the superordinate controller The instruction checks the request of the superordinate controller to make a data record available, rejects this request and sends a negative acknowledgement to the superordinate controller. Enter the reason for the rejection in the input parameters CODE1 and CODE2.

Note

After the receipt of a request (NEW=1) you must call the instruction twice to ensure complete processing. You must do this in the following order:

- First call with MODE=1 or MODE=2
- Second call with MODE=3 or MODE=4

Parameters

The following table shows the parameters of the "PRVREC" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	INT	I, Q, M, D, L or constant	Mode
F_ID	Input	HW_SUBMODULE	I, Q, M, D, L or constant	Subslot in the transfer area of the I-device for the data record to be sent (only relevant for MODE=2). The high word is always set to zero.
CODE1	Input	BYTE	I, Q, M, D, L or constant	"0" (for MODE=3) and/or Error Code 1 (for MODE=4)
CODE2	Input	BYTE	I, Q, M, D, L or constant	"0" (for MODE=3) and/or Error Code 2 (for MODE=4)
LEN	Input	UINT	I, Q, M, D, L or constant	Maximum length of the data record to be sent in bytes
NEW	Output	BOOL	I, Q, M, D, L	The new data record was requested by the superordinate controller.
STATUS	Output	DWORD	I, Q, M, D, L	Error information
SLOT	Output	HW_SUBMODULE	I, Q, M, D, L	Identical to F_ID
SUBSLOT	Output	HW_SUBMODULE	I, Q, M, D, L	Identical to F_ID
INDEX	Output	UINT	I, Q, M, D, L	Number of the data record to be sent
RLEN	Output	UINT	I, Q, M, D, L	Length of the data record to be sent
RECORD	InOut	VARIANT	I, Q, M, D, L	Data record made available Note: Note that for S7-300 CPUs, the parameter RECORD always requires the full specification of the DB parameters (for example, P#DB13.DBX0.0 Byte 100) The omission of an explicit DB number is illegal for S7-300 CPUs and leads to an error message in the user program.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

For interpretation of the STATUS parameter, refer to section: Parameter STATUS (Page 3638)

PROFIBUS (S7-1200, S7-1500)

DPSYC_FR: Synchronize DP slaves / Freeze inputs (S7-1500)

Description

You use the instruction to synchronize one or more groups of DP slaves.

The function involves sending one of the control commands below, or a combination of them, to the relevant groups:

- SYNC (simultaneous output and freezing of output states on the DP slaves)
- UNSYNC (cancels the SYNC control command)
- FREEZE (freeze the input states on the DP slaves and read in the frozen inputs)
- UNFREEZE (cancels the FREEZE control command)

Before you send the control commands listed above, you must assign the DP slaves to groups per configuration. You must know which DP slave is assigned to which group, with which number and know the reactions of the various groups to SYNC/FREEZE.

Note

Note that the control commands SYNC and FREEZE also remain valid when you perform a warm/cold restart.

Please note also that you may initiate only one SYNC/UNSYNC job or only one FREEZE/ UNFREEZE job at a time.

Functional description

"DPSYC_FR" is an asynchronous instruction. Processing takes place across several calls. You start the job by calling "DPSYC_FR" with REQ=1.

The output parameters RET_VAL and BUSY indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571).

Identification of a job

If you have triggered a SYNC/FREEZE job and called "DPSYC_FR" again before the first job was completed, the response of the instruction depends on whether the new call is for the same job. If the input parameters LADDR, GROUP and MODE match, the call is interpreted as a follow-on call.

Writing outputs of DP modules

The writing of outputs of DP modules is triggered as follows:

- By transfer commands to the DP I/O
- By writing the process image of the outputs to the modules (by the operating system at the end of OB 1 or by calling the instruction "UPDAT_PO (Page 3565)")
- Calling the "DPWR_DAT (Page 3702)" instruction.

In normal operation, the DP master transfers the output bytes cyclically (within the cycle of the PROFIBUS DP bus) to the outputs of the DP slaves.

If you want to have certain output data (possibly distributed over several slaves) applied to the outputs to the process at exactly the same time, you can send the SYNC control command to the relevant DP master using the "DPSYC_FR" instruction.

What are the effects of SYNC?

With the SYNC control command, the DP slaves of the selected groups are switched to Sync mode. The DP master transfers the current output data and instructs the DP slaves involved to freeze their outputs. With the following output message frames, the DP slaves enter the output data in an internal buffer and the state of the outputs remains unchanged.

Following each SYNC control command, the DP slaves of the selected groups apply the output data of their internal buffer to the outputs on the process.

The outputs are only updated cyclically again when you send the UNSYNC control command using the "DPSYC_FR" instruction.

Note

If the DP slaves of the selected group(s) are not currently connected to the network or have failed when the control command was sent, they will not be switched to SYNC mode. This information will not be communicated in the return value of the instruction.

Reading input data of DP modules

The input data of the DP modules are read as follows:

- Using load commands to the DP I/O
- When the process image of the inputs is updated (by the operating system at the start of OB 1 or by calling the "UPDAT_PI (Page 3562)" instruction)
- By calling the "DPRD_DAT (Page 3700)" instruction.

In normal operation, the DP master receives this input data cyclically (within the cycle of the PROFIBUS DP bus) from its DP slaves and makes them available to the CPU.

If you want to have certain input data (possibly distributed over several slaves) to be read from the process at exactly the same time, send the FREEZE control command to the relevant DP master using the "DPSYC_FR" instruction.

What are the effects of FREEZE?

With the FREEZE control command, the DP slaves involved are switched to Freeze mode. The DP master instructs the DP slaves involved to freeze the current state of the inputs. It then transfers the frozen data to the input area of the CPU.

Following each FREEZE control command, the DP slaves freeze the state of their inputs again.

The DP master receives the current state of the inputs cyclically again after you have sent the UNFREEZE control command with the "DPSYC_FR" instruction.

Note

If the DP slaves of the selected group(s) are not currently connected to the network or have failed when the control command has been sent, they will not be switched to FREEZE mode. This information will not be communicated in the return value of the instruction.

Data consistency

Because the DPSYC_FR functions are asynchronous and can be interrupted by higher priority classes, you should make sure that the process images are consistent with the actual inputs and outputs of the I/O when using the "DPSYC_FR" instruction.

This is guaranteed if you comply with the following consistency rules:

- Define suitable process image partitions for the "SYNC outputs" and the "FREEZE inputs" (only possible on the S7-400). Call the "UPDAT_PO (Page 3565)" instruction immediately before each first call of a SYNC job. Call the "UPDAT_PI (Page 3562)" instruction immediately after the respective last call of a FREEZE job.
- As an alternative: Use only direct I/O access for outputs involved in a SYNC job and for inputs involved in a FREEZE job. Do not write to these outputs when a SYNC job is active and do not read in these inputs when a FREEZE job is active.

Use of DPWR_DAT and DPRD_DAT

If you use the "DPWR_DAT (Page 3702)" instruction, it must be complete before you send a SYNC job for the outputs involved.

If you use the "DPRD_DAT (Page 3700)" instruction, it must be complete before you send a FREEZE job for the inputs involved.

Startup and "DPSYC_FR"

The user alone must take responsibility for sending the SYNC and FREEZE control commands in the startup OBs.

If you want the outputs of one or more groups to be in SYNC mode when the user program starts, you must initialize these outputs during startup and completely execute the "DPSYC_FR" instruction with the SYNC control command.

If you want the inputs of one or more groups to be in FREEZE mode when the user program starts, you must execute the "DPSYC_FR" instruction with the FREEZE control command completely for these inputs during startup.

Parameters

The following table shows the parameters of the "DPSYC_FR" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Level-triggered control parameter REQ = 1: Triggering of the SYNC/FREEZE job
LADDR	Input	HW_INTERFACE	I, Q, M, D, L or constant	Hardware identifier of the DP master interface The number can be taken from the properties of the DP master interface in the Network view or from the "System constants" tab of the standard tag table.
GROUP	Input	BYTE	I, Q, M, D, L or constant	Group selection Bit 0 = 1: group 1 selected Bit 1 = 1: group 2 selected : Bit 7 = 1: group 8 selected You can select several groups per job. The value B#16#0 is invalid.

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	BYTE	I, Q, M, D, L or constant	Job ID (coding complies with EN 50 170 Volume 2, PROFIBUS) Bit 0: reserved (value 0) Bit 1: reserved (value 0) Bit 2: • = 1: UNFREEZE will be executed • = 0: no meaning Bit 3: • = 1: FREEZE will be executed • = 0: no meaning Bit 4: • = 1: UNSYNC will be executed • = 0: no meaning Bit 5: • = 1: SYNC will be executed • = 0: no meaning Bit 6: reserved (value 0) Bit 7: reserved (value 0) Possible values: • with exactly one ID per job: – B#16#04 (UNFREEZE) – B#16#08 (FREEZE) – B#16#10 (UNSYNC) – B#16#20 (SYNC) • with more than one ID per job: – B#16#14 (UNSYNC, UNFREEZE) – B#16#18 (UNSYNC, FREEZE) – B#16#24 (SYNC, UNFREEZE) – B#16#28 (SYNC, FREEZE)
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code. Make sure that you evaluate RET_VAL each time the block has been executed.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY=1: The SYNC/FREEZE job is not yet complete.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Note

If you access DPV1 slaves, error information of these slaves can be forwarded from the DP master to the instruction. For a description of this error information, refer to STATUS[3], STATUS (Page 3638) parameter.

Error code* (W#16#...)	Explanation
0000	Job completed without error.
7000	First call with REQ= 0: The job specified with LADDR , GROUP and MODE is not active; BUSY has the value "0".
7001	First call with REQ= 1. The job specified with LADDR , GROUP and MODE was triggered; BUSY has the value "1".
7002	Intermediate call (REQ irrelevant). The activated SYNC/FREEZE job is still running; BUSY has the value 1.
8090	The module selected with LADDR is not a DP master.
8093	This instruction is not permitted for the module selected with LADDR (configuration or version of the DP master).
8094	GROUP parameter is incorrect
8095	MODE parameter is incorrect
80A4	Communication on PROFIBUS disrupted.
80B0	The group selected with GROUP is not configured.
80B1	The group selected with GROUP is not assigned to this CPU.
80B2	The SYNC job specified with MODE is not permitted on the group selected with GROUP.
80B3	The FREEZE job specified with MODE is not permitted on the group selected with GROUP .
80C2	Temporary shortage of resources on the DP master: The DP master is currently processing the maximum number of jobs for a CPU.
80C3	This SYNC/UNSYNC job cannot be activated at present because only one SYNC/UNSYNC job can be triggered at a time. Check your user program.
80C4	This FREEZE/UNFREEZE job cannot be activated at present because only one FREEZE/UNFREEZE job can be triggered at a time. Check your user program.
80C5	Short circuit directly at DP interface
80C6	Job aborted due to I/O disconnection by CPU
80C7	Job aborted due to warm or cold restart on the DP master
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

DPNRM_DG: Read diagnostics data from a DP slave (S7-1200, S7-1500)

Description

You use the "DPNRM_DG" instruction to read the current diagnostic data of a DP slave in the form specified in EN 50 170 Volume 2, PROFIBUS.

Refer to the following table for the basic structure of the slave diagnostic data and to the manuals of the DP slaves for further information.

Byte	Meaning
0	Station status 1
1	Station status 2
2	Station status 3
3	Master station number
4	Vendor ID (high byte)
5	Vendor ID (low byte)
6 ...	Additional slave-specific diagnostic information

The data that has been read is entered in the destination area indicated by RECORD following without error data transfer. You start the read process by assigning the value "1" to the REQ input parameter when the "DPNRM_DG" instruction is called.

Functional description

The read process is executed asynchronously. It can span several calls. The output parameters RET_VAL and BUSY indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

Parameters

The following table shows the parameters of the "DPNRM_DG" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = 1: Read request
LADDR	Input	HW_DPSLAVE	I, Q, M, D, L or constant	Hardware identifier of the DP slave Note: The identifier must be specified in hexadecimal form. For example, hardware identifier 1022 means: LADDR:=W#16#3FE.
RET_VAL	Return	DINT, INT, LREAL, REAL	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code. If no error has occurred, the length of the data actually transferred is entered in RET_VAL.

Parameter	Declaration	Data type	Memory area	Description
RECORD	Output	VARIANT	I, Q, M, D, L	Destination area for the diagnostic data that were read. Only the data type BYTE is valid. The minimum length of the data record to be read or the destination area is 6. The maximum length of the data record to be read is 240. For standard slaves, which have a diagnostic data length of more than 240 bytes up to a maximum of 244 bytes, the first 240 bytes are transferred to the destination area. Then, the corresponding overflow bit is set in the data.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The reading process is not yet complete.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

You can find information on data type conversion in the different programming languages under "Overview of data type conversions (Page 2531)".

Parameter RECORD

The CPU evaluates the actual length of the read diagnostic data:

If the length specified for RECORD

- is less than the number of data bytes supplied, the data is discarded. The corresponding error code is entered in RET_VAL.
- is greater than or equal to the number of data bytes supplied, the data is accepted in the destination area and the actual length is entered in RET_VAL as a positive value.

Note

You must ensure that the actual parameters of RECORD correspond in all calls belonging to a job.

A job is uniquely identified by the LADDR input parameter.

Standard slaves with more than 240 bytes of diagnostic data

With standard slaves on which the number of standard diagnostic data is between 241 and 244 bytes, note the following points:

If the length specified for RECORD

- is less than 240 bytes, the data is discarded and the corresponding error code is entered in RET_VAL.
- is greater than or equal to 240 bytes, the first 240 bytes of the standard diagnostic data is transferred to the destination area and the overflow bit is set in the data.

Parameter RET_VAL

- If an error occurred while the function was being executed, the return value contains an error code.
- If no errors have occurred during data transfer, RET_VAL contains the length of the data read in bytes as a positive number.

Note

The amount of data read in a DP slave depends on its diagnostics status.

For the evaluation of the error information of the RET_VAL parameter, refer to the following table.

Error code (W#16#...)	Explanation	Restriction
7000	First call with REQ = 0: No data transfer active; BUSY has the value "0".	-
7001	First call with REQ = 1: Data transfer triggered; BUSY has the value "1".	Distributed I/O
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value "1".	Distributed I/O
8090	Address specified at parameter LADDR is invalid.	-
8093	This instruction is not permitted for the module selected by means of LADDR and IOID.	-
80A2	• DP protocol error at layer 2 (for example, slave failure or bus problems) • For ET 200S, data record cannot be read in DPV0 mode.	Distributed I/O
80A3	DP protocol error with user interface/user	Distributed I/O
80A4	Communication on PROFIBUS disrupted.	Distributed I/O
80B0	• Instruction not possible for module type. • The module does not recognize the data record. • Data record number 241 not permitted. • With "WR_REC (Page 3596)", data records 0 and 1 are not permitted.	-
80B1	The length specified in parameter RECORD is incorrect.	specified length < data record length
80B2	The configured slot is not occupied.	-
80B3	Actual module type does not match specified module type	-
80C0	There are no diagnostic data available.	-
80C1	The data of the previous write job on the module for the same data record has not yet been processed by the module.	-
80C2	The module is currently processing the maximum possible number of jobs for a CPU.	-
80C3	The required resources (memory, etc.) are currently occupied.	-

Error code (W#16#...)	Explanation	Restriction
80C4	Internal temporary error. Job could not be executed. Repeat the job. If this error occurs often, check your installation for sources of electrical interference.	-
80C5	Distributed I/O not available.	Distributed I/O
80C6	Data record transfer stopped due to priority class abort (restart or background)	Distributed I/O
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)	-
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

See also

Overview of data type conversions (Page 2531)

DP_TOPOL: Determine topology for DP master system (S7-1500)

Description

You use the instruction to trigger the topology determination for a selected DP master system. Calling the instruction will address all diagnostics repeaters on a DP master system.

Note

The topology can only be determined for one DP master system at a time.

Topology determination is the prerequisite for the detailed display of the error location if line errors occur. After configuration and after every change to the physical configuration of a DP master system, you must repeat the topology determination with "DP_TOPOL".

Changes to the physical configuration are:

- Changes in line lengths
- Adding or removing stations or components with repeater function
- Changing station addresses

If an error is reported by a diagnostics repeater, "DP_TOPOL" writes the DPR and DPRI outputs for the duration of one "DP_TOPOL pass. If errors are reported by multiple diagnostics repeaters of the selected DP master system, "DP_TOPOL" writes information on the first error-reporting diagnostics repeater to DPR and DPRI . You can read out the entire diagnostic information on the programming device or using the "DPNRM_DG (Page 3715)" instruction. If no diagnostics repeater reports an error, the DPR and DPRI outputs have the value "0".

If you want to repeat a topology determination after an error occurs, you must first reset "DP_TOPOL". This is done by calling "DP_TOPOL" with REQ=0 and R=1.

Functional description

"DP_TOPOL" is an asynchronous instruction. Processing takes place across several calls. You start determining the bus topology by calling "DP_TOPOL" with REQ=1 . If you want to cancel the process, call "DP_TOPOL" with R=1 .

The output parameters RET_VAL and BUSY indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

Note

Determining the topology may take several minutes.

Identification of a job

The input parameter DP_ID uniquely specifies a job.

If you call "DP_TOPOL" again before the topology is discovered, the manner in which the instruction reacts depends on whether the new call involves the same job: If the DP_ID parameter matches a job that has not yet been completed, then the call is interpreted as a follow-up call. The value W#16#7002 is entered in RET_VAL. On the other hand, if another job is involved, the CPU rejects it.

Parameters

The following table shows the parameters of the "DP_TOPOL" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ=1: Trigger topology determination
R	Input	BOOL	I, Q, M, D, L or constant	R=1: Cancel topology determination
DP_ID	Input	HW_IOSYSTEM	I, Q, M, D, L or constant	DP master system ID of those master systems for which the topology will be determined
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY=1: Topology determination is not yet complete.
DPR	Output	BYTE	I, Q, M, D, L	PROFIBUS address of the diagnostics repeater reporting the error.
DPRI	Output	BYTE	I, Q, M, D, L	Measuring segment diagnostics repeater reporting the error: • Bit 0 = 1: Temporary faults on segment DP2 • Bit 1 = 1: Permanent faults on segment DP2 • Bit 4 = 1: Temporary faults on segment DP3 • Bit 5 = 1: Permanent faults on segment DP3

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

The "real" error information (error codes W#16#8xyz) can be divided into two classes:

- Temporary errors (error codes W#16#80A2 to 80A4, 80C3, 80C5):
  It is possible that this type of error is eliminated without user action. It is advisable to call "DP_TOPOL" again (multiple calls, if necessary).
  Example of a temporary error: Resources required are currently in use (W#16#80C3).
- Permanent errors (error codes W#16#8082, 80B0, 80B2):
  With this error type, you must resolve the error. A new call of "DP_TOPOL" is required once you have eliminated the error. Example of a permanent error: The DP master/CPU does not support this service. (W#16#80B0).

Error code* (W#16#...)	Explanation
0000	Job completed without error.
7000	First call with REQ=0. Topology determination is not triggered. BUSY has the value "0".
7001	First call with REQ=1. The request to execute topology determination was sent. BUSY has the value "1".
7002	Intermediate call (REQ irrelevant): Topology determination is not yet complete. BUSY has the value "1".
7010	You have attempted to cancel topology determination. But there is no running job with the specified DP_ID. BUSY has the value "0".
7011	First call with R=1. The cancellation of the topology determination was triggered. BUSY has the value "1".
7012	Intermediate call: The cancellation of the topology determination is not yet complete. BUSY has the value "1".
7013	Final call: The topology determination was cancelled. BUSY has the value "0".
8082	No DP master system is configured with the specified DP_ID.
80A2	Error during topology determination; for more detailed information, refer to output parameters DPR and DPRI.
80A3	Error during topology determination: Monitoring time has elapsed (timeout).
80A4	Communication on PROFIBUS disrupted.
80B0	The DP master/CPU does not support this service.
80B2	Error during topology determination: No diagnostics repeater was found at the selected DP master system.
80C3	Resources required are currently in use. Possible cause: You have initiated a second topology determination (only one topology determination is permitted at one time).
80C5	The DP master system is currently not available.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

ASi (S7-1500)

ASI_CTRL: Control AS-i master behavior (S7-1500)

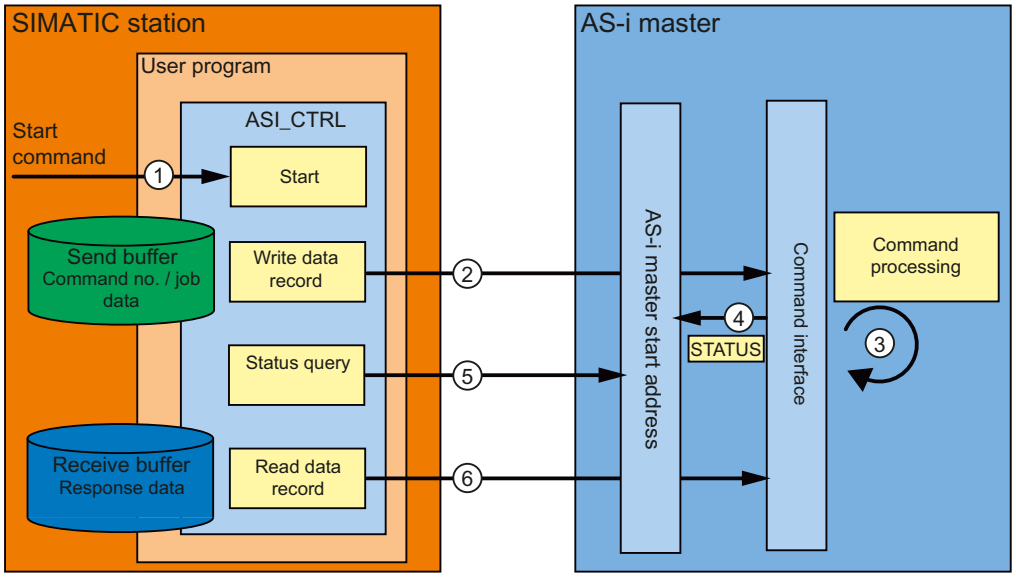
Description of ASI_CTRL (S7-1500)

Description

Using the instruction "ASI_CTRL", the behavior of the AS-i master is controlled from the user program of the PLC. The instruction processes the command protocol automatically. It also enables parameter assignment on SIMATIC AS-i masters and reading out information data. The functions and operation of the command interface are detailed in the manual for the AS-i master.

Both centrally inserted AS-i masters and distributed AS-i masters via PROFIBUS DP are supported. Combinations with PROFINET IO (e.g. IE/PB Link PN IO) are also possible.

The schematic diagram below shows the functions of the "ASI_CTRL" instruction:



① Start of processing at the REQ parameter.

② The program sends the required command to the AS-i master via the instruction "RDREC".

③ The AS-i master processes the command.

④ The current status of the AS-i master is stored in the input area of the binary data (logical start address).

⑤ The instruction "ASI_CTRL" cyclically queries and evaluates the 4 status bits.

⑥ Once command processing is complete, the command job is completed with "RDREC". Depending on the command, the data field of "RDREC" may contain the response data for the command or additional status information.

Dependencies between instruction versions

Version V1.3 of instruction "ASI_CTRL" requires version V1.1 of instruction "WRREC".

Version V1.2 of instruction "ASI_CTRL" requires version V1.1 of instruction "WRREC".

Differences in the command call between IE/ AS-i Link and DP/AS-i Links

There are significant differences in how a controller exchanges commands with an AS-i master.

- **IE/AS-i Link** (PROFINET) used the data record interface. The different commands are called with either "Write data record" ("WRREC" instruction) or "Read data record" ("RDREC" instruction) by various data record numbers.
- **DP/AS-i Links** (PROFIBUS) use the command interface. All commands are called by data record number 2 with "Write data record" ("WRREC" instruction) plus "Read data record" ("RDREC" instruction). The type of command is defined by the data content in the write job.

Operation of the "ASI_CTRL" instruction

The instruction "ASI_CTRL" is an asynchronous function block. Processing takes place across several calls.

- A job is started with REQ = TRUE.
- The job status is displayed via the BUSY output parameters and the two central bytes of the output parameter STATUS.
- The BUSY parameter is set during job processing. Upon the first call, STATUS is assigned the value 00700100_H. Upon all subsequent calls for this job, it is assigned the value 00700200_H. When the job is complete, the result is output at the parameters DONE or ERROR.
  - If no errors have occurred, DONE is set. For jobs with response data from the AS-i master, this data is provided in the specified receive buffer. In such cases, the STATUS parameter also displays the volume of data supplied in bytes. For jobs without response data, the value 00000000_H is entered in STATUS.
  - If an error occurs during job processing, ERROR is set. The content of the receive buffer is invalid in such a case. An error code is entered in the STATUS parameter for a more detailed description of the error which has occurred.

Number of command calls

If you use the instruction "ASI_CTRL" to send commands, you may not simultaneously send other commands to the same AS-i master with "RDREC (Page 3578)" or "WRREC (Page 3596)". This also applies to multiple calls of the instruction for the same AS-i master.

The instruction "ASI_CTRL" cannot be run with interruptions. Calls therefore must not be programmed in program priority classes which interrupt each other (for example with calls in OB 1 and in OB 35).

Parameters

The following table shows the parameters of the "ASI_CTRL" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, constant	REQ = TRUE starts a new job unless a job is already in progress. No edge evaluation takes place.
LADDR	Input	HW_IO	I, Q, M, D, L, constant	Hardware identifier of the AS-i master. You can obtain the address from the module properties.
SD	Input	VARIANT	I, Q, M, D, L	Send buffer The parameter refers to a memory area in which the command is to be specified (see "ASi commands (Page 3725)").
RD	Input	VARIANT	I, Q, M, D, L	Receive buffer This buffer is only relevant for commands which deliver answer data. The parameter refers to a memory area in which a command response is stored (see "ASi commands (Page 3725)").
DONE	Output	BOOL	I, Q, M, D, L	DONE = TRUE: Job completed without errors.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = TRUE: Job in process.
ERROR	Output	BOOL	I, Q, M, D, L	ERROR = TRUE: Job aborted with error.
STATUS	Output	DWORD	I, Q, M, D, L	Job status / error code See the "STATUS parameter" description.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

LADDR, SD and RD parameters

The parameters LADDR, SD and RD must not be changed in any block cycle when a job is in progress: they must remain constant.

Parameter STATUS

The following table shows the possible STATUS displays dependent upon DONE and ERROR.

DONE	BUSY	ERROR	STATUS	Meaning
0	0	0	00700000_H	First call with REQ = FALSE; no active job.
0	1	0	00700100_H	First call with REQ = TRUE; job has started.
0	1	0	00700200_H	Subsequent call (REQ irrelevant); job is still in progress.
1	0	0	00000000_H	Job completed without errors. No response data.
1	0	0	$0000xx00_H$	Job completed without errors. Number of xx bytes of response data.
0	0	1	$C0818400_H$	Data type of formal operand RD invalid.

DONE	BUSY	ERROR	STATUS	Meaning
0	0	1	C0818500 H	Error in communication with AS-i master (incorrect address configured at the LADDR parameter).
0	0	1	C0838100 H	Incorrect AS-i slave address.
0	0	1	C0838200 H	AS-i slave is not enabled (not in LAS).
0	0	1	C0838300 H	Error at the AS interface (the SD parameter may have been set too small).
0	0	1	C0838400 H	The command is not valid with the current AS-i master status.
0	0	1	C0838500 H	There is an AS-i slave with the address "0".
0	0	1	C0838600 H	The AS-i slave has invalid configuration data (I/O or ID codes).
0	0	1	C083A100 H	The AS-i slave addressed has not been found at the AS interface.
0	0	1	C083A200 H	There is an AS-i slave with the address "0".
0	0	1	C083A300 H	There is already an AS-i slave with the new address at the AS interface.
0	0	1	C083A400 H	The AS-i slave address cannot be deleted.
0	0	1	C083A500 H	The AS-i slave address cannot be set.
0	0	1	C083A600 H	The AS-i slave address cannot be permanently saved.
0	0	1	C083A700 H	Error during reading of the extended ID1 code.
0	0	1	C083A800 H	The target address is not plausible (e.g. a B slave address has been used for a standard slave).
0	0	1	C083B100 H	A length error has occurred during string transfer.
0	0	1	C083B200 H	A protocol error has occurred during string transfer.
0	0	1	C083F800 H	Unknown job number or job parameter.
0	0	1	C083F900 H	The AS-i master has detected an EEPROM error.

See also

RD_REC: Read data record from I/O (Page 3694)

GET_ERR_ID: Get error ID locally (Page 2840)

WR_REC: Write data record to I/O (Page 3699)

Difference between synchronous and asynchronous instructions (Page 2571)

ASi commands (S7-1500)

Description

The command interface allows the controller and AS-i master to exchange parameter assignment and information data.

These commands

- provide the complete functionality of the M4 master profile of the AS-i master specifications.
- enable the AS-i master to be completely configured from the controller.

Note

AS-i commands supported

Please see the manual of the relevant AS-i master for the AS-i commands supported and a detailed description.

General structure of the send buffer

The general structure of the send buffer for commands and job data is set out in the table below. The area for the command number must always be filled. The number of bytes for the job data can be found in the description of the command (see AS-i master documentation). In this case, "q" refers to the start address of the send buffer.

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Meaning							
q + 0	Command number							
q + 1	Job data							
q + 2	Job data							
q + ...	Job data							

General structure of the receive buffer

The general structure of the receive buffer for the command response data is set out in the table below. The number of bytes for the response data depends on the command. Some commands do not return response data, and therefore only require the definition of a virtual receive buffer which is not filled with data. In this case, "n" refers to the start address of the receive buffer.

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Meaning							
n + 0	Response data							
n + 1	Response data							
n + 2	Response data							
n + ...	Response data							

NOTICE
Memory areas can be overwritten If the receive buffer of the "ASI_CTRL" instruction is too short, neighboring memory areas may be overwritten. The length specified in the ANY pointer of the RD parameter when the instruction "ASI_CTRL" is called is irrelevant. The required length for the receive buffer can be found in the description of the command. The following applies for command numbers 39_H, 41_H, 42_H, 43_H and 44_H: The receive buffer must be 221 bytes long (bytes 0 to 220), even if the command returns less data. Depending on the command, the highest bytes in the receive buffer may be overwritten with zero values by the AS-i master.

AS-i commands

A selection of possible AS-i commands is set out in the table below.

Name	Parameter	Return	Coding
Set_permanent_parameter (Set_Permanent_Parameter)	Slave address, parameter		00_H
Get_permanent_parameter (Get_Permanent_Parameter)	Slave address	Parameter	01_H
Write_parameter (Write_Parameter)	Slave address, parameter	Parameter echo	02_H
Read_parameter (Read_Parameter)	Slave address	Parameter value	03_H
Store_actual_parameters (Store_Actual_Parameters)			04 H
Set_configuration_data	Slave address, configuration		25 H
Get_configuration_data	Slave address	Set configuration data	26 H
Store_actual_configuration (Store_Actual_Configuration)			07_H
Get_actual_configuration	Slave address	Actual configuration data	28_H
Configure_LPS	LPS		29_H
Set_offline_mode	Mode		$0A_H$
Select_auto-program	Mode		$0B_H$
Set_mode	Mode		$0C_H$
Change_AS–iSlave_address (Change_AS–iSlave_Address)	Address1, Address2		$0D_H$
Get_AS-iSlave_status	Slave address	Error record of the AS-i slave	$0F_H$
Read_lists_and_flags		LDS, LAS, LPS, flags	30_H

Name	Parameter	Return	Coding
Get_overall_configuration		Actual configuration data, current parameters, LAS, flags	39 $_H$
Set_overall_configuration	Overall configuration		3A $_H$
Write_parameter_list	Parameter list		3C $_H$
Read_parameter_echo_list		Parameter echo list	33 $_H$
Write_CTT2_request	Slave address CTT2 string	CTT2 string	44 $_H$
Read_version_identifier		Version string	14 $_H$
Read_AS-i_slave	Slave address	ID code	17 $_H$
Read_AS-i_slave_extended_ID1	Slave address	Extended ID1 code	37 $_H$
Write_AS-iSlave_extended_ID1	Extended ID1 code		3F $_H$
Read_AS-iSlave_extended_ID2	Slave address	Extended ID2 code	38 $_H$
Read_AS-iSlave_IO	Slave address	I/O configuration	18 $_H$
Read_I/O_error_list		LPF	3E $_H$
Write_AS–i-slave_parameter_string	Slave address, parameter string		40 $_H$
Read_AS–iSlave_parameter_string	Slave address	Parameter string	41 $_H$
Read_AS-iSlave_ID_string	Slave address	ID string	42 $_H$
Read_AS-iSlave_diagnostic_string	Slave address	Diagnostic string	43 $_H$
Read_AS-i_line_error_counter			4A $_H$
Read_and_clear_AS-i_line_error_counter			4B $_H$
Read_AS-iSlave_error_counter	Slave address		4C $_H$
Read_and_clear_AS-iSlave_error_counter	Slave address		4D $_H$
Additional command for DP/ AS-i F-Link:			
AS–i_status/diag_of_F_slaves		Status/diagnostics of all AS-i safe slaves	51 $_H$

Note

Re-initializing the AS-i master command interface

Another command which is not mentioned in the table is command 77 $_H$. This call serves to re-initialize the command interface of the AS-i master. Any command which the AS-i master specified is currently processing will be terminated.

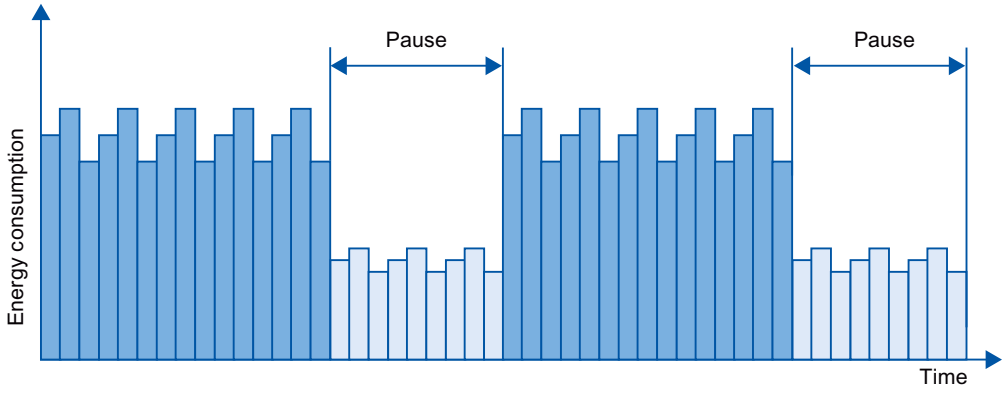
As of version V2.1.20 of DP/AS-i LINK Advanced, command 0E $_H$ is also available. This call is used to release or block the ground fault monitoring function for a line.

11.4.4.5 PROFIenergy (S7-1200, S7-1500)

Description of PROFIenergy (S7-1200, S7-1500)

PROFIenergy

PROFIenergy is a manufacturer- and device-neutral profile for energy management with PROFINET. To reduce electricity consumption during breaks in production and unplanned interruptions, it is possible to shut down equipment in a coordinated and centralized manner using PROFIenergy.



The PROFINET devices/power modules are switched off via special commands in the user program of the PROFINET IO controller. No additional hardware is required. The PROFINET devices interpret the PROFIenergy commands directly.

PROFIenergy controller (PE controller)

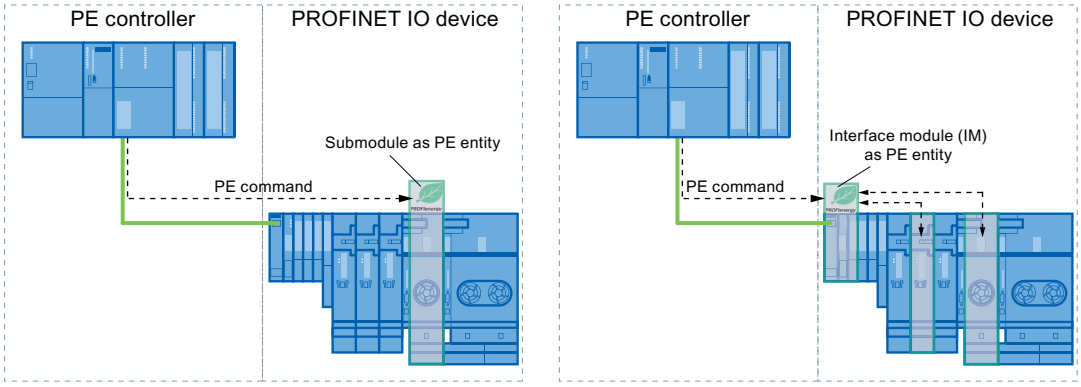
The PE controller is a PLC that activates or deactivates the idle state of lower-level devices. Specific production components or complete production lines are deactivated and reactivated via the user program. Commands (e.g. "Start_Pause" and "End_Pause") are sent to the lower-level device using corresponding instructions (function blocks). The commands are sent via the PROFINET communication protocol.

PROFIenergy entity (PE entity)

The PE entity receives the PROFIenergy commands of the PE controller and executes these accordingly (for example, by returning a measured value or activating an energy saving mode). Implementation of the PE entity in a PROFIenergy-capable device is device- and manufacturer-specific.

The PE entity can, for example, be executed:

- Within the proxy of a submodule: The PE commands are valid for the addressed submodule and any existing lower-level submodules.
- Within the proxy of a module: The PE commands are valid for different submodules within the module.



- For networked submodules without a proxy function: The PE commands in this case are only valid for the respective submodule.

PROFIenergy instructions

- Instructions for IO controller
  - The "PE_START_END (Page 3730)" instruction represents the simplest way to activate or deactivate the idle state of the PROFINET devices (PROFIenergy commands "Start_Pause" and "End_Pause"). This is done using a positive and negative signal edge in the instruction.
  - The "PE_CMD (Page 3735)" instruction allows you to transmit all PROFIenergy commands, including "Start_Pause" and "End_Pause". With other commands, you can query the current status of the PROFINET devices or the behavior during breaks, for example.
  - The instruction "PE_DS3_Write_ET200S (Page 3740)" is used to define the switching characteristic of up to 8 slots of ET 200S. The instruction is not a PROFIenergy instruction; however, it supplements the PROFIenergy functions for an ET 200S.
- Instruction for iDevices
  The "PE_I_DEV (Page 3765)" instruction allows you to implement PROFIenergy on iDevices as well. The instruction receives PROFIenergy commands on the iDevice and forwards these to the user program for processing. After processing the command, the user program calls the "PE_I_DEV (Page 3765)" instruction again in order to send the acknowledgement to the IO controller. For these replies, each command offers you a corresponding auxiliary block that supplies the response data to the instruction "PE_I_DEV (Page 3765)".

PROFIenergy commands (PE commands)

The PE controller transmits PE commands to the PE entity. The PE command can be either a control command to switch a PE entity to the energy-saving mode, or a command to read a status or measured value:

- PI commands for control
  PROFIenergy supports two control commands that can be executed using either the "PE_Start_End (Page 3730)" instruction or the "PE_CMD (Page 3735)" instruction:
  - Start_Pause: Starting a suitable energy-saving mode (PE Energy-saving mode)
  - End_Pause: Exiting the energy-saving mode (switch to PE_ready_to_operate mode)
- PI commands for reading a status or measured value
  Certain condition information can be read by the control system using the following status commands with the instruction "PE_CMD (Page 3735)":
  - PE_Identify: Reading the list of PE commands supported by the PE entity.
  - PEM_Status: Reading the currently active mode of a PE entity (e.g. PE_ready_to_operate).
  - Query_Modes: Output of an overview of all supported energy-saving modes, including the time and energy information
  - Query_Measurement: Output of the measured values of a PE entity

Example applications

Examples for use of PROFIenergy instructions can be found in the Industry Online Support in the entry "PROFIenergy - Saving Energy with SIMATIC S7 (http://support.automation.siemens.com/WW/view/en/41986454)".

See also

Service and Support (http://support.automation.siemens.com/)

IO controller (S7-1200, S7-1500)

PE_START_END: Start and exit energy-saving mode (S7-1200, S7-1500)

Description

The instruction "PE_START_END" is used to start and exit the energy-saving mode of a specified PE entity (e.g. ET 200S).

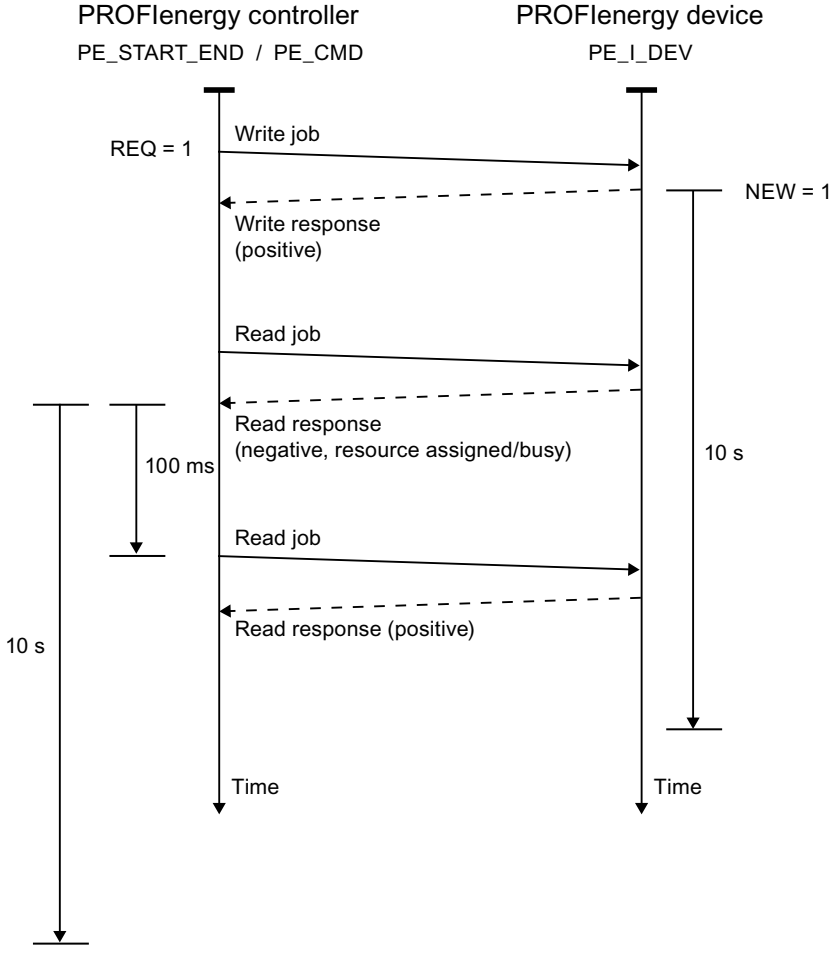
The instruction "PE_START_END" is used in the PE controller, preferably when only field devices from which energy data does not need to be read out are connected to the corresponding PE devices. Alternatively, the instruction "PE_CMD (Page 3735)" can be used to read energy data.

The energy-saving modes are configured in the user program of the PE controller. On completion of the "PE_START_END" instruction, the PE entity reports its current energy-saving mode and outputs this data at parameter PE_MODE_ID.

Write and read jobs of the "PE_START_END" instruction.

The instruction "PE_START_END" sends a PROFIenergy command internally as a write job to the PE entity using "WRREC (Page 3596)". "PE_START_END" then waits for acknowledgment from the PE entity. The acknowledgment data record is read with the instruction "RDREC (Page 3578)" every 100 milliseconds. Until acknowledgment is received from the PE entity, the function repeats the read job for 10 seconds at intervals of 100 milliseconds. The response data of the PE entity is also read with the instruction "RDREC (Page 3578)".

Below is a flowchart of the write and read jobs:



Parameter

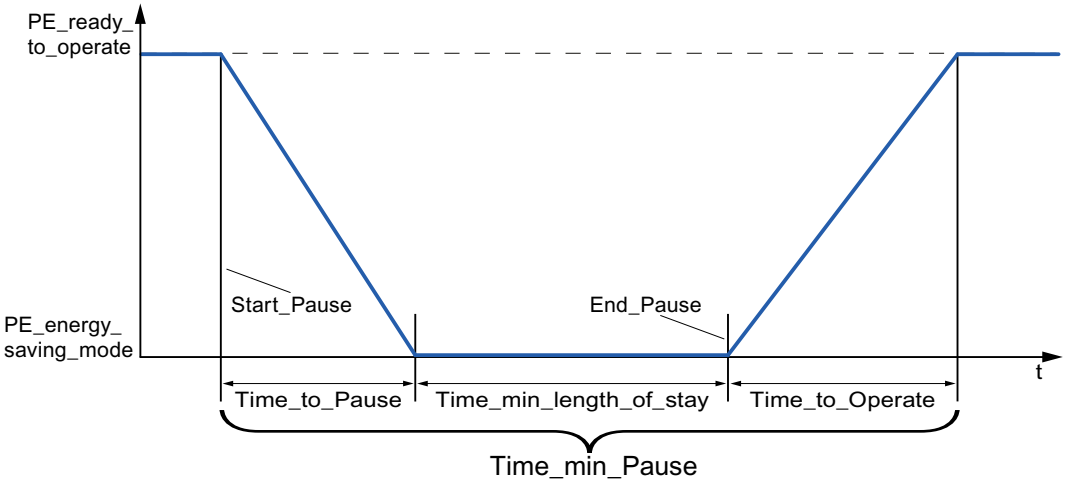
The following table shows the parameters of the "PE_START_END" instruction:

Parameter	Declaration	Data type	Memory area	Description
START	Input	BOOL	I, Q, M, D, L or constant	Transmitting the PE command "Start_Pause" to the PE entity with the address set at parameter ID.
END	Input	BOOL	I, Q, M, D, L or constant	Transmitting the PE command "End_Pause" to the PE entity with the address set at parameter ID.
ID	Input	HW_SUBMODULE	I, Q, M, D, L or constant	Address of the PE entity Use the hardware ID of the head module for a PROFINET IO device. Refer to the system constants of the assigned IO controller for the hardware ID. The name of the head module is composed of the name of the IO device and the suffix [Head] (example: "IO_Device_1[Head]"). If the PE entity is an I-device, use the hardware identifier of a transfer area instead.
PAUSE_TIME	Input	TIME	I, Q, M, D, L, P or constant	Planned pause duration. • Range: T#1MS to T#24D20H31M23S647MS • Start value: T#0MS
VALID	Output	BOOL	I, Q, M, D, L	PE command successfully sent.
BUSY	Output	BOOL	I, Q, M, D, L	PE command still being processed.
ERROR	Output	BOOL	I, Q, M, D, L	Error occurred during processing. The error message is output at the STATUS parameter.
STATUS	Output	DWORD	I, Q, M, D, L, P	Block status / error number (see "STATUS parameter")
PE_MODE_ID	Output	BYTE	I, Q, M, D, L, P	Identification number of energy-saving mode (energy-saving level for the duration of the pause).

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter PAUSE_TIME

The parameter PAUSE_TIME is used to set the default duration of the energy-saving period at the PE entity. The PE entity checks whether the pause period is of sufficient length and whether it can be implemented. The minimum pause duration (Time_min_Pause) must be greater than the sum of the times the device needs to switch to energy-saving mode (Time_to_Pause) and to switch back to operating mode (Time_to_Operate).



ET 200S checks whether the scheduled pause duration is greater than or equal to the minimum pause duration (PM-E_Pause_Min) saved on the ET 200S. This is fixed at 10 seconds. If a shorter pause is used, the power modules (PM-E) of the ET 200S will remain on.

The module does not switch back on automatically at the end of the pause; it remains in OFF mode until the "END" command is sent. This prevents uncoordinated switch-on, which can cause undesired peak loads.

Parameter STATUS

Error information is output at the STATUS output parameter. If it is interpreted as ARRAY[1...4] of BYTE, the error information has the following structure:

Field element	Name	Meaning
STATUS[1]	Function_Num	Cause of the error • B#16#**00**: No error • B#16#**DE**: Error while reading the data record • B#16#**DF**: Error while writing the data record • B#16#**C0**: Error message caused by the instruction or the communication instructions "RDREC (Page 3578)" and "WRREC (Page 3596)" used internally.
STATUS[2]	Error_Decode	Location of the error ID • **80**: DPV1 error as defined in IEC 61158-6 or application-specific • **FE**: DP/PNIO profiles - PROFIenergy-specific error

Field element	Name	Meaning
STATUS[3]	Error_Code_1	Error ID • When Error_Decode = **80**: – **80**: Simultaneous positive edge at the input parameters START and END. – **81**: Length conflict at the CMD_PARAM and CMD_PARAM_LEN parameters. – **82-8F**: Other error messages (reserved) • When Error_Decode = **FE**: – **01**: "Service Request ID" invalid – **02**: Incorrect "Request_Reference" – **03**: "Modifier" invalid – **04**: "Data Structure Identifier RQ" invalid – **05**: "Data Structure Identifier RS" invalid – **06**: "PE energy-saving modes" are not supported – **07**: "Response" is too long (maximum transmissible length exceeded) – **08**: "Count" invalid – **50**: No suitable "energy mode" is available. – **51**: Time value specified is not supported. – **52**: "PE_Mode_ID" invalid – **53**: No switch to "PE energy-saving mode" because the device is in operation – **54**: Function not available at this time. Wrong device configured or incorrect setup. – **55** to **FF**: Reserved
STATUS[4]	Error_Code_2	Manufacturer-specific extension of the error ID

Note

Error messages of the instructions RDREC and WRREC

The instruction "PE_START_END" uses the instructions "WRREC (Page 3596)" and "RDREC (Page 3578)" for communication. Error messages for these instructions are output in the field elements STATUS[1] to STATUS[4].

Please see the description of the corresponding STATUS (Page 3638) parameter for an explanation of the error codes of the "WRREC (Page 3596)" and "RDREC (Page 3578)" instructions.

See also

Description of PROFIenergy (Page 3728)

Difference between synchronous and asynchronous instructions (Page 2571)

PE_CMD: Start and exit energy-saving mode / Read out status information (S7-1200, S7-1500)

Description

The instruction "PE_CMD" is used in the PE controller to start or terminate a pause in the energy-saving mode in the PE entity. Additional information and energy measurements are also read out from a PE entity using "PE_CMD".

The instruction is best used with PE controllers with assigned PE devices to which field devices are connected from which energy measurements are to be read out. If this is not the case, the instruction "PE_START_END (Page 3730)" can also be used to start and end the pauses.

Transfer of the PROFIenergy commands (PE commands)

The instruction "PE_CMD" transfers a PROFIenergy command to a PE entity.

The instruction can also be used if additional commands are to be added to the PROFIenergy profile in future. The commands which can be used with the current PROFIenergy profile are listed in the description of the CMD and CMD_MODIFIER parameters (see the "CMD and CMD_MODIFIER parameters" table).

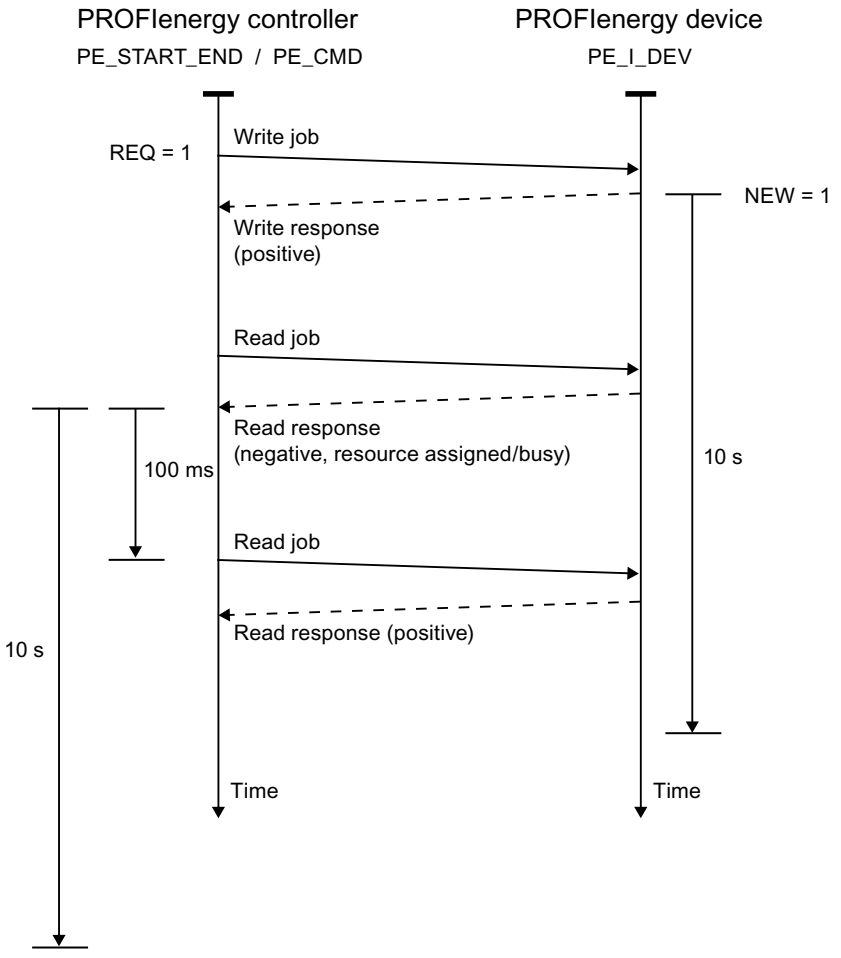
- The individual PE commands which are transferred to the PE entity using the instruction are assigned defined "Service_Request_IDs". The Service_Request_IDs 01 to 05 and 16 are assigned at the CMD parameter.
- The two PE commands 04 (Query_Modes) and 16 (Query_Measurement) are defined in more detail with the CMD_MODIFIER parameter.
- Additional values are transferred at the CMD_PARA parameter for individual PE commands (see the description of the individual PE commands). The parameter CMD_PARA_LEN defines the length of the data at the parameter CMD_PARA.

The commands are transferred without plausibility test. The response data of the PE entity is saved to the RESPONSE_DATA data area that is addressed by VARIANT pointer (for information about the response frame contents, refer to the description of the respective PE command).

Write and read jobs of the "PE_CMD" instruction.

The instruction "PE_CMD" uses "WRREC (Page 3596)" internally to transmit a PROFIenergy command as a write job to the PE entity. "PE_CMD" then waits for the acknowledgment from the PE entity. The acknowledgment data record is read with the instruction "RDREC (Page 3578)" every 100 milliseconds. Until acknowledgment is received from the PE entity, the function repeats the read job for 10 seconds at intervals of 100 milliseconds. The response data of the PE entity is also read with the instruction "RDREC (Page 3578)".

Below is a flowchart of the write and read jobs:



Parameter

The following table shows the parameters of the "PE_CMD" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Starts transferring the PE command upon a positive edge.
ID	Input	HW_ SUBMODULE	I, Q, M, D, L or constant	Address of the PE entity Use the hardware ID of the head module for a PROFINET IO device. Refer to the system constants of the assigned IO controller for the hardware ID. The name of the head module is composed of the name of the IO device and the suffix [Head] (example: "IO_Device_1[Head]"). If the PE entity is an I-device, use the hardware identifier of a transfer area instead.

Parameter	Declaration	Data type	Memory area	Description
CMD	Input	BYTE	I, Q, M, D, L, P or constant	Service-Request-ID of the PROFIenergy command in accordance with the PROFIenergy profile (see "CMD and CMD_MODIFIER parameters"). Further service request IDs are possible following extensions of the PROFIenergy profile.
CMD_MODIFIER	Input	BYTE	I, Q, M, D, L, P or constant	PROFIenergy sub-command (only when CMD=3 or CMD=16, see "CMD and CMD_MODIFIER parameters") Further sub-commands are possible following extensions of the PROFIenergy profile.
CMD_PARA	Input	VARIANT	I, Q, M, D, L	Parameters for the PE commands: • Get mode: PE_mode_ID • Get measurement values: List of Measurement_Ids The complete Service Data Request is entered.
CMD_PARA_LEN	Input	INT	I, Q, M, D, L, P or constant	The actual length of the command parameters (<= length defined in CMD_PARA; is verified by the instruction).
RESPONSE_DATA	InOut	VARIANT	I, Q, M, D, L	PROFIenergy information May be complete response frame including block header, depending on the command. Note: If the buffer is too small, only the number of bytes which is specified in the VARIANT pointer will be entered.
VALID	Output	BOOL	I, Q, M, D, L	Command successfully sent.
BUSY	Output	BOOL	I, Q, M, D, L	Command still being processed.
ERROR	Output	BOOL	I, Q, M, D, L	Error occurred during processing.
STATUS	Output	DWORD	I, Q, M, D, L	Block status / error number (see "STATUS parameter"):

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameters CMD and CMD_MODIFIER

CMD	CMD_ MODIFIER	PROFIenergy command	Description
01	0	Start_Pause (Page 3755)	Start energy-saving mode or switch to a different energy-saving mode.
02	0	End_Pause (Page 3755)	End energy-saving mode.

CMD	CMD_ MODIFIER	PROFIenergy command	Description
03	1	Query_Modes - List of energy-saving modes (Page 3756)	Output the energy-saving modes supported.
	2	Query_Modes - Get mode (Page 3757)	Output attributes of the energy-saving mode currently enabled.
04	0	PEM_Status (Page 3759)	Query status of energy-saving mode.
05	0	PE_Identify (Page 3761)	Read out the number and description of PE commands supported.
16	1	Query_Measurement - Get_Measurement_List (Page 3761)	List of measured values supported by the PE entity.
	2	Query_Measurement - Get_Measurement_Values (Page 3764)	Output the measured values of the PE entity.

Parameter STATUS

Error information is output at the STATUS output parameter. If it is interpreted as ARRAY[1...4] of BYTE, the error information has the following structure:

Field element	Name	Meaning
STATUS[1]	Function_Num	Cause of the error • B#16#**00**: No error • B#16#**DE**: Error while reading the data record • B#16#**DF**: Error while writing the data record • B#16#**C0**: Error message from the internal communication instructions "RDREC (Page 3578)" and "WRREC (Page 3596)".
STATUS[2]	Error_Decode	Location of the error ID • **80**: DPV1 error as defined in IEC 61158-6 or application-specific • **FE**: DP/PNIO profiles - PROFIenergy-specific error

Field element	Name	Meaning
STATUS[3]	Error_Code_1	Error ID • When Error_Decode = **80**: – **81**: Length conflict at the CMD_PARA and CMD_PARA_LEN parameters, or the maximum data record length (4095 bytes) was exceeded. – **82-8F**: Other error messages (reserved) • When Error_Decode = **FE**: – **01**: "Service Request ID" invalid – **02**: Incorrect "Request_Reference" – **03**: "Modifier" invalid – **04**: "Data Structure Identifier RQ" invalid – **05**: "Data Structure Identifier RS" invalid – **06**: "PE energy-saving modes" are not supported – **07**: "Response" is too long (maximum transmissible length exceeded) – **08**: "Count" invalid – **50**: A suitable energy-saving mode (energy mode) is not available. – **51**: Time value specified is not supported. – **52**: "PE_Mode_ID" invalid
STATUS[4]	Error_Code_2	Manufacturer-specific extension of the error ID

Note

Error messages of the instructions RDREC and WRREC

The instruction "PE_CMD" uses the instructions "WRREC (Page 3596)" and "RDREC (Page 3578)" for communication. Error messages for these instructions are output in the field elements STATUS[1] to STATUS[4].

Please see the description of the corresponding STATUS (Page 3638) parameter for an explanation of the error codes of the "WRREC (Page 3596)" and "RDREC (Page 3578)" instructions.

See also

Description of PROFIenergy (Page 3728)

Difference between synchronous and asynchronous instructions (Page 2571)

PE_DS3_Write_ET200S: Set power module switching behavior (S7-1200, S7-1500)

Description

The instruction "PE_DS3_Write_ET200S" sends basic settings for power module switching behavior to the ET 200S. The switching behavior of up to 8 slots in the ET 200S (e.g. for power modules) is defined with the instruction "PE_DS3_Write_ET200S".

Note

This instruction is not part of the PROIenergy profile but instead supplements SIMATIC-specific functions.

Parameter

The following table shows the parameters of the "PE_DS3_Write_ET200S" instruction:

Parameter	Declaration	Data type	Memory area	Description
ENABLE	Input	BOOL	I, Q, M, D, L or constant	A positive edge triggers the transfer of the data record. Transfer the data record again following voltage OFF/ON.
ID	Input	HW_SUBMODULE	I, Q, M, D, L or constant	Address of the ET 200S Accept the address from the hardware configuration.
SLOT_NO_X	Input	INT	I, Q, M, D, L, P or constant	Slot number of switchable power module X.
FUNC_X	Input	INT	I, Q, M, D, L, P or constant	Function of the module in this slot. Use the parameter FUNC_X to define the switching behavior of the PM-E (power module of the ET 200S): • FALSE: – With "PAUSE_START": -No effect on PM-E -PM-E remains on – With "PAUSE_STOP": -Switches PM_E back on • TRUE: – With "PAUSE_START": -Switches PM_E off – With "PAUSE_STOP": -Switches PM-E back on
BUSY	Output	BOOL	I, Q, M, D, L	Transfer not yet complete.
DONE	Output	BOOL	I, Q, M, D, L	Transfer completed without errors.
ERROR	Output	BOOL	I, Q, M, D, L	Transfer completed with error.
STATUS	Output	DWORD	I, Q, M, D, L, P	Error number (see STATUS parameter of the instruction "PE_Start_End (Page 3730)")

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

See also

Description of PROFIenergy (Page 3728)

Difference between synchronous and asynchronous instructions (Page 2571)

PE_WOL: Starting and stopping energy-saving mode via WakeOnLan (S7-1200, S7-1500)

Description PE_WOL (S7-1200, S7-1500)

Description

The "PE_WOL" instruction sends the PROFIenergy commands "Start_Pause" and "End_Pause" to multiple PROFIenergy-enabled devices in the PROFINET I/O systems.

Multiple PE devices can be coordinated via the instruction provided that the PE devices support the "Wake on LAN" function over a UDP connection.

The "PE_WOL" instruction can only be executed on a CPU with and integrated Ethernet interface. This CPU should be able to load blocks with a size of approx. 400 KB. If this CPU is connected via an Ethernet CP, it is not possible to use the block with PROFINET I/O systems.

The instruction "PE_WOL" is executed asynchronously.

The "PE_WOL" instruction only supports devices with a device number lower than 256. Devices with a higher device number are not taken into account.

Definition: Wake on LAN

Using Wake on LAN, deactivated devices can be reactivated through the receipt of a special Ethernet packet.

For this procedure to work, the data processing equipment must have a network controller that is capable of receiving such a packet.

This packet (Magic Packet™) has a special format. It contains the MAC address of the network adapter 15 times.

Selection of devices

The devices are selected using the user data block at the parameter PENERGY (Type: "PE_PLUS"). The user DB represents the database for processing multiple devices.

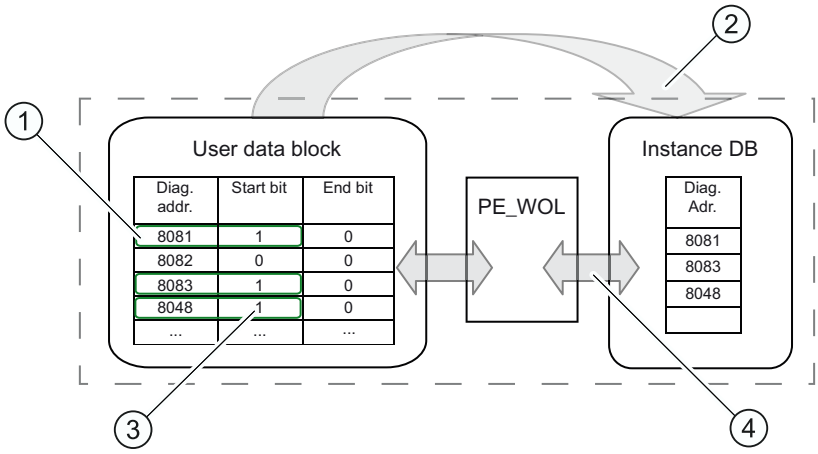
Prior to initialization of "PE_WOL", you must store at least the following information in the user DB:

- ID of the PROFINET I/O system
- Data of the connection which is used for "Wake On LAN".

- Port number which is used for "Wake On LAN".
- For each device
  - Pause time (PauseTime)
  - Switching of the device to the PE_SLEEP_MODE (EnableSleep)

You initialize the "PE_WOL" instruction with the COM_RST parameter. The jobs stored in the user DB are processed one after the other after initialization.

The graphic below illustrates how the PE command "Start_Pause" is transmitted to several devices:



(1) Step 1: The bit "CmdStartPause" of the devices to be shut down is set to "1" by the user.

(2) Step 2: The diagnostic addresses of the devices to be shut down (CmdStartPause = "1") are linked to the queue.

(3) Step 3: The bit "CmdStartPause" is reset automatically once the jobs have been linked.

(4) Step 4: The instruction "PE_WOL" processes the jobs as soon as they are linked.

A PROFIenergy command "CmdStartPause" or "CmdEndPause" can be sent for all recognized devices in the PROFINET IO system via the START and END parameters.

The status of the job processing as well as possible errors during processing are output by the STATUS parameter.

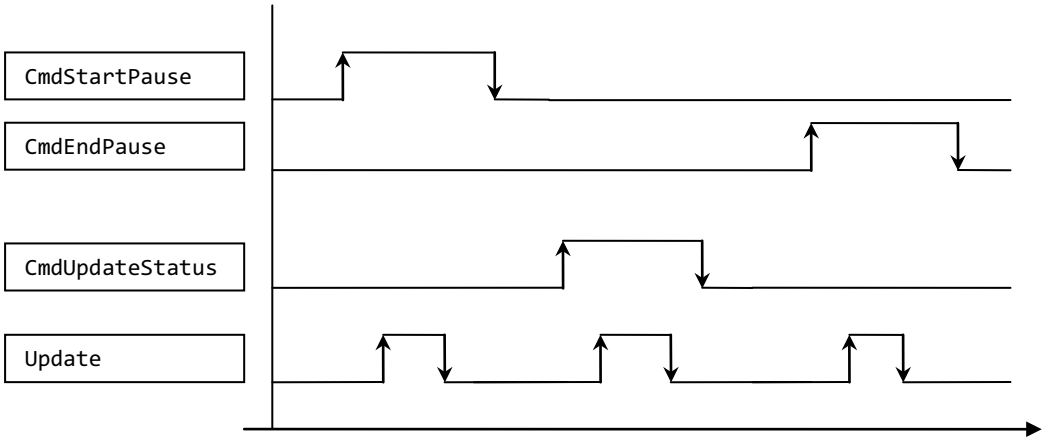
Operation of the instruction using the user DB

The instruction "PE_WOL" can only be operated via the user DB. A basic procedure applies in this case:

1. Selection of the command to be executed for a device:
   - START_PAUSE ("CmdStartPause" in user DB)
   - ENDE_PAUSE ("CmdEndPause" in user DB)
   - UPDATE_STATUS ("CmdUpdateStatus" in user DB)
2. Setting the bit for updating ("Update" in the header of the user DB)
   At least one CPU cycle should pass with an "Update" = False between two updates; otherwise, edge detection cannot be guaranteed.

Prioritizing the PE commands

The graphic below shows the chronological sequence of three possible commands.



These are processed one after the other, regardless of whether the previous command call was successful or completed with an error.

Only one command is executed if you set two commands, such as "CmdEndPause" and "CmdUpdateStatus" simultaneously. There is a prioritization within the block:

- The command "CmdStartPause" has the highest priority and is therefore always executed if selected.
- The command "CmdEndPause" has the second highest priority.
- The command "CmdUpdateStatus" has the lowest priority.

If all three commands are set at the same time, the commands that are not executed remain preselected. In this case, the next rising edge executes the next command.

Parameters

The following table shows the parameters of the "PE_WOL" instruction:

Parameter	Declaration	Data type	Memory area	Description
COM_RST (Page 3744)	Input	BOOL	I, Q, M, D, L	Resets the block and performs a re-initialization. As long as True is set here, the initialization is started but not yet fully completed. Only the falling edge continues the initialization and switches to the normal operating mode after initialization.
START (Page 3745)	Input	BOOL	I, Q, M, D, L	A rising edge performs a "CmdStartPause" PROFIenergy command for all detected devices that support this function.
END (Page 3746)	Input	BOOL	I, Q, M, D, L	A rising edge performs a "CmdEndPause" PROFIenergy command for all detected devices that support this function.

Parameter	Declaration	Data type	Memory area	Description
PENERGY (Page 3746)	InOut	PE_PLUS	D	Pointer to the user DB that contains the database for processing multiple devices.
STATUS (Page 3751)	Output	DWORD	I, Q, M, D, L	Status/error number for the current status of the instruction (see "STATUS parameter").

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

See also

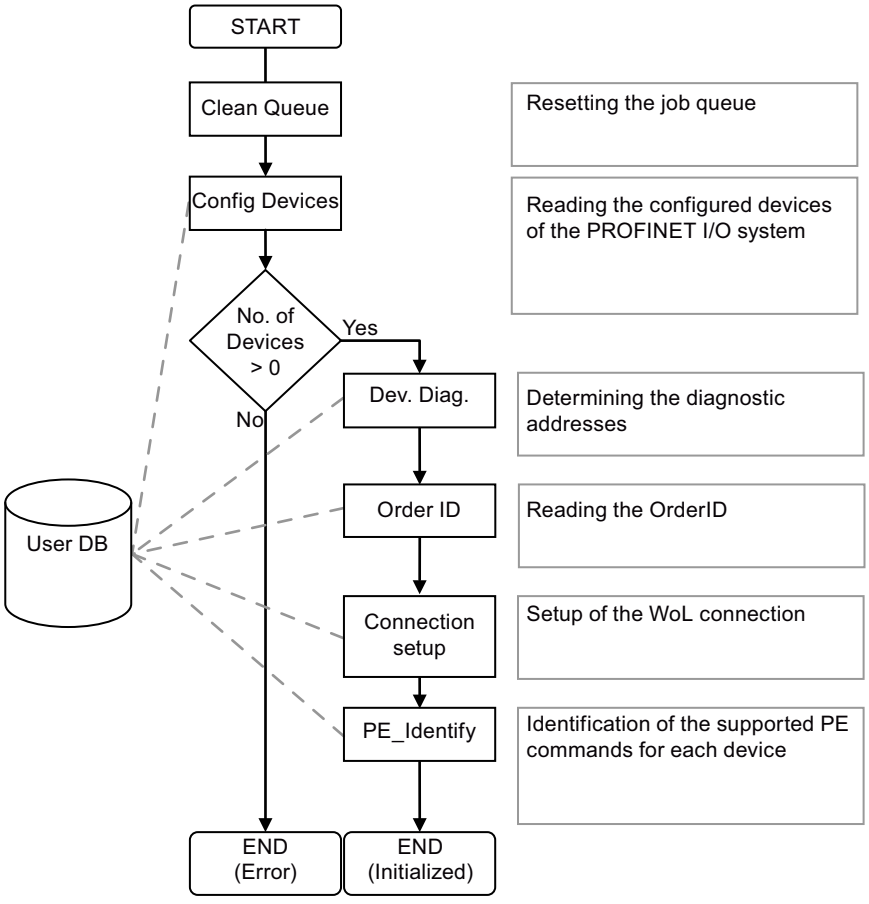
Difference between synchronous and asynchronous instructions (Page 2571)

Parameter COM_RST (S7-1200, S7-1500)

Sequence of the initialization routine

You start the initialization of the "PE_WOL" instruction with the COM_RST parameter.

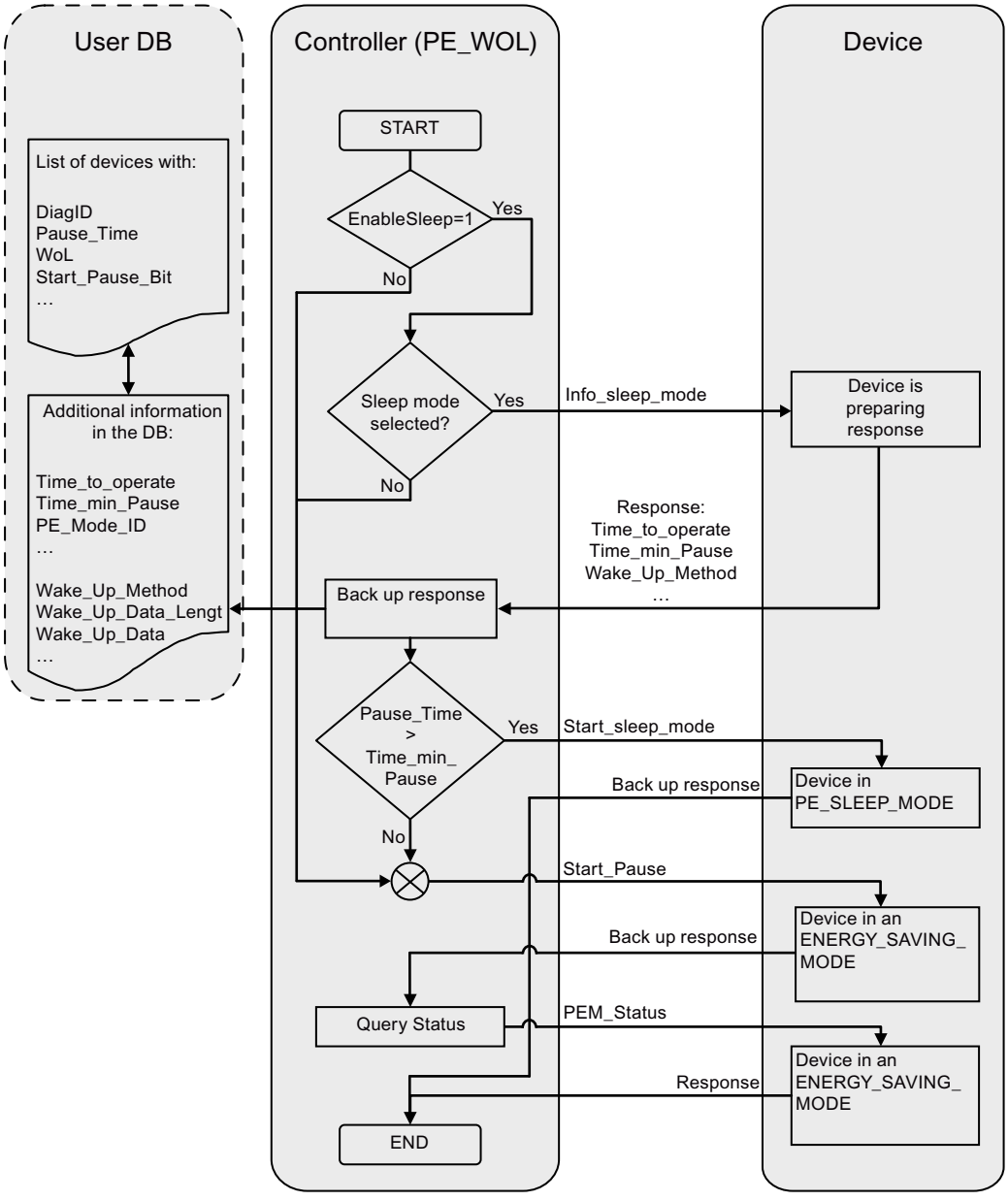
The following flow diagram shows the initialization routine.



Parameter START (S7-1200, S7-1500)

Sequence of the CmdStartPause command

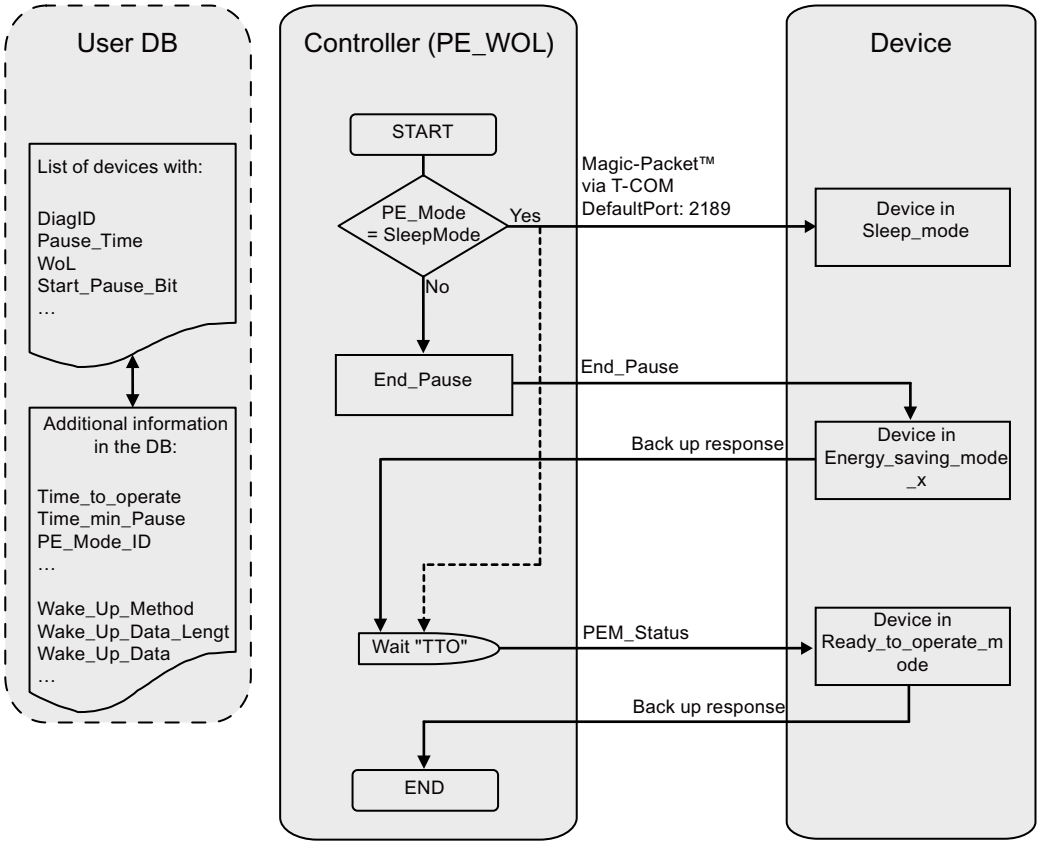
The flow diagram shows the internally used functions and interaction with a device when executing the CmdStartPause command.



Parameter END (S7-1200, S7-1500)

Sequence of the CmdEndPause command

The flow diagram shows the internally used functions and interaction with a device when executing the CmdEndPause command.



PENERGY parameter (S7-1200, S7-1500)

Data block at the PENERGY parameter

The user DB for the "PE_WOL" PROFIenergy instruction represents a data base for processing multiple devices.

The data block is usually divided into two parts. These are:

- Header 110 bytes
- Device section for a maximum of 256 devices each with 100 bytes (Device). These include:
  - Device-specific data (Device)
  - PROFIenergy-specific data (PE)
  - Data for the job processing (Task)
  - User data (UserData)

The data block operates with optimized access.

Connection parameter "Connection"

The "PE_WOL" instruction reserves a connection resource in the area of "Open User Communication". This is used as a UDP connection. The following parameters must be defined in the data block for this:

- Connection ID ("Connection.id" parameter)
  The connection ID is an integer within the range of 1 to 4095. It is used to identify the communication resources allocated by the firmware, for example the send and receive buffers.
  The connection ID must be unique throughout the CPU.
- Port number that is used for the "Wake on LAN" function ("Header.PortNo" parameter)
  Number of the UDP port via which a "Wake On LAN" packet is sent. These port numbers are part of the communication resources, which are identified and provided by the firmware through the connection ID. The default setting for port 2189 used here is currently not assigned by IANA. The port number is transferred to the connection configuration and applied to parameter "Connection.local_tsap_id[1]". The port number and the length of the remote port are defined using parameters "Connection.rem_tsap_id[1]" and "Connection.rem_tsap_id_LEN" and must be assigned manually.

Structure of the data block

The data block has the following structure:

Name	Data type	Offset	Comment
Header	PE_HEADER	-	Header information
Update (1)	BOOL	-	Signal for indicating a change in the data area. • True= indicates a change by the user. • False = indicates the application of changes.
Initialized	BOOL	-	Signal indicating the completed initialization. • True=Initialization finished. • False= indicates that the block is not initialized.
LinkUp	BOOL	-	Indicates successful configuration of the Ethernet interface. • True=interface ready for use. • False=interface not yet configured.
LinkDown	BOOL	-	Indicates an unconfigured interface. • True=interface is not yet configured. • False=interface is currently being configured or is now configured.
PROFINET_ID (1)	INT	-	ID of the PROFINET I/O system
Reserved	ARRAY [1..37] OF BYTE	-	Reserved
LastDeviceID	INT	-	Contains the highest Device-ID in this PROFINET I/O system.
PortNo (1)	WORD	-	Port number that is used for the "Wake on LAN" function (default= 2189).
Connection	TCON_Param	-	Contains the connection configuration of the "Wake on LAN" connection.

Name	Data type	Offset	Comment
BLOCK_LENGTH	UInt	-	Length of the structure (always B#16#40).
ID (1)	CONN_OUC	-	Connection ID
CONNECTION_TYPE (1)	USINT	-	Connection type (UDP = B#16#13)
ACTIVE_EST (1)	BOOL	-	Active connection establishment (always passive for UDP)
LOCAL_DEVICE_ID (1)	USINT	-	Contains the interface ID (CPU dependent)
LOCAL_TSAP_ID_LEN (1)	USINT	-	Contains the length in bytes of the own/local UDP port
REM_SUBNET_ID_LEN (1)	USINT	-	Unused (always B#16#00)
REM_STADDR_LEN (1)	USINT	-	Contains the length of the remote IP address or B#16#00.
REM_TSAP_ID_LEN (1)	USINT	-	Contains the length in bytes of the remote UDP port. The length specification must be entered manually.
NEXT_STADDR_LEN (1)	USINT	-	Contains the length of the default router address (not relevant).
LOCAL_TSAP_ID (1)	ARRAY[1..16] OF BYTE	-	Contains your own local port number. The value is applied from parameter PortNo during initialization.
REM_SUBNET_ID (1)	ARRAY[1..6] OF USINT	-	Unused (always B#16#00)
REM_STADDR (1)	ARRAY[1..6] OF USINT	-	Contains the remote IP address.
REM_TSAP_ID (1)	ARRAY[1..16] OF BYTE	-	Contains the remote UDP port number. The port number must be entered manually.
NEXT_STADDR (1)	ARRAY[1..6] OF BYTE	-	Irrelevant
SPARE (1)	WORD	-	
Device	ARRAY[1..256] OF PE_DEVICE	-	Array of devices
Device	PE_DEV	-	Contains data for each device
DeviceID	HW_DEVICE	-	Hardware identifier of the device. Assigned by the hardware configuration.
PE_EntityID	HW_IO	-	Hardware identifier of the PROFIenergy entity. Assigned by the hardware configuration.
MACAdr	ARRAY[1..6] OF BYTE	-	Contains the MAC address of the device.
IPAdr	ARRAY[1..4] OF BYTE	-	Contains the IP address of the device.
OrderID_MxLen	Byte	-	Contains the maximum length of the OrderID.
OrderId_ActLen	Byte		Contains the actual length of the OrderID.
OrderID_Data	ARRAY[1..20] OF CHAR		Contains the OrderID of the device.
PE	PE_PE	-	PROFIenergy-specific data.

Name	Data type	Offset	Comment
ModeID	BYTE	-	PE_MODE_ID according to PROFIenergy specification.
Result	BYTE	-	PE ErrorCode according to PROFIenergy specification.
PauseTime (1)	TIME	-	Contains the idle time in ms.
TimeToPause	TIME	-	Contains the time it takes for the device to go into Pause mode.
TimeToOperate	TIME	-	Contains the time it takes for the device to go into operating mode.
MinSleepTime	TIME	-	Contains the minimum time of the device in PE_SLEEP_MODE.
SleepToOperate	TIME	-	Contains the time it takes for the device to go from PE_SLEEP_MODE into operating mode.
StatusOperate	BOOL	-	Indicates the operating mode of the device.
StatusPause	BOOL	-	Indicates the Pause mode of the device.
StatusSleep	BOOL	-	Indicates the PE_SLEEP_MODE of the device.
StatusTransitOK	BOOL	-	Indicates the transition from one energy state to another.
StatusInTransit	BOOL	-	Indicates a current state transition.
StatusTransitNOK	BOOL	-	Indicates that the state change was not successful.
StatusError	BOOL	-	Indicates an error with the device.
StatusRetryEx	BOOL	-	Indicates an unsuccessful execution of a command. No more attempts will be made to execute this command.
CmdStartPause (1)	BOOL	-	Places a START_PAUSE command for this device in the queue.
CmdEndPause (1)	BOOL	-	Places a END_PAUSE command for this device in the queue.
CmdUpdateStatus (1)	BOOL	-	Places a PEM_STATUS command for this device in the queue.
EnableSleep (1)	BOOL	-	Allows PE_SLEEP_MODE for this device. • True= device should go to PE_SLEEP_MODE if idle time is sufficient • False=device may not go into PE_SLEEP_MODE.
Services	WORD	-	Shows all supported PROFIenergy services.
UserData (2)	ARRAY[1..24] OF BYTE	-	User-defined data
Task	PE_TASK	-	Job processing

Name	Data type	Offset	Comment
Cmd	BYTE	-	Internal bits for job processing
CmdJ	BYTE	-	Internal bits for job processing
TimeStart	BOOL	-	Starts a delay time.
TimeStarted	BOOL	-	The delay time has just started.
TimeDone	BOOL	-	Shows the expiration of the delay time.
Done	BOOL	-	Indicates that this job has been completed.
DelayedCmd	BOOL	-	Indicates that a delayed command is still pending.
IsV1_0	BOOL	-	Indicates that this device is a Spec. V1.0 device.
IsWakeOnLAN	BOOL	-	Indicates that this device is woken up by "Wake On LAN".
RetryCount	BYTE	-	Repetition counter for PE_COMMANDS
Duration	TIME	-	Contains the value for the delay in ms.
StartTime	TIME	-	Contains the start time for the delay time.
MachineState	INT	-	Contains the internal status of the job.

(1) To be filled in by the user.

(2) Available for use by the user.

Parameter STATUS (S7-1200, S7-1500)

Parameter STATUS

The output value at the STATUS parameter is divided into three areas:

- Bits 31 to 24: MESSAGE
- Bits 23 to 16: LOCATION
- Bits 15 to 0: INFORMATION

You can learn the meaning of each error code for the three areas in the following tables:

Table 11-40 Possible values for MESSAGE

Error code (W#16#...)	Description
00	No error.
50	Instruction initialized.
51	Determining configuration of the PROFINET I/O system.
52	The instruction could not determine any configured devices in the PROFINET I/O system.
53	Determining the logical addresses of configured devices.
54	Reading interface information out of the devices.
55	Determining I&M data (data record 0 only) of the configured devices.
56	Configuring PROFINET interface for sending the "Wake on LAN" MagicPaket™ via UDP.
57	Determining PROFIenergy capabilities of the connected devices.
62	Invalid PROFINET I/O system ID detected. The causative number is displayed in the INFORMATION field.

Error code (W#16#...)	Description
70	The instruction is initiated and processing jobs. The value in the INFORMATION field contains the number of currently active jobs.
80	The instruction has lost the initialization during processing of jobs. This generally occurs when either the instance DB or the user DB has been reloaded.
FF	Unknown error has occurred.

Table 11-41 Possible values for LOCATION

Error code (W#16#...)	Description
00	The instruction is either not initialized or it is idle.
70	The instruction is waiting for jobs.
71	The instruction has appended a job to the job list.
72	The instruction is preparing to send a job.
73	The instruction is sending a job to a device.
74	The instruction is waiting for a response from the device.
75	The instruction is evaluating the response of the device.
76	The instruction is removing the job from the job list.
FF	Unknown error has occurred.

Table 11-42 Possible values for INFORMATION

Error code (W#16#...)	Description
0000	No additional information available and no jobs active.
0001 -00FF	1 – 255 jobs are currently being processed.
7000	Re-initialization with COM_RST started but not yet completed.
8001	Error at 1st parameter
8002	Error at 2nd parameter
8003	Error at 3rd parameter
8004	Error at 4th parameters
8005	Error at 5th parameter This error is reported if there is no interconnection or an invalid interconnection to the user DB. Possible causes: • The user DB is too small. • The user DB is write-protected. • There is no user DB in the RAM. • The user DB is invalid for the employed CPU.

Error code (W#16#...)	Description
8085 to 80CE	Error when configuring the connection. The error messages of the internally used TCON instruction are output. The description of the error messages can be found in the table for the STATUS (Page 4478) parameter.
8100	An attempt has been made to place more than the maximum of 256 possible jobs. This is a temporary error, which is resolved by completing a few jobs. The placed jobs are not accepted and must be placed again.
8200	An attempt was made to send an invalid or an unsupported PROFIenergy command (PE_COMMAND).
8400	Connection ID outside the permitted range. Initialization aborted. Check the ID of the connection configuration for "Wake on LAN". See data block at parameter PENERGY > Header > Connection > ID.
84xx	A communication error has occurred. The number of the device that generated the error is output to "xx".
85xx	An error has been returned from device xx. The number of the device that generated the error is output to "xx".
8600	The requested wake-up method is currently not supported.
FFFF	Unknown error has occurred.

See also

PENERGY parameter (Page 3746)

PROFIenergy commands (S7-1200, S7-1500)

Structure of the message frames (S7-1200, S7-1500)

Structure of the response frame in accordance with the PROFIenergy profile

The table below shows the basic structure of the response frame in accordance with the PROFIenergy profile. The response frame consists of a general section (Header) and a specific section (Service Data Response). The contents of the specific section of the response frame can be found in the description of the relevant PROFIenergy command.

Block definition	Attributes	Value	Data type	Description
BlockHeader	BlockType	801_{hex}	WORD	
	BlockLength		WORD	Number of bytes without consideration of the fields BlockType and BlockLength.
	BlockVersionHigh	1_{hex}	BYTE	
	BlockVersionLow	0_{hex}	BYTE	

Block definition	Attributes	Value	Data type	Description
Response Header	Service_Request_ID	1_{hex} to FF $_{hex}$	BYTE	ID of the PI command executed. The ID of the PE command processed by the PE entity is returned in the response frame: • 01: Start_Pause • 02: End_Pause • 03: Query_Modes • 04: PEM_Status • 05: PE_Identify • 06 to 09: Reserved • 16: Query_Measurement • 11 to CF: Reserved • D0 to FF: Manufacturer-specific
	Request_Reference	1_{hex} to FF $_{hex}$	BYTE	Unique number to identify the query/response pair (returned by the server in the response).
Service Header Response	Status	1_{hex} to FF $_{hex}$	BYTE	Information whether or not the PI command was executed: • 00: Reserved • 01: Completed • 02: Completed with error • 03: Data incomplete • 04 to CF: Reserved • D0 to FF: Depends on the Service_Request_ID
	Data_Structure_ Identifier_RS	1_{hex} to FF $_{hex}$	BYTE	• 00: Reserved • 01 to FF: Data structure depends on the Service_Request_ID • 0xFF - Error
Service Data Response				Depends on the Service-Request-ID: • For the service request IDs, see the CMD and CMD_MODIFIER parameters of the instruction "PE_CMD (Page 3735)". • The specific contents of the response frame can be found in the description of the PE command (see for example the "Start_Pause (Page 3755)" command).

PI Command "Start_Pause" (S7-1200, S7-1500)

Description

Use the PE command "Start_Pause" to start the energy-saving mode. The command Start_Pause can be used to:

- Switch the PE from "ready" state (PE_ready_to_operate) to an energy-saving mode (PE_energy_saving_mode).
- The PE entity can automatically switch between the energy-saving modes. Energy consumption can increase or decrease when you switch energy-saving mode.

Calling the PI command "Start_Pause"

The command "Start_Pause" is called with the instruction "PE_CMD (Page 3735)" with the following parameters:

Parameter	Value	Description
CMD	1	Call of PE command "Start_Pause".
CMD_MODIFIER	0	There are no additional specifications of the command call for command "Start_Pause".
CMD_PARA_LEN	4	Length of the CMD_PARA parameter of 4 bytes.
CMD_PARA	VARIANT	VARIANT pointer to the value for "Pause_Time" (TIME).

Response frame (Service Data Response)

The following response frame data of the PE entity is written to the data block that is referenced at parameter RESPONSE_DATA (see instruction "PE_CMD (Page 3735)"):

Attribute	Value	Data type	Description
PE_Mode_ID	1 to 255	BYTE	Identification number of the energy-saving mode
Reserved	0	BYTE	-

PI Command "End_Pause" (S7-1200, S7-1500)

Description

The PE command "End_Pause" is used to end the energy-saving mode of the PE entity.

Calling the PE command "End_Pause"

The command "End_Pause" is called with the instruction "PE_CMD (Page 3735)" with the following parameters:

Parameter	Value	Description
CMD	2	Call of PE command "End_Pause".
CMD_MODIFIER	0	There are no additional specifications of the command call for command "End_Pause".
CMD_PARA_LEN	0	Length of the CMD_PARA parameter of 0 bytes.
CMD_PARA	irrelevant	-

Response frame (Service Data Response)

The following data of the PE entity response frame is written to the data block referenced at the RESPONSE_DATA parameter (see "PE_CMD (Page 3735)"):

Attribute	Value	Data type	Description
Time_to_operate	-	DWORD	Expected time it takes to switch to the "PE_ready_to_operate" mode.

PI command "Query_modes" - "List_Energy_Saving_Modes" (S7-1200, S7-1500)

Description

Use PE command "Query_modes" and sub-command (modifier) "List_Energy_Saving_Modes" to output all energy-saving modes (PE_Mode_ID) supported by the PE entity.

The query result is written in the form of a response frame to the data block referenced by theRESPONSE_DATA parameter.

Calling the PE command "Query_modes" - "List_Energy_Saving_Modes"

The command "List_Energy_Saving_Modes" is called with the instruction "PE_CMD (Page 3735)" with the following parameters:

Parameter	Value	Description
CMD	3	Call of PE command "Query_modes".
CMD_MODIFIER	1	Specification of the command call: Select the sub-command "List_Energy_Saving_Modes" to output the number and types of energy-saving modes supported.
CMD_PARA_LEN	0	Length of the CMD_PARA parameter of 0 bytes.
CMD_PARA	irrelevant	-

Response frame (Service Data Response)

The following data of the PE entity response frame is written to the data block referenced at the RESPONSE_DATA parameter (see "PE_CMD (Page 3735)"):

Attribute	Value	Data type	Description
Number_of_PE_Mode_IDs	1	BYTE	The number of energy-saving modes.
PE_Mode_IDs	-	Array [...] of BYTE	Array with the IDs of the supported energy-saving modes. The meaning of the individual IDs depends on the PE entity.

PI command "Query_modes" - "Get_Mode" (S7-1200, S7-1500)

Description

You use the PE command "Query_modes" and the sub-command (modifier) "Get_Mode" to output the attributes of the energy-saving mode which is currently enabled.

Calling the PE command "Query_modes" - "Get_Mode"

The call of the command with the instruction "PE_CMD" occurs with the following parameters:

Parameter	Value	Description
CMD	3	Call of the PE command "Query_modes"
CMD_MODIFIER	2	Specification of the command call: Select the sub-command "Get_Mode" to output the status of the mode which is currently enabled
CMD_PARA_LEN	1	Length of the CMD_PARA parameter of 1 byte
CMD_PARA	VARIANT	VARIANT pointer to the value for PE_MODE_ID

Response frame (Service Data Response)

The following data of the PE entity response frame is written to the data block referenced at the RESPONSE_DATA parameter (see "PE_CMD (Page 3735)"):

Attribute	Value	Data type	Description
PE_Mode_ID	• 0 "PE_power_off" mode • 1...254 Energy-saving mode of the PE entity (manufacturer-specific) • 255 "PE_ready_to_operate" mode	BYTE	ID of the energy-saving mode currently enabled
PE_Mode_Attributes	Bit 0: • = 0: Only static energy consumer and time values are available. • = 1: Dynamic energy consumer and time values are available. Bit 1 to 7: • Reserved	BYTE	
Time_min_Pause [1]	Time difference without date	DWORD	Minimum pause time for the PI mode. The minimum pause time is the sum of the values of the following attributes: • Time_to_Pause • Time_to_operate • Time_min_length_of_stay See the description "PAUSE_TIME parameter" of the instruction "PE_START_END: Start and exit energy-saving mode (Page 3730)".
Time_to_Pause [1]	Time difference without date	DWORD	Switch off time: Duration from call of energy-saving mode to start of energy-saving mode (transition duration of PE_ready_to_operate to PE_energy_saving_mode). The switch-off time depends on the PE entity.
Time_to_operate [1]	Time difference without date	DWORD	Switch-on time: Duration of the transition from energy-saving mode (PE_energy_saving_mode) to ready to operate mode (PE_ready_to_operate). The PE entity calculates the duration dynamically in the output operation.
Time_min_length_of_stay [1]		DWORD	Minimum duration of the energy-saving mode in the PE entity
Time_max_length_of_stay [1]		DWORD	Maximum duration of the energy-saving mode in the PE entity
Mode_Power_Consumption [2]		REAL	Power consumption of the PE entity in active energy-saving mode Unit: kW

Attribute	Value	Data type	Description
Energy_ Consumption_ to_pause [2]		REAL	Energy consumption of the PE entity during transition from ready-to-operate mode (PE_ready_to_operate) to energy-saving mode (PE_energy_saving_mode) Unit: kWh
Energy_ Consumption_ to_operate [2]		REAL	Energy consumption of the PE entity during transition from energy-saving mode (PE_energy_saving_mode) to ready-to-operate mode (PE_ready_to_operate) Unit: kWh
[1] If the duration is infinite, 0xFFFFFFFF is output. If the duration is zero, "0" is output. [2] If the data for energy and power consumption by the PE entity is not defined, "0.0" is output as value.			

PI command "PEM_Status" (S7-1200, S7-1500)

Description

The PE command "PEM_Status" is used to query the status of a PE entity energy-saving mode that is currently enabled.

Calling the PE command "PEM_Status"

The command "PEM_Status" is called with the instruction "PE_CMD" with the following parameters:

Parameter	Value	Description
CMD	4	Call of PE command "PEM_Status".
CMD_MODIFIER	0	There are no other specifications of the command call for command "PEM_Status".
CMD_PARA_LEN	0	Length of the CMD_PARA parameter of 0 bytes.
CMD_PARA	irrelevant	-

Response frame (Service Data Response)

The following data of the PE entity response frame is written to the data block referenced at the RESPONSE_DATA parameter (see "PE_CMD (Page 3735)"):

Attribute	Value	Data type	Description
PE_Mode_ID_ Source	• 0 "PE_power_off" mode • 1 to 254 Energy-saving mode of the PE entity (manufacturer-specific) • 255 "PE_ready_to_operate" mode	BYTE	Mode of the PE entity before a PE command is sent.
PE_Mode_ID_ Destination	• 0 "PE_power_off" mode • 1 to 254 Energy-saving mode of the PE entity (manufacturer-specific) • 255 "PE_ready_to_operate" mode	BYTE	Mode of the PE entity after a PE command is executed.
Time_to_operate	Time difference without date	DWORD	Switch-on time: Duration of the transition from energy-saving mode (PE_energy_saving_mode) to ready to operate mode (PE_ready_to_operate). The PE entity calculates the duration dynamically during output.
Remaining_time_to_ destination	Time difference without date	DWORD	Remaining time to switch to another mode.
Mode_Power_ Consumption		REAL	Power consumption of the PE entity in active energy-saving mode. Unit: kW
Energy_ Consumption_ to_Destination		REAL	Energy consumption for the current PI transition Unit: kWh
Energy_ Consumption_ to_operate		REAL	Energy consumption of the PE entity during transition from energy-saving mode (PE_energy-saving mode) to ready-to-operate mode (PE_ready_to_operate) Unit: kWh

PI command "PE_Identify" (S7-1200, S7-1500)

Description

The PE command "PE_Identify" is used to read out the number and description of PE commands supported by the PE entity. The number of specific commands supported depends on the PE entity. As PE_Identify is itself a PE command, at least three supported PE commands are output for a positive response: Start_Pause, End_Pause and PE_Identify.

Call of the PE command "PE_Identify"

The command "PE_Identify" is called with the instruction "PE_CMD (Page 3735)" with the following parameters:

Parameter	Value	Description
CMD	5	Call of the command "PE_Identify".
CMD_MODIFIER	0	There are no other specifications of the command call for command "PE_Identify".
CMD_PARA_LEN	0	Length of the CMD_PARA parameter of 0 bytes.
CMD_PARA	Irrelevant	-

Response frame (Service Data Response)

The following data of the PE entity response frame is written to the data block referenced at the RESPONSE_DATA parameter (see "PE_CMD (Page 3735)"):

Attribute	Value	Data type	Description
Count [1]	6	BYTE	Number of supported PROFIenergy commands
Start_Pause	1	BYTE	First PE command supported (Service_Request_ID)
End_Pause	2	BYTE	...
Query_Modes	3	BYTE	...
PEM_Status	4	BYTE	...
PE_Identify	5	BYTE	...
Query_Measurement	16	BYTE	Last PE command supported (Service_Request_ID)

[1] The number of commands supported is manufacturer-specific and depends on the PE entity used. When all 6 PE commands are supported, the values given are an example of a response frame.

PI Command "Query_Measurement" - "Get_Measurement_list" (S7-1200, S7-1500)

Description

Use the PE command "Query_Measurement" and sub-command (modifier) "Get_measurement_list" to query the specific measured values supported by the PE entity. The supported measured values are output as a list in the data block referenced by the RESPONSE_DATA parameter.

Calling the PE command "Query_Measurement" - "Get_Measurement_list"

The command is called with the instruction "PE_CMD (Page 3735)" with the following parameters:

Parameter	Value	Description
CMD	16	Call of the command "Query_Measurement".
CMD_MODIFIER	1	Specification of the command call: Select the sub-command "Get_Measurement_List" to output a list of supported measured values.
CMD_PARA_LEN	0	Length of the CMD_PARA parameter of 0 bytes.
CMD_PARA	irrelevant	-

Response frame (Service Data Response)

The following data of the PE entity response frame is written to the data block referenced at the RESPONSE_DATA parameter (see "PE_CMD (Page 3735)"):

Attribute	Value	Data type	Description
Count	-	BYTE	Number of measurement IDs
reserved	-	BYTE	
...			
Measurement_ID	-	WORD	First supported Measurement_ID The Measurement_ID is manufacturer-specific. For more information, refer to the manual of the respective PE entity.
Accuracy_Domain	-	BYTE	See "accuracy domain" table.
Accuracy_Class	-	BYTE	See "accuracy classes" table.
Range	-	REAL	Specifies the scale end value of the measured value (only for accuracy domain 1). The attribute range uses the same unit as defined using attribute Measurement_ID (only one unit is used for each Measurement_ID).
...			
Measurement_ID	-	WORD	Last supported Measurement_ID
Accuracy_Domain	-	BYTE	See "accuracy domain" table.
Accuracy_Class	-	BYTE	See "accuracy classes" table.
Range	-	REAL	Specifies the scale end value of the measurement value (only for accuracy domain 1). The attribute range uses the same unit as defined using attribute Measurement_ID (only one unit is used for each Measurement_ID).

Accuracy domains

Accuracy domain	Description
0	Reserved
1	The accuracy deviation is output in percentages of the scale end value. The percentage of the possible deviation is divided into accuracy classes (see table: Accuracy classes of the accuracy domains 1 and 2).

Accuracy domain	Description
2	The accuracy deviation is output in percentages of the current measurement value. The percentage of the possible deviation is divided into accuracy classes (see table: Accuracy classes of the accuracy domains 1 and 2).
3	The measuring accuracy adheres to the standard IEC 61557-12. The function performance classes for performance measurement and monitoring devices (PMD) without external sensors, and the system performance classes for PMD with external sensors, are coded as set out in the "Accuracy classes of accuracy domain 3" table.
4	The entry of the accuracy adheres to the standard EN 50470-3, chapter 8 (see also table: Accuracy classes of accuracy domain 4).

Accuracy classes

Table 11-43 Accuracy classes of the accuracy domains 1 and 2

Accuracy class	0	1	2	3	4	5	6	7	8
Meaning	Reserved	0.01%	0.02%	0.05%	0.1%	0.2%	0.5%	1%	1.5%

Accuracy class	9	10	11	12	13	14	15	>15
Meaning	2%	2.5%	3%	5%	10%	20%	>20%	Not defined

Table 11-44 Accuracy classes of accuracy domain 3

Accuracy class	0	1	2	3	4	5	6	7	8
Meaning	Reserved	0,02	0,05	0,1	0,2	0,5	1	1,5	2

Accuracy class	9	10	11	12	13	14	>13
Meaning	2,5	3	5	10	20	20%	Not defined

Table 11-45 Accuracy classes of accuracy domain 4

Accuracy class	0	1	2	3	4	5	6	>7
Meaning	Reserved	0,5	1,0	1,5	2,0	2,5	3,0	Not defined

PI Command "Query_Measurement" - "Get_Measurement_values" (S7-1200, S7-1500)

Description

Use the PE command "Query_Measurement" and sub-command (modifier) "Get_measurement_values" to output a list of measured values supported by the PE entity. The measured values are output as a list in the data block referenced by the RESPONSE_DATA parameter.

Calling the PE command "Query_Measurement" - "Get_Measurement_values"

The command is called with the instruction "PE_CMD (Page 3735)" with the following parameters:

Parameter	Value	Description
CMD	16	Call of the command "Query_Measurement".
CMD_MODIFIER	2	Specification of the command call: Select the command "Get_Measurement_Values" to output a list of supported measurement values.
CMD_PARA_LEN	0	Depends on the number of measurement values. The length of the parameters results from the attribute count and the sum of the lengths of the attributes for the transferred measurement values.
CMD_PARA	VARIANT	VARIANT pointer to data structure with list of measured values to be queried (see "CMD_PARA parameter").

Parameter CMD_PARA

The structure specified with the VARIANT pointer at the CMD_PARA parameter must have the following setup:

Attribute	Value	Data type	Description
Count	-	BYTE	Number of measured values (Measurement-IDs)
reserved	0	BYTE	Not used
Measurement_ID	-	WORD	First measured value queried
...			
Measurement_ID	-	WORD	Last measured value queried

Response frame (Service Data Response)

The following response frame data of the PE entity is written to the data block that is referenced at parameter RESPONSE_DATA (see "PE_CMD (Page 3735)"):

Attribute	Value	Data type	Description
Count [1]	-	BYTE	Number of measured values (Measurement-IDs)
reserved	0	BYTE	Not used
Length_of_Structure	2 to 65535	WORD	Length of the structure in bytes

Attribute	Value	Data type	Description
Measurement_Data_Structure_ID	1 = simple value	BYTE	Defines the following structure.
Measurement_ID	0 to 65535	WORD	ID of the supported measurement value.
Status_of_ Measurement_Value	1 to 3	BYTE	Status of the measurement value: • 1: Valid • 2: Not supported • 3: Invalid
Transmission_Data_Type	-	REAL	
End_of_demand	-	TOD	Optional time stamping with data type TimeOfDay.
...			
Length_of_Structure	-	WORD	Length of the structure in bytes
Measurement_Data_Structure_ID	-	BYTE	Defines the following structure.
Measurement_ID	-	WORD	ID of the supported measurement value.
Status_of_Measurement_Value	-	BYTE	Status of the measurement value: • 1: Valid • 2: Not supported • 3: Invalid
Transmission_Data_Type	-	REAL	
End_of_demand	-	TOD	Optional time stamping with data type TimeOfDay.

[1] If the data length of the measurement values queried exceeds the size of the PDU (Protocol Data Unit) of the protocol layer, the data will not be completely transferred and only the supported measurement values will be output.

iDevice / iSlave (S7-1200, S7-1500)

PE_I_DEV: Control PROFIenergy commands in the I-Device (S7-1200, S7-1500)

Description

The instruction "PE_I_DEV" is used to process the PROFIenergy profile in the intelligent IO device (iDevice).

- The instruction "PE_I_DEV" is called cyclically by the user program of the iDevice and receives all PROFIenergy commands.
- The PROFIenergy response is generated by configuring an auxiliary block. The response in the pause is fully programmable. The response data must be provided within 10 seconds; otherwise, "State conflict 0x80B5" will occur at the STATUS parameter of the instruction in the IO controller.

Note: The functions are executed by the firmware in a standard PROFIenergy-compatible IO device such as the ET 200S.

No specific knowledge of the PROFIenergy standard is required to use the instruction.

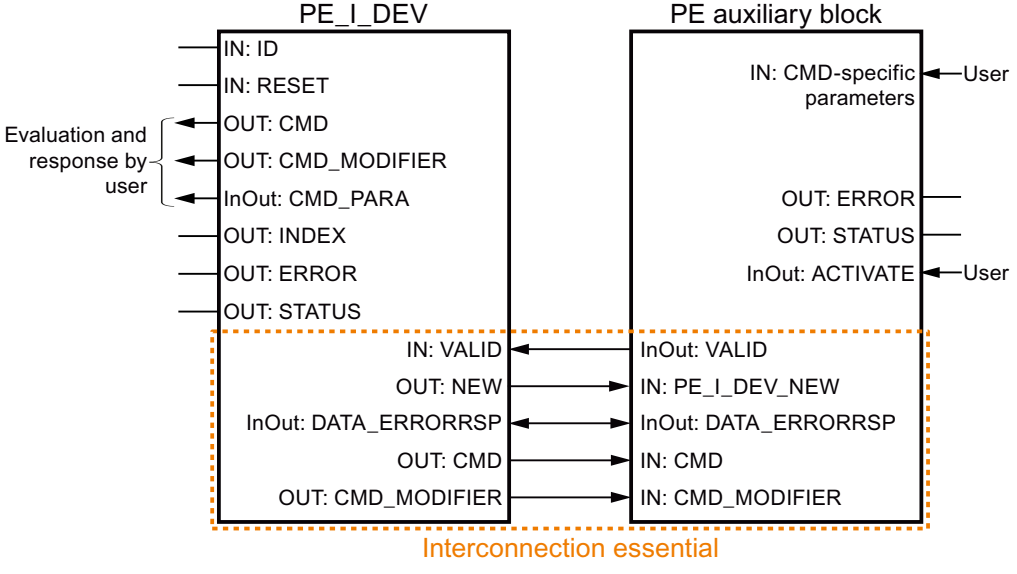
PROFIenergy auxiliary blocks (PE auxiliary blocks)

The PE auxiliary blocks are used to generate the response frame. Enter the response data (in plain text) at the input parameters of the relevant block.

- For each PROFIenergy command, there is a corresponding auxiliary block for a positive response:
  - PE command "Start_Pause": PE_Start_RSP (Page 3770)
  - PE command "End_Pause": PE_End_RSP (Page 3771)
  - PE command "Query_modes" - "List_Energy_Saving_Modes": PE_List_Modes_RSP (Page 3773)
  - PE command "Query_modes" - "Get_Mode": PE_Get_Mode_RSP (Page 3774)
  - PE command "PEM_Status": PE_PEM_Status_RSP (Page 3776)
  - PE command "PE_Identify": PE_Identify_RSP (Page 3777)
  - PE command "Query_Measurement" - "Get_Measurement_list": PE_Measurement_List_RSP (Page 3779)
  - PE command "Query_Measurement" - "Get_Measurement_values": PE_Measurement_Value_RSP (Page 3780)
- Irrespective of the PROFIenergy command, there is an additional shared auxiliary block for a negative response (see PE_Error_RSP (Page 3769)).

Interconnection of the auxiliary blocks

The instruction "PE_I_DEV" and the auxiliary blocks are coordinated. Some of the parameters are simply interconnected. The diagram below shows which of the parameters need to be interconnected.



Parameters

The following table shows the parameters of the instruction "PE_I_DEV":

Parameter	Declaration	Data type	Memory area	Description
RESET	Input	BOOL	I, Q, M, D, L or constant	Resets the instruction.
ID	Input	HW_SUBMODULE	I, Q, M, D, L or constant	Address of the transfer area, which is supplied in the IO controller with the data for PROFIenergy. The hardware identifier can be found in the system constants.
VALID	Input	BOOL	I, Q, M, D, L or constant	The response data for the PROFIenergy controller is ready and can be sent.
CMD_PARA	Output	VARIANT	I, Q, M, D, L	Parameters for: • Get mode: PE_mode_ID • Get measurement values: List of Measurement_IDs (list of IDs of the tags to be read; either one tag or multiple tags may be read at any given time). Maximum length: 234 bytes Note: With S7-1200 the pointer must not reference 64-bit data types (in other words not the data types LWORD, LINT und ULINT).
DATA_ERRORRSP	InOut	VARIANT	I, Q, M, D, L	Pointer to the data area containing the acknowledgment data for the PROFIenergy controller. This must correspond to the pointer which is also used with the auxiliary blocks. Note: With S7-1200 the pointer must not reference 64-bit data types (in other words not the data types LWORD, LINT und ULINT).
INDEX	Output	INT	I, Q, M, D, L	Data record number of the PROFIenergy record (set at 0x80A0)
CMD	Output	INT	I, Q, M, D, L	Service-Request-ID of the PROFIenergy command in accordance with the PROFIenergy profile (see "CMD and CMD_MODIFIER parameters"). Further PE commands (Service-Request-IDs) are possible following extensions of the PROFIenergy profile.
CMD_ MODIFIER	Output	INT	I, Q, M, D, L	PROFIenergy sub-command: • Only when CMD=3 or CMD=16; see "CMD and CMD_MODIFIER parameters". • For all other commands: "0". Further sub-commands are possible following extensions of the PROFIenergy profile.
NEW	Output	BOOL	I, Q, M, D, L	New data available from the PROFIenergy controller.

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	Command completed with error.
STATUS	Output	DWORD	I, Q, M, D, L	Error information (see "STATUS parameter").

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

CMD and CMD_MODIFIER parameters

CMD	CMD_ MODIFIER	PROFIenergy command	Description
01	0	Start_Pause	Start energy-saving mode or switch to a different energy-saving mode.
02	0	End_Pause	End energy-saving mode.
03	1	Query_Modes - List energy saving Modes	Output the energy-saving modes supported.
	2	Query_Modes - Get Mode	Output attributes of the energy-saving mode currently enabled.
04	0	PEM_Status	Query status of energy-saving mode.
05	0	PE_Identify	Read out the number and description of PE commands supported.
16	1	Query_Measurement - Get_Measurement_List	List of measured values supported by the PE entity.
	2	Query_Measurement - Get_Measurement_Values	Output of the measured values of the PE entity.

STATUS parameter

Error information is output at the STATUS output parameter. If it is interpreted as ARRAY[1...4] of BYTE, the error information has the following structure:

Field element	Name	Meaning
STATUS[1]	Function_Num	Cause of the error • B#16#**00**: No error • B#16#**DE**: Error while reading the data record • B#16#**DF**: Error while writing the data record • B#16#**C0**: Error message from the "PE_I_DEV" instruction or the internally used "PRVREC (Page 3706)" and "RCVREC (Page 3704)" communication instructions.
STATUS[2]	Error_Decode	Location of the error ID • **80**: DPV1 error as defined in IEC 61158-6 or application-specific
STATUS[3]	Error_Code_1	Error ID (when Error_Decode = **80**): • **B1**: Write length error (error in write length or insufficient length information on the data type VARIANT).
STATUS[4]	Error_Code_2	For PROFINET errors: Output of IO controller error message If no PROFINET error has occurred, the value in STATUS[4] = "0".

Note

Error messages of the "PRVREC" and "RCVREC" instructions

The "PE_I_DEV" instruction uses the "PRVREC (Page 3706)" and "RCVREC (Page 3704)" instructions for communication. Error messages for these instructions are output in the field elements STATUS[1] to STATUS[4].

For the meaning of the error codes of the "PRVREC" and "RCVREC" instructions, see the description of the corresponding STATUS (Page 3638) parameter.

See also

WRREC: Write data record (Page 3596)

RDREC: Read data record (Page 3578)

Description of PROFIenergy (Page 3728)

Auxiliary blocks of the instruction PE_I_DEV (S7-1200, S7-1500)

PE_Error_RSP: Generate negative answer to command (S7-1200, S7-1500)

Description

The auxiliary block "PE_Error_RSP" (Response with failure) generates a negative response if the command requested is not supported (either in general or temporarily). Response generation does not depend on the command requested.

Parameter

The following table shows the parameters of the "PE_Error_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
ERROR_CODE	Input	BYTE	I, Q, M, D, L or constant	Error number
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.

Parameter	Declaration	Data type	Memory area	Description
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.
DATA_ ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

PE_Start_RSP: Generate answer to command at start of pause (S7-1200, S7-1500)

Description

The auxiliary block "PE_Start_RSP" (Start Pause) generates the response to the PE command Start_Pause (Page 3755). The instruction returns the energy-saving mode to which the device has switched (PE_MODE_ID parameter).

If the response differs depending on the length of the pause, you can return this fact in the feedback on the energy-saving mode (PE_Mode_ID = 1 for a brief pause, PE_Mode_ID = 2 for a longer pause, etc.).

Parameter

The following table shows the parameters of the "PE_Start_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
CMD	Input	INT	I, Q, M, D, L or constant	Service-Request-ID of the PROFIenergy command The parameter must be interconnected with the CMD output parameter of the "PE_I_DEV (Page 3765)" instruction.

Parameter	Declaration	Data type	Memory area	Description
PE_MODE_ID	Input	BYTE	I, Q, M, D, L or constant	PE mode that the process assumes
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.
DATA_ ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

PE_End_RSP: Generate answer to command at end of pause (S7-1200, S7-1500)

Description

The auxiliary block "PE_End_RSP" generates the response to the PE command End_Pause (Page 3755). To switch from the current mode to "Ready_To_Operate" mode, the time required is returned as response.

Parameter

The following table shows the parameters of the "PE_End_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
CMD	Input	INT	I, Q, M, D, L or constant	Service-Request-ID of the PROFIenergy command The parameter must be interconnected with the CMD output parameter of the "PE_I_DEV (Page 3765)" instruction.
Time_to_Operate	Input	DWORD	I, Q, M, D, L or constant	Time required to switch from the current mode to "Ready_To_Operate".
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.
DATA_ ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

PE_List_Modes_RSP: Generate queried energy savings modes as answer (S7-1200, S7-1500)

Description

The auxiliary block "PE_List_Modes_RSP" generates the response to the PE command List_Energy_Saving_Modes (Page 3756). The response generated includes the number and the IDs of energy-saving modes supported.

Parameter

The following table shows the parameters of the "PE_List_Modes_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
CMD	Input	INT	I, Q, M, D, L or constant	Service-Request-ID of the PROFIenergy command The parameter must be interconnected with the CMD output parameter of the "PE_I_DEV (Page 3765)" instruction.
CMD_MODIFIER	Input	INT	I, Q, M, D, L or constant	PROFIenergy sub-command (evaluated only if CMD=3, or CMD=16). The parameter must be interconnected with the CMD_MODIFIER output parameter of the "PE_I_DEV" instruction.
Number_of_PE_Mode_IDs	Input	BYTE	I, Q, M, D, L or constant	Number of energy-saving modes supported. Permitted values: 1 to 254
PE_MODE_ID	Input	VARIANT	I, Q, M, D, L	Points to the area in which the IDs of the energy-saving modes supported (PE_Mode_ID) are stored. Valid range: 1 to 254.
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.

Parameter	Declaration	Data type	Memory area	Description
DATA_ ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

PE_Get_Mode_RSP: Generate queried energy data as answer (S7-1200, S7-1500)

Description

The auxiliary block "PE_Get_Mode_RSP" generates the response to the command Get_Mode (Page 3757). The response frame contains the time, performance, or energy data of an energy-saving mode.

Parameters

The following table shows the parameters of the "PE_Get_Mode_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
CMD	Input	INT	I, Q, M, D, L or constant	Service-Request-ID of the PROFIenergy command The parameter must be interconnected with the CMD output parameter of the "PE_I_DEV (Page 3765)" instruction.
CMD_MODIFIER	Input	INT	I, Q, M, D, L or constant	PROFIenergy sub-command (evaluated only if CMD=3, or CMD=16). The parameter must be interconnected with the CMD_MODIFIER output parameter of the "PE_I_DEV" instruction.
PE_Mode_ID	Input	BYTE	I, Q, M, D, L or constant	ID of the energy-saving mode currently enabled.
PE_Mode_Attributes	Input	BYTE	I, Q, M, D, L or constant	Additional information regarding energy saving mode

Parameter	Declaration	Data type	Memory area	Description
Time_min_Pause	Input	DWORD	I, Q, M, D, L or constant	Minimum pause duration for this PE energy-saving mode. This is the sum of the following three parameters: • Time_to_Pause • Time_to_operate • Time_min_length_of_stay
Time_to_Pause	Input	DWORD	I, Q, M, D, L or constant	Time between the edge at the START parameter (see "PE_I_DEV (Page 3765)") and the requested PE energy-saving mode being reached.
Time_to_Operate	Input	DWORD	I, Q, M, D, L or constant	Maximum switch-on time to "PE_ready_to_operate". The "Time_to_operate" parameter can be used directly for the relevant calculations. The value may be a static maximum, or be calculated dynamically by the PE entity.
Time_min_Lenght_of_stay	Input	DWORD	I, Q, M, D, L or constant	Minimum dwell time of the PE entity in this PE mode.
Time_max_Lenght_of_stay	Input	DWORD	I, Q, M, D, L or constant	Maximum dwell time of the PE entity in this PE mode.
Mode_Power_Consumption	Input	DWORD	I, Q, M, D, L or constant	Energy consumption in the current PE mode in [kW].
Energy_Consum_to_Pause	Input	DWORD	I, Q, M, D, L or constant	Energy consumption from "PE_ready_to_operate" to the current PE mode in [kWh].
Energy_Consum_to_operate	Input	DWORD	I, Q, M, D, L or constant	Energy consumption from the current PE mode to "PE_ready_to_operate" in [kWh].
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.
DATA_ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

PE_PEM_Status_RSP: Generate PEM status as answer (S7-1200, S7-1500)

Description

The auxiliary block "PE_PEM_Status_RSP" generates the response to the command PEM_Status (Page 3759).

Parameter

The following table shows the parameters of the "PE_PEM_Status_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
CMD	Input	INT	I, Q, M, D, L or constant	Service-Request-ID of the PROFIenergy command The parameter must be interconnected with the CMD output parameter of the "PE_I_DEV (Page 3765)" instruction.
PE_MODE_ID_ Source	Input	BYTE	I, Q, M, D, L or constant	Source and Destination for PEM_STATUS. Values: • 0x00: PE_POWER_OFF • 0x01 – 0xFE: manufacturer-specific • 0xFF: PE_READY_TO_OPERATE
PE_MODE_ID_ Destination	Input	BYTE	I, Q, M, D, L or constant	
Time_to_Operate[1]	Input	DWORD	I, Q, M, D, L or constant	Maximum switch-on time to "PE_ready_to_operate". Time_to_operate" can be used directly for the relevant calculations. The value may be a static maximum, or be calculated dynamically by the PE entity.
Remaining_time_ to_destination[1]	Input	DWORD	I, Q, M, D, L or constant	Optional: Time remaining until the requested PE mode. Dynamic value or static maximum value
Mode_Power_ Consumption[2]	Input	DWORD	I, Q, M, D, L or constant	Energy consumption in the current PE mode in [kW].

Parameter	Declaration	Data type	Memory area	Description
Energy_ Consumption_ to_Destination[2]	Input	DWORD	I, Q, M, D, L or constant	Energy consumption in the time until the requested PE mode in [kW].
Energy_ Consumption_ to_operate[2]	Input	DWORD	I, Q, M, D, L or constant	Energy consumption from the current PE mode to "PE_ready_to_operate" in [kWh].
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.
DATA_ ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

[1] If the time period is unlimited, the maximum value "0xFFFFFFFF" can be specified. If the time period is "Zero", "0x00" can be used.

[2] If no energy consumption value has been defined, "0.0" can be specified.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

PE_Identify_RSP: Generate supported PROFIenergy commands as answer (S7-1200, S7-1500)

Description

The auxiliary block "PE_Identify_RSP" generates the response to the command PE_Identify (Page 3761). In the response to the command, specify which PROFIenergy commands are supported. Please note that PE_IDENTIFY is itself a PE command and has to be included in the response.

Parameter

The following table shows the parameters of the "PE_Identify_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
CMD	Input	INT	I, Q, M, D, L or constant	Service-Request-ID of the PROFIenergy command The parameter must be interconnected with the CMD output parameter of the "PE_I_DEV (Page 3765)" instruction.
Start_Pause	Input	BOOL	I, Q, M, D, L or constant	One parameter for each of the relevant PROFIenergy commands: • 0: PE command is not supported • 1: PE command is supported
End_Pause	Input	BOOL	I, Q, M, D, L or constant	
Query_Modes	Input	BOOL	I, Q, M, D, L or constant	
PEM_Status	Input	BOOL	I, Q, M, D, L or constant	
PE_Identify	Input	BOOL	I, Q, M, D, L or constant	
Query_ Measurement	Input	BOOL	I, Q, M, D, L or constant	
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.
DATA_ ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

PE_Measurement_List_RSP: Generate list of supported measured values as answer (S7-1200, S7-1500)

Description

The auxiliary block "PE_Measurement_List_RSP" generates the response to the command Get_measurement_list (Page 3761). In the response, specify which measured values (Measurement-IDs) are supported.

Parameter

The following table shows the parameters of the "PE_Measurement_List_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
CMD	Input	INT	I, Q, M, D, L or constant	Service-Request-ID of the PROFIenergy command The parameter must be interconnected with the CMD output parameter of the "PE_I_DEV (Page 3765)" instruction.
CMD_MODIFIER	Input	INT	I, Q, M, D, L or constant	PROFIenergy sub-command (evaluated only if CMD=3, or CMD=16). The parameter must be interconnected with the CMD_MODIFIER output parameter of the "PE_I_DEV" instruction.
Count	Input	BYTE	I, Q, M, D, L or constant	Number of measured values supported (measurement IDs)
Measurement_List	Input	VARIANT	D	Pointer to the array with the Measurement_IDs supported. For information on the structure of the array in accordance with the PROFIenergy profile, see: PI Command "Query_Measurement" - "Get_Measurement_list" (Page 3761)

Parameter	Declaration	Data type	Memory area	Description
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.
DATA_ ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

PE_Measurement_Value_RSP: Generate queried measured values as answer (S7-1200, S7-1500)

Description

The auxiliary block "PE_Measurement_Value_RSP" generates the response to the command Get_measurement_values (Page 3764). In the response, return the values of the requested measurements.

Parameter

The following table shows the parameters of the "PE_Measurement_Value_RSP" auxiliary block:

Parameter	Declaration	Data type	Memory area	Description
PE_I_DEV_NEW	Input	BOOL	I, Q, M, D, L or constant	The parameter must be interconnected with the NEW output parameter of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block is processed if the parameter value is set to "1".
CMD	Input	INT	I, Q, M, D, L or constant	Service-Request-ID of the PROFIenergy command The parameter must be interconnected with the CMD output parameter of the "PE_I_DEV (Page 3765)" instruction.
CMD_MODIFIER	Input	INT	I, Q, M, D, L or constant	PROFIenergy sub-command (evaluated only if CMD=3, or CMD=16). The parameter must be interconnected with the CMD_MODIFIER output parameter of the "PE_I_DEV" instruction.
Count	Input	BYTE	I, Q, M, D, L or constant	Number of measured values (Measurement_Values).
Measurement_Values	Input	VARIANT	D	Pointer to the array with the measured values (Measurement_IDs). For information on the structure of the array in accordance with the PROFIenergy profile, see PI Command "Query_Measurement" - "Get_Measurement_values" (Page 3764)
ACTIVATE	InOut	BOOL	I, Q, M, D, L	The instruction copies the input parameters to the DATA_ERROR_RSP data area at a rising edge at input ACTIVATE. The parameter is then reset by the instruction. The parameter must be set within 10 seconds after a rising edge is detected at parameter NEW of the "PE_I_DEV (Page 3765)" instruction.
VALID	InOut	BOOL	I, Q, M, D, L	The parameter must be interconnected with the VALID input of the "PE_I_DEV (Page 3765)" instruction. The auxiliary block sets the parameter once the response data for the PROFIenergy controller is available and ready for transmission.
DATA_ERRORRSP	InOut	VARIANT	D	Pointer on the data area in which the response data is stored. The parameter is identical to the pointer for DATA_ERRORRSP of the "PE_I_DEV (Page 3765)" instruction. The addressed data area contains the complete PROFIenergy frame. Minimum length: 244 bytes

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	• "0": No error • "1": Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	• "0": No error • "0x80B1": Error in VARIANT setting, e.g. incorrect range

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

11.4.4.6 Module parameter assignment (S7-1500)

Writing and reading data records (S7-1500)

Principle

Some modules have a write-only system data area to which your program can transfer data records. This area contains data records with numbers from 0 to a maximum of 240. Not every module contains all of the data records (see following table).

Modules can also have a system data area. From your program, you have read-access only to the system data area. This area contains data records with numbers from 0 to a maximum of 240. Not every module contains all of the data records (see following table).

Note

There are modules that have both system data areas. These are physically separate areas and the only thing they have in common is their logical structure.

Write-only system data area

The following table shows the structure of the write-only system data area. This table also shows how the permitted size of the individual data records and which instructions can be used to write them.

Data record number	Contents	Size	Can be written with instruction
0	Parameter	-	WR_DPARM (Page 3793)
1	Parameter	-	WR_DPARM (Page 3793)
2 to 127	User data	Each ≤ 240 bytes	WR_DPARM (Page 3793) WR_REC (Page 3699)
128 to 240	Parameter	Each ≤ 240 bytes	WR_DPARM (Page 3793) WR_REC (Page 3699)

Read-only system data area

The following table shows the structure of the read-only system data area. This table also shows the permitted size of the individual data records and which instructions can be used to read them.

Data record number	Contents	Size	Can be read with instruction
0	Module-specific diagnostics data (set as standard for the entire system)	4 bytes	RD_REC (Page 3694)
1	Channel-specific diagnostics data (incl. data record 0)	4 to 220 Bytes	RD_REC (Page 3694)
2 to 127	User data	Each ≤ 240 bytes	RD_REC (Page 3694)
128 to 240	Diagnostics data	Each ≤ 240 bytes	RD_REC (Page 3694)

System resources

If you start several asynchronous data record transfers one after the other with only short intervals between them, the allocation of system resources by the operating system ensures that all the jobs are executed and that they do not interfere with each other.

If the limits of the system resources are reached, this is indicated in RET_VAL. You remedy this temporary error situation by repeating the job.

The maximum number of "simultaneously" active jobs of a single instruction type depends on the CPU.

RD_DPAR: Read module data record (S7-1500)

Description

You use the instruction to read the data record with the number INDEX of the addressed component from the configured system data. This may be a module in a central rack or a distributed component (PROFIBUS DP or PROFINET IO).

The value TRUE for the output parameter VALID indicates that the data record was successfully transferred to the target range RECORD. In this case, the LEN output parameter contains the length of the read data in bytes.

If an error has occurred during transfer of the data record, this is indicated by the output parameter ERROR . In this case, the output parameter STATUS contains the error information.

Note

Information about the data records

For information on the data record numbers and the data record structure, refer to the product manual for the relevant module.

Functional description

The "RD_DPAR" instruction works asynchronously, that is, its execution extends over multiple calls. You start the data record transfer by calling "RD_DPAR" with REQ = 1.

The output parameter BUSY and bytes 2 and 3 of the output parameter STATUS show the status of the job. Bytes 2 and 3 of STATUS match the output parameter RET_VAL of the instructions that operate asynchronously.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

The transfer of the data record is complete when the output parameter BUSY has the value FALSE.

Parameter

The following table shows the parameters of the "RD_DPAR" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = 1: Read request
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the module (e.g. input module). The number is assigned automatically and is stored in the properties of the module under "HW identifier" (e.g. "Input 0 - 15 > HW identifier").
INDEX	Input	INT	I, Q, M, D, L or constant	Data record number
RECORD	InOut	VARIANT	I, Q, M, D, L	Target range for the read data record
VALID	Output	BOOL	I, Q, M, D, L	New data record was received and is valid
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The job is not yet complete.
ERROR	Output	BOOL	I, Q, M, D, L	ERROR = 1: An error occurred during the reading process.
STATUS	Output	DWORD	I, Q, M, D, L	• Call ID (bytes 2 and 3) or error code • Byte 1: B#16#00, if no error. Otherwise function ID from DPV1-PDU: In the case of error for data record reading B#16#DE, in the case of error for data record writing B#16#DF. If no DPV1 protocol element is used: B#16#C0. • Byte 4: Manufacturer-specific error ID extension
LEN	Output	INT	I, Q, M, D, L	Length of the read data record information

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

Error code* (W#16#...)	Explanation	Restriction
0000	No error	-
7000	First call with REQ=0: No data transfer active; BUSY has the value "0".	-
7001	First call with REQ=1: Data transfer triggered; BUSY has the value "1".	Distributed I/O
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value "1".	Distributed I/O
8092	A data type other than (Array of) bit string or integer was specified at the RECORD parameter.	-
8093	Address specified at parameter LADDR is invalid.	-
80B0	Instruction for module type not possible, or module does not recognize the data record.	-
80B1	The length of the data record to be sent is incorrect.	-
80C1	The data of the previous write job on the module for the same data record has not yet been processed by the module.	-
80C2	The module is currently processing the maximum possible number of jobs for a CPU.	-
80C3	The required resources (memory, etc.) are occupied at this time.	
80C4	Internal temporary error. Job could not be executed. Repeat the job. If this error occurs often, check your installation for sources of electrical interference.	-
80C5	Distributed I/O not available or deactivated.	Distributed I/O
80C6	Data record transfer cancelled due to priority class abort (restart or background).	Distributed I/O
80D0	There is no entry for the module.	-
80D1	The data record number is not configured for the module (data record numbers > 241 are rejected).	-
80D2	The module cannot be assigned parameters according to its type identifier.	-
80D5	The data record is static.	-
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)	-
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

Example

In the following example, you read the complete parameter data record of a digital input module.

Create nine tags in a global data block for storing the data and the structure “record“. (You can find the organization of the structure below.)

SLI_gDB_RD_DPAR			
	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	laddr	HW_IO	257
4	dataRecNbr	Int	128
5	checkedValid	Bool	false
6	busy	Bool	false
7	error	Bool	false
8	status	DWord	16#0
9	lengthRec	Int	0
10	record	Struct	
11	Mode_C0_Bool	Array[0..7] of Bool	

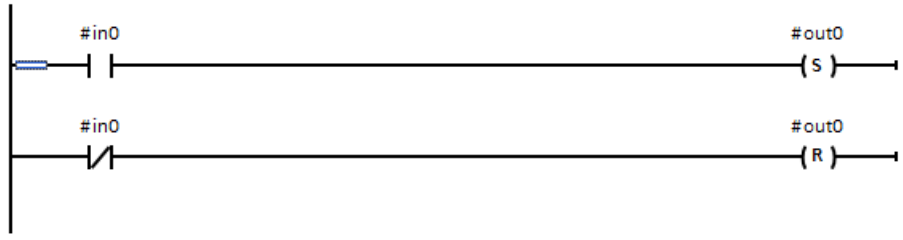
Interconnect parameters: "SLI_itemiseByte_RD_DPAR" FC

To transfer the values of a bit sequence (0..7) create the FC "SLI_itemiseByte_RD_DPAR".

Create the following local tags in it.

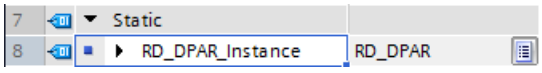
	Name	Data type
1	Input	
2	in0	Bool
3	in1	Bool
4	in2	Bool
5	in3	Bool
6	in4	Bool
7	in5	Bool
8	in6	Bool
9	in7	Bool
10	Output	
11	out0	Bool
12	out1	Bool
13	out2	Bool
14	out3	Bool
15	out4	Bool
16	out5	Bool
17	out6	Bool
18	out7	Bool

Network 1: According to the example of network 1, create interconnections for all bits (0..7).

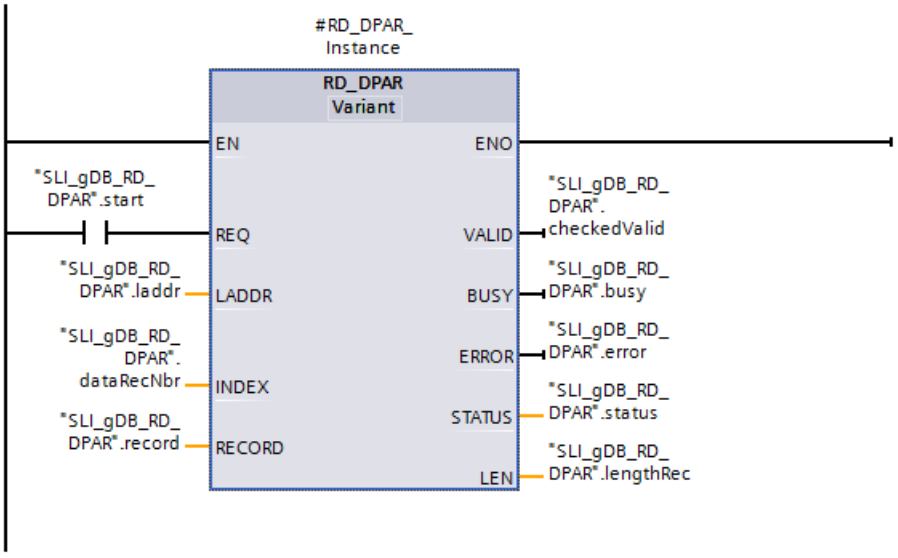


Interconnect parameters: "SLI_FB_RD_DPAR" FB

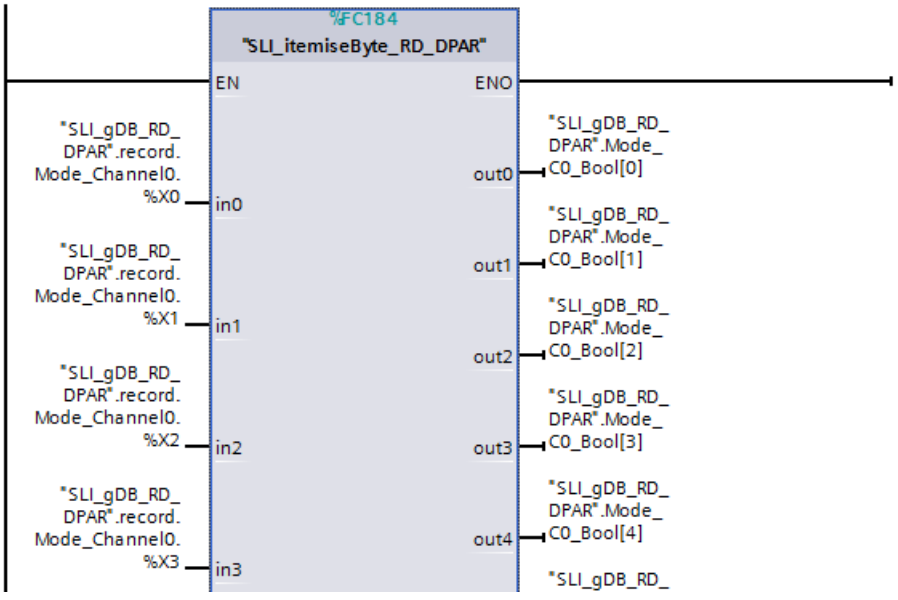
Create the following local tags in the "SLI_FB_RD_DPAR" FB.



Network 1: Interconnect the parameters of the "RD_DPAR" instruction as follows.



Network 2: To convert the tag, interconnect the “Mode_Channel0“ tag as follows All inputs and outputs (0..7) of the "SLI_itemiseByte_RD_DPAR" FC are interconnected in this case.



Hardware identifier

You create an input module and read the hardware ID in the properties of the input module. You store the hardware identifier via the input parameter LADDR ("laddr").

Result (FB "SLI_FB_RD_DPAR")

When the input parameter REQ ("start") returns the signal state "TRUE", the "RD_DPAR" instruction is started. Via the input parameter LADDR ("laddr") the instruction "RD_DPAR" calls the module. Using the input parameter INDEX ("dataRecNbr") the data record with the number "128" of the module is called. During several calls the "RD_DPAR" instruction reads out the data record and saves the data record using the parameter RECORD ("record").

The length of the read data record is stored using the output parameter LEN ("lengthRec"). According to the value of the input parameter MLEN ("maxLen"), has the value “0“ the data record information that is to be read is unlimited.

During processing the output parameter BUSY ("busy") is set to "TRUE" and VALID ("checkedValid") to "FALSE". A successful reading result ("checkedValid" to "TRUE") is not output until processing is complete. The block status is displayed at the output parameter STATUS ("status"). Processing takes place without errors.

SLI_gDB_RD_DPAR

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	laddr	HW_IO	257	16#0101
4	dataRecNbr	Int	128	128
5	checkedValid	Bool	false	TRUE
6	busy	Bool	false	FALSE
7	error	Bool	false	FALSE
8	status	DWord	16#0	16#0000_0000
9	lengthRec	Int	0	34
10	record	Struct		
11	Mode_C0_Bool	Array[0..7] of Bool		

The data record 128 contains the parameter information of all channels of the module. The value "2" of the tag "ByteOfDataRec" relates to the length of the data per channel. It also provides information on the status and the operating mode.

Excerpt of the read data record up to channel 1:

record	Struct		
Version	USInt	0	16
ByteOfDataRec	USInt	0	2
Diag_Channel0	Byte	16#0	16#00
Mode_Channel0	Byte	16#0	16#06
Diag_Channel1	Byte	16#0	16#00
Mode_Channel1	Byte	16#0	16#06

For a more detailed explanation of the data of the data record 128 convert the BYTE values to a bit string. Here an example of the operating mode of the channel 0: Since bits 1 and 2 are set to "TRUE" the channel 0 has an input delay of 3.2 ms. You can find the explanation of the structure of the parameter data record or the channel in the device manual of the module.

11	Mode_C0_Bool	Array[0..7] o...		
12	Mode_C0_Bool[0]	Bool	false	FALSE
13	Mode_C0_Bool[1]	Bool	false	TRUE
14	Mode_C0_Bool[2]	Bool	false	TRUE
15	Mode_C0_Bool[3]	Bool	false	FALSE
16	Mode_C0_Bool[4]	Bool	false	FALSE
17	Mode_C0_Bool[5]	Bool	false	FALSE
18	Mode_C0_Bool[6]	Bool	false	FALSE
19	Mode_C0_Bool[7]	Bool	false	FALSE

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

RD_DPARM: Read data record from configured system data (Page 3791)

Overview of the types of data records (Page 3571)

RD_DPARA: Read module data record asynchronously (S7-1500)

Description

You use the instruction to read the data record with the number RECNUM of a selected module from the configured system data. The read data record is entered in the target range defined by the parameter RECORD .

Note

Information about the data records

For information on the data record numbers and the data record structure, refer to the product manual for the relevant module.

Functional description

The "RD_DPARA" instruction works asynchronously, that is, its execution extends over multiple calls. You start the reading process by calling the instruction with REQ = 1.

The output parameters RET_VAL and BUSY indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

Parameters

The following table shows the parameters of the "RD_DPARA" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = 1: Read request
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the module (e.g. input module). The number is assigned automatically and is stored in the properties of the module under "HW identifier" (e.g. "Input 0 - 15 > HW identifier").
RECNUM	Input	BYTE	I, Q, M, D, L or constant	Data record number (permitted values: 0 to 240)
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code. If no error occurred during the transmission, the following cases are distinguished: • RET_VAL contains the length of the actually read data record in bytes if the target range is larger than the read data record. • RET_VAL contains "0" if the length of the read data record is equal to the length of the target range.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The job is not yet complete.
RECORD	Output	VARIANT	I, Q, M, D, L	Target range for the data record read.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation	Restriction
0000	No error	-
7000	First call with REQ=0: No data transfer active; BUSY has the value 0.	-
7001	First call with REQ=1: Data transfer triggered; BUSY has the value "1".	Distributed I/O
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value "1".	Distributed I/O
8092	A data type other than (Array of) bit string or integer was specified at the RECORD parameter.	-
8093	Address specified at parameter LADDR is invalid.	-
80B0	Instruction for module type not possible, or module does not recognize the data record.	-

Error code* (W#16#...)	Explanation	Restriction
80B1	The length of the data record to be sent is incorrect. With RD_DPARM (Page 3791): The target range spanned by RECORD is too short.	-
80C3	The required resources (memory, etc.) are occupied at this time.	
80C4	Internal temporary error. Job could not be executed. Repeat the job. If this error occurs often, check your installation for sources of electrical interference.	-
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)	-
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

RD_DPARM: Read data record from configured system data (S7-1500)

Use

Note
Restriction on use

The "RD_DPARM" instruction is only intended for the migration from S7-300/400 CPUs to S7-1500 CPUs. In all other cases, use the "RD_DPARA" instruction.

Description

You use instruction to read the data record with the number RECNUM of the addressed module from the configured system data. The read data record is entered into the target range spanned by the RECORD parameter.

Note
Information about the data records

For information on the data record numbers and the data record structure, refer to the product manual for the relevant module.

Functional description

The "RD_DPARM" instruction works asynchronously, that is, its execution extends over multiple calls.

The job status is indicated by the output parameter RET_VAL.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

Parameters

The following table shows the parameters of the "RD_DPARM" instruction:

Parameter	Declaration	Data type	Memory area	Description
IOID	Input	BYTE	I, Q, M, D, L or constant	Address area identifier: • B#16#54 = Peripheral input (PI) • B#16#55 = Peripheral output (PQ) If the module is a mixed module, the area identifier of the lower address must be specified. If the addresses are identical, specify B#16#54.
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the module (e.g. input module). The number is assigned automatically and is stored in the properties of the module under "HW identifier" (e.g. "Input 0 - 15 > HW identifier").
RECNUM	Input	BYTE	I, Q, M, D, L or constant	Data record number (permitted values: 0 to 240)
RET_VAL	Return	INT	I, Q, M, D, L	Length of the data record read in bytes if the read data record fits in the target range and no error occurred in the transfer. If an error occurs while the instruction is being executed, the return value contains an error code.
RECORD	Output	VARIANT	I, Q, M, D, L	Target range for the data record read.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation	Restriction
0000	No error	-
7001	First call: Data transfer started	Distributed I/O
7002	Intermediate call: Data transfer already active	Distributed I/O
8090	Specified logical base address invalid.	-
8092	A type other than BYTE is specified in the VARIANT reference at the RECORD parameter.	-
8093	This instruction is not valid for the module selected with LADDR.	-
80B0	Instruction for module type not possible, or module does not recognize the data record	-
80B1	The target range spanned by RECORD is too short.	-
80C1	The data of the previous write job on the module for the same data record has not yet been processed by the module.	-

Error code* (W#16#...)	Explanation	Restriction
80C2	The module is currently processing the maximum possible number of jobs for a CPU.	-
80C3	The required resources (memory, etc.) are occupied at this time.	-
80C4	Internal temporary error. Job could not be executed. Repeat the job. If this error occurs often, check your installation for sources of electrical interference.	-
80C5	Distributed I/O not available or deactivated	Distributed I/O
80C6	Data record transfer stopped due to priority class abort (restart or background)	Distributed I/O
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)	-
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

WR_DPARM: Transfer data record (S7-1500)

Description

You use the instruction "WR_DPARM" to transfer the data record with the number RECNUM from the configuration data to the addressed module. The data record can be static or dynamic.

Note

Information about the data records

For information on the data record numbers and the data record structure, refer to the product manual for the relevant module.

Parameters

The following table shows the parameters of the "WR_DPARM" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = 1: Write request
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the module (e.g. input module). The number is assigned automatically and is stored in the properties of the module under "HW identifier" (e.g. "Input 0 - 15 > HW identifier").
RECNUM	Input	BYTE	I, Q, M, D, L or constant	Data record number

Parameter	Declaration	Data type	Memory area	Description
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The writing process is not yet complete.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation	Restriction
0000	No error	-
7000	First call with REQ=0: No data transfer active; BUSY has the value 0.	-
7001	First call with REQ=1: Data transfer triggered; BUSY has the value "1".	Distributed I/O
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value "1".	Distributed I/O
8093	Address specified at parameter LADDR is invalid.	-
80A1	Negative acknowledgment during sending of data record to module (module defective or unplugged during sending).	-
80A2	DP protocol error at layer 2, possible hardware/interface error in DP slave.	Distributed I/O
80A3	DP protocol error at user interface/user.	Distributed I/O
80A4	Communication on PBUS+ disrupted	-
80B0	Instruction for module type not possible, or module does not recognize the data record.	-
80B1	The length of the data record to be sent is incorrect.	-
80B2	The configured slot is not occupied.	-
80B3	Actual module type does not correspond to specified module type.	-
80C1	The data of the previous write job on the module for the same data record has not yet been processed by the module.	-
80C2	The module is currently processing the maximum possible number of jobs for a CPU.	-
80C3	The required resources (memory, etc.) are occupied at this time.	
80C4	Internal temporary error. Job could not be executed. Repeat the job. If this error occurs often, check your installation for sources of electrical interference.	-
80C5	Distributed I/O not available or deactivated.	Distributed I/O
80C6	Data record transfer cancelled due to priority class abort (restart or background).	Distributed I/O
80D0	There is no entry for the module.	-

Error code* (W#16#...)	Explanation	Restriction
80D1	The data record number is not configured for the module (data record numbers > 241 are rejected).	-
80D2	The module cannot be assigned parameters according to its type identifier.	-
80D5	The data record is static.	-
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)	-
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

See also

Difference between synchronous and asynchronous instructions (Page 2571)

11.4.4.7 Interrupts (S7-1200, S7-1500)

ATTACH: Attach an OB to an interrupt event (S7-1200, S7-1500)

Description

You use the instruction "ATTACH" to assign an organization block (OB) to a hardware interrupt event.

- You enter the symbolic or numeric name of the organization block in the OB_NR parameter. This will then be assigned to the event specified in the EVENT parameter.
- You select the hardware interrupt event at the EVENT parameter. Hardware interrupt events already generated are listed in the PLC tags under "System constants".

If the event in the EVENT parameter occurs following error-free execution of the "ATTACH" instruction, the organization block in the OB_NR parameter will be called and its program executed.

With the ADD parameter, you specify whether previous assignments of the organization block to other events should be canceled or retained. If the ADD parameter has the value "0", the existing assignments will be replaced by the current assignment.

Hardware interrupt events

Hardware interrupts can be used if events are not pending long enough. They react to these during runtime. Each hardware interrupt can be assigned to hardware interrupt OBs. These OBs contain the reaction to a particular event.

Hardware interrupts can be created for different events. Examples for this are:

- The detection of rising or falling edges for digital inputs.
- Violation of defined low and high limits for analog inputs.
- External reset, overflow/underflow, direction reversal etc. for high-speed counters.

Functional principle

Each hardware interrupt can be assigned to a hardware interrupt OB, which is placed in the queue for processing when the hardware interrupt event occurs. The assignment of OB and event can occur during configuration or at runtime:

- To assign an event to an OB during configuration, select a hardware interrupt OB for an event in the hardware configuration under "Hardware interrupts".
- Use the ATTACH instruction to make the assignment during runtime. The assignment of event and hardware interrupt OB is made via the EVENT and OB_NR parameters.

Parameters

The following table shows the parameters of the "ATTACH" instruction:

Parameter	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_ATT	I, Q, M, D, L or constant	Organization block (numbers up to 32767 are supported.)
EVENT	Input	EVENT_ATT	I, Q, M, D, L or constant	Hardware interrupt event to be assigned to the OB. The hardware interrupt event must first be enabled in the hardware device configuration for inputs or high-speed counters.
ADD	Input	BOOL	I, Q, M, D, L or constant	Effects on previous assignments: • ADD=0 (default): This event replaces all previous event assignments for this OB. • ADD=1: This event is added to the previous event assignments for this OB.
RET_VAL	Return	INT	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#....)	Description
0	No error
8090	OB does not exist
8091	OB is incorrect type

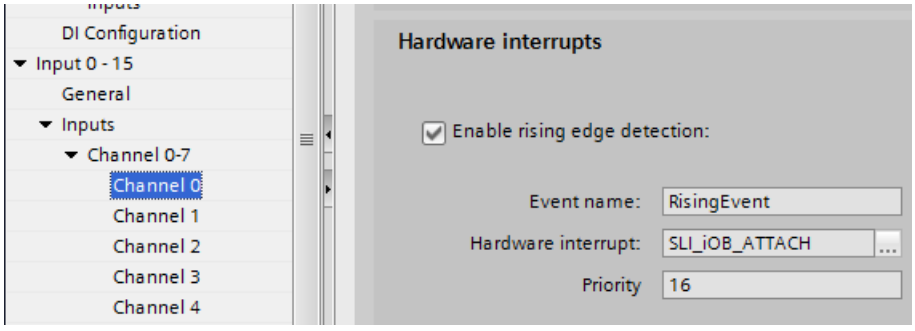
Error code* (W#16#....)	Description
8093	Event does not exist
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

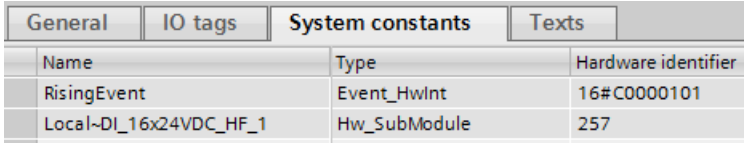
In the following example, you assign a hardware interrupt OB (OB40) to a created hardware interrupt event.

Requirements:

You create an input module (DI) and configure a hardware interrupt event for a positive edge under "Properties > Inputs 0-15 > ... > Channel 0".

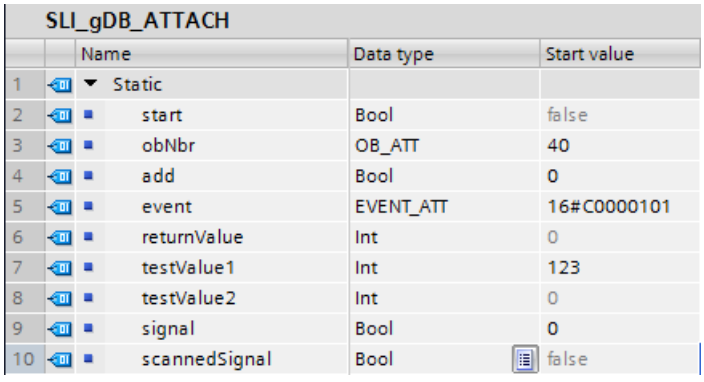


The following hardware ID of the configured hardware interrupt event is stored at the input parameter EVENT ("event").



Data storage

Create nine tags in a global data block for storing the data. You store the number of the OB to be assigned (OB40) with the input parameter OB_NR ("obNbr") of the instruction "ATTACH".



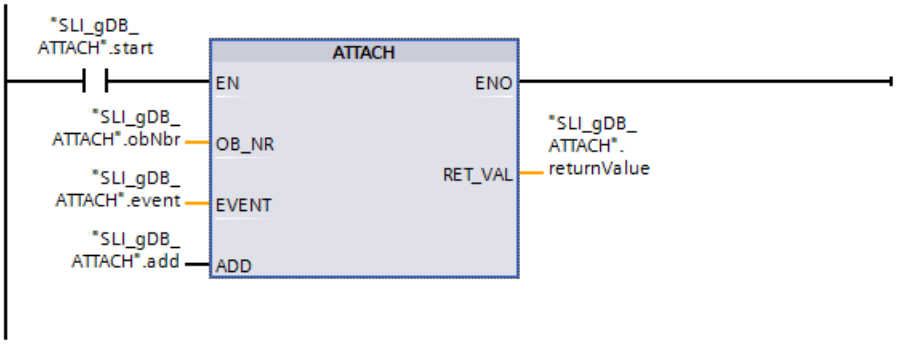
To call the hardware interrupt event, you create a tag table with a PLC tag (address "I0.0").

SLI_Var_ATTACH

		Name	Data type	Address
1		SLI_ATTACH_force	Bool	%I0.0

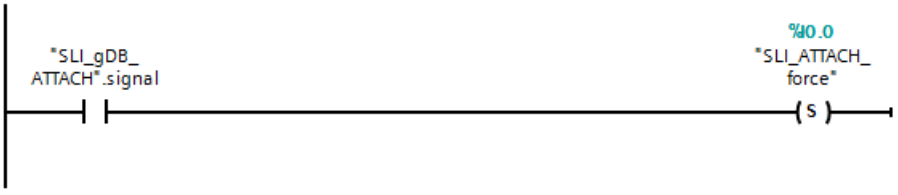
Interconnection

Interconnect the parameters of the instruction as follows.

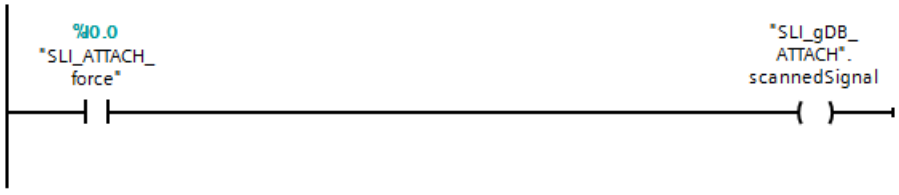


Interconnect the PLC tag as follows to call the hardware interrupt event.

Network 2:

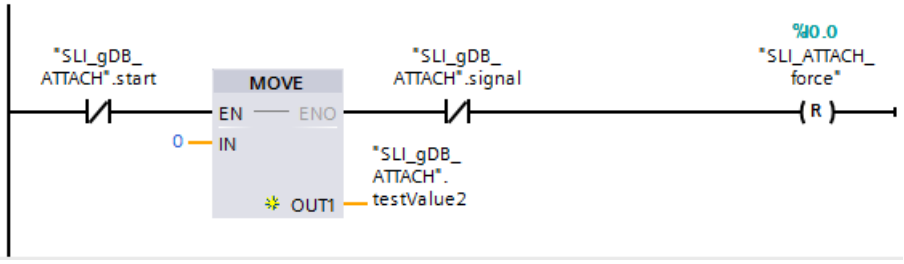


Network 3:

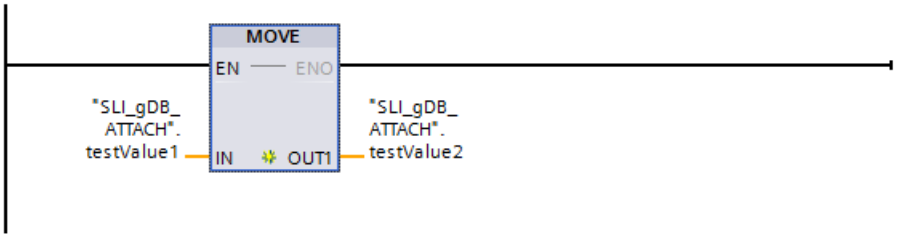


To reset, interconnect "testValue2" and "SLI_ATTACH_force" as follows.

Network 4:



Create the following interconnection in OB40:



Processing

When the normally open contact ("start") supplies the signal state "TRUE", the "ATTACH" instruction is executed and the event is assigned to the OB. With the value "FALSE" of the input parameter ADD ("add"), the event replaces all previous event assignments for OB40. The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.

When the normally open contact ("signal") in network 2 supplies the signal state "TRUE", the PLC tag "SLI_ATTACH_force" is set to the signal state "TRUE" and OB40 is executed. While OB40 is being called, the value of "testValue1" is transferred to "testValue2".

When the normally open contacts ("start" and "signal") supply the signal state "FALSE", the values of "testValue2" and "SLI_ATTACH_force" are reset in network 4.

SLI_gDB_ATTACH

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	obNbr	OB_ATT	40	40
4	add	Bool	0	FALSE
5	event	EVENT_ATT	16#C0000101	16#C000_0101
6	returnValue	Int	0	0
7	testValue1	Int	123	123
8	testValue2	Int	0	123
9	signal	Bool	0	TRUE
10	scannedSignal	Bool	false	TRUE

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

DETACH: Detach an OB from an interrupt event (S7-1200, S7-1500)

Description

You use this instruction to cancel the existing assignment of an organization block to one or more hardware interrupt events during runtime.

You enter the symbolic or numeric name of the organization block in the OB_NR parameter. This assignment to the event specified in the EVENT parameter will be canceled.

- If you have selected an individual hardware interrupt event at the EVENT parameter, the assignment of the OB to this hardware interrupt event is cancelled. All other currently existing assignments remain active. You can select an individual hardware interrupt event using the drop-down list of the operand placeholder.
- If you have not selected a hardware interrupt event, all the events currently assigned to this OB_NR organization block are separated.

Parameters

The following table shows the parameters of the "DETACH" instruction:

Parameter	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_ATT	I, Q, M, D, L or constant	Organization block (numbers up to 32767 are supported.)
EVENT	Input	EVENT_ATT	I, Q, M, D, L or constant	Hardware interrupt event
RET_VAL	Return	INT	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

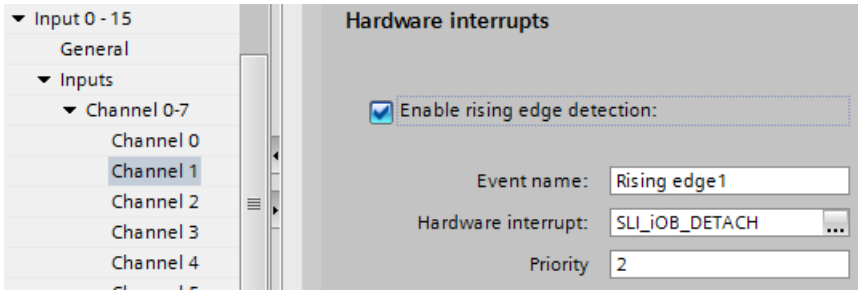
Error code* (W#16#....)	Description
0	No error
1	No assignment exists (warning)
8090	OB does not exist
8091	OB has incorrect type
8093	Event does not exist
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

In the following example, you assign a hardware interrupt event to a hardware interrupt OB (OB41), and dissolve the assignment.

Requirements:

A hardware interrupt OB (OB41 with the name "SLI_iOB_DETACH") has already been created. You create an input module (DI) and configure a hardware interrupt event for a positive edge under "Properties > Inputs 0-15 > ... > Channel 1".



The following hardware ID of the configured hardware interrupt event ("Rising edge 1") is stored at the input parameter EVENT ("event").

DI 16x24VDC HF_1 [DI 16x24VDC HF]

General | IO tags | System constants | Texts

Name	Type	Hardware identi.
Rising edge0	Event_HwInt	16#C0000101
Rising edge1	Event_HwInt	16#C0010101
Local~DI_16x24VDC_HF_1	Hw_SubModule	257

Data storage:

Create ten tags in a global data block for storing the data. You store the number of the OB to be assigned (OB41) with the input parameter OB_NR ("obNbr") of the instructions "ATTACH" and "DETACH".

SLI_gDB_DETACH

	Name	Data type	Start value
1	Static		
2	startAttach	Bool	false
3	add	Bool	0
4	retValAtt	Int	0
5	obNbr	OB_ATT	41
6	event	EVENT_ATT	16#C0010101
7	retValDet	Int	0
8	testValue1	Int	125
9	testValue2	Int	0
10	signal	Bool	0
11	scannedSignal	Bool	false

To call the hardware interrupt event, you create a tag table with a PLC tag (address "I1.0").

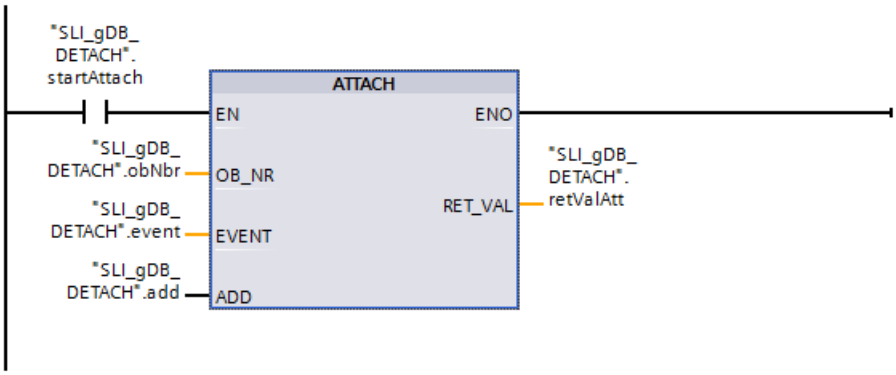
SLI_Var_DETACH

	Name	Data type	Address
1	SLI_DETACH_event	Bool	%I1.0

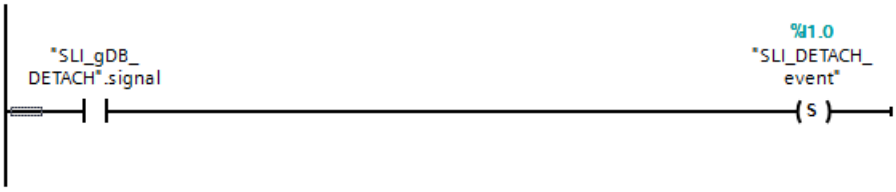
Interconnection:

To call the instructions, create an FB ("SLI_FB_DETACH").

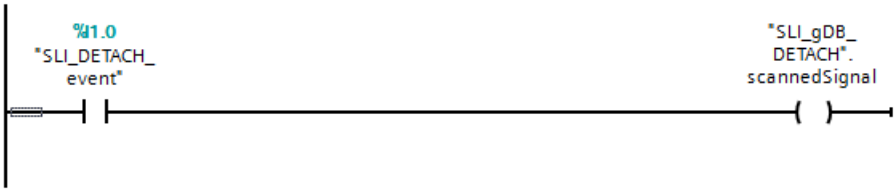
Network 1: Interconnect the parameters of the "ATTACH" instruction as follows.



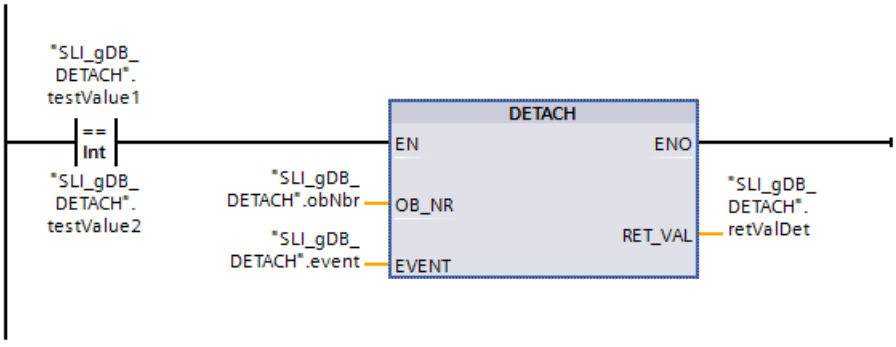
Network 2: Interconnect the PLC tag as follows to call the hardware interrupt event.



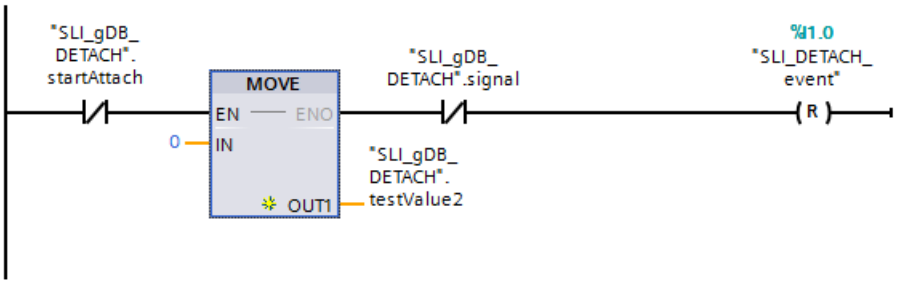
Network 3: You forward the call of the hardware interrupt event as follows.



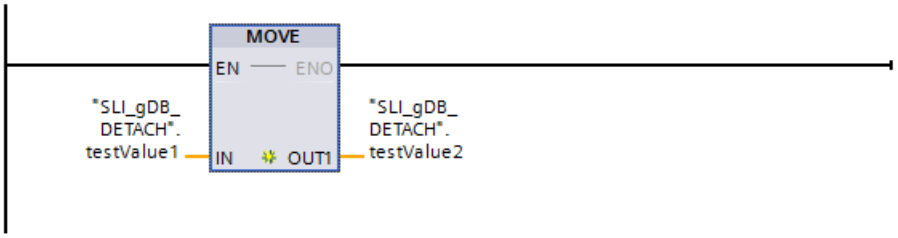
Network 4: Interconnect the parameters of the "DETACH" instruction as follows.



Network 5: To reset, interconnect "testValue2" and "SLI_DETACH_event" as follows.

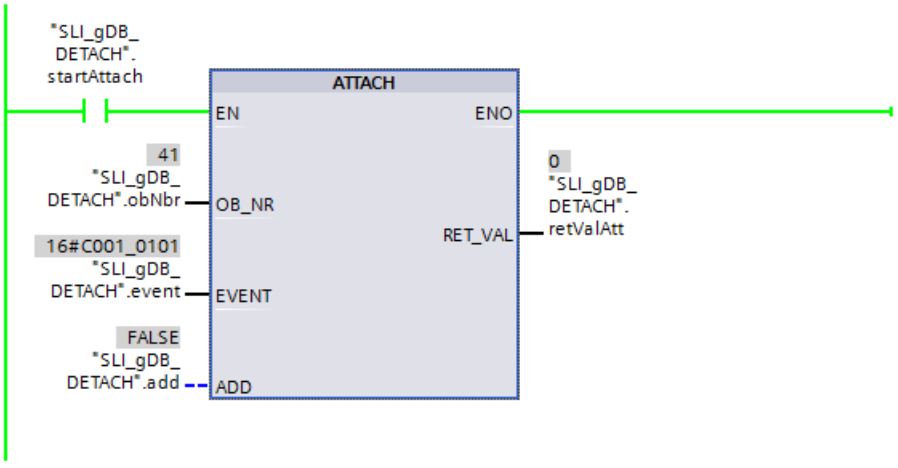


Create the following interconnection in OB41:



Processing

When the normally open contact ("startAttach") supplies the signal state "TRUE", the "ATTACH" instruction is executed and the event is assigned to OB41. With the value "FALSE" of the input parameter ADD ("add"), the event replaces all previous event assignments for OB41. The output parameter RET_VAL ("retValAtt") indicates that processing took place without errors.

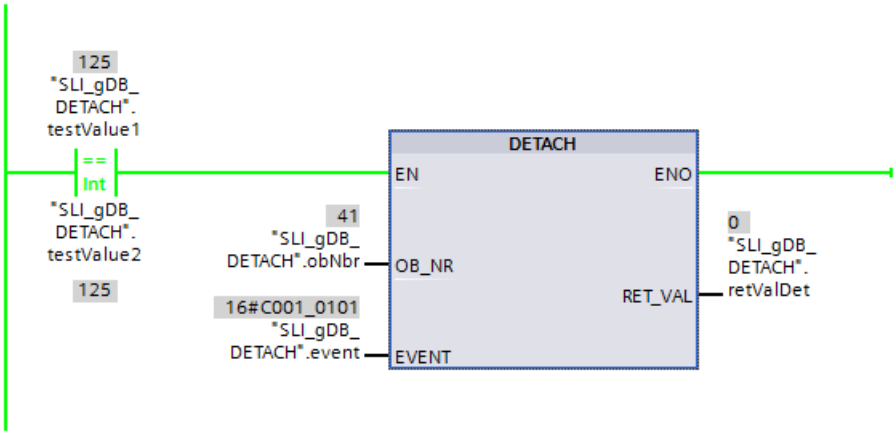


When the normally open contact ("signal") in network 2 supplies the signal state "TRUE", the hardware interrupt event ("SLI_DETACH_event") is set to "TRUE". The OB41 is called and the value of "testValue1" is transferred to "testValue2" in OB41. As long as the hardware interrupt event returns the signal state "TRUE", the OB41 is called and displayed in network 3 by the coil ("scannedSignal").

SLI_gDB_DETACH

	Name	Data type	Start value	Monitor value
1	Static			
2	startAttach	Bool	false	TRUE
3	add	Bool	0	FALSE
4	retValAtt	Int	0	0
5	obNbr	OB_ATT	41	41
6	event	EVENT_ATT	16#C0010101	16#C001_0101
7	retValDet	Int	0	0
8	testValue1	Int	125	125
9	testValue2	Int	0	125
10	signal	Bool	0	TRUE
11	scannedSignal	Bool	false	TRUE

When "testValue1" and "testValue2" supply the same value, the instruction "DETACH" is executed and the event is separated from OB41. The output parameter RET_VAL ("retValDet") indicates that processing took place without errors.



When the tag "startAttach" supplies the signal state "FALSE", the values of "testValue2" and the PLC tag are reset in network 5.

You can find additional information and the program code for the above-named example here:Sample Library for Instructions (Page 2568)

Cyclic interrupt (S7-1200, S7-1500)

SET_CINT: Set cyclic interrupt parameters (S7-1200, S7-1500)

Description

You use this instruction to set the parameters for a cyclic interrupt OB. The start time for a cyclic interrupt OB is generated from the respective time interval of the OB and the phase offset.

- The time interval of an OB is the interval at which the OB is periodically called. For example, if the time interval is 100 µs, the OB will be called every 100 µs during program execution.
- The phase offset is a time interval by which the call of a cyclic interrupt OB is offset. You can use the phase offset to process low priority organization blocks in a precise time base.

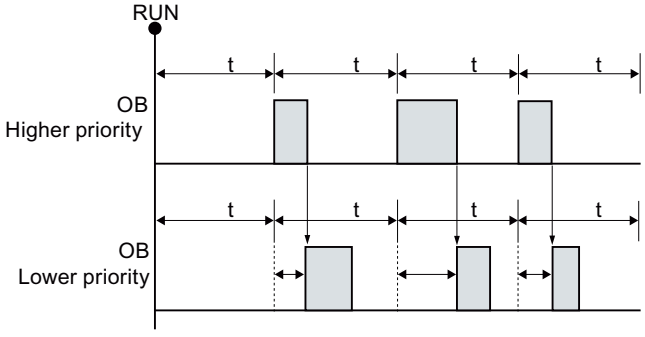
If the OB does not exist or if the time interval used is not supported, a corresponding error alarm is output in the RET_VAL parameter.

A time interval in the CYCLE parameter of "0" means that the OB will not be called.

Functional description

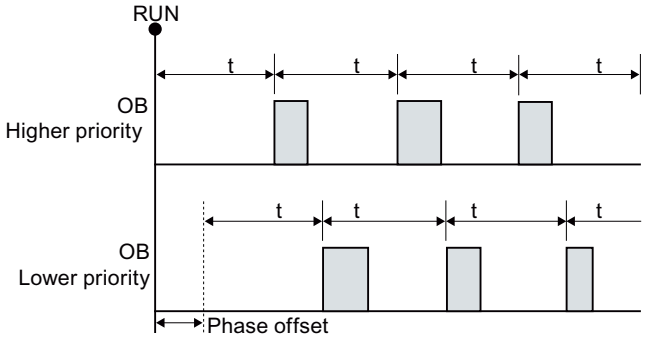
If a lower priority OB and a higher priority OB are called in the same time interval, the lower priority OB will only be called once the higher priority OB has been executed. The call time for the lower priority OB can be offset according to the length of time to execute the higher-priority OB.

OB call without phase offset



If a phase offset is configured for the lower priority OB and the phase offset is greater than the current execution time of the respective higher priority OB, then the block will be called in a fixed time base.

OB call with phase offset



Parameters

The following table shows the parameters of the "SET_CINT" instruction:

Parameter	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_CYCLIC	I, Q, M, D, L or constant	OB number (<32768)
CYCLE	Input	UDINT	I, Q, M, D, L or constant	Time interval in microseconds
PHASE	Input	UDINT	I, Q, M, D, L or constant	Phase offset
RET_VAL	Return	INT	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#....)	Description
0	No error
8090	OB does not exist or is of the wrong type
8091	Incorrect time interval
8092	Incorrect phase offset
80B2	No event assigned to OB
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for cyclic interrupt functions (Page 3808).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

QRY_CINT: Query cyclic interrupt parameters (S7-1200, S7-1500)

Description

You can use this instruction to query the current parameters of a cyclic interrupt OB. The cyclic interrupt OB is identified using the OB_NR parameter.

The values of the queried cyclic interrupt parameters correspond to those at the time the "QRY_CINT" instruction is executed.

Parameters

The following table shows the parameters of the "QRY_CINT" instruction:

Parameter	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_CYCLIC (INT)	I, Q, M, D, L or constant	OB number (<32768) or symbolic addressing via the name of the OB (e.g. OB_MyOB)
CYCLE	Output	UDINT	I, Q, M, D, L	Time interval in microseconds
PHASE	Output	UDINT	I, Q, M, D, L	Phase offset
STATUS	Output	WORD	I, Q, M, D, L	Status of the cyclic interrupt: • Bit 0 to bit 4: see parameter STATUS • Other bits: Always "0"
RET_VAL	Return	INT	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

Bit	Value	Meaning
0	0	Not used (always "0").
1	0	The cyclic interrupt is enabled.
	1	The cyclic interrupt is delayed.
2	0	The cyclic interrupt is not enabled or has expired.
	1	The cyclic interrupt is enabled.
3	0	Not used (always "0").
4	0	An OB with the specified number does not exist.
	1	An OB with the specified number exists.
Other bits		Not used (always "0").

Parameter RET_VAL

If an error occurs, the relevant error code is displayed in the RET_VAL parameter and the STATUS parameter is set to "0".

Error code* (W#16#....)	Description
0	No error.
8090	OB does not exist or is of the wrong type
80B2	No result assigned to OB.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for cyclic interrupt functions (Page 3808).

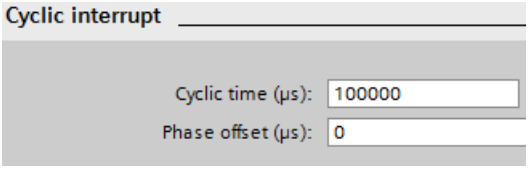
You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

Program example for cyclic interrupt functions (S7-1200, S7-1500)

Introduction

In the following example, you read the phase and the cycle time of a cyclic interrupt OB (OB31) and set a new cycle time.

The OB31 has the following defaults:



Requirement

- To call, create an OB31.
- Create twelve tags in a global data block for storing the data.

SLI_gDB_SET_CINT

	Name	Data type	Start value
1	Static		
2	executeSet	Bool	false
3	obNbr	OB_CYCLIC	31
4	setCycle	UDInt	1000
5	setPhase	UDInt	0
6	returnValueSet	Int	0
7	testTime	Time	T#0ms
8	testTimeLimit	Time	T#2m
9	executeQry	Bool	false
10	returnValueQry	Int	0
11	itemiseStatus	Struct	
12	Cyclic_interrupt_enabled	Bool	false
13	Cyclic_interrupt_not_expired	Bool	false
14	OB_with_obNbr_exists	Bool	false
15	qryCycle	UDInt	0
16	qryPhase	UDInt	0

Interconnect parameters: "SLI_itemiseStatus_CINT" FC

To evaluate the status information, create an FC "SLI_itemiseStatus_CINT".

Create the following local tag in it.

SLI_itemiseStatus_CINT

	Name	Data type
1	Input	
2	inWord	Word

Create interconnections for all relevant bits.

Network 1:



Network 2:

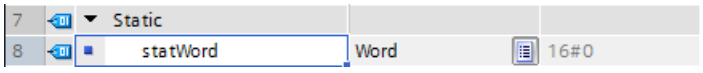


Network 3:

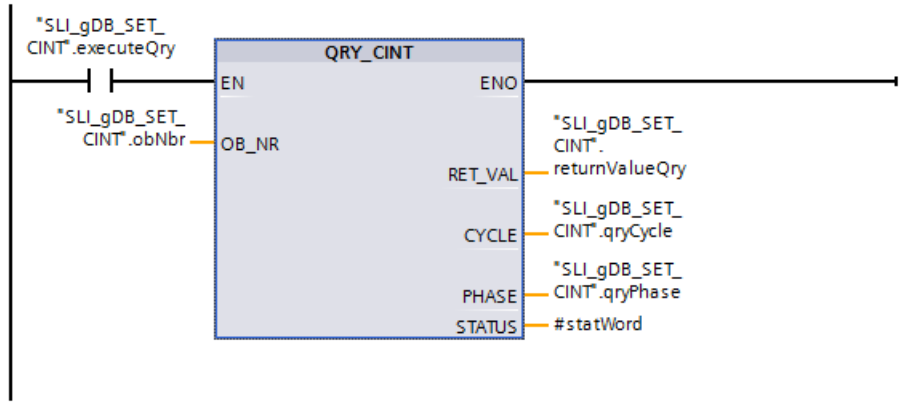


Interconnect parameters: "SLI_FB_SET_CINT" FB

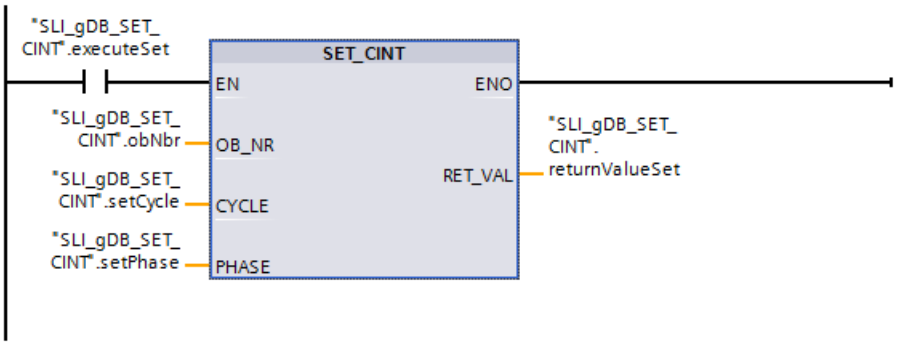
To call the instructions, create an FB ("SLI_FB_SET_CINT"). Create the following local tag in the function block "SLI_FB_SET_CINT".



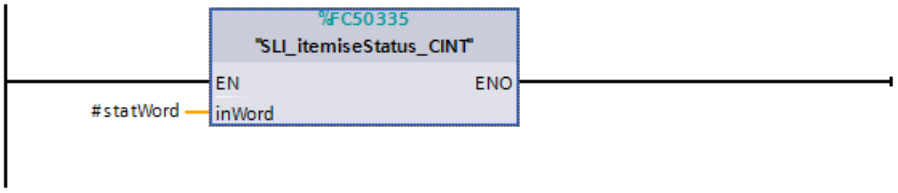
Network 1: Interconnect the parameters of the "QRY_CINT" instruction as follows.



Network 2: Interconnect the parameters of the "SET_CINT" instruction as follows.

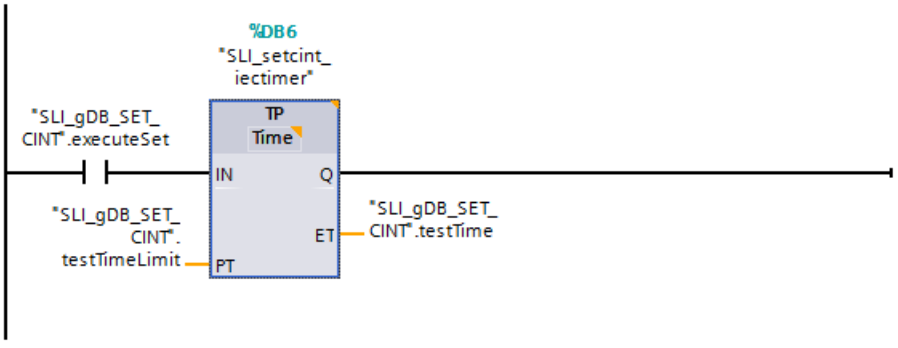


Network 3: Interconnect the parameter of the "SLI_itemiseStatus_CINT" FC as follows.



Interconnect parameters: OB31

Create the following interconnections in OB31.



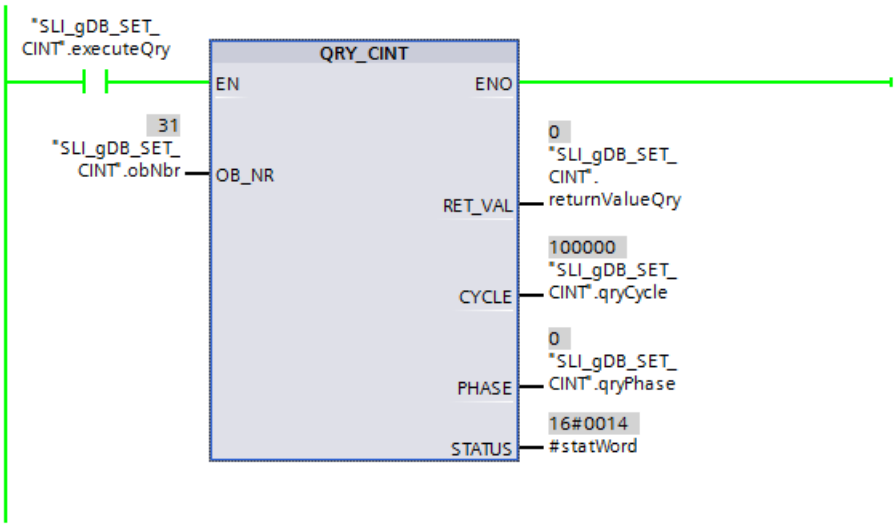
Result of "QRY_CINT"

The number of the OB to be called (OB31) is stored with the input parameter OB_NR ("obNbr") of the instructions "QRY_CINT" and "SET_CINT".

When the normally open contact ("executeQry") supplies the signal state "TRUE", the "QRY_CINT" instruction is executed and the data of OB31 is read.

The cycle time is output at the output parameter CYCLE ("qryCycle"). The phase offset is output at the output parameter PHASE ("qryPhase").
The output parameter STATUS ("#statWord") displays the status of OB31. The output parameter RET_VAL ("returnValueQry") indicates that processing is taking place without errors.

Thee current status information ("#statWord") of the OB31 is interpreted in network 3 via the FC "SLI_itemiseStatus_CINT" and output via the structure "itemiseStatus".



The value of the tag "#statWord" is mapped in the structure "itemiseStatus".

12		itemiseStatus	Struct		
13		Cyclic_interrupt_enabled	Bool	false	TRUE
14		Cyclic_interrupt_not_expired	Bool	false	TRUE
15		OB_with_obNbr_exists	Bool	false	TRUE

Result of "SET_CINT"

When the normally open contact ("executeSet") supplies the signal state "TRUE", the "SET_CINT" instruction is executed. A new cycle time is transferred to OB31 with the input parameter CYCLE ("setCycle") of the instruction "SET_CINT". The value "0" is stored as phase offset at the input parameter PHASE ("setPhase").

The output parameter RET_VAL ("returnValueSet") indicates that processing took place without errors.

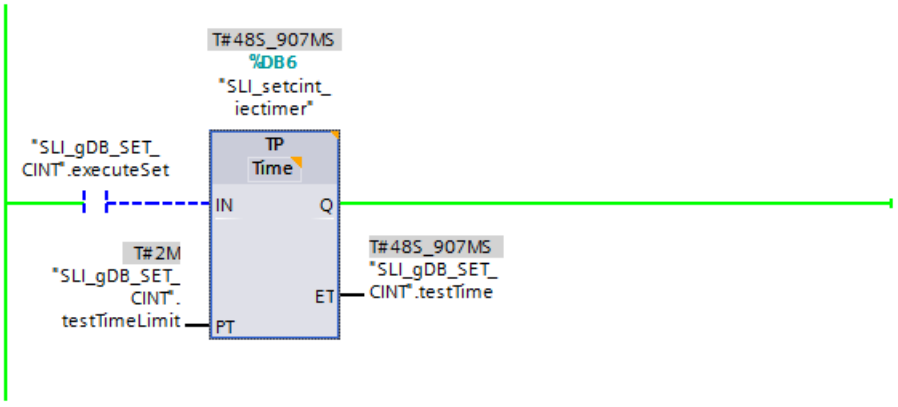
SLI_gDB_SET_CINT

	Name	Data type	Start value	Monitor value
1	Static			
2	executeSet	Bool	false	TRUE
3	obNbr	OB_CYCLIC	31	31
4	setCycle	UDInt	1000	1000
5	setPhase	UDInt	0	0
6	returnValueSet	Int	0	0
7	testTime	Time	T#0ms	T#14S_487MS
8	testTimeLimit	Time	T#2m	T#2M
9	executeQry	Bool	false	TRUE
10	returnValueQry	Int	0	0
11	itemiseStatus	Struct		
12	Cyclic_interrupt_enabled	Bool	false	TRUE
13	Cyclic_interrupt_not_expired	Bool	false	TRUE
14	OB_with_obNbr_exists	Bool	false	TRUE
15	qryCycle	UDInt	0	1000
16	qryPhase	UDInt	0	0

After execution of the instruction "SET_CINT", OB31 retains the new cyclic interrupt data until STOP of the CPU. After the execution, you can set the normally open contact ("executeSet") to the signal state "FALSE" once again. The default properties of OB31 are not affected by the execution of "SET_CINT".

Result of the OB31

When the normally open contact ("executeSet") supplies the signal state "TRUE", an IEC timer "TP" is started in OB31. The IEC timer stops after 2 minutes.



Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Time-of-day interrupt (S7-1200, S7-1500)

SET_TINT: Set time-of-day interrupt (S7-1500)

Description

The instruction "SET_TINT" is used to set the start data and time of the time-of-day interrupt organization blocks from the user program, without having to make settings in the hardware configuration.

- Enter the number of the time-of-day interrupt-OBs at the parameter OB_NR for which you would like to set the start date and the time-of-day.
- Use the parameter SDT and PERIOD to specify when and how often the time-of-day interrupt OB should be called:
  - One-time call: Enter the date and the time-of-day at the parameter SDT. Use the value "0" at parameter PERIOD.
  - Repeated call: Enter the date and the time-of-day of the first call at the parameter SDT. Use the parameter PERIOD to define the time interval in which subsequent calls of the OBs should take place.

Observe the following when setting the start date and start time:

- The date and the time-of-day specification at the parameter SDT refers to the system time.
- The seconds and milliseconds specification is ignored and set to "0" at the start time.
- Only the starting date days 1, 2, ... 28 are possible if you want to specify a monthly time-of-day interrupt OBs. This restriction prevents skipping the monthly call (e.g. 30-day months or in February).
  The setting "end of the month" (W#16#2001) can be used at the parameter PERIOD as an alternative to the 29th 30th and the 31st of the month.

After setting the time-of-day interrupt with "SET_TINT" it still must be activated with the instruction "ACT_TINT".

Note

Additional information about time-of-day interrupt OBs

Other special features for the use of time-of-day interrupt OBs can be found in the descriptions of the organization blocks of the respective CPU:

For the S7-1200: AUTOHOTSPOT

For the S7-1500: AUTOHOTSPOT

The settings at the parameters SDT and PERIOD coincide with the settings for time-of-day interrupt in the properties of the time-of-day interrupt OBs.

Parameters

The following table shows the parameters of the "SET_TINT" instruction:

Parameters	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_TOD	I, Q, M, D, L or constant	Number of the time-of-day interrupt OBs • The numbers 10 to 17 are available for the time-of-day interrupt OBs. • An OB number starting with 123 can also be assigned as an alternative. The OB number is displayed in the program block folder and in the system constants.
SDT	Input	DT	D, L or constant	Start date and start time

Parameters	Declaration	Data type	Memory area	Description
PERIOD	Input	WORD	I, Q, M, D, L or constant	Execution intervals from the starting point SDT to: • W#16#0000 = single execution • W#16#0201 = once every minute • W#16#0401 = once hourly • W#16#1001 = once daily • W#16#1201 = once weekly • W#16#1401 = once monthly • W#16#1801 = once yearly • W#16#2001 = at month's end
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the actual parameter of RET_VAL contains an error code.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error occurred.
8090	Error at the parameter OB_NR (no time-of-day interrupt OB addressed).
8091	Error at the parameter SDT (invalid date and time-of-day specification).
8092	Incorrect input at the parameter PERIOD.
80A1	The set start time is in the past. This error code only occurs at PERIOD = W#16#0000.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for time-of-day interrupt functions (Page 3821).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

See also

ACT_TINT: Enable time-of-day interrupt (Page 3819)

SET_TINTL: Set time-of-day interrupt (S7-1200, S7-1500)

Description

The instruction "SET_TINTL" is used to set the start data and time of the time-of-day interrupt organization blocks from the user program, without having to make settings in the hardware configuration.

- Enter the number of the time-of-day interrupt-OBs at the parameter OB_NR for which you would like to set the start date and the time-of-day.
- Use the parameter SDT and PERIOD to specify when and how often the time-of-day interrupt OB should be called:
  - One-time call: Enter the date and the time-of-day at the parameter SDT. Use the value "0" at parameter PERIOD.
  - Repeated call: Enter the date and the time-of-day of the first call at the parameter SDT. Use the parameter PERIOD to define the time interval in which subsequent calls of the OBs should take place.
- Use the parameter LOCAL to select whether the time specified at the parameter SDT refers to the local time or the system time.
- With the ACTIVATE parameter, you specify when the settings made for the organization block are to be applied:
  - ACTIVATE = true: The settings made are applied directly.
  - ACTIVATE = false: The settings are applied only after "ACT_TINT (Page 3819)" is called:

Observe the following when setting the start date and start time:

- The seconds and milliseconds specification is ignored and set to "0" at the start time.
- Only the starting date days 1, 2, ... 28 are possible if you want to specify a monthly time-of-day interrupt OBs. This restriction prevents skipping the monthly call (e.g. 30-day months or in February).
  The setting "end of the month" (W#16#2001) can be used at the parameter PERIOD as an alternative to the 29th 30th and the 31st of the month.

Please observe the following during the use of local time:

- Changeover from daylight saving to standard time: When calling time-of-day interrupt organization blocks with a start time within the second hour during changeover from daylight saving time to standard time, also use a time-delay interrupt during the first hour of the time changeover.
- Changeover from standard to daylight saving time: If you specify the skipped hour as the time-of-day for the daylight saving time changeover day, the error code 16#8091 will be output at a single execution (PERIOD = W#16#0000).

Note

Additional information about time-of-day interrupt OBs

Other special features for the use of time-of-day interrupt OBs can be found in the descriptions of the organization blocks of the respective CPU:

For the S7-1200: AUTOHOTSPOT

For the S7-1500: AUTOHOTSPOT

The settings at the parameters SDT and PERIOD coincide with the settings for time-of-day interrupt in the properties of the time-of-day interrupt OBs.

Parameters

The following table shows the parameters of the "SET_TINTL" instruction:

Parameters	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_TOD	I, Q, M, D, L or constant	Number of the time-of-day interrupt OBs • The numbers 10 to 17 are available for the time-of-day interrupt OBs. • An OB number starting with 123 can also be assigned as an alternative. The OB number is displayed in the program block folder and in the system constants.
SDT	Input	DTL	D, L or constant	Start date and start time
LOCAL	Input	BOOL	I, Q, M, D, L or constant	• true: Use local time • false: Use system time
PERIOD	Input	WORD	I, Q, M, D, L or constant	Execution intervals from the starting point SDT to: • W#16#0000 = single execution • W#16#0201 = Once every minute • W#16#0401 = once hourly • W#16#1001 = once daily • W#16#1201 = once weekly • W#16#1401 = once monthly • W#16#1801 = once yearly • W#16#2001 = at month's end
ACTIVATE	Input	BOOL	I, Q, M, D, L or constant	• true: Set and activate the time-of-day interrupt • false: Set the time-of-day interrupt and activate only at call of "ACT_TINT (Page 3819)"
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the actual parameter of RET_VAL contains an error code.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error occurred.
8090	Error at the parameter OB_NR (no time-of-day interrupt OB addressed).
8091	Error at the parameter SDT (invalid date and time-of-day specification).
8092	Incorrect input at the parameter PERIOD.
80A1	The set start time is in the past. This error code only occurs at PERIOD = W#16#0000.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

CAN_TINT: Cancel time-of-day interrupt (S7-1200, S7-1500)

Description

The instruction "CAN_TINT" is used to delete the start date and time of a specified time-of-day interrupt organization block. This deactivates the time-of-day interrupt, and the organization block is no longer called.

If you want to reuse the time-of-day interrupt, the start time must be reset (instruction "SET_TINTL (Page 3816)" or "SET_TINT (Page 3813)").

After this, you must reactivate the time-of-day interrupt:

- If you use the instruction "SET_TINT (Page 3813)" or "SET_TINTL (Page 3816)" with parameter ACTIVE=false to set the time-of-day interrupt, call "ACT_TINT (Page 3819)".
- By using the instruction "SET_TINTL (Page 3816)" you can also activate the time-of-day interrupt directly with the parameter ACTIVE=true.

Parameters

The following table shows the parameters of the "CAN_TINT" instruction:

Parameters	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_TOD	I, Q, M, D, L or constant	Number of the time-of-day interrupt OB, whose starting date and time-of-day are to be deleted.
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the actual parameter of RET_VAL contains an error code.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error occurred.
8090	Error at the parameter OB_NR.
80A0	No starting date/time-of-day defined for the affected time-of-day interrupt OB.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for time-of-day interrupt functions (Page 3821).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

ACT_TINT: Enable time-of-day interrupt (S7-1200, S7-1500)

Description

The instruction "ACT_TINT" is used to activate a time-of-day interrupt organization block from the user program. A prerequisite for the execution of the instruction is that the starting date and time-of-day are already set for the time-of-day interrupt OB.

Parameters

The following table shows the parameters of the "ACT_TINT" instruction:

Parameters	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_TOD	I, Q, M, D, L or constant	Number of the time-of-day interrupt OBs • The numbers 10 to 17 are available for the time-of-day interrupt OBs. • An OB number starting with 123 can also be assigned as an alternative. The OB number is displayed in the program block folder and in the system constants.
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the actual parameter of RET_VAL contains an error code.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error occurred.
8090	Error at the parameter OB_NR (no time-of-day interrupt OB addressed).
80A0	Start date and time-of day not set for the relevant time-of-day interrupt OB.
80A1	The activated time is in the past. If the time-of-day interrupt is to be executed only once, only this error occurs.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for time-of-day interrupt functions (Page 3821).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

QRY_TINT: Query status of time-of-day interrupt (S7-1200, S7-1500)

Description

You can use this instruction to display the status of a time-of-day interrupt organization block in the STATUS output parameter.

Parameters

The following table shows the parameters of the "QRY_TINT" instruction:

Parameter	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_TOD	I, Q, M, D, L or constant	Number of the time-of-day interrupt OB whose status is queried The OB number is displayed in the program block folder and in the system constants.
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the actual parameter of RET_VAL contains an error code.
STATUS	Output	WORD	I, Q, M, D, L	Status of the time-of-day interrupt (see below)

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error occurred. Exception: Status message "0" on bit 4 (no OB with this number created).
8090	Error at the parameter OB_NR. Possible causes: • The value at the parameter OB_NR is not an OB number (<1 or > 32767) supported by the CPU. • The OB number does not address a time-of-day interrupt OB.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Parameter STATUS

If an error occurs (see RET_VAL parameter), "0" is output in the STATUS parameter.

Bit	Value	Meaning
0	Always "0"	Irrelevant
1	0	The time-of-day interrupt is enabled.
	1	The time-of-day interrupt is disabled.
2	0	Time-of-day interrupt is not activated or has elapsed.
	1	The time-of-day interrupt is activated.
4	0	An OB with an OB number as specified at OB_NR parameter does not exist.
	1	An OB with an OB number as specified at OB_NR parameter does exist.
6	0	The time-of-day interrupt is based on the system time
	1	The time-of-day interrupt is based on the local time
Other		Always "0"

Example

You can find the example here: Program example for time-of-day interrupt functions (Page 3821).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

Program example for time-of-day interrupt functions (S7-1200, S7-1500)

Introduction

In the following example, you read the status of a time-of-day interrupt OB (OB10), and set a start time as well as the call frequency. Once OB10 has been started, disable OB10 once again.

The OB10 has the following defaults:



Requirement

Create 13 tags and a structure in a global data block for storing the data.

SLI_gDB_SET_TINT

		Name	Data type	Start value
1		Static		
2		obNbr	OB_TOD	10
3		executeQry	Bool	false
4		itemiseStatus	Struct	
5		T-o-d_interrupt_enabled	Bool	false
6		T-o-d_interrupt_activated	Bool	false
7		OB_with_obNbr_exists	Bool	false
8		T-o-d_interrupt_on_system_time	Bool	false
9		T-o-d_interrupt_on_local_time	Bool	false
10		returnValueQry	Int	0
11		executeSet	Bool	false
12		startDateTime	Date_And_Time	DT#1990-01-01-0
13		presetTime	Time	T#5M
14		periodOfExecute	Word	16#0201
15		returnValueSet	Int	0
16		executeAct	Bool	false
17		returnValueAct	Int	0
18		executeCan	Bool	false
19		returnValueCan	Int	0
20		testCurrCount	Int	0

Interconnect parameters: "SLI_itemiseStatus_TINT" FC

To evaluate the status information, create an FC "SLI_itemiseStatus_TINT".

Create the following local tag in it.

SLI_itemiseStatus_TINT

		Name	Data type
1		Input	
2		inWord	Word

Create interconnections for all relevant bits.

Network 1:



Network 2:



Network 3:



Network 4:



Network 5:

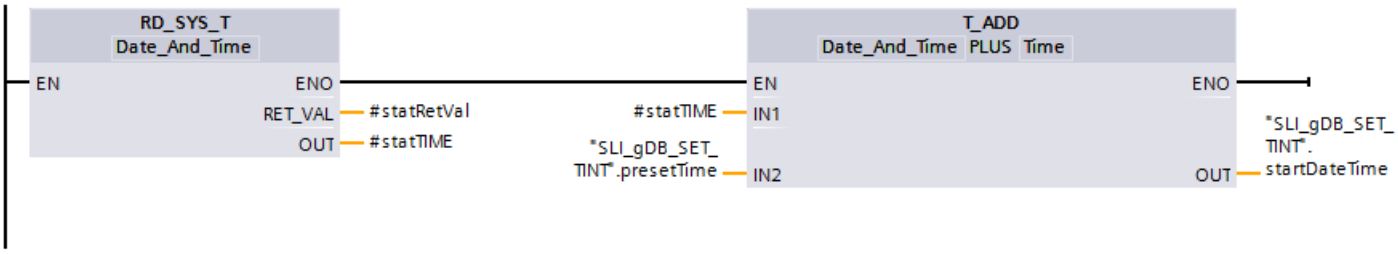


Interconnect parameters: "SLI_retrievetime_TINT" FC

To read the date (system time) and define the start time of OB10, create an FC "SLI_retrievetime_TINT". Create the following local tags in the FC.

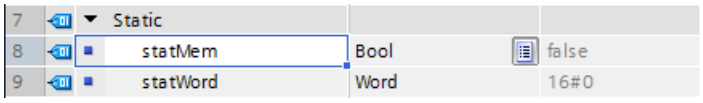


Network 1: Interconnect the parameters as follows. "RD_SYS_T" reads the system time. Use "T_ADD" to add 5 minutes to the system time.

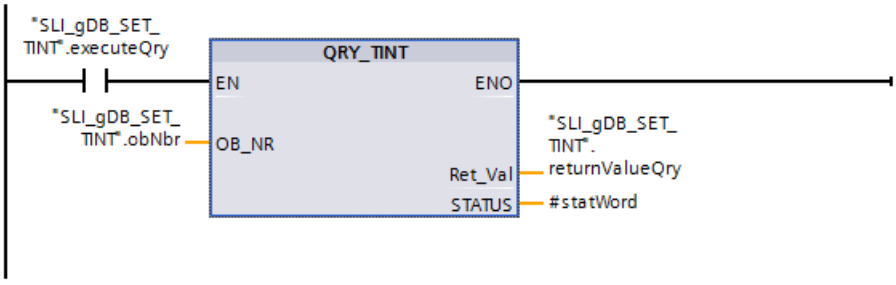


Interconnect parameters: "SLI_FB_SET_TINT" FB

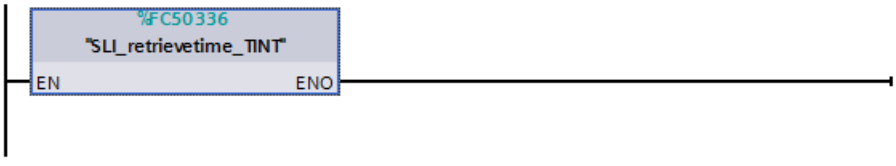
To call the instructions, create an FB ("SLI_FB_SET_TINT"). Create the following local tags in the function block "SLI_FB_SET_TINT".



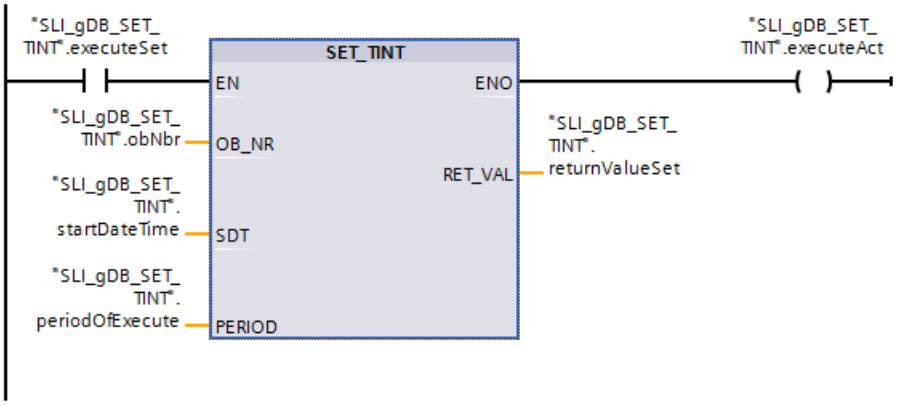
Network 1: Interconnect the parameters of the "QRY_TINT" instruction as follows.



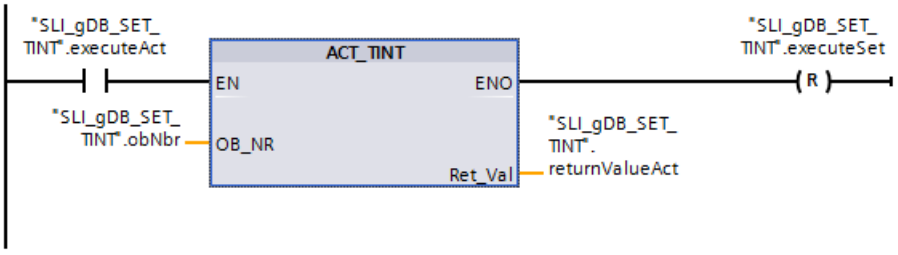
Network 2: You call the FC "SLI_retrievetime_TINT" in network 2 of the FB.



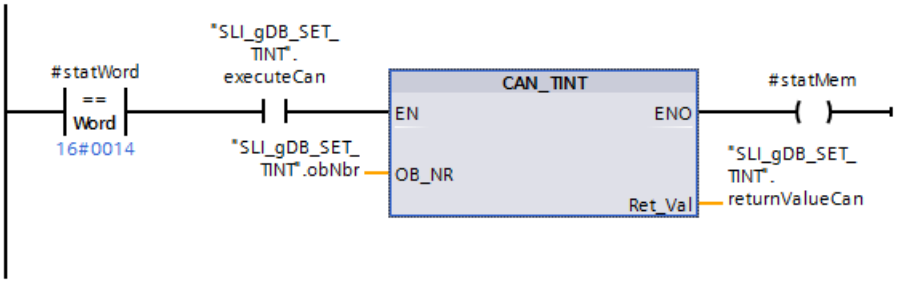
Network 3: Interconnect the parameters of the "SET_TINT" instruction as follows.



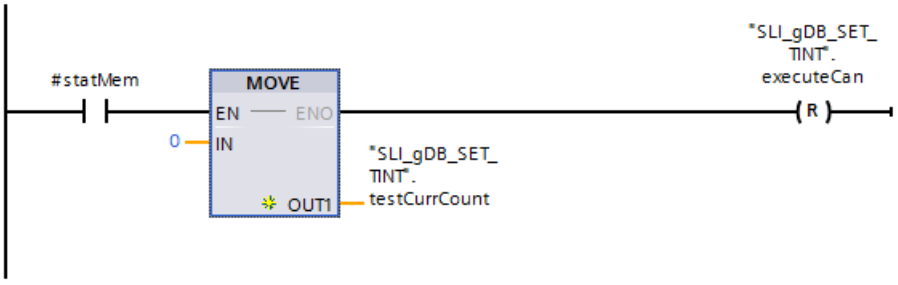
Network 4: Interconnect the parameters of the "ACT_TINT" instruction as follows.



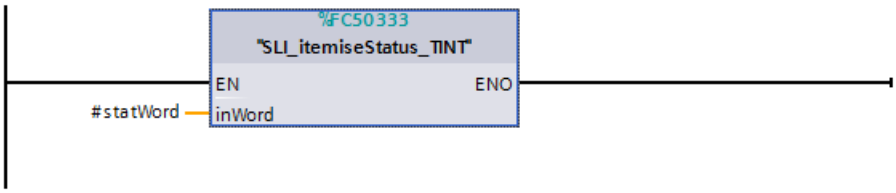
Network 5: Interconnect the parameters of the "CAN_TINT" instruction as follows.



Network 6: Interconnect the parameters that are to be reset with the execution of "CAN_TINT" as follows.

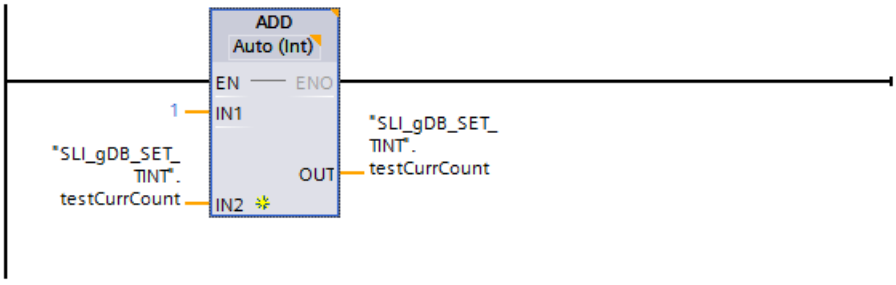


Network 7: Interconnect the parameter of the "SLI_itemiseStatus_TINT" FC as follows.



Interconnect parameters: OB10

Create the following interconnections in OB10.

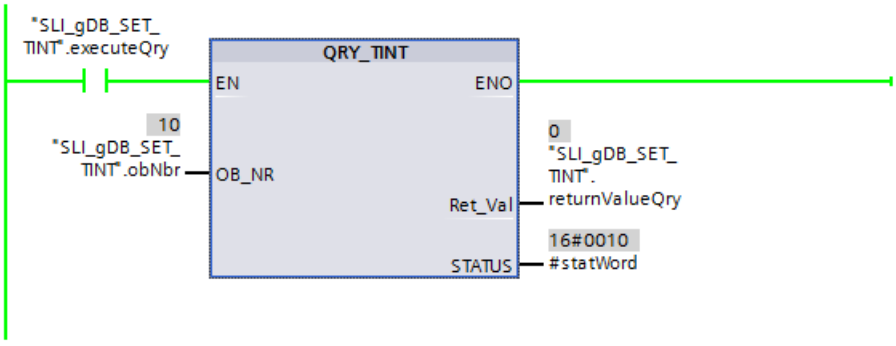


Result of "QRY_TINT"

The instructions receive the number of the OB to be called (OB10) with the input parameter OB_NR ("obNbr").

Network1:

When the normally open contact ("executeQry") supplies the signal state "TRUE", the "QRY_TINT" instruction is executed and the data of OB10 is read. The output parameter STATUS ("#statWord") displays the status of OB10. The output parameter RET_VAL ("returnValueQry") indicates that processing is taking place without errors.



The value of the tag "#statWord" is mapped in the structure "itemiseStatus".

4		itemiseStatus	Struct		
5		T-o-d_interrupt_enabled	Bool	false	TRUE
6		T-o-d_interrupt_activated	Bool	false	FALSE
7		OB_with_obNbr_exists	Bool	false	TRUE
8		T-o-d_interrupt_on_system_time	Bool	false	TRUE
9		T-o-d_interrupt_on_local_time	Bool	false	FALSE

Result of "SET_TINT"

Network 3: When the normally open contact ("executeSet") supplies the signal state "TRUE", the "SET_TINT" instruction is executed. The input parameter SDT ("startDateTime") transfers the defined start time (see Network 2) to OB10. "Every minute" (value "16#0201") is stored as call frequency at the input parameter PERIOD ("periodOfExecute").

The output parameter RET_VAL ("returnValueSet") indicates that processing took place without errors.

The execution of SET_TINT sets the coil ("executeAct") to "TRUE" and the instruction "ACT_TINT" is executed in network 4.

Result of "ACT_TINT"

Network 4: The instruction "ACT_TINT" activates OB10. The output parameter RET_VAL ("returnValueAct") indicates that processing took place without errors. The reset ("executeSet") in the signal state "TRUE" automatically ends the execution of the SET_TINT instruction and therefore also the execution of the ACT_TINT instruction.

Note

After execution of the instruction "SET_TINT", OB10 retains the new call data until STOP of the CPU. The default properties of OB10 are not affected by the execution of the instruction "SET_TINT".

Result of the OB10

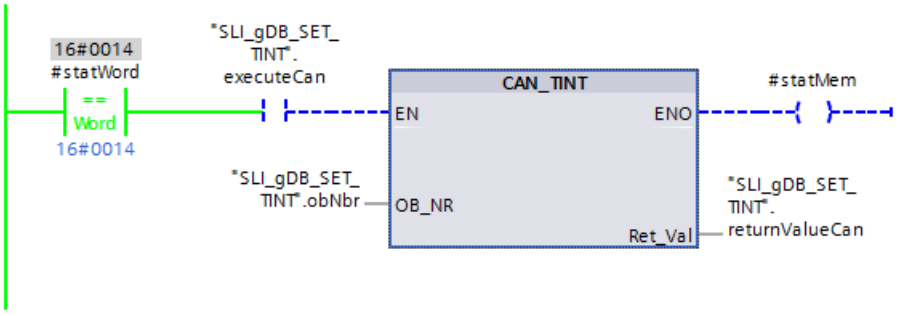
With each call of OB10, the instruction "ADD" is executed and the tag "testCurrCount" is incremented by "1". OB10 is called (according to the value of "periodOfExecute") once every minute.

SLI_gDB_SET_TINT

	Name	Data type	Start value	Monitor value
1	Static			
2	obNbr	OB_TOD	10	10
3	executeQry	Bool	false	TRUE
4	itemiseStatus	Struct		
5	T-o-d_interrupt_enabled	Bool	false	TRUE
6	T-o-d_interrupt_activated	Bool	false	TRUE
7	OB_with_obNbr_exists	Bool	false	TRUE
8	T-o-d_interrupt_on_system_time	Bool	false	TRUE
9	T-o-d_interrupt_on_local_time	Bool	false	FALSE
10	returnValueQry	Int	0	0
11	executeSet	Bool	false	FALSE
12	startDateTime	Date_And_Time	DT#1990-01-01-0	DT#2016-11-30-17...
13	presetTime	Time	T#5M	T#5M
14	periodOfExecute	Word	16#0201	16#0201
15	returnValueSet	Int	0	0
16	executeAct	Bool	false	FALSE
17	returnValueAct	Int	0	0
18	executeCan	Bool	false	FALSE
19	returnValueCan	Int	0	0
20	testCurrCount	Int	0	5

Result of "CAN_TINT"

Network 5: When OB10 is set and enabled ("#statWord" has the value "16#0014"), the comparator supplies the signal state "TRUE" and CAN_TINT can be executed.



When the normally open contact ("executeCan") supplies the signal state "TRUE", the "CAN_TINT" instruction is executed. The instruction disables OB10 and deletes the call data of OB10 set with SET_TINT. The output parameter RET_VAL ("returnValueCan") indicates that processing took place without errors.

The execution of CAN_TINT sets the coil ("#statMem") to "TRUE" and network 6 is executed.

Result of the reset

Network 6: When the normally open contact ("#statMem") supplies the signal state "TRUE", the tag "testCurrCount" is reset to the value "0" with the instruction "MOVE". Reset ("executeCan") in the signal state "TRUE" also automatically ends the execution of the instruction "CAN_TINT".

Result of the status information "#statWord"

The FC "SLI_itemiseStatus_TINT" is called in network 7. The significant bits of the tag "#statWord" are evaluated and output as Boolean tags with plain-text name. When a tag of the structure "itemiseStatus" supplies the signal state "TRUE", its statement is correct.

Additional information

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Time-delay interrupt (S7-1200, S7-1500)

Using time-delay interrupts (S7-1200, S7-1500)

Definition

After you have called the "SRT_DINT (Page 3830)" instruction, the operating system generates an interrupt after the specified delay time has elapsed. The configured time-delay interrupt OB is called.

Prerequisites for the call

Before a time-delay interrupt can be called by the operating system, the following conditions must be met:

- The time-delay interrupt OB must be started by the "SRT_DINT (Page 3830)" instruction.
- The time-delay interrupt OB must not be deselected during configuration.
- The time-delay interrupt OB must exist in the CPU.

Purpose of the instructions "SRT_DINT", "CAN_DINT" and "QRY_DINT"

You use the instructions to

- Start time-delay interrupts ("SRT_DINT (Page 3830)")
- Cancel time-delay interrupts ("CAN_DINT (Page 3832)")
- Query time-delay interrupts ("QRY_DINT (Page 3833)")

Effects on the time-delay interrupt

The following table lists a number of different situations and explains the effect they have on a time-delay interrupt.

If ...	and ...	Then ...
A time-delay interrupt is started (by calling "SRT_DINT (Page 3830)")	The time-delay interrupt has already started,	The delay time is overwritten; the time-delay interrupt is started again.
	The time-delay interrupt OB does not exist at the time of the call,	a diagnostic buffer entry is made and the CPU continues to run.
	The interrupt is started in a startup OB and the delay time elapses before the CPU changes to RUN,	The call of the time-delay interrupt OB is delayed until the CPU is in RUN mode.
The delay time has elapsed,	A previously started time-delay interrupt OB is still being executed,	The operating system generates a time error (calls OB 80). If OB 80 does not exist, the CPU continues to run).

Response to warm restart and cold restart

During a warm restart or a cold restart, all the time-delay interrupt settings made in the user program by means of instructions are cleared.

Starting in a startup OB

A time-delay interrupt can be started in a startup OB. Two conditions must be satisfied to call the time-delay OB:

- The delay time must have elapsed.
- The CPU must be in the RUN mode.

If the delay time has elapsed and the CPU is not yet in the RUN mode, the time-delay interrupt OB call is delayed until the CPU is in RUN mode. The time-delay interrupt OB is then called before the first instruction in OB Main [OB 1] is executed.

SRT_DINT: Start time-delay interrupt (S7-1200, S7-1500)

Description

The instruction "SRT_DINT" is used to start a time-delay interrupt, which calls a time-delay interrupt OB after the expiry of the delay time specified at the parameter DTIME. The time delay interrupt OB is executed once for the time delay interrupt.

If the time-delay interrupt has not been executed yet and the instruction "SRT_DINT" is called again, the existing time-delay interrupt is deleted and a new time-delay interrupt is started.

Note

Delay time

If the delay time used is less than or equal to the currently used CPU cycle time and SRT_DINT is called cyclically, the time delay interrupt OB is executed with each CPU cycle.

Make sure that you select a delay time that is greater than the CPU cycle time.

With the SIGN parameter, you can enter an identifier that identifies the start of the time-delay interrupt. The values of the DTIME and SIGN parameters are shown in the start information of the organization blocked called.

Accuracy

The maximum time between the call of the "SRT_DINT" instruction and the start of the time-delay interrupt OB is one millisecond longer than the configured delay time, provided that no interruption events delay the call.

Parameters

The following table shows the parameters of the "SRT_DINT" instruction:

Parameter	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_DELAY (INT)	I, Q, M, D, L or constant	Number of the OB to be executed after a delay time
DTIME	Input	TIME	I, Q, M, D, L or constant	Delay time (1 to 60000 ms) You can realize longer times, for example, by using a counter in a time-delay interrupt OB.
SIGN	Input	WORD	I, Q, M, D, L or constant	Identifier that appears when the time-delay interrupt OB is called in the start event information of the OB.
RET_VAL	Return	INT	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error
8090	Incorrect parameter OB_NR
8091	Incorrect parameter DTIME

Error code* (W#16#...)	Description
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for time-delay interrupt functions (Page 3834).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

CAN_DINT: Cancel time-delay interrupt (S7-1200, S7-1500)

Description

You use this instruction to cancel a started time-delay interrupt and, thus, also cancel the call of the time-delay interrupt OB that is executed after the configured delay time. You specify the number of the organization block whose call is to be canceled in the OB_NR parameter.

Parameters

The following table shows the parameters of the "CAN_DINT" instruction:

Parameter	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_DELAY (INT)	I, Q, M, D, L or constant	Number of the OB whose call will be canceled
RET_VAL	Return	INT	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error
8090	Incorrect parameter OB_NR
80A0	Time-delay interrupt has not started.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for time-delay interrupt functions (Page 3834).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

QRY_DINT: Query time-delay interrupt status (S7-1200, S7-1500)

Description

The instruction "QRY_DINT" is used to query the status of the time-delay interrupt.

Parameters

The following table shows the parameters of the "QRY_DINT" instruction:

Parameter	Declaration	Data type	Memory area	Description
OB_NR	Input	OB_DELAY (INT)	I, Q, M, D, L or constant	Number of the OB whose status is being queried.
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the actual parameter of RET_VAL contains an error code. The value "0" is displayed in the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Status of the time-delay interrupt, see following table.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

Bit	Value	Meaning
0	0	Irrelevant
1	0	Time-delay interrupt is enabled by the operating system.
	1	Time-delay interrupt is disabled.
2	0	Time-delay interrupt is not activated or has elapsed.
	1	Time-delay interrupt is activated.
3	-	-
4	0	Time-delay interrupt OB with the specified number does not exist.
	1	Time-delay interrupt OB with the specified number exists.
Other bits		Always "0"

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error occurred.
8090	Incorrect information in the OB_NR parameter
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for time-delay interrupt functions (Page 3834).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

Program example for time-delay interrupt functions (S7-1200, S7-1500)

Introduction

In the following example, you read the status of a time-delay interrupt OB (OB20), and set a new delay time and start or disable OB20.

Requirement

Create 10 tags and a structure in a global data block for storing the data.

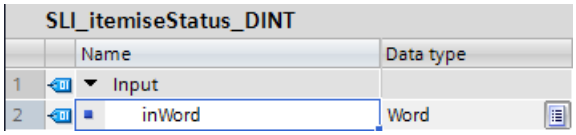
SLI_gDB_SRT_DINT

	Name	Data type	Start value
1	Static		
2	obNbr	OB_DELAY	20
3	executeQry	Bool	false
4	itemiseStatus	Struct	
5	Time-delay_interrupt_enabled	Bool	false
6	Time-delay_interrupt_activated	Bool	false
7	OB_with_obNbr_exists	Bool	false
8	returnValueQry	Int	0
9	executeSrt	Bool	false
10	delayTime	Time	T#6000ms
11	sign	Word	16#0001
12	returnValueSrt	Int	0
13	executeCan	Bool	false
14	returnValueCan	Int	0
15	testCurrCount	Int	0

Interconnect parameters: "SLI_itemiseStatus_DINT" FC

To evaluate the status information, create an FC "SLI_itemiseStatus_DINT".

Create the following local tag in it.



Create interconnections for all relevant bits.

Network 1:



Network 2:

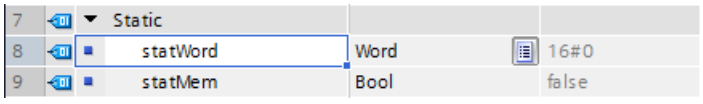


Network 3:

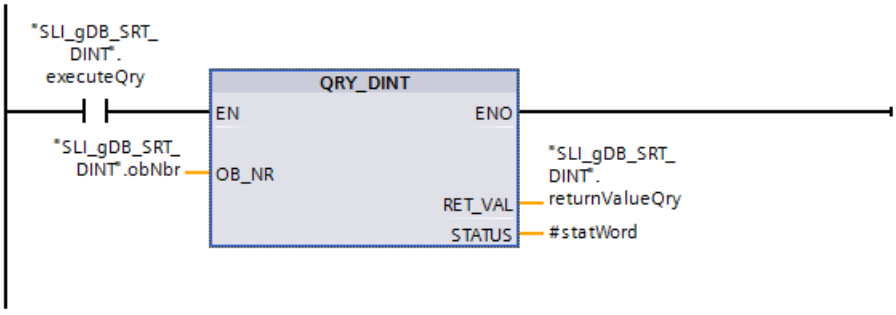


Interconnect parameters: "SLI_FB_SRT_DINT" FB

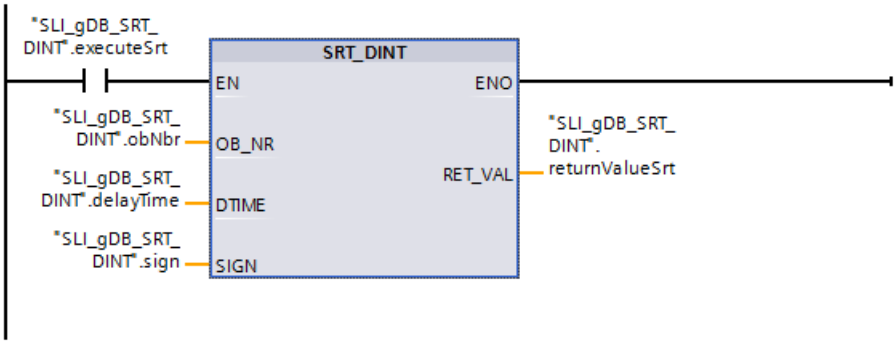
To call the instructions, create an FB "SLI_FB_SRT_DINT". Create the following local tags in the function block "SLI_FB_SRT_DINT".



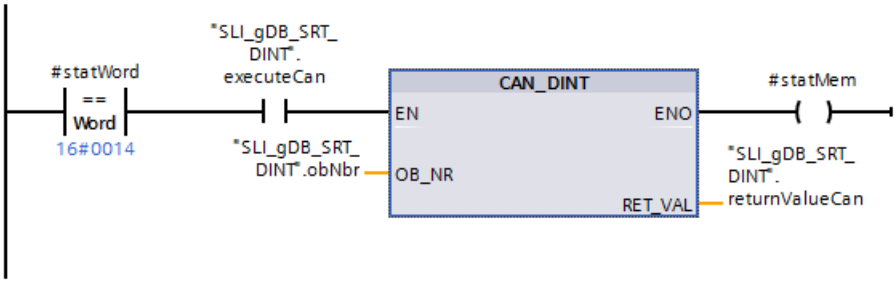
Network 1: Interconnect the parameters of the "QRY_DINT" instruction as follows.



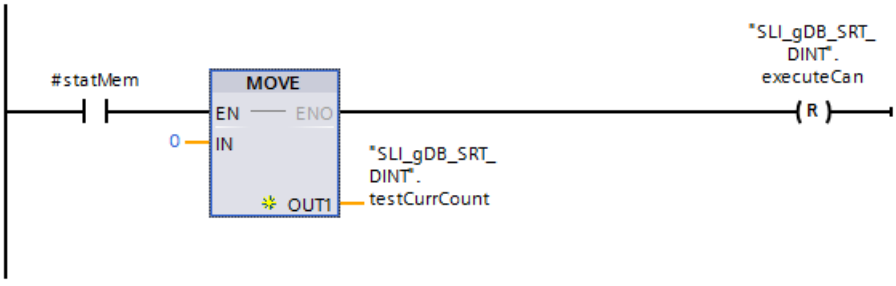
Network 2: Interconnect the parameters of the "SRT_DINT" instruction as follows.



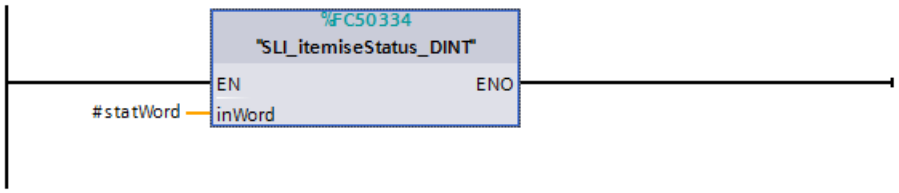
Network 3: Interconnect the parameters of the "CAN_DINT" instruction as follows.



Network 4: Interconnect the parameters that are to be reset with the execution of "CAN_DINT" as follows.



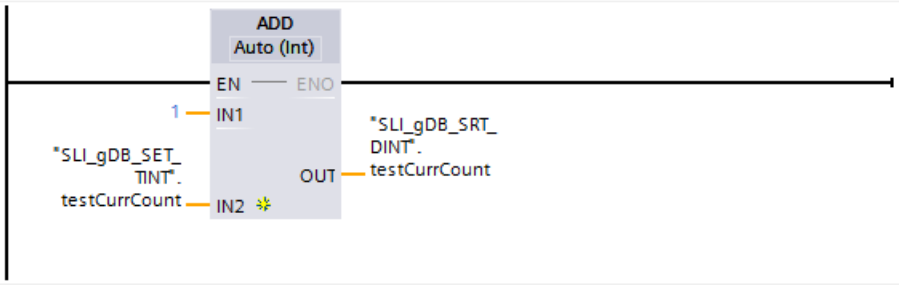
Network 5: Interconnect the parameter of the "SLI_itemiseStatus_DINT" FC as follows.



You call the FB "SLI_FB_SRT_DINT" in a cyclic OB (OB1).

Interconnect parameters: OB20

Create the following interconnections in OB20.

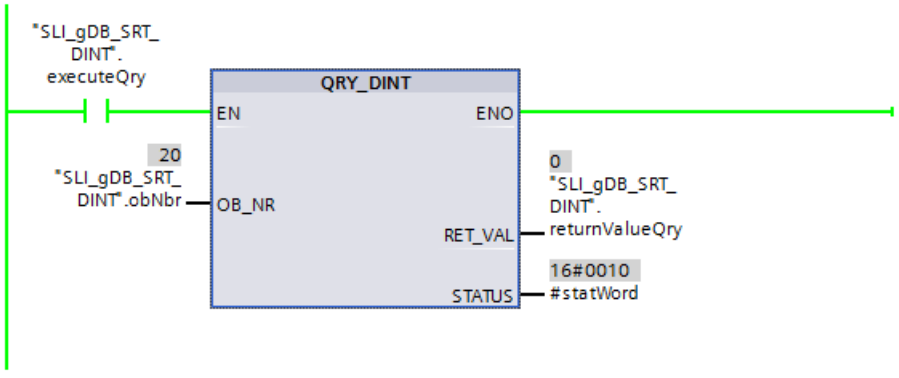


Result of "QRY_DINT"

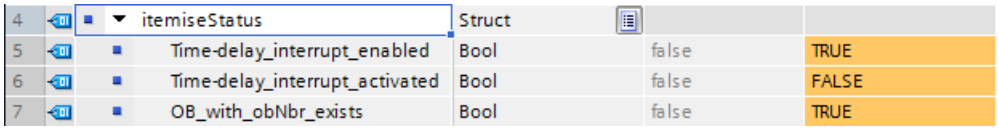
The instructions receive the number of the OB to be called (OB20) with the input parameter OB_NR ("obNbr").

Network 1:

When the normally open contact ("executeQry") supplies the signal state "TRUE", the "QRY_DINT" instruction is executed and the data of OB20 is read. The output parameter STATUS ("#statWord") displays the status of OB20. The output parameter RET_VAL ("returnValueQry") indicates that processing is taking place without errors.



The value of the tag "#statWord" is mapped in the structure "itemiseStatus".



4	itemiseStatus	Struct		
5	Time-delay_interrupt_enabled	Bool	false	TRUE
6	Time-delay_interrupt_activated	Bool	false	FALSE
7	OB_with_obNbr_exists	Bool	false	TRUE

Result of "SRT_DINT"

Network 2: When the normally open contact ("executeSrt") supplies the signal state "TRUE", the "SRT_DINT" instruction is executed. The input parameter DTIME ("delayTime") transfers the delay time (6 s) to OB20 and OB20 is enabled.

Note

Its delay time starts running at the time OB20 is enabled. OB20 is then called unless it has received the delay time again beforehand in a new CPU cycle.

The selected identifier (value "16#0001") to be output with the execution of OB20 is stored at the input parameter SIGN ("sign"). The identifier is shown in the start event information of OB20. The output parameter RET_VAL ("returnValueSrt") indicates that processing took place without errors.

If the NO contact ("executeSrt") returns the signal state "FALSE" after successful processing of SRT_DINT, the delay time is not transferred again to the OB20. As soon as the delay time has expired, OB20 is executed once.

Note

The described reaction of the OB20 in combination with the EN parameter only applies when the currently used CPU cycle time is less than the delay time of the OB20.

Result of the OB20

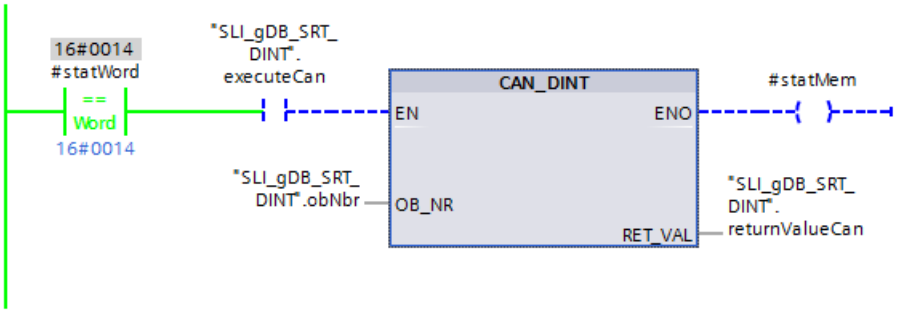
With each call of OB20, the instruction "ADD" is executed and the tag "testCurrCount" is incremented by "1". OB20 is called only once after expiration of the delay time defined with SRT_DINT. After it has been called, OB20 is once again "disabled".

SLI_gDB_SRT_DINT

	Name	Data type	Start value	Monitor value
1	Static			
2	obNbr	OB_DELAY	20	20
3	executeQry	Bool	false	TRUE
4	itemiseStatus	Struct		
5	Time-delay_interrupt_enabled	Bool	false	TRUE
6	Time-delay_interrupt_activated	Bool	false	FALSE
7	OB_with_obNbr_exists	Bool	false	TRUE
8	returnValueQry	Int	0	0
9	executeSrt	Bool	false	FALSE
10	delayTime	Time	T#6000ms	T#6S
11	sign	Word	16#0001	16#0001
12	returnValueSrt	Int	0	0
13	executeCan	Bool	false	FALSE
14	returnValueCan	Int	0	0
15	testCurrCount	Int	0	1

Result of "CAN_DINT"

Network 3: As long as OB20 is set and enabled ("#statWord" has the value "16#0014"), the comparator returns the signal state "TRUE" and CAN_DINT can be executed.



When the normally open contact ("executeCan") supplies the signal state "TRUE", the "CAN_DINT" instruction is executed. The instruction disables OB20 and deletes the set call data of OB20.
The output parameter RET_VAL ("returnValueCan") indicates that processing took place without errors.

The execution of CAN_DINT sets the coil ("#statMem") to "TRUE" and network 4 is executed.

Result of the reset

Network 4: When the normally open contact ("#statMem") supplies the signal state "TRUE", the tag "testCurrCount" is reset to the value "0" with the instruction "MOVE". Reset ("executeCan") in the signal state "TRUE" also automatically ends the execution of the instruction "CAN_DINT".

Result of the status information "#statWord"

The FC "SLI_itemiseStatus_DINT" is called in network 7. The significant bits of the tag "#statWord" are evaluated and output as Boolean tags with plain-text name. When a tag of the structure "itemiseStatus" supplies the signal state "TRUE", its statement is correct.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Synchronous errors (S7-1500)

Mask synchronous error events (S7-1500)

Introduction

Synchronous errors are programming and access errors. Such errors occur as a result of programming with incorrect operand areas or numbers, or incorrect addresses. **Masking** these synchronous errors means the following:

- Masked synchronous errors do not trigger an error OB call and do not lead to a programmed alternative reaction.
- The CPU "records" the masked errors that have occurred in an error status register.

Masking is carried out by calling the "MSK_FLT (Page 3846)" instruction.

Unmasking errors means canceling a previously set mask and clearing the corresponding bit in the event status register of the current priority class. Masking is canceled as follows:

- By calling the "DMSK_FLT (Page 3847)" instruction.
- Once the current priority class has been completed.

If an error event occurs after it has been unmasked, the operating system will start the associated error OB.

You can use the "READ_ERR (Page 3848)" instruction to read the masked errors that have occurred.

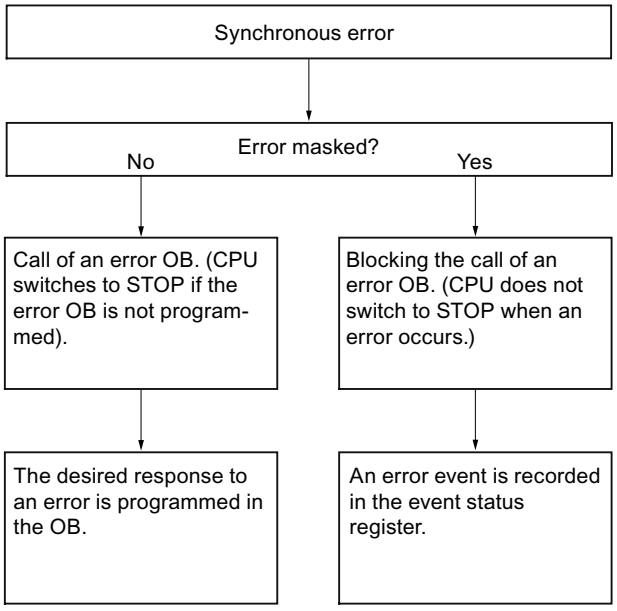
Note

Regardless of masking or unmasking of the error events for the S7-1500, the error event is entered in the diagnostic buffer and the group error LED of the CPU lights up.

Handling errors in general

Different actions are possible if programming and I/O area access errors occur in a user program:

- You can program an error OB that is called by the operating system when the corresponding error occurs.
- You can disable the error OB call individually for each priority class. If an error of this type occurs in the priority class, the CPU does not change to STOP in this case. The CPU enters the error in an error register. From this entry, however, you do not recognize when or how often this error has occurred.

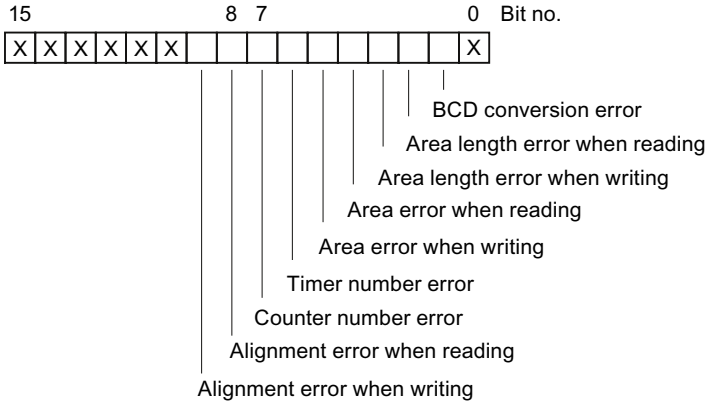


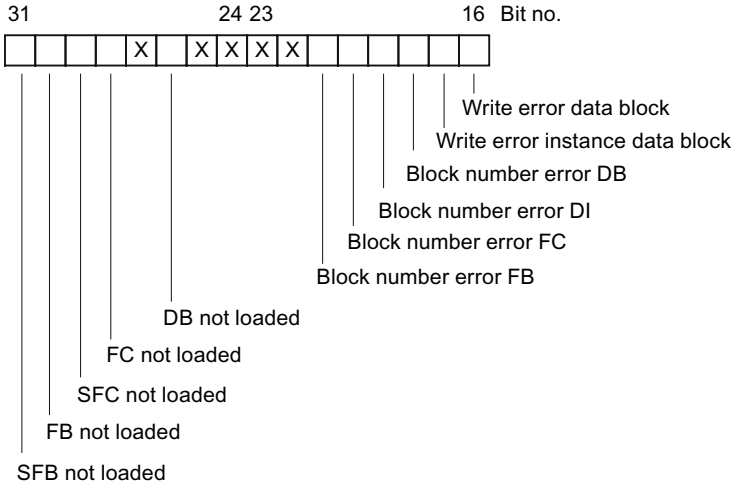
Synchronous errors are assigned to a particular bit pattern known as the **error filter**. You can find this error filter in the input and output parameters of the "MSK_FLT (Page 3846)", "DMSK_FLT (Page 3847)", and "READ_ERR (Page 3848)" instructions.

Synchronous errors are divided into **programming and access errors** which you can mask in two error masks. The error filters are illustrated in the following figures.

Programming error filter

The following figure shows the bit pattern of the error filter for programming errors. The error filter for the programming errors is available in the "PRGFLT_..." parameters (see below "Programming errors, Low-Word" or "Programming errors, High-Word").



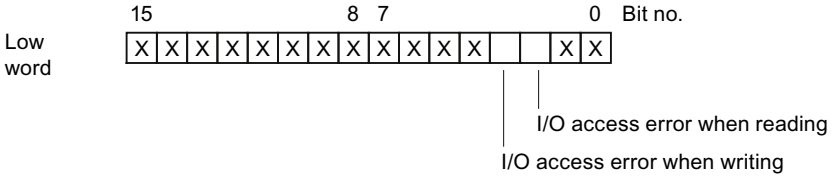


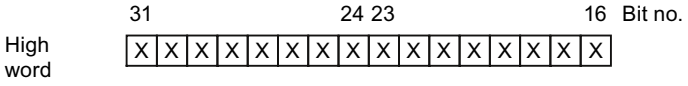
Legend: X Not relevant

The non-relevant bits "x" are set to "0" for the input and output parameters of the instructions "MSK_FLT (Page 3846)", "DMSK_FLT (Page 3847)", "READ_ERR (Page 3848)".

Access error mask

The following figure shows the bit pattern of the error mask for access errors. The error mask for access errors is located in the parameters ACCFLT_...

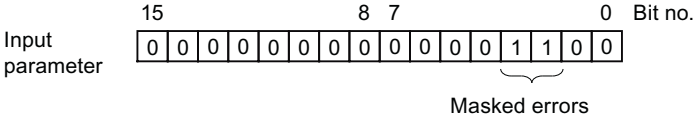


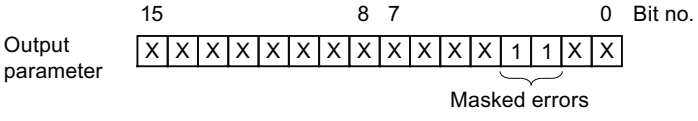


Legend: X Not relevant

Example: The following figure shows how the low word of the error mask for access errors appears with all masked errors.

- As input parameter for "MSK_FLT (Page 3846)"
- As output parameter for "MSK_FLT (Page 3846)"





Legend:
- X Not relevant
- 0 Not masked
- 1 Masked

Programming error low word

The following table lists the errors assigned to the low word of the error filter for programming errors. The possible causes of the errors are also listed.

Error	Event ID (W#16#...)	Possible cause of error
BCD conversion error	2521	The value to be converted is not a BCD number (e.g. 5E8).
Area length error when reading	2522	An addressed operand is not located entirely within the possible operand range. Example: MW 320 should be read although the memory area is only 256 bytes long.
Area length error when writing	2523	An addressed operand is not located entirely within the possible operand range. Example: A value should be written to MW 320 although the memory area is only 256 bytes long.
Area error when reading	2524	An incorrect range identifier for the operand was specified during indirect, overlapping range addressing. Example: • Correct: LAR1 P#E 12.0 L W[AR1, P#0.0] • Incorrect: LAR1 P#12.0 L W[AR1, P#0.0] The area error is reported during this operation.
Area error when writing	2525	An incorrect range identifier for the operand was specified during indirect, overlapping range addressing. Example: • Correct: LAR1 P#E 12.0 T W[AR1, P#0.0] • Incorrect: LAR1 P#12.0 T W[AR1, P#0.0] The area error is reported during this operation.
Timer number error (not implemented with the S7-1500)	2526	A non-existent timer is accessed. Example: • SI T [MW 0] with MW 0 = 129; the timer 129 should be started although there are only 128 timers available.
Counter number error (not implemented with the S7-1500)	2527	A non-existent counter is accessed. • Example: ZV Z [MW 0] with MW 0 = 600; the counter 600 is accessed although only 512 counters are available.

Error	Event ID (W#16#...)	Possible cause of error
Alignment error when reading	2528	A byte, word or double word operand with a bit address ≠ 0 is addressed. Example: • Correct: LAR1 P#M12.0 L B[AR1, P#0.0] • Incorrect: LAR1 P#M12.4 L B[AR1, P#0.0]
Alignment error when writing	2529	A byte, word or double word operand with a bit address ≠ 0 is addressed. Example: • Correct: LAR1 P#M12.0 T B[AR1, P#0.0] • Incorrect: LAR1 P#M12.4 T B[AR1, P#0.0]

Programming error high word

The following table lists the errors assigned to the high word of the error filter for programming errors. The possible causes of the errors are also listed.

Error	Event ID (W#16#...)	Possible cause of error
Write error data block	2530	The data block to be written to is write-protected.
Write error instance data block	2531	The instance data block to be written to is write-protected.
Block number error DB	2532	The number of the data block is greater than the highest permissible number.
Block number error DI	2533	The number of the instance data block is greater than the highest permissible number.
Block number error FC	2534	The number of a called function (FC) is greater than the highest permissible number.
Block number error FB	2535	The number of a called function block (FB) is greater than the highest permissible number.
DB not loaded	253A	The data block is not loaded.
Instruction not loaded	253C to 253F	The instruction to be called is not loaded.

Access error

The following table contains the errors that are assigned to the error mask for access errors. The possible causes of the errors are also listed.

Error	Event ID (W#16#...)	Possible cause of error
I/O access error when reading	2942	• The address in the I/O area has no assigned signal module. • Access to this I/O area was not acknowledged within the specified module monitoring time (timeout).
I/O access error when writing	2943	

MSK_FLT: Mask synchronous error events (S7-1500)

Description

You use the instruction to control the reaction of the CPU to synchronous errors. This is done by masking the respective synchronous errors (for error filters, see Mask synchronous error events (Page 3840)). When you call "MSK_FLT", you mask the synchronous errors in the current priority class.

If you set individual bits of the synchronous error filter to "1" in the input parameters, other bits that were set previously retain their value of "1". You obtain new error filters that you read out using the output parameters. The synchronous errors you have masked do not call an OB but are simply entered in an error register. You read the event status register with the instruction "READ_ERR (Page 3848)".

Parameters

The following table shows the parameters of the instruction "MSK_FLT":

Parameter	Declaration	Data type	Memory area	Description
PRGFLT_SET_MASK	Input	DWORD	I, Q, M, D, L or constant	Programming error to be masked
ACCFLT_SET_MASK	Input	DWORD	I, Q, M, D, L or constant	Access error to be masked
RET_VAL	Return	INT	I, Q, M, D, L	Error information
PRGFLT_MASKED	Output	DWORD	I, Q, M, D, L	Masked programming errors
ACCFLT_MASKED	Output	DWORD	I, Q, M, D, L	Masked access errors

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	None of the errors were already masked.
0001	At least one of the errors was already masked. Nevertheless, the other errors will be masked.
-	General error information See also: Getting error ID locally with GetErrorID (Page 3409)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for synchronous error events (Page 3849).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

DMSK_FLT: Unmask synchronous error events (S7-1500)

Description

You use the instruction to unmask the errors masked with "MSK_FLT (Page 3846)". To do this, you must set the corresponding bits of the error filter to "1" in the input parameters. With the "DMSK_FLT" call, you unmask the corresponding synchronous errors of the current priority class. At the same time, the queried entries are cleared in the error register. You read out the new error filters using the output parameters.

Parameters

The following table shows the parameters of the instruction "DMSK_FLT":

Parameter	Declaration	Data type	Memory area	Description
PRGFLT_RESET_MASK	Input	DWORD	I, Q, M, D, L or constant	Programming errors to be unmasked
ACCFLT_RESET_MASK	Input	DWORD	I, Q, M, D, L or constant	Access errors to be unmasked
RET_VAL	Return	INT	I, Q, M, D, L	Error information
PRGFLT_MASKED	Output	DWORD	I, Q, M, D, L	Still masked programming errors
ACCFLT_MASKED	Output	DWORD	I, Q, M, D, L	Still masked access errors

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	All specified errors were unmasked.
0001	At least one of the errors was not masked. Nevertheless, the other errors will be unmasked.
-	General error information See also: GET_ERR_ID: Get error ID locally (Page 3409)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for synchronous error events (Page 3849).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

READ_ERR: Read out event status register (S7-1500)

Description

You use the instruction to read the error register. The structure of the error register corresponds to that of the programming and access error masks, which you can program as input parameters with "MSK_FLT (Page 3846)" and "DMSK_FLT (Page 3847)".

In the input parameters, you enter the synchronous errors you want to read from the error register. When you call "READ_ERR", you read the required entries from the error register and at the same time clear these entries.

The error register contains information that tells you which of the masked synchronous errors in the current priority class occurred at least once. If a bit is set, this means that the corresponding masked synchronous error occurred at least once.

Parameters

The following table shows the parameters of the instruction "READ_ERR":

Parameter	Declaration	Data type	Memory area	Description
PRGFLT_QUERY	Input	DWORD	I, Q, M, D, L or constant	Query programming error
ACCFLT_QUERY	Input	DWORD	I, Q, M, D, L or constant	Query access error
RET_VAL	Return	INT	I, Q, M, D, L	Error information
PRGFLT_CLR	Output	DWORD	I, Q, M, D, L	Programming errors that occurred
ACCFLT_CLR	Output	DWORD	I, Q, M, D, L	Access errors that occurred

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	All queried errors are masked.
0001	At least one of the queried errors is not masked.
-	General error information See also: Getting error ID locally with GetErrorID (Page 3409)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for synchronous error events (Page 3849).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

Program example for synchronous error events (S7-1500)

Introduction

In the example below, mask the synchronous error events (program errors) with the instruction "MSK_FLT", unmask a part of the synchronous error events with the instruction "DMSK_FLT" and read out the event status register with the instruction "READ_ERR". When the program error is unmasked, call a program error OB.

Storage of data

Create the following structures and tags for storing the data in a global data block.

- Tags for MSK_FLT:

SLI_gDB_MSK_FLT			
	Name	Data type	Start value
1	Static		
2	maskedProgErrs	DWord	16#0
3	maskedAccessErrs	DWord	16#0
4	MSK_FLT	Struct	
5	execute	Bool	false
6	progErrMask_Details	Struct	
7	accessErrMask_Details	Struct	
8	progErrSetMask	DWord	16#0
9	accessErrSetMask	DWord	16#0
10	returnValue	Int	0

- MSK_FLT / content of the structure "progErrMask_Details":

6	progErrMask_Details	Struct	
7	BCD_conversion_error	Bool	false
8	Area_length_error_when_reading	Bool	1
9	Area_length_error_when_writing	Bool	1
10	Range_error_when_reading	Bool	1
11	Range_error_when_writing	Bool	1

You can find the complete organization of the structure here: Mask synchronous error events (Page 3840).

- MSK_FLT / content of the structure "accessErrMask_Details":

7	accessErrMask_Details	Struct	
8	IO_access_error_when_reading	Bool	false
9	IO_access_error_when_writing	Bool	false

- Tags for DMSK_FLT:

11	DMSK_FLT	Struct	
12	execute	Bool	false
13	progErrResetMask	DWord	16#0000_000C
14	accessErrResetMask	DWord	16#0
15	returnValue	Int	0

- Tags for READ_ERR:

16	READ_ERR	Struct	
17	busyReading	Bool	false
18	peOB_unmasked_active	Bool	false
19	returnValue	Int	0
20	occuredMskProgErrs	DWord	16#0
21	occuredMskAccessErrs	DWord	16#0
22	occProgErrMask_Details	Struct	
23	occAccErrMask_Details	Struct	

The organization of the structures "occProgErrMask_Details" and "occAccErrMask_Details" is similar to the organization of the structures "progErrMask_Details" and "accessErrMask_Details".

- Tags for the test result:

24	testErrorEvent	Struct	
25	index	Int	0
26	field	Array[0..10] of Real	
27	result	Real	0.0
28	startOut	Bool	false

Interconnect parameters: "SLI_FB_genErr_SyncErrEv" FB

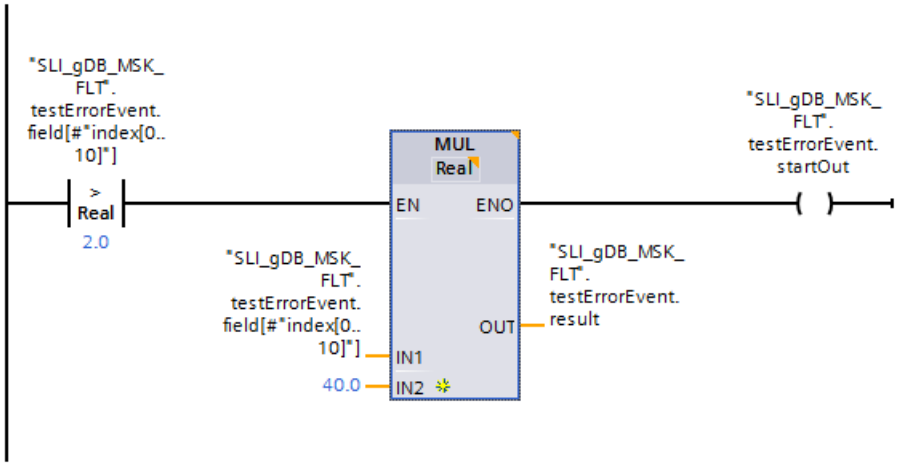
Create an "SLI_FB_genErr_SyncErrEv" FB to generate a program error.

Create the following local tag in it.

SLI_FB_genErr_SyncErrEv

	Name	Data type	Default value
1	Input		
2	index[0..10]	Int	0

Create the following interconnections in the FB.



Interconnect parameters: "SLI_itemiseDW_SyncErrEv" FC

To transfer a bit sequence in the size of a DWORD, create the FC "SLI_itemiseDW_SyncErrEv".

Create the following local tags (inputs and outputs from 1 to 31) in it.

- Excerpt of the inputs:

SLI_itemiseDW_SyncErrEv		
	Name	Data type
1	Input	
2	in1	Bool
3	in2	Bool
4	in3	Bool
5	in4	Bool
6	in5	Bool
7	in6	Bool
8	in7	Bool
9	in8	Bool

- Excerpt of the outputs:

21	in31	Bool
22	Output	
23	out1	Bool
24	out2	Bool
25	out3	Bool
26	out4	Bool
27	out5	Bool
28	out6	Bool
29	out7	Bool
30	out8	Bool
39	out31	Bool

Create the following interconnections for the inputs and outputs in the FC based on the example of network 1.

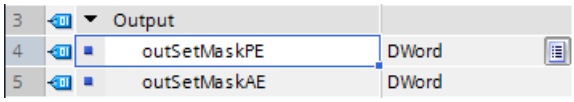
Network 1:



Interconnect parameters: "SLI_mask_SyncErrEv" FC

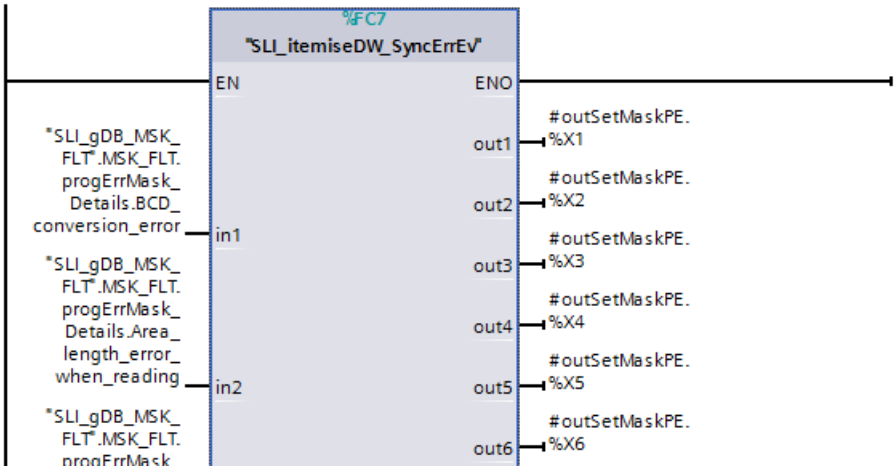
To convert the masks to be set for program and access errors, create the FC "SLI_mask_SyncErrEv".

Create the following local tags in it.



Network 1:

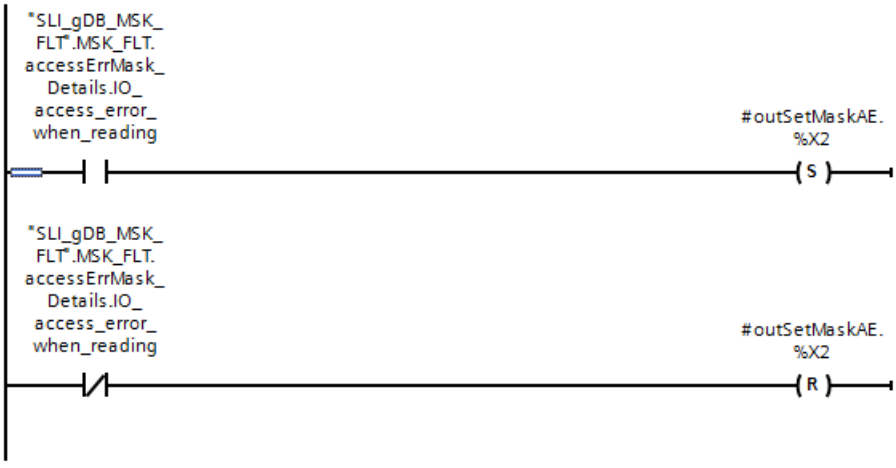
Interconnect the parameters of the "SLI_itemiseDW_SyncErrEv" FC according to the following example.



You can find the complete organization of the structure of the program errors ("progErrMask_Details") here: Mask synchronous error events (Page 3840).

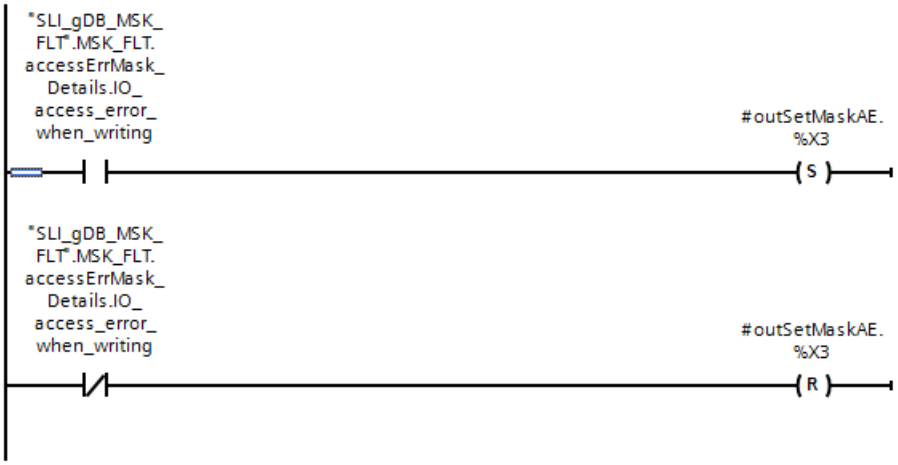
Network 2:

Interconnect bit 2 of the DWORD of the access errors as follows.



Network 3:

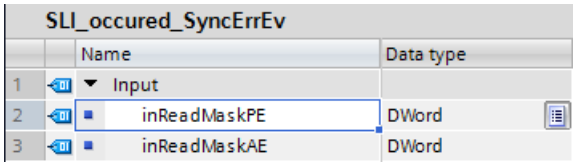
Interconnect bit 3 of the DWORD of the access errors as follows.



Interconnect parameters: "SLI_occured_SyncErrEv" FC

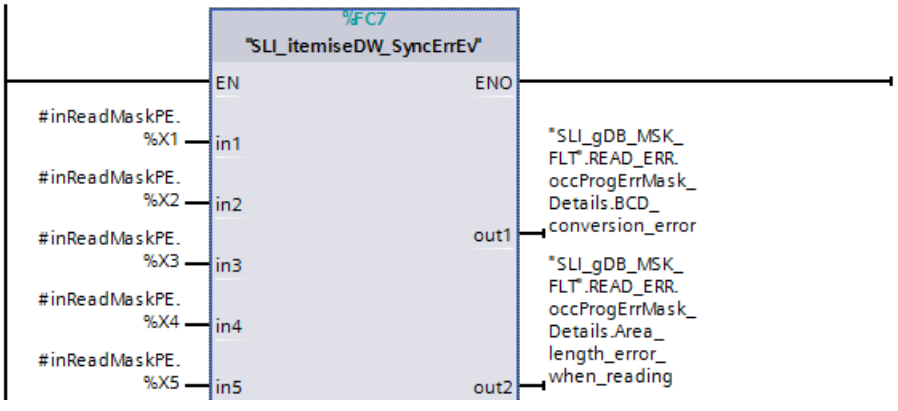
To convert read program and access errors, create the FC "SLI_occured_SyncErrEv".

Create the following local tags in it.



Network 1:

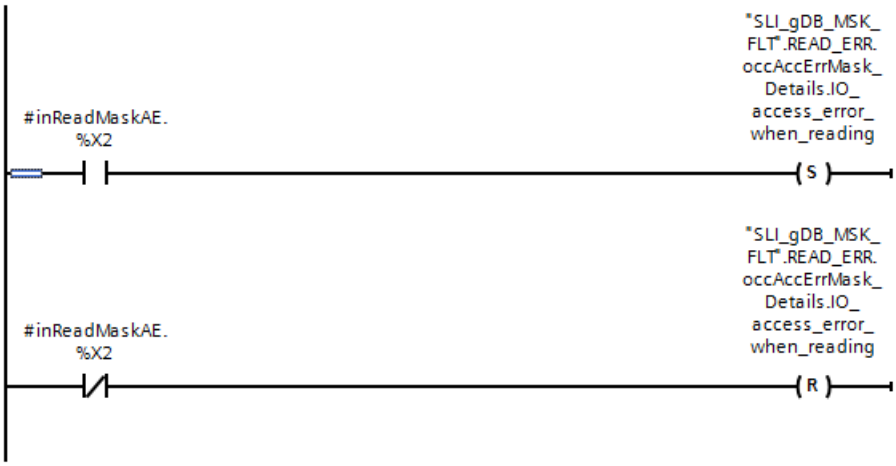
Interconnect the parameters of the "SLI_itemiseDW_SyncErrEv" FC according to the following example.



You can find the complete organization of the structure of the program errors ("occProgErrMask_Details") here: Mask synchronous error events (Page 3840).

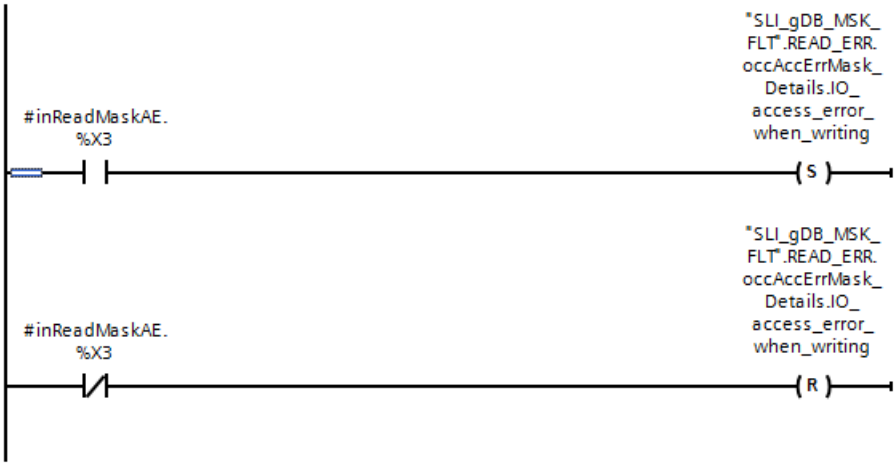
Network 2:

Interconnect bit 2 of the DWORD of the access errors as follows.



Network 3:

Interconnect bit 3 of the DWORD of the access errors as follows.



Interconnect parameters: "SLI_peOB_SyncErrEV" OB

Create the OB "SLI_peOB_SyncErrEV" in case of a program error in the unmasked state.

Network 1:

Create the following interconnection in the OB.

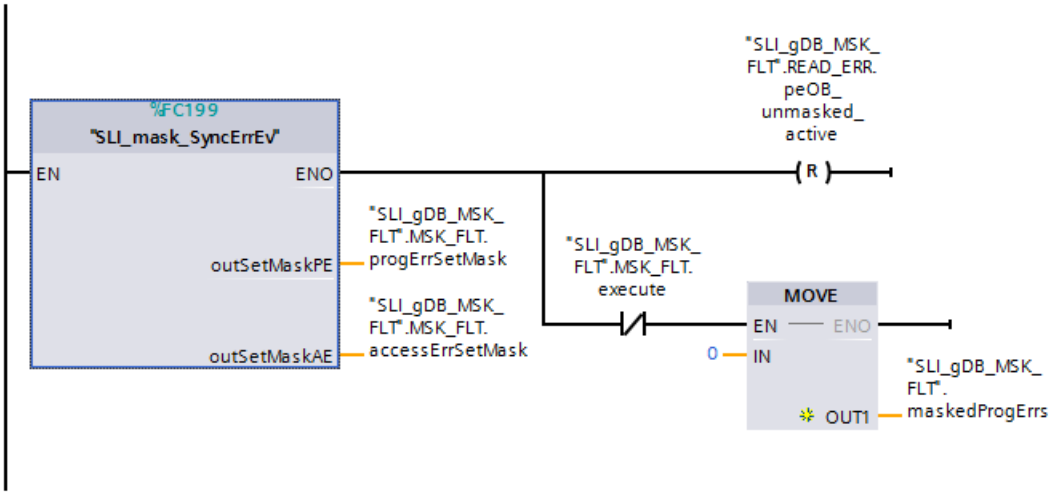


Interconnect parameters: "SLI_FC_MSK_FLT" FC

To call the program parts, create the FC "SLI_FC_MSK_FLT".

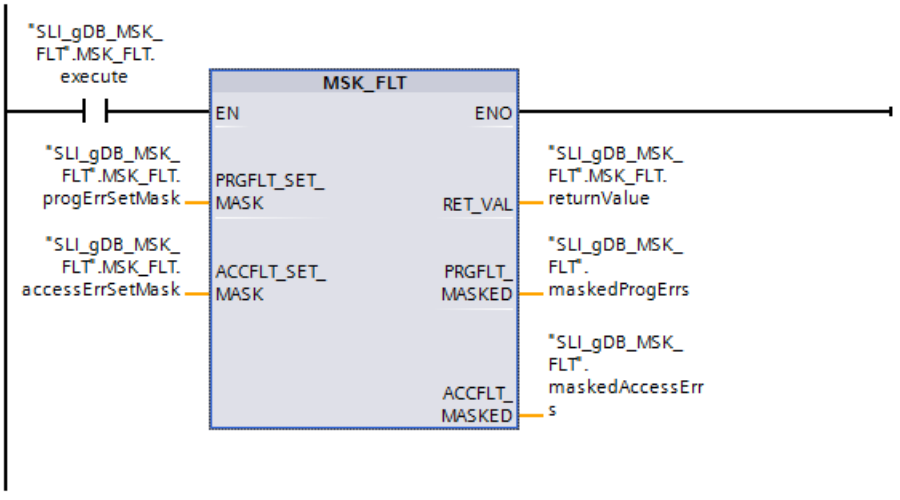
Network 1:

Interconnect the parameters of the FC "SLI_mask_SyncErrEv" as well as the tags "peOB_unmased_active", "MSK_FLT.execute" and "maskedProgErrs" as follows.



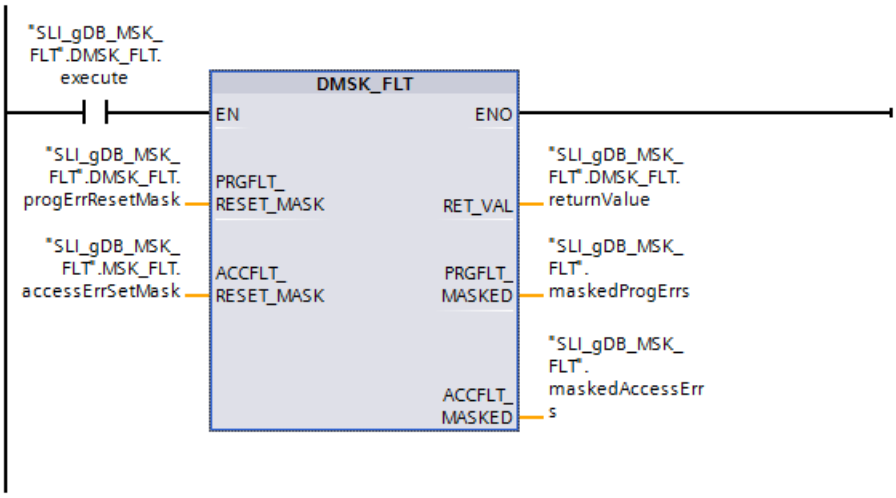
Network 2:

Interconnect the parameters of the "MSK_FLT" instruction as follows.



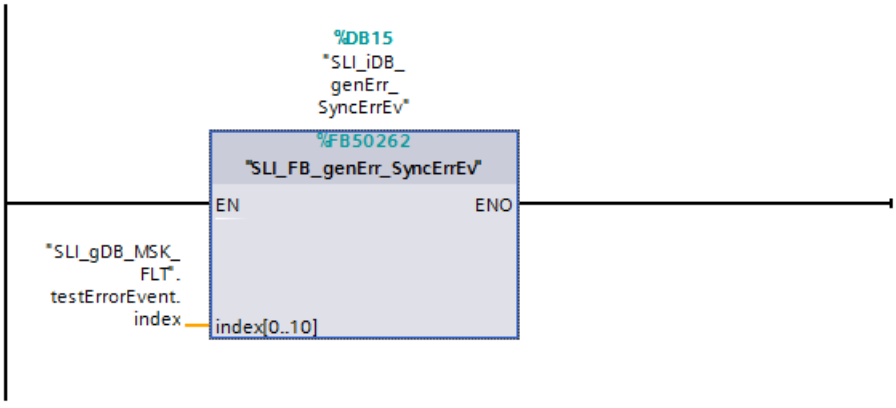
Network 3:

Interconnect the parameters of the "DMSK_FLT" instruction as follows.



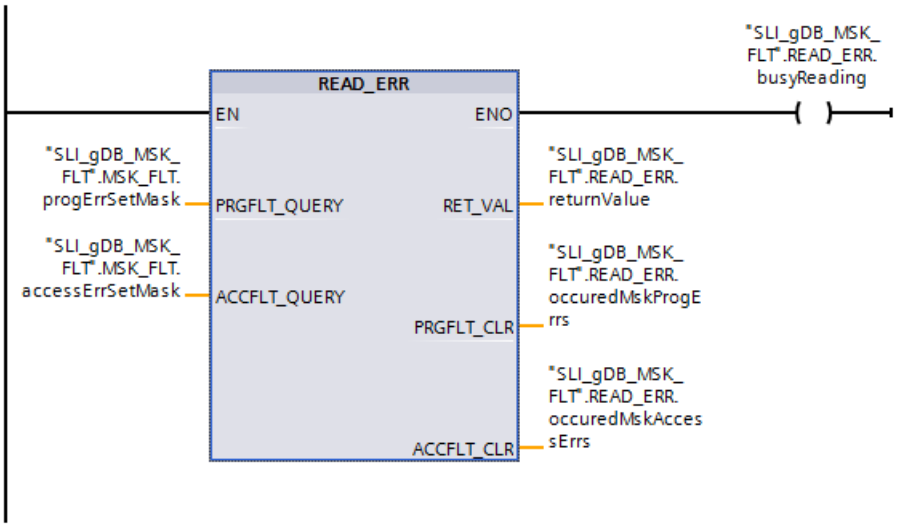
Network 4:

Interconnect the parameters of the "SLI_FB_genErr_SyncErrEv" FC as follows.



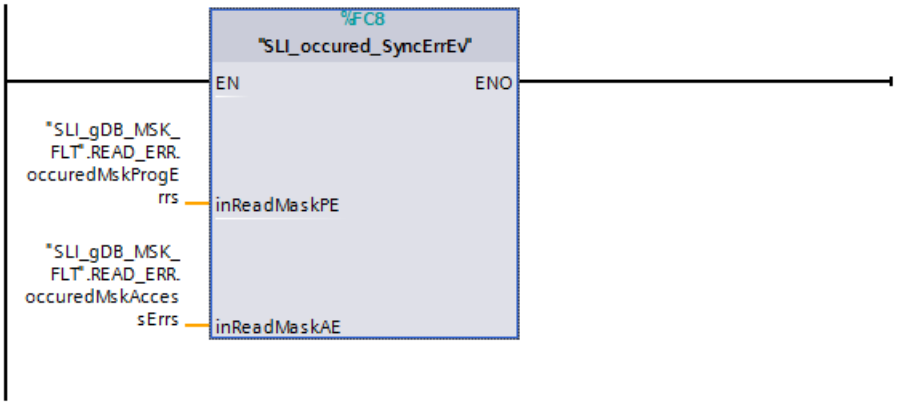
Network 5:

Interconnect the parameters of the "READ_ERR" instruction as follows.



Network 6:

Interconnect the parameters of the "SLI_occured_SyncErrEv" FC as follows.



Result of OB "SLI_peOB_SyncErrEv"

When a program error was triggered and is not masked by a filter, the program error OB "SLI_peOB_SyncErrEv" is executed.

The tag "READ_ERR.peOB_unmasked_active" is then set to "TRUE" in network 1 of the OB.

Result of FC "SLI_FC_MSK_FLT"

Network 1 (FC "SLI_mask_SyncErrEv"):

By calling the FC "SLI_mask_SyncErrEv", the structures for program errors and access errors are converted into the data type DWORD ("progErrSetMask" and "accessErrSetMask").

Reset the value because the value "TRUE" of the tag "READ_ERR.peOB_unmasked_active" is still displayed even after calling the program error OB. If the instruction "MSK_FLT" is not executed successfully ("MSK_FLT.execute" has the value "FALSE"), the tag "maskedProgErrs" is reset.

Network 2 ("MSK_FLT"):

When the normally open contact ("MSK_FLT.execute") supplies the signal state "TRUE", the "MSK_FLT" instruction is executed.

The synchronous error events that have to be controlled are specified according to the tags "progErrMask_Details" and "accessErrMask_Details":

- You specify the program errors you want to mask with "progErrMask_Details".
- No access errors ("accessErrMask_Details") were masked in the example.

The synchronous error events that have to be controlled are stored in the data type DWORD with the input parameters PRGFLT_SET_MASK ("MSK_FLT.progErrSetMask") and ACCFLT_SET_MASK ("MSK_FLT.accessErrSetMask").

The output parameter RET_VAL ("MSK_FLT.returnValue") indicates that none of the errors was already masked. The currently masked synchronous error events are displayed at the output parameters "PRGFLT_MASKED" ("maskedProgErrs") and ACCFLT_MASKED ("maskedAccessErrs").

2		maskedProgErrs	DWord	16#0	16#0000_003C
3		maskedAccessErrs	DWord	16#0	16#0000_0000
4		MSK_FLT	Struct		
5		execute	Bool	false	TRUE
6		progErrMask_Details	Struct		
7		accessErrMask_Details	Struct		
8		progErrSetMask	DWord	16#0	16#0000_003C
9		accessErrSetMask	DWord	16#0	16#0000_0000
10		returnValue	Int	0	0

Network 3 ("DMSK_FLT"):

When the normally open contact ("DMSK_FLT.execute") supplies the signal state "TRUE", the "DMSK_FLT" instruction is executed. The synchronous error events that have to be unmasked are stored in the data type DWORD with the input parameters PRGFLT_RESET_MASK ("DMSK_FLT.progErrResetMask") and ACCFLT_RESET_MASK ("DMSK_FLT.accessErrResetMask").

Output parameter RET_VAL ("DMSK_FLT.returnValue") indicates that all specified errors were unmasked.

11		DMSK_FLT	Struct		
12		execute	Bool	false	TRUE
13		progErrResetMask	DWord	16#0000_000C	16#0000_000C
14		accessErrResetMask	DWord	16#0	16#0000_0000
15		returnValue	Int	0	0

The currently masked synchronous error events are displayed at the output parameters "PRGFLT_MASKED" ("maskedProgErrs") and ACCFLT_MASKED ("maskedAccessErrs").

2		maskedProgErrs	DWord	16#0	16#0000_0030
3		maskedAccessErrs	DWord	16#0	16#0000_0000

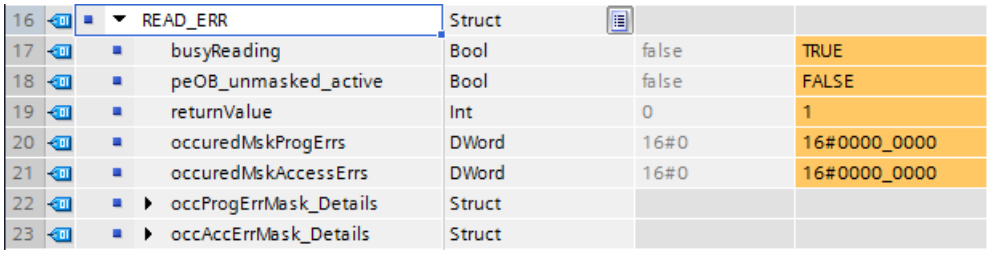
Network 4 (FB "SLI_FB_genErr_SyncErrEv"):

When the tag "testErrorEvent.index" returns a value greater than "10", a program error is triggered.

Network 5 ("READ_ERR"):

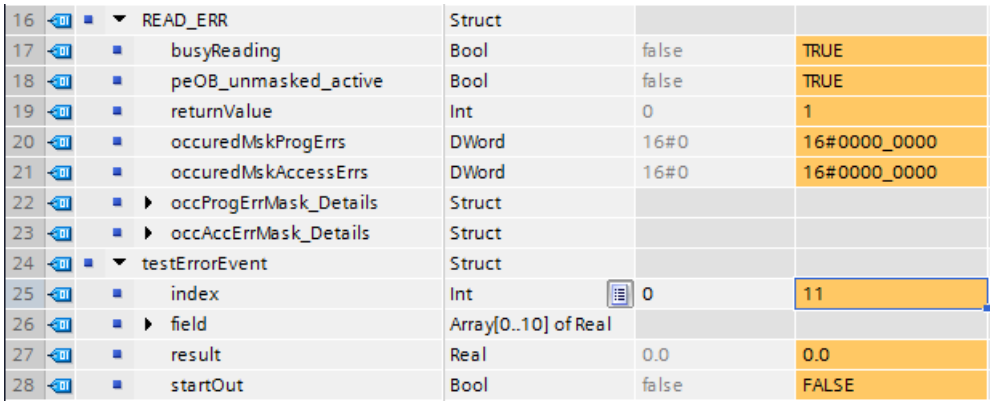
The instruction "READ_ERR" reads the event status register. The synchronous error events that are to be read are stored in the data type DWORD with the input parameters PRGFLT_QUERY ("READ_ERR.progErrSetMask") and ACCFLT_QUERY ("READ_ERR.accessErrSetMask").

- Case 1 - The program error was not triggered. No masks are set:
  The output parameter RET_VAL ("READ_ERR.returnValue") indicates that at least one of the queried errors is not masked.
  The currently occurring error events are displayed at the output parameters "PRGFLT_CLR" ("READ_ERR.occuredMskProgErrs") and ACCFLT_CLR ("READ_ERR.occuredMskAccessErrs").



16	READ_ERR	Struct		
17	busyReading	Bool	false	TRUE
18	peOB_unmasked_active	Bool	false	FALSE
19	returnValue	Int	0	1
20	occuredMskProgErrs	DWord	16#0	16#0000_0000
21	occuredMskAccessErrs	DWord	16#0	16#0000_0000
22	occProgErrMask_Details	Struct		
23	occAccErrMask_Details	Struct		

- Case 2 - The program error was triggered and is not masked:
  MSK_FLT, DMSK_FLT are inactive. The program error OB is executed.



16	READ_ERR	Struct		
17	busyReading	Bool	false	TRUE
18	peOB_unmasked_active	Bool	false	TRUE
19	returnValue	Int	0	1
20	occuredMskProgErrs	DWord	16#0	16#0000_0000
21	occuredMskAccessErrs	DWord	16#0	16#0000_0000
22	occProgErrMask_Details	Struct		
23	occAccErrMask_Details	Struct		
24	testErrorEvent	Struct		
25	index	Int	0	11
26	field	Array[0..10] of Real		
27	result	Real	0.0	0.0
28	startOut	Bool	false	FALSE

- Case 3 - The triggered program error is masked:
  MSK_FLT is active. DMSK_FLT and the program error OB are inactive.

16	READ_ERR	Struct		
17	busyReading	Bool	false	TRUE
18	peOB_unmasked_active	Bool	false	FALSE
19	returnValue	Int	0	0
20	occuredMskProgErrs	DWord	16#0	16#0000_0004
21	occuredMskAccessErrs	DWord	16#0	16#0000_0000
22	occProgErrMask_Details	Struct		
23	occAccErrMask_Details	Struct		
24	testErrorEvent	Struct		
25	index	Int	0	11
26	field	Array[0..10] of Real		
27	result	Real	0.0	0.0
28	startOut	Bool	false	FALSE

  The following program error occurred according to the structure "occProgErrMask_Details":

22	occProgErrMask_Details	Struct		
23	BCD_conversion_error	Bool	false	FALSE
24	Area_length_error_when_reading	Bool	false	TRUE
25	Area_length_error_when_writing	Bool	false	FALSE
26	Range_error_when_reading	Bool	false	FALSE
27	Range_error_when_writing	Bool	false	FALSE

- Case 4 - The triggered program error is unmasked:
  MSK_FLT, DMSK_FLT and the program error OB are active.

16	READ_ERR	Struct		
17	busyReading	Bool	false	TRUE
18	peOB_unmasked_active	Bool	false	TRUE
19	returnValue	Int	0	1
20	occuredMskProgErrs	DWord	16#0	16#0000_0000
21	occuredMskAccessErrs	DWord	16#0	16#0000_0000
22	occProgErrMask_Details	Struct		
23	occAccErrMask_Details	Struct		
24	testErrorEvent	Struct		
25	index	Int	0	11
26	field	Array[0..10] of Real		
27	result	Real	0.0	0.0
28	startOut	Bool	false	FALSE

Network 6 (FC "SLI_occured_SyncErrEv"):

By calling the FC "SLI_occured_SyncErrEv", the read program errors ("occuredMskProgErrs") and access errors ("occuredMskAccessErrs") are converted into the structures "occProgErrMask_Details" and "occAccErrMask_Details".

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

MSK_FLT: Mask synchronous error events (Page 3846)

DMSK_FLT: Unmask synchronous error events (Page 3847)

READ_ERR: Read out event status register (Page 3848)

Asynchronous error event (S7-1200, S7-1500)

DIS_IRT: Disable interrupt event (S7-1500)

Description

The instruction "DIS_IRT" is used to disable the processing of new interrupts and asynchronous error events. Disabling means that if an interrupting event occurs, the operating system of the CPU reacts as follows:

- It **neither** calls an interrupt OB nor an asynchronous error OB,
- **nor** does it trigger the normal reaction if an interrupt OB or asynchronous error OB is not programmed.

If you disable interrupts and asynchronous error events, this remains in effect for all priority classes. The disable is only canceled with the instruction "EN_IRT (Page 3863)" or by a warm or a cold restart.

Note

Remember that when you program the "DIS_IRT" instruction, all interrupts that occur will be discarded.

Parameters

The following table shows the parameters of the "DIS_IRT" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	BYTE	I, Q, M, D, L or constant	Specifies which interrupts and asynchro-nous errors are disabled.
OB_NR	Input	INT	I, Q, M, D, L or constant	OB number
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter MODE

MODE (B#16#...)	Meaning
00	All newly occurring interrupts and asynchronous error events are disabled (with the exception of the programming error and the error during direct access to I/O data and the Motion Control OBs). Assign the value "0" to the OB_NR parameter. Entries continue to be made in the diagnostics buffer.
01	All newly occurring events belonging to a specified interrupt class are disabled. Identify the interrupt class by specifying it as follows: • Time-of-day interrupts: 10 • Time-delay interrupts: 20 • Cyclic interrupts: 30 • Process interrupts: 40 • Interrupts for DPV1: 50 • Synchronous cycle interrupt: 60 • Redundancy error interrupts: 70 • Asynchronous error interrupts: 80 Entries continue to be made in the diagnostics buffer.
02	All new occurrences of a specified interrupt are disabled. You designate the interrupt using the OB number. Entries continue to be made in the diagnostics buffer.

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error occurred.
8090	The OB_NR input parameter contains an invalid value.
8091	The MODE input parameter contains an invalid value.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for DIS_IRT & EN_IRT (Page 3864).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

EN_IRT: Enable interrupt event (S7-1500)

Description

You use the instruction to enable the processing of new interrupts and asynchronous error events that you have previously disabled with the "DIS_IRT (Page 3861)" instruction. This means that if an interrupt event occurs, the CPU operating system reacts in one of the following ways:

- It calls an interrupt OB or asynchronous error OB.
  or
- it triggers the specified reaction if an interrupt OB or asynchronous error OB is not programmed.

Parameters

The following table shows the parameters of the "EN_IRT" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	BYTE	I, Q, M, D, L or constant	Specifies which interrupts and asynchronous error events will be enabled.
OB_NR	Input	INT	I, Q, M, D, L or constant	OB number
RET_VAL	Return	INT	I, Q, M, D, L	If an error occurs while the instruction is being executed, the return value contains an error code.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter MODE

MODE	Meaning
0	All newly occurring interrupts and asynchronous error events are enabled with the exception of the programming error and the error during direct access to I/O data and the Motion Control OBs.
1	All newly occurring events belonging to a specified interrupt class are enabled. Identify the interrupt class by specifying it as follows: • Time-of-day interrupts: 10 • Time-delay interrupts: 20 • Cyclic interrupts: 30 • Process interrupts: 40 • Interrupts for DPV1: 50 • Synchronous cycle interrupt: 60 • Redundancy error interrupts: 70 • Asynchronous error interrupts: 80
2	All newly occurring events of a specified interrupt are enabled. You designate the interrupt using the OB number.

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error occurred.
8090	The OB_NR input parameter contains an invalid value.
8091	The MODE input parameter contains an invalid value.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for DIS_IRT & EN_IRT (Page 3864).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

Program example for DIS_IRT & EN_IRT (S7-1200, S7-1500)

Introduction

In the following example you disable processing of a cyclic interrupt OB (OB33) with the help of the "DIS_IRT" instruction and enable OB33 once again with the instruction "EN_IRT".

Storage of data

Create the following structures and tags for storing the data in a global data block.

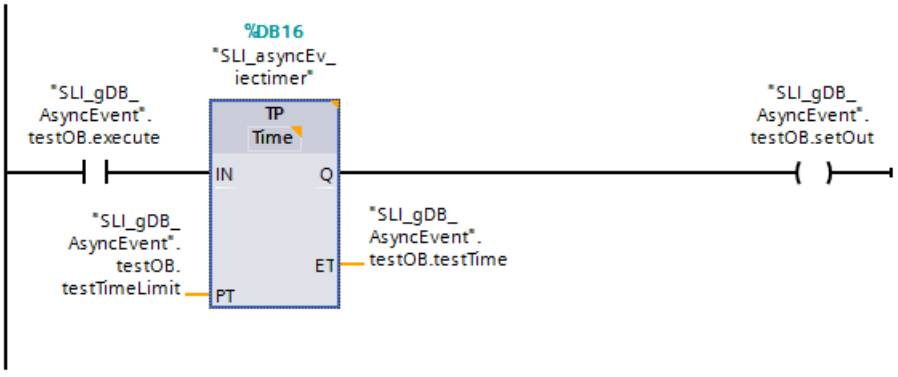
SLI_gDB_AsyncEvent

	Name	Data type	Start value
1	Static		
2	DIS_IRT	Struct	
3	execute	Bool	false
4	done	Bool	false
5	mode	Byte	16#02
6	obNbr	Int	33
7	returnValue	Int	0
8	EN_IRT	Struct	
9	execute	Bool	false
10	done	Bool	false
11	mode	Byte	16#02
12	obNbr	Int	33
13	returnValue	Int	0
14	testOB	Struct	
15	execute	Bool	false
16	testTime	Time	T#0ms
17	testTimeLimit	Time	T#2m
18	setOut	Bool	false

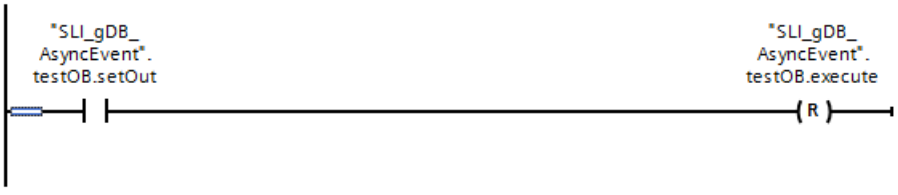
"SLI_ciOB_AsyncEvent" OB

You create an "SLI_ciOB_AsyncEvent" OB and monitor its behavior during disabling/enabling.

Network 1: You now call the instruction "TP" in the OB and create the following interconnections.



Network 2: Reset the tag "testOB.execute" as follows:



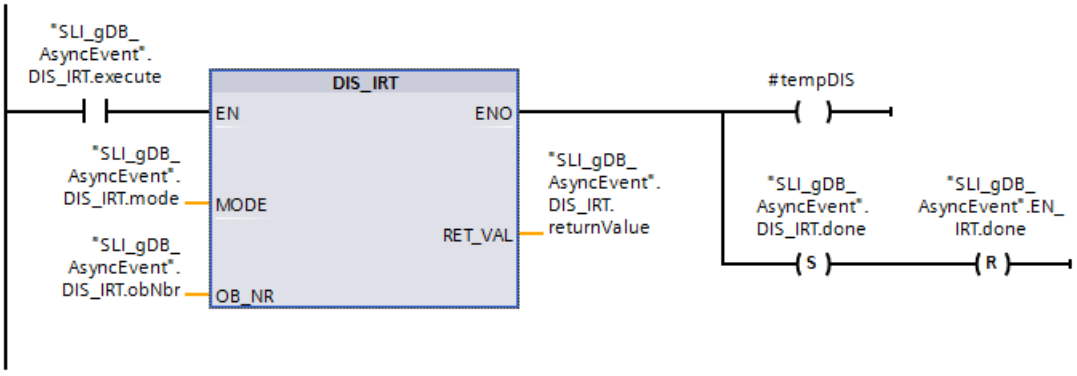
"SLI_FC_AsyncEvent" FC

To call the program parts, create the FC "SLI_FC_AsyncEvent".

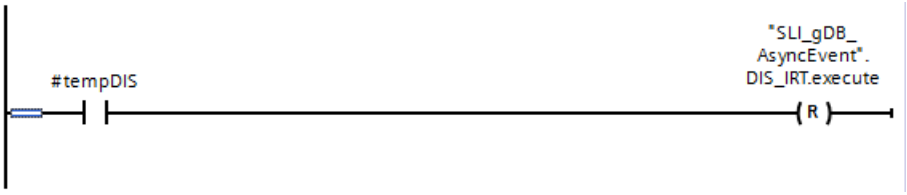
Create the following local tags in the FC.

7	Temp	
8	tempDIS	Bool
9	tempEN	Bool

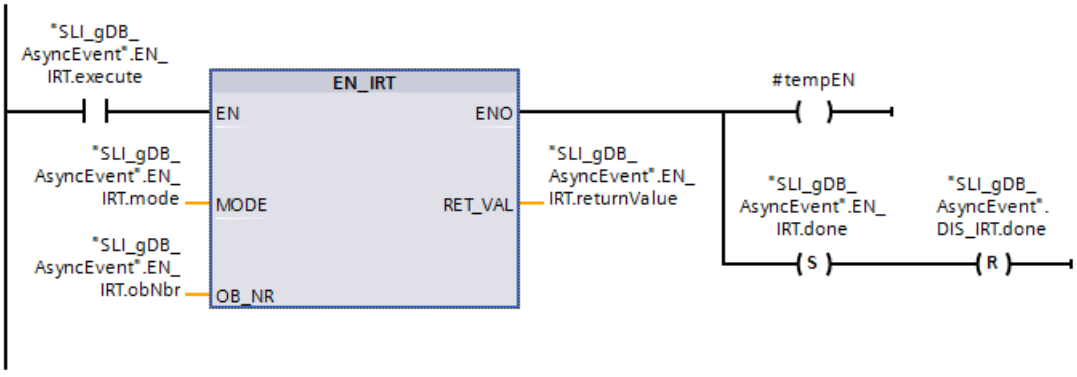
Network 1: Interconnect the parameters of the "DIS_IRT" instruction as follows.



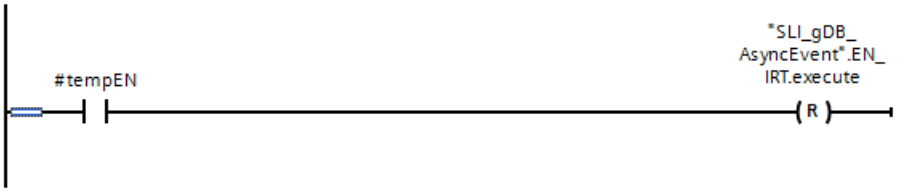
Network 2: Reset the tag "DIS_IRT.execute" as follows:



Network 3: Interconnect the parameters of the "EN_IRT" instruction as follows.



Network 4: Reset the tag "EN_IRT.execute" as follows:



Result of OB "SLI_ciOB_AsyncEvent"

If the normally open contact ("testOB.execute") supplies the signal state "TRUE" once, the "TP" instruction is executed. The tag "testOB.execute" is automatically reset to the signal state "FALSE". The value for the "testTime" tag is incremented until it corresponds to the value of the "testTimeLimit" tag.

SLI_gDB_AsyncEvent

	Name	Data type	Start value	Monitor value
1	Static			
2	DIS_IRT	Struct		
3	execute	Bool	false	FALSE
4	done	Bool	false	FALSE
5	mode	Byte	16#02	16#02
6	obNbr	Int	33	33
7	returnValue	Int	0	0
8	EN_IRT	Struct		
9	execute	Bool	false	FALSE
10	done	Bool	false	FALSE
11	mode	Byte	16#02	16#02
12	obNbr	Int	33	33
13	returnValue	Int	0	0
14	testOB	Struct		
15	execute	Bool	false	FALSE
16	testTime	Time	T#0ms	T#19S_100MS
17	testTimeLimit	Time	T#2m	T#2M
18	setOut	Bool	false	TRUE

Result of FC "SLI_FC_AsyncEvent"

Network 1:

If the normally open contact ("DIS_IRT.execute") supplies the signal state "TRUE", the "DIS_IRT" instruction is executed. The operating mode "2" to be used is specified according to the input parameter MODE ("DIS_IRT.mode"). OB33 is addressed and its processing disabled according to the input parameter OB_NR ("DIS_IRT.obNbr").

The output parameter RET_VAL ("DIS_IRT.returnValue") indicates that processing in the example is taking place without errors. You save the processing of "DIS_IRT" in the tag "DIS_IRT.done". You also reset the "EN_IRT.done" tag.

The value of the "testOB.testTime" tag is no longer incremented. Nevertheless, the IEC timer of the "TP" instruction continues to count in the background.

SLI_gDB_AsyncEvent

	Name	Data type	Start value	Monitor value
1	Static			
2	DIS_IRT	Struct		
3	execute	Bool	false	FALSE
4	done	Bool	false	TRUE
5	mode	Byte	16#02	16#02
6	obNbr	Int	33	33
7	returnValue	Int	0	0
8	EN_IRT	Struct		
9	execute	Bool	false	FALSE
10	done	Bool	false	FALSE
11	mode	Byte	16#02	16#02
12	obNbr	Int	33	33
13	returnValue	Int	0	0
14	testOB	Struct		
15	execute	Bool	false	FALSE
16	testTime	Time	T#0ms	T#28S_500MS
17	testTimeLimit	Time	T#2m	T#2M
18	setOut	Bool	false	TRUE

Network 2:

If the normally open contact ("#tempDIS") supplies the signal state "TRUE", the "DIS_IRT.execute" tag is reset.

Network 3:

If the normally open contact ("EN_IRT.execute") supplies the signal state "TRUE", the "EN_IRT" instruction is executed. The operating mode "2" to be used is specified according to the input parameter MODE ("EN_IRT.mode"). OB33 is addressed and its processing enabled once again according to the input parameter OB_NR ("EN_IRT.obNbr").

The output parameter RET_VAL ("EN_IRT.returnValue") indicates that processing in the example is taking place without errors. You save the processing of "EN_IRT" in the tag "EN_IRT.done". You also reset the "DIS_IRT.done" tag.

The value of the "testOB.testTime" tag is still incremented or changed according to the IEC timer of the "TP" instruction.

SLI_gDB_AsyncEvent

	Name	Data type	Start value	Monitor value
1	Static			
2	DIS_IRT	Struct		
3	execute	Bool	false	FALSE
4	done	Bool	false	FALSE
5	mode	Byte	16#02	16#02
6	obNbr	Int	33	33
7	returnValue	Int	0	0
8	EN_IRT	Struct		
9	execute	Bool	false	FALSE
10	done	Bool	false	TRUE
11	mode	Byte	16#02	16#02
12	obNbr	Int	33	33
13	returnValue	Int	0	0
14	testOB	Struct		
15	execute	Bool	false	FALSE
16	testTime	Time	T#0ms	T#1M_40S_800MS
17	testTimeLimit	Time	T#2m	T#2M
18	setOut	Bool	false	TRUE

Network 4:

If the normally open contact ("#tempEN") supplies the signal state "TRUE", the "EN_IRT.execute" tag is reset.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

DIS_IRT: Disable interrupt event (Page 3861)

EN_IRT: Enable interrupt event (Page 3863)

DIS_AIRT: Delay execution of higher priority interrupts and asynchronous error events (S7-1200, S7-1500)

Description

You use "DIS_AIRT" to delay the processing of interrupt OBs whose priority are higher than the priority of the current organization block.

You can call "DIS_AIRT" multiple times in an organization block. The "DIS_AIRT" calls are counted by the operating system. Processing is delayed more and more each time "DIS_AIRT" is executed. To cancel a delay, you must execute the "EN_AIRT (Page 3870)" instruction. The processing delay applies until the number of executions of "EN_AIRT (Page 3870)" matches the number of calls of "DIS_AIRT" or the current OB has been worked through.

You query the number of delays in the RET_VAL parameter of the "DIS_AIRT instruction". If the value in the RET_VAL parameter is "0", there are no delays.

Parameters

The following table shows the parameters of the "DIS_AIRT" instruction:

Parameter	Declaration	Data type	Memory area	Description
RET_VAL	Return	INT	I, Q, M, D, L	Number of delays

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Return value	Description
n	"n" shows the number of processing delays, i.e., the number of DIS_AIRT calls, after the instruction is completed. If n = 0, interrupt handling is first enabled again; see EN_AIRT (Page 3870).

Example

You can find the example here: Program example for DIS_AIRT & EN_AIRT (Page 3871).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

EN_AIRT: Enable execution of higher priority interrupts and asynchronous error events (S7-1200, S7-1500)

Description

You use "EN_AIRT" to enable processing of organization blocks when interrupts occur that have been delayed by the "DIS_AIRT (Page 3869)" instruction.

When "EN_AIRT" is executed, you cancel a processing delay that was registered by the operating system when "DIS_AIRT (Page 3869)" was called. To cancel all delays, the number of "EN_AIRT" executions must be equal to the number of "DIS_AIRT (Page 3869)" calls. Example: You have called "DIS_AIRT (Page 3869)" five times. As a result, the processing is also delayed five times. After this, you must call the "EN_AIRT" instruction five times in order to cancel all five delays.

In the EN_AIRT parameter of the "EN_AIRT" instruction, you query the number of interrupt delays that have not yet been enabled after the execution of "RET_VAL". The value "0" in the RET_VAL parameter means that all delays enabled by "DIS_AIRT (Page 3869)" have been cancelled.

Parameters

The following table shows the parameters of the "EN_AIRT" instruction:

Parameter	Declaration	Data type	Memory area	Description
RET_VAL	Return	INT	I, Q, M, D, L	Number of configured delays

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Return value/ Error code* (W#16#...)	Description
n	"n" shows the number of processing delays that are not yet enabled after the instruction is completed. If n = 0, interrupt handling is first enabled again.
8080	Although the interrupt processing was already enabled, the function was called again.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for DIS_AIRT & EN_AIRT (Page 3871).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

Program example for DIS_AIRT & EN_AIRT (S7-1200, S7-1500)

Introduction

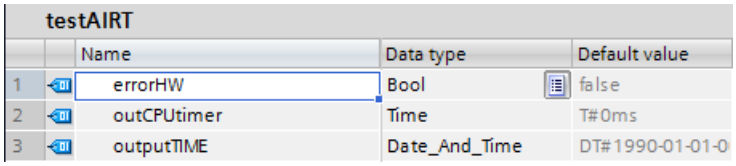
In the following example you delay processing of a cyclic interrupt OB or of interrupt OBs with a higher priority with the help of the instructions "DIS_AIRT" and "EN_AIRT". A test program is not interrupted by an interrupt when you call the instructions.

You use the test program to save the time and diagnostics data in a data memory.

- To do this, you read out the CPU time with the instruction "TIME_TCK" and the diagnostics data with the instruction "GET_DIAG" in a cyclic interrupt OB.
- In a cyclic OB (OB1), you also read out the system time of the CPU clock with the instruction "RD_SYS_T".

Storage of data

Create the PLC data type "testAIRT" for the data memory.



testAIRT

		Name	Data type	Default value
1		errorHW	Bool	false
2		outCPUtimer	Time	T#0ms
3		outputTIME	Date_And_Time	DT#1990-01-01-0

Create the following structures and tags for storing the data in a global data block.

- For DIS_AIRT, EN_AIRT, and time recording:

SLI_gDB_delayAIRT			
	Name	Data type	Start value
1	Static		
2	startOB1	Bool	false
3	startOB_ci	Bool	false
4	delay	Struct	
5	executeDel	Bool	false
6	executeEn	Bool	false
7	nbrOfDelays	Int	0
8	timeData	Struct	
9	retValRDsys	Int	0
10	outCPUtimer	Time	T#0ms

- For diagnostics:

Note

Find hardware identifier ("laddr")

Use "PLC tags > System constants" to find the hardware identifier. For an S7-1500 CPU: Find the entry "Local~Common" with the data type "HW_Submodule". The associated cell "value" contains the hardware identifier.

11	diag	Struct	
12	mode	UInt	2
13	laddr	HW_ANY	50
14	returnValue	Int	0
15	countDiagDetails	UInt	0
16	getDIAG	DNN	
17	SubordinateState	UInt	0
18	SubordinateIOState	Word	16#0
19	DNNmode	Word	16#0
20	itemiseIOState	Struct	
21	No_maintenance_required	Bool	false
22	The module_or_device_is_disabled	Bool	false
23	Maintenance_required	Bool	false
24	Maintenance_demanded	Bool	false
25	Error	Bool	false
26	The_module_or_the_device_can_not_be_reached_from_the_CPU	Bool	false
27	Inputs_or_outputs_are_not_available	Bool	false

- For the data memory:

28	dataStorage	Array[0..3] of "testAIRT"	
29	pos	Int	0

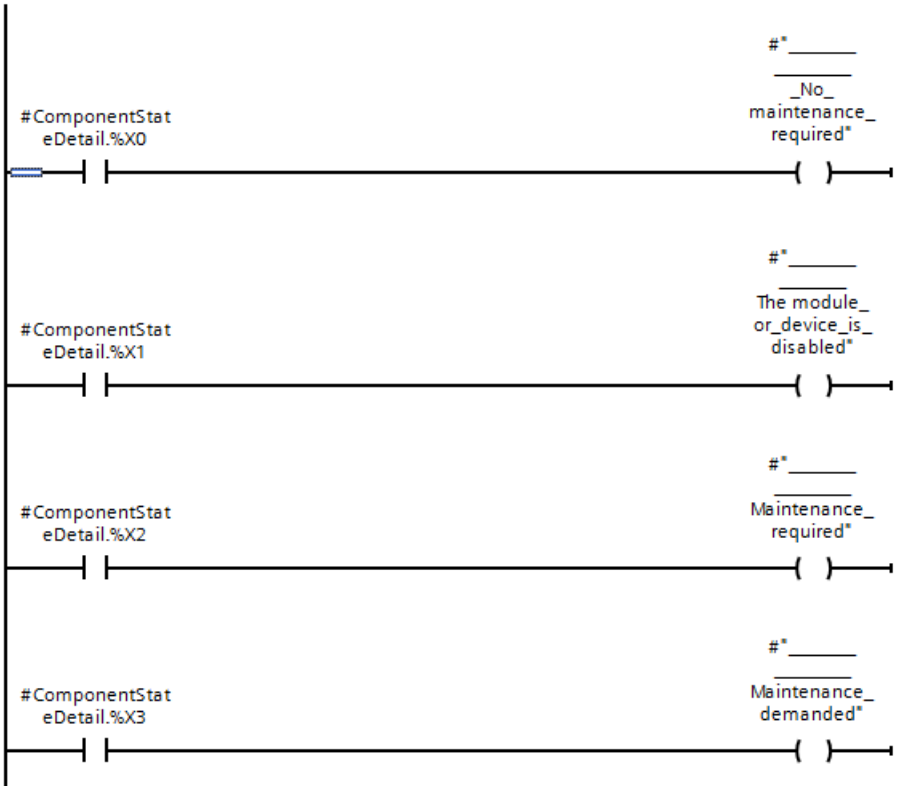
Interconnect parameters: in the FC "SLI_FC_itemise_IOState_AIRT"

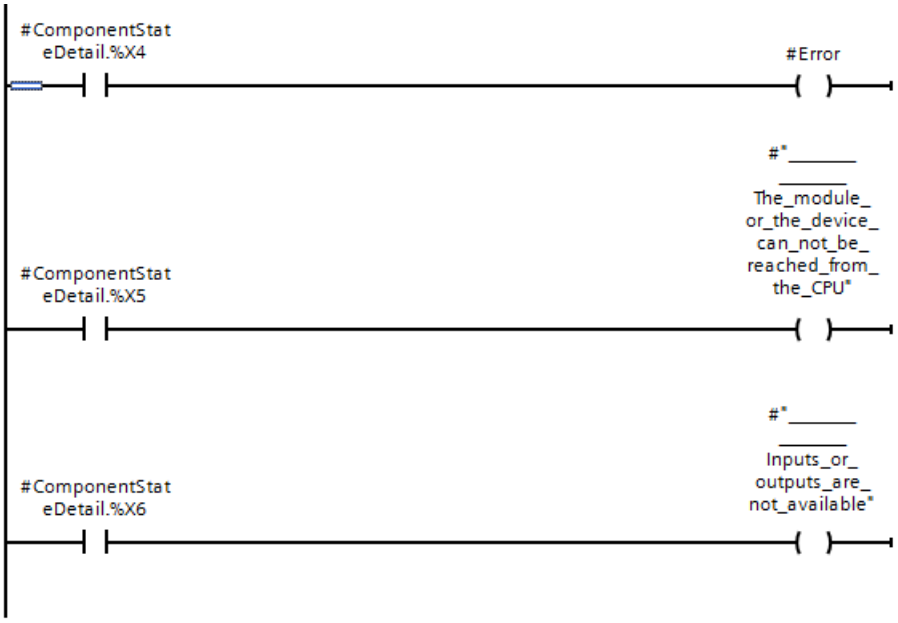
To further evaluate the diagnostics data of the "GET_DIAG" instruction, create an FC "SLI_FC_itemise_IOState_AIRT".

Create the following local tags in the FC.

SLI_FC_itemise_IOState_AIRT		
	Name	Data type
1	Input	
2	ComponentStateDetail	Word
3	Output	
4	_____No_maintenance_required	Bool
5	_____The module_or_device_is_disabled	Bool
6	_____Maintenance_required	Bool
7	_____Maintenance_demanded	Bool
8	Error	Bool
9	_____The_module_or_the_device_can_not_be_reached_from_the_CPU	Bool
10	_____Inputs_or_outputs_are_not_available	Bool

Network 1: You create the following interconnections.





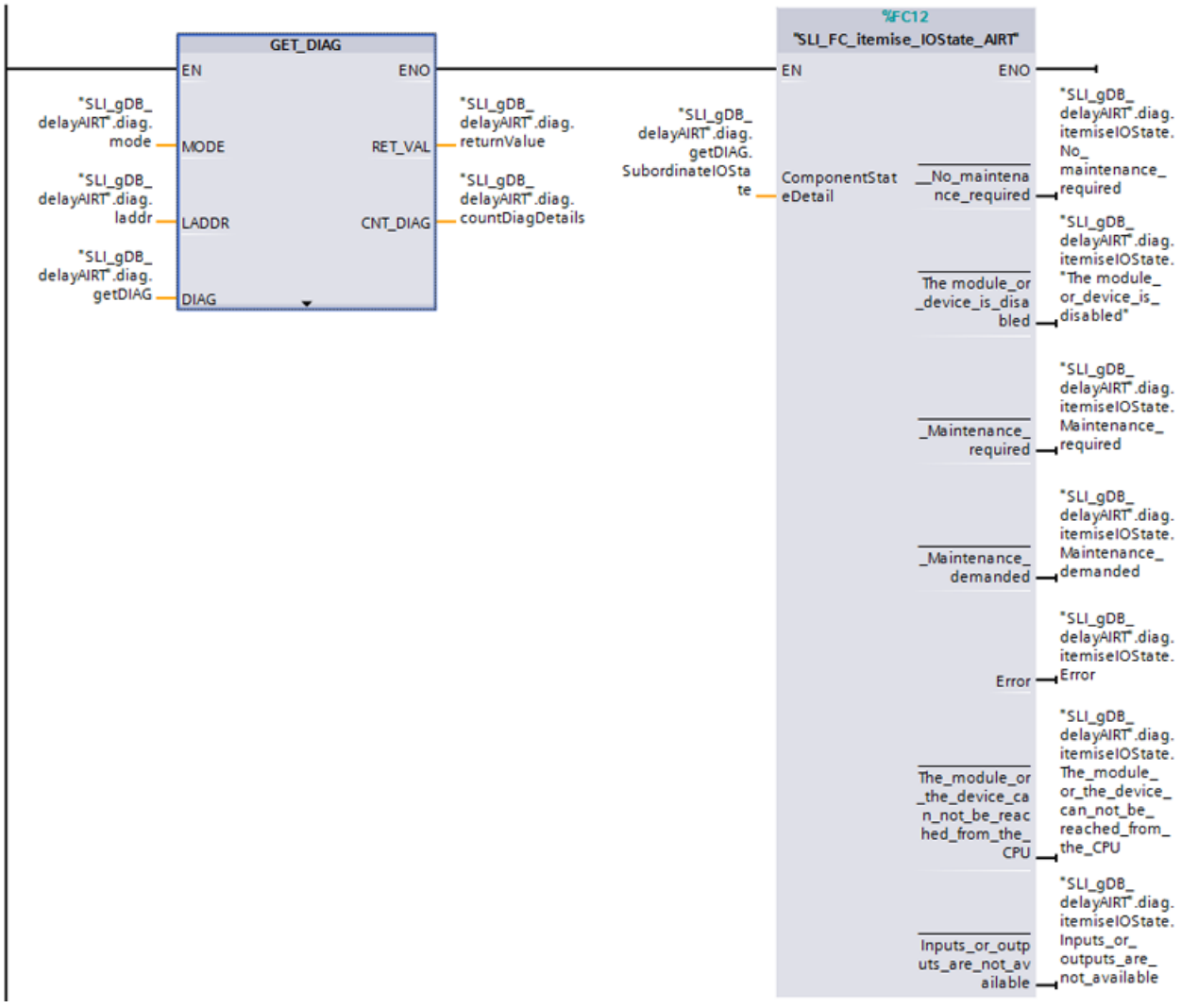
Interconnect parameters: in the "SLI_ciOB_diagProg_AIRT" OB

You create a cyclic interrupt OB "SLI_ciOB_diagProg_AIRT" and monitor its behavior during the delay.

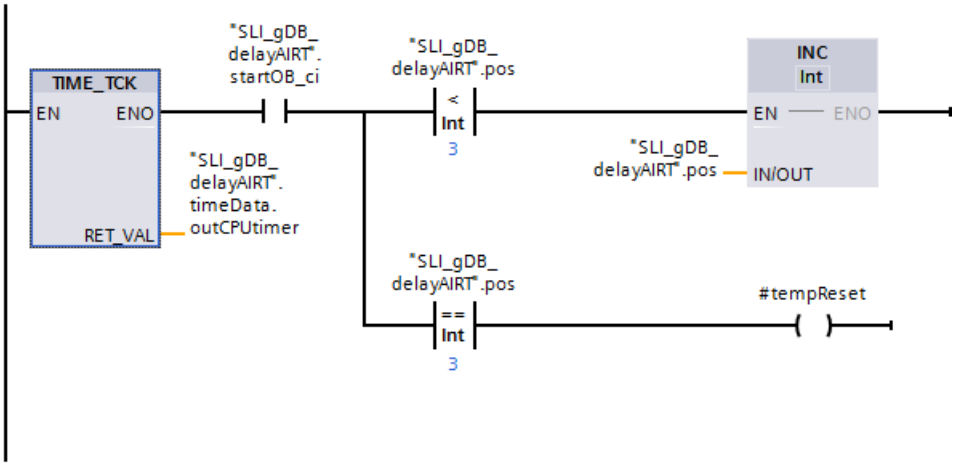
You create the following local tag in the OB.



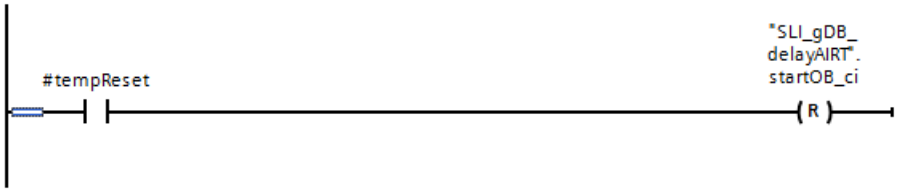
Network 1: You now call the instruction "GET_DIAG" in the OB and create the following interconnections.



Network 2: You record the CPU time as follows and increment the "pos" tag according to the following condition.



Network 3: Reset the tag "startOB_ci" as follows:



Interconnect parameters: in the FC "SLI_FC_saveDiag_AIRT"

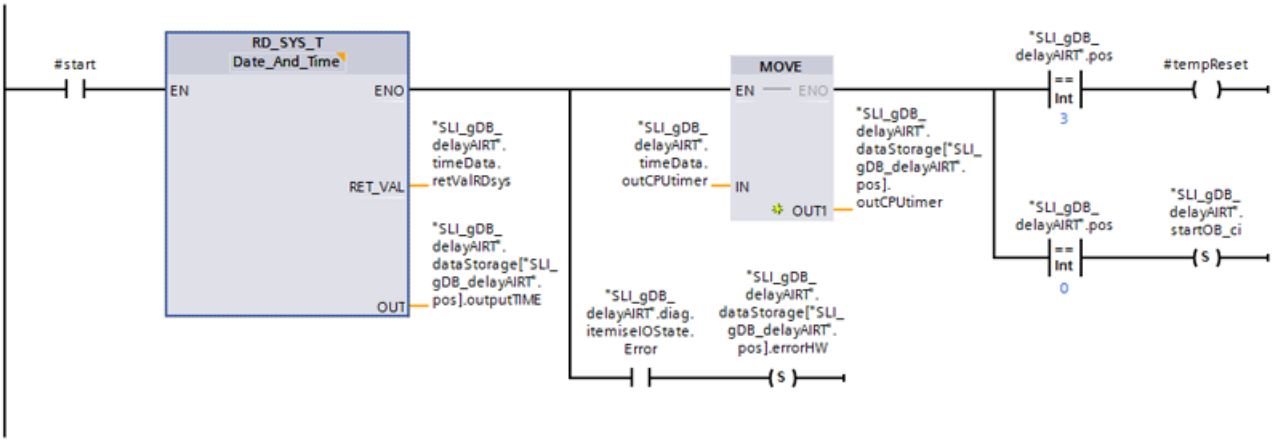
To read the system time and merge the time and diagnostics data, create the FC "SLI_FC_saveDiag_AIRT".

Create the following local tags in the FC.

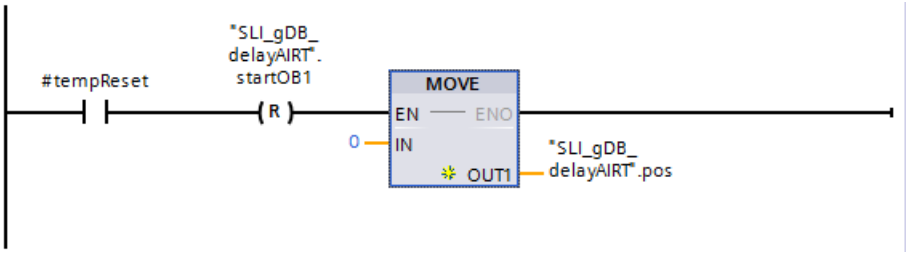
SLI_FC_saveDiag_AIRT

	Name	Data type
1	Input	
2	start	Bool
3	Output	
4	<Add new>	
5	InOut	
6	<Add new>	
7	Temp	
8	tempReset	Bool

Network 1: You create the following interconnections.



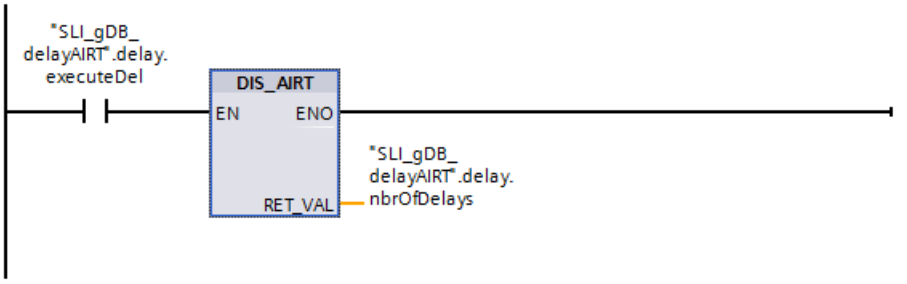
Network 2: Reset the tag "startOB1" as follows:



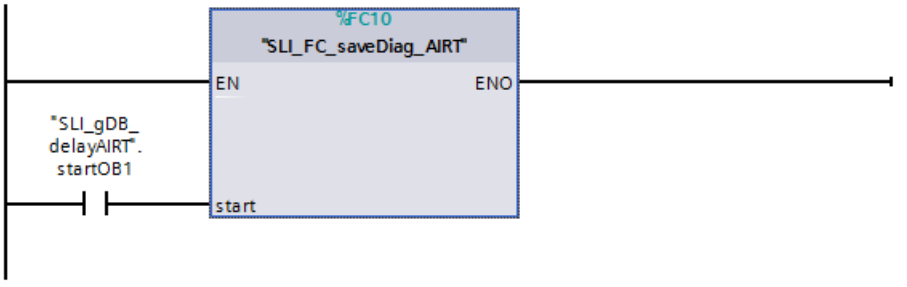
Interconnect parameters: in the FC "SLI_FC_delayAIRT"

To call the test program, create the FC "SLI_FC_delayAIRT".

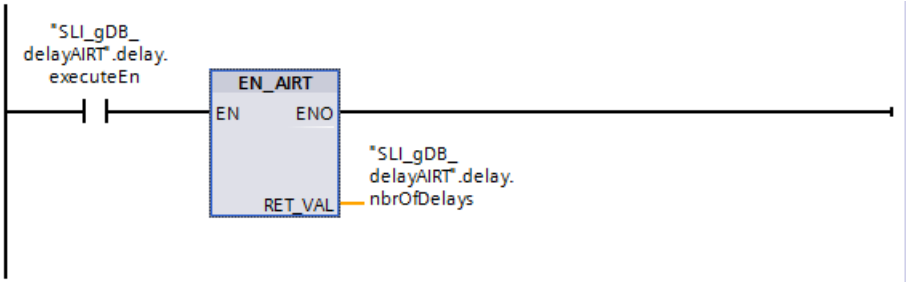
Network 1: Interconnect the parameters of the "DIS_AIRT" instruction as follows.



Network 2: Interconnect the FC "SLI_FC_saveDiag_AIRT" as follows.



Network 3: Interconnect the parameters of the "EN_AIRT" instruction as follows.



Procedure for starting the test scenario

To start the test program, follow these steps:

Note

If you do not want to delay processing of the cyclic interrupt OB ("SLI_ciOB_diagProg_AIRT"), skip steps 1 and 2.

1. Set the tag "delay.executeDel" to "TRUE".
2. Set the tag "delay.executeEn" to "TRUE".
3. Set the tag "startOB1" to "TRUE". Afterwards, the tag "startOB_ci" is set automatically. The "startOB1" tag and the "startOB_ci" tag are automatically reset after filling the data memory.

Procedure for stopping the processing of DIS_AIRT and EN_AIRT

To stop the delayed processing of the cyclic interrupt OB ("SLI_ciOB_diagProg_AIRT"), follow these steps:

1. Set the tag "delay.executeEn" to "FALSE".
2. Set the tag "delay.executeDel" to "FALSE".

Result of OB "SLI_ciOB_diagProg_AIRT"

The cyclic interrupt OB ("SLI_ciOB_diagProg_AIRT") is called every 100 ms. It reads the diagnostics data with the instruction "GET_DIAG" and outputs the CPU time.

If the normally open contact ("startOB_ci") supplies the signal state "TRUE" and the "pos" tag returns a value greater than "3", the "pos" tag is incremented. If the "pos" tag returns the value "3", the "startOB_ci" tag is reset.

SLI_gDB_delayAIRT

	Name	Data type	Start value	Monitor value
1	Static			
2	startOB1	Bool	false	FALSE
3	startOB_ci	Bool	false	FALSE
4	delay	Struct		
5	executeDel	Bool	false	FALSE
6	executeEn	Bool	false	FALSE
7	nbrOfDelays	Int	0	0
8	timeData	Struct		
9	retValRDsys	Int	0	0
10	outCPUtimer	Time	T#0ms	T#5M_26S_588MS

If execution of the cyclic interrupt OB ("SLI_ciOB_diagProg_AIRT") is not delayed, the values of the data memory appear as below.

28	dataStorage	Array[0..3] o...		
29	dataStorage[0]	"testAIRT"		
30	errorHW	Bool	false	FALSE
31	outCPUtimer	Time	T#0ms	T#2M_9S_388MS
32	outputTIME	Date_And_Time	DT#1990-01-01-0	DT#2017-01-20-09:58:24.384
33	dataStorage[1]	"testAIRT"		
34	errorHW	Bool	false	FALSE
35	outCPUtimer	Time	T#0ms	T#2M_9S_488MS
36	outputTIME	Date_And_Time	DT#1990-01-01-0	DT#2017-01-20-09:58:24.484
37	dataStorage[2]	"testAIRT"		
38	errorHW	Bool	false	FALSE
39	outCPUtimer	Time	T#0ms	T#2M_9S_588MS
40	outputTIME	Date_And_Time	DT#1990-01-01-0	DT#2017-01-20-09:58:24.585
41	dataStorage[3]	"testAIRT"		
42	errorHW	Bool	false	FALSE
43	outCPUtimer	Time	T#0ms	T#2M_9S_688MS
44	outputTIME	Date_And_Time	DT#1990-01-01-0	DT#2017-01-20-09:58:24.587
45	pos	Int	0	0

Result of FC "SLI_FC_delayAIRT"

Network 1:

If the normally open contact ("delay.executeDel") supplies the signal state "TRUE", the "DIS_AIRT" instruction is executed. Processing of the cyclic interrupt OB ("SLI_ciOB_diagProg_AIRT") or of an interrupt OB with higher priority is delayed. The output parameter RET_VAL ("delay.nbrOfDelays") indicates that a delay is active in the calling OB (OB1).

Network 2:

If the normally open contact ("startOB1") supplies the signal state "TRUE", the test program of the FC "SLI_FC_saveDiag_AIRT" is executed.

In the FC "SLI_FC_saveDiag_AIRT":

The system time is read with the instruction "RD_SYS_T" and stored in the data memory "dataStorage". The cell of the data memory that is to be used is determined by the "pos" tag.

The CPU time ("outCPUtimer") is also stored in the data memory. If the "diag.itemiseIOState.Error" tag has the signal state "TRUE", the "errorHW" tag is set to "TRUE" in the data memory.

The following is decided in the next steps:

- If "pos" has the value "0": The value of the "startOB_ci" tag is set to "TRUE". This means the "pos" tag can be incremented in the cyclic interrupt OB ("SLI_ciOB_diagProg_AIRT").
- If "pos" has the value "3": The tags "startOB1" and "pos" are reset.

If execution of the cyclic interrupt OB ("SLI_ciOB_diagProg_AIRT") is delayed, the values of the data memory appear as below.

28	dataStorage	Array[0..3] o...		
29	dataStorage[0]	"testAIRT"		
30	errorHW	Bool	false	FALSE
31	outCPUtimer	Time	T#0ms	T#25M_52S_688MS
32	outputTIME	Date_And_Time	DT#1990-01-01-0	DT#2017-01-20-10:22:07.684
33	dataStorage[1]	"testAIRT"		
34	errorHW	Bool	false	FALSE
35	outCPUtimer	Time	T#0ms	T#25M_52S_788MS
36	outputTIME	Date_And_Time	DT#1990-01-01-0	DT#2017-01-20-10:22:07.784
37	dataStorage[2]	"testAIRT"		
38	errorHW	Bool	false	FALSE
39	outCPUtimer	Time	T#0ms	T#25M_52S_888MS
40	outputTIME	Date_And_Time	DT#1990-01-01-0	DT#2017-01-20-10:22:07.884
41	dataStorage[3]	"testAIRT"		
42	errorHW	Bool	false	FALSE
43	outCPUtimer	Time	T#0ms	T#25M_52S_988MS
44	outputTIME	Date_And_Time	DT#1990-01-01-0	DT#2017-01-20-10:22:07.887
45	pos	Int	0	0

Network 3:

If the normally open contact ("delay.executeEn") supplies the signal state "TRUE", the "EN_AIRT" instruction is executed. Processing of the cyclic interrupt OB ("SLI_ciOB_diagProg_AIRT") or of an interrupt OB with higher priority is enabled once again. The output parameter RET_VAL ("delay.nbrOfDelays") indicates that a delay is not active in the calling OB (OB1). Processing in the example takes place without errors.

SLI_gDB_delayAIRT

	Name	Data type	Start value	Monitor value
1	Static			
2	startOB1	Bool	false	FALSE
3	startOB_ci	Bool	false	FALSE
4	delay	Struct		
5	executeDel	Bool	false	TRUE
6	executeEn	Bool	false	TRUE
7	nbrOfDelays	Int	0	0
8	timeData	Struct		
9	retValRDsys	Int	0	0
10	outCPUtimer	Time	T#0ms	T#27M_36S_988MS

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

DIS_AIRT: Delay execution of higher priority interrupts and asynchronous error events (Page 3869)

EN_AIRT: Enable execution of higher priority interrupts and asynchronous error events (Page 3870)

11.4.4.8 Alarms (S7-1200, S7-1500)

Program_Alarm: Generate program alarm with associated values (S7-1500)

Description

The instruction "Generate program alarm with associated values" monitors a signal and on a signal change at the parameterSIG generates a program alarm (for definition, see also: AUTOHOTSPOT). An incoming program alarm is generated with a signal change from 0 to 1 and an outgoing program alarm is generated with a signal change from 1 to 0. The program alarm is triggered synchronously to program execution.

You can append up to ten associated values to the program alarm at the parametersSD_i (0 ≤ i ≤10). The associated values are detected at the time of signal change at parameter SIG and assigned to the program alarm. For more information on configuring associated values, refer to the following sections: AUTOHOTSPOT, AUTOHOTSPOT.

Each incoming or outgoing alarm is assigned a time stamp:

- The current system time of the PLC when the signal change occurs is used by default (default value at the TIMESTAMP parameter).
- If you wish to specify a different time stamp, you can create one at the parameter TIMESTAMP.
  The time value must always be specified in system time (i.e. UTC) as this is the time used for time synchronization throughout the plant.
- If an alarm is to be time-stamped with a local time, a conversion module, which converts the local time to system time, must be connected in series. This is the only way to make sure that the time stamp is shown properly in the alarm display.

To use the current system time of the CPU, set the TIMESTAMP parameter to its default value (LDT#1970-01-01-00:00:00.0).

Call of the instruction "Generate program alarm with associated values"

Call the instruction only in a function block (FB). The block is processed synchronously. An alarm is triggered as soon as the block is exited. If there is an error during processing, an error code is output.

Once the instruction has been inserted in the FB, a multi-instance of the data type "Program_Alarm" is created in the "Static" section of the block interface. Choose any name for this multi-instance in the dialog which appears. It is also the name of the program alarm.

Finally, add the parameters of the instruction in line with your requirements (see the "Parameters" table).

Configuring the program alarm

The program alarm settings are displayed in the "Properties" window when you select the name of the program alarm in the "Static" section or in the network of the FB. Select the alarm class, priority, etc. and edit the alarm text.

The settings that are made here can be edited in the project tree. To do this, open the "Alarms" tab under "PLC supervisions & alarms"; in it open the "Program alarms" tab. All existing program alarms are then displayed in the "Alarm types" table.

Parameters

The following table shows the parameters of the instruction "Generate program alarm with associated values":

Parameter	Declaration	Data type	Memory area	Description
SIG	Input	BOOL	I, Q, M, D, L, T, C or constant	The signal to be monitored. • Positive signal edge: An incoming program alarm is generated • Negative signal edge: An outgoing program alarm is generated
TIMESTAMP	Input	LDT	M, D, L or constant	This parameter is used to assign an alarm a time stamp, for example, from an input signal with a distributed stamp. The time value must always be specified in system time (i.e. UTC), as this is the time used for time synchronization throughout the plant. • "Not assigned" means that the system time of the CPU will be used as the interrupt time stamp when a signal change occurs (default). • Any system time input will be used as the interrupt time stamp when a signal change occurs. Note: If an interrupt is to be time-stamped with a local time, a conversion module must be connected in series to convert the local time to system time. This is the only way to make sure that the time stamp is shown correctly in the interrupt display.

Parameter	Declaration	Data type	Memory area	Description
SD_i	Input	VARIANT	I, Q, M, D, L	i-th associated value (1 ≤ i ≤ 10) You can use binary numbers, integers, floating-point numbers, or character strings as associated values.
Error	Output	BOOL	I, Q, M, D, L	Status parameter Error Error = TRUE indicates that an error has occurred during processing. The possible error cause is displayed at the Status parameter.
Status	Output	WORD	I, Q, M, D, L	Status parameter Status Display of error information (see "Error and Status parameters").

You can find additional information on valid data types under "See also".

Error and Status parameters

The following table contains all specific error information that is output via the Error and Status parameters.

Error	Status*	Explanation
0	0000	No error, or the instruction was not processed because there was no signal edge at the parameter SIG.
1	0085	"Only information" alarm type
1	8001	Invalid static alarm information
1	8002	No valid static alarm information
1	8004	Maximum size of 512 bytes reached for the associated values of the alarm.
1	8005	A positive signal edge is pending at parameter SIG and there is still an alarm pending which has not yet been acknowledged.
1	8007	Outgoing alarm which was not preceded by an incoming alarm.
1	8087	Static alarms are deactivated.
1	8089	Alarm is too long.
1	80Ax	The SD_i parameter has an invalid value.
1	80C1	The CPU does not generate any alarms at this time because initialization routines are running (this is the case after download in RUN, for example). Try again later.
1	80C2	The maximum number of alarms per time unit has been sent. Try again later.
1	80C3	All dynamic alarm instances are being used. Try again later.
1	80C4	Alarm is output and is not overwritten. Try again later.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

Example

In the following example, you generate a program alarm with associated value for a signal change.

Create one tag in a global data block for storing the signal value to be monitored.

Program_Alarm_Tags

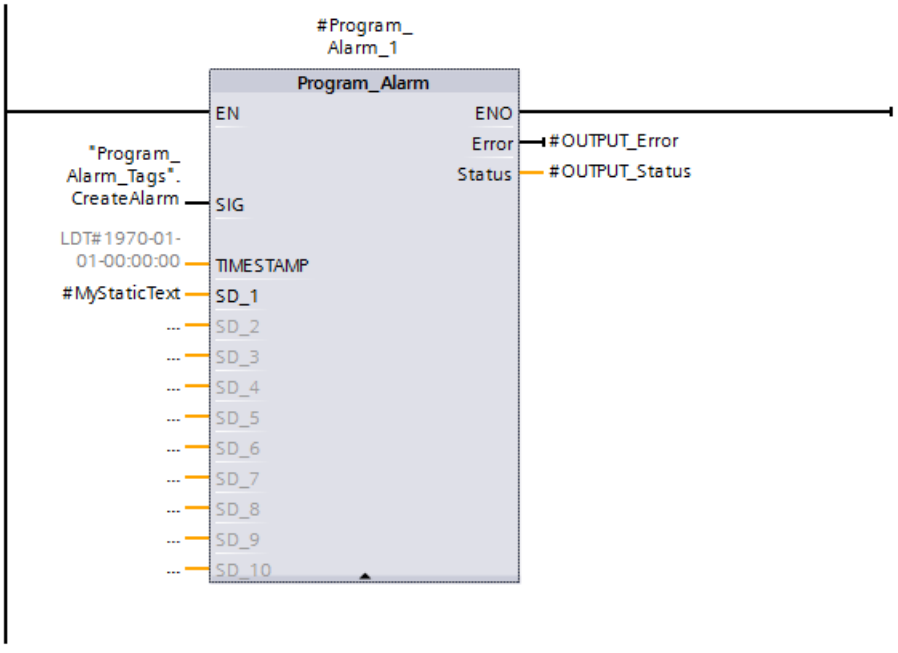
		Name	Data type	Start value
1		Static		
2		CreateAlarm	Bool	false

The instruction is called in a function block. Create four parameters for the function block for interconnection of the instruction.

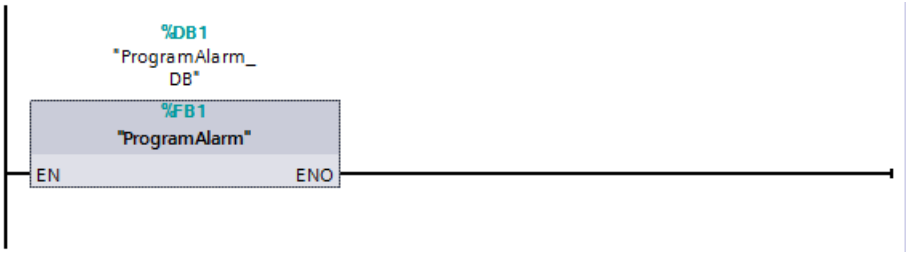
ProgramAlarm

		Name	Data type	Default value
1		Input		
2		<Add new>		
3		Output		
4		<Add new>		
5		InOut		
6		<Add new>		
7		Static		
8		MyStaticText	String	'MyStaticText'
9		Program_Alarm_1	Program_Alarm	
10		Temp		
11		OUTPUT_Error	Bool	
12		OUTPUT_Status	Word	

Interconnect the parameters of the instruction as follows.



Call the function block in an OB.



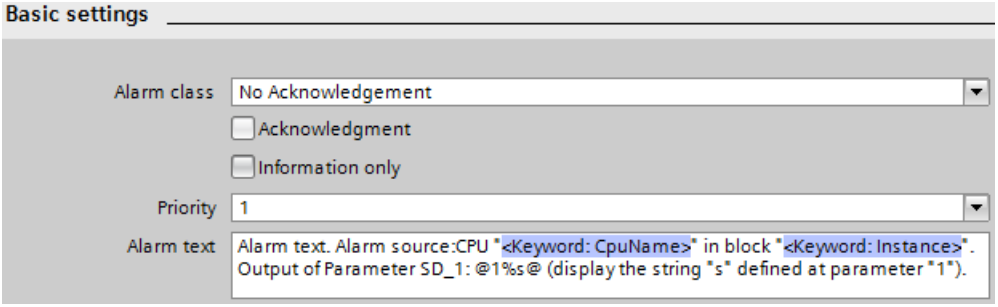
A PLC alarm is generated automatically when the instruction is created. Open the PLC supervisions & alarms dialog and select the alarm to be edited in the Alarms > Program alarms tab. Create the alarm text, including two key words in the Properties > Basic settings of the alarm.

Note: If you right-click in the text box, you can insert a key word, tag, or text list.

As a result of the character sequence "@1%s@", the value of parameter SD_1 ("#MyStaticText") is read out and output as a character string.

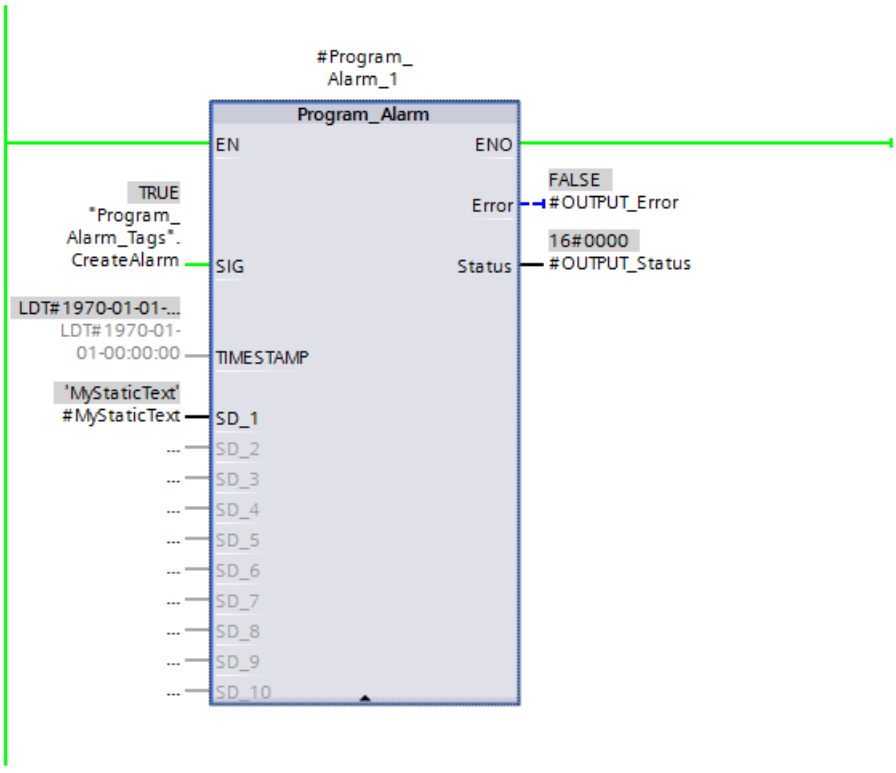
See also:

- AUTOHOTSPOT
- AUTOHOTSPOT
- Keywords for alarm texts



If parameter SIG ("CreateAlarm") has the signal state "TRUE", the PLC alarm is output. At output parameter STATUS ("OUTPUT_Status"), the value "0001" indicates that a signal change has occurred. Then, it is indicated that no further processing is taking place (the value is "0000"). At parameter SD_1 ("#MyStaticText"), the associated value of the PLC alarm is output.

A time stamp can be transferred to parameter TIMESTAMP. If not interconnected, parameter TIMESTAMP outputs the local time of the CPU clock. At output parameter ERROR ("OUTPUT_Error"), it is indicated that the processing of the instruction was error-free.



Use the web server of the CPU, for example, to output the PLC alarm. The following is necessary in order to use the web server:

- The web server must be activated in the CPU settings.

Use the Internet browser to open the Web server (via the IP address of the CPU) and log on in the Web server menu. The CPU outputs the alarm text as long as the signal being monitored ("CreateAlarm") supplies the value "TRUE".

Time	Message text
02:32:09.226 pm	Alarm text. Alarm source:CPU "PLC_2" in block "ProgramAlarm_DB". Output of Par

Program code

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

Example

You can find a detailed application example in the Siemens Industry Online Support (https://support.industry.siemens.com/cs/document/98210758?dti=0&lc=en-WW).

See also

Overview of the valid data types (Page 2239)

Get_AlarmState: Output alarm status (S7-1500)

Description

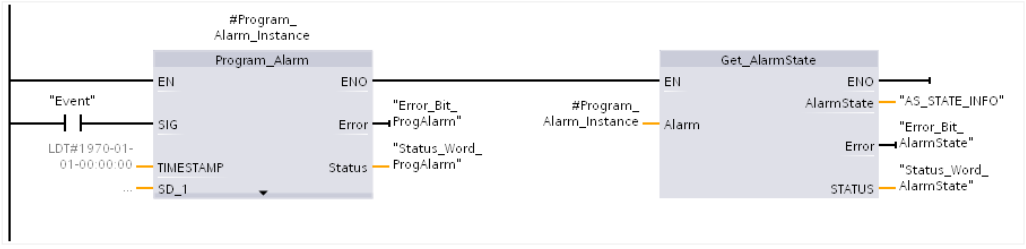
Use the instruction "Output alarm status" to output the alarm status of a program alarm.

There are three possible alarm statuses:

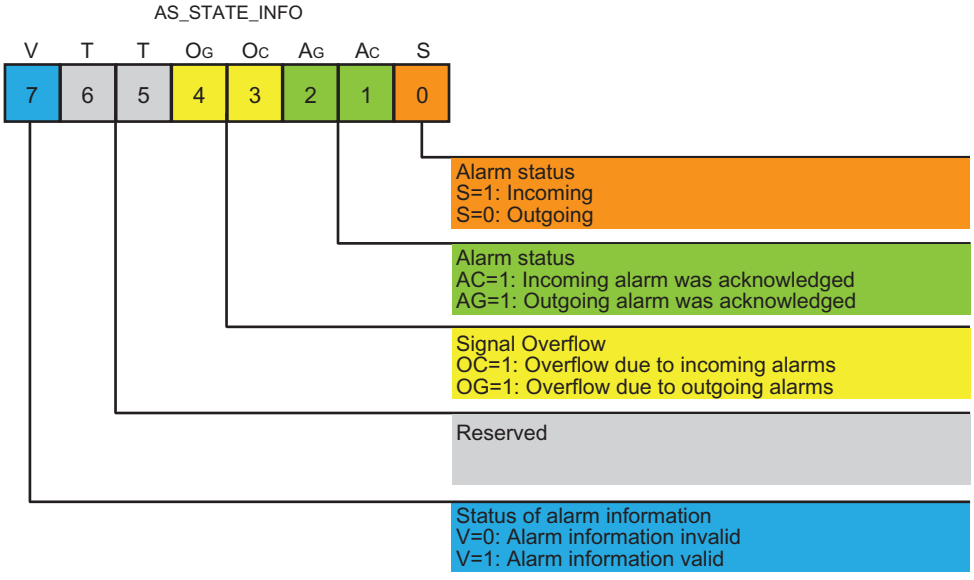
- Incoming
- Outgoing
- Acknowledged

The output of the alarm status always refers to a program alarm that was created using the instruction "Generate program alarm with associated values".

The program alarm is selected with the Alarm input parameter. Specify the instance DB of the instruction "Generate program alarm with associated values" at the parameter Alarm.



The alarm status is returned in a byte via the AlarmState output parameter. The meaning of the individual bits is shown in the following figure:



The execution status of the instruction is shown with the output parameters Error and STATUS.

Parameters

The following table lists the parameters of the instruction "Output alarm status":

Parameter	Declaration	Data type	Memory area	Description
Alarm	Input	ALARM_BASE	D	Instance of the instruction "Generate program alarm with associated values" • Alarm.Messagetype = Alarm_AP, the the bit Ac has either the signal status 0 or 1 and the bit Ag has the signal status 1 – not active: 0x86 (1000 0110) – active/not acknowledged: 0x85 (1000 0101) – active/acknowledged: 0x87 (1000 0111) – outgoing/not acknowledged: 0x84 (1000 0100) • Alarm.Messagetype = Notify_AP, the bits Ac and Ag both have the signal state 1. – not active: 0x86 (1000 0110) – active: 0x87 (1000 0111) • Alarm.Messagetype = Inforeport_AP, the bits Ac and Ag both have the signal status 1 – not active: 0x86 (1000 0110) – (temporarily) active: 0x87 (1000 0111) If the alarm is not active, bit S always has the signal state 0.
AlarmState	Output	BYTE	I, Q, M, D, L	Status of the alarm as a bit array

Parameter	Declaration	Data type	Memory area	Description
Error	Output	BOOL	I, Q, M, D, L	Status parameter • 0: No error • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy STATUS to a free data area.

You can find additional information on valid data types under "See also".

Parameter STATUS

Error code* (W#16#...)	Explanation
8001	Invalid static alarm instance
8002	ID of the alarm is invalid
8003	No active alarms within the alarm class • Alarm_AP: The alarm is outgoing and acknowledged. • Notify_AP: The alarm is outgoing. • Inforeport Bit V is set to signal status 0.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

In the following example, you output the alarm status of a program alarm.

Create four tags in a global data block for storing the data.

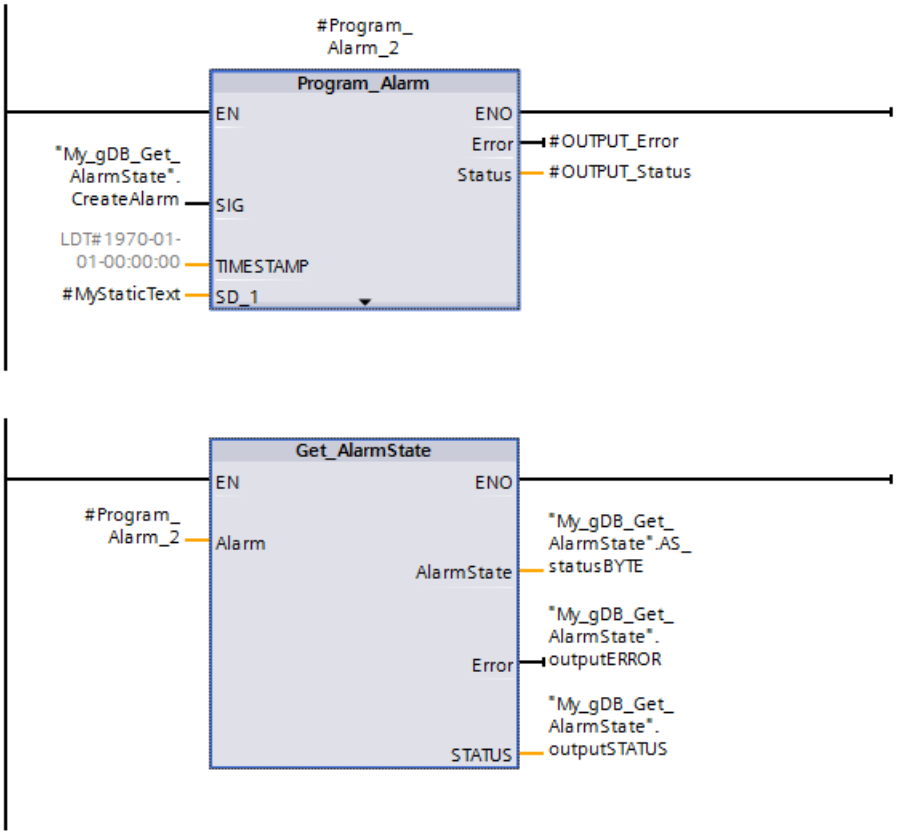
My_gDB_Get_AlarmState

	Name	Data type	Start value
1	Static		
2	AS_statusBYTE	Byte	16#0
3	outputERROR	Bool	false
4	outputSTATUS	Word	16#0
5	CreateAlarm	Bool	false

The "Get_AlarmState" instruction is called together with the "Program_Alarm" instruction in a function block. Create four parameters for the function block for interconnection of the instructions.

7		Static	
8		MyStaticText	String	'MyStaticText'
9		Program_Alarm_2	Program_Alarm	
10		Temp	
11		OUTPUT_Error	Bool	
12		OUTPUT_Status	Word	

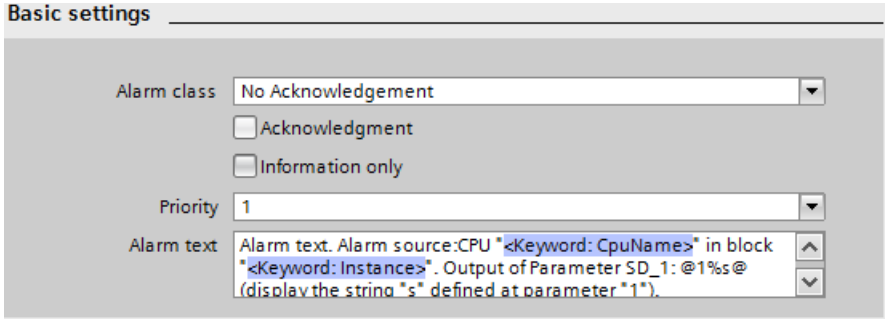
Interconnect the parameters of the instructions as follows.



Call the function block in an OB.

A PLC alarm is generated automatically when the "Program_Alarm" instruction is created. Open the PLC supervisions & alarms dialog and select the alarm to be edited in the Alarms > Program alarms tab. Create the alarm text, including two key words.

Select the following settings for the alarm.



If input parameter SIG ("CreateAlarm") of the "Program_Alarm" instruction has the signal state "TRUE", the PLC alarm is output.

In the case of the "Get_AlarmState" instruction, the following occurs: The PLC alarm is communicated to the "Get_AlarmState" instruction using input parameter ALARM. At output parameter AlarmState ("AS_statusBYTE"), it is indicated that the alarm corresponding to the alarm class "No Acknowledgement" is active.

At output parameters ERROR ("outputERROR") and STATUS ("outputSTATUS"), it is indicated that the processing of the instruction is proceeding error-free.

My_gDB_Get_AlarmState

		Name	Data type	Start value	Monitor value
1		Static			
2		AS_statusBYTE	Byte	16#0	16#85
3		outputERROR	Bool	false	FALSE
4		outputSTATUS	Word	16#0	16#0000
5		CreateAlarm	Bool	false	TRUE

For the output of the PLC alarm, use the web server of the CPU.

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568)

Example

ou can find a detailed application example in the Siemens Industry Online Support. (https://support.industry.siemens.com/cs/document/98210758?dti=0&lc=en-WW)

See also

Overview of the valid data types (Page 2239)

Program_Alarm: Generate program alarm with associated values (Page 3882)

Gen_UsrMsg: Generate user diagnostic alarms (S7-1200, S7-1500)

Description

You use the "Gen_UsrMsg" instruction to generate an alarm that is entered in the diagnostic buffer.

You use the Mode parameter to select whether an incoming or an outgoing alarm is to be generated:

- When Mode = 1: Create an incoming alarm.
- When Mode = 2: Create an outgoing alarm.
- Regardless of whether an incoming or outgoing alarm is generated, the alarm always has the attribute "Information only".

The entry in the diagnostic buffer is created synchronously. Alarm transmission is asynchronous.

If an error occurs during the execution of an instruction, it is output via the parameter RET_VAL.

Content of the alarm

The content of the alarm is defined with a text list:

- Specify the text list you want to use with the parameter TextListID. For this purpose, open the dialog "PLC alarm text lists" in the project navigation. Show the column "ID" in the dialog "Text lists". Apply the ID at the parameter TextListID.
- Use the parameter TextID to select the text list entry you want to write in the diagnostic buffer. For this purpose select an entry from the "Text lists entries" dialog by applying a number from the columns "Range from / range to" at the parameter TextID. The same number must be used from both the "Range from" and "Range to" columns for the text list entry.
- Additional information about text lists can be found here: Text lists (Page 417)

Defining associated values

New associated values to be added to the alarm are defined in the text list entry:

- Associated values are defined by adding the following information in the text list entry: @<No. of the associated value><Element type><Format specification>@
- Use the system data type AssocValues to specify which associated value is added when generating the alarm.
- Additional information on the structure of associated values can be found here: AUTOHOTSPOT

Parameters

The following table shows the parameters of the "Gen_UsrMsg" instruction:

Parameter	Declaration	Data type	Memory area	Description
Mode	Input	UInt	I, Q, M, D, L or constant	Parameters for selecting the status of the alarm: • 1: incoming alarm • 2: outgoing alarm
TextID	Input	UInt	I, Q, M, D, L or constant	ID of the text list entry that should be used for the alarm text.
TextListID	Input	UInt	I, Q, M, D, L or constant	ID of the text list that contains text list entry.
Ret_Val	Return	Int	I, Q, M, D, L	Error code of the instruction.
AssocValues	InOut	VARIANT	D, L	Pointer to the system data type AssocValues that allows you to define the associated values.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter AssocValues

Use the system data type AssocValues to define which associated values will be sent. A maximum of 8 associated values are possible. The structure of the system data type is created by entering the data type "AssocValues" as a data block.

The associated values are selected by entering the numbers of the associated values for the parameters Value[x]. Note the following:

- The values for TextID and TextListID are treated by the instruction as the associated values to be sent. As a result, "1" and "2" are already assigned as numbers for addressing associated values. Do not use the numbers "1" or "2" to address associated values.
- Address the associated value at parameter Value [1] as number "3", at parameter Value [2] as number "4", and so forth.

Byte	Parameter	Data type	Start value	Description	Number of the associated value
0..1	Value[1]	UINT	0	First associated value of the alarm.	3
2..3	Value[2]	UINT	0	Second associated value of the alarm.	4
4..5	Value[3]	UINT	0	...	5
6..7	Value[4]	UINT	0	...	6
8..9	Value[5]	UINT	0	...	7
10..11	Value[6]	UINT	0	...	8
12..13	Value[7]	UINT	0	...	9
14..15	Value[8]	UINT	0	Eighth associated value of the alarm.	10

Parameter RET_VAL

The following table contains all specific error information that is output via the RET_VAL parameter.

Error code* (W#16#...)	Explanation
0000	No error
8080	Value in the Mode parameter is not supported.
80C1	Resource bottleneck due to too many parallel calls.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

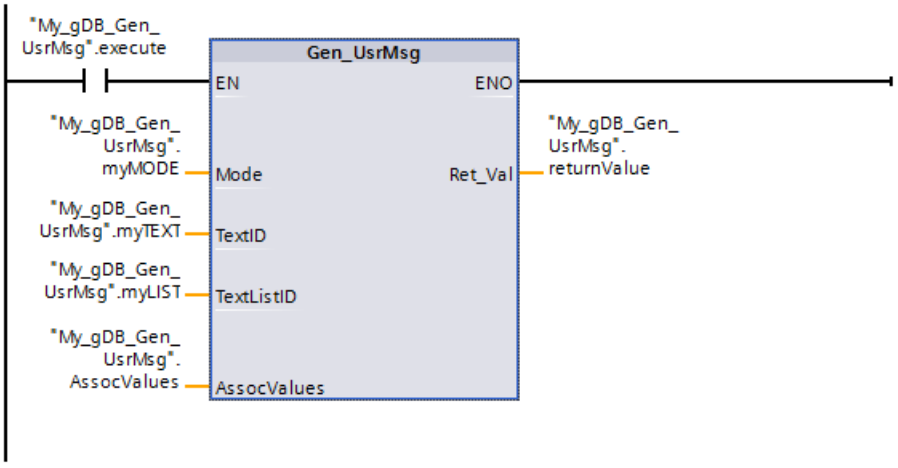
In the following example, you generate an alarm that is entered in the diagnostic buffer.

Create five tags and an "AssocValues" structure (with the AssocValues data type) in a global data block for storing the data.

My_gDB_Gen_UsrMsg

	Name	Data type	Start value
1	Static		
2	myMODE	UInt	1
3	myTEXT	UInt	3
4	myLIST	UInt	512
5	returnValue	Int	0
6	execute	Bool	false
7	AssocValues	AssocValues	
8	Value	Array[1..8] of UInt	
9	Value[1]	UInt	33
10	Value[2]	UInt	411
11	Value[3]	UInt	578
12	Value[4]	UInt	6122
13	Value[5]	UInt	722
14	Value[6]	UInt	8111
15	Value[7]	UInt	9829
16	Value[8]	UInt	10457

Interconnect the parameters of the instruction as follows.



Use the "Text lists" entry to create a Text list and a Text list entry for the alarm. Apply the ID of the text list at parameter TextListID ("myLIST"). Apply the ID (Range ...) of the text list entry at parameter TextID ("myTEXT"). Assign the alarm parameters as follows.

Text lists

Name	Id	Selection	Comment
USER_1	512	Decimal	

Text list entries of USER_1

Range from	Range to	Entry
3	3	This is a user generated message. Return Value[3]: @5I%6d@

If the normally open contact ("execute") supplies the signal state "TRUE", the "Gen_UsrMsg" instruction is executed. According to the value of parameter Mode ("myMODE"), an incoming

alarm is generated. The alarm to be output is made known to the instruction using parameters TextListID ("myLIST") and TextID ("myTEXT"). The associated values of the alarm are transferred using parameter AssocValues ("AssocValues").

When the alarm is generated, the character string "@5I%6d@" contained in the alarm text is interpreted as follows:

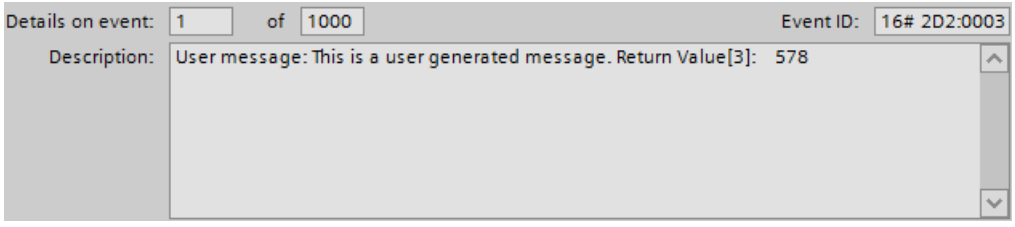
- The associated value with the number "5" is read out in the INT data type. The number corresponds to parameter Value[3] of the "AssocValues" structure.
- The associated value is output as a decimal number. The decimal number is limited to six places.

At output parameter Ret_Val ("returnValue"), it is indicated that the processing of the instruction was error-free.

My_gDB_Gen_UsrMsg

	Name	Data type	Start value	Monitor value
1	Static			
2	myMODE	UInt	1	1
3	myTEXT	UInt	3	3
4	myLIST	UInt	512	512
5	returnValue	Int	0	0
6	execute	Bool	false	FALSE
7	AssocValues	AssocValues		
8	Value	Array[1..8] of UInt		
9	Value[1]	UInt	33	33
10	Value[2]	UInt	411	411
11	Value[3]	UInt	578	578
12	Value[4]	UInt	6122	6122
13	Value[5]	UInt	722	722
14	Value[6]	UInt	8111	8111
15	Value[7]	UInt	9829	9829
16	Value[8]	UInt	10457	10457

For the output of the alarm, for a CPU of the S7-1500 series, open the entry "Online & Diagnostics > Diagnostic buffer".



You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Example

You will find a detailed application example in the Siemens Industry Online Support (https://support.industry.siemens.com/cs/document/98210758?dti=0&lc=en-WW).

Get_Alarm: Read pending alarm (S7-1500)

Description

Using the "Read pending alarms" instruction, you can log on for alarms generated in the user program and copy an incoming or outgoing alarm to a referenced data block. This allows you to pass on for example alarms as a STRING to a master computer.

When you call the instruction with the "OperateMode = 1" and log on with this, you have to specify at least one or more display classes at the input "DispClassNr". After logging on no more alarms are lost either incoming or outgoing.

The alarms are processed sequentially, starting with the oldest alarm. They are read out and stored in a data block in the CPU. You can then forward the data, e.g. using the instruction "TSEND_C", to a master computer.

The instruction operates asynchronously and therefore avoids long runtimes of the function block "Get_Alarm [FB702]" that could lead to a cycle overflow. The instruction must be called recurrently in a cycle organization block and in a period of 100 ms up to maximum 1 second; otherwise, the instruction is logged off after 3 minutes. As a result of the logoff, no more alarms can be read out.

In the following cases, you need to log on the instruction again using the input "OperateMode = 1":

- After automatic logoff because the function block "Get_Alarm [FB702]" is called too rarely.
- When the operating mode changes from "STOP" > "RUN".
- As soon as you change existing alarm configurations, such as the Program_Alarm or ProDiag, the instruction automatically logs off.

You can use the instruction for alarms from the following areas:

- ProDiag
- System diagnostics
- GRAPH
- Program_Alarm
- Motion Control

Note

CPU property "Central alarm management in the PLC"

If you do not enable the CPU property "Central alarm management in the PLC" in the device configuration, only the IDs of the alarm text fields are output instead of the alarm texts (alarm text, information text and additional alarm texts 1 - 9).

Parameter

The following table shows the parameters of the instruction Read pending alarm":

Parameter	Declaration	Data type	Memory area	Description
EN *)	Input	BOOL	I, Q, M, D, L	Enable input
ENO *)	Output	BOOL	I, Q, M, D, L	Enable output
Operate-Mode	Input	BYTE	I, Q, M, D, L	There are three possible values: • 1: Logon • 2: Read out data • 3: Logoff
DataMode	Input	BYTE	I, Q, M, D, L	There are four different possible values: • 0: Only the column titles are read out (Producer-ID, ID_1, ID_2, Prio, Timestamp, State) • 1: The column headers and the alarm text is read out including associated values. • 2: The column headers, the alarm text including associated values and the information text are read out. • 3: The column headers, the alarm text including associated values, the information text and the additional alarm texts are read out.
DispClassNr (Display class number)	Input	ARRAY[0..31] of UINT	I, Q, M, D, L	You can define up to 32 different display classes: • 0 ... 16: User-defined display classes that can be set on the HMI device • 257: System diagnostics • 259: Security If you do not use the parameter, all display classes are reported. A change of the display class is only taken into consideration if mode 1 is set at the parameter mode.
Lcid (Local ID)	Input	UDINT	I, Q, M, D, L	There are different possible values: • 0: All languages • 1: First language with the lowest LC number • 2: First language plus second language with the second lowest LC number • 3: All languages • or: Enter the exact LC number to select a specific language.
Data	InOut	"AlarmData"	I, Q, M, D, L	Target range for the alarm data that was read. Alarm text structure from the PLC data type "AlarmData" according to the value at the "DataMode" parameter.

Parameter	Declaration	Data type	Memory area	Description
Data-Ready	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: No data is available. • 1: The data is available. The data is valid until the next call of the instruction. Therefore, it must be processed in the same program cycle, e.g. using a move instruction.
Busy	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: The execution of the instruction has not yet been started or the asynchronous processing was completed. Additional information on the processing status can be found in the STATUS parameter. • 1: Execution of the instruction not yet complete Further entries exist
Error	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: No error • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
Init	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: No action necessary. • 1: A new logon is required. The alarm data received so far are no longer valid. This may be necessary in the following situations: • You have logged on. • The CPU changes from "STOP" mode to "RUN" mode. • Alarms were re-configured and the changes were made by means of "Download to device" in "RUN" mode, for example, a change in the alarm text structure.
Status	Output	WORD	I, Q, M, D, L	Error and status information
*): The enable input EN and the enable output ENO are only visible in the programming languages LAD and FBD.				

Parameter Status

The following table shows the meaning of the values of the Status parameter:

Status code* (W#16#...)	Explanation
0000	No error
0001	New data is available.
8001	The value at parameter OperateMode is invalid.
8002	The Data parameter has the wrong data type.
8003	The Lcid parameter is invalid or a language selected with the Lcid parameter is not loaded.
8004	You have already logged on via the OperateMode = 1 parameter in the user program to read the pending messages. You are only allowed and required to log on again if you log out using the OperateMode = 3, have re-configured alarm or performed a STOP > RUN transition.
8005	The value at parameter DataMode is invalid.
8085	• You did not specify a display class when you logged on. • The instruction was logged off during the reconfiguration of supervisions (ProDiag). • Automatic logoff because the function block "Get_Alarm [FB702]" is called too rarely.
80C5	The processing of the instruction is not yet completed. Please try again later.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

PLC data type "AlarmData"

The PLC data type "AlarmData" reflects the structure of an alarm.

Name	Data type	Default value	Description
ProducerID	UINT	0	There are different possible values: • 0: Program_Alarm • 4: System diagnostics • 5: Standard Motion Control (T-CPU) • 6: Security (for example, false login) • 7: SINUMERIK • 8: GRAPH • 9: ProDiag
ID_1	UINT	0	Alarm ID This ID is also displayed in the alarm editor.
ID_2	UINT	0	ID of the alarm runtime instance This ID is an internal range ID that is significant for ProDiag and sequencers: • In ProDiag, it is the supervision ID. • In a sequencer, it is the step number. For the "Program_Alarm" instruction the ID is 0, and it has no significance for system diagnostics.

Name	Data type	Default value	Description
PRIO	UINT	0	Priorities: 0 to 16
TimeStamp	LDT	DT#1970-01-01-0:0:0.000000000	Time stamp
State	BYTE	16#0	Status parameter: • 0: Outgoing • 1: Incoming
LC_ID	ARRAY[0..2] of UINT	-	Returns the value that is specified at the input "Lcid": • 0: All languages • 1: First language with the lowest LC number • 2: First language plus second language with the second lowest LC number • 3: All languages or: Enter the exact LC number to select a specific language.
AlarmText	ARRAY[0..2] of WSTRING	-	Alarm text with up to 254 characters (508 bytes). If the text is longer, it is truncated at 251 characters, for example WSTRING#'Meld... .
InfoText	ARRAY[0..2] of WSTRING	-	Information text
AddText_1 - 9	ARRAY[0..2] of WSTRING	-	Additional text 1 - 9

Example

You can find the example here: Program example for Get_Alarm & Ack_Alarms - Part 1 (Page 3903).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Overview of the valid data types (Page 2239)

Ack_Alarms: Acknowledge alarms (S7-1500)

Description

You can use the "Acknowledge alarms" instruction to acknowledge all incoming and outgoing alarms in your program code with a single command.

The instruction is called as soon as there is a positive signal edge either at the enable input EN (LAD / FBD) or as a preceding logic operation (STL / SCL), and the value = 1 is set at the MODE parameter. The instruction operates asynchronously and therefore avoids long runtimes of the function block "Ack_Alarms [FB701]" that could lead to a cycle overflow. The

CPU acknowledges all incoming and outgoing alarms and informs all HMI devices about this acknowledgment. Any pending alarms on the displays disappear as soon as they "went out".

In each case, only one acknowledgment (max. 100 alarms requiring acknowledgment) can be carried out at the same time. If additional alarms are pending, the next 100 alarms requiring acknowledgment will be confirmed automatically.

The instruction can also be used for ProDiag supervision alarms.

Parameters

The following table shows the parameters of the instruction "Acknowledge alarms":

Parameter	Declaration	Data type	Memory area	Description
EN *)	Input	BOOL	I, Q, M, D, L	Enable input
ENO *)	Output	BOOL	I, Q, M, D, L	Enable output
MODE	Input	BYTE	I, Q, M, D, L	There are two different possible values: • 0: The instruction is not executed. • 1: Confirm all alarms. All active alarms that can be confirmed are confirmed.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: No error • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Error and status information
*): The enable input EN and the enable output ENO are only visible in the programming languages LAD and FBD.				

Parameter STATUS

The following table shows the meaning of the values of the STATUS parameter:

Status code* (W#16#...)	Explanation
0000	No error
7001	No error occurred and the next 100 alarms requiring acknowledgment will be confirmed.
8001	The value at parameter MODE is invalid.
80C5	A confirmation process is already running. Please try again later.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Program example for Get_Alarm & Ack_Alarms - Part 1 (Page 3903).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Overview of the valid data types (Page 2239)

Program example for Get_Alarm & Ack_Alarms (S7-1200, S7-1500)

Program example for Get_Alarm & Ack_Alarms - Part 1 (S7-1200, S7-1500)

Introduction

In the following example, you collect all alarms of the CPU 1 (S7-1500 as of firmware V2.0) and save the alarms. You then use a programmed connection to the CPU 2 (S7-1500) to send the alarms (in abbreviated form; in PLC data type "rcvGetAlarm") from CPU 1 to CPU 2. You also save the alarms on CPU 2.

In the first part of the programming example for Get_Alarm and Ack_Alarms, you deal with the requirements and interconnection of the parameters.

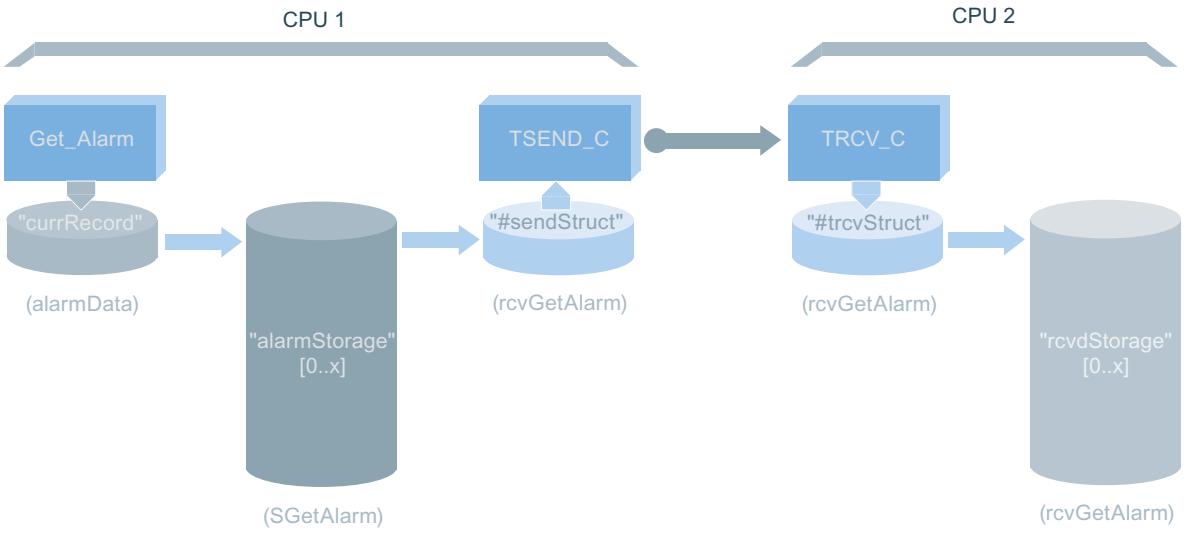
The results and the procedure for the connection are described here: Program example for Get_Alarm & Ack_Alarms - Part 2 (Page 3921).

Note

Size of the memory arrays for the alarms

The memory arrays for the alarms ("alarmStorage" and "rcvdStorage") can hold up to 16 alarms when configured as described.

If you want to change the memory size, you must change the array size and the maximum value "maxPosition".



Requirements for a programmed connection

You create two CPUs (S7-1500) and connect the CPUs with each other over PROFINET. You do not assign parameters to the connection.

Note

Using Get_Alarm and Ack_Alarms

The CPU 1 (for the call of Get_Alarm and Ack_Alarms) must have firmware v2.0 or higher. In addition, the CPU property "Central alarm management in the PLC (Page 3897)" must be activated for the CPU 1.

Note

Permitting read and write access

By setting a low protection level under Properties > Protection of the CPU, you ensure that read and write access is permitted.

CPU 1: Program requirements

Creating PLC data types

You need the following PLC data types for the transfer and storage of alarms:

- You need "SGetAlarm" to save the alarms, including information about the current read loss

SGetAlarm

	Name	Data type	Default value
1	countReadLoss	UDInt	0
2	alarm	AlarmData	

- You need "rcvGetAlarm" to transfer the alarms over TSEND_C.

rcvGetAlarm

	Name	Data type	Default value
1	countReadLoss	UDInt	0
2	ProducerID	UInt	0
3	ID_1	UInt	0
4	ID_2	UInt	0
5	PRIO	UInt	0
6	TimeStamp	LDT	LDT#1970-01-01-(
7	State	Byte	16#0
8	LCID	Array[0..2] of UInt	
9	AlarmText	WString[254]	WSTRING#''

Storage of data

Create the following structures and tags for storing the data in a global data block ("SLI_gDB_GAalarm").

- For parameter assignment of TSEND_C:

SLI_gDB_GAalarm

	Name	Data type	Start value
1	Static		
2	TSEND_C	Struct	
3	start	Bool	false
4	stop	Bool	false
5	comControl	Bool	false
6	done	Bool	false
7	busy	Bool	false
8	error	Bool	false
9	status	Word	16#0
10	memErrStatus	Word	16#0

- For parameter assignment of Get_Alarm:

SLI_gDB_GAalarm

	Name	Data type	Start value
11	GetAlarm	Struct	
12	start	Bool	false
13	stop	Bool	false
14	check	Struct	
15	anew	Bool	false
16	operateMode	Struct	
17	value	Byte	16#0
18	Logon_done	Bool	false
19	ReadData_ready	Bool	false
20	ReadData_active	Bool	false
21	Logoff_done	Bool	false
22	outputMode	Byte	16#3
23	languageCodeID	UDInt	0
24	currRecord	AlarmData	
25	currReadLoss	UDInt	0
26	dataReady	Bool	false
27	initiateLogin	Bool	false
28	busy	Bool	false
29	status	Word	16#0
30	memError	Bool	false
31	memErrStatus	Word	16#0

- As part of the "GetAlarm" structure for saving the alarms in the memory array:

32	maxPosition	UInt	15
33	alarmStorage	Array[0..15] of "SGetAlarm"	
34	limitReach	Bool	false
35	savingDone	Bool	false
36	initiateSending	Bool	false
37	countAlarms	UInt	0
38	posForAlarm	UInt	0

- For parameter assignment of Ack_Alarms:

40	AckAlarm	Struct	
41	executeAck	Bool	false
42	error	Bool	false
43	status	Word	16#0

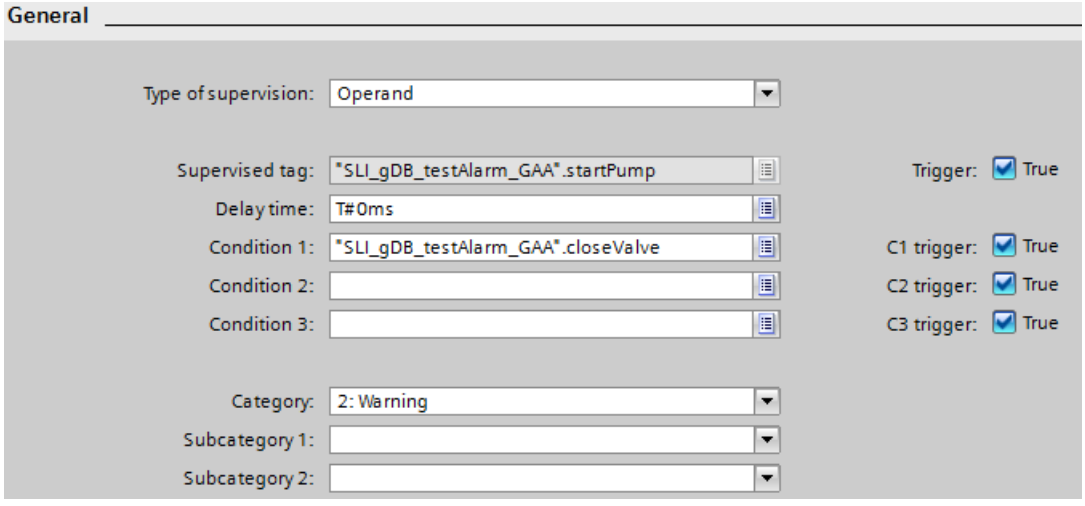
ProDiag alarm

Create three tags in a global data block ("SLI_gDB_testAlarm_GAA") for a test scenario.

SLI_gDB_testAlarm_GAA

	Name	Data type	Start value
1	Static		
2	startPump	Bool	false
3	closeValve	Bool	false
4	valveOut	Bool	false

Create the following rule in the global data block ("SLI_gDB_testAlarm_GAA") to trigger a ProDiag alarm. To do so, open the global data block > the properties of the tag "startPump" > Supervisions > Add new supervision.



CPU 2: Program requirements

Creating PLC data type

Create the PLC data type "rcvGetAlarm" as described for CPU 1.

Storage of data

Create the following structures and tags for storing the data in a global data block ("SLI_gDB_trcv_GAA").

SLI_gDB_trcv_GAA

	Name	Data type	Start value
1	Static		
2	TRCV_C	Struct	
3	start	Bool	false
4	comControl	Bool	false
5	done	Bool	false
6	busy	Bool	false
7	error	Bool	false
8	status	Word	16#0
9	memErrStatus	Word	16#0
10	rcvLen	UDInt	0
11	dataReceived	Bool	false
12	GetAlarm	Struct	
13	maxPosition	UInt	15
14	rcvdStorage	Array[0..15] of "rcvGetAlarm"	
15	limitReach	Bool	false
16	savingDone	Bool	false
17	countAlarms	UInt	0
18	posForAlarm	UInt	0

CPU 1: FB "SLI_FB_send_GAA" - Interconnect parameters

Create the FB "SLI_FB_send_GAA" to set up the connection between CPU 1 and CPU 2 and data transfer.

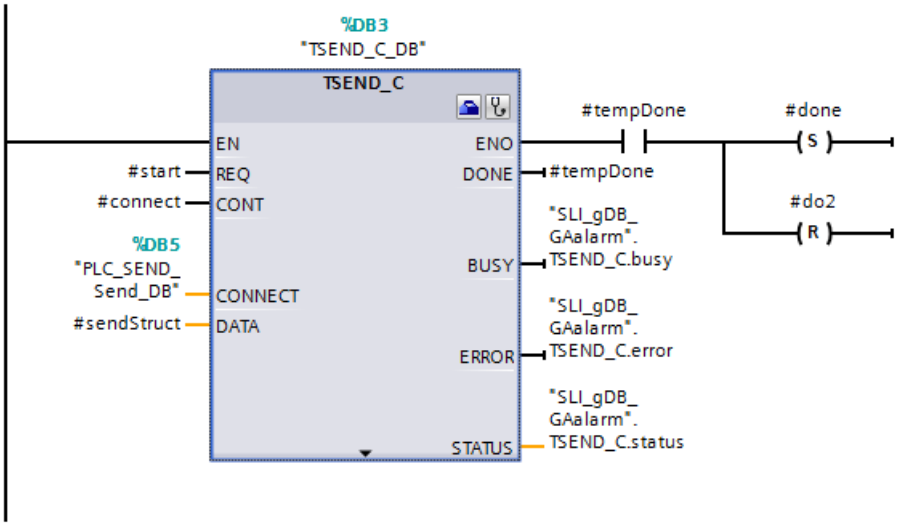
Create the following local tags in the FB.

SLI_FB_send_GAA

	Name	Data type	Default value	Retain
1	Input			
2	start	Bool	false	Non-ret...
3	connect	Bool	false	Non-retain
4	Output			
5	<Add new>			
6	InOut			
7	done	Bool	false	Non-retain
8	Static			
9	posForAlarm	UInt	0	Set in IDB
10	sendStruct	"rcvGetAlarm"		Non-retain
11	initiateSending	Bool	false	Set in IDB
12	do	Bool	false	Non-retain
13	do2	Bool	false	Non-retain
14	Temp			
15	tempDone	Bool		
16	Constant			
17	<Add new>			

In the FB, you use:

- Interconnections for canceling ("TSEND_C.stop") the data transfer.
- Interconnections for starting ("#do2") the data transfer.
- Interconnections to initiate a new data transfer ("#initiateSending", "#do").
- The instruction "TSEND_C" for connection establishment and data transfer.



Note

Interconnection of the input parameter CONNECT of the "TSEND_C" instruction

To interconnect the input parameter CONNECT, open the wizard of the "TSEND_C" instruction with "Properties > Configuration".

Make the following settings for the connection of TSEND_C:

- End point Select the CPUs using the drop-down list. Interface, subnet and address are entered automatically.
- Connection data: Use the drop-down list and the selection "New" to create one data block per CPU. This data block is required for storing the connection data. The name of the data block can be freely selected for each CPU (communication partner).
- Select the connection type "TCP" and the configuration type "Use program block". These settings set up an Ethernet connection with the "TCP" protocol and as programmed connection.
- You enter a freely selected connection ID. Use a connection ID that has not already been assigned.
- You enter a value for the partner port. The value must be >=2000.

Additional information on the configuration of TSEND_C (Page 4423) is available here: Program example for send functions (Page 4446).

Additional information on setup and interconnections of the FB "SLI_FB_send_GAA" is available in the program code of the example in the Sample Library for Instructions (Page 2568).

CPU 1: FC "SLI_FC_switchMode_GAA" - Interconnect parameters

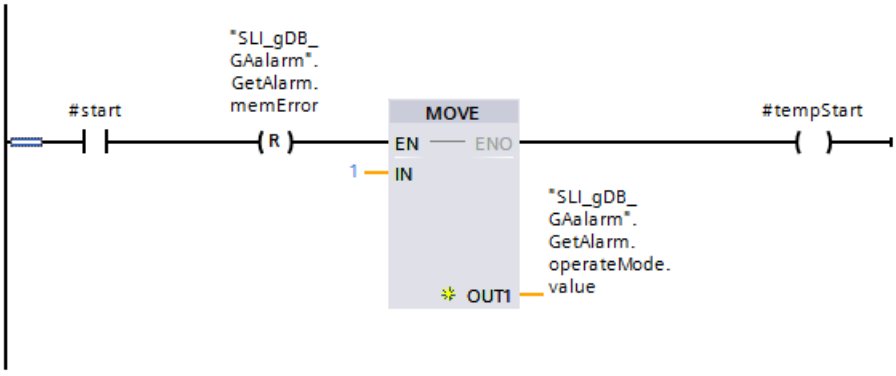
You use the FC "SLI_FC_switchMode_GAA" to control the "Get_Alarm" instruction.

Create the following local tags in the FC.

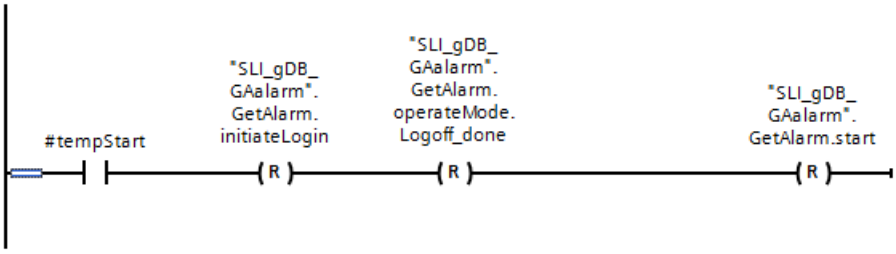
SLI_FC_switchMode_GAA		
	Name	Data type
1	Input	
2	start	Bool
3	stop	Bool
4	Output	
5	<Add new>	
6	InOut	
7	mode2Reinit	Bool
8	Temp	
9	tempStart	Bool
10	tempStop	Bool
11	Constant	
12	<Add new>	
13	Return	
14	SLI_FC_switchMode_GAA	Void

Create the following interconnections in the FC.

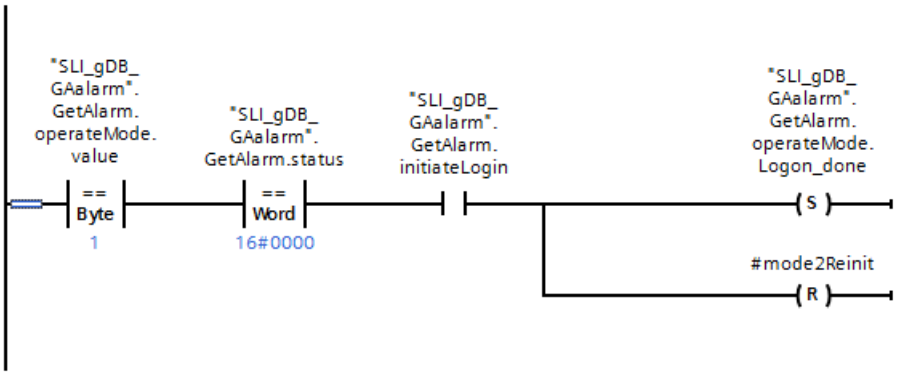
Network 1: To start the processing of Get_Alarm, create the following interconnections.



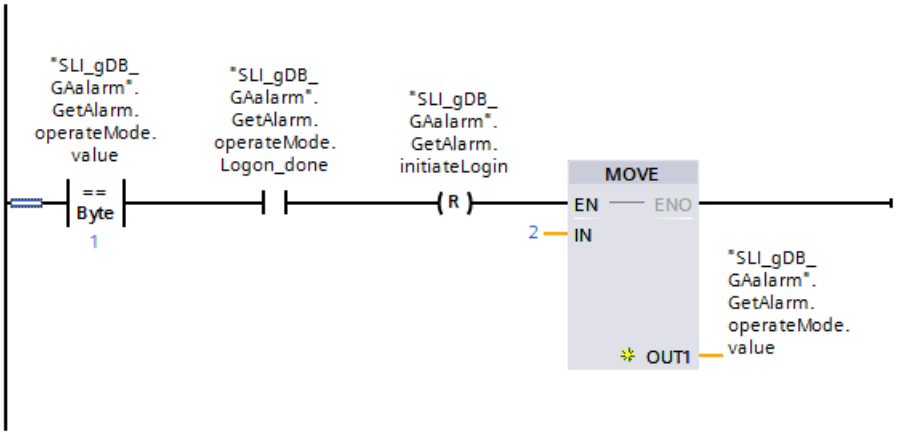
Network 2: To start the processing of Get_Alarm, create the following interconnections (Part 2).



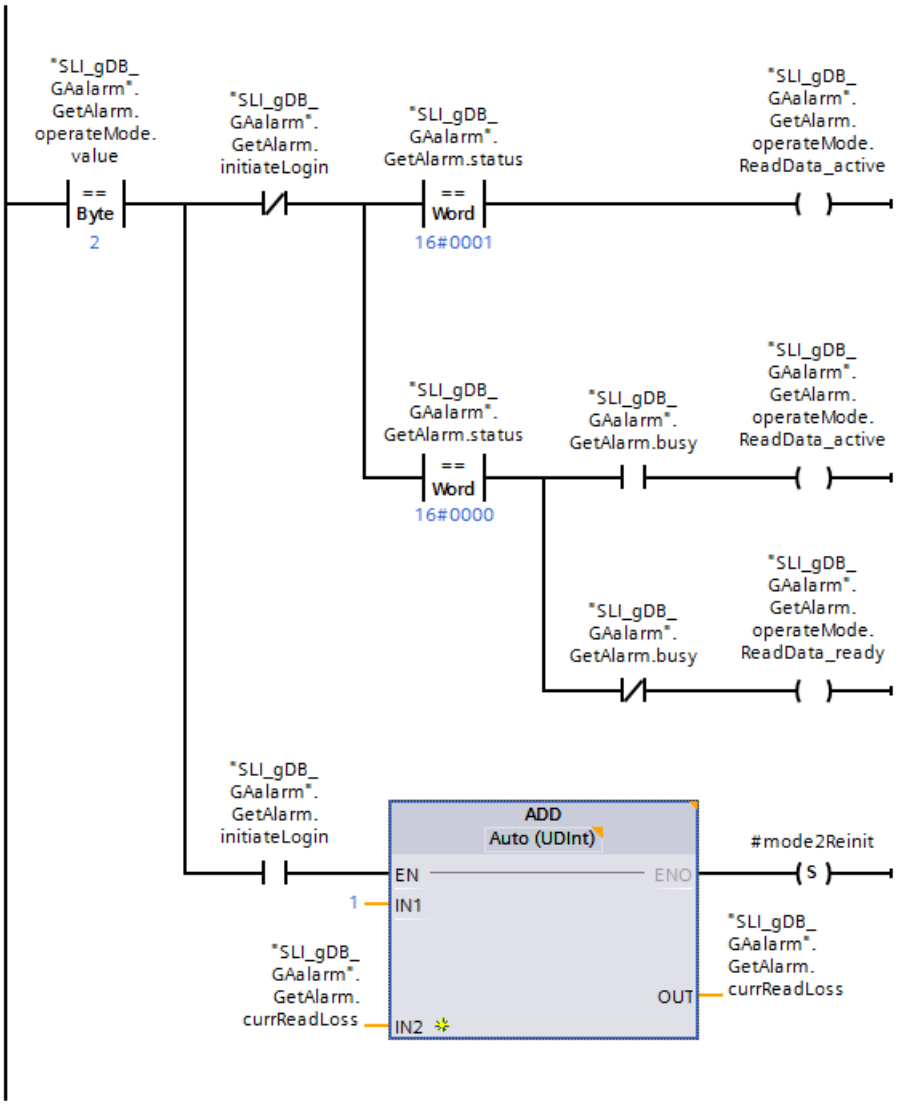
Network 3: For login of Get_Alarm, create the following interconnections.



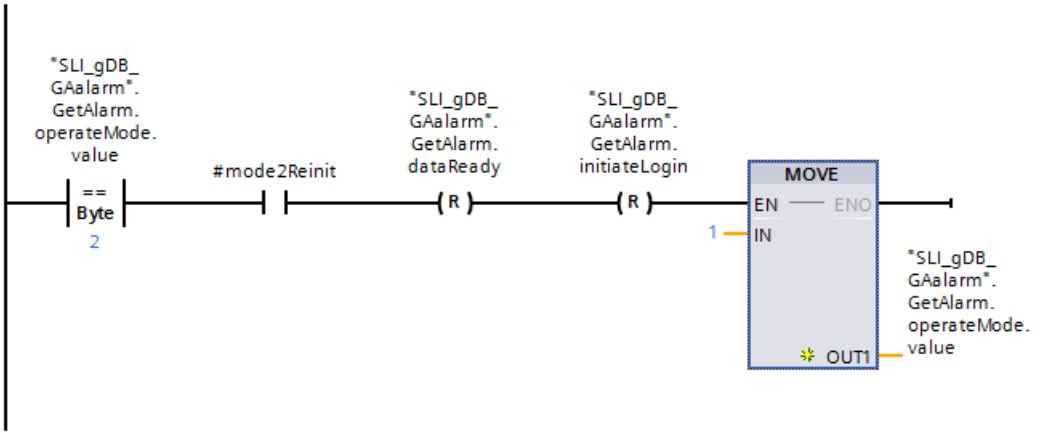
Network 4: To switch to the "Read" mode of Get_Alarm, create the following interconnections.



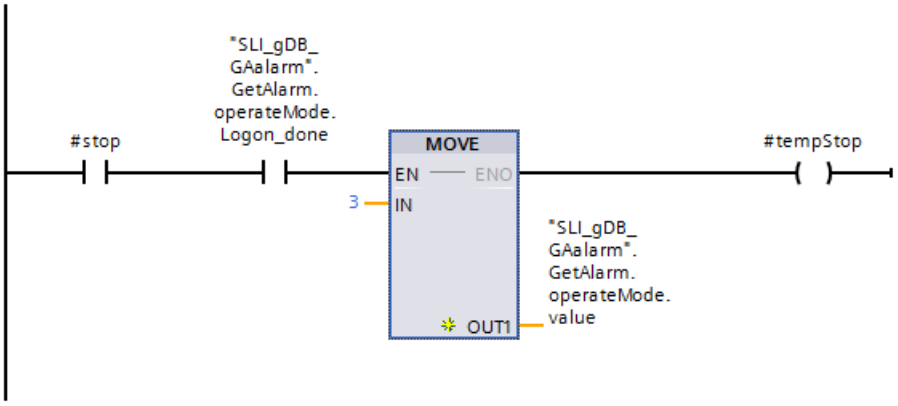
Network 5: To process the "Read" mode of Get_Alarm, create the following interconnections.



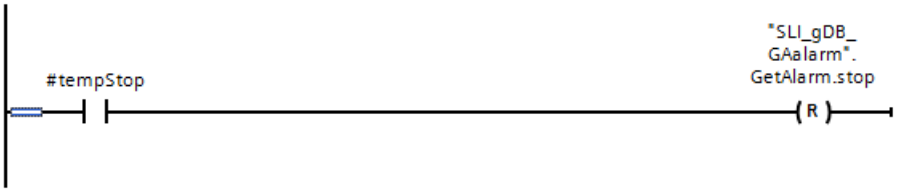
Network 6: To process the "Read" mode of Get_Alarm, create the following interconnections (Part 2).



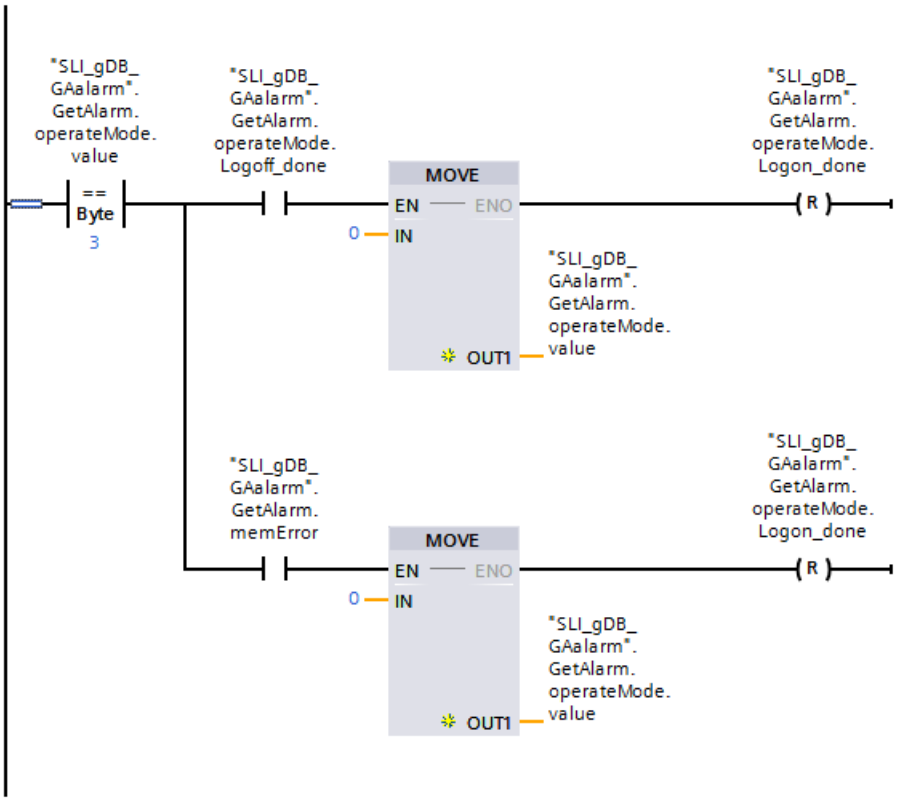
Network 7: To end the processing of Get_Alarm, create the following interconnections.



Network 8: To end the processing of Get_Alarm, create the following interconnections (Part 2).



Network 9: To end the processing of Get_Alarm, create the following interconnections (Part 3).



CPU 1: FB "SLI_FB_saveAlarms_GAA" - Interconnect parameters

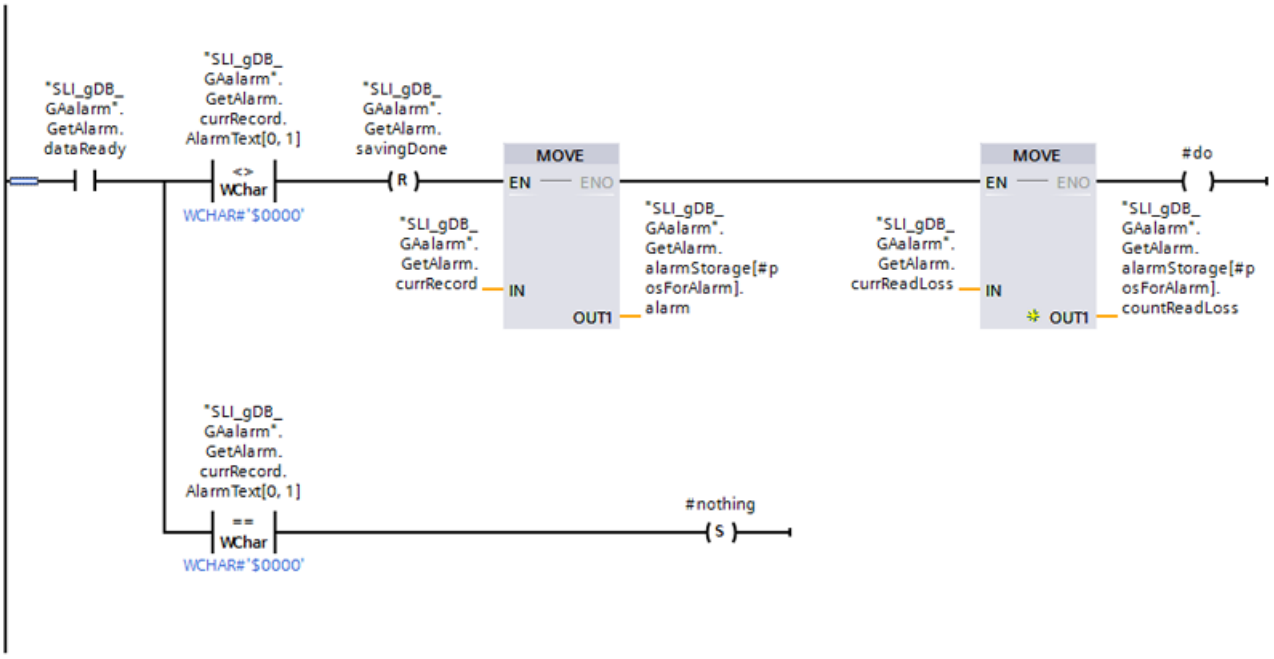
You create the FB "SLI_FB_saveAlarms_GAA" to copy the alarms to the memory array "GetAlarm.alarmStorage".

Create the following local tags in the FB.

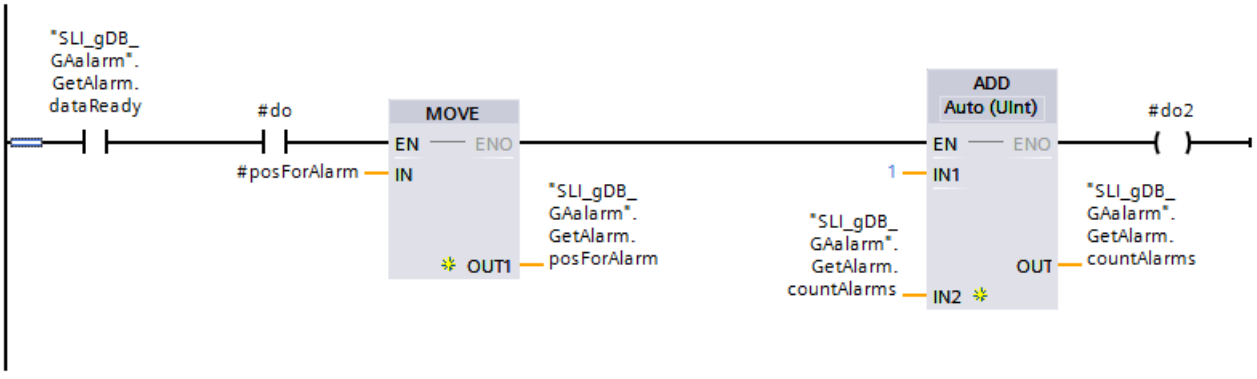
7		Static			
8		posForAlarm	UInt	0	Set in IDB
9		do	Bool	false	Set in IDB
10		do2	Bool	false	Set in IDB
11		nothing	Bool	false	Set in IDB
12		newRound	Bool	false	Set in IDB
13		dbComplete	Bool	false	Set in IDB
14		allDone	Bool	false	Set in IDB

Create the following interconnections in the FB.

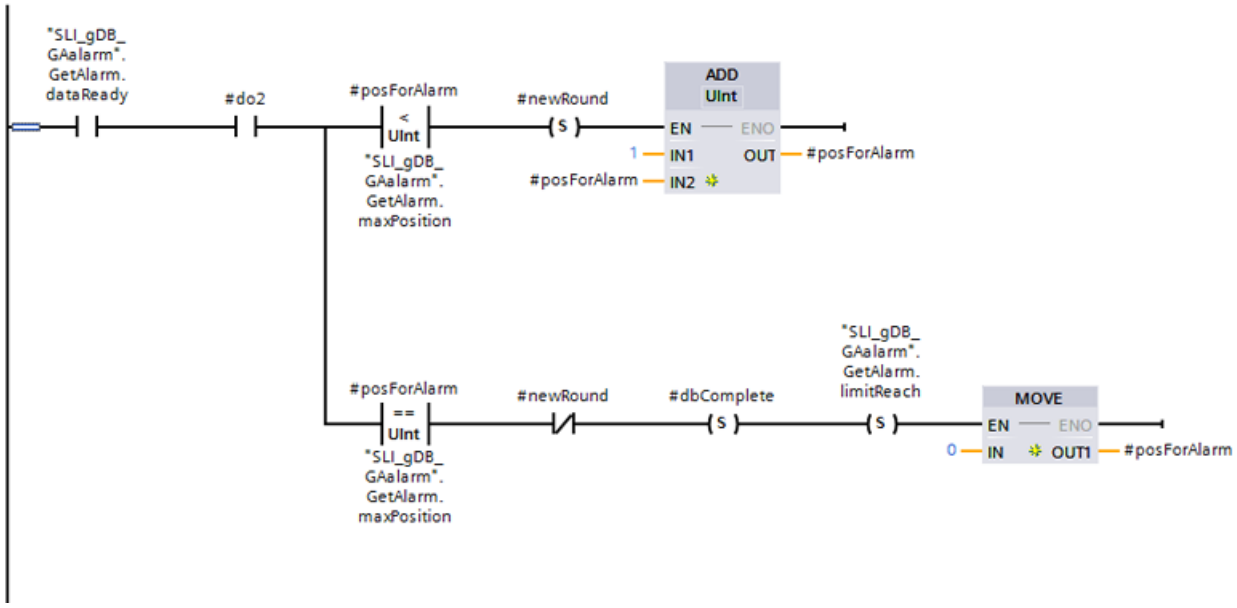
Network 1: The copying of alarms to the memory array "GetAlarm.alarmStorage" is decided as follows:



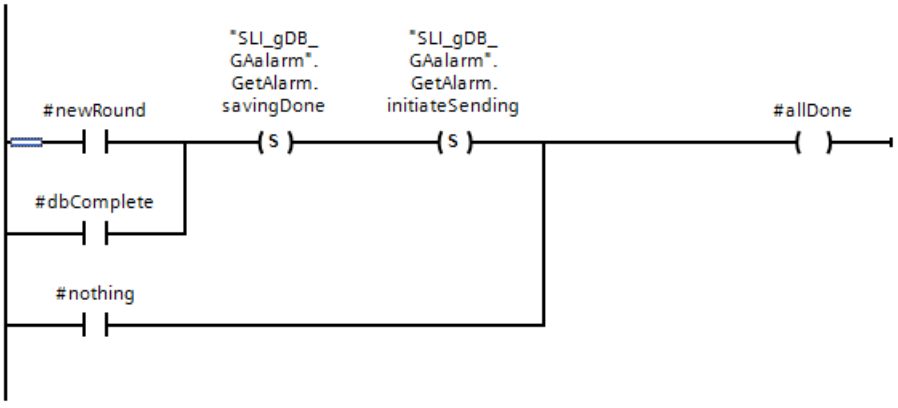
Network 2: To save the cell number of the memory array used last and to count the alarms, create the following interconnections.



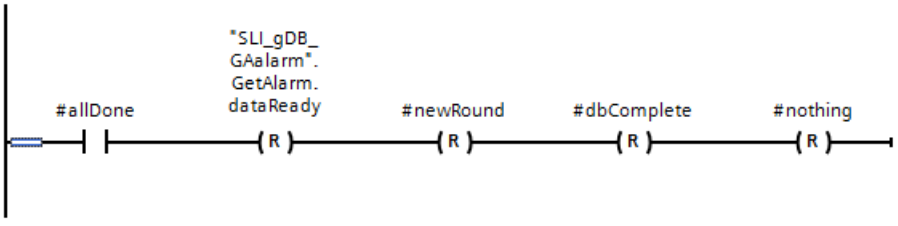
Network 3: Incrementing or resetting the cell number of the memory array is decided as follows:



Network 4: To save the success when copying to the memory array, create the following interconnections.



Network 5: To end the processing of SLI_FB_saveAlarms_GAA, create the following interconnections.



CPU 1: FB "SLI_FBCall_GetAlarm_AckA" - Interconnect parameters

You create the FB "SLI_FBCall_GetAlarm_AckA" for the following purposes:

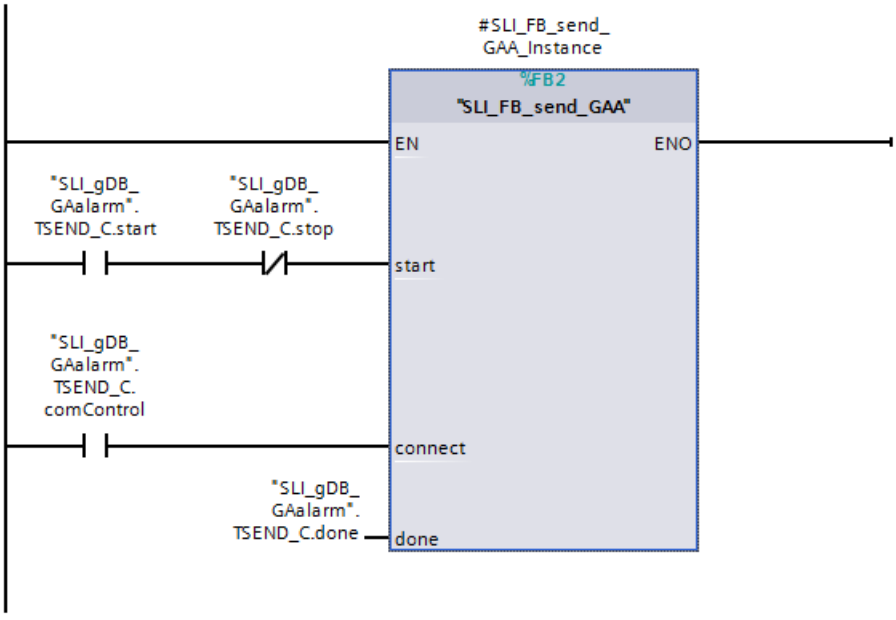
- To call and modify Get_Alarm
- To call the blocks for copying and send processes.
- To call Ack_Alarms.

Create the following local tags in the FB.

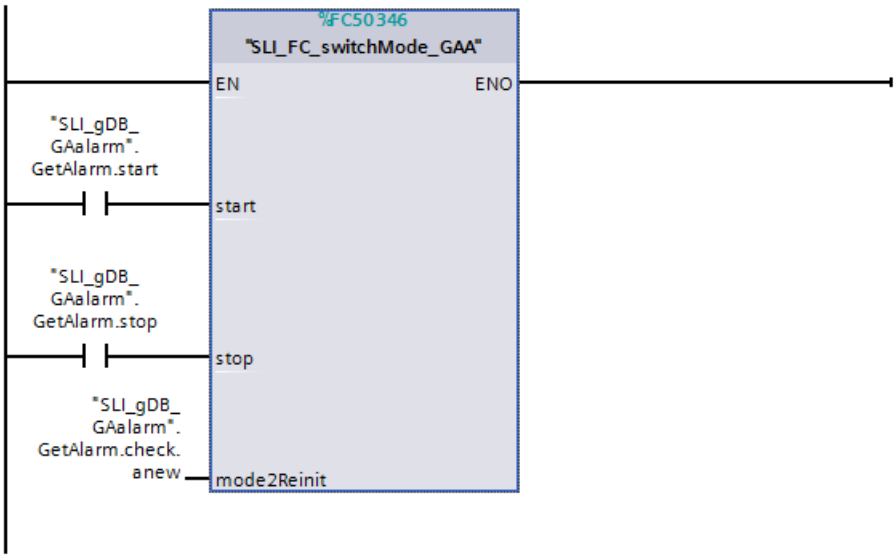
7		Static		
8		acknowledge	UInt	1	Non-ret...
9		dataReady	Bool	false	Non-retain
10		initiate	Bool	false	Non-retain
11		error	Bool	false	Non-retain
12		SLI_FB_send_GAA_Instance	"SLI_FB_send_GAA"		

Create the following interconnections in the FB.

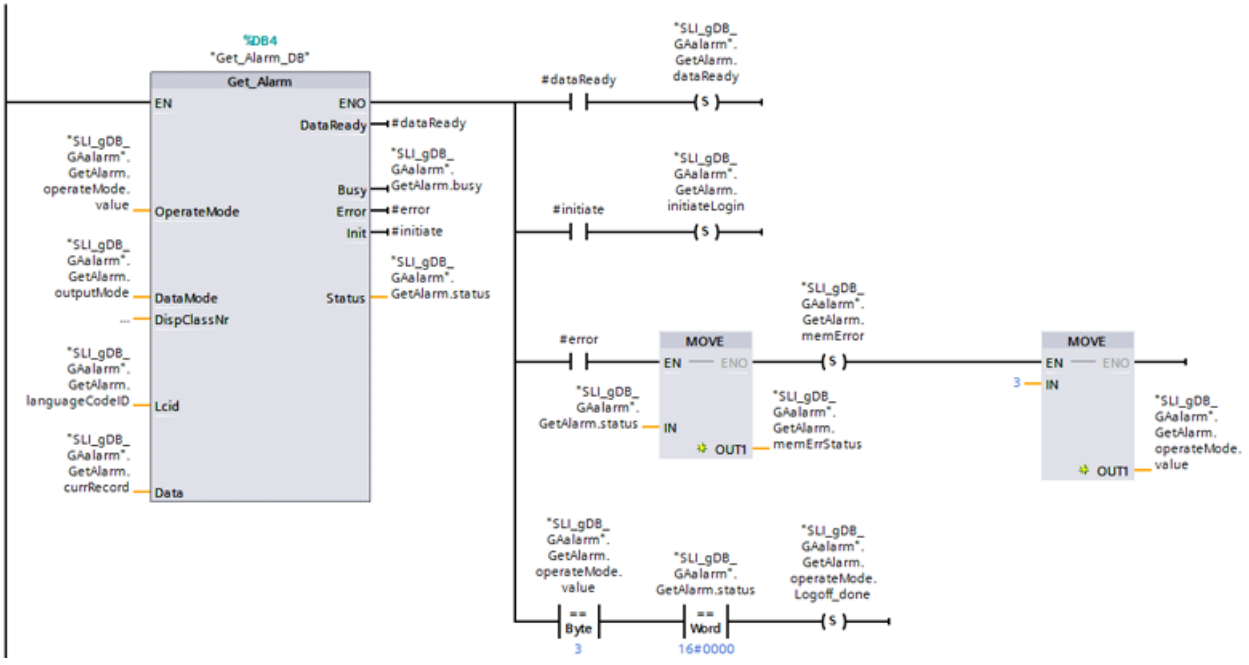
Network 1: Interconnect the parameters of the "SLI_FB_send_GAA" function block as follows.



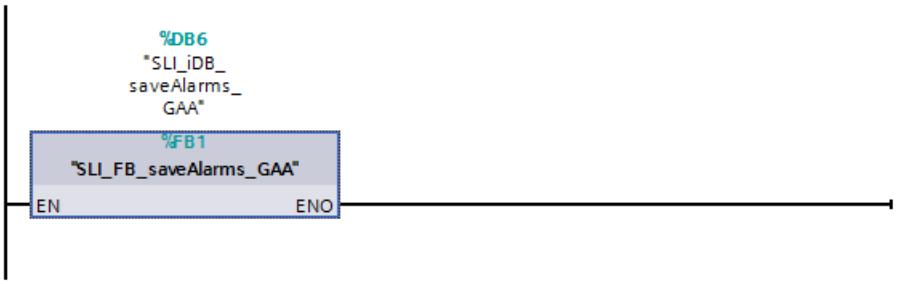
Network 2: Interconnect the parameters of the "SLI_FC_switchMode_GAA" FC as follows.



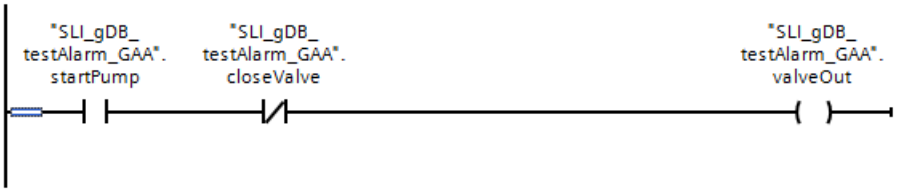
Network 3: Interconnect the parameters of the "Get_Alarm" instruction as follows.



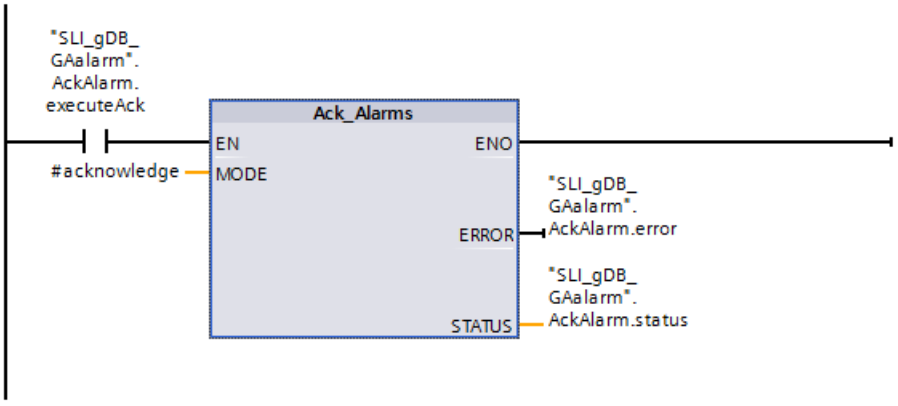
Network 4: Interconnect the parameters of the "SLI_FB_saveAlarms_GAA" function block as follows.



Network 5: Create the following interconnections to test the ProDiag alarms.



Network 6: Interconnect the parameters of the "Ack_Alarms" instruction as follows.



CPU 2: FB "SLI_FB_rcvdAlarms_GAA" - Interconnect parameters

You create the FB "SLI_FB_rcvdAlarms_GAA" to copy the alarms to the memory array "GetAlarm.rcvdStorage". The setup and interconnections are similar to those of the FB "SLI_FB_saveAlarms_GAA".

Additional information on setup and interconnections of the FB "SLI_FB_rcvdAlarms_GAA" is available in the program code of the example in the Sample Library for Instructions (Page 2568).

CPU 2: FB "SLI_FBCall_trcv_GAA" - Interconnect parameters

You create the FB "SLI_FBCall_trcv_GAA" to call and process TRCV_C as well as all subsequent processes.

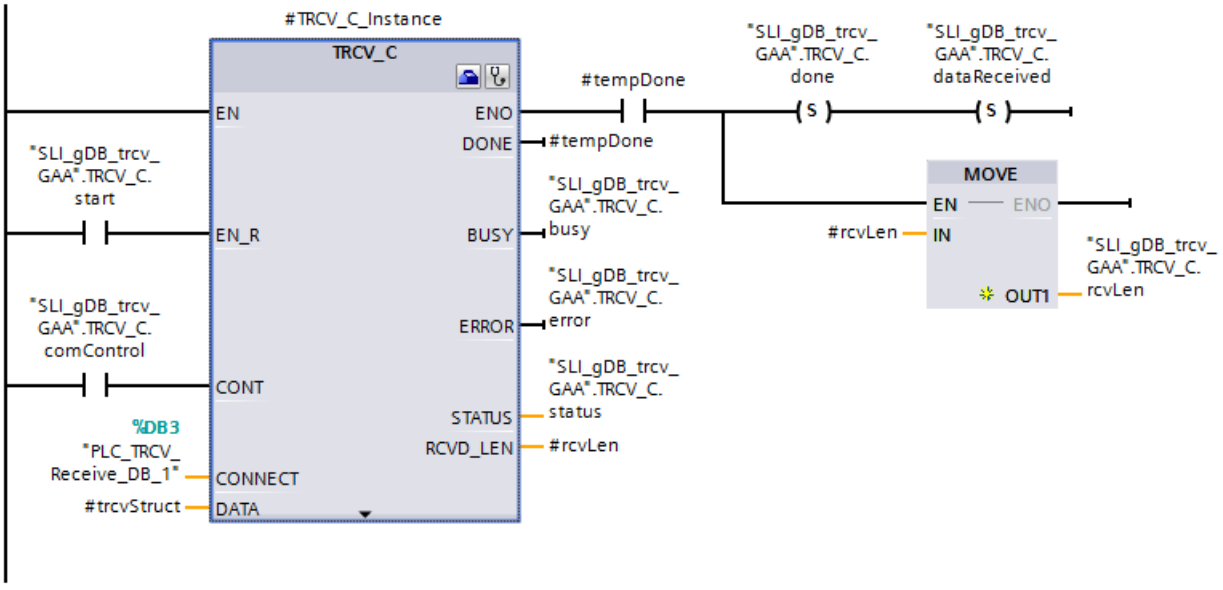
Create the following local tags in the FB.

7		Static		
8		TRCV_C_Instance	TRCV_C	
9		trcvStruct	"rcvGetAlarm"	Non-retain

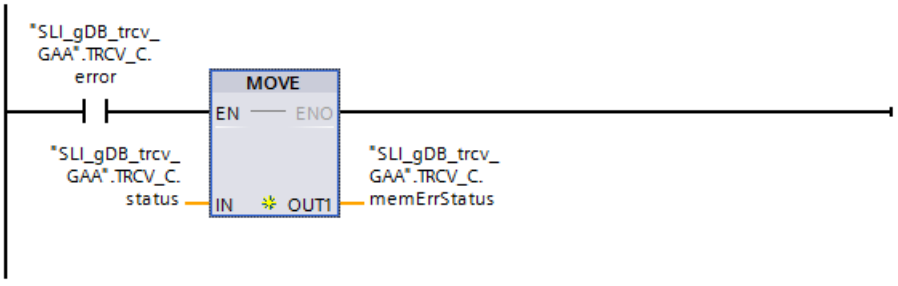
Create the following interconnections in the FB.

Network 1: Interconnect the parameters of the "TRCV_C" instruction as follows. Configure the CONNECT parameter of TRCV_C using the TRCV_C wizard.

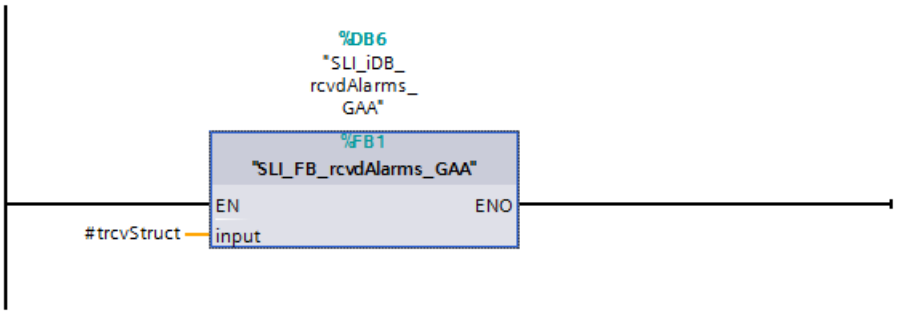
Additional information on the configuration of TRCV_C (Page 4437) is available here: Program example for send functions (Page 4446).



Network 2: To save the status if a TRCV_C error occurs, interconnect the parameter as follows.



Network 3: Interconnect the parameters of the "SLI_FB_rcvdAlarms_GAA" function block as follows.



Results and procedure

The results and the procedure for the communication connection are described here: Program example for Get_Alarm & Ack_Alarms - Part 2 (Page 3921).

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Ack_Alarms: Acknowledge alarms (Page 3901)

Program example for Get_Alarm & Ack_Alarms - Part 2 (S7-1200, S7-1500)

Introduction

In the second part of the program example for Get_Alarm and Ack_Alarms, you will become familiar with the results of the configuration and the ProDiag alarm.

The requirements and the interconnection of the parameters are available here: Program example for Get_Alarm & Ack_Alarms - Part 1 (Page 3903).

Procedure for connection setup between CPU 1 and CPU 2

Observe the order of the action steps:

1. For CPU 2, change the tag "TRCV_C.start" to "TRUE".
2. For CPU 2, change the tag "TRCV_C.comControl" to "TRUE".
3. For CPU 1, change the tag "TSEND_C.start" to "TRUE".
4. For CPU 1, change the tag "TSEND_C.comControl" to "TRUE".

Procedure for disconnection of CPU 1 and CPU 2

Observe the order of the action steps:

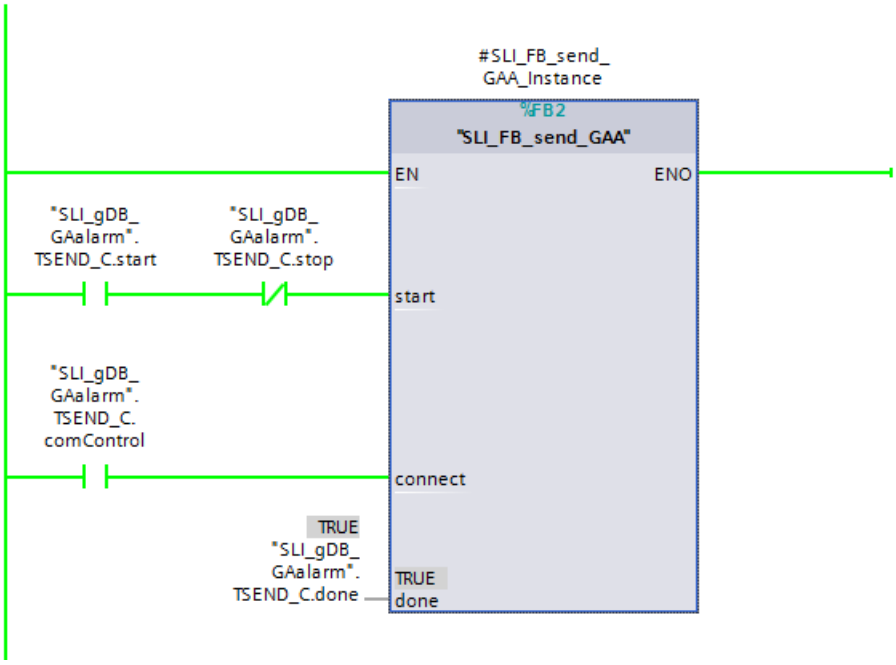
1. For CPU 1, change the tag "TSEND_C.stop" to "TRUE".
2. For CPU 2, change the tag "TRCV_C.start" to "FALSE".
3. For CPU 2, change the tag "TRCV_C.comControl" to "FALSE".
4. For CPU 1, change the tag "TSEND_C.start" to "FALSE" and the tag "TSEND_C.stop" to "FALSE".
5. For CPU 1, change the tag "TSEND_C.comControl" to "FALSE".

Result of CPU 1: FB "SLI_FBCall_GetAlarm_AckA" - Network 1

If the normally open contact ("TSEND_C.start") supplies the signal state "TRUE", the "TSEND_C" instruction is started. When the input parameter CONNECT ("TSEND_C.comControl") supplies the signal state "TRUE", the "TSEND_C" instruction creates a communication connection between CPU 1 and CPU 2.

Successful transfer of the data record (#sendStruct) is indicated by the output parameter DONE ("TSEND_C.done") with the signal state "TRUE" ("TSEND_C.status has the value "0000"). After transmitting the data record (#sendStruct) to CPU 2, the communication connection continues to be monitored (status "7004").

If the normally open contact ("TSEND_C.stop") supplies the signal state "TRUE", the "TSEND_C" instruction is stopped.

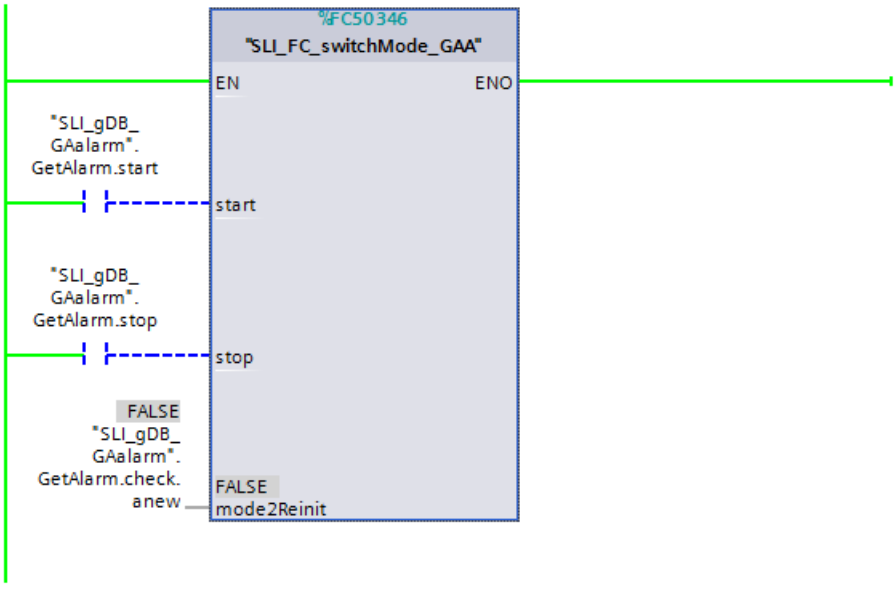


Additional information on the operating principle of TSEND_C (Page 4423) and TRCV_C (Page 4437) is available here: Program example for send functions (Page 4446).

Result of CPU 1: FB "SLI_FBCall_GetAlarm_AckA" - Network 2

If the normally open contact ("GetAlarm.start") supplies the signal state "TRUE", the "Get_Alarm" instruction is started. For this purpose, the "Get_Alarm" instruction is called in mode 1 ("GetAlarm.operateMode.value") and then changes to mode 2. After it has been processed, the tag "GetAlarm.start" is automatically reset to "FALSE".

If the normally open contact ("GetAlarm.stop") supplies the signal state "TRUE", the "Get_Alarm" instruction is stopped. For this purpose, the "Get_Alarm" instruction is called in mode 3 ("GetAlarm.operateMode.value") and then changes to inactivity (mode 0). A requirement for the stop command is successful logon of Get_Alarm ("GetAlarm.operateMode.Logon_done" is "TRUE"). After it has been processed, the tag "GetAlarm.stop" is automatically reset to "FALSE".



The success state of the processed mode of Get_Alarm is recorded:

- "GetAlarm.operateMode.Logon_done" is "TRUE" = mode 1 was completed successfully.
- "GetAlarm.operateMode.ReadData_ready" is "TRUE" = mode 2 is set. Get_Alarm is ready to read.
- "GetAlarm.operateMode.ReadData_active" is "TRUE" = Get_Alarm is in mode 2 and busy reading the data.
- "GetAlarm.operateMode.Logoff_done" is "TRUE" = mode 3 was completed successfully.

Note

Unintentional log off of Get_Alarm

When Get_Alarm is logged off in mode 2, the output parameter INIT ("GetAlarm.initiateLogin") returns the signal state "TRUE". This event is counted as read loss ("GetAlarm.currReadLoss") in FC "SLI_FC_switchMode_GAA" and the (INOUT) parameter MODE2REINIT ("GetAlarm.check.anew") of "SLI_FC_switchMode_GAA" is set to "TRUE". This results in a reset of the tags "GetAlarm.dataReady" and "GetAlarm.initiateLogin" and the "Get_Alarm" instruction is automatically restarted in mode 1. Afterwards Get_Alarm is once again ready to read in mode 2.

18		currReadLoss	UDInt	0	0
19		dataReady	Bool	false	FALSE
20		initiateLogin	Bool	false	FALSE
21		busy	Bool	false	FALSE
22		status	Word	16#0	16#0000
23		memError	Bool	false	FALSE
24		memErrStatus	Word	16#0	16#0000

Result of CPU 1: FB "SLI_FBCall_GetAlarm_AckA" - Network 3

When the input parameter OPERATEMODE ("GetAlarm.operateMode.value") shows the value "2", the "Get_Alarm" instruction reads all alarms of CPU 1. For this purpose, the respective alarm is recorded at the input parameter DATA ("GetAlarm.currRecord").

SLI_gDB_GAalarm

	Name	Data type	Start value	Monitor value
1	Static			
2	TSEND_C	Struct		
3	GetAlarm	Struct		
4	start	Bool	false	FALSE
5	stop	Bool	false	FALSE
6	check	Struct		
7	anew	Bool	false	FALSE
8	operateMode	Struct		
9	value	Byte	16#0	16#02
10	Logon_done	Bool	false	TRUE
11	ReadData_ready	Bool	false	TRUE
12	ReadData_active	Bool	false	FALSE
13	Logoff_done	Bool	false	FALSE
14	outputMode	Byte	16#3	16#03
15	languageCodeID	UDInt	0	0
16	currRecord	AlarmData		

The signal state "TRUE" at the output parameter DATAREADY ("#dataReady") indicates a new alarm. Because the value of "#dataReady" (as well as that of "#initiate") is only set for one cycle, the status is saved with the tag "GetAlarm.dataReady".

When the output parameter INIT ("#initiate") returns the signal state "TRUE" in mode 2, Get_Alarm was logged off and must be logged on again. See the note on network 2. In mode 1, "#initiate" / "GetAlarm.initiateLogin" shows the successful logon.

In case of an error, the status of Get_Alarm ("GetAlarm.status") is saved with the tags "GetAlarm.memErrStatus" and "GetAlarm.memErr", and Get_Alarm is set to mode 3 to log off. When Get_Alarm is to be logged off and no error occurs ("GetAlarm.status" has the value "16#0000"), logoff is saved through "GetAlarm.operateMode.Logoff_done" with the value "TRUE".

3	GetAlarm	Struct		
4	start	Bool	false	FALSE
5	stop	Bool	false	FALSE
6	check	Struct		
7	anew	Bool	false	FALSE
8	operateMode	Struct		
9	value	Byte	16#0	16#00
10	Logon_done	Bool	false	FALSE
11	ReadData_ready	Bool	false	FALSE
12	ReadData_active	Bool	false	FALSE
13	Logoff_done	Bool	false	TRUE
14	outputMode	Byte	16#3	16#03

The input parameter DATAMODE ("GetAlarm.outputMode") with the value "16#03" specifies that the alarms are read out completely (with all information). The input parameter DISPCLASSNR is not connected. This means Get_Alarm reads all display classes. The input parameter LCID ("GetAlarm.languageCodeID") with the value "0" specifies that all language versions of alarms should be read.

Result of CPU 1: FB "SLI_FBCall_GetAlarm_AckA" - Network 4

Under the following conditions, the currently read alarm ("GetAlarm.currRecord") as well as the read losses counted until then ("GetAlarm.currReadLoss") are saved as a combined data record in the memory array "GetAlarm.alarmStorage":

- "GetAlarm.dataReady" returns the signal state "TRUE". This means Get_Alarm has read a new alarm.
- In addition, the data record of the read alarm ("GetAlarm.currRecord") is not empty. Its alarm text is checked for this reason.

26	alarmStorage	Array[0..15] of "SG...		
27	alarmStorage[0]	"SGetAlarm"		
28	countReadL...	UDInt	0	0
29	alarm	AlarmData		
30	Produce...	UInt	0	9
31	ID_1	UInt	0	53
32	ID_2	UInt	0	1
33	PRIO	UInt	0	0
34	TimeSta...	LDT	LDT#1970-01-01-(	LDT#2016-06-22-0...
35	State	Byte	16#0	16#01
36	LCID	Array[0..2] of UInt		
37	LCID...	UInt	0	0
38	LCID...	UInt	0	0
39	LCID...	UInt	0	0
40	AlarmText	Array[0..2] of WStri...		
41	Alar...	WString[254]	WSTRING#''	WSTRING#'Warning ...

Next, the position used ("#posForAlarm") is saved in the memory array "GetAlarm.alarmStorage" in the global tag "GetAlarm.posForAlarm". In addition, the alarms are counted and the current value is recorded with the global tag "GetAlarm.countAlarms".

The position information ("#posForAlarm") is incremented so that a new alarm can be saved in a different cell in the memory array. A requirement for incrementing is that the limit of the memory array ("GetAlarm.maxPositon") is not exceeded. When the limit is exceeded, this fact is noted ("GetAlarm.limitReach") and the position information ("#posForAlarm") is set to the value "0". Any subsequent alarms would therefore overwrite old alarm entries in the memory array with this setting.

Successful copying is noted in the tag "GetAlarm.savingDone". The "GetAlarm.initiateSending" tag is set to initiate subsequent sending of the alarm.

At the end of processing of "SLI_FB_saveAlarms_GAA", the tag "GetAlarm.dataReady" (and local tags) is reset to "FALSE". The program of "SLI_FB_saveAlarms_GAA" is only processed again when "GetAlarm.dataReady" returns the value "TRUE".

25		maxPosition	UInt	15	15
26		alarmStorage	Array[0..15] of "SG...		
27		limitReach	Bool	false	FALSE
28		savingDone	Bool	false	TRUE
29		initiateSending	Bool	false	FALSE
30		countAlarms	UInt	0	1
31		posForAlarm	UInt	0	0

Result of CPU 1: FB "SLI_FBCall_GetAlarm_AckA" - Network 5

When the normally open contact ("startPump") and the normally closed contact ("closeValve") return the signal state "TRUE", the configured ProDiag alarm is triggered.



Get_Alarm reads the alarm and saves it in the assigned tag ("GetAlarm.currRecord"). In network 4, the alarm is subsequently copied to the memory array ("GetAlarm.alarmStorage").

17		currRecord	AlarmData		
18		ProducerID	UInt	0	9
19		ID_1	UInt	0	53
20		ID_2	UInt	0	1
21		PRIO	UInt	0	0
22		TimeStamp	LDT	LDT#1970-01-01-0	LDT#2016-06-22-0...
23		State	Byte	16#0	16#01
24		LCID	Array[0..2] of UInt		
25		LCID[0]	UInt	0	0
26		LCID[1]	UInt	0	0
27		LCID[2]	UInt	0	0
28		AlarmText	Array[0..2] of WStri...		
29		AlarmText[0]	WString[254]	WSTRING#''	WSTRING#'Warning ...
30		AlarmText[1]	WString[254]	WSTRING#''	WSTRING#''
31		AlarmText[2]	WString[254]	WSTRING#''	WSTRING#''

Result of CPU 1: FB "SLI_FBCall_GetAlarm_AckA" - Network 1 (data transmission)

When the tag "GetAlarm.initiateSending" returns the signal state "TRUE", the saved position ("GetAlarm.posForAlarm") is transferred in the memory array of a local tag with the same name ("#posForAlarm") and the command is forwarded for a new data transfer. The "#initiateSending" tag is set to "TRUE".

If, in addition, no error is signaled and TSEND_C is signaled as busy ("TSEND_C.error" and "TSEND_C.busy" are "FALSE"), the respective alarm ("GetAlarm.alarmStorage[..]") is transferred in abbreviated form to the structure "#sendStruct". If no stop command ("TSEND_C.stop" to "TRUE") was given, the tag "TSEND_C.start" is set to "TRUE".

The start command for data transmission ("TSEND_C.start" is "TRUE") sends the abbreviated alarm ("#sendStruct") to CPU 2 over multiple cycles.

Result of CPU 1: FB "SLI_FBCall_GetAlarm_AckA" - Network 6

If the normally open contact ("AckAlarm.executeAck") supplies the signal state "TRUE", the "Ack_Alarms" instruction is started. According to the value "1" of the input parameter MODE ("#acknowledge"), Ack_Alarms acknowledges all active alarms. Processing in the example takes place without errors ("AckAlarm.error" is "FALSE" and "AckAlarm.status" has the value "16#0000").

To view the confirmations, use an HMI device or the Web server of the CPU.

Result of CPU 2: FB "SLI_FBCall_trcv_GAA" - Network 1

If the normally open contact ("TRCV_C.start") supplies the signal state "TRUE", the "TRCV_C" instruction is started. When the input parameter CONNECT ("TRCV_C.comControl") supplies the signal state "TRUE", the "TRCV_C" instruction creates a communication connection between CPU 2 and CPU 1.

Successful transfer of the data record (#trcvStruct) is indicated by the output parameter DONE ("TRCV_C.done" and "TRCV_C.dataReceived") with the signal state "TRUE" ("TRCV_C.status has the value "0000"). After transmitting the data record (#trcvStruct) to CPU 2, the communication connection continues to be monitored and TRCV_C is ready to receive (status

"7006"). The length in Byte of the actually transmitted data record is detected with the output parameter RCVD_LEN ("TRCV_C.rcvLen").

SLI_gDB_trcv_GAA

	Name	Data type	Start value	Monitor value
1	Static			
2	TRCV_C	Struct		
3	start	Bool	false	TRUE
4	comControl	Bool	false	TRUE
5	done	Bool	false	TRUE
6	busy	Bool	false	TRUE
7	error	Bool	false	FALSE
8	status	Word	16#0	16#7006
9	memErrStatus	Word	16#0	16#0000
10	rcvLen	UDInt	0	540
11	dataReceived	Bool	false	FALSE
12	GetAlarm	Struct		
13	maxPosition	UInt	15	15
14	rcvdStorage	Array[0..15] of "rcv...		
15	limitReach	Bool	false	FALSE
16	savingDone	Bool	false	FALSE
17	countAlarms	UInt	0	0
18	posForAlarm	UInt	0	0

Additional information on the operating principle of TSEND_C (Page 4423) and TRCV_C (Page 4437) is available here: Program example for send functions (Page 4446).

Result of CPU 2: FB "SLI_FBCall_trcv_GAA" - Network 2

If the normally open contact ("TRCV_C.error") returns the signal state "TRUE", the status ("TRCV_C.status") of the error is saved ("TRCV_C.memErrStatus").

Result of CPU 2: FB "SLI_FBCall_trcv_GAA" - Network 3

An alarm is saved in the memory array "GetAlarm.rcvdStorage" under the following conditions:

- "TRCV_C.dataReceived" returns the signal state "TRUE". This means a new data record was sent to TRCV_C.
- In addition, the new received data record (#trcvStruct) is not empty.

12	GetAlarm	Struct		
13	maxPosition	UInt	15	15
14	rcvdStorage	Array[0..15] of "rcv...		
15	rcvdStorage[0]	"rcvGetAlarm"		
16	countReadL...	UDInt	0	0
17	ProducerID	UInt	0	9
18	ID_1	UInt	0	53
19	ID_2	UInt	0	1
20	PRIO	UInt	0	0
21	TimeStamp	LDT	LDT#1970-01-01-(	LDT#2016-06-22-0...
22	State	Byte	16#0	16#01
23	LCID	Array[0..2] of UInt		
24	AlarmText	WString[254]	WSTRING#''	WSTRING#'Warning ...

Next, the position used ("#posForAlarm") is saved in the memory array "GetAlarm.rcvdStorage" in the global tag "GetAlarm.posForAlarm". In addition, the alarms are counted and the current value is recorded with the global tag "GetAlarm.countAlarms".

The position information ("#posForAlarm") is incremented so that a new alarm can be saved in a different cell in the memory array. A requirement for incrementing is that the limit of the memory array ("GetAlarm.maxPositon") is not exceeded. When the limit is exceeded, this fact is noted ("GetAlarm.limitReach") and the position information ("#posForAlarm") is set to the value "0". Any subsequent alarms would therefore overwrite old alarm entries in the memory array with this setting.

Successful copying is noted in the tag "GetAlarm.savingDone".

At the end of processing of "SLI_FB_rcvdAlarms_GAA", the tag "TRCV_C.dataReceived" (and local tags) is reset to "FALSE". The program of "SLI_FB_rcvdAlarms_GAA" is only processed again when "TRCV_C.dataReceived" returns the value "TRUE".

SLI_gDB_trcv_GAA

	Name	Data type	Start value	Monitor value
1	Static			
2	TRCV_C	Struct		
3	start	Bool	false	TRUE
4	comControl	Bool	false	TRUE
5	done	Bool	false	TRUE
6	busy	Bool	false	TRUE
7	error	Bool	false	FALSE
8	status	Word	16#0	16#7006
9	memErrStatus	Word	16#0	16#0000
10	rcvLen	UDInt	0	540
11	dataReceived	Bool	false	FALSE
12	GetAlarm	Struct		
13	maxPosition	UInt	15	15
14	rcvdStorage	Array[0..15] of "rcv...		
15	limitReach	Bool	false	FALSE
16	savingDone	Bool	false	TRUE
17	countAlarms	UInt	0	1
18	posForAlarm	UInt	0	0

Part 1 of the program example

The requirements and the interconnection of the parameters are available here: Program example for Get_Alarm & Ack_Alarms - Part 1 (Page 3903).

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

11.4.4.9 Diagnostics (S7-1200, S7-1500)

RD_SINFO: Read current OB start information (S7-1200, S7-1500)

Description

You use the instruction "RD_SINFO" to read the start information

- of the last OB called that has not yet been completely executed, and
- of the last startup OB started.

There is no time stamp in either case. If the call is in OB 100, OB 101 or OB 102, two identical start information messages will be returned.

Parameter

The following table shows the parameters of the "RD_SINFO" instruction:

Parameter	Declaration	Data type	Memory area	Description
RET_VAL	Return	INT	I, Q, M, D, L	Error information
TOP_SI	Output	VARIANT	D, L	Start information of the current OB
START_UP_SI	Output	VARIANT	D, L	Start information of the startup OB last started

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

SDTs of the TOP_SI parameter

The following table shows the possible SDTs for the TOP_SI parameter:

Organization blocks (OB)	System data types (SDT)	System data type numbers
Any	SI_classic	592
	SI_none	593
ProgramCycleOB	SI_ProgramCycle	594
TimeOfDayOB	SI_TimeOfDay	595
TimeDelayOB	SI_Delay	596
CyclicOB	SI_Cyclic	597
ProcessEventOB	SI_HWInterrupt	598
	SI_HWInterrupt_Extended	616
ProfileEventOB StatusEventOB UpdateEventOB	SI_Submodule	601
SynchronousCycleOB	SI_SynchCycle	602
IOredundancyErrorOB	SI_IORedundancyError	604
CPUredundancyErrorOB	SI_CPURedundancyError	605
TimeErrorOB	SI_TimeError	606

Organization blocks (OB)	System data types (SDT)	System data type numbers
DiagnosticErrorOB	SI_DiagnosticInterrupt	607
PullPlugEventOB	SI_PlugPullModule	608
PeripheralAccessErrorOB	SI_AccessError	609
RackStationFailureOB	SI_StationFailure	610
ServoOB	SI_Servo	611
IpoOB	SI_Ipo	612
StartupOB	SI_Startup	613
ProgrammingErrorOB IOaccessErrorOB	SI_ProgIOAccessError	614

SDTs of the START_UP_SI parameter

The following table shows the possible SDTs for the START_UP_SI parameter:

System data types (SDT)	System data type numbers
SI_classic	592
SI_none	593
SI_Startup	613

Structures

The following tables set out the meaning of the structure elements of the individual structures:

Table 11-46 SI_classic structure

Structure element	Data type	Description
EV_CLASS	BYTE	• Bits 0 to 3: Event ID • Bits 4 to 7: Event class
EV_NUM	BYTE	Event number
PRIORITY	BYTE	Priority class number (Meaning of B#16#FE: OB not available or disabled or cannot be started in current operating mode)
NUM	BYTE	OB number
TYP2_3	BYTE	Data ID 2_3: Identifies the information entered in ZI2_3
TYP1	BYTE	Data ID 1: Identifies the information entered in ZI1
ZI1	WORD	Additional information 1
ZI2_3	DWORD	Additional information 2_3

Table 11-47 SI_none structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)

Table 11-48 SI_ProgramCycle structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 1	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
Initial_Call	BOOL	For OB_Class = 1, 30, 52, 61, 65
Remanence	BOOL	For OB_Class = 1

Table 11-49 SI_TimeOfDay structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 10	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
CaughtUp	BOOL	For OB_Class = 10
SecondTime	BOOL	For OB_Class = 10

Table 11-50 SI_Delay structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 20	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
Sign	WORD	For OB_Class = 20

Table 11-51 SI_Cyclic structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 30	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
Initial_Call	BOOL	For OB_Class = 1, 30, 52, 61, 65
Event_Count	INT	For OB_Class = 30, 51, 52, 61, 65, 91, 92

Table 11-52 SI_HWInterrupt structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 40	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
LADDR	HW_IO	For OB_Class = 40, 51, 55, 56, 57, 70, 82, 83, 85, 86, 91, 92
USI	WORD	For OB_Class = 40
IChannel	USINT	For OB_Class = 40
EventType	BYTE	For OB_Class = 40

Table 11-53 Structure SI_HWInterrupt_Extended

Structure element	Data type	Byte	Description
SI_Format	USINT	0	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 40	1	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	2	OB number (1 ... 32767)
LADDR	HW_IO	4	For OB_Class = 40, 51, 55, 56, 57, 70, 82, 83, 85, 86, 91, 92
USI	WORD	6	For OB_Class = 40
IChannel	USINT	8	For OB_Class = 40
EventType	BYTE	9	For OB_Class = 40
PointAddr	DWORD	12	For OB_Class = 40

Table 11-54 SI_Submodule structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
LADDR	HW_IO	For OB_Class = 40, 51, 55, 56, 57, 70, 82, 83, 85, 86, 91, 92

Structure element	Data type	Description
Slot	UINT	For OB_Class = 55, 56, 57
Specifier	WORD	For OB_Class = 55, 56, 57

Table 11-55 SI_SynchCycle structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 61	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
Initial_Call	BOOL	For OB_Class = 1, 30, 52, 61, 65
PIP_Input	BOOL	For OB_Class = 61, 91, 92
PIP_Output	BOOL	For OB_Class = 61, 91, 92
IO_System	USINT	For OB_Class = 61, 91, 92
Event_Count	INT	For OB_Class = 30, 51, 52, 61, 65, 91, 92
SyncCycleTime	LTIME	Calculated cycle time

Table 11-56 SI_IORedundancyError structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 70	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
LADDR	HW_ANY	For OB_Class = 40, 51, 55, 56, 57, 70, 82, 83, 85, 86, 91, 92
Event_Class	BYTE	For OB_Class = 70, 83, 85, 86
Fault_ID	BYTE	For OB_Class = 70, 80, 83, 85, 86

Table 11-57 SI_CPURedundancyError structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 72	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
Switch_Over	BOOL	For OB_Class = 72

Table 11-58 SI_TimeError structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 80	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
Fault_ID	BYTE	For OB_Class = 70, 80, 83, 85, 86
Csg_OBnr	OB_ANY	For OB_Class = 80
Csg_Prio	UINT	For OB_Class = 80

Table 11-59 SI_DiagnosticInterrupt structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 82	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
IO_State	WORD	For OB_Class = 82
LADDR	HW_ANY	For OB_Class = 40, 51, 55, 56, 57, 70, 82, 83, 85, 86, 91, 92
Channel	UINT	For OB_Class = 82
MultiError	BOOL	For OB_Class = 82

Table 11-60 SI_PlugPullModule structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 83	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
LADDR	HW_IO	For OB_Class = 40, 51, 55, 56, 57, 70, 82, 83, 85, 86, 91, 92
Event_Class	BYTE	For OB_Class = 70, 83, 85, 86
Fault_ID	BYTE	For OB_Class = 70, 80, 83, 85, 86

Table 11-61 SI_AccessError structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 85	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
LADDR	HW_IO	For OB_Class = 40, 51, 55, 56, 57, 70, 82, 83, 85, 86, 91, 92
Event_Class	BYTE	For OB_Class = 70, 83, 85, 86

Structure element	Data type	Description
Fault_ID	BYTE	For OB_Class = 70, 80, 83, 85, 86
IO_Addr	UINT	For OB_Class = 85
IO_LEN	UINT	For OB_Class = 85

Table 11-62 SI_StationFailure structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 86	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
LADDR	HW_IO	For OB_Class = 40, 51, 55, 56, 57, 70, 82, 83, 85, 86, 91, 92
Event_Class	BYTE	For OB_Class = 70, 83, 85, 86
Fault_ID	BYTE	For OB_Class = 70, 80, 83, 85, 86

Table 11-63 SI_Servo structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 91	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
Initial_Call	BOOL	For OB_Class = 1, 30, 52, 61, 65
PIP_Input	BOOL	For OB_Class = 61, 91, 92
PIP_Output	BOOL	For OB_Class = 61, 91, 92
IO_System	USINT	For OB_Class = 61, 91, 92
Event_Count	INT	For OB_Class = 30, 51, 52, 61, 65, 91, 92
Synchronous	BOOL	

Table 11-64 SI_Ipo structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 92	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
Initial_Call	BOOL	For OB_Class = 1, 30, 52, 61, 65
PIP_Input	BOOL	For OB_Class = 61, 91, 92
PIP_Output	BOOL	For OB_Class = 61, 91, 92
IO_System	USINT	For OB_Class = 61, 91, 92

Structure element	Data type	Description
Event_Count	INT	For OB_Class = 30, 51, 52, 61, 65, 91, 92
Reduction	UINT	For OB_Class = 92

Table 11-65 SI_Startup structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT := 100	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
LostRetentive	BOOL	For OB_Class = 100
LostRTC	BOOL	For OB_Class = 100

Table 11-66 SI_ProgIOAcessError structure

Structure element	Data type	Description
SI_Format	USINT	• 16#FF = No information • 16#FE = Optimized start information
OB_Class	USINT	OB class for "No information" or "Optimized start information"
OB_Nr	UINT	OB number (1 ... 32767)
BlockNr	UINT	For OB_Class = 121, 122
Reaction	USINT	For OB_Class = 121, 122
Fault_ID	BYTE	For OB_Class = 121, 122
BlockType	USINT	For OB_Class = 121, 122
Area	USINT	For OB_Class = 121, 122
DBNr	DB_ANY	For OB_Class = 121, 122
Csg_OBNr	OB_ANY	For OB_Class = 121, 122
Csg_Prio	USINT	For OB_Class = 121, 122
Width	USINT	For OB_Class = 121, 122

Note

If this was created with the block property "Standard, the structure elements specified for the SI_classic structure are identical in content to the temporary tags of an OB.

Note, however, that temporary tags of the individual OBs can have different names and different data types. Also note that the call interface of each OB includes additional information regarding the date and the time of the OB request.

Bits 4 to 7 of the EV_CLASS structure element contain the event class. The following values are possible here:

- 1: Start events from standard OBs
- 2: Start events from synchronous error OBs
- 3: Start events from asynchronous error OBs

The PRIORITY structure element supplies the priority class belonging to the current OB.

Apart from these two elements, NUM is also relevant. NUM contains the number of the current OB or the startup OB that was started last.

Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
8081	Start information of the current OB does not correspond to the specified system data type
8083	Start information of the last startup OB started does not correspond to the specified system data type

Example

OB 80 is the OB that was called last and that has not yet been completely processed and OB 100 is the start-up OB that was started last.

The following table shows the assignment between the structure elements of the TOP_SI parameter of the "RD_SINFO" instruction and the associated local tags of OB 80.

TOP_SI structure element	Data type	OB 80 - Associated local tag	Data type
EV_CLASS	BYTE	OB80_EV_CLASS	BYTE
EV_NUM	BYTE	OB80_FLT_ID	BYTE
PRIORITY	BYTE	OB80_PRIORITY	BYTE
NUM	BYTE	OB80_OB_NUMBR	BYTE
TYP2_3	BYTE	OB80_RESERVED_1	BYTE
TYP1	BYTE	OB80_RESERVED_2	BYTE
ZI1	WORD	OB80_ERROR_INFO	WORD
ZI2_3	DWORD	OB80_ERR_EV_CLASS	BYTE
		OB80_ERR_EV_NUM	BYTE
		OB80_OB_PRIORITY	BYTE
		OB80_OB_NUM	BYTE

The following table shows the assignment between the structure elements of the START_UP_SI parameter of the "RD_SINFO" instruction and the associated local tags of OB 100.

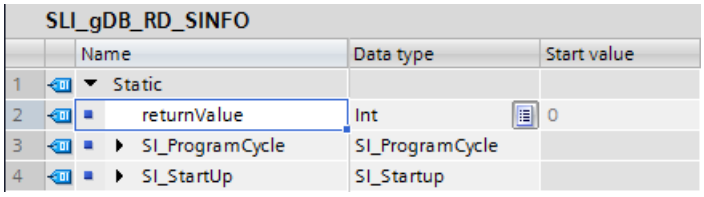
START_UP_SI structure element	Data type	OB 100 - Local tag	Data type
EV_CLASS	BYTE	OB100_EV_CLASS	BYTE
EV_NUM	BYTE	OB100_STRTUP	BYTE
PRIORITY	BYTE	OB100_PRIORITY	BYTE
NUM	BYTE	OB100_OB_NUMBR	BYTE
TYP2_3	BYTE	OB100_RESERVED_1	BYTE
TYP1	BYTE	OB100_RESERVED_2	BYTE

START_UP_SI structure element	Data type	OB 100 - Local tag	Data type
ZI1	WORD	OB100_STOP	WORD
ZI2_3	DWORD	OB100_STRT_INFO	DWORD

Example

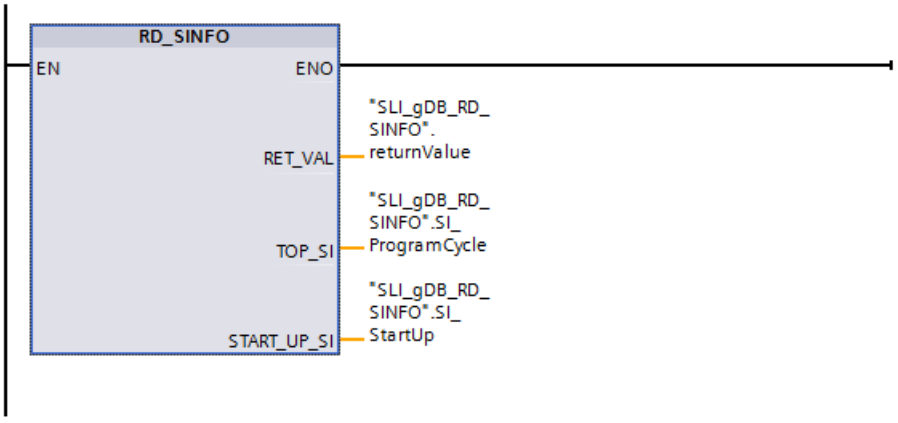
In the following example, you read out the start information of the program cycle OB that was last called and the startup OB of the CPU that was last started.

Create three tags in a global data block for storing the data.



Call the "RD_SINFO" instruction in an OB1. Interconnect the parameters of the "RD_SINFO" instruction as follows.

Use only one OB for the CPU, for the example.



When the input parameter EN returns the signal state "TRUE", the "RD_SINFO" instruction is started. The instruction "RD_SINFO" reads the start information of the last called program cycle OB (OB1) and outputs this at the output parameter TOP_SI ("SI_ProgrammCycle"). Start information of the startup OB that was
last started is output at the output parameter START_UP_SI ("SI_Startup").

According to the value of the parameter "SI_Format", OB1 uses optimized start information. In addition, the class of the OB ("OB_Class") and the assigned number of the OB ("OB_Nr") are read out. No number ("OB_Nr" with the value "0") is read out for the startup OB ("SI_Startup"), because no startup OB is used by the CPU. In addition, standard values ("SI_Format", "OB_Class") are displayed for the startup OB.

The output parameter RET_VAL ("returnValue") indicates that the processing of the example is taking place without errors (the value is "0").

SLI_gDB_RD_SINFO

	Name	Data type	Start value	Monitor value
1	Static			
2	returnValue	Int	0	0
3	SI_ProgramCycle	SI_ProgramCycle		
4	SI_Format	USInt	0	254
5	OB_Class	USInt	1	1
6	OB_Nr	UInt	0	1
7	Initial_Call	Bool	false	FALSE
8	Remanence	Bool	false	TRUE
9	SI_StartUp	SI_Startup		
10	SI_Format	USInt	0	254
11	OB_Class	USInt	100	100
12	OB_Nr	UInt	0	0
13	LostRetentive	Bool	false	FALSE
14	LostRTC	Bool	false	FALSE

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

RT_INFO: Read out runtime statistics (S7-1500)

Description

Use the "RT_INFO" instruction to generate statistics on the runtime of specific organization blocks, communication or the user program.

Use the MODE parameter to select the information to be output:

- MODE 1 to 3 return the data on the runtime of a specific organization block whose number was specified at the OB parameter.
- MODE 10 and 11 measure over a time period of 1 s and return the mean value of the percentage of total runtime used by higher-priority OBs (MODE 10) or mean value of the percentage of total runtime used by communication (MODE 11).
- MODE 20 and 21 return the percentages of the runtime allotted to communication and the user program for the last program cycle.
- MODE 23 to 25 output the shortest, longest, and current cycle time of the user program.
- MODE 30 to 32 return data on configured settings of the user program.

All measurements are started once again when the CPU transitions from startup to RUN.

Parameters

The following table shows the parameters of the "RT_INFO" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	UINT	I, Q, M, D, L or constant	Use the MODE parameter to select the information you want to read out (see table "MODE parameter").
OB	Input	OB_ANY	I, Q, M, D, L or constant	Use the OB parameter to select the OB for which you want to read out information.
RET_VAL	Return	INT	I, Q, M, D, L	Error information (see " RET_VAL parameter").
INFO	InOut	VARIANT	I, Q, M, D, L	Pointer to the area to which the read data is to be written. The data type required for INFO depends on the MODE parameter (see table "MODE parameter").

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter MODE

The following table shows which information is returned for the values at the MODE parameter.

MODE (decimal)	Description	Note	Value at OB parameter	Data type of INFO	Available as of CPU version
1	Runtime of a specific OB	The runtime information always refers to the specified OB. Processes, such as interruptions by OBs of a higher priority or communication, are not included. You can use the instruction "RUNTIME" to measure the runtime of the entire program.	OB number	LTIME	S7-1500 V1.5
2	Maximum runtime of a specific OB		OB number	LTIME	S7-1500 V1.5
3	Minimum runtime of a specific OB		OB number	LTIME	S7-1500 V1.5

MODE (decimal)	Description	Note	Value at OB parameter	Data type of INFO	Available as of CPU version
10	Mean value of the percentage of total runtime used by higher-priority OBs The runtime of all OBs used in the user program with a higher priority than cyclical program OBs is output (ProgramCycle). These usually include all OB types except startup OBs. The CPU determines which OBs can be used and their priority. This information is included in the basic chapters on programming.	Averaging takes place over a time period of 1 s.	Not relevant	UINT	S7-1500 V1.5
11	Mean value of the percentage of total runtime used by communication The percentage of communication processes in the total runtime of the user program is output.		Not relevant	UINT	S7-1500 V1.5
20	Identical to MODE 10 except that the evaluation is based on the last program cycle executed.	• If the cycle time is too high (> 1 second), the calculation is not performed. The instruction outputs the value 65535 (0xFFFF). • If the cycle time is too low (< 1 millisecond), the calculation cannot be performed. In this case, the last cycle with a duration of at least one millisecond is evaluated. By assigning a minimum cycle time in the CPU properties, you can prevent cycle times from falling below 1 ms.	Not relevant	UINT	S7-1500 V1.7
21	Identical to MODE 11 except that the evaluation is based on the last program cycle executed.		Not relevant	UINT	S7-1500 V1.7
23	Longest cycle time Duration of the longest cycle since the last transition from STOP to RUN.	The times correspond to the "Cycle times measured" values in the "Cycle time" dialog of the TIA Portal. You open the dialog using Online & Diagnostics > Diagnostics > Cycle time.	Not relevant	LTIME	S7-1500 V1.7
24	Shortest cycle time Duration of the shortest cycle since the last transition from STOP to RUN.		Not relevant	LTIME	S7-1500 V1.7
25	Current/last cycle time Duration of the last cycle.		Not relevant	LTIME	S7-1500 V1.7

MODE (decimal)	Description	Note	Value at OB parameter	Data type of INFO	Available as of CPU version
30	Cycle monitoring time Maximum permitted duration of the CPU program. If the cycle time exceeds the cycle monitoring time, the CPU goes to STOP or calls a time error OB.	The times correspond to the "Configured cycle time" values in the "Cycle time" dialog of the TIA Portal. You open the dialog using Online & Diagnostics > Diagnostics > Cycle time.	Not relevant	LTIME	S7-1500 V1.5
31	Output of the configured minimum cycle time for the user program. If a minimum cycle time has been assigned in the CPU properties, the operating system delays the start of the new cycle until the minimum cycle time has been reached.		Not relevant	LTIME	S7-1500 V1.5
32	Output of the configured maximum communication load as a percentage	The specification of the percentage of the cycle load allotted to communication is made in the CPU properties under "Communication load".	Not relevant	UINT	S7-1500 V1.5

Parameter RET_VAL

The following table shows the meaning of the values of the RET_VAL parameter:

Error code* (W#16#...)	Explanation
0	No error
8080	Value at the MODE parameter is not supported.
8081	Organization block selected at OB parameter does not exist in the user program.
8092	Selected MODE parameter is not supported by this CPU version.
80C3	Insufficient resources. Try to call the instruction again at a later time.
8452	Data type at INFO parameter is incorrect. Check whether the correct data type was selected depending on the MODE parameter.

Example

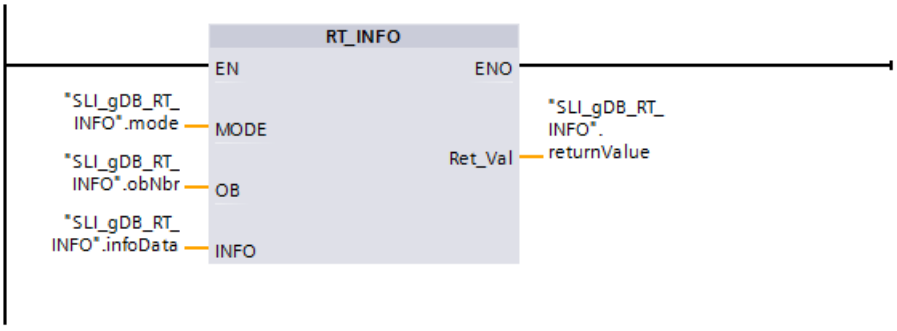
In the following example, you read out the runtime of a cycle organization block.

Create four tags in a global data block for storing the data.

SLI_gDB_RT_INFO

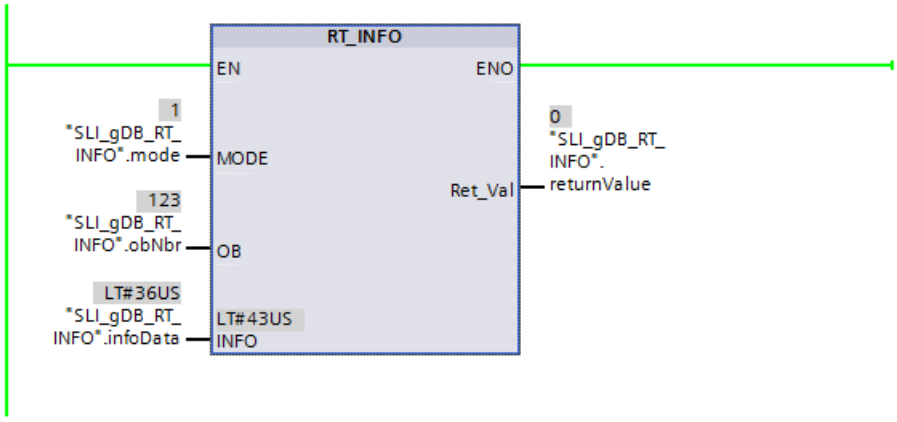
	Name	Data type	Start value
1	Static		
2	mode	UInt	1
3	obNbr	OB_ANY	123
4	infoData	LTime	LT#0ns
5	returnValue	Int	0

Interconnect the parameters of the instruction as follows.



- Create a new block of type "Program cycle". Specify the OB number at the OB parameter.
- Enter "1" (read out runtime of a particular OB) at the MODE parameter.
- Specify a tag with the data type LTIME at the INFO parameter ("infoData" in this case).
- Specify a tag with the data type INT at the RET_VAL parameter to output the error messages of the instruction.

After calling the instruction, the currently measured runtime is written to the "infoData" tag.



Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

LED: Read LED status (S7-1200, S7-1500)

Description

You can use the "LED" instruction to read out the status (e.g., "On" or "Off") of a particular module LED.

- With the LADDR parameter, you address the CPU or the interface.
- With the LED parameter, you select the module LED whose current status is read out using the instruction.
- The RET_VAL parameter outputs the status of the selected LED when the instruction is called. Depending on the LED selected, only certain status information can be displayed. For example, some LEDs have only one color. Refer to the hardware documentation of the respective module for information on the possible statuses of a particular LED.

Parameters

The following table shows the parameters of the "LED" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_IO	I, Q, M, L or constant	Hardware identifier of the CPU or interface The number is assigned automatically and is stored in the properties of the CPU or interface in the hardware configuration (*CPU name* + *~Common*).
LED	Input	UINT	I, Q, M, D, L or constant	Identification number of the LED: • 1: STOP/RUN • 2: ERROR • 3: MAINT (maintenance) • 4: Redundant • 5: Link (green) • 6: Rx/Tx (yellow)
RET_VAL	Return	INT	I, Q, M, D, L	Status of LED

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

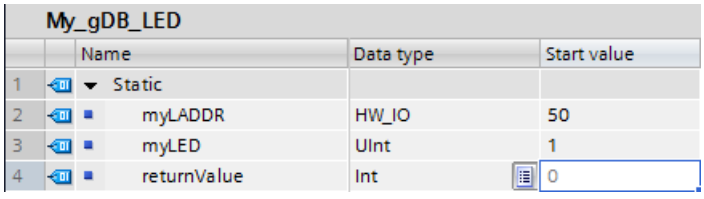
Parameter RET_VAL

RET_VAL	Description
0 to 9	LED status: • 0 = LED does not exist or status information not available • 1 = Permanently switched off • 2 = Color 1 (e.g., for LED STOP/RUN: green) permanently ON • 3 = Color 2 (e.g., for LED STOP/RUN: orange) permanently ON • 4 = Color 1 flashing at 2 Hz • 5 = Color 2 flashing at 2 Hz • 6 = Colors 1 and 2 flashing alternately at 2 Hz • 7 = LED is active, color 1 • 8 = LED is active, color 2 • 9 = LED does not exist or status information not available For reasons of compatibility, ENO is set to FALSE for this value of RET_VAL.
8091	Hardware component addressed using the LADDR parameter is not available.
8092	A hardware component was addressed with the LADDR parameter that does not return the required information.
8093	The identification number specified at the LED parameter is not defined.
80Bx	The CPU specified in the LADDR parameter does not support the "LED" instruction.

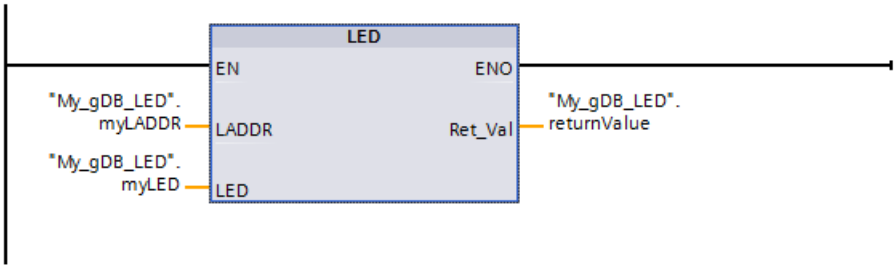
Example

In the following example, you read out the status of the CPU LED.

Create three tags in a global data block for storing the data.

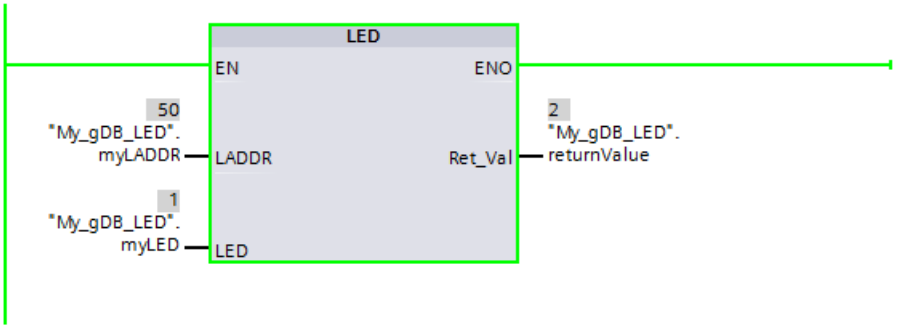


Interconnect the parameters of the instruction as follows.



The HW identifier of the CPU is made known to the "LED" instruction through parameter LADDR ("myLADDR"). The CPU LED to be monitored is made known through parameter LED ("myLED"). The status of the CPU LED (STOP/RUN) is queried. If the CPU is placed in RUN operating mode from STOP, the value "6" (green and orange alternating) is indicated at output

parameter RET_VAL ("returnValue"). Then, the value "2" (permanently green) is indicated as the LED status ("returnValue").



Note: The meaning of the colors for the STOP/RUN LED are as follows:

Color	Meaning
Red	STOP
Green	RUN
Alternating green and orange	The CPU is loading
Alternating red and green	Program processing is running

Additional information on the meaning of the LED colors can be found in the hardware description of the CPU.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Get_IM_Data: Reading identification and maintenance data (S7-1200, S7-1500)

Description

The "Get_IM_Data" instruction reads the identification and maintenance data (I&M) data from a device. Use the LADDR parameter to select the device from which you want to read the I&M via the hardware identifier.

The IM_TYPE parameter is used to select the data that is read out via the instruction:

- IM_TYPE = 0: I&M 0 data
  The I&M 0 data are the device-specific basic information of a device and contain information such as the manufacturer ID, order number, serial number and the hardware and firmware version. Only read access is possible to the I&M 0 data. The information is also displayed in the TIA Portal with the "Online & Diagnostics" view of a device.
- IM_TYPE = 11: I&M 1 data from the parameter assignment data of the CPU
  The I&M 1 data contain a functional description of the device and the location designation. i.e. the information about how the device is designated within the plant.

- IM_TYPE = 12: I&M 2 data from the parameter assignment data of the CPU
  The I&M 2 data includes the date of installation, i.e. the information about when the device was installed in the plant.
- IM_TYPE = 13: I&M 3 data from the parameter assignment data of the CPU
  The I&M 3 data contains additional information about the installed device. The additional information is free text and is assigned as desired.

The I&M data read is written to the addressed area defined at the DATA parameter.

The execution status of the read job is displayed via the BUSY, DONE, ERROR output parameters and the two middle bytes of the STATUS output parameter.

Definition: Identification and maintenance data (I&M)

Identification and maintenance (I&M) data refers to information stored in a module, which assists you in checking the plant configuration, locating hardware changes in a plant and eliminating errors.

- Identification data (I-data) is static information about a device that can only be read.
- Maintenance data (M-data) refers to plant-dependent information, such as the installation location or date. Maintenance data are created during configuration and written to the module.

Parameters

The following table shows the parameters of the "Get_IM_Data" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware ID of the device The number is assigned automatically and is stored in the properties of the device or in the hardware configuration.
IM_TYPE	Input	UINT	I, Q, M, D, L or constant	Identification and maintenance data number Possible values: • 0: I&M 0 data • 11: I&M 1 data from the parameter assignment data of the CPU • 12: I&M 2 data from the parameter assignment data of the CPU • 13: I&M 3 data from the parameter assignment data of the CPU
DATA	InOut	VARIANT	I, Q, M, D, L	Area for storing the read identification and maintenance data (see below).
DONE	Output	BOOL	I, Q, M, D, L	The instruction was executed successfully. The I&M data were transferred to the DATA parameter.

Parameter	Declaration	Data type	Memory area	Description
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Execution of the instruction completed or not yet started. • 1: Execution of the instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter The parameter is only set for the duration of one call. To display it, therefore, copy STATUS to a free data area.

You can find more information on the data types in "Overview of the valid data types (Page 2239)".

Parameter DATA for I&M 0 data

Use an array (ARRAY of BYTE) or a special data structure to store the I&M 0 data:

- If you address an array (ARRAY of BYTE) at parameter DATA, the read I&M 0 data is copied as a byte sequence to DATA. If the addressed array is longer than the read data, the unneeded bytes are filled with zeros.
- The following "IM0_Data" structure can also be used at the DATA parameter for I&M 0 data:

Parameter		Data type	Bytes	Description
Manufacturer_ID		UINT	2	Manufacturer ID (e.g. "42" for SIEMENS)
Order_ID		CHAR[20]	20	Order number
Serial_Number		CHAR[16]	16	Serial number
Hardware_Revision		UNIT	2	Hardware revision
Software_Revision		STRUCT	4	Firmware revision
	Type	CHAR	1	-
	Functional	USInt	1	-
	Bugfix	USInt	1	-
	Internal	USInt	1	-
Revision_Counter		UINT	2	Revision counter
Profile_ID		UNIT	2	Profile
Profile_Specific_Type		UNIT	2	Device class
IM_Version		UNIT	2	I&M version
I&M_Supported		UNIT	2	I&M data (I&M 0-I&M 4) supported at the device end

If a different data type is used at the DATA parameter, the STATUS parameter outputs error code 8093.

DATA parameter for I&M 1, I&M 2, and I&M 3 data

Use a string (STRING), an array (ARRAY of CHAR/BYTE) or a data structure (STRUCT) to store the I&M data:

- If you address a string (STRING data type) at the the DATA parameter, the length of the string automatically adjusts the length of the read I&M data (up to 254 characters).
- If you address a data structure (ARRAY of CHAR/BYTE or STRUCT) a the DATA parameter, the I&M data read are written to the individual components of the data type used. If the addressed data structure is longer than the read data, the remaining components are filled with zeros.
- If you crate a data structure with a STRUCT data type at the DATA parameter, use a data block without optimized block access (see category "Attributes" of the block properties).

If a data type other than STRING, ARRAY of BYTE/CHAR or STRUCT is used at the DATA parameter, the STATUS parameter generates error code 8093.

Note

Additional information about I&M data

You can find more information about I&M data, for example, on the PROFIBUS & PROFINET International websites (link: http://www.profibus.com (http://www.profibus.com)).

Parameter STATUS

Error code* (W#16#...)	Description
0	No error
7001	First call of the asynchronous instruction "Get_IM_Data". Execution of the instruction not yet complete (BUSY = 1, DONE = 0).
7002	Additional call of the asynchronous instruction "Get_IM_Data". Execution of the instruction not yet complete (BUSY = 1, DONE = 0).
8091	The device addressed at the LADDR parameter does not exist.
8092	LADDR addresses a device that does not support the output of the I&M data.
8093	Data type at the DATA parameter is not supported.
80A1	LADDR addresses a device that does not support the output of I&M 1, I&M 2 or I&M 3 data.
80B1	No valid data available. (Either the I&M data is not supported by the (sub)module or you have not configured I&M data for this device.)
80B2	Invalid value at the IM_TYPE parameter or the selected IM_TYPE is not supported by the CPU or the addressed device.
80C3	Maximum of 10 parallel executions of the instances exceeded: • Avoid more than 10 parallel calls. • Repeat the call at a later time, as this is a temporary error.

Error code* (W#16#...)	Description
8752	The memory area specified at the DATA parameter is too small to store all the I&M data. The read I&M data are stored only up to the maximum length of the specified memory area.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

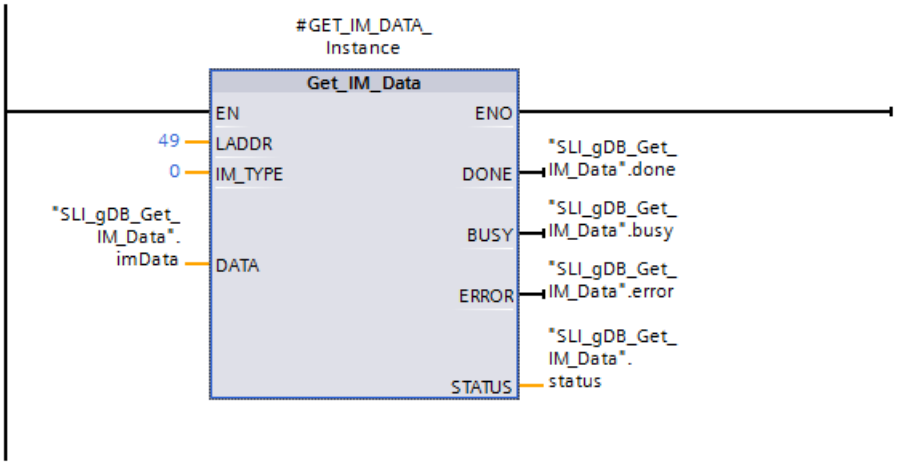
In the following example, you read out the IM0 data of an S7-1500 CPU. The IM0 data are the basic information of a device and contain information such as the manufacturer ID, order number, serial number and the hardware and firmware version.

Create four tags and a structure with the IM0_Data data type in a global data block for storing the IM0 data. You can assign any name to the structure ("MyPLC_IM_Data") in this case.

SLI_gDB_Get_IM_Data

	Name	Data type	Start value
1	Static		
2	imData	IM0_Data	
3	done	Bool	false
4	busy	Bool	false
5	error	Bool	false
6	status	Word	16#0

Enter the hardware identifier of the CPU at the LADDR parameter. The hardware identifier uniquely identifies the product. To determine the hardware identifier of your CPU, open the PLC tag table and the System constants tab. Then, search for the CPU in the Name column. The associated value is the hardware identifier that you enter at the LADDR parameter.



Once the instruction is successfully executed, the IM0 data is written to the data block.

SLI_gDB_Get_IM_Data

		Name	Data type	Start value	Monitor value
1		Static			
2		imData	IM0_Data		
3		Manufacturer_ID	UInt	0	42
4		Order_ID	String[20]	''	'6ES7 511-1AK00-0...
5		Serial_Number	String[16]	''	'S C-DOS710132013'
6		Hardware_Revision	UInt	0	3
7		Software_Revision	IM0_Version		
8		Type	Char	' '	'V'
9		Functional	USInt	0	1
10		Bugfix	USInt	0	5
11		Internal	USInt	0	0
12		Revision_Counter	UInt	0	0
13		Profile_ID	UInt	0	0
14		Profile_Specific_Ty...	UInt	0	0
15		IM_Version	Word	16#0	16#0101
16		IM_Supported	Word	16#0	16#001E
17		done	Bool	false	TRUE
18		busy	Bool	false	FALSE
19		error	Bool	false	FALSE
20		status	Word	16#0	16#0000

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

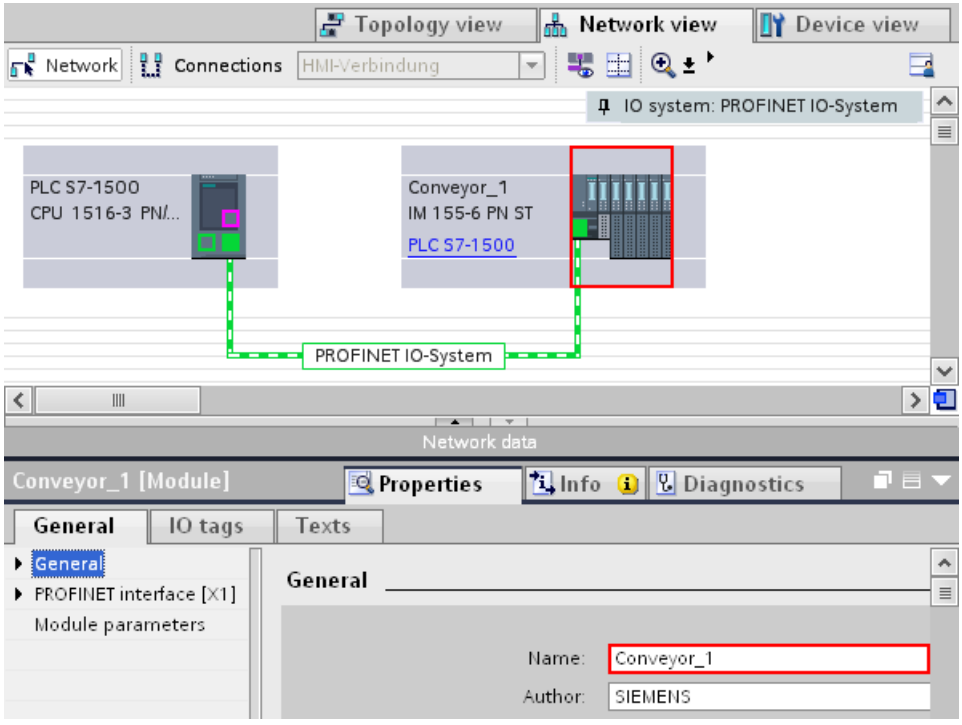
See also

Difference between synchronous and asynchronous instructions (Page 2571)

GET_NAME: Read out name of an IO device or a DP slave (S7-1200, S7-1500)

Description

The instruction "GET_NAME" reads out the name of an IO device or a DP slave. The name is displayed in the network view and in the properties of the IO device or the DP slave. The following screenshot shows this using an IO device as example.



You select the IO device or the DP slave by using the hardware identifier of the PROFINET IO system or the DP master system (at the LADDR parameter) and the device number of the IO device or the PROFIBUS address of the DP slave (STATION_NR parameter).

After the instruction has been executed successfully, the name of the IO device or the DP slave is written to the area addressed with the DATA parameter.

The name that is read depends on the type of IO device:

- For a DP slave or IO device, the name of the head module is output.
- For an I-slave or I-device, the name of the interface module is output.
- The name of the interface is output for an HMI panel.
- The name of the interface module is output for a PC station.
- The name of the Device Access Point (DAP) is displayed (name of the interface or head module) for GSD devices.

The length of the name is output at the LEN parameter. If the name is longer than the area specified at the DATA parameter, only that section which corresponds to the maximum length of the addressed area will be written.

The maximum length for a name is 128 characters.

Note

Name of the CPU readout (Version 1.1)

If you assign a "0" at each of the parameters LADDR and STATION_NR the instruction will output the name of the CPU.

Parameters

The following table shows the parameters of the "GET_NAME" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_IOSYSTEM	I, Q, M, D, L or constant	Hardware identifier of the PROFINET IO or DP master system. The number is taken from the system constants or the properties of the PROFINET IO system or DP master system.
STATION_NR	Input	UINT	I, Q, M, D, L or constant	Device number of the PROFINET IO device or PROFIBUS address of the DP slave. This number is displayed in the network view and in the properties of the IO device or the DP slave.
DATA	InOut	VARIANT	I, Q, M, D, L	Pointer to the area to which the name is written.
DONE	Output	BOOL	I, Q, M, D, L	The instruction was executed successfully. The name of the IO device or the DP slave was transferred to the area at the DATA parameter.
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Execution of the instruction complete. • 1: Execution of the instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
LEN	Output	DINT	I, Q, M, D, L	Length of the name of the IO device or the DP slave (number of characters).
STATUS	Output	WORD	I, Q, M, D, L	Status parameter The parameter is only set for the duration of one call. To display the status, therefore, copy STATUS to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

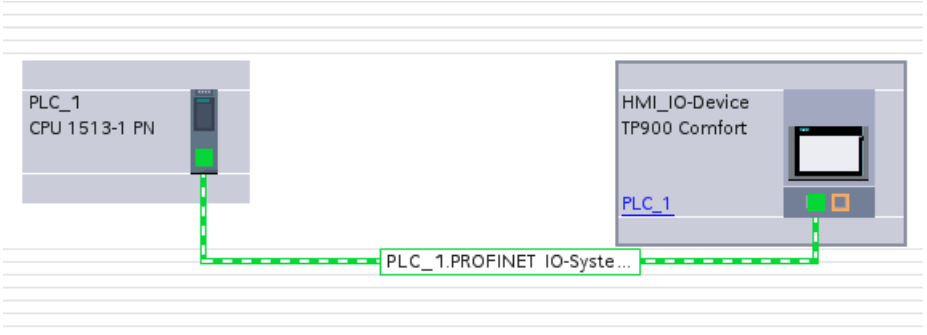
Error code* (W#16#...)	Explanation
0	No error
7001	First call of the asynchronous instruction "GET_NAME". Execution of the instruction not yet complete (BUSY = 1, DONE = 0).
7002	Additional call of the asynchronous instruction "GET_NAME". Execution of the instruction not yet complete (BUSY = 1, DONE = 0).
8090	The hardware identifier specified at the LADDR parameter does not exist in the project.
8092	The value at the LADDR parameter does not address a PROFINET IO system or a DP master system.
8093	Instruction does not support data type at the DATA parameter.
8095	• Device number (STATION_NR parameter) does not exist in the selected PROFINET IO system or does not address an IO device. • PROFIBUS address (STATION_NR parameter) does not exist in the selected DP master system.
80B1	The CPU used does not support the instruction.
80C3	Temporary resource error: The CPU is currently processing the maximum possible number of simultaneous block calls. "GET_NAME" cannot be executed until at least one of the block calls is finished.
8852	The area specified at the DATA parameter is too short for the full name of the IO device or the DP slave. The name will be written up to the maximum possible length. To read the full name, use a longer data area at the DATA parameter. The area must have at least as many characters as there are at the LEN parameter.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

The following example shows how you can read out the station name of a HMI panel.

Configuration of the HMI panel:

- The HMI panel with the station name "HMI_IO-Device" is created in the network view and assigned to the same PROFINET IO system as the CPU.
- The operating mode "IO device" was activated in the properties of the hardware configuration for the HMI panel and the CPU was assigned as the IO controller.
- The device number "20" assigned in the properties under "Ethernet addresses".



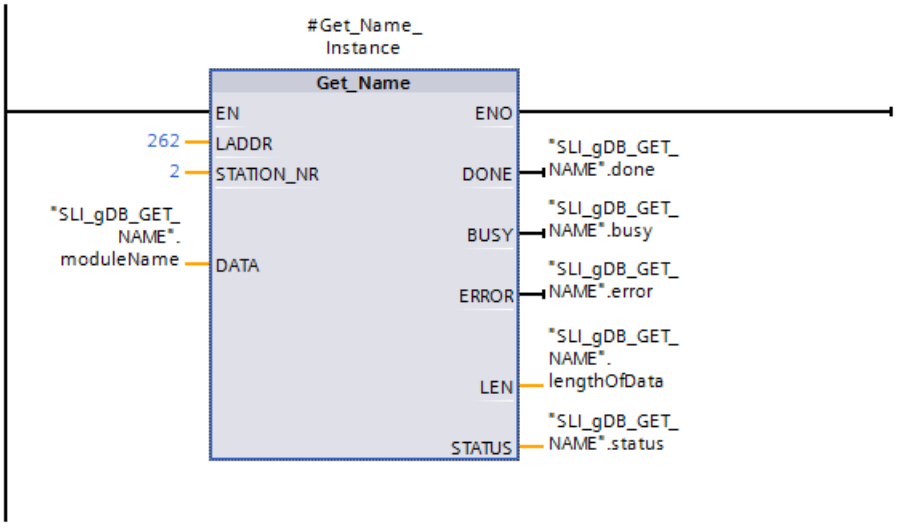
Create the following tags for storing the data in a global data block:

SLI_gDB_GET_NAME

	Name	Data type	Start value
1	Static		
2	moduleName	String	''
3	done	Bool	false
4	busy	Bool	false
5	error	Bool	false
6	lengthOfData	DInt	0
7	status	Word	16#0

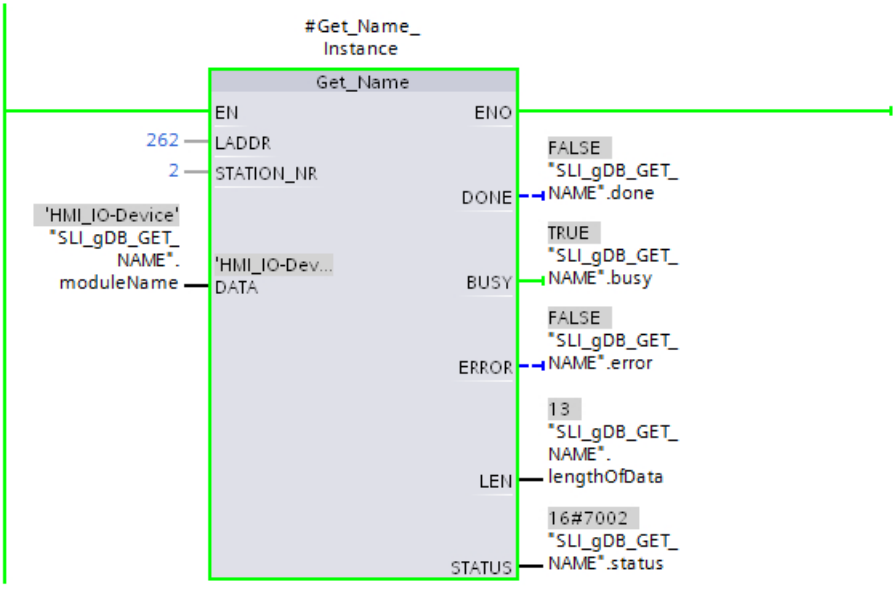
Assigning parameters to the instruction "GET_NAME":

- Enter the hardware identifier of the IO system (262) at the input parameter LADDR.
- Enter the device number of the HMI panel (2) at the input parameter STATION_NR.
- Interconnect the remaining parameters of the "GET_NAME" instruction as follows.



Execution of the instruction:

- After execution of the instruction, the name of the station of the HMI panel (HMI_IO-Device) is stored at parameter DATA ("moduleName").
- The number of characters of the name is output (13) at the parameter LEN.
- As GET_NAME was not stopped after the successful reading of the IO device, output parameter BUSY ("busy") shows the value "TRUE" and output parameter STATUS ("status") shows the value "16#7002".



Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

GetStationInfo: Read information of an IO device (S7-1200, S7-1500)

Description

You can use the "GetStationInfo" instruction to read information of a PROFINET IO device. The instruction also enables you to read the information of an IO device located in a lower-level IO system (connected via CP/CM).

The IO device is addressed via the hardware identifier of the station at the LADDR parameter. The hardware identifier is displayed in the "Devices & Networks" view in the station properties.

Use the MODE parameter to select the information to be read.

At the DATA parameter, specify the data area to which the read address data will be written. For storing the IP address, use the "IF_CONF_v4" structure. For storing the MAC address, use the "IF_CONF_MAC" structure.

Enable reading of the address data using the REQ control parameter. This requires the IO device to be accessible.

The execution status of the read job is displayed via the BUSY, DONE, ERROR output parameters and the STATUS output parameter.

Note

Address the IO device using only the hardware identifier of the station

The station, the IO device and PROFINET interface have their own hardware identifier. Use only the hardware identifier of the station for the "GetStationInfo" instruction.

If a PROFINET interface is addressed via the LADDR parameter, for example, the address data is not read and the 8092 error code is generated.

To read the address data of an integrated PROFINET interface or a CM/CP in the central configuration, use the "RDREC" instruction.

Parameters

The following table shows the parameters of the "GetStationInfo" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L	Control parameter Request Activates the reading of the information with REQ = "1".
LADDR	Input	HW_DEVICE	I, Q, M, D, L or constant	Hardware identifier of the station of the IO device The number is taken from the properties of the station in the Network view or from the "System constants" tab of the default tag table.
DETAIL	Input	HW_SUBMODULE	I, Q, M, D, L or constant	The DETAIL parameter is not used. Leave the parameter unconnected.
MODE	Input	UINT	I, Q, M, D, L or constant	Selection of address data to be read: • MODE = 1: Address parameter according to IPv4 (S7-1500 CPUs as of firmware version V1.1) • MODE = 2: MAC address (S7-1500 CPUs as of firmware version V1.5)
DATA	InOut	VARIANT	D, L	Pointer to the area to which the address data of the IO device is written. Use the "IF_CONF_v4" structure for MODE = 1 and the "IF_CONF_MAC" structure for MODE = 2.
DONE	Output	BOOL	I, Q, M, D, L	The instruction was executed successfully. The address data was transferred to the DATA parameter.
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Execution of the instruction complete. • 1: Execution of the instruction not yet complete.

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy STATUS to a free data area.

You can find more information on the data types in "Overview of the valid data types (Page 2239)".

Parameter DATA

- Use the "IF_CONF_v4" structure at the DATA parameter to store the address parameter according to IPv4.

Byte	Parameter	Data type	Start value	Description
0 … 1	Id	UINT	30	ID of the "IF_CONF_v4" structure.
2 … 3	Length	UNIT	18	Length of data read in BYTE.
4 ... 5	Mode	UNIT	0	Not relevant for the "GetStationInfo" instruction (left at "0").
6 ... 9	InterfaceAddress	ARRAY [1..4] of BYTE	-	IP address of the IO device in the format IP_V4, e.g. for 192.168.3.10: • addr[1] = 192 • addr[2] = 168 • addr[3] = 3 • addr[4] = 10
10 … 13	SubnetMask	ARRAY [1..4] of BYTE	-	Subnet mask of the IO device in the format IP_V4, e.g. for 255.255.255.0: • addr[1] = 255 • addr[2] = 255 • addr[3] = 255 • addr[4] = 0
14 ... 17	DefaultRouter	ARRAY [1..4] of BYTE	-	IP address of the router in the format IP_V4, e.g. for 192.168.3.1: • addr[1] = 192 • addr[2] = 168 • addr[3] = 3 • addr[4] = 1

- Use the "IF_CONF_MAC" structure at parameter DATA for storing the MAC address.

Byte	Parameter	Data type	Start value	Description
0 … 1	Id	UINT	3	ID of the "IF_CONF_MAC" structure.
2 … 3	Length	UNIT	12	Length of data read in BYTE.
4 ... 5	Mode	UNIT	0	Not relevant for the "GetStationInfo" instruction (left at "0").
6 ... 11	MACAddress	ARRAY [1..6] of BYTE	-	MAC address of the IO device, e.g. for 08-00-06-12-34-56 • Mac[1] = 8 • Mac[2] = 0 • Mac[3] = 6 • Mac[4] = 12 • Mac[5] = 34 • Mac[6] = 56

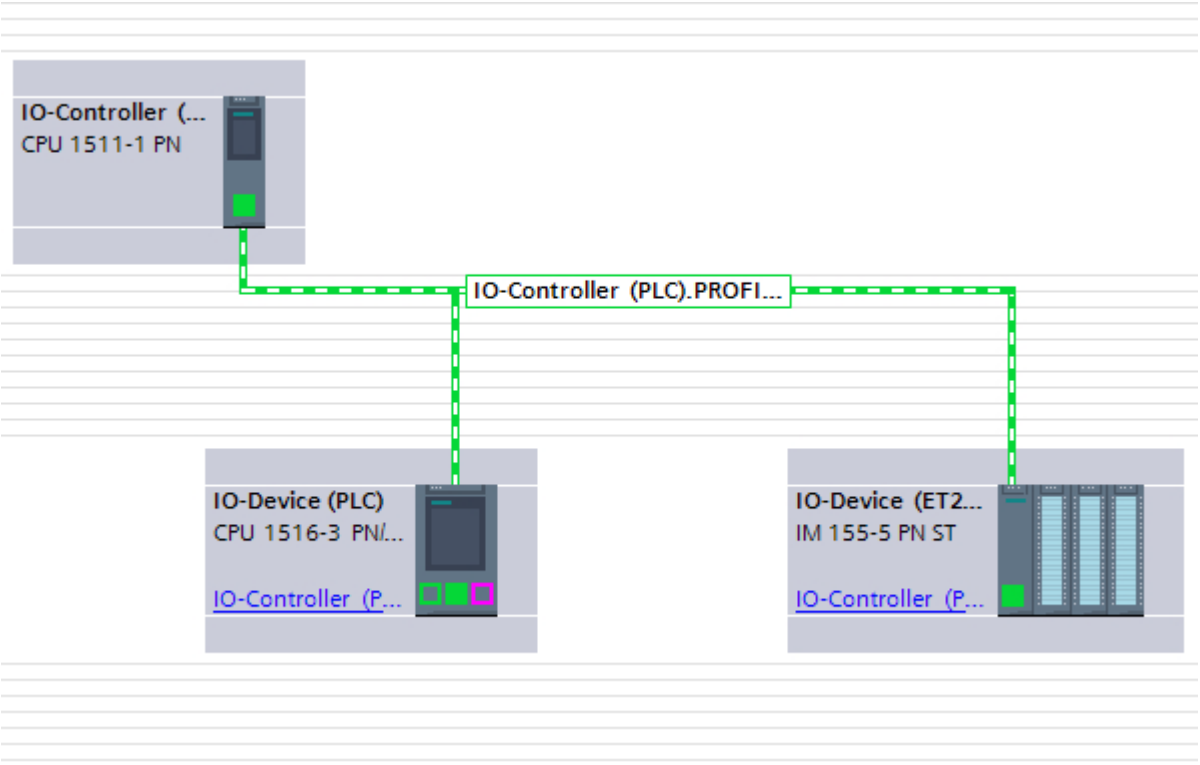
Parameter STATUS

Error code* (W#16#...)	Explanation
0	No error
7000	No job in progress.
7001	First call of the asynchronous instruction "GetStationInfo". Execution of the instruction not yet complete (BUSY = 1, DONE = 0).
7002	Additional call of the asynchronous instruction "GetStationInfo". Execution of the instruction not yet complete (BUSY = 1, DONE = 0).
8080	Value at the MODE parameter is not supported.
8090	The hardware identifier specified at the LADDR parameter is not configured.
8092	The LADDR parameter does not address a PROFINET IO device.
8093	Invalid data type at the DATA parameter.
80A0	Requested information is not read.
80C0	Addressed IO device is not reachable.
80C3	The maximum number of simultaneous calls of the "GetStationInfo" instruction (10 instances) has been reached.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

Below, you use the „GetStationInfo“ instruction to read the IP address data of an IO device and write the information to a data block. The IP address data includes the IP address, subnet mask and (if used) the address data of the router.

The instruction is executed on the IO controller and reads the IP address information of a lower-level IO device (an ET200MP in this example).



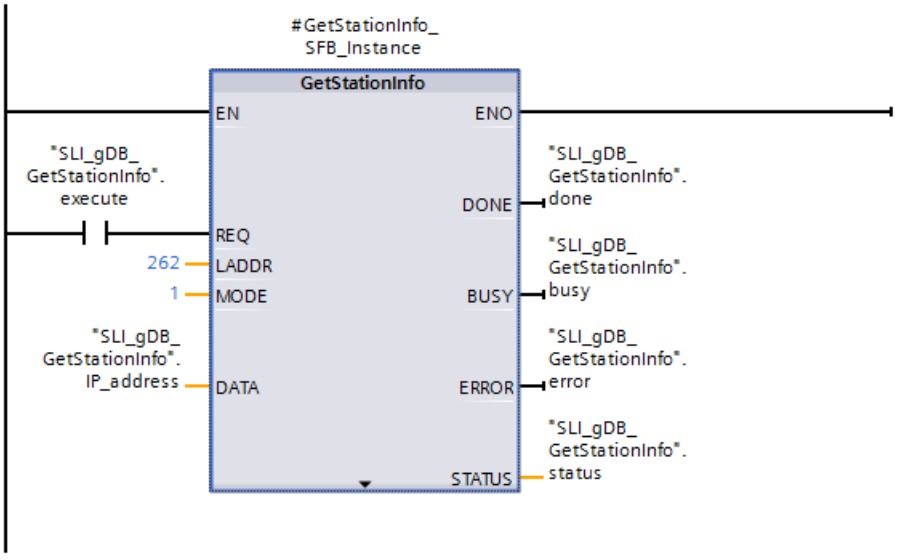
Create five tags and a structure with the IF_CONF_v4 data type in a global data block for storing the address data. Assign any name to the structure (in this case "IP_Address").

SLI_gDB_GetStationInfo

		Name	Data type	Start value
1		Static		
2		execute	Bool	false
3		IP_address	IF_CONF_v4	
4		done	Bool	false
5		busy	Bool	false
6		error	Bool	false
7		status	Word	16#0

Then call the "GetStationInfo" instruction:

- You use the IF_CONF_v4 structure at the DATA parameter.
- Enter the hardware identifier of the IO device at the LADDR parameter. The hardware identifier uniquely identifies the product. To determine the hardware identifier of your IO device, open the PLC tag table and the System constants tab. Then, search for the device in the Name column and for the „Hw_Device“ in the Data type column. The associated value is the hardware identifier that you enter at the LADDR parameter.
- Select "1" (read address parameters according to IPv4) for the MODE parameter.



You start reading the address data with REQ= 1 (TRUE). Once the instruction is successfully executed, the IP address data is written to the data block.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

GetChecksum: Read out checksum (S7-1200, S7-1500)

Description

The "GetChecksum" instruction reads the checksum of a group of objects. The objects are selected with the parameter "Scope".

Note

Generation of the checksum for blocks

There are differences between standard and safety blocks in terms of the generation of the checksum. You can find additional information in the help for the STEP 7 Safety option package.

Parameter

The following table shows the parameters of the "GetChecksum" instruction:

Parameter	Declaration	Data type	Description
Scope	Input	UINT	With the Scope parameter, you select the objects for which you want to read out the checksum: • 1: Standard blocks (i.e. no safety blocks) • 2: Safety blocks • 3: Text lists
Range (hidden)	Input	UINT	"Range" must have the value 0.
Subrange (hidden)	Input	UINT	"Subrange" must have the value 0.
Done	Output	BOOL	The instruction was executed successfully. The checksum was transferred to the Checksum parameter.
Busy	Output	BOOL	Status parameter • 0: Execution of the instruction complete. • 1: Execution of the instruction not yet complete.
Error	Output	BOOL	Status parameter • 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the "Status" parameter.

Parameter	Declaration	Data type	Description
Status	Output	WORD	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy STATUS to a free data area.
Checksum	InOut	VARIANT	Checksum The "Checksum" parameter must point to a tag of the data type ARRAY[0..7] of BYTE; unused bytes are set to 0.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter "Status"

Error code* (W#16#...)	Explanation
0	No error
7001	First call: Generation of the checksum initiated
7002	Intermediate call: Generation of the checksum already active
8080	Invalid value of the "Scope" parameter
8081	The "Range" parameter is not supported in this firmware version and must have the value 0.
8082	The "Subrange" parameter is not supported in this firmware version and must have the value 0.
8091	The "GetChecksum" instruction is not supported by the CPU.
8852	The tag to which the "Checksum" parameter points has the wrong data type.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

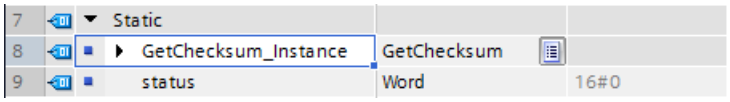
In the following example, you read out the checksum of the standard blocks of a S7-1500 CPU.

Create the following tags for storing the data in a global data block:

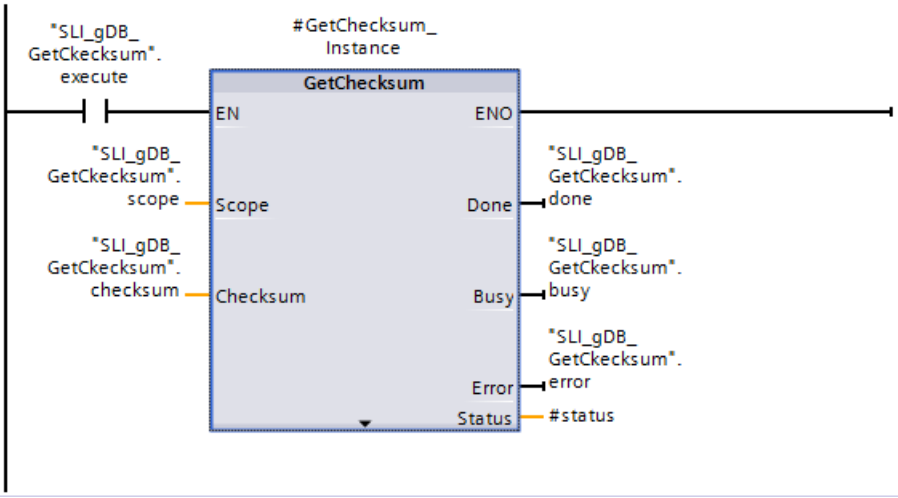
SLI_gDB_GetChecksum

	Name	Data type	Start value
1	Static		
2	execute	Bool	false
3	scope	UInt	1
4	checksum	Array[0..7] of Byte	
5	busy	Bool	false
6	done	Bool	false
7	error	Bool	false
8	memErrStatus	Word	16#0

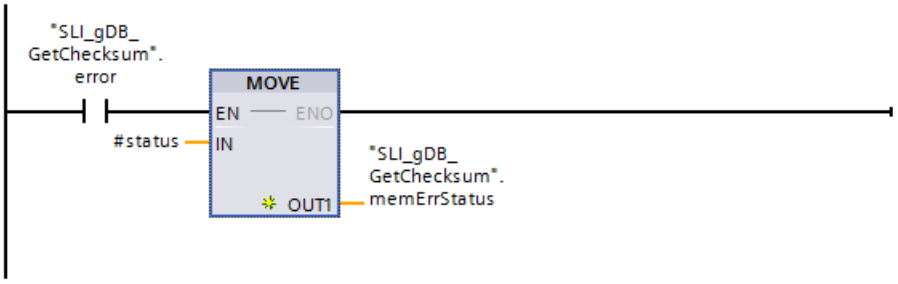
Create an FB. Create the following local tags in the FB.



Network 1: Interconnect the parameters of the "GetChecksum" instruction as follows.



Network 2: If an error occurs, save the status as follows.



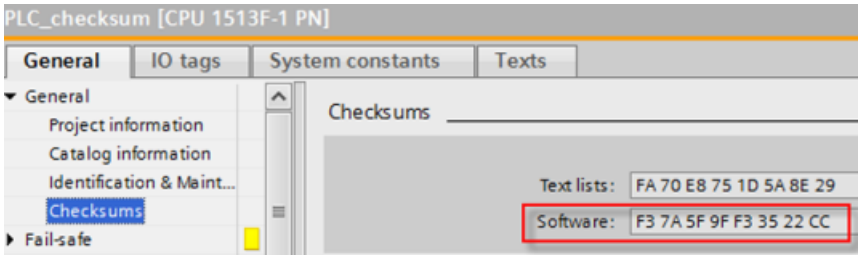
If the normally open contact ("execute") supplies the signal state "TRUE", the "GetChecksum" instruction is executed. The checksum of the standard blocks is read according to the value "1" of the input parameter SCOPE ("scope"). The checksum is output at the CHECKSUM ("checksum") parameter. The success status of GetChecksum is displayed at the output parameter DONE ("done").

Output parameters STATUS ("#status" and "memErrStatus") and ERROR ("error") indicate that the processing in the example was completed without error.

SLI_gDB_GetCkecksum

	Name	Data type	Start value	Monitor value
1	Static			
2	execute	Bool	false	TRUE
3	scope	UInt	1	1
4	checksum	Array[0..7] of Byte		
5	checksum[0]	Byte	16#0	16#F3
6	checksum[1]	Byte	16#0	16#7A
7	checksum[2]	Byte	16#0	16#5F
8	checksum[3]	Byte	16#0	16#9F
9	checksum[4]	Byte	16#0	16#F3
10	checksum[5]	Byte	16#0	16#35
11	checksum[6]	Byte	16#0	16#22
12	checksum[7]	Byte	16#0	16#CC
13	busy	Bool	false	FALSE
14	done	Bool	false	TRUE
15	error	Bool	false	FALSE
16	memErrStatus	Word	16#0	16#0000

You can also find the checksum in the CPU properties.



Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Comparison of PLC programs based on checksums (Page 5086)

GetSMCinfo: Reading out information about the SIMATIC memory card (S7-1200, S7-1500)

Description

Use the instruction "GetSMCinfo" to retrieve information about the inserted SIMATIC memory card. The information to be read out is selected with the parameter "Mode".

Enable reading of the address data via the REQ control parameter. The output parameters Done, Busy, Error and Status indicate the status of the job.

Parameter

The following table shows the parameters of the instruction "GetSMCinfo":

Parameter	Declaration	Data type	Description
REQ	Input	BOOL	Control parameter Request Activates the reading of the information with REQ = "1".
Mode	Input	UINT	Use the Mode parameter to select the information about the SIMATIC memory card that you want to read out: • 0: Memory size in KB (1 KB = 1024 bytes) • 1: Assigned memory space in KB • 2: Maintenance information: Previously used up portion of the service life in % Note: – With the S7-1500 software controllers and S7-PLCSIM, 0x00 is always entered in Info for Mode=2 (no maintenance information available). – With SIMATIC Memory Cards, which do not generally support this information or which cannot supply this information with the current firmware of your CPU, 0xFF is entered in Info. • 3: Set proportion of the service life as a percentage after which the CPU creates a diagnostics buffer entry and activates the maintenance LED and a diagnostics interrupt. Note: This information is read from the memory card, but instead from the planning data. If you have disabled the generation of diagnostics interrupt, 0xFF is entered in Info.
Done	Output	BOOL	1: The instruction has been executed successfully. The information that was read is transferred to the Info parameter.
Busy	Output	BOOL	Status parameter • 0: Execution of the instruction complete. • 1: Execution of the instruction not yet complete.
Error	Output	BOOL	Status parameter • 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the Status parameter.
Status	Output	INT	Error code
Info	InOut	UDINT	Buffer for the information that was read.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

Determining the I&M 0 data

The I&M 0 data of the SIMATIC memory card cannot be determined with the GetSMCinfo instruction. For this use the instruction "Get_IM_Data (Page 3947)".

Parameter Status

Error code* (W#16#...)	Explanation
0	No error.
7000	No job in progress.
7001	First call: Trigger job (Busy = 1, Done = 0).
7002	Intermediate call: Job already active (Busy = 1, Done = 0).
8080	Invalid value of the "Mode" parameter
8081 (only S7-1200)	No SIMATIC memory card inserted.
8092	No data available, for example, because "GetSMCinfo" is not supported by the CPU.
80C3	The maximum number of simultaneous calls of the "GetSMCinfo" instruction has been reached.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

In the following example, you determine the memory size of the SIMATIC memory card used.

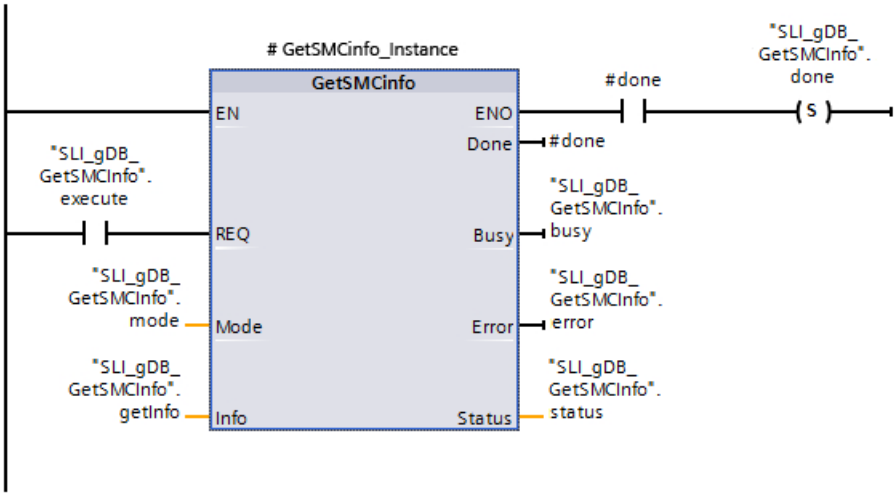
Create the following tags for storing the data in a global data block:

SLI_gDB_GetSMCinfo

	Name	Data type	Start value
1	Static		
2	execute	Bool	false
3	mode	UInt	0
4	getInfo	UDInt	0
5	done	Bool	false
6	busy	Bool	false
7	error	Bool	false
8	status	Word	16#0

Create an FB. In the status area of the FB, create a local tag"#done" with the Bool data type.

Interconnect the parameters of the "GetSMCinfo" instruction as follows.



The GetSMCinfo instruction is executed only when the input parameter REQ ("execute") returns the signal state "TRUE". The mode to be used for reading out the SIMATIC Memory Card is stored in the input parameter MODE ("mode"). The memory size of the SIMATIC memory card is read according to the value "0" and displayed in KB at parameter INFO ("getInfo"). The success status of GetSMCinfo is displayed at the output parameter DONE ("#done") and stored in the tag "done".

Output parameters STATUS ("status") and ERROR ("error") indicate that the processing in the example was completed without error.

SLI_gDB_GetSMCinfo

	Name	Data type	Start value	Monitor value
1	Static			
2	execute	Bool	false	TRUE
3	mode	UInt	0	0
4	getInfo	UDInt	0	2025008
5	done	Bool	false	TRUE
6	busy	Bool	false	FALSE
7	error	Bool	false	FALSE
8	status	Word	16#0	16#0000

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

DeviceStates: Read module status information in an IO system (S7-1200, S7-1500)

Description

You use the instruction "DeviceStates" to query specific status information for all modules in an IO system, which means:

- Either for all IO devices in a PROFINET IO system
- Or for all DP slaves in a DP master system

The Boolean value that is output indicates the modules to which the selected status applies. For example, read out which IO devices are currently disabled in a PROFINET IO system.

Information is also displayed as to whether the status information to be read applies to at least one of the IO devices or DP slaves.

The instruction can be called in a cyclic OB as well as in an interrupt OB (e.g. OB82 - diagnostic interrupt).

Parameters

The following table shows the parameters of the "DeviceStates" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_IOSYSTEM	I, Q, M, L or constant	Hardware identifier of the PROFINET IO or DP master system (see description below)
MODE	Input	UINT	I, Q, M, D, L or constant	Selection of status information to be read (see description below)
RET_VAL	Return	INT	I, Q, M, D, L	Status of instruction (see description below)
STATE	InOut	VARIANT	I, Q, M, D, L	Buffer for status of the IO devices or the DP slaves (see description below)

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter LADDR

You select the PROFINET IO or DP master system at the LADDR parameter by means of the hardware identifier.

The hardware identifier is available:

- Either in the Network view of the properties of the PROFINET IO or DP master system.
- Or in the PLC tag table in the listed system constants with the data type HW_IOSYSTEM.

Parameter MODE

You use the MODE parameter to read out status information. You can read out one of the following status information items for the entire PROFINET IO or DP master system:

- 1: IO devices/DP slaves are configured
- 2: IO devices/DP slaves are faulty
- 3: IO devices/DP slaves are disabled
- 4: IO devices/DP slaves exist
- 5: IO devices/DP slaves for which a problem has occurred. For example:
  - Maintenance demanded or recommended
  - Not accessible
  - Not available
  - Error occurred

Parameter STATE

With the STATE parameter, the status of the IO devices/DP slaves that was selected with the MODE parameter is output.

If the status selected using MODE applies to an IO device/DP slave, the following bits are set to "1" at the STATE parameter:

- Bit 0 = 1: Group display. The bit n of at least one IO device/DP slave was set to "1".
- Bit n = 1: The status selected with MODE applies to the IO device/DP slave.
  - With a PROFINET IO system, bit n corresponds to the device number of the respective IO device (see properties of the PROFINET interface in the device view and network view)
  - With a PROFIBUS DP system, bit n corresponds to the PROFIBUS address of the DP slave (see properties of the DP slave in the device view and network view)

Use "BOOL" or "Array of BOOL" as data type:

- To only output the bit for group display of the status information, you can use the data type BOOL at the STATE parameter.
- To output the status information for all IO devices/DP slaves, use Array of BOOL with the following length:
  - With PROFINET IO system: 1024 bits
  - With DP master system: 128 bits

Parameter RET_VAL

Error code* (W#16#...)	Description
0	No error
8091	Hardware identifier of the LADDR parameter does not exist. Check (for example, in the system constants) whether the value for LADDR exists in the project.
8092	LADDR does not address a PROFINET IO or DP master system.
8093	Invalid data type at the STATE parameter.
80B1	Instruction "DeviceStates" is not supported by the CPU.
80B2	The selected MODE parameter is not supported by the used CPU for the IO system specified in the LADDR parameter.
8452	The complete status information does not fit in the tag configured in the STATE parameter. Note: When the tag configured at STATE is checked for adequate field length, the CountOfElements instruction is called. This counts the fill elements when the data type VARIANT points to an Array of BOOL; with an Array [0...120] of BOOL e.g. 128 is determined as the field length. This has the consequence that the error code W#16#8452 is only returned by DeviceStates when the number of field elements you have set plus the number of fill elements created by the CPU falls below the value 1024 or 128
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example - Reading out the presence of IO devices in the PROFINET IO master system

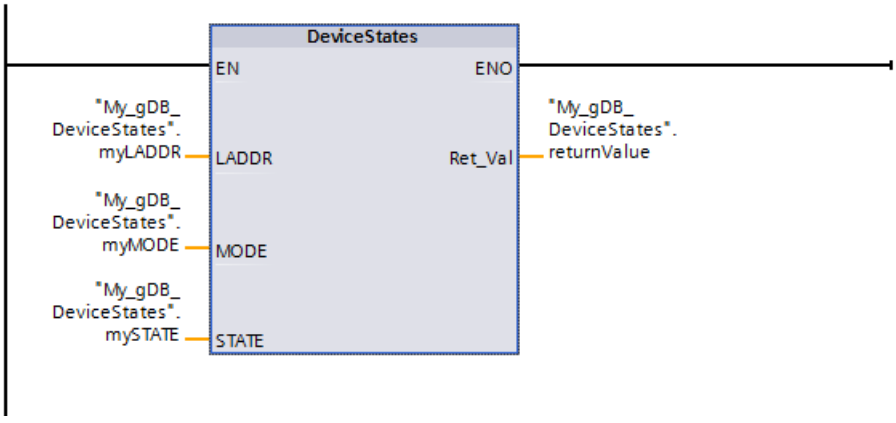
In the following example, you query the existence of IO devices in an IO system. The IO system consists of two CPUs of the S7-1500 series. The "PLC_14" CPU will contain the program, including the "DeviceStates" instruction. The "PLC_13" CPU will be configured as an IO device.

In the "PLC_14" CPU: Create three tags and a "mySTATE" structure (with the Array of BOOL data type) in a global data block for storing the data.

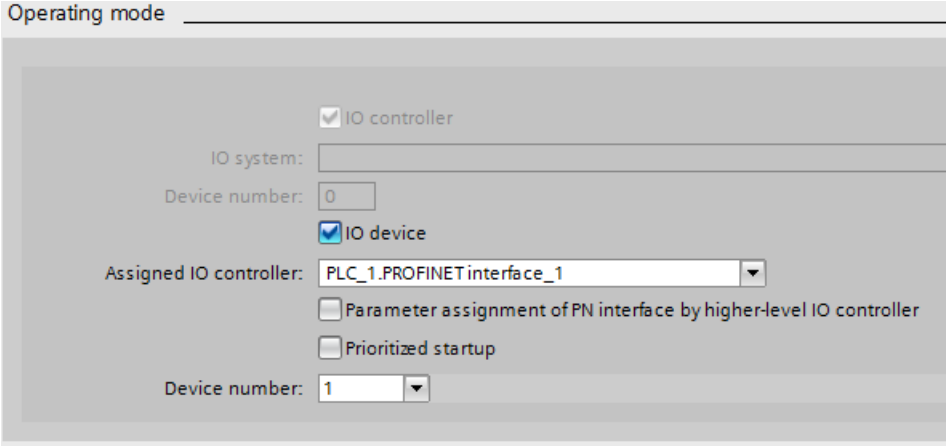
My_gDB_DeviceStates

		Name	Data type	Start value
1		Static		
2		myLADDR	HW_IOSYSTEM	258
3		myMODE	UInt	4
4		returnValue	Int	0
5		mySTATE	Array[0..1023] ...	
6		mySTATE[0]	Bool	false
7		mySTATE[1]	Bool	false
8		mySTATE[2]	Bool	false
9		mySTATE[3]	Bool	false
10		mySTATE[4]	Bool	false
11		mySTATE[5]	Bool	false
12		mySTATE[6]	Bool	false

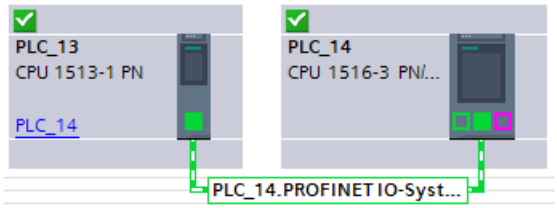
In the "PLC_14" CPU: The instruction is called in a cyclic OB. Interconnect the parameters of the instruction as follows.



In the "PLC_13" CPU: Set up this CPU "PLC_13" as an IO device using the CPU properties. The IO device receives the device number 1.



An IO system is displayed in the Network view.



In the "PLC_14" CPU: The HW identifier of the IO system is made known to the "DeviceStates" instruction through parameter LADDR ("myLADDR"). According to the value "4" of parameter MODE ("myMODE"), the IO system is searched for IO devices.

At parameter STATE ("mySTATE"), the presence is output for the IO devices (based on the value of parameter MODE). Bit 0 serves as a group value and indicates that IO devices are present. Bit 1 indicates that the IO device with the device number 1 is present.

The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.

My_gDB_DeviceStates

	Name	Data type	Start value	Monitor value
1	Static			
2	myLADDR	HW_IOSYSTEM	258	16#0102
3	myMODE	UInt	4	4
4	returnValue	Int	0	0
5	mySTATE	Array[0..1023] ...		
6	mySTATE[0]	Bool	false	TRUE
7	mySTATE[1]	Bool	false	TRUE
8	mySTATE[2]	Bool	false	FALSE
9	mySTATE[3]	Bool	false	FALSE
10	mySTATE[4]	Bool	false	FALSE
11	mySTATE[5]	Bool	false	FALSE
12	mySTATE[6]	Bool	false	FALSE

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Example - Reading the faulty stations of a PROFINET IO master system

A PROFINET IO system consists of 4 IO devices with the device numbers 1, 2, 3 and 4. The IO device with number 2 is faulty.

The instruction "DeviceStates" is executed for the PROFINET IO system with MODE = 2 (faulty/ not faulty).

The following bits are set in the STATE parameter:

- Bit 0 = 1: A fault exists for at least one of the IO devices.
- Bit 1 = 0: IO device with device number 1 is not faulty.
- Bit 2 = 1: IO device with device number 2 is faulty.
- Bit 3 = 0: IO device with device number 3 is not faulty.
- Bit 4 = 0: IO device with device number 4 is not faulty.

- Bit 5 = 0: Irrelevant
- Bit 6 = 0: Irrelevant
- ...

Example - Reading the faulty stations of a PROFIBUS DP master system

A DP master system consists of 4 DP slaves with the PROFIBUS addresses 3, 4, 5 and 6. The DP slave with address 4 is faulty.

The instruction "DeviceStates" is executed for the DP master system with MODE = 2 (faulty/ not faulty).

The following bits are set in the STATE parameter:

- Bit 0 = 1: A fault exists for at least one of the DP slaves.
- Bit 1 = 0: Irrelevant
- Bit 2 = 0: Irrelevant
- Bit 3 = 0: DP slave with address 3 is not faulty.
- Bit 4 = 1: DP slave with address 4 is faulty.
- Bit 5 = 0: DP slave with address 5 is not faulty.
- Bit 6 = 0: DP slave with address 6 is not faulty.
- Bit 7 = 0: Irrelevant
- Bit 8 = 0: Irrelevant
- ...

ModuleStates: Read module status information of a module (S7-1200, S7-1500)

Description

You can use the "ModuleStates" instruction to read the status information of the modules of a PROFINET IO device or PROFIBUS DP slave.

The Boolean value that is output indicates the modules to which the selected status applies. You can, for example, read which modules are currently disabled in a PROFINET IO device.

Information is also displayed as to whether the status information to be read applies to at least one of the modules.

The instruction can be called in a cyclic OB as well as in an interrupt OB (e.g. OB82 - diagnostic interrupt).

Parameters

The following table shows the parameters of the "ModuleStates" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_DEVICE	I, Q, M, D, L or constant	Hardware identifier of the station (see description below)
MODE	Input	UINT	I, Q, M, D, L or constant	Selection of module status information to be read (see description below)
RET_VAL	Return	INT	I, Q, M, D, L	Status of instruction (see description below)
STATE	InOut	VARIANT	I, Q, M, D, L	Buffer for the module status (see description below)

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter LADDR

You select the IO device or the DP slave at the LADDR parameter by means of the hardware identifier of the station.

The hardware identifier is available:

- Either in the network view of the properties of the IO device station or DP slave station.
- Or in the PLC tag table for the listed system constants with the data type HW_DEVICE (for an IO device) or with the data type HW_DPSLAVE (for a DP slave).

Parameter MODE

You use the MODE parameter to read out status information. One of the following status information items can be read for the modules:

- 1: Modules are configured
- 2: Modules are faulty
- 3: Modules are disabled
- 4: Modules exist
- 5: There is a problem in the modules. For example:
  - Maintenance demanded or recommended
  - Not accessible
  - Not available
  - Error occurred

Parameter STATE

The STATE parameter outputs the status of the modules selected with the MODE parameter.

If the status selected using MODE applies to a module, the following bits are set to "1":

- Bit 0 = 1: Group display. The bit n of at least one module was set to "1".
- Bit n = 1: The status selected with MODE applies to the module in slot n-1 (example: bit 3 = slot 2).

Use "BOOL" or "Array of BOOL" as data type:

- To only output the bit for group display of the status information, use the BOOL data type at the STATE parameter.
- To output the status information for all modules, use Array of BOOL with a length of 128 bits.

Parameter RET_VAL

Error code* (W#16#...)	Description
0	No error
8091	Hardware identifier of the LADDR parameter does not exist. Check (for example, in the system constants) whether the value for LADDR exists in the project.
8092	LADDR does not address an IO device or DP slave.
8093	Invalid data type at the STATE parameter.
80B1	The instruction "ModuleStates" is not supported by the CPU.
80B2	The selected MODE parameter is not supported by the used CPU for the IO device/the DP slave in the LADDR parameter.
8452	The complete status information does not fit in the tag configured in the STATE parameter. Note: When the tag configured at STATE is checked for adequate field length, the CountOfElements instruction is called. This counts the fill elements when the data type VARIANT points to an Array of BOOL; with an Array [0...120] of BOOL e.g. 128 is determined as the field length. This has the consequence that the error code W#16#8452 is only returned by ModuleStates when the number of field elements you have set plus the number of fill elements created by the CPU falls below the value 128
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

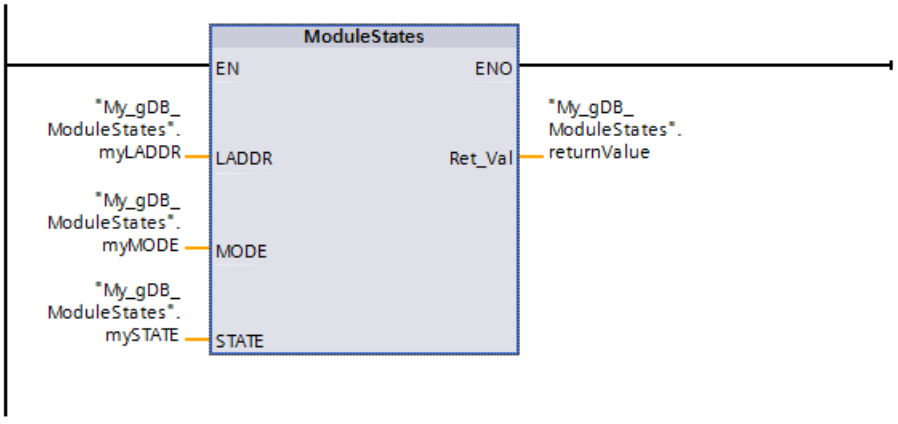
In the following example, you query the existence of modules of a PROFINET IO device. The IO system consists of two CPUs of the S7-1500 series. The "PLC_14" CPU will contain the program, including the "ModuleStates" instruction. The "PLC_13" CPU will be configured as an IO device.

In the "PLC_14" CPU: Create three tags and a "mySTATE" structure (with the Array of BOOL data type) in a global data block for storing the data.

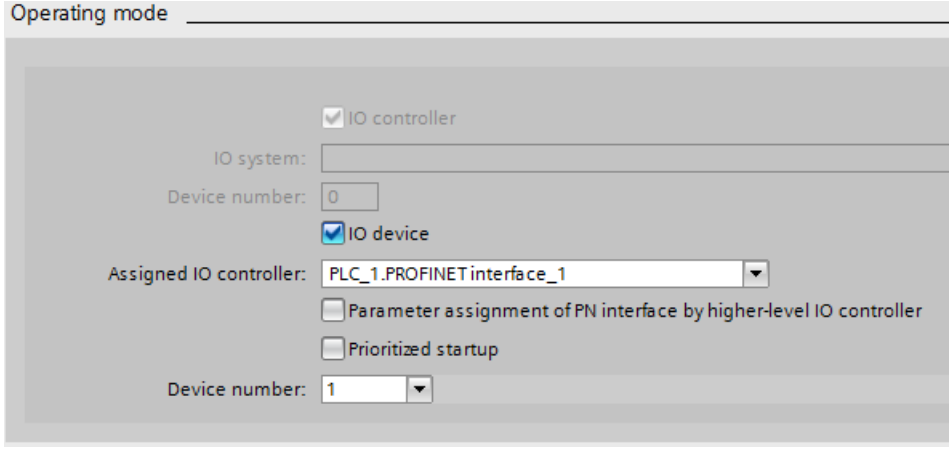
My_gDB_ModuleStates

	Name	Data type	Start value
1	Static		
2	myLADDR	HW_DEVICE	260
3	myMODE	UInt	4
4	returnValue	Int	0
5	mySTATE	Array[0..127] of Bool	
6	mySTATE[0]	Bool	false
7	mySTATE[1]	Bool	false
8	mySTATE[2]	Bool	false
9	mySTATE[3]	Bool	false
10	mySTATE[4]	Bool	false
11	mySTATE[5]	Bool	false
12	mySTATE[6]	Bool	false

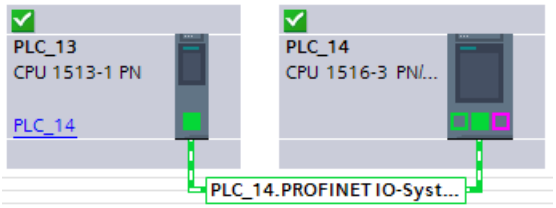
In the "PLC_14" CPU: The instruction is called in a cyclic OB. Interconnect the parameters of the instruction as follows.



In the "PLC_13" CPU: Set up this CPU "PLC_13" as an IO device using the CPU properties.



An IO system is displayed in the Network view.



In the "PLC_14" CPU: A module is located in slot 1 of the IO device.



In the "PLC_14" CPU: The HW identifier of the IO device is made known to the "ModuleStates" instruction through parameter LADDR ("myLADDR"). According to the value "4" of parameter MODE ("myMODE"), the IO device is searched for modules.

At parameter STATE ("mySTATE"), the presence is output for the modules (based on the value of parameter MODE) . Bit 0 serves as a group value and indicates that modules are present. Bit 2 indicates that a module is present in slot 1.

The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.

My_gDB_ModuleStates

		Name	Data type	Start value	Monitor value
1		Static			
2		myLADDR	HW_DEVICE	260	16#0104
3		myMODE	UInt	4	4
4		returnValue	Int	0	0
5		mySTATE	Array[0..127] of Bool		
6		mySTATE[0]	Bool	false	TRUE
7		mySTATE[1]	Bool	false	FALSE
8		mySTATE[2]	Bool	false	TRUE
9		mySTATE[3]	Bool	false	FALSE
10		mySTATE[4]	Bool	false	FALSE
11		mySTATE[5]	Bool	false	FALSE
12		mySTATE[6]	Bool	false	FALSE

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Example

An IO device contains 4 modules in slots 1 to 4. The module in slot 2 is faulty.

The instruction "ModuleStates" is executed for the IO device with MODE = 2 (faulty/not faulty).

The following bits are set in the STATE parameter:

- Bit 0 = 1: A fault exists for at least one module.
- Bit 1 = 0: Slot number 0 (used by IO device)
- Bit 2 = 0: Module in slot number 1 is not faulty.
- Bit 3 = 1: Module in slot number 2 is faulty.
- Bit 4 = 0: Module in slot number 3 is not faulty.
- Bit 5 = 0: Module in slot number 4 is not faulty.
- Bit 6 = 0: Irrelevant
- Bit 7 = 0: Irrelevant

GEN_DIAG: Generate diagnostics information (S7-1500)

Description

The "GEN_DIAG" instruction generates diagnostics information for hardware components from other manufacturers for use in TIA Portal diagnostics. The GSD(GSDL/GSDML) file supplied by the manufacturer must first be installed before the instruction can be used.

The instruction generates all diagnostic events (including those for maintenance).

- Use the LADDR parameter to select the hardware component for which you want to generate a diagnostic event.
- Use the MODE parameter to specify whether the event is to be outgoing or incoming.
- Use the DiagEvent parameter to define the diagnostic event in the DiagnosticDetail structure. When you define a tag at the DiagEvent parameter, the structure is created automatically in the local interface of the block.

The diagnostics information is provided synchronously. Diagnostics information transfer and alarm output are asynchronous.

NOTICE
Failsafe-specific errors messages are not valid If you define failsafe-specific diagnostics information at the DiagEvent parameter, the instruction will check this information and output the error code 80A1.

Parameters

The following table shows the parameters of the "GEN_DIAG" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_ANY	I, Q, M, D, L or constant	Identification number of the hardware component
MODE	Input	UINT	I, Q, M, D, L or constant	Selection of incoming/outgoing information: • 1: The diagnostic event specified is an incoming event • 2: The diagnostic event specified is an outgoing event • 3: All diagnostic events are outgoing. There are therefore no hardware component faults (green diagnostics symbol). The DiagEvent parameter is not evaluated when MODE = 3.
DiagEvent	InOut	DiagnosticDetail	L, D	Specifies the diagnostic event (see "DiagEvent parameter").
RET_VAL	Return	INT	I, Q, M, D, L	Status of instruction / error message (see "RET_VAL parameter")

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter DiagEvent

The structure DiagnosticDetail is a system data type for specifying the diagnostic event. Its structure is as follows:

Parameter		Data type	Description
DiagnosticDetail		Struct	
	ChannelInfo	WORD	Channel properties (0...7)
	ALID	UINT	Local ID of the alarm. The ID uniquely identifies the alarm.
	TextID	UINT	ID of an alarm text in a text list.
	ChannelNumber	UINT	Manufacturer-specific channel number (0x0000 — 0x7FFF)
	Addval_0	DWORD	Placeholder for additional information The value / list of values depends on the connection error.
	TextID2	UINT	Texts for CPU response (mode, OB calls, etc.).
	LADDR	HW_ANY	Identical to the LADDR parameter.
	TextListId	UINT	• 0: No text list • ≠0: ID of the text list
	ChannelDirection	UINT	• 0000: Irrelevant • FFF1: Input • FFF2: Output • FFF3: Input/Output
	Addval_1	DWORD	Placeholder for additional information on the channel error (depends on the GSD file). For channel error types, see also: IEC 61158 (PROFINET IO Type 10 and PROFIBUS DP Type 3).

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0	No error
8080	Value in the MODE parameter is not supported.
8090	Identification for hardware component not available at the LADDR parameter.
8091	Diagnostics information cannot be generated for the hardware component addressed at the LADDR parameter.
80A1	• Content of the DiagnosticsDetail structure at the DiagEvent parameter is invalid or inconsistent. • Failsafe-specific diagnostics information defined at the DiagEvent parameter (not valid).
80A4	Hardware component addressed cannot be accessed.
80C1	Insufficient resources for parallel execution.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

GET_DIAG: Read diagnostic information (S7-1200, S7-1500)

Description

The "GET_DIAG" instruction is used to read the diagnostic information of a hardware component. The hardware component is selected with the parameter LADDR. With the MODE parameter, you select which diagnostic information is read.

Parameters

The following table shows the parameters of the "GET_DIAG" instruction:

Parameter	Declaration	Data type	Memory area	Description
MODE	Input	UINT	I, Q, M, D, L or constant	Use the MODE parameter to select which diagnostics data is output.
LADDR	Input	HW_ANY (WORD)	I, Q, M, L or constant	Hardware ID of the hardware component
RET_VAL	Return	INT	I, Q, M, D, L	Status of the instruction
CNT_DIAG	Output	UINT	I, Q, M, D, L	Reserved (always "0").
DIAG	InOut	VARIANT	I, Q, M, D, L	Diagnostic information corresponds to the selected mode, see table below
DETAIL	InOut	VARIANT	I, Q, M, D, L	Parameter is hidden. Do not use this parameter!

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter MODE

Depending on the value at the MODE parameter, different diagnostic data is output at the DIAG, CNT_DIAG and DETAIL output parameters.

MODE	Description	DIAG	CNT_DIAG
0	Output of all supported diagnostic information for a hardware component as DWORD, where Bit X=1 indicates that mode X is supported.	Bits of the DWORD data type: • Bit 0 = 1: MODE 0 is supported • Bit 1 = 1: MODE 1 is supported • Bit 2 = 1: MODE 2 is supported • Bits 3 to 31: Not relevant	0
1	Output of the diagnostic status of the addressed hardware component.	Structure DIS (see below for description): • MaintenanceState • ComponentStateDetail • OwnState • IOState • OperatingState	0
2	Output of the status of all subordinate components of the addressed hardware component.	Structure DNN (see below for description): • SubordinateState • SubordinateIOState • DNNmode	0

DIS structure

With parameter MODE = 1, the diagnostic information is output in accordance with the DIS structure. In this case, enter the system data type "DIS" as data type for the tag declaration.

Note

Determining the OperatingState

When determining the OperatingState, you need to select LADDR = 52 (hardware identifier of CPUExecUnit).

The following table shows the meaning of the individual parameter values.

Parameter	Data type	Value	Description
MaintenanceState	DWORD	Enum	
		0	Good
		1	The hardware component is disabled.
		2	-
		3	-
		4	-
		5	Maintenance required
		6	Maintenance demanded
		7	Error
		8	Status unknown / error in higher-level hardware component
		9	-
		10	Inputs/outputs are not available.

Parameter	Data type	Value	Description
ComponentState-Detail	DWORD	Bit array	Status: • Bit 0 to 15: Status of the hardware component • Bit 16 to 31: Status of the CPU
		0 to 2 (Enum)	Additional information: • 0: No additional information • 1: Transfer not permitted
		3	Bit 3 = 1 if at least one qualified diagnostics is available
		4	Bit 4 = 1: Maintenance required for at least one channel or the component
		5	Bit 5 = 1: Maintenance demanded for at least one channel or the component
		6	Bit 6 = 1: Error in at least one channel or the component
		7 to 10 (Enum)	• 0: Own - Hardware component is available • 1: Application Ready Pending - No I/O access • 2: Locked by Superordinated - No access to hardware component • 3: Locked by IOController - No access to hardware component • 4: Locked by IOSupervisor - No access to hardware component
		11 to 14 (Enum)	• 0 (bit 11 = 0, bit 12 = 0, bit 13 = 0, bit 14 = 0): OK - The correct element exists. • 1 (bit 11 = 1, bit 12 = 0, bit 13 = 0, bit 14 = 0): Substitute - The element was replaced by a compatible element. • 2 (bit 11 = 0, bit 12 = 1, bit 13 = 0, bit 14 = 0): Wrong - The element was replaced by an incompatible element (The element is reachable, but not ready for operation.). • 3 (bit 11 = 1, bit 12 = 1, bit 13 = 0, bit 14 = 0): No Submodule - The element does not exist.
		15	Reserved (always "1").
		16 to 31	Status information for modules generated by the CPU: • Bit 16 = 1: Submodule or module disabled • Bit 17 = 1: CiR operation active • Bit 18 = 1: Input not available • Bit 19 = 1: Output not available • Bit 20 = 1: Overflow diagnostics buffer • Bit 21 = 1: Diagnostics not available • Bit 22 = 1: Partial device failure (with PNIO for Shared iDevices) • Bit 23 - 31: Reserved (always 0)

Parameter	Data type	Value	Description
OwnState	UINT	Enum	Maintenance status
		0	Good
		1	Disabled
		2	Maintenance required
		3	Maintenance demanded
		4	Error
		5	Not accessible
		6	Diagnostic status unknown
		7	IO not available - Inputs/outputs are not available.
IOState	WORD	Bit array	I/O status of the hardware component
		Bit 0 = 1	Good
		Bit 1 = 1	Disabled
		Bit 2 = 1	Maintenance required
		Bit 3 = 1	Maintenance demanded
		Bit 4 = 1	Error
		Bit 5 = 1	Hardware component not reachable - Hardware component cannot be accessed.
		Bit 6 = 1:	Qualified: Bit 6 = 1 if at least one qualified diagnostics is available
		Bit 7 = 1:	I/O data not available - I/O data not available
		8 to 14	Reserved (always = 0)
		Bit 15	Network/hardware fault • S7-1200: Reserved (always = 0) • S7-1500: If bit 4 = 1 or bit 5 = 1: – Bit 15 = 0: Network error – Bit 15 = 1: Hardware error

Parameter	Data type	Value	Description
OperatingState	UINT	Enum	
		0	Not Supported - for I/O modules OperatingState always has the value 0.
		1	STOP / firmware update
		2	STOP / Memory reset
		3	STOP / self start
		4	STOP
		5	Memory reset
		6	Startup
		7	-
		8	RUN
		9	-
		10	HOLD
		11	-
		12	-
		13	DEFECTIVE
		14	-
		15	De-energized
		16	CiR
		17	STOP without ODIS
		18	RUN ODIS
		19	Resetting to factory settings
		20	STOP / no memory card

DNN structure

With parameter MODE = 2, the diagnostic information details are output in accordance with the DNN structure. In this case, enter the system data type "DNN" as data type for the tag declaration.

The following table shows the meaning of the individual parameter values.

Parameter	Data type	Value	Description
SubordinateState	UINT	Enum	Status of the subordinate component (see parameter OwnState of the DIS structure)
SubordinateIOState	WORD	Bitarray	Status of the inputs and outputs of the subordinate component (see parameter IO State of the DIS structure)
DNNmode	WORD	Bitarray	• Bit 0 = 0: Diagnostics enabled • Bit 0 = 1: Diagnostics disabled • Bit 1 to 15: Reserved

Parameter RET_VAL

Error code* (W#16#...)	Description
0	No error
n	The data area at the DETAIL parameter is too small. Not all details of the diagnostics data are output.
8080	Value in the MODE parameter is not supported.
8081	Type at the DIAG parameter is not supported with the selected mode (parameter MODE).
8082	Type at the DETAIL parameter is not supported with the selected mode (parameter MODE).
8090	LADDR does not exist
80C1	Insufficient resources for parallel execution.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

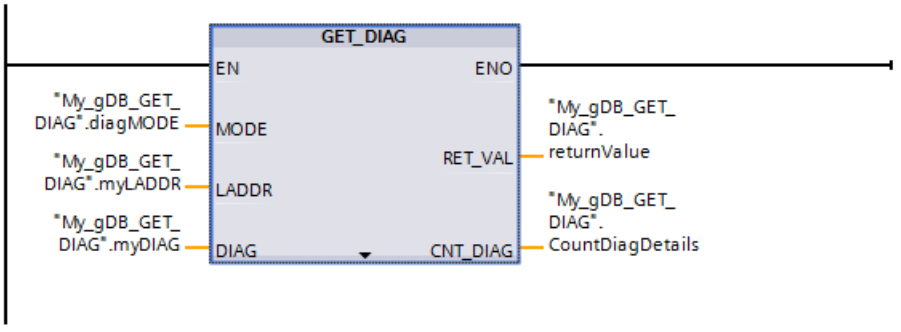
In the following example, you read out the diagnostic information of a CPU.

Create four tags and a "myDIAG" structure (with the DIS data type) in a global data block for storing the data.

My_gDB_GET_DIAG

	Name	Data type	Start value
1	Static		
2	diagMODE	UInt	1
3	myLADDR	HW_ANY	50
4	returnValue	Int	0
5	CountDiagDetails	UInt	0
6	myDIAG	DIS	
7	MaintainanceState	DWord	16#0
8	ComponentStateDetail	DWord	16#0
9	OwnState	UInt	0
10	IOState	Word	16#0
11	OperatingState	UInt	0

Interconnect the parameters of the instruction as follows.



The HW identifier of the CPU is made known to the "GET_DIAG" instruction through parameter LADDR ("myLADDR"). According to the value "1" of parameter MODE ("diagMODE"), the following applies:

- The instruction reads the status of the addressed hardware object (of the CPU).
- At parameter DIAG ("myDIAG"), the diagnostic information is output in a structure (DIS data type).

To understand the diagnostic information, the hex values must be converted to binary code. At parameter DIAG ("myDIAG"), the following is indicated:

- MaintenanceState: According to the value "0", the CPU requires no maintenance.
- ComponentStateDetail: According to the hex value "0000_8000", bit 15 is active.
- OwnState: According to the value "0", no fault has occurred.
- IOState: According to the hex value "0001", there is no maintenance required.
- OperatingState: Outputs "0".

The output parameter RET_VAL ("returnValue") indicates that processing took place without errors. At output parameter CNT_DIAG ("CountDiagDetails"), it is indicated that "0" diagnostic details of parameter DETAIL have been output.

My_gDB_GET_DIAG

	Name	Data type	Start value	Monitor value
1	Static			
2	diagMODE	UInt	1	1
3	myLADDR	HW_ANY	50	16#0032
4	returnValue	Int	0	0
5	CountDiagDetails	UInt	0	0
6	myDIAG	DIS		
7	MaintainanceState	DWord	16#0	16#0000_0000
8	ComponentStateDetail	DWord	16#0	16#0000_8000
9	OwnState	UInt	0	0
10	IOState	Word	16#0	16#0001
11	OperatingState	UInt	0	0

Note: For example, you could individually read out bit 3 (channel diagnostics yes/no) from the ComponentStateDetail tag.

- Address the bit as follows: ComponentStateDetail.%X3

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Example - Diagnostics of an IO device

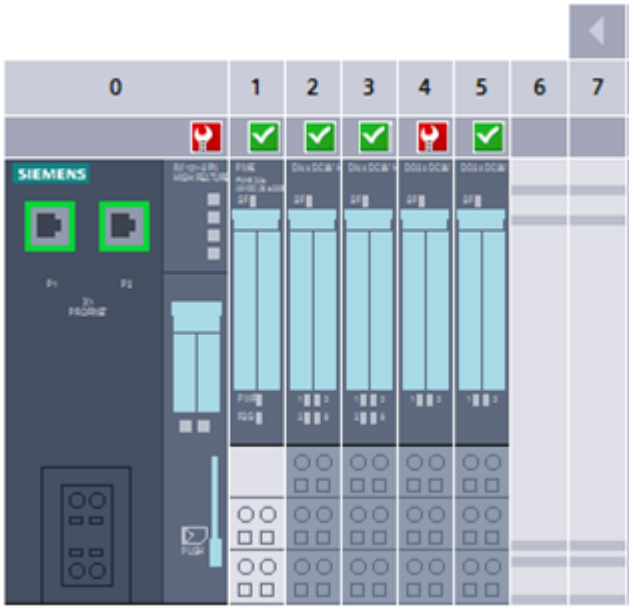
In the following example, you read out the diagnostic information of an IO device. For diagnostics, you use the device view of the IO device and the instruction GET_DIAG.

Diagnostics in the device view

One of the IO device modules signals an error. The device status changes and is displayed as follows:



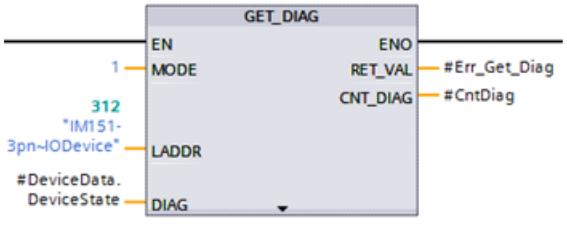
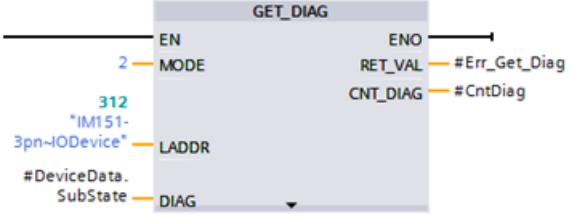
The module and the head module of the IO device show a group status in the device view. An error is therefore displayed for both.



Diagnostics with the GET_DIAG instruction

To read out the diagnostic information of the IO device, address the head module of the IO device (via LADDR).

- In mode 1 (MODE has the value 1), you read out the status of the IO device.
- In mode 2 (MODE has the value 2), you read out the status of the subordinate modules.

Call MODE 1: Self-diagnostics	Call MODE 2: Summary of the subordinate diagnostics
	

LADDR (HW ID) = IM151-3pn~IODevice
DIAG (DIS structure) = #DeviceData.DeviceState
DIS.OwnState = 0 (no error)
DIS.IOState = bit sequence 0000 0000 0000 0001
	LADDR (HW ID) = IM151-3pn~IODevice
	DIAG (DNN structure) = #DeviceData.SubState
	DNN.SubordinateState = 4 (error)
	DNN.SubordianteIOState = bit sequence 1000 0000 0001 0001

Result

Unlike the device view, the GET_DIAG instruction does not return the group status of the IO device. The error in the subordinate module is not read until mode 2.

11.4.4.10 Pulse (S7-1200)

CTRL_PWM: Pulse-width modulation (S7-1200)

Description

The "CTRL_PWM" instruction is used to enable and disable a pulse output supported by the CPU using the software.

- Enter the hardware ID of the pulse generator that you want to control with the instruction at the input PWM.
- When the bit in the ENABLE input of the instruction is set, the pulse output is enabled.
  - If ENABLE has a value of TRUE the pulse generator will generate a pulse with the properties defined in the device configuration.
  - When the bit in the ENABLE input is reset or the CPU changes to STOP, the pulse output is disabled and no more pulses are generated.

BUSY always has the value of FALSE for S7-1200 since S7-1200 activates the pulse generator if the instruction "CTRL_PWM" is executed.

When the EN enable input has signal state "1" and no errors have occurred during execution of the instruction, the enable output ENO is set.

Note

Use of the force table for PWM and PTO

Digital inputs and outputs that are used for PWM and PTO cannot be forced. Digital inputs and outputs that were assigned via device configuration are not controlled by either the force table or the watch table.

Note

Process image of PTO/PWM outputs

Do not use PTO/PWM outputs in the process image (for example, for access in the user program, for online functions or in the HMI). The update rate of the process image is much slower than the rate of the signal changes. The display in the process image therefore does not reflect the signal flow.

Requirement

A prerequisite for the correct execution of the instruction is that the specified pulse generator is activated in the hardware configuration.

Open the properties of the module in the device view. Go to "Pulse generators (PTO/PWM)", open the desired PTO/PWM and activate the function the "Activate this pulse generator" under "General".

Go to "Assigning parameters" and set the Pulse options.

Note

Pulse output parameters are assigned exclusively in the device configuration and not using the "CTRL_PWM" instruction. Any change of parameters that is intended to have an effect on the CPU must therefore be made while the CPU is in STOP mode. Changing the duration of the pulse is an exception.

Changing the pulse duration from the user program

The pulse duration setting made in the dialog "Pulse options" can be changed from the user program.

The value set for "Initial pulse duration" is written in the output bytes of the pulse generator. The start address and the end address are displayed in the properties of the pulse generator under "I/O-addresses".

To change the pulse duration, write the desired values in the output word address specified in the device configuration.

Example:

- A value of 500 (decimal) is used for the "Initial pulse duration". The start address of the PTO/PWM is "1000" and the end address is "1001".
- The binary value of "0000000111110100" (=500 decimal) is written in both of the output bytes.
  - Start address (AB1000): 0000_0001 (BIN)
  - End address (AB1001): 1111_0100 (BIN)

Please note that the pulse duration always depends on the parameters set for the Pulse duration format (hundredths, thousandths, ...).

Change the cycle time from the user program.

Select the check box "Permit change to cycle time in runtime" in the "Pulse options" of the corresponding pulse generator. When this check box is selected, the first two output bytes contain the pulse duration and the third to sixth output byte contains the cycle time.

During the run time of the pulse generator, you can change the value of the double word at the end of allocated memory of the outputs. This changes the cycle time of the PWM signal.

Example: Select the check box so that the CPU assigns PWM1 six output bytes, and select AB1008 to AB1013 for this. After loading the program into the CPU and starting the pulse generator, you can change the pulse duration by writing AW1008 and the cycle time by writing AD1010.

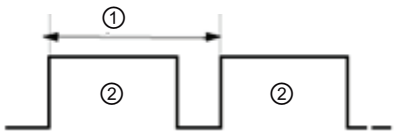


Figure 11-1 PWM_Example

① Cycle time
② Pulse duration

Parameters

The following table shows the parameters of the "CTRL_PWM" instruction:

Parameter	Declaration	Data type	Memory area	Description
PWM	Input	HW_PWM	I, Q, M, D, L or constant	Hardware ID of the pulse generator The hardware ID can be found in the properties of the pulse generator in the Device view. The hardware IDs of the pulse generators are also listed in the system constants.
ENABLE	Input	BOOL	I, Q, M, D, L or constant	The pulse output is enabled when ENABLE = TRUE and disabled when ENABLE = FALSE.
BUSY	Output	BOOL	I, Q, M, D, L	Processing status
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction (see below).

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

Error code* (W#16#...)	Description
0	No errors
80A1	Hardware ID of the pulse generator is invalid.
80D0	Pulse generator with the specified hardware ID is not activated. Activate the pulse generator in the CPU properties under "Pulse generators (PTO/PWM)".
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

CTRL_PTO: Output a pulse sequence with a preset frequency (S7-1200)

Description

The "CTRL_PTO" instruction makes a pulse sequence available to you with a preset frequency. For this you do not require an axis DB of a technological object.

Requirement

The use of the "CTRL_PTO" instruction does not require an activated pulse generator. You activate this in the hardware configuration as well as selecting the signal type. You need to specify the pulse generator at the "PTO" parameter and set the "REQ" parameter to TRUE.

Note

Using a PLC pulse generator

At any one time a pulse generator can only be used by one instruction.

Note

Use of the force table with PWM and PTO

Digital inputs and outputs that are used for PWM and PTO cannot be forced. Digital inputs and outputs that were assigned via the device configuration are not controlled by either the force table or the watch table.

Note

Process image of PTO/PWM outputs

Do not use PTO/PWM outputs in the process image (for example, for access in the user program, for online functions or in the HMI). The update rate of the process image is much slower than the rate of the signal changes. The display in the process image therefore does not reflect the signal flow.

Parameter

The following table shows the parameters of the "CTRL_PTO" instruction:

Parameter	Declaration	Data type	Description
REQ	Input	BOOL	• REQ=1: Sets the frequency of the pulse generator to the value of FREQUENCY REQ=1 and FREQUENCY=0: Deactivates the pulse generator • REQ=0: No change at pulse generator
PTO	Input	HW_PTO	Hardware identifier of the pulse generator This can be taken from the properties of the pulse generator in the device view or the list of system constants.
FREQUEN-CY	Input	UDINT	Frequency of the pulse sequence to be output in Hz
DONE	Output	BOOL	Status parameter with the following values: • 0: Job not yet started or still in progress • 1: Job completed without error
BUSY	Output	BOOL	Processing status Since the S7-1200 enables the pulse generator when the "CTRL_PTO" instruction is executed, BUSY always has the value FALSE with the S7-1200.

Parameter	Declaration	Data type	Description
ERROR	Output	BOOL	Status parameter • 0: No error • 1: An error occurred during execution of the instruction.
STATUS	Output	WORD	Status of the instruction (see below)

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter Status

Error code* (W#16#...)	Explanation
0	No error
8090	The pulse generator with the specified hardware ID is already being used.
8091	Range exceeded for the "FREQUENCY" parameter; for more detailed information, refer to the manual.
80A1	The PTO parameter does not address a hardware ID of a pulse generator.
80D0	The pulse generator with the specified hardware ID is not activated. Activate the pulse generator in the CPU properties under "Pulse generators (PTO/PWM)".
80D1	The pulse generator with the specified hardware ID does not have the "PTO" property. Select "PTO" in the hardware configuration.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

11.4.4.11 Recipes and data logging (S7-1200, S7-1500)

Recipe functions (S7-1200, S7-1500)

Recipe functions - overview (S7-1200, S7-1500)

Description

With the instructions "RecipeExport" and "RecipeImport" you transfer recipe data

- from a recipe data block to a CSV file on the memory card of the CPU ("RecipeExport")
- from a CSV file on the memory card of the CPU to a recipe data block ("RecipeImport")

Note

Service life of memory cards

The service life of memory cards is limited. Additional information is available here: AUTOHOTSPOT

Protection of recipe data

The instructions "RecipeExport" and RecipeImport" are not suitable for applications in which special protection is necessary for the recipe data. For higher protection of the data use the recipe functions of HMI.

Position of the recipe data block with "RecipeExport"

The data block with the recipe data exported with "RecipeExport" can be located in work or load memory. If the data block is both in the work memory and in the load memory, the values from the work memory are used.

Delimiters

You can use different delimiters to separate the row values in the recipe. Note that if you open the CSV file with the exported recipe data, for example in Excel, commas can be replaced by periods and vice versa. The reason for this is that different languages use different decimal separators. This can have an effect when importing CSV files. The following convention applies for "RecipeImport"; the first detected possible delimiter is assumed to be the delimiter for all values of a table row.

- Example 1: If a semicolon is detected as the first delimiter, during import it is assumed that the semicolon is used as the delimiter for all further values. As the decimal delimiter with a value of the type REAL, a comma is then used.
- Example 2: If a comma is detected as the first delimiter, during import it is assumed that the comma is used as the delimiter for all further values. As the decimal delimiter with a value of the type REAL, a period is then used.

If the recipe data record contains a character string, make sure that the character string does not contain the delimiter you are using. With character strings, also make sure that they do not contain any control characters that specify a line break (with ASCII: LF, CR LF, CR).

Note: Spaces and tabs are not used as delimiters. These lead to error code 80B2 during import.

Note

Asynchronous processing of the recipe functions

Because the recipe functions are executed asynchronously, you must keep the data in the source area consistent until the DONE parameter or the ERROR parameter changes to the value "1".

RecipeExport: Exporting recipes (S7-1200, S7-1500)

Description

The "RecipeExport" instruction exports the recipe data from a data block to a CSV file in the load memory.

With S7-1500 CPUs, the DB that is to be exported can be located either in the work memory or in the load memory. If the data block is in both the work memory and the load memory, the

values from the work memory are exported. For S7-1200 CPUs, the DB to be exported must be present in the load memory.

The CSV file is created in the load memory. For S7-1500 CPUs, this is the memory card; for S7-1200 CPUs the internal or external load memory (if a memory card is present).

During the export, the CSV file is created in the "Recipes" folder in the main directory of the load memory. The name of the data block is used as file name of the created CSV file. If a CSV file with an identical name already exists, it is overwritten during the export.

Each recipe data record generates a line in the CSV file. These lines have the following structure:

- The first line contains the "Index" entry and the name of the recipe components.
- Each additional line contains the number of the recipe data record (index) and the associated components.

When you take the data types of the recipe DB, the maximum size of a data record is 5000 bytes. If this is exceeded, the error code W#16#8092 is returned. This restriction to 5000 bytes also applies to the size of the header; any characters in the DB exceeding this limit are truncated, and the process continues with the first recipe data record.

The S7-1200 CPUs also monitor the number of columns. These must not exceed 255 (whereby the index column is not counted).

Functional description

The export is triggered by the "REQ" parameter. The BUSY parameter is set to "1" during the export. After the execution of the instruction, BUSY is reset to "0" and the completion of the operation is indicated with "1" at the DONE parameter.

If an error occurs during execution, this is signaled by the parameters ERROR and STATUS.

Parameters

The following table shows the parameters of the "RecipeExport" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Control parameter REQUEST: Activates the export on a rising edge.
RECIPE_DB	InOut	VARIANT	D	Pointer to the recipe data block. For information on the structure of the data block, refer to: Structure of a recipe DB (Page 4001)
DONE	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Job not yet started or still executing. • 1: Job executed without errors.
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter • 0: The instruction is not executed. • 1: The instruction is executed.

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Neither warning nor error. • 1: An error has occurred, STATUS supplies detailed information on the type of error.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter See the "STATUS" parameter table.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

Error code* (W#16#...)	Explanation
0	No error occurred
7000	No job processing active
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8090	File name of the CSV file contains invalid characters. The file name of the CSV file is identical to the name of the data block.
8091	The data structure referenced by RECIPE_DB cannot be processed or the number of data columns in a S7-1200 CPU is greater than 255.
8092	Data structure at RECIPE_DB parameter exceeds 5000 bytes.
80B3	Insufficient memory space on the memory card or the internal load memory.
80B4	The memory card is write-protected.
80B6	With (S7-1200 CPUs only) attribute "Save only in load memory", the recipe data block is not activated.
80C0	CSV files is temporarily locked.
80C1	Recipe data block temporarily disabled.
80C3	Too few resources
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Example program for recipe functions (Page 4004).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

RecipeImport: Importing recipes (S7-1200, S7-1500)

Description

The "RecipeImport" instruction imports the recipe data from a CSV file of the load memory into the data block at the RECIPE_DB parameter.

For S7-1500 CPUs, the CSV file is on the memory card; for S7-1200 CPUs, it is in the internal or external load memory (if a memory card is present).

The values in the data block are overwritten in the process. With S7-1500 CPUs, the DB to which the recipe data is to be imported can be located either in the work memory or in the load memory. If it is in both the work memory and the load memory, the values are imported to the data block located in the work memory. For S7-1200 CPUs, the DB into which the recipe data is to be imported must be located in the load memory.

Note the following when importing the CSV file:

- The CSV file must be located in the "Recipes" directory of the load memory.
- The name of the CSV file must match the name of the data block at the RECIPE_DB parameter.
- Each line of the CSV file corresponds to one recipe data record. The first line (header) of the CSV file contains the name of the recipe components (see also: Structure of a recipe DB (Page 4001)). The first line is ignored completely during the import. The names of the recipe components in the CSV file and the data block are not synchronized during the import, and the number of entries is also irrelevant.
- In each case the first value in each line of the CSV file is the index number of the recipe. The individual recipes are imported in the order of the index. For this, the index in the CSV file has to be in ascending order and may contain no gaps (if this is not the case, the error message 80B0 is output at the STATUS parameter.
- The CSV file may not contain more recipe data records than provided for in the data block. The maximum number of data records is indicated by the array limits in the data block.
- If a value of the CSV file does not fit in the associated data type of the DB, the import is rejected. If necessary, character strings are truncated after their maximum permissible length.
  If the data type in the DB is BOOL type, an S7-1500 CPU accepts all integer values. The value 1 is then set to TRUE and all other values are set to FALSE. An S7-1200 CPU accepts only the values 0 and 1, all other values are rejected.

Functional description

The import is triggered by the "REQ" parameter. The BUSY parameter is set to "1" during the import. After the execution of the instruction, BUSY is reset to "0" and the completion of the operation is indicated with "1" at the DONE parameter.

If an error occurs during execution, this is signaled by the parameters ERROR and STATUS.

Parameters

The following table shows the parameters of the "RecipeImport" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Control parameter REQUEST: Activates the import on a positive edge.
RECIPE_DB	InOut	VARIANT	D	Pointer to the recipe data block. For information on the structure of the data block, refer to: Structure of a recipe DB (Page 4001)
DONE	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Job not yet started or still executing. • 1: Job executed without errors.
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter • 0: The instruction is not executed. • 1: The instruction is executed.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Neither warning nor error. • 1: An error has occurred, STATUS supplies detailed information on the type of error.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter See the "STATUS" parameter table.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

Error code* (W#16#...)	Explanation
0	No error occurred
7000	No job processing active
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8090	The file name contains invalid characters.
8092	No matching CSV file found for the import. Possible cause: The name of the CSV file does not match the name of the recipe DB.
80C0	CSV file is temporarily locked.
80C1	Recipe data block is temporarily locked.
80B0	Numbering in the index of the CSV file is not continuous, not ascending or exceeds the maximum number (array limit) in the data block.
80B1	Structure of the recipe data block and the CSV file do not match: The CSV file contains too many fields.
80B2	Structure of the recipe data block and the CSV file do not match: The CSV file contains too few fields.

Error code* (W#16#...)	Explanation
80B6	With (S7-1200 CPUs only) attribute "Save only in load memory", the recipe data block is not activated.
80C3	Too few resources
80D0 +n	Structure of the recipe data block and the CSV file do not match: Data type in field n does not match (n<=46).
80FF	Structure of the recipe data block and the CSV file do not match: Data type in field n does not match (n>46).
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Example program for recipe functions (Page 4004).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

Structure of a recipe DB (S7-1200, S7-1500)

Introduction

The following sections describes the structure of a recipe DB based on a simple example. The recipe DB consists of five data records, three of which are used. The fourth and fifth data records are left free for later expansions. Each data record contains one recipe, which is made up of the recipe name and eight ingredients.

product-name	water	barley	wheat	hops	yeast	wa-terTmp	mashTmp	mash-Time	QTest
Pils	10	9	3	280	39	40	30	100	0
Lager	10	9	3	150	33	50	30	120	0
BlackB-eer	10	9	3	410	47	60	30	90	1
Not_used	0	0	0	0	0	0	0	0	0
Not_used	0	0	0	0	0	0	0	0	0

Structure of the recipe data block

The recipe data are implemented as follows in a global data block:

- The template for all recipes is the PLC data type "Beer_Recipe" with the recipe components "procutname", "water", etc. with the corresponding data types.
- In a global data block, the PLC data type is used as Array [1.. 5] of "Beer_Recipe". The array limits (1 to 5 in this case) indicate the maximum number of recipes that the DB may contain.
- The values for the recipe components are added as start value in the data block.
- The global DB is interconnected with the instruction via the InOut parameter RECIPE_DB.

Recipe_DB

	Name	Data type	Offset	Start value
1	Static			
2	Products	Array [1 .. 5] of "Beer_Recipe"	...	
3	Products[1]	"Beer_Recipe"	...	
4	Products[2]	"Beer_Recipe"	...	
5	Products[3]	"Beer_Recipe"	...	
6	productname	String[20]	...	'BlackBeer'
7	water	UInt	...	10
8	barley	UInt	...	9
9	wheat	UInt	...	3
10	hops	UInt	...	410
11	yeast	UInt	...	47
12	waterTmp	UInt	...	60
13	mashTmp	UInt	...	30
14	mashTime	UInt	...	90
15	QTest	UInt	...	1
16	Products[4]	"Beer_Recipe"	...	
17	Products[5]	"Beer_Recipe"	...	

Exporting to CSV file

After the execution of the "RecipeExport (Page 3996)" instruction the data of the DB are written to a CSV file with the following structure:

Recipe_DB.csv

```
index,productname,water,barley,wheat,hops,yeast,waterTmp,mashTmp,mashTime,QTest
1,”Pils”,10,9,3,280,39,40,30,100,0
2,”Lager”,10,9,3,150,33,50,30,120,0
3,”BlackBeer”,10,9,3,410,47,60,30,90,1
4,”Not_used”,0,0,0,0,0,0,0,0,0
5,”Not_used”,0,0,0,0,0,0,0,0,0
```

Display in Excel

The CSV file can be opened in Excel to make reading and editing easier. If the commas are not recognized as separators when you open the file, use the Excel import function to output the data in structured form:

	A	B	C	D	E	F	G	H	I	J	K
1	index	product	water	barley	wheat	hops	yeast	waterTmp	mashTmp	mashTime	QTest
2	1	"Pils"	10	9	3	280	39	40	30	100	0
3	2	"Lager"	10	9	3	150	33	50	30	120	0
4	3	"BlackBeer"	10	9	3	410	47	60	30	90	1
5	4	"Not_used"	0	0	0	0	0	0	0	0	0
6	5	"Not_used"	0	0	0	0	0	0	0	0	0

Editing the CSV file

The CSV file is uploaded to the PC/programming device by means of the Web server and edited. After editing, the modified file can be reloaded in the CPU. You must delete the existing CSV file for this.

You can use the "RecipeImport (Page 3999)" instruction to re-import the modified data from the CSV file into the data block.

Note that the modified data must still be compatible with the data block. In other words:

- No changes may be made to the structure in the table (for example, by adding ingredients in a new column).
- If you add data records to the file, make sure when importing into the data block that the array limits through which you specify the maximum number of data records correspond to at least the number of data records.
- An index is automatically generated during export to the CSV file. If you create additional data records, add consecutive index numbers accordingly.
- The values in the table cells must correspond to the data types used in the data block with respect to format as well as length.
  - Example 1: If INT data type was used in the data block, use only integers in the table.
  - Example 2: If SINT data type was used in the data block, use only integers with the values -128 to +127 in the table.

  Always keep in mind the permitted data types and data areas as listed in the table below when you make changes in the table.

Data type		Format	Note
Floating-point numbers	LReal	+9.9999999999999E+999	Always written exponentially
	Real	+9.9999999E+99	Always written exponentially
Signed integers	LInt	+9999999999999999999	Value range of signed integers: -9223372036854775808 .. +9223372036854775807
	DInt	+9999999999	Value range of integers: -2147483648 to +2147483647
	Int	+99999	Value range of integers: -32768 to +32767
	SInt	+999	Value range of integers: -128 to +127
Unsigned integers	ULInt	+99999999999999999999	Value range of integers: 0 to +18446744073709551615
	UDInt	+9999999999	Value range of integers: 0 to +4294967295
	UInt	+99999	Value range of integers: 0 to +65535
	USInt	+999	Value range of integers: 0 to +255
Binary numbers	LWord	+99999999999999999999	Value range of integers: 0 to +18446744073709551615
	DWord	+9999999999	Value range of integers: 0 to +4294967295
	Word	+99999	Value range of integers: 0 to +65535
	Byte	+999	Value range of integers: 0 to +255
	Bool	9	Value range: 0 or 1

Data type		Format	Note
Date and time	LTIME	dddddd:hh:mm:ss. 999_999_999	ISO format with milliseconds, microseconds and nanoseconds
	TIME	hhh:mm:ss.999	ISO format with milliseconds
	S5TIME	hhh:mm:ss.999	ISO format with milliseconds
	LDT	YYYY-MM-DD hh:mm:ss. 999_999_999	ISO format with milliseconds, microseconds and nanoseconds
	DTL	YYYY-MM-DD hh:mm:ss. 999_999_999	ISO format with milliseconds, microseconds and nanoseconds
	DT	YYYY-MM-DD hh:mm:ss.999	ISO format with milliseconds
	DATE	YYYY-MM-DD	ISO format
	LTime_Of_Day	hh:mm:ss.999_999_999	ISO format
	TOD	hh:mm:ss.999	ISO format
Characters	WString	"abcd"	• Character string inside double quotation marks. When a recipe DB is imported, the double quotation marks may be omitted. • Current length • A character string of data type WString consists of elements of the WChar data type. When a recipe DB is exported, the contents of characters of the WChar data type are limited to 16#FF (example: 16#1255 is changed to 16#55).
	String	"abcd"	• Character string inside double quotation marks. When a recipe DB is imported, the double quotation marks may be omitted. • Current length
	WChar	"a"	Individual character inside double quotation marks. When a recipe DB is imported, the double quotation marks may be omitted. When a recipe DB is exported, the contents of characters of the WChar data type are limited to 16#FF (example: 16#1255 is changed to 16#55).
	Char	"a"	Individual character inside double quotation marks. When a recipe DB is imported, the double quotation marks may be omitted.

Example program for recipe functions (S7-1200, S7-1500)

In the following example, you export the recipe data from a data block (recipe DB) to a CSV file and import new recipe data into the data block.

Requirements

As the template for the recipe data create a PLC data type "Beer_Recipe" an.

Beer_Recipe

	Name	Data type	Default value
1	productname	String[20]	''
2	water	UInt	0
3	barley	UInt	0
4	wheat	UInt	0
5	hops	UInt	0
6	yeast	UInt	0
7	waterTmp	UInt	0
8	mashTmp	UInt	0
9	mashTime	UInt	0
10	QTest	UInt	0

To store all recipe data records, create a recipe DB. You will find further information on creating a recipe DB in "Structure of a recipe DB (Page 4001)".

Recipe_DB

	Name	Data type	Start value
1	Static		
2	Products	Array[1..5] of "Beer_Recipe"	
3	Products[1]	"Beer_Recipe"	
4	Products[2]	"Beer_Recipe"	
5	Products[3]	"Beer_Recipe"	
6	productname	String[20]	'Blackbeer'
7	water	UInt	10
8	barley	UInt	9
9	wheat	UInt	3
10	hops	UInt	410
11	yeast	UInt	47
12	waterTmp	UInt	60
13	mashTmp	UInt	30
14	mashTime	UInt	90
15	QTest	UInt	1
16	Products[4]	"Beer_Recipe"	
17	Products[5]	"Beer_Recipe"	

To store the data for the interconnection, create twelve tags in a global data block.

My_gDB_Recipefunctions

	Name	Data type	Start value
1	Static		
2	expREQ	Bool	false
3	expDONE	Bool	false
4	expBUSY	Bool	false
5	expERROR	Bool	false
6	expSTATUS	Word	16#0
7	impREQ	Bool	false
8	impDONE	Bool	false
9	impBUSY	Bool	false
10	impERROR	Bool	false
11	impSTATUS	Word	16#0
12	outputErr_EXP	Byte	0
13	outputErr_IMP	Byte	0

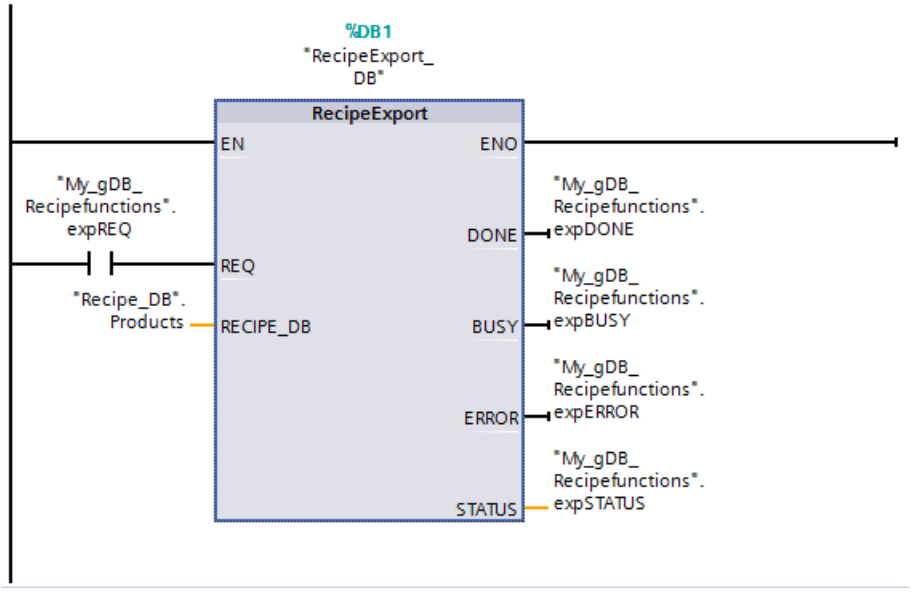
To display and store a CSV file, use the Web server of the CPU. In order to use the web server and theCSV file, the following is necessary:

- The web server must be activated in the CPU settings.
- A user profile with the Web server properties of the CPU must be set up with the following settings:
  - The user must be able to read files.
  - The user must be able to write/delete files.

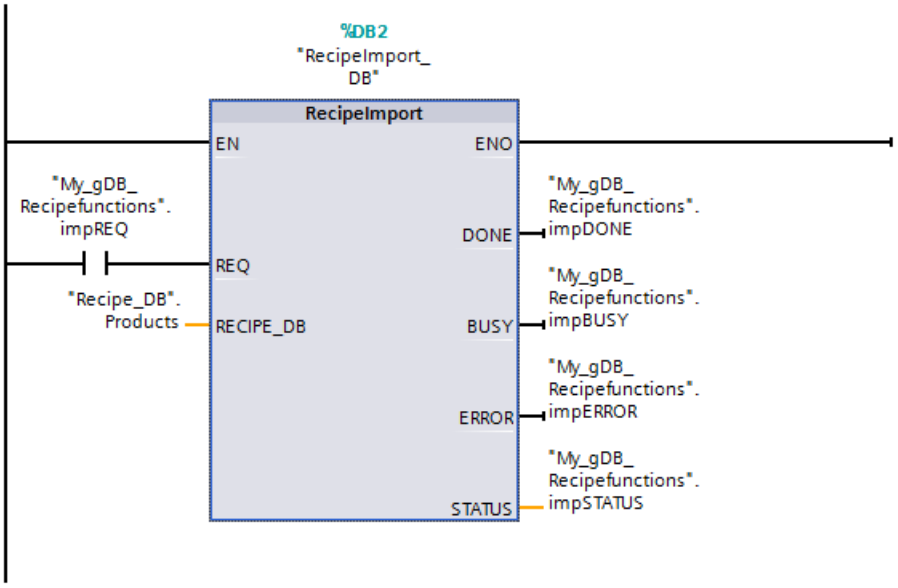
Interconnection of the parameters

The instructions are called in a function block.

Network 1: Interconnect the parameters of the "RecipeExport" instruction as follows.

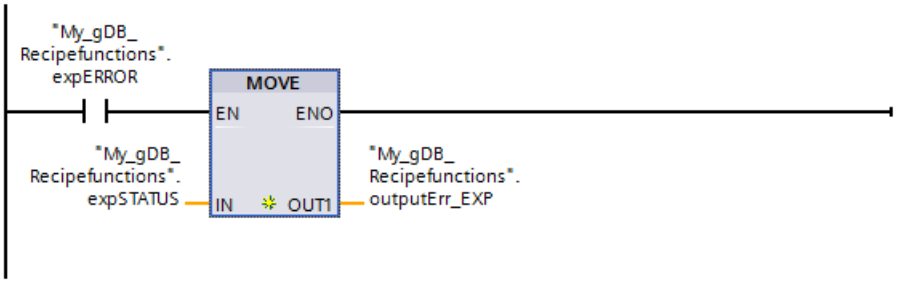


Network 3: Interconnect the parameters of the "RecipeImport" instruction as follows.

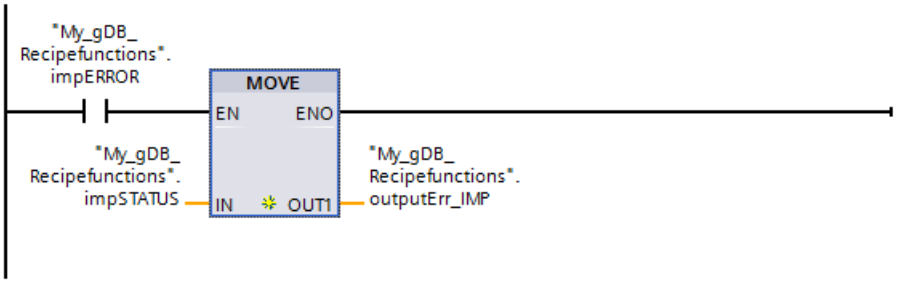


Network 2 and 4: Since the parameters STATUS ("expSTATUS" and "impSTATUS") are only valid for one cycle, record the parameters with a "MOVE" instruction. At the output parameters OUT1 ("outputErr_EXP" or "outputErr_IMP") output the status in case of an error.

Interconnect the parameters in network 2 as follows.



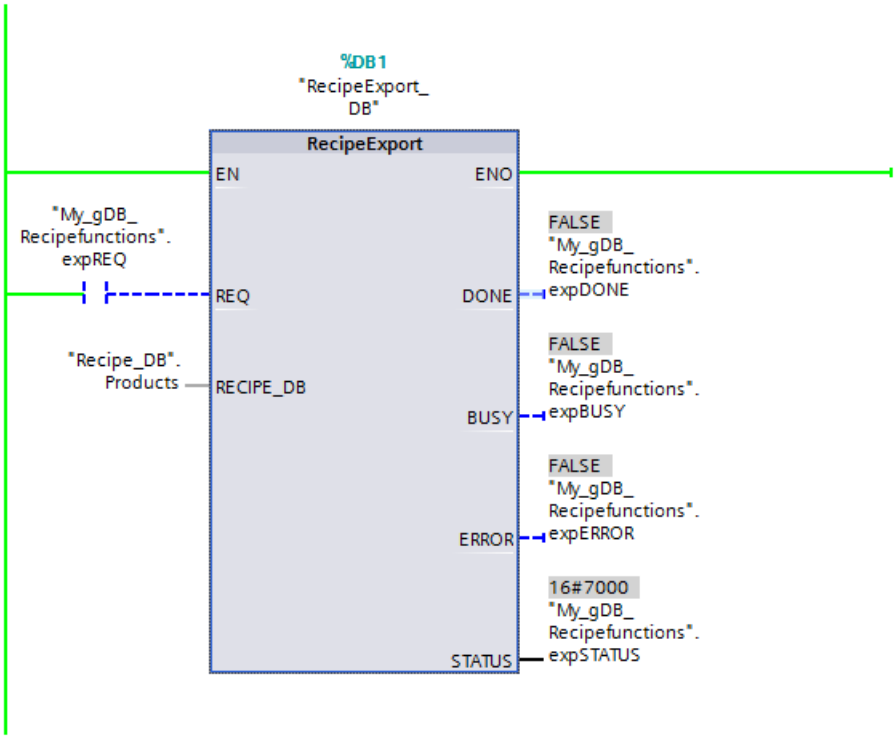
Interconnect the parameters in network 4 as follows.



"RecipeExport" instruction

The recipe DB to be exported needs to be interconnected at the RECIPE_DB parameter. If the REQ ("expREQ") parameter has the signal status "TRUE" export is started. From the data of the recipe DB, a CSV file is generated and stored in the recipe folder in the CPU memory. At output parameter STATUS ("expSTATUS") it is displayed that a signal change has taken place. Then, it is indicated that no further processing is taking place (the value is "7000").

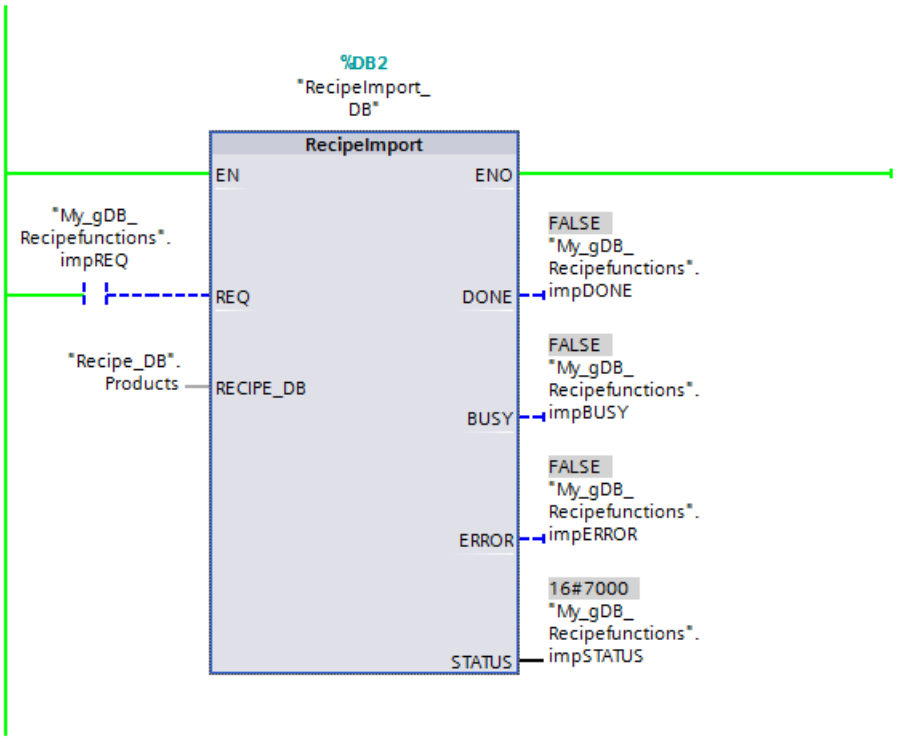
At output parameter ERROR ("expERROR"), it is indicated that the processing of the instruction was error-free.



"RecipeImport" instruction

The recipe DB to be imported needs to be interconnected at the RECIPE_DB parameter. If the REQ ("impREQ") parameter has the signal status "TRUE" export is started. In the existing recipe DB, all the recipe data is loaded from the corresponding CSV file-that is located in the recipe folder of the CPU memory. At output parameter STATUS ("impSTATUS") it is displayed that a signal change has taken place. Then, it is indicated that no further processing is taking place (the value is "7000").

At output parameter ERROR ("impERROR"), it is indicated that the processing of the instruction was error-free.



Displaying and saving the CSV file

Opening the Web server

Use the Internet browser to open the Web server (via the IP address of the CPU) and log on in the Web server menu.

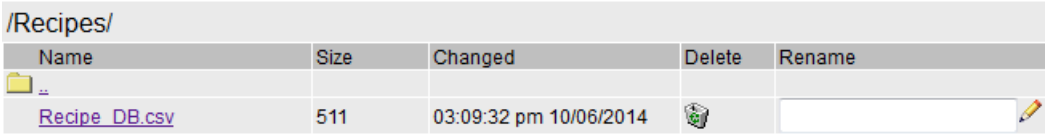
Via the "File browser" tab open the "Recipes" folder in the CPU memory.

The CPU outputs the CSV file as soon as the "RecipeExport" instruction executes. To have the current status of the CSV file displayed, click on the "Update automatically" button. Below the function is displayed as "On".

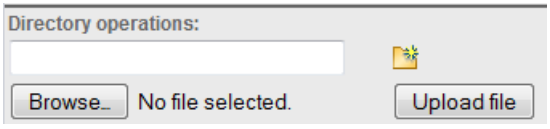
Editing the CSV file

To edit the CSV file, you have the following options:

- You open the CSV file in the Web server.
- You save the CSV file on the PG/PC.
  - **Note**: To be able to load the edited CSV file from the PC in the CPU memory: First delete the CSV file with the same name in the CPU memory



  - After editing, load the CSV file in the CPU memory using the Web server.



Changing component values during runtime

To change component values of the recipe DB during runtime (e.g. Recipe_DB.Product[4]) use a watch table.

	i	Name	...	Display format	Monitor value	Modify value
1		"Recipe_DB".Products[4].productname		String	'Not_used'	'Mix'
2		"Recipe_DB".Products[4].water		DEC	0	10
3		"Recipe_DB".Products[4].barley		DEC	0	2
4		"Recipe_DB".Products[4].wheat		DEC	0	2
5		"Recipe_DB".Products[4].hops		DEC	0	4
6		"Recipe_DB".Products[4].yeast		DEC	0	30
7		"Recipe_DB".Products[4].waterTmp		DEC	0	60
8		"Recipe_DB".Products[4].mashTmp		DEC	0	60
9		"Recipe_DB".Products[4].mashTime		DEC	0	10
10		"Recipe_DB".Products[4].QTest		DEC	0	0

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

Data logging (S7-1200, S7-1500)

Data logging - Overview (S7-1200, S7-1500)

Saving process values

The data logging instructions are used in the user program to save process values to data logs. Data logs can be saved on the Memory Card (MC) or in the internal load memory. The data logs are saved in CSV format (Comma Separated Value).

NOTICE

Special characters in character strings

If you use character strings within the data log, it is strongly recommended that you do not use the delimiter of the CSV file within these character strings. If you ignore this recommendation, opening the CSV file e.g. with MS Excel may lead to problems. Instead of the delimiter of the CSV file, use a different character such as "-", "/" or the space " ".

Also the character that indicates the end of a character string and CR andLF are special characters in a character string. They are not handled correctly by the instructions and cause problems.

The data types are converted into a character string according to the following rules:

Data type		Format	Note
Floating-point numbers	LReal	+9.9999999999999E+999	Always written exponentially
	Real	+9.9999999E+99	Always written exponentially
Signed integers	LInt	+9999999999999999999	Value range of signed integers: -9223372036854775808 .. +9223372036854775807
	DInt	+9999999999	Value range of integers: -2147483648 to +2147483647
	Int	+99999	Value range of integers: -32768 to +32767
	SInt	+999	Value range of integers: -128 to +127
Unsigned integers	ULInt	+99999999999999999999	Value range of integers: 0 to +18446744073709551615
	UDInt	+9999999999	Value range of integers: 0 to +4294967295
	UInt	+99999	Value range of integers: 0 to +65535
	USInt	+999	Value range of integers: 0 to +255
Binary numbers	LWord	+99999999999999999999	Value range of integers: 0 to +18446744073709551615
	DWord	+9999999999	Value range of integers: 0 to +4294967295
	Word	+99999	Value range of integers: 0 to +65535
	Byte	+999	Value range of integers: 0 to +255
	Bool	9	Value range: 0 or 1

Data type		Format	Note
Date and time	LTIME	dddddd:hh:mm:ss. 999_999_999	ISO format with milliseconds, microseconds and nanoseconds
	TIME	hhh:mm:ss.999	ISO format with milliseconds
	S5TIME	hhh:mm:ss.999	ISO format with milliseconds
	LDT	YYYY-MM-DD hh:mm:ss. 999_999_999	ISO format with milliseconds, microseconds and nanoseconds
	DTL	YYYY-MM-DD hh:mm:ss. 999_999_999	ISO format with milliseconds, microseconds and nanoseconds
	DT	YYYY-MM-DD hh:mm:ss.999	ISO format with milliseconds
	DATE	YYYY-MM-DD	ISO format
	LTime_Of_ Day	hh:mm:ss.999_999_999	ISO format
	TOD	hh:mm:ss.999	ISO format
Characters	WString	"abcd"	• Character string inside double quotation marks • The current length is padded with spaces up to the maximum length. • A character string of data type WString consists of elements of the WChar data type. When a data log is saved, the contents of characters of the WChar data type are limited to 16#FF (example: 16#1255 is changed to 16#55).
	String	"abcd"	• Character string inside double quotation marks • The current length is padded with spaces up to the maximum length.
	WChar	"a"	Individual character inside double quotation marks. When a data log is saved, the contents of characters of the WChar data type are limited to 16#FF (example: 16#1255 is changed to 16#55).
	Char	"a"	Individual character inside double quotation marks

The data logging instructions are used in your program to create or open a data log, to write an entry and to close the data log file.

⚠ WARNING

Runtime of the data logging instructions

The conditions specified below slow down the data logging instructions. This effect becomes stronger with the growing size and number of data logging files.

- Large file size of the data log file
- Large number of data log files on the memory card
- Simultaneous access to a log file by the Web server

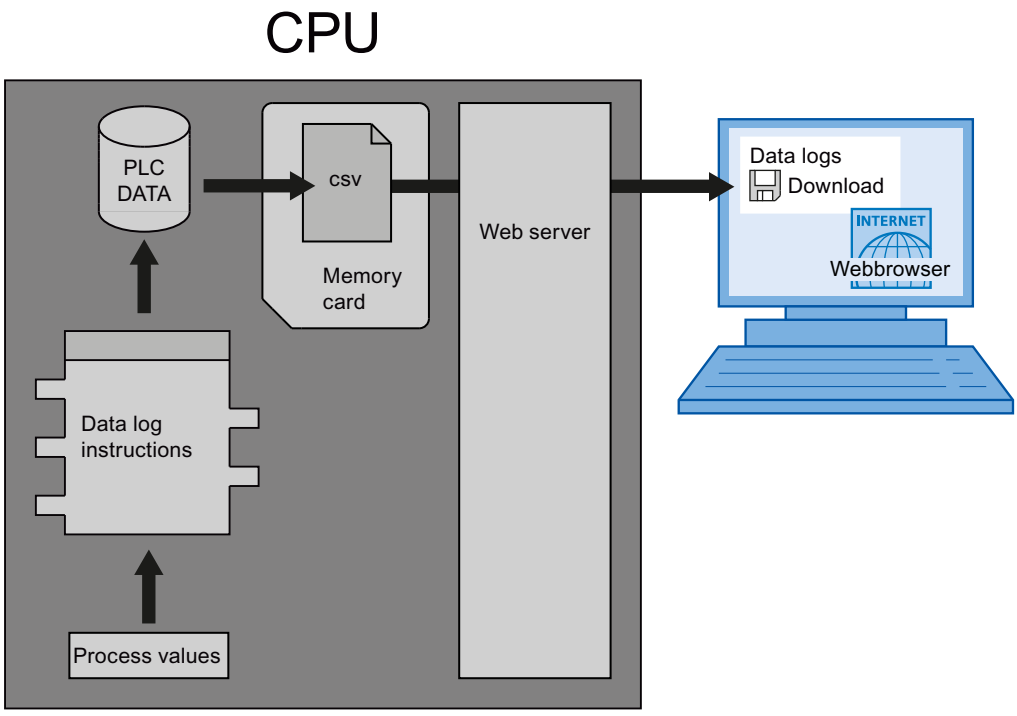
The runtime of some data logging instructions is not typically impaired by the size and number of data logs (e.g. "DataLogWrite"); the instructions "DataLogCreate", "DataLogNewFile", "DataLogClear" and "DataLogOpen", on the other hand, show the behavior described.

Note

Data logging instructions with which data is transferred

Because the data logging functions are executed asynchronously, you must keep the data in the source area consistent until the DONE parameter or the ERROR parameter changes to the value "1".

You decide by the creation of the data buffer which program values are stored in the data log during the creation of the data buffer. The data buffer is used as a memory for new data log entries. New values have to be written to the buffer before "DataLogWrite (Page 4029)" is called. During execution of the "DataLogWrite (Page 4029)" instruction, data is written from the buffer to a data log record.



Data log files can be copied to the PC as follows:

- If the PROFINET interface is connected to the PC, you use a Web browser to access the data logs via the Web server. The CPU can be in RUN or STOP mode for this. If the CPU is in "RUN" mode, the program continues running while the Web server is transferring data.
- If there is a memory card in the CPU, you can remove this card and insert it into a standard slot for SD (Secure Digital) cards or MMC (MultiMediaCard) cards on a PC or programming device. Use File Manager to transfer the data log files from the memory card to the PC. When you remove the memory card, the CPU goes to "STOP".

Note

Service life of memory cards

The service life of memory cards is limited. Additional information is available here: AUTOHOTSPOT

Properties of the data log

Writing of the data records of a data log is carried out in accordance with the principle of a ring buffer. New data records are added until the maximum number of data records is reached (RECORD parameter). The next data record then overwrites the "oldest" data record of the data log.

If you want to prevent data records from being overwritten, use the "DataLogNewFile (Page 4034)" instruction to create a new data log file based on the current data log. New data records are then written into the new data log.

Creating data logs

With the "DataLogCreate (Page 4016)" instruction, you can create a new data log file in the ""\DataLogs" directory of the load memory.

- The name assigned at the NAME parameter is the designation for the data log and is also used as the file name for the CSV file. The file is stored in the directory "DataLogs".
- The block parameter DATA determines the data buffer for the new data log object and the columns and data types in the data log. The columns and data types of a data record in the data log are determined by the elements of the structure declaration or array declaration of this data buffer. Each element of a structure or of an array corresponds to a column in a line in the data log.
- You use the HEADER block parameter to assign a header text in the header to each column.
- The "DataLogCreate (Page 4016)" instruction returns an ID. This ID is used by the other data logging instructions as a reference for the created data log.

Opening data logs

You use the instruction "DataLogOpen" (S7-1200 and S7-1500) to open an existing data log on the memory card. A data log must be open before you can write new data records to it.

Data log opens automatically when you execute the instructions "DataLogCreate (Page 4016)" and "DataLogNewFile (Page 4034)".

Up to 10 data logs can be open simultaneously. The data log to be opened can be selected using the ID or name of the data log.

- If you specify the ID and the name of the data log in the ID and NAME parameters, respectively, the data log will be identified based on the ID. The data log name will not be compared.
- If you select the data log using the parameter NAME and specify 0 as the ID, the ID belonging to the data log will be displayed in the ID parameter when you open the data log.
- If you select the data log using the ID parameter and no name is specified, the name will be displayed in the NAME parameter when the data log is opened.

You use the MODE parameter to specify whether the data records of the data log are to be deleted upon opening.

You can run a consistency check for "DataLogOpen (S7-1500).

Writing to data logs

A data log must be open ("DataLogOpen (Page 4023)" instruction) before a data record can be written to a data log. The "DataLogWrite (Page 4029)" instruction writes a data record to the data log.

Closing data logs

With the "DataLogClose (Page 4031)" instruction, you close and open data log. You select the data log using the ID parameter.

The data log is closed automatically when the CPU switches to STOP and upon restart.

Deleting data logs

You use the instruction "DataLogDelete (Page 4033)" (S7-1500) to delete a data log file from the memory card. The data log and the data records it contains can only be deleted if the log was created with the instruction "DataLogCreate (Page 4016)".

Select the data log to be deleted using the NAME and ID parameters. The ID parameter is evaluated first. If there is a data log with the relevant ID, the NAME parameter will not be

evaluated. If the value "0" is used at the ID parameter, a value with the data type STRING must be used at the NAME parameter.

Note

Deleting data logs on the memory card

You can copy data logs from a memory card using a PC card reader, but you must not change them or delete them using a PC card reader.

In addition to the data log instructions, the use of a Web server is recommended to display, download (copy) and delete data logs.

Calling the file system of the memory card in the file Explorer directly brings with it the risk that you accidentally delete or change data log or other system files so that a file becomes damaged or the memory card becomes unusable. Changes to data logs on the memory card that you made outside the PLC have the effect POWER ON time of the PLC is increased.

Clearing data logs

You use the instruction "DataLogClear (Page 4027)" (S7-1500) to delete all the data records in an existing data log. The instruction does not delete the optional header of the CSV file (see the description of the HEADER parameter of the instruction "DataLogCreate (Page 4016)").

You use the ID parameter to select the data log whose data records are to be deleted. Before you can delete data records, the data log must be open.

New file for data logs

You use the instruction "DataLogNewFile (Page 4034)" (S7-1200) or "DataLogTypedNewFile (Page 4036)" (S7-1500) to create a new data log with the same properties as an existing data log. This allows the contents of an existing data log to be retained.

When called, the instruction creates a new data log on the memory card or in the internal load memory with the name defined at the NAME parameter. You use the ID parameter to specify the ID of the old data log whose properties you want to apply to the new data log. The ID of the new data log is then output via the ID parameter.

You specify the file size of the new data log with the RECORDS parameter of the instruction.

You can run a consistency check for "DataLogTypedNewFile (Page 4036)" (S7-1500).

DataLogCreate: Create data log (S7-1200, S7-1500)

Description

With the "DataLogCreate" instruction you create a data log.

The data log is saved on the memory card or in the internal load memory in the directory "\DataLogs". The amount of data that is stored in a data log is dependent on the available space on the memory card or the storage space in the internal load memory of the CPU used. The maximum size of the generated CSV file is 500,000,000 bytes for S7-1200 CPUs, 1,000,000,000 bytes for S7-1500 CPUs.

You specify the maximum number of data records that is stored in a data log in the RECORDS parameter. When the specified maximum number of data records in the data log is reached, the oldest data record is overwritten. To avoid overwriting of existing data records, use the "DataLogNewFile (Page 4034)" instruction. The instruction is used to create a new data log with the same structure when the number specified at the RECORDS parameter is reached (return value 1 at the STATUS parameter of the "DataLogWrite (Page 4029)" instruction). The data records are then saved in the new data log.

You specify the name for the data log in the NAME parameter. The data log is created in CSV (Comma Separated Value) format. With the HEADER parameter, you create an (optional) header for the data log.

Once the data log is created, it is opened automatically. This means that data can be written.

Parameters

The following table shows the parameters of the "DataLogCreate" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Execution of the instruction The data log is created with a rising edge at the parameter REQ.
RECORDS	Input	UDInt	I, Q, M, L, D or constant	Maximum number of data records in the data log If the "DataLogWrite (Page 4029)" instruction writes more records than specified in this parameter, the oldest record is overwritten.
FORMAT	Input	UInt	I, Q, M, L, D or constant	Data format: • 0: Internal (not supported) • 1: CSV (Comma separated values)
TIMESTAMP	Input	UInt	I, Q, M, L, D or constant	Time stamp: • 0: No time stamp • 1: Date and time The extra columns in the header are automatically added if time-stamping is activated.

Parameter	Declaration	Data type	Memory area	Description
NAME	Input	VARIANT	L, D	Name of the data log The specified name is also used as a file name for the csv file. With S7-1200 CPUs. the following restrictions apply to name of the data log: • The name must not be longer than 35 characters. • All ASCII characters between 0x20 and 0x7E are permitted with the exception of \ ' / " : ; [ ] | = . * ? < > With S7-1500 CPUs. the following restrictions apply to name of the data log: • The name must not be longer than 55 characters. • The following characters are permitted: 0 ... 9, a ... z, A ... Z as well as - and _
ID	InOut	DWORD	I, Q, M, L, D	Object ID of the data log (only output) The ID of the data log is required for further data logging instructions to address the created data log.
HEADER	InOut	VARIANT	D	Header of the data log (optional) The parameter is hidden after the instruction is added. The header is written as the first line in the CSV file.
DATA	InOut	VARIANT	D	Pointer to the data structure that is written as data record when executing the "DataLogWrite (Page 4029)" instruction.
DONE	Output	BOOL	I, Q, M, L, D	Status parameter: • 0: Processing not yet complete. • 1: Processing of instruction finished successfully.
BUSY	Output	BOOL	I, Q, M, L, D	Status parameter: • 0: Processing of the instruction has not started, completed, or canceled. • 1: Processing of the instruction is in progress.
ERROR	Output	BOOL	I, Q, M, L, D	Status parameter: • 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, L, D	Detailed status information: Detailed error and status information is output at the parameter STATUS. The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter HEADER

The HEADER parameter is a VARIANT pointer to a data block that defines a header for the CSV file (header). The header is always the first line in the CSV file representation.

- The header preset using HEADER is transferred to the CSV file. Here there is no check as to whether it is consistent with the remainder of the data log. In particular, there is no check whether the number of columns separated by commas matches the number of data columns.
- A STRING, Array of BYTE, or Array of CHAR data type can be used for the individual column names. When the STRING data type is used, the length is limited to 254 bytes. When the Array of BYTE and Array of CHAR are used, 5000 elements are possible.

If no header is to be created, do not specify a value in the HEADER parameter.

Parameter DATA

The DATA parameter is a VARIANT pointer to an elementary data type, a structure or an array in a data block. An element of a structure or an array corresponds to a column in the data log with a specific data type.

Note the following when creating the data block:

- Normally, the number of columns matches the number of columns defined in the HEADER parameter. This is however not stipulated.
- Each element of the structure or the array corresponds to a column entry in the CSV file. Therefore, no nesting structures (STRUCT in STRUCT) may be used when using the data type STRUCT. However the use of arrays within a data type is permitted STRUCT with the following exception: Arrays whose elements are of the data type "PLC data type (UDT)" are not permitted.
- The data structure can contain up to 256 elements. If there are more than 256 elements, error code 8C52 is output at the STATUS parameter.
- The tags of the data block can be set as retentive or non-retentive tags. However, the retentivity setting must be the same for all tags of the data block.

Parameter STATUS (S7-1200)

Error code* (W#16#...)	Description
0	No errors.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogCreate instructions has already been reached.
8090	Invalid file name (see description of the NAME parameter).

Error code* (W#16#...)	Description
8093	Data log already exists.
8097	The length of the file exceeds the limitation of the file system or the CPU-specific limitation.
80A2	Write error returned from file system.
80B2	Maximum number of data logs exceeded
80B3	Insufficient memory space on the memory card.
80B4	The memory card is write-protected.
80C0	Access currently not possible.
80C1	Too many data logs are open.
8253	The RECORDS parameter has an invalid value.
8353	Invalid format selection
8453	Invalid time stamp
8B24	Invalid assignment at parameter HEADER (for example: points to local memory).
8B51	Impermissible data type at parameter HEADER or length exceeds the maximum size.
8C24	Invalid assignment at parameter DATA (for example: points to local memory).
8C51	Impermissible data type at parameter DATA / data structure cannot be used.
8C52	Structure at the parameter DATA contains more than 256 elements.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Parameter STATUS (S7-1500)

Error code* (W#16#...)	Description
0	No errors.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
8070	The maximum number of simultaneously active DataLogCreate instructions has already been reached.
8090	Invalid file name (see description of the NAME parameter).
8091	"NAME" parameter is not a string.
8093	Data log already exists.
8097	The length of the file exceeds the limitation of the file system or the CPU-specific limitation.
80A2	Write error returned from file system.
80B2	Maximum number of data logs exceeded or data log resources used up
80B3	Insufficient memory space on the memory card.
80B4	The memory card is write-protected.
80C0	Access currently not possible.
80C1	Too many data logs are open.
8253	The RECORDS parameter has an invalid value.
8353	Invalid format selection.
8453	Invalid time stamp.
8B24	Invalid assignment at parameter HEADER (for example: points to local memory).
8B51	Impermissible data type at parameter HEADER or length exceeds the maximum size.

Error code* (W#16#...)	Description
8C24	Invalid assignment at parameter DATA (for example: points to local memory).
8C51	Impermissible data type at parameter DATA / data structure cannot be used.
8C52	Structure at the parameter DATA contains more than 256 elements.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

In the following example, a simple data log is created with a time stamp and three process values.

Tags in the global data block

The values for the input parameters of the data log are stored in the global data block "DataLogDB":

- DataLogName (String): The tag contains the name of the data log which is also used as the file name for the CSV file.
- DataLogID (DInt): The ID of the data log is written in this tag when the instruction is called.
  - The ID is automatically assigned by the instruction.
  - Use the tag "DataLogID" in other DataLog instructions in order to address this data log.
- MyHeader (String): This tag contains the header of the data log, i.e. the column headings for the process values.
- MyData (Struct): The tag contains the three process values that are written in the data log. Each time a record is written (instruction "DataLogWrite (Page 4029)") the current values are written in a new record.

DataLogDB

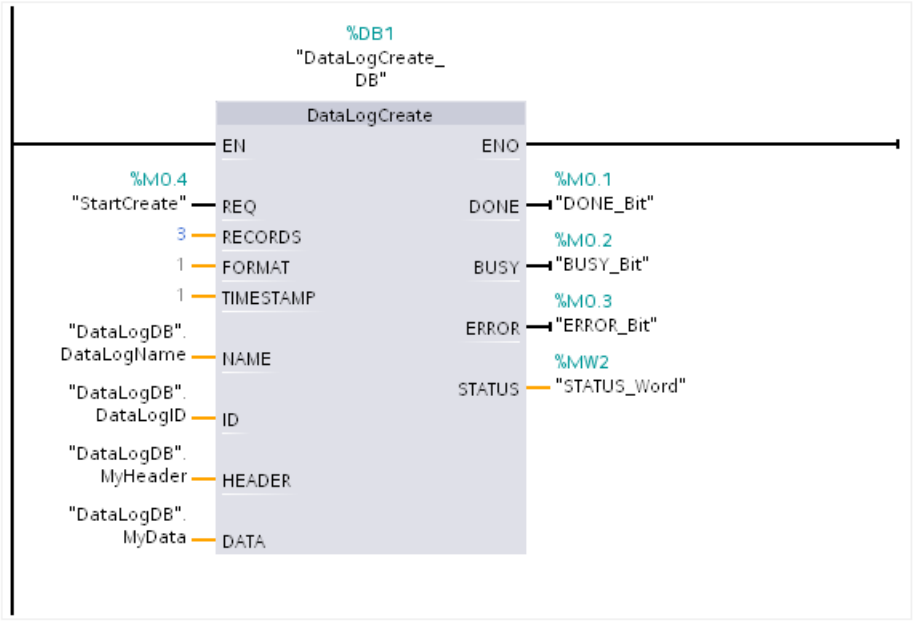
	Name	Data type	Start value	Retain
1	Static			
2	DataLogName	String	'MyDataLog'	
3	DataLogID	DInt	0	
4	MyHeader	String	'Value1,Value2,Value3'	
5	MyData	Struct		
6	ProcessValue1	Int	2	
7	ProcessValue2	Int	3	
8	ProcessValue3	Int	4	

Calling the "DataLogCreate" instruction

The instruction is called with the following input parameters:

- REQ (BOOL): When REQ = "1" the data log is created.
- RECORD (3): A maximum of three data records can be written in the data log. Thereafter, the oldest data record is overwritten.

- FORMAT (1): The data log is created as a CSV file.
- TIMESTAMP (1): Enabled: Two additional columns are automatically created (date and time) for the data log. The current time stamp is written in the data record each time "DataLogWrite (Page 4029)" is executed.
- NAME (VARIANT): Pointer to the tag "DataLogName" in the data block "DataLogDB".
- ID (VARIANT): Pointer to the tag "DataLogID" in the data block "DataLogDB" (output only).
- HEADER (VARIANT): Pointer to the tag "MyHeader" in the data block "DataLogDB".
- DATA (VARIANT): Pointer to the tag "MyData" in the data block "DataLogDB".



Readout of the data log via the web server

You read out the created data log with the web server:

- Activate the web server in the properties of the CPU. The authorization "...to read files" must be activated for the web server in order to readout the data log.
- You can access the web server via the Internet browser by entering the IP address of the CPU as a URL.
- Under "Filebrowser" the directory "\DataLogs" was automatically created which contains the data log.
- The DataLog only contains one entry "//END" if the instruction "DataLogWrite (Page 4029)" has still not been executed. The first data record is written after the initial execution of "DataLogWrite (Page 4029)".

	A	B	C	D	E	F
1	SeqNo	Date	Time	ProcessValue1	ProcessValue2	ProcessValue3
2	1	01.07.2013	12:33:42.917	2	3	4
3						
4						
5						

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Data logging - Overview (Page 4011)

Difference between synchronous and asynchronous instructions (Page 2571)

DataLogOpen: Open data log (S7-1200, S7-1500)

DataLogOpen: Open data log (S7-1200)

Description

You use the "DataLogOpen" instruction to open an existing data log on the memory card. To write new data records to the data log, a data log must be open.

Data log opens automatically when you execute the instructions "DataLogCreate (Page 4016)" and "DataLogNewFile (Page 4034)".

Up to 10 data logs can be open simultaneously. The data log to be opened can be selected using the ID or name of the data log.

- If you specify the ID and the name of the data log in the ID and NAME parameters, respectively, the data log will be identified based on the ID. The data log name will not be compared.
- If you select the data log using the parameter NAME and specify 0 as the ID, the ID belonging to the data log will be displayed in the ID parameter when you open the data log.
- If you select the data log using the ID parameter and no name is specified, the name will be displayed in the NAME parameter when the data log is opened.

You use the MODE parameter to specify whether the data records of the data log are to be deleted upon opening.

Parameters

The following table shows the parameters of the "DataLogOpen" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D or constant	Execution of the instruction upon a rising edge.
MODE	Input	UInt	I, Q, M, L, D or constant	Mode for opening the data log: • MODE= "0" Data records of the data log are retained • MODE= "1" Data records of the data log are deleted, but the header is retained
NAME	Input	VARIANT	L, D	(File) name of the data log.
ID	InOut	DWORD	I, Q, M, L, D	Object ID of the data log.
DONE	Output	BOOL	I, Q, M, L, D	Instruction was executed successfully.
BUSY	Output	BOOL	I, Q, M, L, D	Execution of instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, L, D	• 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, L, D	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
0	No errors.
2	Warning: Data log file has already been opened by this application.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogOpen instructions has already been reached.
8090	There are inconsistencies between the data log definition and existing data log data.
8091	A data type other than String was used at the NAME parameter.
8092	Data log does not exist.
80B4	The memory card is write-protected.

Error code* (W#16#...)	Description
80C1	Too many files open.
The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Example program for working with data logs (Page 4038).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Data logging - Overview (Page 4011)

Difference between synchronous and asynchronous instructions (Page 2571)

DataLogOpen: Open data log (S7-1500)

Description

You use the "DataLogOpen" instruction to open an existing data log on the memory card. A data log must be open to write new data records to it.

Data log opens automatically when you execute the instructions "DataLogCreate (Page 4016)" and "DataLogNewFile (Page 4034)".

Up to 10 data logs can be open simultaneously. The data log to be opened can be selected using the ID or name of the data log.

- If you specify the ID and the name of the data log in the ID and NAME parameters, respectively, the data log will be identified based on the ID. The data log name will not be compared.
- If you select the data log using the parameter NAME and specify 0 as the ID, the ID belonging to the data log will be displayed in the ID parameter when you open the data log.
- If you select the data log using the ID parameter and no name is specified, the name will be displayed in the NAME parameter when the data log is opened.

You use the MODE parameter to specify whether the data records of the data log are to be deleted upon opening.

The parameter DATA enables a consistency check between the data log to be opened and the definition of the data log of the instruction "DataLogCreate (Page 4016)". A check is made as to whether the data type of the DATA parameter is identical to the data type of "DataLogCreate". If this is not the case, the error code W#16#8090 is output at the STATUS parameter.

Parameters

The following table shows the parameters of the "DataLogOpen" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Execution of the instruction upon a rising edge.
MODE	Input	UInt	I, Q, M, L, D or constant	Mode for opening the data log: • MODE= "0" Data records of the data log are retained • MODE= "1" Data records of the data log are deleted, but the header is retained
NAME	Input	VARIANT	L, D	(File) name of the data log.
ID	InOut	DWORD	I, Q, M, L, D	Object ID of the data log.
DATA	InOut	VARIANT	D	With consistency check: Pointer to the data area at the DATA parameter of the "DataLogCreate (Page 4016)" instruction.
DONE	Output	BOOL	I, Q, M, L, D	Instruction was executed successfully.
BUSY	Output	BOOL	I, Q, M, L, D	Execution of instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, L, D	• 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, L, D	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
0	No errors.
2	Warning: Data log file has already been opened by this application.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogOpen instructions has already been reached.

Error code* (W#16#...)	Description
8090	Data types inconsistent. The data log at parameter ID uses different data types than specified at parameter DATA.
8091	A data type other than String was used at the NAME parameter.
8092	Data log does not exist.
80B4	The memory card is write-protected.
80C1	Too many files open.
The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Example program for working with data logs (Page 4038).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Data logging - Overview (Page 4011)

Difference between synchronous and asynchronous instructions (Page 2571)

DataLogClear: Empty data log (S7-1200, S7-1500)

Description

The "DataLogClear" instruction deletes all data records in an existing data log. The instruction does not delete the optional header of the CSV file (see the description of the HEADER parameter of the instruction "DataLogCreate (Page 4016)").

You use the ID parameter to select the data log whose data records will be deleted.

Requirement

Before you can delete data records, the data log must be open (see "DataLogOpen (Page 4023) instruction").

Parameters

The following table shows the parameters of the "DataLogClear" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Execution of the instruction upon a rising edge.
ID	InOut	DWORD	I, Q, M, D, L	Object ID of the data log
DONE	Output	BOOL	I, Q, M, D, L	Instruction was executed successfully.
BUSY	Output	BOOL	I, Q, M, D, L	Execution of the instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, D, L	• 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output at the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

Error code* (W#16#...)	Explanation
0000	No error.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogClear instructions has already been reached.
8080	The data log file selected with the ID parameter is not processed by the "DataLogClear" instruction.
8092	Data log does not exist.
80A2	Write error signaled back by the file system.
80B0	Data log is not open.
80B4	The memory card is write-protected.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

See also

Data logging - Overview (Page 4011)

Difference between synchronous and asynchronous instructions (Page 2571)

DataLogWrite: Write data log (S7-1200, S7-1500)

Description

With the "DataLogWrite" instruction you write a data record into an existing data log. With the ID parameter you select the data log into which the data record is to be written. To create a new data record, the data log must be open. The instruction creates a new data record in the format that was specified in the DATA parameter when the data log was created.

Before calling the "DataLogWrite" instruction, transfer the data to the tag that you interconnected at the DATA parameter of the "DataLogCreate" instruction. When the "DataLogWrite" instruction is executed, the transferred data is copied to the data log.

NOTICE
Data log data loss when the power supply to the CPU is interrupted If the power supply is interrupted during execution of the "DataLogWrite" instruction, the data record to be transferred is lost.

Parameters

The following table shows the parameters of the "DataLogWrite" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Execution of the instruction upon a rising edge.
ID	InOut	DWORD	I, Q, M, L, D	Object ID of the data log
DONE	Output	BOOL	I, Q, M, L, D	Instruction was executed successfully.
BUSY	Output	BOOL	I, Q, M, L, D	Execution of the instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, L, D	• 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, L, D	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS (S7-1200)

Error code* (W#16#...)	Description
0	No errors
0001	Last possible data record created at the end of the file. Creating another data record will overwrite an older data record.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogWrite instructions has already been reached.
8090	Data log definition does not match the existing data log.
8092	Data log does not exist.
80A2	Write error signaled back by the file system.
80B0	Data log is not open.
80B3	Insufficient memory space on the memory card.
80B4	The memory card is write-protected.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Parameter STATUS (S7-1500)

Error code* (W#16#...)	Description
0	No errors
0001	Last possible data record created at the end of the file. Creating another data record will overwrite an older data record.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
8070	The maximum number of simultaneously active DataLogWrite instructions has already been reached.
8090	Data log definition does not match the existing data log.
8092	Data log does not exist.
80A2	Write error signaled back by the file system.
80B0	Data log is not open.
80B3	Insufficient memory space on the memory card.
80B4	The memory card is write-protected.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Example program for working with data logs (Page 4038).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Data logging - Overview (Page 4011)

Difference between synchronous and asynchronous instructions (Page 2571)

DataLogClose: Close data log (S7-1200, S7-1500)

Description

With the "DataLogClose" instruction, you close an open data log. You select the data log using the ID parameter.

Note

Closing data logs automatically

The data log is closed automatically when the CPU goes to STOP or if there is a restart.

Parameters

The following table shows the parameters of the "DataLogClose" instruction:

Parameters	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Execute function on a rising edge.
ID	InOut	DWORD	I, Q, M, L, D	Object ID of the data log
DONE	Output	BOOL	I, Q, M, L, D	Instruction was executed successfully.
BUSY	Output	BOOL	I, Q, M, L, D	Execution of the instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, L, D	• 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, L, D	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS (S7-1200)

Error code* (W#16#...)	Description
0	No errors
1	Data log is not open
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogClose instructions has already been reached.
8092	Data log does not exist.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Parameter STATUS (S7-1500)

Error code* (W#16#...)	Description
0	No errors
1	Data log is not open
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
8070	The maximum number of simultaneously active DataLogClose instructions has already been reached.
8092	Data log does not exist.
80B4	The memory card is write-protected.
80C0	Access currently not possible.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Example program for working with data logs (Page 4038).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

DataLogDelete: Delete data log (S7-1200, S7-1500)

Description

You use the "DataLogDelete" instruction to delete a data log file from the memory card. The data log and the data records it contains are only deleted if they were created with the "DataLogCreate" or "DataLogNewFile" instruction.

Parameters

The following table shows the parameters of the "DataLogDelete" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Execution of the instruction upon a rising edge.
NAME	Input	VARIANT	L, D	File name of the data log
DELFILE	Input	BOOL	I, Q, M, D, L or constant	• 0: Data log is retained. • 1: Data log is deleted.
ID	InOut	DWORD	I, Q, M, D, L	Object ID of the data log
DONE	Output	BOOL	I, Q, M, D, L	Instruction was executed successfully.
BUSY	Output	BOOL	I, Q, M, D, L	Deletion of the data log is not yet complete.
ERROR	Output	BOOL	I, Q, M, D, L	• 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output at the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameters NAME and ID

Select the data log to be deleted using the NAME and ID parameters. The ID parameter is evaluated first. If there is a data log with the relevant ID, the NAME parameter will not be evaluated. If the value "0" is used at the ID parameter, a value with the data type STRING must be used at the NAME parameter.

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0	No error.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogDelete instructions has already been reached.
8091	A data type other than STRING is being used at the NAME parameter.
8092	Data log does not exist.
80A2	Write error signaled back by the file system.
80B4	The memory card is write-protected.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

See also

DataLogCreate: Create data log (Page 4016)

Difference between synchronous and asynchronous instructions (Page 2571)

DataLogNewFile: Data log in new file (S7-1200, S7-1500)

DataLogNewFile: Data log in new file (S7-1200)

Description

With the "DataLogNewFile" instruction you create a new data log with the same properties as an already existing data log. This allows the contents of an existing data log to be retained.

When called, the instruction creates a new data log on the memory card or in the internal load memory with the name defined at the NAME parameter. You use the ID parameter to specify the ID of the old data log whose properties you want to apply to the new data log. The ID of the new data log is then output via the ID parameter.

You specify the file size of the new data log with the RECORDS parameter of the instruction.

Once the new data log is created, it is opened automatically. This means that data can be written.

Parameters

The following table shows the parameters of the "DataLogNewFile" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D or constant	Execution of the instruction upon a rising edge.
RECORDS	Input	UDInt	I, Q, M, L, D or constant	Number of data records in the data log.
NAME	Input	VARIANT	L, D	File name of the new data log.
ID	InOut	DWORD	I, Q, M, L, D	Object ID of the data log • In: ID of the existing data log • Out: ID of the new data log
DONE	Output	BOOL	I, Q, M, L, D	Instruction was executed successfully.
BUSY	Output	BOOL	I, Q, M, L, D	Execution of the instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, L, D	• 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, L, D	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
0	No errors.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogNewFile instructions has already been reached.
8090	Invalid file name.
8091	Data type at parameter NAME is not STRING.
8092	Source data log does not exist.
8093	New data log already exists.
8097	File length exceeds the file system limit.
80A0	Data types inconsistent. The data log at parameter ID uses different data types than specified at parameter DATA.
80A2	Write error signaled back by the file system.
80B2	Maximum number of data logs exceeded or data log resources used up

Error code* (W#16#...)	Description
80B3	Load memory not sufficient.
80B4	The memory card is write-protected.
80C1	Too many files open.
8253	The RECORDS parameter has an invalid value.
The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Example program for working with data logs (Page 4038).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

DataLogNewFile: Data log in new file (S7-1500)

Description

With the "DataLogNewFile" instruction you create a new data log with the same properties as an already existing data log. This allows the contents of an existing data log to be retained.

When called, the instruction creates a new data log on the memory card or in the internal load memory with the name defined at the NAME parameter. You use the ID parameter to specify the ID of the old data log whose properties you want to apply to the new data log. The ID of the new data log is then output via the ID parameter.

You specify the file size of the new data log with the RECORDS parameter of the instruction.

The parameter DATA enables a consistency check between the new data log to be created and the definition of the data log of the instruction "DataLogCreate (Page 4016)". A check is made as to whether the data type of the DATA parameter is identical to the data type of "DataLogCreate". If this is not the case, the error code W#16#80A0 is output at the STATUS parameter.

Once the new data log is created, it is opened automatically. This means that data can be written.

Parameters

The following table shows the parameters of the "DataLogNewFile" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, L, D, T, C or constant (T and C are only available in LAD and FBD with S7-1500)	Execution of the instruction upon a rising edge.
RECORDS	Input	UDInt	I, Q, M, L, D or constant	Number of data records in the data log.
NAME	Input	VARIANT	L, D	File name of the new data log.
ID	InOut	DWORD	I, Q, M, L, D	Object ID of the data log • In: ID of the existing data log • Out: ID of the new data log
DATA	InOut	VARIANT	D	Data type for consistency check
DONE	Output	BOOL	I, Q, M, L, D	Instruction was executed successfully.
BUSY	Output	BOOL	I, Q, M, L, D	Execution of the instruction not yet complete.
ERROR	Output	BOOL	I, Q, M, L, D	• 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, L, D	Status parameter The parameter is only set for the duration of one call. To display the status, you should therefore copy the STATUS parameter to a free data area.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
0	No errors.
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8070	The maximum number of simultaneously active DataLogNewFile instructions has already been reached.
8090	Invalid file name
8091	Data type at parameter NAME is not STRING.
8092	Source data log does not exist.
8093	New data log already exists.
8097	The length of the file exceeds the limitation of the file system or the CPU-specific limitations.

Error code* (W#16#...)	Description
80A0	Data types inconsistent. The data log at parameter ID uses different data types than specified at parameter DATA.
80A2	Write error signaled back by the file system.
80B2	Maximum number of data logs exceeded or data log resources used up
80B3	Load memory not sufficient.
80B4	The memory card is write-protected.
80C0	Access currently not possible.
80C1	Too many files open.
8253	The RECORDS parameter has an invalid value.
The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Example program for working with data logs (Page 4038).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

Example program for working with data logs (S7-1200, S7-1500)

Introduction

The following example program shows the main functions of the data log instructions. For detailed information about individual instructions, open the corresponding help description using the corresponding links.

General notes on using data logs

- The data log created is automatically opened when the "DataLogCreate" and "DataLogNew" instructions are executed.
- Data logs are automatically closed after the CPU switches from RUN to STOP or after a restart of the CPU.
- For the "DataLogWrite" instruction to be executed, the data log must be open.
- For S7-1200 CPUs, a maximum of eight data logs can be open simultaneously. For S7-1500 CPUs, a maximum of ten data logs can be open simultaneously.

Example program

Create the following tags for storing the data in a global data block (SLI_gDB_Datalogging).

SLI_gDB_DataLogging

	Name	Data type	Start value
1	Static		
2	newName	String	'myNewDataLog1a'
3	name	String	'myDataLog1a'
4	logID	DWord	16#0
5	logHeader	String	'Count,Temperature,Pressure'
6	myData	Struct	
7	count	Int	0
8	temperature	Real	0.0
9	pressure	Real	0.0
10	executeCreate	Bool	false
11	createDone	Bool	false
12	createBusy	Bool	false
13	createError	Bool	false
14	createStatus	Word	16#0
15	createStatusMem	Word	16#0
16	dlogCreated	Bool	false

The DB ("SLI_gDB_Datalogging") is used to create a data log (DataLogCreate (Page 4016)). The three entries of the structure "myData" are used as process values: count, temerature and pressure. In the data block, these three values are stored temporarily and then transferred with the "DataLogWrite (Page 4029)" instruction to a data log as a data record.

A data record is composed later from the following entries:

- The data record number (assigned automatically)
- The date (assigned automatically when "1" is used for DataLogCreate at the TIMESTAMP parameter).
- The time of day (assigned automatically when "1" is used for DataLogCreate at the TIMESTAMP parameter).
- The current value of "count" from the "myData" structure.
- The current value of "temperature" from the "myData" structure.
- The current value of "pressure" from the "myData" structure.

Use the global data block "SLI_gDB_DataLogW" to make available the process values for writing a data record DataLogWrite (Page 4029)).

SLI_gDB_DataLogW

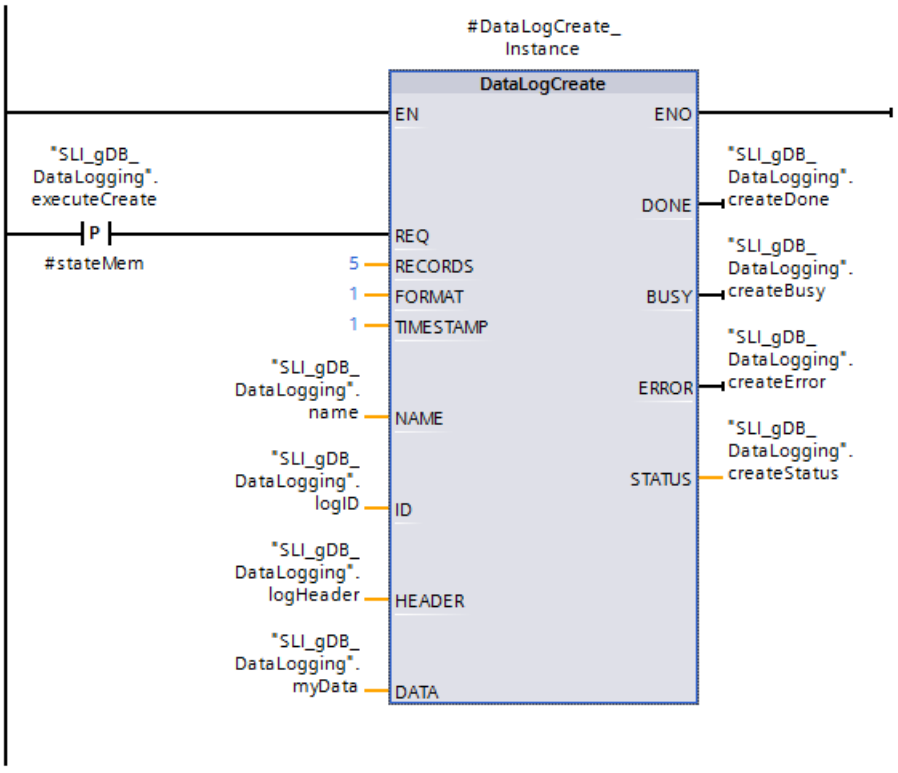
	Name	Data type	Start value
1	Static		
2	executeWrite	Bool	false
3	count	Int	5
4	scaledTemp	Real	25.7
5	scaledPressure	Real	45.0
6	writeDone	Bool	false
7	writeBusy	Bool	false
8	writeError	Bool	false
9	writeFullStatus	Word	16#0
10	closeDone	Bool	false
11	closeBusy	Bool	false
12	closeError	Bool	false
13	closeStatus	Word	16#0
14	executeOpen	Bool	false
15	openDone	Bool	false
16	openBusy	Bool	false
17	openError	Bool	false
18	openStatus	Word	16#0
19	executeNewFile	Bool	false
20	fileDone	Bool	false
21	fileBusy	Bool	false
22	fileError	Bool	false
23	fileStatus	Word	16#0

Create an FB "SLI_FB_DataLogging". Create the following local tags in the FB.

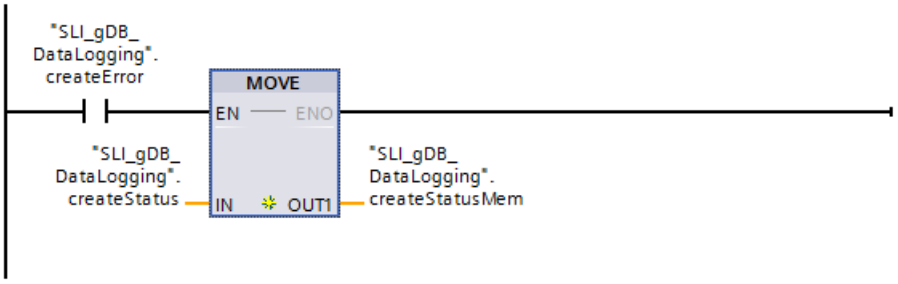
7	Static			
8	DataLogCreate_Instance	DataLogCreate		
9	stateMem	Bool	false	Non-retain
10	stateMem2	Bool	false	Non-retain
11	stateMem3	Bool	false	Non-retain
12	stateMem4	Bool	false	Non-retain
13	stateMem5	Bool	false	Non-retain
14	DataLogWrite_Instance	DataLogWrite		
15	DataLogClose_Instance	DataLogClose		
16	DataLogOpen_Instance	DataLogOpen		
17	DataLogNewFile_Instance	DataLogNewFile		

Create the following interconnections in the FB.

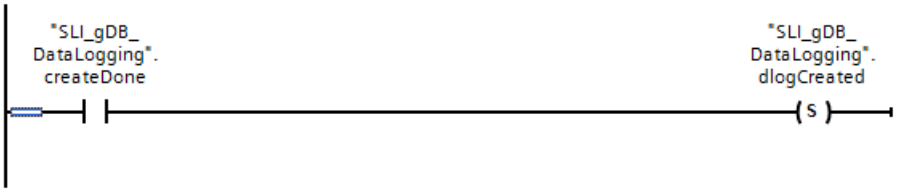
Network 1: A positive edge at the input parameter REQ (" executeCreate") of the instruction DataLogCreate (Page 4016) starts the creation of the data log.



Network 2: In case of error of DataLogCreate, save the status as follows.

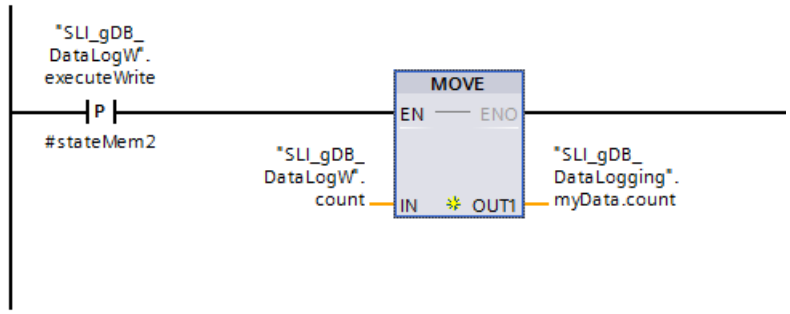


Network 3: The output parameter DONE ("createDone") of DataLogCreate (Page 4016) is only valid for one cycle. Therefore, save its value in the tag "dlogCreated".

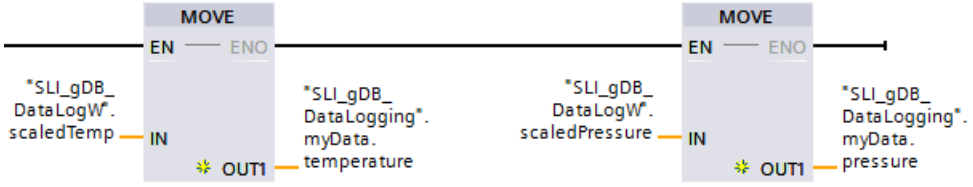


Network 4: A rising edge triggers the point in time at which new process values are stored in the myData structure. This step is used to store the desired process values temporarily in the data block.

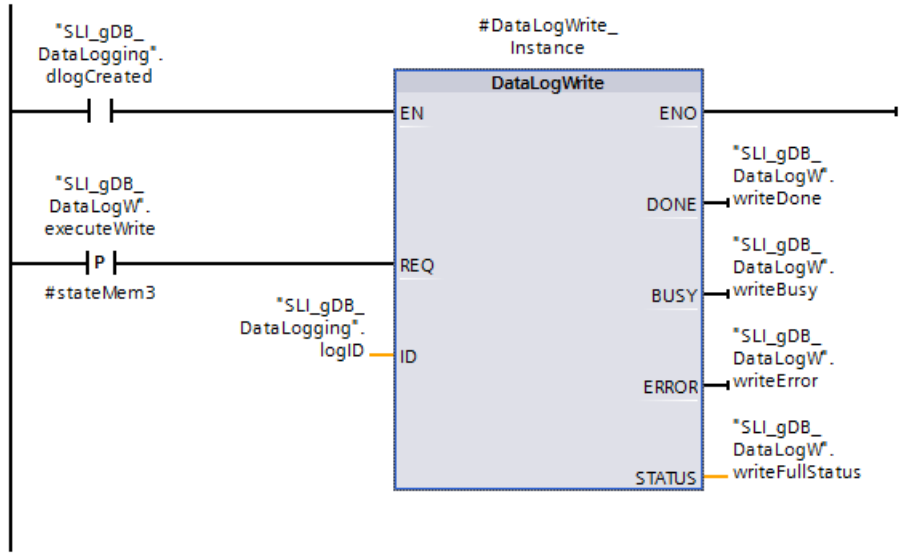
Part 1:



Part 2:

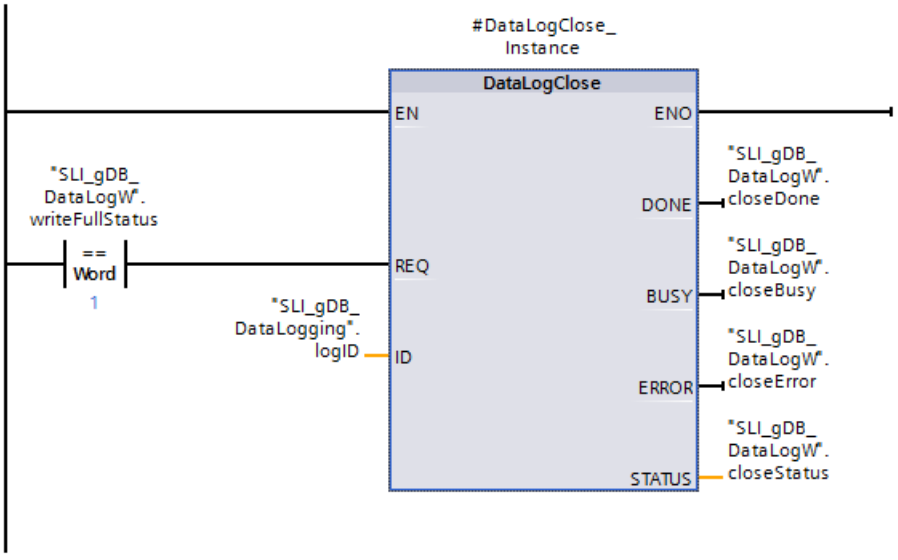


Network 5: Once the execution of DataLogCreate (Page 4016) is complete, ("dlogCreated" is "TRUE"), the EN input of DataLogWrite (Page 4029) is set. The reason for this is that a generation process extends over several cycles and must be completed in order for a write operation to be performed. Writing is triggered for a data record by a rising edge at the input parameter REQ ("executeWrite").

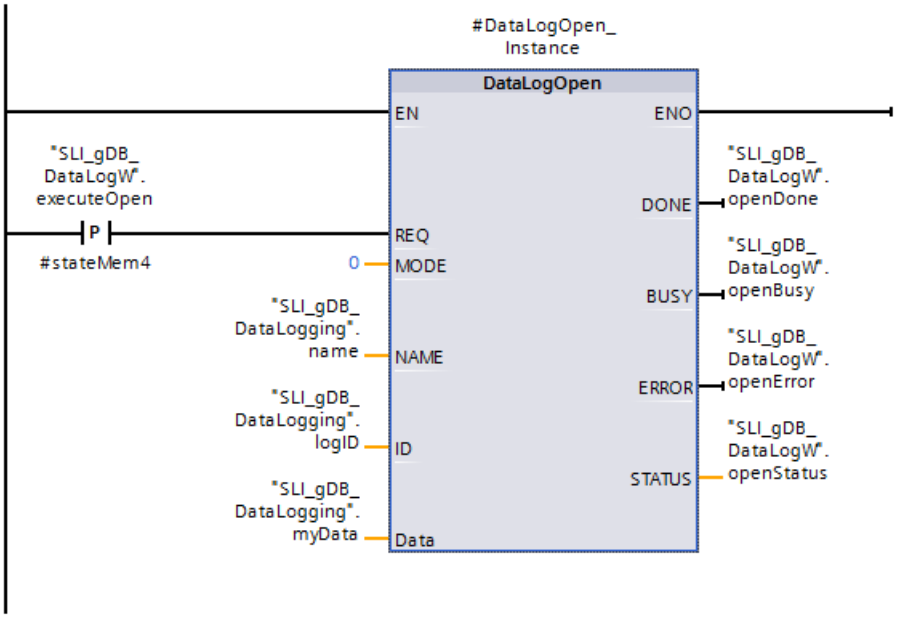


Network 6: Close the data log after the last data record has been written. The data log was created for 5 data records (see Network 1). This means that after five data records the value "0001" is output at the instruction DataLogWrite (Page 4029) at the parameter STATUS ("writeFullStatus"). (Last possible data record has been created at the end of the file. Creating another data record will overwrite an older data record.)

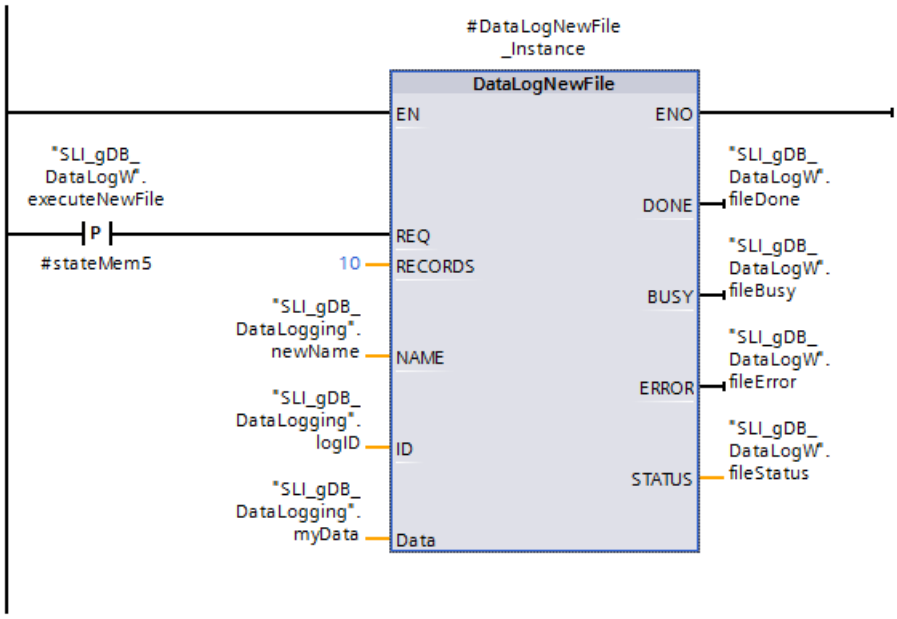
In this case, the input REQ is set and the instruction DataLogClose (Page 4031) is executed. When the data log is closed, no more data records are written.



Network 7: The data log must be opened again with the DataLogOpen (Page 4025) instruction in order to write a data record again at a later time. If another data record is written with DataLogWrite (Page 4029) after this, the oldest data record is overwritten.



Network 8: If you do not want to overwrite the older data records, you can use the DataLogNewFile (Page 4036) instruction to create a new data log with the same structure. To do this, enter the ID of the existing data log whose structure you want to copy at the ID parameter of the instruction. Once the DataLogNewFile (Page 4036) instruction has been executed, a new and unique ID value for the new data log is assigned.



Note that the call for DataLogNewFile (Page 4036) also extends over several cycles. Therefore, similar to the DataLogCreate (Page 4016) instruction, you should also query the DONE bit to prevent premature execution of DataLogWrite (Page 4029) (see Networks 1, 2 and 4).

Result

Opening of written data via the web server

The data logs created with the example program are displayed via the web server. To do this, open the Web server using an Internet browser and open the "\DataLogs" directory.

Contents of the CSV files

- "5" is set as the maximum number of data records when creating the data log with the DataLogCreate instruction. If this number is not exceeded, all written data records are included in the data log.

	A	B	C	D	E	F
1	Record	Date	UTC Time	Count	Temperature	Pressure
2	1	9/30/2010	20:26:56	1	9.86E+01	3.52E+01
3	2	9/30/2010	20:28:43	2	1.00E+02	3.73E+01
4	3	9/30/2010	20:29:03	3	9.99E+01	3.68E+01
5	4	9/30/2010	20:29:21	4	9.95E+01	3.64E+01
6	5	9/30/2010	20:30:19	5	9.92E+01	3.74E+01
7						

- If another data record is added to this, the oldest data record (Record 1) is overwritten.

	A	B	C	D	E	F
1	Record	Date	UTC Time	Count	Temperature	Pressure
2	6	9/30/2010	20:32:03	6	9.86E+01	3.58E+01
3	2	9/30/2010	20:28:43	2	1.00E+02	3.73E+01
4	3	9/30/2010	20:29:03	3	9.99E+01	3.68E+01
5	4	9/30/2010	20:29:21	4	9.95E+01	3.64E+01
6	5	9/30/2010	20:30:19	5	9.92E+01	3.74E+01
7						

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

DataLogClear: Empty data log (Page 4027)

DataLogDelete: Delete data log (Page 4033)

11.4.4.12 Data block functions (S7-1200, S7-1500)

CREATE_DB: Create data block (S7-1200, S7-1500)

Description

The "CREATE_DB" instruction is used to create a new data block in the load and/or work memory.

The instruction "CREATE_DB" does not change the checksum of the user program.

A data block that was only generated in the work memory has the following properties:

- After a memory reset or POWER OFF / POWER ON this block no longer exists.
- When loading or when there is a STOP-RUN transition, its content remains unchanged.

Number of the data block

The data block created is assigned a number from the range defined at the LOW_LIMIT (low limit) and UP_LIMIT (high limit) parameters. "CREATE_DB" assigns the smallest possible number from the specified range to the DB. You cannot assign the numbers of the DBs already contained in the user program.

To create a DB with a specific number, enter the same number for the high and low limit of the range to be specified. If a DB with the same number already exists in the work memory and/or load memory, or if the DB exists as a copied version, the instruction will be terminated and an error message will be generated at the RET_VAL parameter.

Start values of the data block

The SRCBLK parameter is used to define start values for the DB that is to be created. The SRCBLK parameter is a pointer to a DB or a DB area from which you apply the start values. The DB addressed at the SRCBLK parameter must have been generated with standard access ("Optimized block access" attribute disabled).

- If the area specified at the SRCBLK parameter is larger than the DB generated, the values up to the length of the DB generated will be applied as start values.
- If the area specified at the SRCBLK parameter is smaller than the DB generated, the remaining values will be filled with "0".

To ensure data consistency, you must not change this data area while "CREATE_DB" is being executed (which means as long as the BUSY parameter has the value TRUE).

Note

SRCBLK points to a TEMP tag

TEMP tags of standard blocks are located in the temporary local data area. A reference in SRCBLK to a TEMP tag therefore results in the value 80A3 (the data block at the SRCBLK parameter was not created with standard access) being output as RET_VAL.

Functional description

The "CREATE_DB" instruction works asynchronously. Processing takes place across several calls. You start the job by calling "CREATE_DB" with REQ = 1.

The output parameters RET_VAL and BUSY indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

Parameters

The following table shows the parameters of the "CREATE_DB" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Level-triggered control parameter "request to activate" REQ = 1: Request to generate the data block
LOW_LIMIT	Input	UINT	I, Q, M, D, L or constant	Low limit of the range for the assignment of a DB number. The smallest possible number is 60000.
UP_LIMIT	Input	UINT	I, Q, M, D, L or constant	High limit of the area used by "CREATE_DB" to assign a number to your DB (largest possible DB number: 60999)
COUNT	Input	UDINT	I, Q, M, D, L or constant	The count value specifies the number of bytes which you want to reserve for the DB generated. The number of bytes must be an even number. The maximum length is 65534 bytes.

<table>
<tr><th>Parameter</th><th>Declaration</th><th>Data type</th><th>Memory area</th><th>Description</th></tr>
<tr><td rowspan="8">ATTRIB</td><td rowspan="8">Input</td><td rowspan="8">BYTE</td><td rowspan="8">I, Q, M, D, L or constant</td><td>You use the first 4 bits of the byte at parameter ATTRIB to define the properties of the data block *:</td></tr>
<tr><td>• Bit 0 = 0: Attribute "Only store in load memory" is not set.
• Bit 0 = 1: Attribute "Only store in load memory" is set. With this setting, the DB takes up no space in the work memory and is not included in the program. The DB cannot be accessed with bit commands. When bit 0 = 1, the selection for bit 2 is irrelevant.
To ensure compatibility with STEP 7 V5.x, bits 0 and 3 must be considered together (see below).</td></tr>
<tr><td>• Bit 1 = 0: Attribute "Data block write-protected in the device" is not set.
• Bit 1 = 1: Attribute "Data block write-protected in the device" is set.</td></tr>
<tr><td>• Bit 2 = 0: DB is retentive (only for DBs generated in the load and in the work memory). The DB is regarded as retentive if at least one value has been set as retentive.
• Bit 2 = 1: DB is not retentive
Retentivity is not supported with DBs that are only stored in load memory or only in the work memory. If you call the "CREATE_DB" instruction with one of the two combinations "retentive and only load memory" or "retentive and only work memory" the DB to be generated will not be marked as retentive.</td></tr>
<tr><td>• Bit 3 = 0: Creation of the DB either in the load memory or in the work memory (selection using bit 0, see above)
• Bit 3 = 1: Creation of the DB both in the load memory and in the work memory (bit 0 irrelevant)
To ensure compatibility with STEP 7 V5.x, bits 0 and 3 must be used in combination. When bit 3 = 1, bit 0 is irrelevant.</td></tr>
<tr><td><table>
<tr><th>Bit 0</th><th>Bit 3</th><th>DB generation</th></tr>
<tr><td>0</td><td>0</td><td>In work memory only</td></tr>
<tr><td>1</td><td>0</td><td>In load memory only</td></tr>
<tr><td>Irrelevant</td><td>1</td><td>Work and load memory</td></tr>
</table></td></tr>
<tr><td>• Bit 4 = 0 - No start values specified (input values at the SRCBLK parameter will be ignored).
• Bit 4 = 1 - Specify start values (values correspond to the DB addressed by the SRCBLK parameter).</td></tr>
<tr><td>SRCBLK</td><td>Input</td><td>VARIANT</td><td>D</td><td>Pointer to the data block whose values will be used to initialize the data block to be generated.</td></tr>
<tr><td>RET_VAL</td><td>Return</td><td>INT</td><td>I, Q, M, D, L</td><td>Error information</td></tr>
<tr><td>BUSY</td><td>Output</td><td>BOOL</td><td>I, Q, M, D, L</td><td>BUSY = 1: The process is not yet complete.</td></tr>
<tr><td>DB_NUM</td><td>Output</td><td>DB_DYN (UINT)</td><td>I, Q, M, D, L</td><td>Number of the DB created.</td></tr>
<tr><td colspan="5">* The properties selected here correspond to the attributes in the properties of a data block.</td></tr>
</table>

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error
0081	The destination area is greater than the source area. The source area is written completely to the destination area. The remaining bytes of the destination area remain unchanged.
7000	First call with REQ = 0: No data transfer active; BUSY has the value "0".
7001	First call with REQ = 1: Data transfer triggered; BUSY has the value "1".
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value "1".
8081	The source area is larger than the destination area. The complete destination area is written. The remaining bytes of the source area are ignored.
8092	The "Create data block" function is currently unavailable because • The "Compress user memory" function is currently active. • The maximum number of blocks on your CPU has already been reached.
8093	No data block or a data block that is not in the work memory is specified for the SRCBLK parameter.
8094	An invalid value was specified at parameter ATTRIB.
80A1	DB number error: • The number is "0" • The number exceeds the CPU-specific high limit for DB numbers • Low limit > high limit
80A2	DB length error: • The length is "0" • The length is an odd number • The length is greater than permitted by the CPU
80A3	The data block at the SRCBLK parameter was not created with standard access.
80B1	There is no DB number free.
80B2	Insufficient free memory available.
80B4	The memory card is write-protected.
80BB	Not enough load memory.
80C3	The maximum number of simultaneously active "CREATE_DB" instructions has already been reached.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You will find the example here: Program example for CREATE functions (Page 4065).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

READ_DBL: Read from data block in the load memory (S7-1200, S7-1500)

Description

With the instruction, you copy a DB or an area of a DB in load memory (Micro Memory Card) to the data area of a destination DB. The destination data block must be runtime-relevant, which means it must not be created with the Only store in load memory attribute. The content of the load memory is not changed during the copy process.

To ensure data consistency, you must not change the destination area while "READ_DBL" is being executed (i.e., as long as the BUSY parameter has the value TRUE).

The parameters SRCBLK and DSTBLK (source and destination blocks) must both be VARIANT data type. The following restrictions hereby apply:

- Valid for standard memory areas: If you want to interconnect a parameter of VARIANT data type (source or destination area) with a tag of BOOL data type or an ARRAY of BOOL, you have the following options:
  - You can address it symbolically
    Example: Parameter SRCBLK: "Data_block".myArray
  - You can address it absolutely with help of the ANY pointer. However, you must note that the specified length of the area must be divisible by 8, or the instruction will otherwise not be executed.
    Example: Parameter SRCBLK: P#DB123.DBX456.0 BOOL 1000
- For a VARIANT pointer of type STRING, the length must be equal to 1.
- The source and destination blocks must have been created with the same block access, i.e. both must use either the "Optimized" or "Standard" access type.
- When the source and target DB blocks have optimized access, you must not use the STRUCT data type.

Note

"READ_DBL" is processed asynchronously and is therefore, it is not suitable for frequent (or cyclic) reading of tags in the load memory.

Once started, a job is always completed. If the maximum number of simultaneously active "READ_DBL" instructions is reached and you call "READ_DBL" once again at this time in a priority class having higher priority, error code W#16#80C3 will be returned. Consequently it does not make sense to restart the high-priority job right away.

Functional description

The "READ_DBL" instruction works asynchronously. Processing takes place across several calls. You start the job by calling "READ_DBL" with REQ = 1.

The output parameters RET_VAL and BUSY indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

Parameters

The following table shows the parameters of the "READ_DBL" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = 1: Read request
SRCBLK	Input	VARIANT	D	Pointer to data block in the load memory that is to be read from
RET_VAL	Return	INT	I, Q, M, D, L	Error information
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The reading process is not yet complete.
DSTBLK	Output	VARIANT	D	Pointer to the data block in the work memory that is written to

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error
0081	The destination area is greater than the source area. The source area is written completely to the destination area; the remaining bytes of the destination area will not be changed.
7000	First call with REQ = 0: No data transfer active; BUSY has the value "0".
7001	First call with REQ=1: Data transfer triggered; BUSY has the value "1".
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value "1".
8x51	Note: This error code is only for S7-1200 CPUs. Data type error in the data block.
8081	The source area is larger than the destination area. The complete destination area is written. The remaining bytes of the source area are ignored.
8082	• Destination DB type different from source DB type (optimized/standard access). • The destination and source DB were both created with optimized access. You have, however, used structures with the destination and source DB that lead to a different arrangement of the data when compiled. (Note: Use UDTs instead.)
8093	Note: This error code is only for S7-1500 CPUs. No data block or a data block that is not in the work memory is specified for the DSTBLK parameter.
80B1	Note: This error code is only for S7-1500 CPUs. At parameter SRCBLK, only data blocks that are located in the load memory are permitted.
8xB1	Note: This error code is only for S7-1200 CPUs. No data block is specified for the SRCBLK parameter, or the data block specified there is not a load memory object.
80B4	DSTBLK points to a DB with F-attribute (that must not be written)
8xC0	Note: This error code is only for S7-1200 CPUs. The destination DB is currently being processed by another instruction or a communication function.

Error code* (W#16#...)	Description
80C3	The maximum number of simultaneously active "READ_DBL" instructions has already been reached.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

In the following example, you read out the contents of a DB in the load memory to a DB in the work memory.

To store the data for the interconnection, create three tags in a global data block.

SLI_gDB_READ_DBL

	Name	Data type	Start value
1	Static		
2	execute	Bool	false
3	busy	Bool	false
4	returnValue	Int	0

To store the send data, create two global data blocks with three tags each of the INT data type.

- For receive data in the load memory:

SourceDB_LoadMemory

	Name	Data type	Offset	Start value
1	Static			
2	MySourceValue1	Int	0.0	111
3	MySourceValue2	Int	2.0	222
4	MySourceValue3	Int	4.0	333

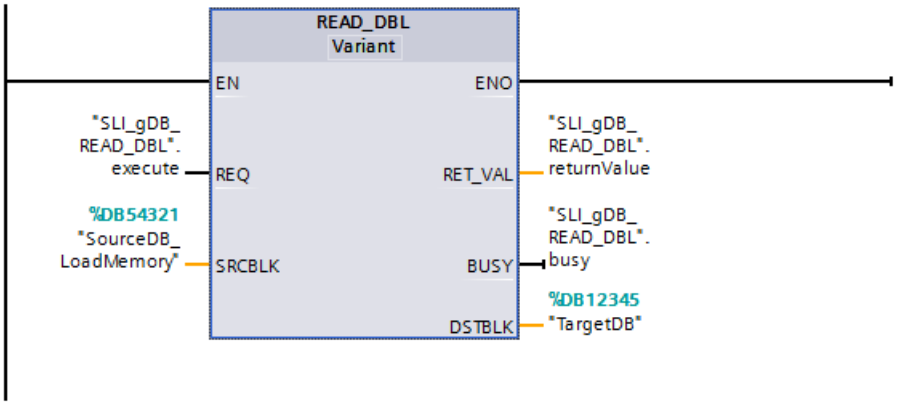
- For receive data in the work memory:

TargetDB

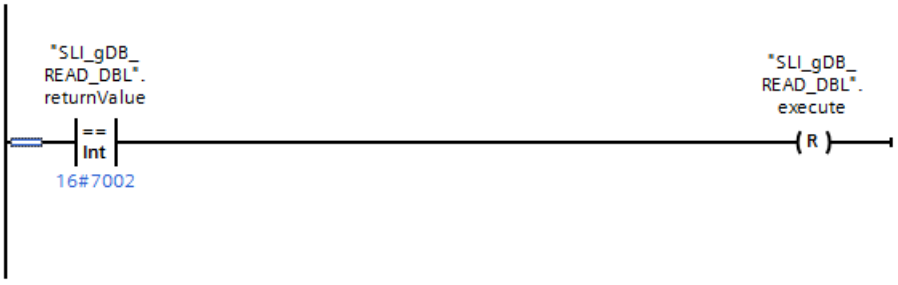
	Name	Data type	Offset	Start value
1	Static			
2	MyTargetValue1	Int	0.0	0
3	MyTargetValue2	Int	2.0	0
4	MyTargetValue3	Int	4.0	0

Create the following interconnections in an FB "SLI_FB_READ_DBL".

Network 1: Interconnect the parameters of the "READ_DBL" instruction as follows.



Network 2: You can stop the processing of READ_DBL as follows.



If the normally open contact ("execute") supplies the signal state "TRUE", the "READ_DBL" instruction is executed. The DB is stored in the load memory with the input parameter SRCBLK ("SourceDB_LoadMemory").
The DB is stored in the work memory with the output parameter DSTBLK ("TargetDB"). The instruction "READ_DBL" calls the data to be transferred in the load memory (SRCBLK) and reads the data in the work memory (DSTBLK).

The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

GET_ERR_ID: Get error ID locally (Page 3409)

WRIT_DBL: Write to data block in the load memory (S7-1200, S7-1500)

Description

The instruction "WRIT_DBL" is used to transfer the contents of a DB or a DB area from the work memory to a DB or a DB area in the load memory (Micro Memory Card). The source DB must be runtime-relevant, which means it must not be created with the Only store in load memory attribute.

Note

Start value behavior during "Upload from device"

The start values that you changed with the "WRIT_DBL" instruction are lost when you perform the "Upload from device" action.

To ensure data consistency, it is not permitted to change the source range while "WRIT_DBL" is being executed (i.e., as long as the BUSY parameter has the value TRUE).

The parameters SRCBLK and DSTBLK (source and destination blocks) must both be VARIANT data type. The following restrictions hereby apply:

- For a VARIANT pointer of type BOOL, the length must be divisible by 8.
- For a VARIANT pointer of type STRING, the length must be equal to 1.
- The source and destination blocks must have been created with the same block access, i.e. both must use "Optimized block access" or optimized access must be disabled for both.
- When the source and target DB blocks have optimized access, you must not use the STRUCT data type.

The "WRIT_DBL" instruction does not change the checksum of the user program if you write a DB that was created using an instruction. However, when a loaded DB is written, the first entry in this DB changes the checksum of the user program.

Note

"WRIT_DBL" is not suitable for frequent (or cyclical) writing of tags in the load memory. This is because the memory card technology limits the number of write accesses that can be made to a memory card.

Functional description

The "WRIT_DBL" instruction works asynchronously. Processing takes place across several calls. You start the job by calling "WRIT_DBL" with REQ=1.

The output parameters RET_VAL and BUSY indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571)

Parameters

The following table shows the parameters of the "WRIT_DBL" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = 1: Write request
SRCBLK	Input	VARIANT	D	Pointer to the DB in the work memory that is to be read from
RET_VAL	Return	INT	I, Q, M, D, L	Error information
BUSY	Output	BOOL	I, Q, M, D, L	BUSY = 1: The writing process is not yet complete.
DSTBLK	Output	VARIANT	D	Pointer to the data block in the load memory that is written to

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Description
0000	No error
0081	The destination area is greater than the source area. The source area is written completely to the destination area; the remaining bytes of the destination area will not be changed.
7000	First call with REQ = 0: No data transfer active; BUSY has the value "0".
7001	First call with REQ = 1: Data transfer triggered; BUSY has the value "1".
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value "1".
8051	Note: This error code is only for S7-1200 CPUs. Type error data block.
8251	Note: This error code is only for S7-1200 CPUs. Data type error in the source DB
8551	Note: This error code is only for S7-1200 CPUs. Data type error in the target DB
8081	The source area is larger than the destination area. The complete destination area is written. The remaining bytes of the source area are ignored.
8082	Note: This error code is only for S7-1500 CPUs. • Destination DB type different from source DB type (optimized/non-optimized access). • The destination and source DB were both created with optimized access. You have, however, used structures with the destination and source DB that lead to a different arrangement of the data when compiled. (Note: Use UDTs instead.)
8093	Note: This error code is only for S7-1500 CPUs. No data block or a data block that is not in the work memory is specified for the SRCBLK parameter.
80B1	Note: This error code is only for S7-1500 CPUs. At parameter DSTBLK, only data blocks that are located in the load memory are permitted.

Error code* (W#16#...)	Description
82B1	Note: This error code is only for S7-1200 CPUs. No data block is specified for the SRCBLK parameter.
85B1	Note: This error code is only for S7-1200 CPUs. For parameter DSTBLK, no data block is specified or the data block specified there is not in the load memory.
80B4	• The memory card is write-protected. • DB with F-attribute must not be read.
80BB	Insufficient load memory available.
82C0	Note: This error code is only for S7-1200 CPUs. The source DB is currently being processed by another instruction or a communication instruction.
85C0	Note: This error code is only for S7-1200 CPUs. The destination DB is currently being processed by another instruction or a communication instruction.
80C3	The maximum number of simultaneously active "WRIT_DBL" instructions has already been reached.
General error codes	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

In the following example, you transfer the contents of a DB in the work memory to a DB in the load memory. To verify successful transfer to the DB, read the contents in another DB in the work memory.

To store the data for the interconnection, create five tags in a global data block.

SLI_gDB_WRIT_DBL

	Name	Data type	Start value
1	Static		
2	execute	Bool	false
3	returnValue	Int	0
4	busy	Bool	false
5	readRetVal	Int	0
6	readBusy	Bool	false

To store the data to be transmitted, create three global data blocks with one "myData" tag each (Array of INT data type):

- A DB for send data in the work memory (SLI_srcDB_WRIT_DBL).

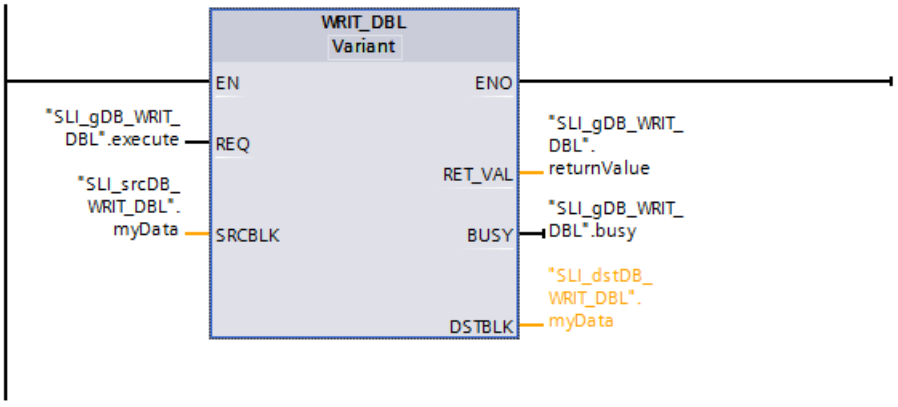
SLI_srcDB_WRIT_DBL

	Name	Data type	Start value
1	Static		
2	myData	Array[0..3] of Int	
3	myData[0]	Int	1
4	myData[1]	Int	2
5	myData[2]	Int	3
6	myData[3]	Int	4

- A DB for receive data in the load memory (SLI_dstDB_WRIT_DBL).
- A DB for receive data in the work memory (SLI_finDB_WRIT_DBL).

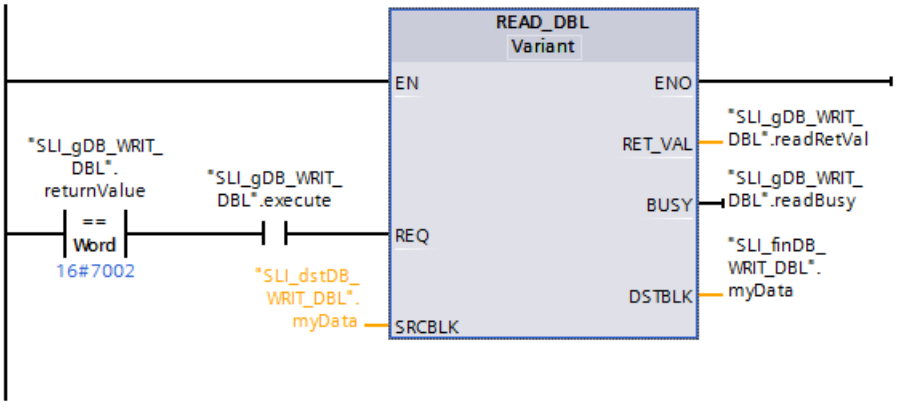
Network 1:

Interconnect the parameters of the "WRIT_DBL" instruction as follows.



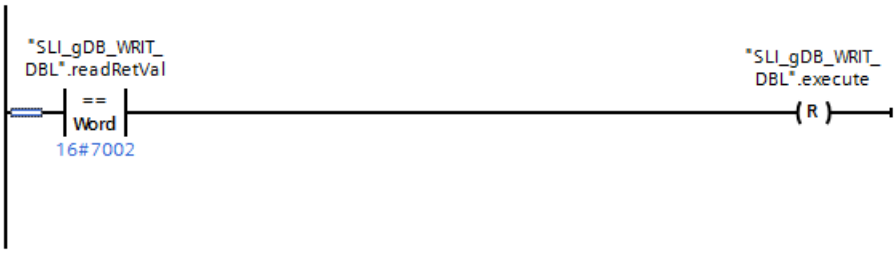
Network 2:

Interconnect the parameters of the "READ_DBL" instruction as follows.



Network 3:

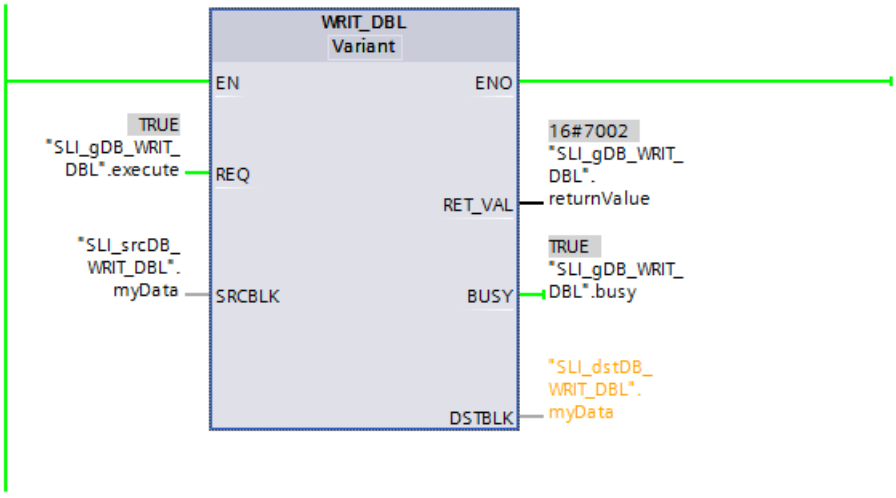
Reset the tag "execute" as follows:



"WRIT_DBL" instruction:

If the normally open contact ("execute") supplies the signal state "TRUE", the "WRIT_DBL" instruction is executed. The DB is stored in the work memory with the input parameter SRCBLK ("SLI_srcDB_WRIT_DBL.myData"). The DB is stored in the load memory with the output parameter DSTBLK ("SLI_dstDB_WRIT_DBL.myData").

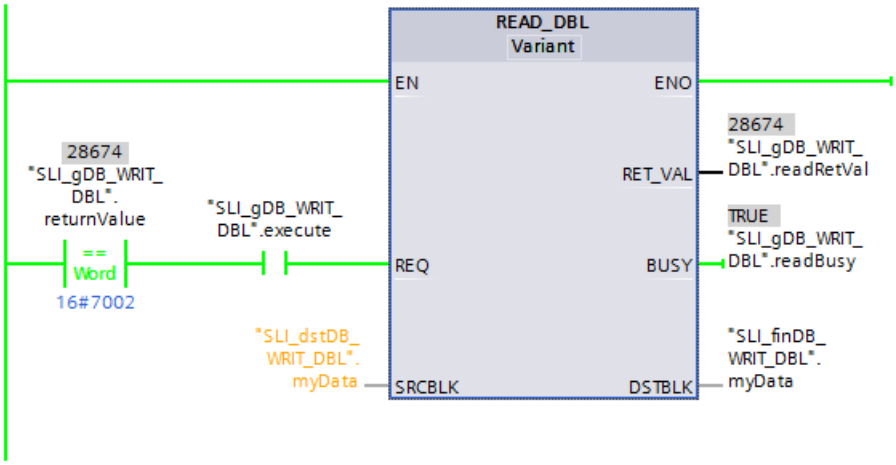
The "WRIT_DBL" instruction calls the data to be transferred in the work memory (SRCBLK) and writes the data to the required location in the load memory (DSTBLK). The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.



"READ_DBL" instruction:

When processing of WRIT_DBL took place without errors and the normally open contact ("execute") supplies the signal state "TRUE", the "READ_DBL" instruction is executed. The DB is stored in the load memory with the input parameter SRCBLK ("SLI_dstDB_WRIT_DBL.myData"). The DB for the receive data is stored with the output parameter DSTBLK ("SLI_finDB_WRIT_DBL.myData").

The "READ_DBL" instruction calls the data to be transferred in the load memory (SRCBLK) and reads the data at the required location in the work memory (DSTBLK). The output parameter RET_VAL ("readRetVal") indicates that processing took place without errors.



The DB for the receive data in the work memory (SLI_finDB_WRIT_DBL) displays the transmitted data.

SLI_finDB_WRIT_DBL

	Name	Data type	Start value	Monitor value
1	Static			
2	myData	Array[0..3] o...		
3	myData[0]	Int	0	1
4	myData[1]	Int	0	2
5	myData[2]	Int	0	3
6	myData[3]	Int	0	4

The tag "execute" is reset after the successful processing of READ_DBL (see Network 3). The processing of READ_DBL and WRIT_DBL is stopped.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

ATTR_DB: Read data block attribute (S7-1200, S7-1500)

Description

You use the instruction "ATTR_DB" to obtain information about a data block (DB) located in the work memory of the CPU. The instruction determines the attributes set at the ATTRIB parameter for the DB selected.

The length cannot be read out for data blocks with optimized access and data blocks that are only in load memory. In these cases, the DB_LENGH parameter has the value "0".

Do not apply ATTR_DB to data blocks with optimized access and activated memory reserve.

Do not read out the data blocks for Motion Control with the "ATTR_DB" instruction. The error code 80B2 is output for this.

Parameters

The following table shows the parameters of the "ATTR_DB" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ = 1: Request to read block attributes
DB_NUM-BER	Input	DB_ANY	I, Q, M, D, L or constant	Number of the DB to be tested
RET_VAL	Output	INT	I, Q, M, D, L	Error information
DB_LENGTH	Output	UDINT	I, Q, M, D, L	• Number of data bytes which the selected DB contains. • "0" for data blocks with optimized access and data blocks that are only in load memory.
ATTRIB	Output	BYTE	I, Q, M, D, L	DB properties:
				• Bit 0*= 0: Attribute "Only store in load memory" is not set. • Bit 0*= 1: Attribute "Only store in load memory" is set.
				• Bit 1 = 0: Attribute "Data block write-protected in the device" is not set. • Bit 1 = 1: Attribute "Data block write-protected in the device" is set.
				If bit 0 = 1, then bit 2 is irrelevant and gets the value 1. • Bit 2 = 0: Retentive - The DB is regarded as retentive if at least one value has been set as retentive. • Bit 2 = 1: Not retentive - The complete DB is not retentive.
				• Bit 3*= 0: The DB is either in the load memory (bit 0 = 1) or in the work memory (bit 0 = 0). • Bit 3*= 1: The DB is generated in both the load and the work memory
* The relationship between bit 0 and bit 3 is explained in the parameters of the instruction "CREATE_DB: Create data block (Page 4045)".				

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
80A1	Error in input parameter DB_NUMBER: the actual parameter selected • Is "0" • Is greater than the maximum permitted DB number for the CPU used.
80B1	The DB with the specified number does not exist on the CPU.

Error code* (W#16#...)	Explanation
80B2	Data blocks of Motion Control technology objects cannot be read with the "ATTR_DB" instruction.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

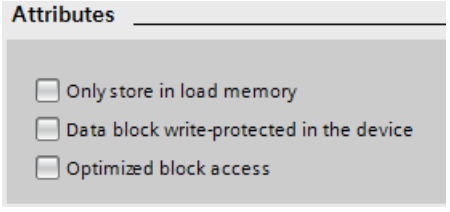
In the following example, you read the status of a data block.

You create a DB with a "myData" tag (Array[0..3] of INT data type).

SLI_srcDB_ATTR_DB

	Name	Data type	Offset	Start value
1	Static			
2	myData	Array[0..3] of Int	0.0	
3	myData[0]	Int	0.0	1
4	myData[1]	Int	2.0	2
5	myData[2]	Int	4.0	3
6	myData[3]	Int	6.0	4

For standard access, disable optimized data access in the DB properties.



To store the data for the interconnection, create four tags in a global data block.

SLI_gDB_ATTR_DB

	Name	Data type	Start value
1	Static		
2	startAttr	Bool	false
3	attrRetVal	Int	0
4	readLenByte	UDInt	0
5	readAttrBool	Array[0..7] of Bool	

Interconnect parameters: "SLI_itemiseByte_ATTR_DB" FC

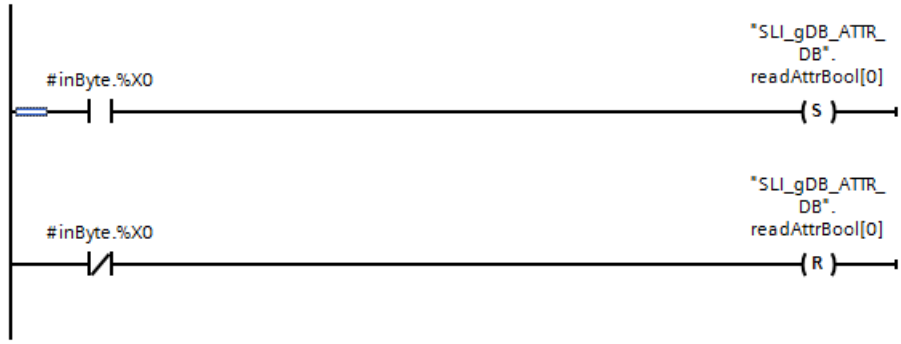
To better evaluate the value of the output parameter ATTRIB, create the FC "SLI_itemiseByte_ATTR_DB".

Create the following local tag in it.

SLI_itemiseByte_ATTR_DB

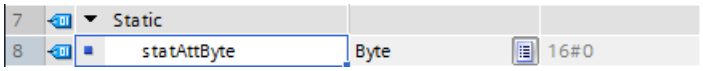
	Name	Data type
1	Input	
2	inByte	Byte

Network 1: Create interconnections for all bits (0..7) of the local tag according to the example of network 1.

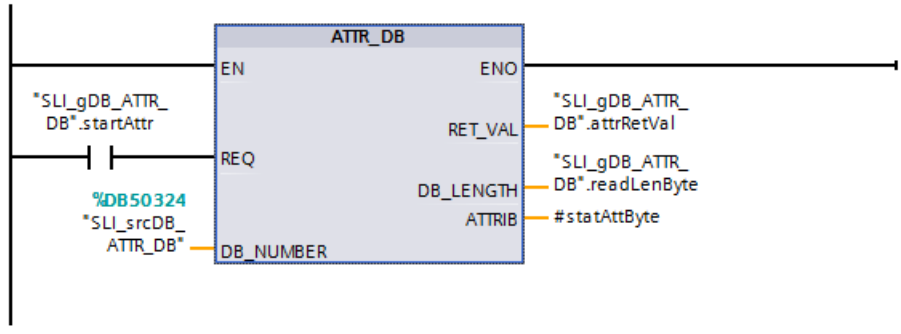


Interconnect parameters: "SLI_FB_ATTR_DB" FB

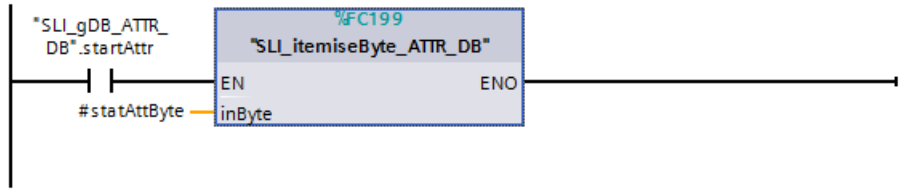
Create an FB "SLI_FB_ATTR_DB" for the call of ATTR_DB. Create the following local tag in it.



Network 1: Interconnect the parameters of the "ATTR_DB" instruction as follows.



Network 2: Interconnect the FC "SLI_itemiseByte_ATTR_DB" as follows.



Result of FB "SLI_FB_ATTR_DB"

If the normally open contact ("startAttr") supplies the signal state "TRUE", the "ATTR_DB" instruction is executed. The number of the DB to be evaluated is stored with the input parameter DB_NUMBER ("SLI_srcDB_ATTR_DB"). The "ATTR_DB" instruction calls the DB based on the number and outputs its status at the output parameter ATTRIB ("#statAttrByte"). The length of the read data (4 INT) of the data block in BYTE (value "8") is output at the output parameter DB_LENGTH ("readLenByte").

The hexadecimal value of the output parameter ATTRIB is converted into a bit string using FC "SLI_itemiseByte_ATTR_DB". The bit sequence "00110000" (Bit 0 to 7) means:

- The DB exists not only in the load memory.
- The DB is not write-protected.
- The DB is not retentive.
- The DB is generated in both the load and the work memory.

The output parameter RET_VAL ("attrRetVal") indicates that processing took place without errors.

SLI_gDB_ATTR_DB

	Name	Data type	Start value	Monitor value
1	Static			
2	startAttr	Bool	false	TRUE
3	attrRetVal	Int	0	0
4	readLenByte	UDInt	0	8
5	readAttrBool	Array[0..7] of Bool		
6	readAttrBool[0]	Bool	false	FALSE
7	readAttrBool[1]	Bool	false	FALSE
8	readAttrBool[2]	Bool	false	TRUE
9	readAttrBool[3]	Bool	false	TRUE
10	readAttrBool[4]	Bool	false	FALSE
11	readAttrBool[5]	Bool	false	FALSE
12	readAttrBool[6]	Bool	false	FALSE
13	readAttrBool[7]	Bool	false	FALSE

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

DELETE_DB: Delete data block (S7-1200, S7-1500)

Description

You use the instruction "DELETE_DB" to delete a data block (DB) that was created from the user program by calling the instruction "CREATE_DB (Page 4045)".

If the data block was not created with "CREATE_DB", the error code W#16#80B5 is output at the RET_VAL parameter.

The selected data block is not deleted immediately, but rather at the cycle control point after execution of the cycle OB.

Functional description

The "DELETE_DB" instruction works asynchronously. Processing takes place across several calls. You start the interrupt transfer by calling the instruction with REQ = 1.

Output parameter BUSY and bytes 2 and 3 of output parameter RET_VAL show the status of the job.

When the output parameter BUSY has the value FALSE, the deletion of the data block is completed.

Parameters

The following table shows the parameters of the "DELETE_DB" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	REQ =1: Request to delete the DB with the number in the DB_NUMBER parameter
DB_NUMBER	Input	UINT	I, Q, M, D, L or constant	Number of the DB to be deleted
RET_VAL	Output	INT	I, Q, M, D, L	Error information (see "RET_VAL parameter")
BUSY	Output	BOOL	I, Q, M, D, L	BUSY =1: The process is not yet complete.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
7000	First call with REQ = 0: No data transfer active; BUSY has the value "0".
7001	First call with REQ = 1: Data transfer triggered; BUSY has the value "1".
7002	Intermediate call (REQ irrelevant): Data transfer already active; BUSY has the value "1".
80A1	Error in input parameter DB_NUMBER: • The value at the parameter is "0". • The value at the parameter is greater than the maximum permitted DB number for the CPU used.
80B1	The DB with the specified number does not exist on the CPU.
80B4	The DB cannot be deleted because the memory card of the CPU is write-protected.
80B5	The DB was not created using "CREATE_DB".
80BB	Not enough load memory.
80C3	The "Delete a DB" function cannot be executed at this time due to a temporary resource bottleneck.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for CREATE functions (Page 4065).

You can find additional information and the program code for the above named example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

Program example for CREATE functions (S7-1200, S7-1500)

Introduction

In the following example, you create a DB in the load memory, read its contents, and then delete the DB.

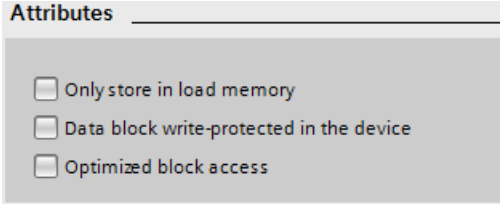
Requirement

Create a global DB ("SLI_srcDB_CREATE_DB") to serve as a template for the new DB.

SLI_srcDB_CREATE_DB

	Name	Data type	Offset	Start value
1	Static			
2	myData	Array[0..3] of Int	0.0	
3	myData[0]	Int	0.0	1
4	myData[1]	Int	2.0	2
5	myData[2]	Int	4.0	3
6	myData[3]	Int	6.0	4
7	readData	Array[0..3] of Int	8.0	
8	readData[0]	Int	0.0	0
9	readData[1]	Int	2.0	0
10	readData[2]	Int	4.0	0
11	readData[3]	Int	6.0	0

The properties of the DB are as follows.



To store the data for the interconnection, create 14 tags in a global data block ("SLI_gDB_CREATE_DB").

SLI_gDB_CREATE_DB			
	Name	Data type	Start value
1	Static		
2	startCreate	Bool	false
3	lowLim	UInt	60003
4	hiLim	UInt	60003
5	setLenByte	UDInt	8
6	setAttrBool	Array[0..7] of Bool	
7	setAttrBool[0]	Bool	1
8	setAttrBool[1]	Bool	0
9	setAttrBool[2]	Bool	1
10	setAttrBool[3]	Bool	0
11	setAttrBool[4]	Bool	1
12	setAttrBool[5]	Bool	false
13	setAttrBool[6]	Bool	false
14	setAttrBool[7]	Bool	false
15	createRetVal	Int	0
16	createBusy	Bool	false
17	memNbrNewDB	UInt	0
18	startRead	Bool	false
19	rdblRetVal	Int	0
20	rdblBusy	Bool	false
21	startDelete	Bool	false
22	delRetVal	Int	0
23	delBusy	Bool	false

To store the data, also create the following local tags in the FB "SLI_FB_CREATE_DB".

7	Static			
8	statAttByte	Byte	16#0	Non-ret...
9	nbrNewDB	UInt	0	Non-retain
10	Temp			
11	tempCreate	Bool		
12	tempRead	Bool		
13	tempDelete	Bool		

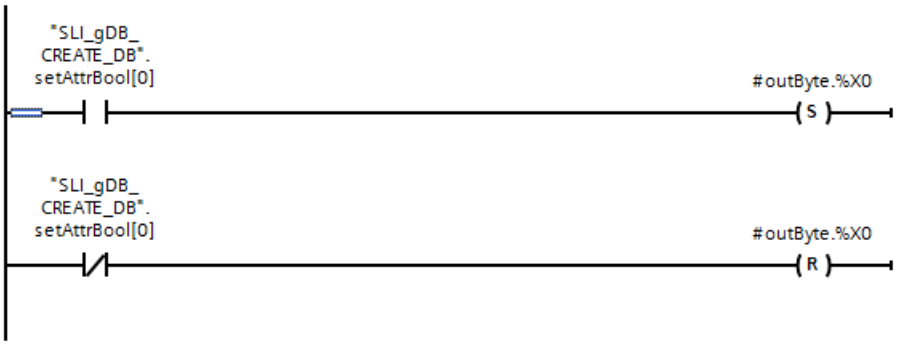
Interconnect parameters: "SLI_itemiseByte_CRDfunctions" FC

To convert the values of the "setAttrBool" tag into the "#statAttByte" tag, create the FC "SLI_itemiseByte_CRDfunctions".

Create the following local tag in it.

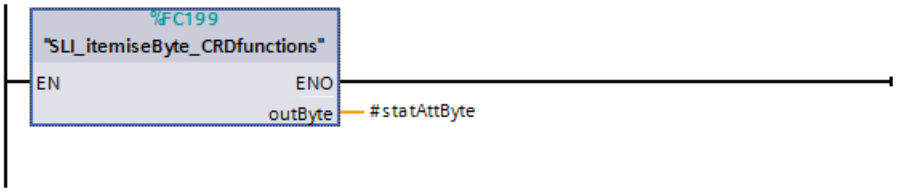
3	Output	
4	outByte	Byte

Network 1: Create interconnections for all bits (0..7) of the local tag according to the example of network 1.

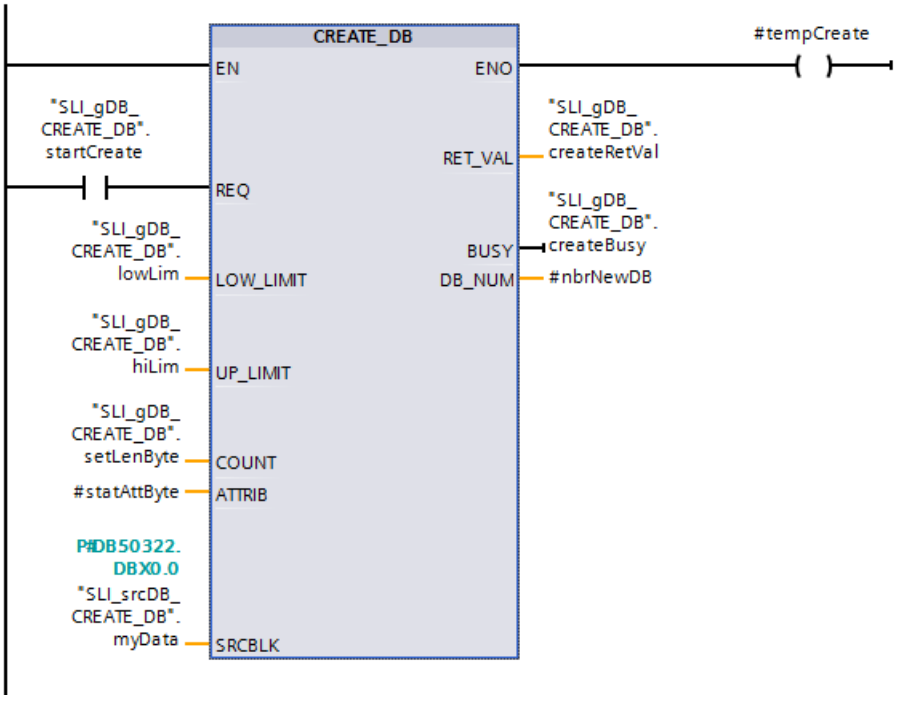


Interconnect parameters: "SLI_FB_CREATE_DB" FB

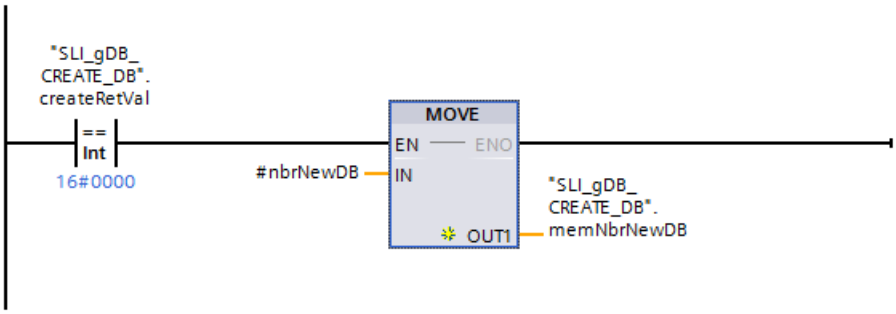
Network 1: Interconnect the parameters of the "SLI_itemiseByte_CRDfunctions" FC as follows.



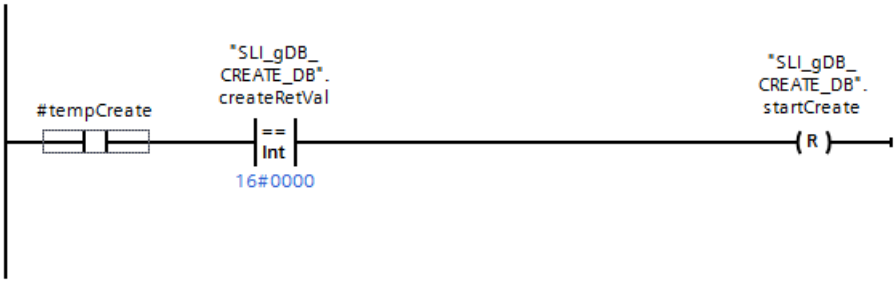
Network 2: Interconnect the parameters of the "CREATE_DB" instruction as follows.



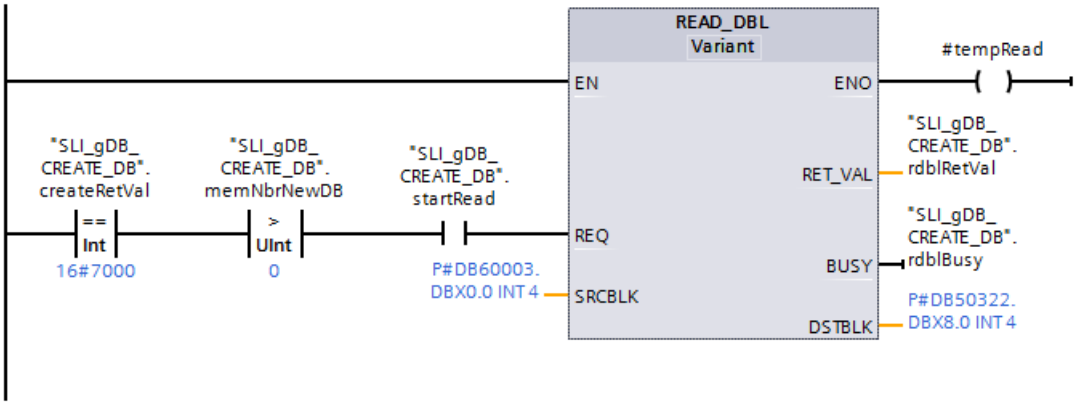
Network 3: To store the number of the newly created DB, create the following interconnections.



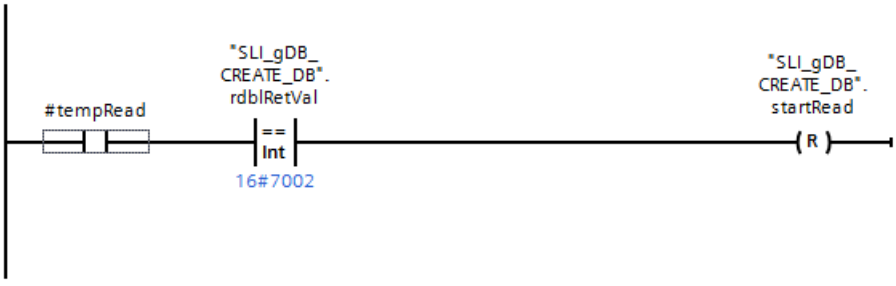
Network 4: To end the execution of CREATE_DB once again, create the following interconnections.



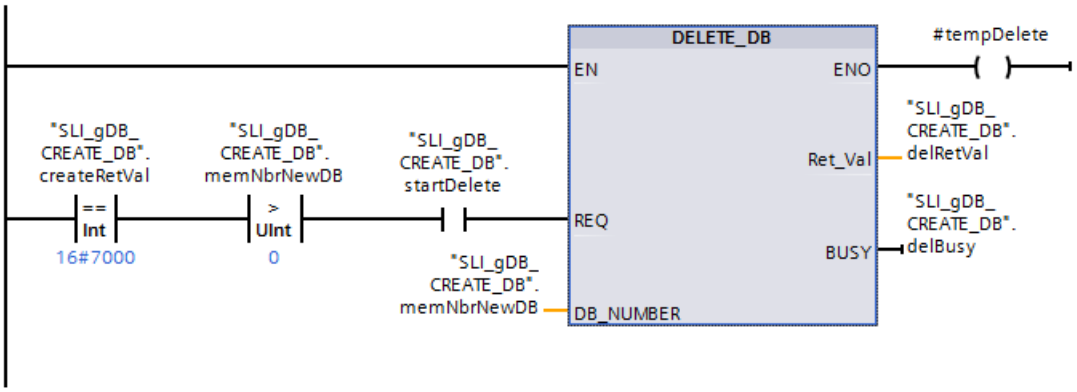
Network 5: Interconnect the parameters of the "READ_DBL" instruction as follows.



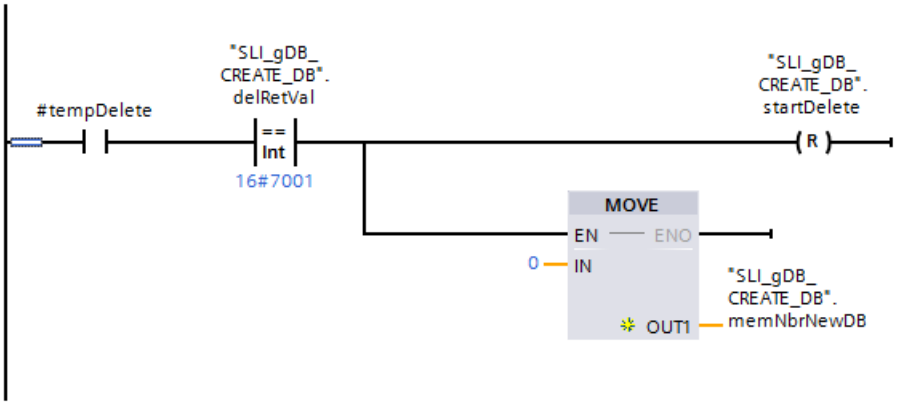
Network 6: To end the execution of READ_DBL once again, create the following interconnections.



Network 7: Interconnect the parameters of the "DELETE_DB" instruction as follows.



Network 8: To end the execution of DELETE_DB once again, create the following interconnections.



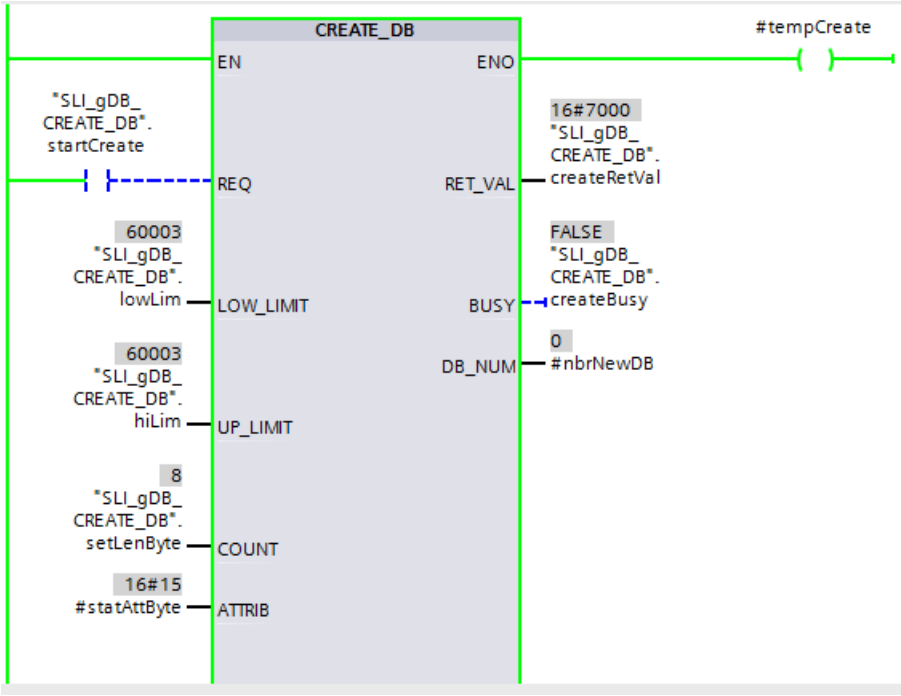
Result of network 1

With an AT construct, the properties ("setAttrBool") stored in the Array of BOOL data type(#statAttBool) are converted into BYTE (#statAttByte) and stored.

Result of network 2 - "CREATE_DB" instruction

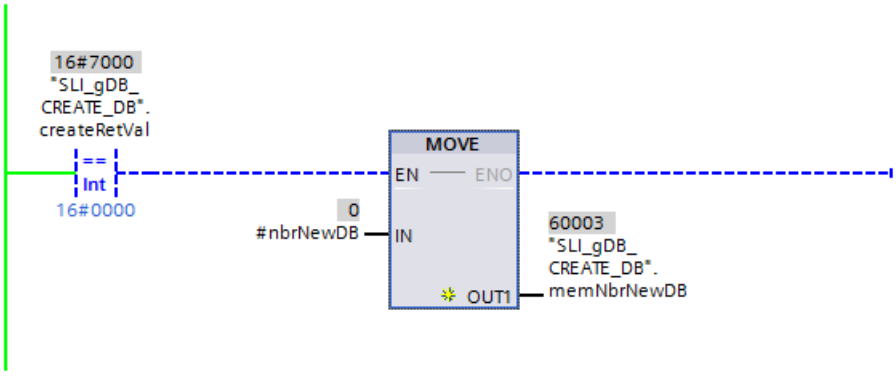
If the normally open contact ("startCreate") supplies the signal state "TRUE", the "CREATE_DB" instruction is executed. According to the input parameter COUNT ("setLenByte"), a new DB with a length of 8 bytes is created in the process. The number of the DB that is to be created is specified with the input parameter LOW_LIMIT ("lowLim") or UP_LIMIT ("hiLim"). The properties (in BYTE) with which the DB is to be created are stored at the input parameter ATTR ("#statAttByte"). The DB is to be created only in the load memory and filled with defined contents. A DB is stored at the input parameter SRCBLK ("SLI_srcDB_CREATE_DB.myData") to serve as template for the contents of the new DB.

The output parameter RET_VAL ("CreateRetVal") indicates that processing took place without errors. The instruction "CREATE_DB" remains inactive after its execution ("createRetVal" has the value 16#7000"). The number of the new DB is briefly output at the end of the execution at the output parameter DB_NUM ("#nbrNewDB").



Result of network 3

When processing of CREATE_DB took place without errors, "createRetVal" has the value "16#0000". As a result, the number ("#nbrNewDB") of the DB is stored in the "memNbrNewDB" tag.



Result of network 4

When CREATE_DB at the ENO parameter ("#tempCreate") supplies the signal state "TRUE", the interconnection of network 4 is active. In addition, when processing of CREATE_DB took place without errors, the reset ("startCreate") is set to "TRUE" and the execution of CREATE_DB is ended.

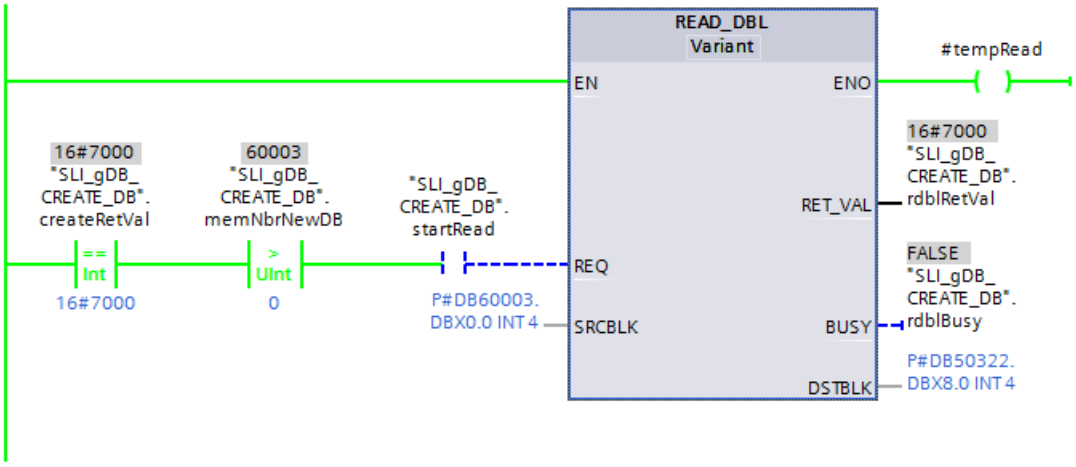
SLI_gDB_CREATE_DB

	Name	Data type	Start value	Monitor value
1	Static			
2	startCreate	Bool	false	FALSE
3	lowLim	UInt	60003	60003
4	hiLim	UInt	60003	60003
5	setLenByte	UDInt	8	8
6	setAttrBool	Array[0..7] of Bool		
7	createRetVal	Int	0	28672
8	createBusy	Bool	false	FALSE
9	memNbrNewDB	UInt	0	60003
10	startRead	Bool	false	FALSE
11	rdblRetVal	Int	0	28672
12	rdblBusy	Bool	false	FALSE
13	startDelete	Bool	false	FALSE
14	delRetVal	Int	0	28672
15	delBusy	Bool	false	FALSE

Result of network 5 - "READ_DBL" instruction

The following preconditions exist for executing READ_DBL:

- CREATE_DB is inactive ("createRetVal" has the value "16#7000").
- CREATE_DB has created a new DB whose number was stored ("memNbrNewDB" has a value > 0).



If the normally open contact ("startRead") supplies the signal state "TRUE", the "READ_DBL" instruction is executed. The newly created DB is referenced in the load memory with the input parameter SCRBLK. The DB is referenced in the work memory ("SLI_srcDB_CREATE_DB.readData") with the output parameter DSTBLK. The "READ_DBL"

instruction calls the data to be transferred in the load memory (SCRBLK) and reads the data at the required location in the work memory (DSTBLK).

The output parameter RET_VAL ("rdblRetVal") indicates that processing took place without errors. The transferred data is displayed for the DB in the work memory ("SLI_srcDB_CREATE_DB.readData").

SLI_srcDB_CREATE_DB

	Name	Data type	Offset	Start value	Monitor value
1	Static				
2	myData	Array[0..3] of Int	0.0		
3	myData[0]	Int	0.0	1	1
4	myData[1]	Int	2.0	2	2
5	myData[2]	Int	4.0	3	3
6	myData[3]	Int	6.0	4	4
7	readData	Array[0..3] of Int	8.0		
8	readData[0]	Int	0.0	0	1
9	readData[1]	Int	2.0	0	2
10	readData[2]	Int	4.0	0	3
11	readData[3]	Int	6.0	0	4

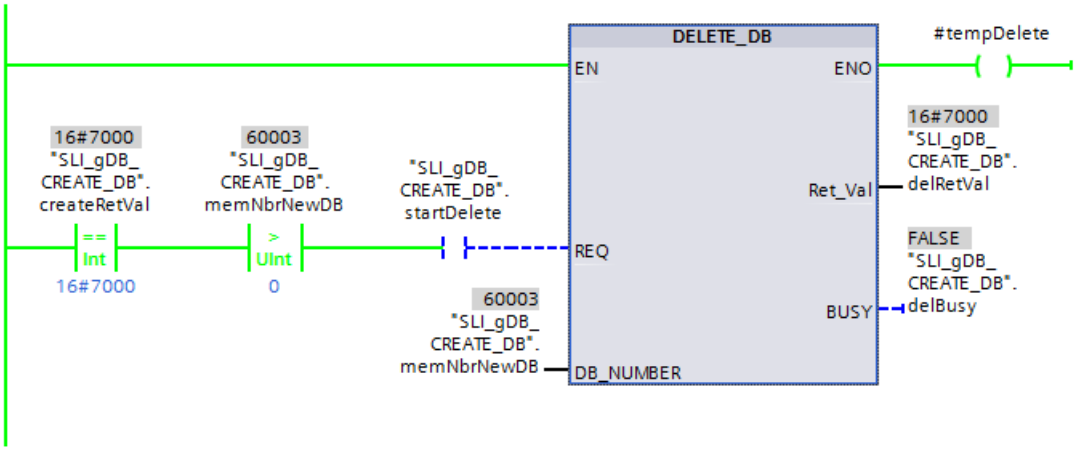
Result of network 6

When READ_DBL at the ENO parameter ("#tempRead") supplies the signal state "TRUE", the interconnection of network 6 is active. In addition, when processing of READ_DBL is complete, the reset ("startRead") is set to "TRUE" and the execution of READ_DBL is ended.

Result of network 7 - "DELETE_DB" instruction

The following preconditions exist for executing DELETE_DB:

- CREATE_DB is inactive ("createRetVal" has the value "16#7000").
- CREATE_DB has created a new DB whose number was stored ("memNbrNewDB" has a value > 0).



If the normally open contact ("startDelete") supplies the signal state "TRUE", the "DELETE_DB" instruction is executed. The newly created DB is referenced based on its

number with the input parameter DB_NUMBER ("memNbrNewDB"). The "DELETE_DB" instruction deletes the newly created DB.

The output parameter RET_VAL ("deleteRetVal") indicates that processing took place without errors.

Result of network 8

When DELETE_DB at the ENO parameter ("#tempRead") supplies the signal state "TRUE", the interconnection of network 8 is active. In addition, when processing of DELETE_DB is complete, the reset ("startDelete") is set to "TRUE" and the execution of DELETE_DB is ended.

The previously created block is deleted after the execution of DELETE_DB The "memNbrNewDB" tag is set to "0" in order to prevent a command being issued to delete an already deleted block.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

11.4.4.13 Addressing (S7-1200, S7-1500)

Instructions for address conversion (S7-1200, S7-1500)

Description

There are different options for addressing a module (IO address, hardware identifier, slot).

You can convert the address data with the following instructions:

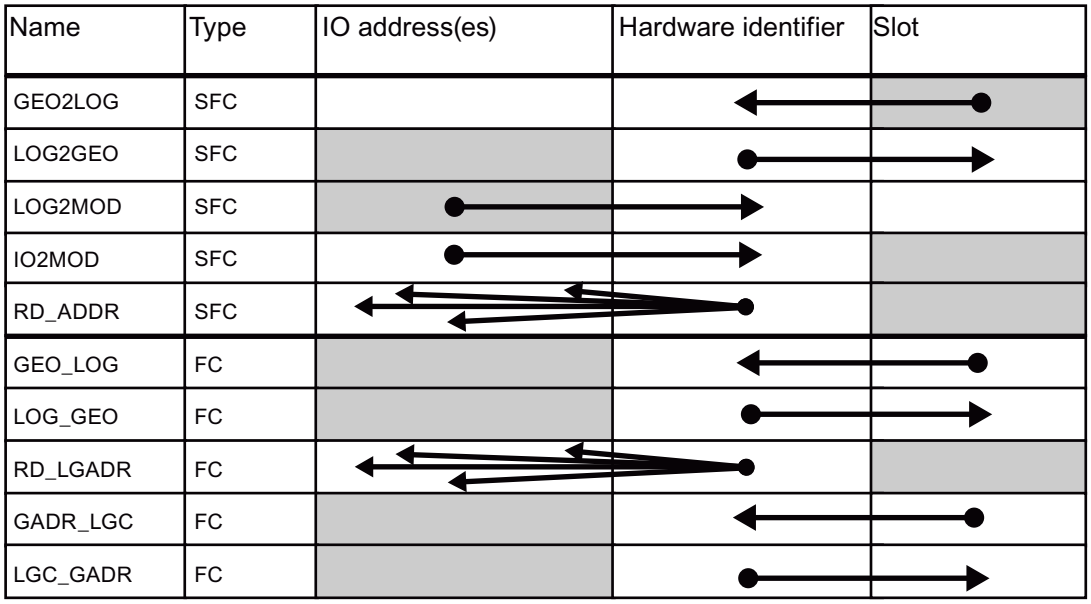
- GEO2LOG: Determine hardware identifier from slot (Page 4074)
- LOG2GEO: Determine slot from hardware identifier (Page 4077)
- LOG2MOD: Determine the hardware identifier from addressing of STEP 7 V5.5 SPx (Page 4079)
- IO2MOD: Determine hardware identifier from an IO address (Page 4080)
- RD_ADDR: Determine IO addresses from the hardware identifier (Page 4083)

The following instructions are supported in addition for migrated projects:

- GEO_LOG: Determine hardware identifier from slot (Page 4086)
- LOG_GEO: Determine slot from hardware identifier (Page 4088)
- RD_LGADR: Determine IO addresses from the hardware identifier (Page 4089)
- GADR_LGC: Determine hardware identifier from slot and offset in the user data address area (Page 4091)
- LGC_GADR: Determine slot from hardware identifier (Page 4092)

Type of address conversion

The figure below shows which instruction runs which address conversion.



GEO2LOG: Determine hardware identifier from slot (S7-1200, S7-1500)

Description

You use the "GEO2LOG" instruction to determine hardware identifier based on slot information that you define using the system data type GEOADDR.

Depending on the type of hardware you define at the parameter HWTYPE the following information is evaluated from the other parameters GEOADDR:

- When HWTYPE = 1 (IO system):
  - Only IOSYSTEM is evaluated. The other parameters of GEOADDR are not taken into consideration.
  - The hardware identifier of the IO system is output.
- When HWTYPE = 2 (IO device):
  - IOSYSTEM and STATION are evaluated. The other parameters of GEOADDR are not taken into consideration.
  - The hardware identifier of the IO device is output.

- With HWTYPE = 4 (module):
  - IOSYSTEM, STATION and SLOT are evaluated. The SUBSLOT parameter of GEOADDR is not taken into consideration.
  - The hardware identifier of the module is output.
- With HWTYPE = 5 (submodule):
  - All parameters of GEOADDR are evaluated.
  - The hardware identifier of the submodule is output.

The AREA parameter of the GEOADDR system data type is not evaluated.

Parameters

The following table shows the parameters of the "GEO2LOG" instruction:

Parameter	Declaration	Data type	Memory area	Description
GEOADDR	Input	VARIANT	D, L	Pointer to the structure of the GEOADDR system data type. The system data type contains the slot information from which the hardware ID is determined. See also: System data type GEOADDR (Page 4085)
RET_VAL	Return	INT	I, Q, M, D, L	Output of error information.
LADDR	Output	HW_ANY	I, Q, M, D, L	Hardware identifier of the assembly or the module. The number is automatically assigned and is stored in the properties in the hardware configuration.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0	No error occurred.
8091	Invalid value for in GEOADDR for HWTYPE.
8094	Invalid value for in GEOADDR for IOSYSTEM.
8095	Invalid value for in GEOADDR for STATION.
8096	Invalid value for in GEOADDR for SLOT.
8097	Invalid value for in GEOADDR for SUBSLOT.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

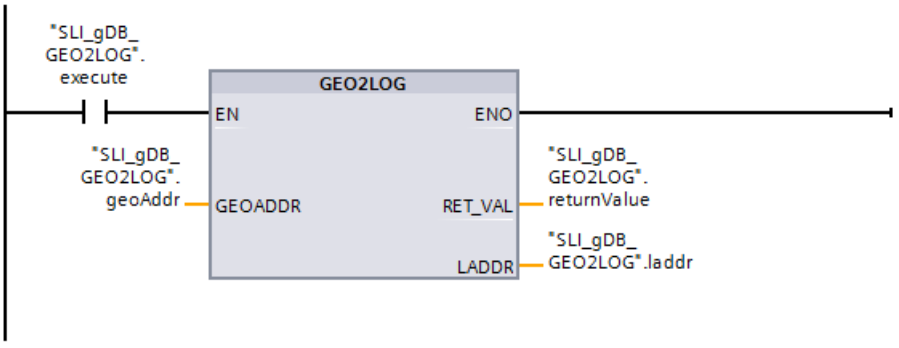
In the following example, you determine the hardware ID of the CPU based on the slot information.

Create four tags in a global data block for storing the data.

SLI_gDB_GEO2LOG

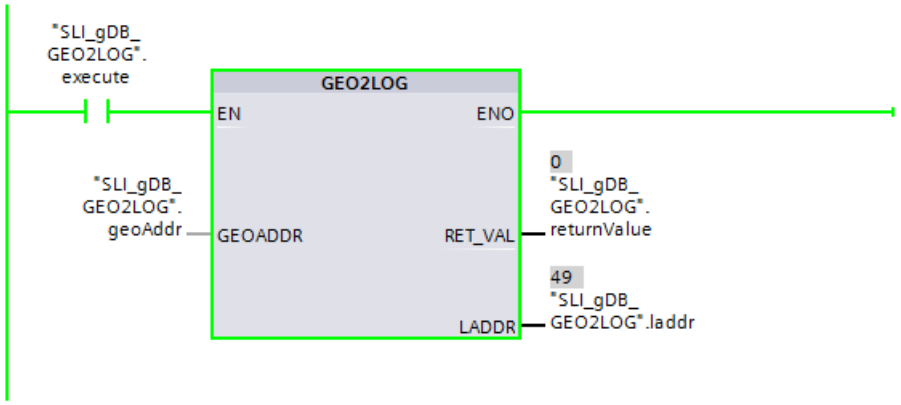
		Name	Data type	Start value
1		Static		
2		execute	Bool	false
3		geoAddr	GEOADDR	
4		HWTYPE	UInt	4
5		AREA	UInt	0
6		IOSYSTEM	UInt	0
7		STATION	UInt	0
8		SLOT	UInt	1
9		SUBSLOT	UInt	0
10		returnValue	Int	0
11		laddr	HW_ANY	16#0

Interconnect the parameters of the instruction as follows.



If the normally open contact ("execute") supplies the signal state "TRUE", the "GEO2LOG" instruction is executed. The slot information is stored in the input parameter GEOADDR ("geoAddr"). The instruction "GEO2LOG" calls the hardware ID for a module (HWTYPE with the value "4") of the central controller (IOSYSTEM with the value "0") in the rack "0" (value of STATION), in slot "1" (value of SLOT).

The determined hardware ID of the local component of the CPU is displayed at the output parameter LADDR ("laddr"). The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.



You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Instructions for address conversion (Page 4073)

LOG2GEO: Determine slot from hardware identifier (S7-1200, S7-1500)

Description

You use the "LOG2GEO" instruction to determine the module slot belonging to a hardware identifier.

Parameters

The following table shows the parameters of the "LOG2GEO" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_ANY	I, Q, M, D, L or constant	Hardware identifier of the module whose slot you want to find. The hardware ID is assigned automatically and is stored in the properties of the module in the hardware configuration and the system constants.
RET_VAL	Output	INT	I, Q, M, D, L	Output of error information.
GEOADDR	InOut	VARIANT	D, L	Pointer to the GEOADDR system data type. The slot information is written in the system data type GEOADDR. See also: System data type GEOADDR (Page 4085)

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0	No error occurred.
8090	The address specified at the LADDR parameter is invalid.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

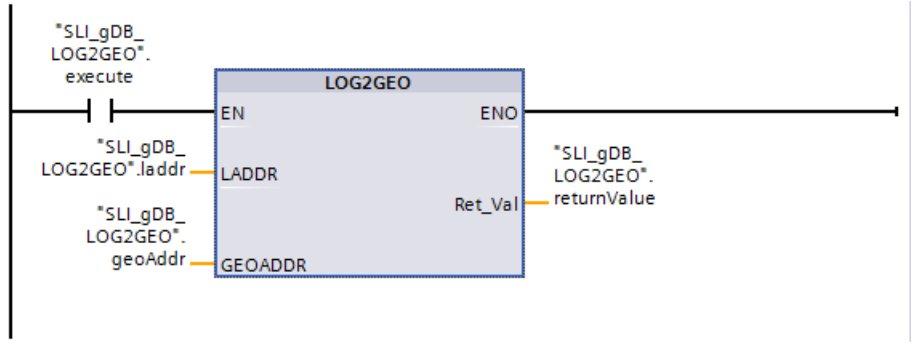
In the following example, you determine the slot information of an input module based on the hardware ID.

Create four tags in a global data block for storing the data.

SLI_gDB_LOG2GEO

		Name	Data type	Start value
1		Static		
2		execute	Bool	false
3		laddr	HW_ANY	257
4		geoAddr	GEOADDR	
5		HWTYPE	UInt	0
6		AREA	UInt	0
7		IOSYSTEM	UInt	0
8		STATION	UInt	0
9		SLOT	UInt	0
10		SUBSLOT	UInt	0
11		returnValue	Int	0

Interconnect the parameters of the instruction as follows.



If the normally open contact ("execute") supplies the signal state "TRUE", the "LOG2GEO" instruction is executed. The hardware ID of the input module is stored with the input parameter LADDR ("laddr"). The instruction "LOG2GEO" calls the slot information of the input module using the hardware ID.

The determined slot information of the input module is displayed at the output parameter GEOADDR ("geoAddr"). A module (HWTYPE with the value "4") of the central controller (IOSYSTEM with the value "0") in the rack "0" (value of STATION), in slot "2" (value of SLOT) is determined.

The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.

SLI_gDB_LOG2GEO

	Name	Data type	Start value	Monitor value
1	Static			
2	execute	Bool	false	TRUE
3	laddr	HW_ANY	257	16#0101
4	geoAddr	GEOADDR		
5	HWTYPE	UInt	0	4
6	AREA	UInt	0	0
7	IOSYSTEM	UInt	0	0
8	STATION	UInt	0	0
9	SLOT	UInt	0	2
10	SUBSLOT	UInt	0	0
11	returnValue	Int	0	0

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

LOG2MOD: Determine the hardware identifier from addressing of STEP 7 V5.5 SPx (S7-1500)

Description

You use the "LOG2MOD" instruction to determine the hardware identifier for an IO (sub)module from the addressing of STEP 7 5.5 SPx (IO data address or diagnostic address).

The hardware identifier is used at the LADDR input parameter for addressing of various instructions. Convert the addressing parameters from STEP 7 5.5 SPx by calling "LOG2MOD" beforehand.

Parameter

The following table shows the parameters of the "LOG2MOD" instruction:

Parameter	Declaration	Data type	Memory area	Description
IOID	Input	BYTE	I, Q, M, D, L or constant	Identifier of the address area as in STEP 7 5.5 SPx: • B#16#00: Bit15 of ADDR specifies whether an input (Bit15=0) or output address (Bit15=1) exists. • B#16#54= Peripheral input (PI) • B#16#55= Peripheral output (PQ)
ADDR	Input	WORD	I, Q, M, D, L or constant	Logical address of the IO data of the module as offset (corresponding addressing in STEP 7 5.5 SPx) or diagnostic address.
RET_VAL	Return	INT	I, Q, M, D, L	Error code of the instruction.
HWID	Output	HW_IO	I, Q, M, D, L	Determined hardware identifier of the IO (sub)module.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0	No error occurred.
8093	• Specified address is not used by any hardware components. • Specified value at IOID parameter is invalid.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

IO2MOD: Determine hardware identifier from an IO address (S7-1200, S7-1500)

Description

The "IO2MOD" instruction determines the hardware identifier of the module from an IO address (I, Q, PI, PQ) of a module.

Enter the IO address at the ADDR parameter.

- If a series of IO addresses is used at this parameter, only the first address is evaluated to determine the hardware identifier.

- If the first address is correctly specified, the length for the address specification at the ADDR parameter is of no significance.

- If an address area is used that encompasses several modules or unused addresses, the hardware identifier of the first module can also be determined.

- If no IO address of a module is specified at parameter ADDR, the error code 8090 is output at parameter RET_VAL.

Note

Input of IO address in SCL

In SCL, you do not program using the I/O access ID "%QWx:P". In this case, use the symbolic tag name or the absolute address in the process image.

Parameters

The following table shows the parameters of the "IO2MOD" instruction:

Parameter	Declaration	Data type	Memory area	Description
ADDR	Input	VARIANT	I, Q	IO address (I, Q, PI, PQ) within a module. Make sure that slice access is not used for the parameter ADDR. If this is the case, incorrect values are output at the LADDR parameter.
RET_VAL	Return	INT	I, Q, M, D, L	Error code of the instruction.
LADDR	Output	HW_IO	I, Q, M, D, L	Determined hardware identifier (logical address) of the IO module.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

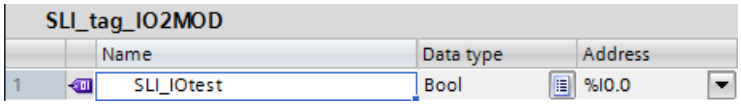
Parameter RET_VAL

Error code* (W#16#...)	Explanation
0	No error occurred.
8090	IO address specified at ADDR parameter is not used by any hardware component.
8092	Invalid data type used at parameter ADDR (for example, WCHAR or WSTRING).
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

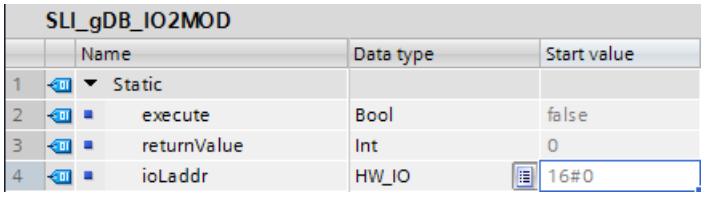
Example

In the following example, you determine the hardware ID of an input module based on an IO address.

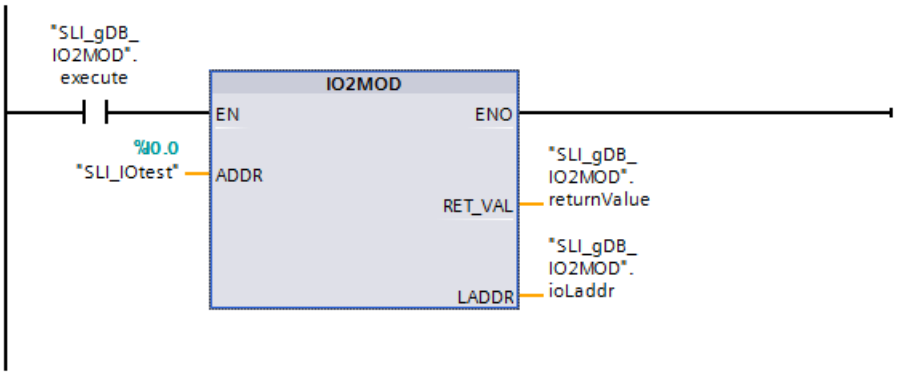
Create a PLC tag in a tag table to read the hardware ID.



Create three tags in a global data block for storing the data.

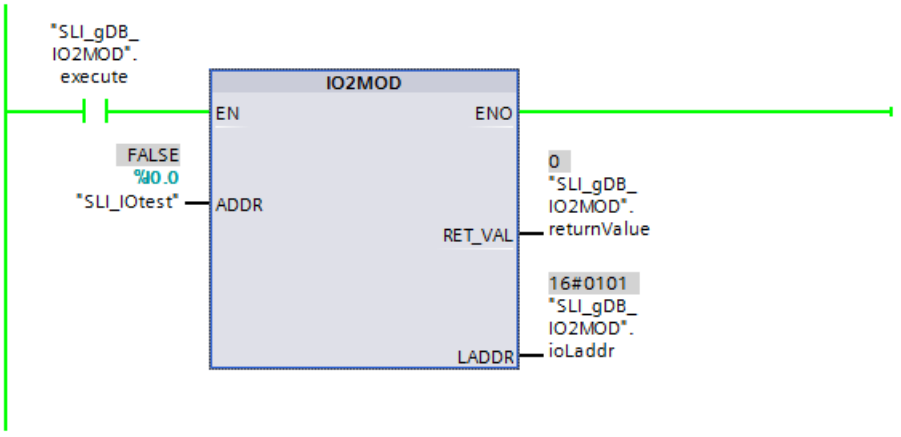


Interconnect the parameters of the instruction as follows.



If the normally open contact ("execute") supplies the signal state "TRUE", the "IO2MOD" instruction is executed. The IO address to be localized is transmitted with the input parameter ADDR ("SLI_IOtest"). The "IO2MOD" instruction calls the IO address and reads the input module.

The determined hardware ID of the input module is displayed at the output parameter LADDR ("ioLaddr"). The output parameter RET_VAL ("returnValue") indicates that processing took place without errors.



You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

RD_ADDR: Determine IO addresses from the hardware identifier (S7-1200, S7-1500)

Description

The "RD_ADDR" instruction determines the length and the start address of the inputs or outputs based on the hardware identifier of a sub(module).

- Use the LADDR parameter to select the input or output module based on the hardware identifier.
- The following output parameters are used depending on whether it is an input module or output module:
  - In the case of an input module the determined values are output at the PIADDR and PICOUNT parameters.
  - In the case of an output module the determined values are output at the PQADDR and PQCOUNT parameters.
- The PIADDR and PQADDR parameters each contain the start address of the I/O addresses of the module.
- The PICOUNT and PQCOUNT parameters each contain the number of bytes of the inputs our outputs (1 byte for 8 inputs/outputs, 2 bytes for 16 inputs/outputs).

Note

Determining the IO addresses with packed addresses of an ET200

With packed addresses of an ET200. the first module of the pack group returns all addresses.

For the other modules, the following is output with the PIADDR and PQADDR parameters:

- with PROFINET as address "0"
- with PROFIBUS as address "0" In addition, the error code W#16#8090 is output.

For the number of bytes of the inputs and outputs (parameters PICOUNT and PQCOUNT), "0" is output in each case.

Parameter

The following table shows the parameters of the "RD_ADDR" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the (sub)module.
RET_VAL	Return	INT	I, Q, M, D, L	Error code of the instruction.
PIADDR	Output	UDINT	I, Q, M, D, L	Start address of the input module.
PICOUNT	Output	UINT	I, Q, M, D, L	Number of bytes of the inputs.
PQADDR	Output	UDINT	I, Q, M, D, L	Start address of the output module.
PQCOUNT	Output	UINT	I, Q, M, D, L	Number of bytes of the outputs.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0	No error occurred.
8090	Hardware identifier of the module at the LADDR parameter is invalid.
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

Example

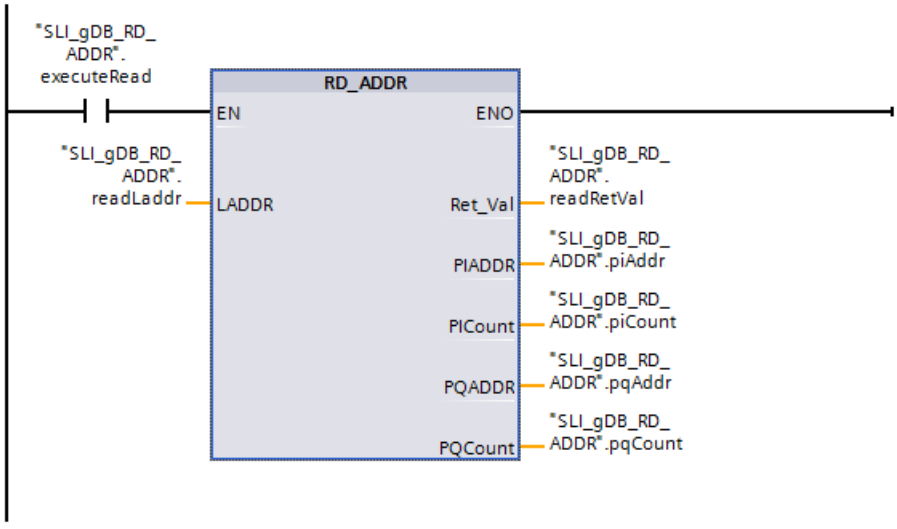
In the following example, you determine the length and start address of the inputs in an input module based on the hardware ID.

Create seven tags in a global data block for storing the data.

SLI_gDB_RD_ADDR

		Name	Data type	Start value
1		Static		
2		executeRead	Bool	false
3		readLaddr	HW_IO	257
4		readRetVal	Int	0
5		piAddr	UDInt	0
6		piCount	UInt	0
7		pqAddr	UDInt	0
8		pqCount	UInt	0

Interconnect the parameters of the instruction as follows.



If the normally open contact ("executeRead") supplies the signal state "TRUE", the "RD_ADDR" instruction is executed. The hardware ID of the input module is stored with the input parameter LADDR ("readLaddr"). The instruction "RD_ADDR" determines the input module using the hardware ID and reads the length of the inputs and start address.

The determined start address is displayed at the output parameter PIADDR ("piAddr"). The number of bytes of the inputs is displayed at the output parameter PICOUNT ("piCount").

The output parameter RET_VAL ("readRetVal") indicates that processing took place without errors.

SLI_gDB_RD_ADDR

	Name	Data type	Start value	Monitor value
1	Static			
2	executeRead	Bool	false	TRUE
3	readLaddr	HW_IO	257	16#0101
4	readRetVal	Int	0	0
5	piAddr	UDInt	0	0
6	piCount	UInt	0	2
7	pqAddr	UDInt	0	0
8	pqCount	UInt	0	0

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

System data type GEOADDR (S7-1200, S7-1500)

Geographical address

The system data type GEOADDR contains the geographical address of a module, i.e. the slot information.

- Geographical address for PROFINET IO
  For PROFINET IO the geographical address was composed of the ID of the PROFINET IO system, the device number, the slot number and the submodule (if a sub-module is used).
- Geographical address for PROFIBUS DP
  For PROFIBUS DP, the geographical address consists of the ID of the DP master system, the station number and the slot number.

The slot information of the modules can be found in the hardware configuration of each module.

System data type GEOADDR

If you enter "GEOADDR" as the data type in a data block, the structure GEOADDR is automatically created.

The system data type GEOADDR has the following structure:

Parameter name	Data type	Description
GEOADDR	STRUCT	
HWTYPE	UINT	Hardware type: • 1: IO system (PROFINET/PROFIBUS) • 2: IO device/DP slave • 3: Rack • 4: Module • 5: Submodule If a hardware type is not supported by the instruction, a HWTYPE "0" is output.
AREA	UINT	Area ID: • 0 = CPU • 1 = PROFINET IO • 2 = PROFIBUS DP • 3 = AS-i
IOSYSTEM	UINT	PROFINET IO system (0=central unit in the rack)
STATION	UINT	• Number of the rack if area identifier AREA = 0 (central module). • Station number if area identifier AREA > 0.
SLOT	UINT	Slot number
SUBSLOT	UINT	Number of the submodule. This parameter has the value "0" if no submodule is available or can be plugged.

Legacy (S7-1200, S7-1500)

GEO_LOG: Determine hardware identifier from slot (S7-1500)

Description

The corresponding module slot of a signal module is known. Use the "GEO_LOG" instruction to determine the corresponding hardware identifier of the module from this.

Parameter

The following table shows the parameters of the "GEO_LOG" instruction:

Parameter	Declaration	Data type	Memory area	Description
MASTER	Input	INT	I, Q, M, D, L or constant	Area ID: • 0, if the slot is located in a centralized configuration. • 1 to 32: DP master system ID of the associated field device if the slot is located in a field device on PROFIBUS • 100 to 115: PROFINET IO system ID of the associated field device if the slot is located in a field device on PROFINET
STATION	Input	INT	I, Q, M, D, L or constant	• If MASTER = 0: Number of the rack • If MASTER > 0: Station number of the field device
SLOT	Input	INT	I, Q, M, D, L or constant	Slot number
SUBSLOT	Input	INT	I, Q, M, D, L or constant	Number of the submodule. If no submodule exists or can be inserted SUBSLOT must have the value "0".
RET_VAL	Return	INT	I, Q, M, D, L	Error information
LADDR	Output	HW_IO	I, Q, M, D, L	Hardware identifier of the module

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8094	No subnet was configured with the specified SUBNETID .
8095	Illegal value for STATION parameter
8096	Illegal value for SLOT parameter
8097	Illegal value for SUBSLOT parameter.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

See also

Instructions for address conversion (Page 4073)

LOG_GEO: Determine slot from hardware identifier (S7-1500)

Description

You use the "LOG_GEO" instruction to determine the module slot belonging to a hardware identifier.

Parameters

The following table shows the parameters of the "LOG_GEO" instruction:

Parameter	Declaration	Data type	Memory area	Description
LADDR	Input	HW_IO	I, Q, M, D, L or constant	Hardware identifier of the module for which the slot is to be determined.
RET_VAL	Return	INT	I, Q, M, D, L	Error information
AREA	Output	INT	I, Q, M, D, L	The area ID indicates how the remaining output parameters are to be interpreted: • 0: central device • 2: PROFIBUS DP / PROFINET IO
MASTER	Output	INT	I, Q, M, D, L	With AREA = 0: • 0: If the slot is located in one of the racks (central device). With AREA = 2: • 1 to 32: DP master system ID of the associated field device if the slot is located in a field device on PROFIBUS • 100 to 115: PROFINET IO system ID of the associated field device if the slot is located in a field device on PROFINET
STATION	Output	INT	I, Q, M, D, L	• With MASTER = 0: Number of the rack • With MASTER > 0: Station number of the field device
SLOT	Output	INT	I, Q, M, D, L	Slot number
SUBSLOT	Output	INT	I, Q, M, D, L	Number of the submodule
OFFSET	Output	INT	I, Q, M, D, L	The OFFSET parameter is not output by the instruction (always "0").

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8090	Specified logical address invalid
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

See also

Instructions for address conversion (Page 4073)

RD_LGADR: Determine IO addresses from the hardware identifier (S7-1200, S7-1500)

Description

Based on the hardware identifier, you can determine the IO addresses of a module, a central submodule or a submodule for PNIO and DP slaves with the "RD_LGADR" instruction.

- You can specify the hardware ID of the submodule at the LADDR parameter.
- The addresses are written to the PEADDR and PAADDR parameters is ascending order.
  - For an input module, only the PEADDR parameter is written. For an output module, the PAADDR parameter is written.
  - An Array of WORD is used to store the addresses in each case.
- The number of addresses is output at the PECOUNT parameter (for an input module) and PACOUNT parameter (for an output module).

Note

Description of the boxes PEADDR and PAADDR

In PEADDR or PAADDR only as many addresses are entered as are returned (This number is entered in PECOUNT or PACOUNT). Addresses of earlier RD_LGADR-calls located behind the currently returned addresses are retained; they must not be evaluated in the current call.

Note

Determining the IO addresses with packed addresses of an ET200

With packed addresses of an ET200. the first module of the pack group returns all addresses.

For the other modules, the following is output with the PEADDR and PAADDR parameters:

- with PROFINET as address "0"
- with PROFIBUS as address "0" In addition, the error code W#16#8090 is output.

For the number of bytes of the inputs and outputs (parameters PECOUNT and PACOUNT), "0" is output in each case.

Parameters

The following table shows the parameters of the "RD_LGADR" instruction:

Parameter	Declaration	Data type	Memory area	Description
IOID	Input	BYTE	I, Q, M, D, L or constant	Is not evaluated
LADDR	Input	HW_ANY	I, Q, M, D, L or constant	Hardware identifier of the module or the sub-module.
RET_VAL	Return	INT	I, Q, M, D, L	Error information
PEADDR	Output	ANY	I, Q, M, D, L	Field for the PI addresses with the Array of WORD data type
PECOUNT	Output	INT	I, Q, M, D, L	Number of returned PI addresses
PAADDR	Output	ANY	I, Q, M, D, L	Field for the PQ addresses with the Array of WORD data type
PACOUNT	Output	INT	I, Q, M, D, L	Number of returned PQ addresses

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code (W#16#...)	Explanation
0000	No error occurred.
8090	Specified hardware identifier is invalid
80A0	Error in the output parameter PEADDR: The data type of the array elements is not WORD.
80A1	Error in the output parameter PAADDR: The data type of the array elements is not WORD.
80A2	Error in the output parameter PEADDR: The specified array does not accommodate all the logical addresses.
80A3	Error in the output parameter PAADDR: The specified array does not accommodate all the logical addresses.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)

See also

Instructions for address conversion (Page 4073)

GADR_LGC: Determine hardware identifier from slot and offset in the user data address area (S7-1500)

Description

You use the "GADR_LGC" instruction to determine the hardware identifier of a signal module. The hardware identifier is determined from the module slot and the offset in the user data address area of the module.

Note

Restriction on use

The "GADR_LGC" instruction cannot be used for modules behind gateways (e.g., IE/PB link). Use the "GEO2LOG" instruction instead.

Parameters

The following table shows the parameters of the "GADR_LGC" instruction:

Parameter	Declaration	Data type	Memory area	Description
SUBNETID	Input	BYTE	I, Q, M, D, L or constant	Area ID: • 0: If the slot is located in the central module • 1 to 32: DP master system ID of the corresponding distributed I/O system if the slot is in a distributed I/O device. • 100 to 115: PROFINET IO system ID of the associated field device if the slot is located in a field device on PROFINET
RACK	Input	WORD	I, Q, M, D, L or constant	• Number of the rack, if area identifier is 0 • Device number of the distributed I/O device if the area identifier > 0
SLOT	Input	WORD	I, Q, M, D, L or constant	Slot no.
SUBSLOT	Input	BYTE	I, Q, M, D, L or constant	Irrelevant
SUBADDR	Input	WORD	I, Q, M, D, L or constant	Offset in the user data address area of the module
RET_VAL	Return	INT	I, Q, M, D, L	Error information
IOID	Output	BYTE	I, Q, M, D, L	The IOID output parameter is not written (always "0").
LADDR	Output	HW_MODULE	I, Q, M, D, L	Hardware identifier of the module

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8094	No subnet was configured with the specified SUBNETID .
8095	Illegal value for RACK parameter.
8096	Illegal value for SLOT parameter.
8098	Illegal value for SUBADDR parameter.
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

See also

Instructions for address conversion (Page 4073)

LGC_GADR: Determine slot from hardware identifier (S7-1500)

Description

You use the "LGC_GADR" instruction to determine the module slot belonging to a hardware identifier.

Note

Restriction on use

The "LGC_GADR" instruction cannot be used for modules behind gateways (e.g., IE/PB link). Use the "LOG2GEO" instruction instead.

Parameters

The following table shows the parameters of the "LGC_GADR" instruction:

Parameter	Declaration	Data type	Memory area	Description
IOID	Input	BYTE	I, Q, M, D, L or constant	Is not evaluated.
LADDR	Input	HW_MODULE	I, Q, M, D, L or constant	Hardware identifier of the module
RET_VAL	Return	INT	I, Q, M, D, L	Error information
AREA	Output	BYTE	I, Q, M, D, L	The area ID indicates how the remaining output parameters are to be interpreted: • 0: Central module • 2: PROFIBUS DP

Parameter	Declaration	Data type	Memory area	Description
RACK	Output	WORD	I, Q, M, D, L	Rack number: • With central module (AREA= 0): – Rack number • For PROFIBUS DP (AREA=2): – Low byte: Station number – High byte: DP master system ID
SLOT	Output	WORD	I, Q, M, D, L	Slot number: • With central module (AREA= 0): – Slot number • For PROFIBUS DP (AREA=2): – Slot no. in the station
SUBADDR	Output	WORD	I, Q, M, D, L	Is not output (always "0").

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8090	Specified logical address invalid
8093	This instruction is not permitted for the module selected by the parameters IOID and LADDR .
General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

See also

Instructions for address conversion (Page 4073)

11.4.5 Technology (S7-1200, S7-1500)

11.4.5.1 S7-1200 Motion Control (S7-1200)

S7-1200 Motion Control as of V6 (S7-1200)

MC_Power (S7-1200)

MC_Power: Enable, disable axis as of V6 (S7-1200)

Description

The Motion Control instruction "MC_Power" enables or disables an axis.

Requirements

- The positioning axis technology object has been configured correctly.
- There is no pending enable-inhibiting error.

Override response

Execution of "MC_Power" cannot be aborted by a Motion Control command.

Disabling the axis (input parameter "Enable" = FALSE) aborts all Motion Control commands for the associated technology object in accordance with the selected "StopMode".

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis	-	Axis technology object	
Enable	INPUT	BOOL	FALSE	TRUE	The axis is enabled.
				FALSE	All current jobs are interrupted in accordance with the "StopMode" configured. The axis is stopped and disabled.
StartMode	INPUT	INT	1	0	Enable positioning axis not position-controlled *)
				1	Enable positioning axis position-controlled *)
				*) This parameter is ignored when a positioning axis with PTO (Pulse Train Output) drive is used. The parameter initially takes effect when the positioning axis is enabled (Enable changes from FALSE to TRUE) and when the axis is enabled after successful acknowledgment of an interrupt that caused the axis to be disabled.	

Parameter	Declaration	Data type	Default value	Description	
StopMode	INPUT	INT	0	0	Emergency stop If a request to disable the axis is pending, the axis brakes at the configured emergency deceleration. The axis is disabled after reaching standstill.
				1	Immediate stop If a request to disable the axis is pending, this set-point zero is output and the axis is disabled. The axis is braked depending on the configuration in the drive, and is brought to a standstill. With drive connection via PTO (Pulse Train Output): When you disable the axis, the pulse output is stopped with a frequency-dependent deceleration: • Output frequency ≥ 100 Hz Deceleration: max. 30 ms • Output frequency < 100 Hz Deceleration: 30 ms up to max. 1.5 s at 2 Hz
				2	Emergency stop with jerk control If a request to disable the axis is pending, the axis brakes at the configured emergency deceleration. If the jerk control is activated, the configured jerk is taken into account. The axis is disabled after reaching standstill.
Status	OUTPUT	BOOL	FALSE	Status of axis enable	
				FALSE	The axis is disabled. The axis does not execute Motion Control commands and does not accept any new commands (exception: MC_Reset command) For drive connection via PTO (Pulse Train Output): The axis is not homed. Upon disabling, the status does not change to FALSE until the axis reaches a standstill.
				TRUE	The axis is enabled. The axis is ready to execute Motion Control commands. Upon axis enabling, the status does not change to TRUE until the signal "Drive ready" is pending. If the "Drive ready" drive interface was not configured in the axis configuration, the status changes to TRUE immediately.
Busy	OUTPUT	BOOL	FALSE	TRUE	MC_Power" is active.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred in Motion Control instruction "MC_Power" or in the associated technology object. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	

Note

If the axis is switched off due to an error, it will be enabled again automatically after the error has been eliminated and acknowledged. This requires that the input parameter "Enable" has retained the value TRUE during this process.

Enabling an axis with configured drive interface

To enable the axis, follow these steps:

1. Check the requirements indicated above.
2. Initialize input parameters "StartMode" and "StopMode" with the desired value. Set the input parameter "Enable" to TRUE.
   The enable output for "Drive enabled" changes to TRUE to enable the power to the drive.
   The CPU waits for the "Drive ready" signal of the drive.
   When the "Drive ready" signal is available at the configured ready input of the CPU, the axis is enabled. The output parameter "Status" and the variable of the technology object <axis name>.StatusBits.Enable indicate the value TRUE.

Enabling an axis without configured drive interface

To enable the axis, follow these steps:

1. Check the requirements indicated above.
2. Initialize input parameters "StartMode" and "StopMode" with the desired value. Set the input parameter "Enable" to TRUE. The axis is enabled. The output parameter "Status" and the variable of the technology object <axis name>.StatusBits.Enable indicate the value TRUE.

Disabling an axis

To disable an axis, you can follow the steps described below:

1. Bring the axis to a standstill.
   You can identify when the axis is at a standstill in the variable of the technology object <axis name>.StatusBits.StandStill.
2. Set input parameter "Enable" to FALSE after standstill is reached.
3. If output parameters "Busy" and "Status" and variable of technology object <axis name>.StatusBits.Enable indicate the value FALSE, disabling of the axis is complete.

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)

MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)

MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)

MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

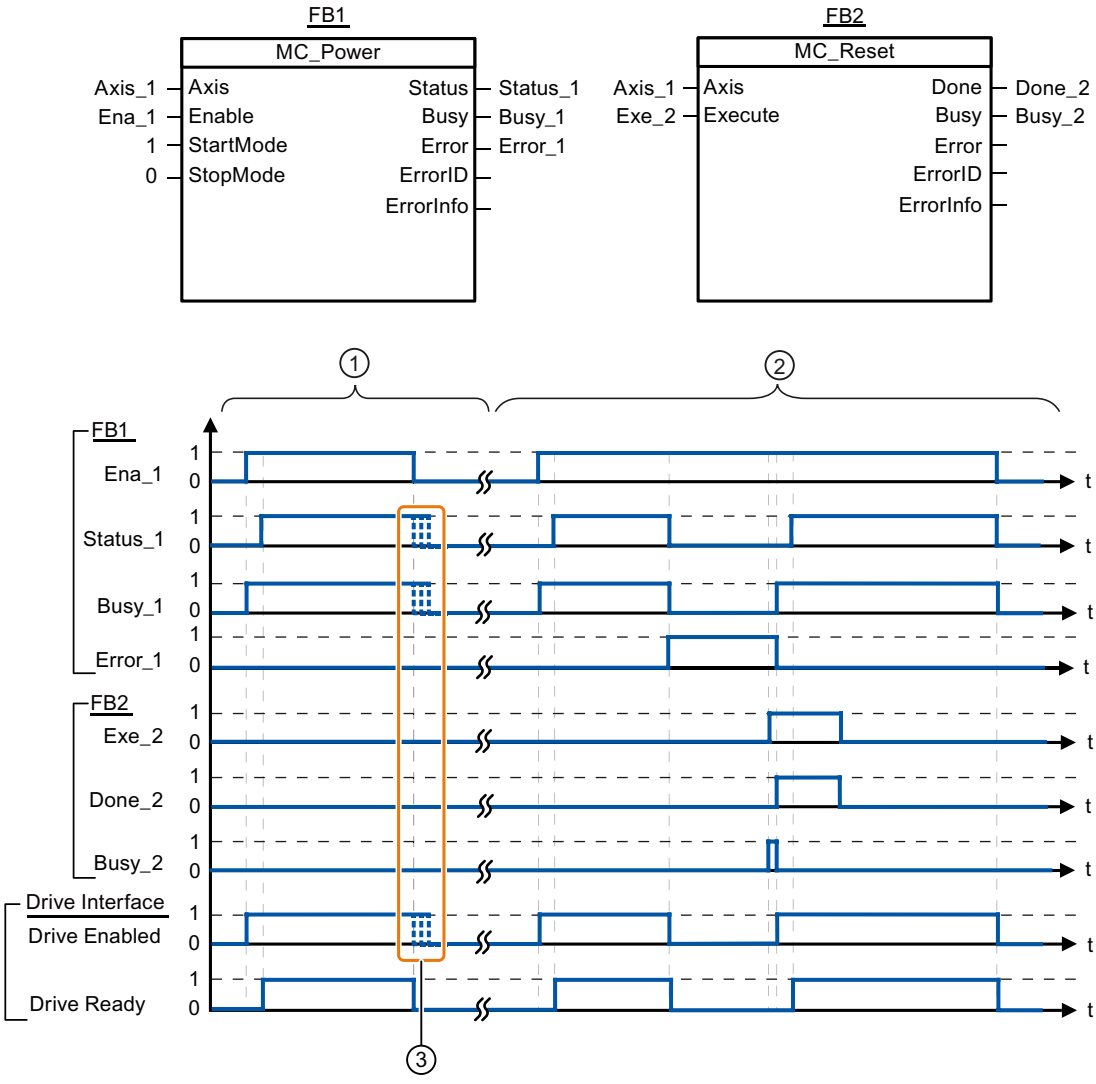
MC_Halt: Stop axis as of V6 (Page 4105)

Configuration - Standstill signal (PROFIdrive and analog drive connection only) (Page 6220)

Move the axis without position control for servicing (Page 6303)

MC_Power: Function chart as of V6 (S7-1200)

Function chart



①	An axis is enabled and then disabled again. When the drive has signaled "Drive ready" back to the CPU, the successful enable can be read out via "Status_1".
②	Following an axis enable, an error has occurred that caused the axis to be disabled. The error is eliminated and acknowledged with "MC_Reset". The axis is then enabled again.
③	The exact end of the signals depends on the selected drive and the StopMode.

MC_Reset (S7-1200)

MC_Reset: Acknowledge fault, restart technology object as of V6 (S7-1200)

Description

Motion Control instruction "MC_Reset" can be used to acknowledge "Operating error with axis stop" and "Configuration error". The errors that require acknowledgment can be found in the "List of ErrorIDs and ErrorInfos" under "Remedy".

The axis configuration can be downloaded to the work memory after a download in RUN mode.

Requirements

- The positioning axis technology object has been configured correctly.
- The cause of a pending configuration error requiring acknowledgment has been eliminated (for example, acceleration in positioning axis technology object has been changed to a valid value).

Override response

The MC_Reset command cannot be aborted by any other Motion Control command.

The new MC_Reset command does not abort any other active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Restart	INPUT	BOOL	FALSE	TRUE	Download the axis configuration from the load memory to the work memory. The command can only be executed when the axis is disabled. Refer to the notes on Download to the CPU.
				FALSE	Acknowledges pending errors
Done	OUTPUT	BOOL	FALSE	TRUE	Error has been acknowledged.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	

Acknowledging an error requiring acknowledgment with MC_Reset

To acknowledge an error, follow these steps:

1. Check the requirements indicated above.
2. Start the acknowledgment of the error with a rising edge at input parameter "Execute".
3. If output parameter "Done" indicates the value TRUE and tag of technology object <Axis name>.StatusBits.Error the value FALSE, the error has been acknowledged.

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

MC_Halt: Stop axis as of V6 (Page 4105)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)

MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)

MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)

MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

MC_Home (S7-1200)

MC_Home: Home axes, set reference point as of V6 (S7-1200)

Description

Motion Control instruction "MC_Home" is used to match the axis coordinates to the real, physical drive position. Homing is required for absolute positioning of the axis. The following types of homing can be executed:

- Active homing (Mode = 3)
  The homing procedure is executed automatically.
- Passive homing (Mode = 2)
  During passive homing, the "MC_Home" Motion Control instruction does not carry out any homing motion. The traversing motion required for this must be implemented by the user via other Motion Control instructions. When the homing switch is detected, the axis is homed.
- Direct homing absolute (Mode = 0)
  The current axis position is set to the value of parameter "Position".

- Direct homing relative (Mode = 1)
  The current axis position is offset by the value of parameter "Position".
- Absolute encoder adjustment relative (Mode = 6)
  The current axis position is offset by the value of parameter "Position".
- Absolute encoder adjustment absolute (Mode = 7)
  The current axis position is set to the value of parameter "Position".

Mode 6 and 7 can only be used with drives with an analog interface and PROFIdrive drive.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.
- No MC_CommandTable command may be active upon start with Mode = 0, 1, or 2.

Override response

The override response depends on the selected mode:

Mode = 0, 1, 6, 7

The MC_Home command cannot be aborted by any other Motion Control command.

The MC_Home command does not abort any active Motion Control commands. Position-related motion commands are resumed after homing according to the new homing position (value at input parameter: "Position").

Mode = 2

The MC_Home command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 2, 3

The new MC_Home command aborts the following active Motion Control command:

- MC_Home command Mode = 2

Position-related motion commands are resumed after homing according to the new homing position (value at input parameter: "Position").

Mode = 3

The MC_Home command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_Home command aborts the following active Motion Control commands:

- MC_Home command Mode = 2, 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description
Axis	INPUT	TO_Axis	-	Axis technology object
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge
Position	INPUT	REAL	0.0	• Mode = 0, 2, and 3 Absolute position of axis after completion of the homing operation • Mode = 1 Correction value for the current axis position Limit values: $-1.0e^{12} \leq$ Position $\leq 1.0e^{12}$

Parameter	Declaration	Data type	Default value	Description	
Mode	INPUT	INT	0	Homing mode	
				0	Direct homing (absolute) New axis position is the position value of parameter "Position".
				1	Direct homing (relative) New axis position is the current axis position + position value of parameter "Position".
				2	Passive homing Homing according to the axis configuration. Following homing, the value of parameter "Position" is set as the new axis position.
				3	Active homing Homing procedure in accordance with the axis configuration. Following homing, the value of parameter "Position" is set as the new axis position.
				6	Absolute encoder adjustment (relative) The current axis position is offset by the value of parameter "Position". The calculated absolute value offset is stored retentively in the CPU. (<AxisName>.StatusSensor.AbsEncoderOffset)
				7	Absolute encoder adjustment (absolute) The current axis position is set to the value of parameter "Position". The calculated absolute value offset is stored retentively in the CPU. (<AxisName>.StatusSensor.AbsEncoderOffset)
Done	OUTPUT	BOOL	FALSE	TRUE	Command completed
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	
ReferenceMarkPosition	OUTPUT	REAL	0.0	Display of the position at which the technology object was homed (valid when "Done" = TRUE)	

Resetting the "Homed" status

The "Homed" status of a technology object (<Axis name>.StatusBits.HomingDone) is reset under the following conditions:

- **Drive connection via PTO (Pulse Train Output):**
  - Start an "MC_Home" command for active homing
    (After successful completion of the homing operation, the "Homed" status is set again.)
  - Disabling of axis by the "MC_Power" Motion Control instruction
  - Changeover between automatic mode and manual control
  - After POWER OFF -> POWER ON of the CPU
  - After CPU restart (RUN-STOP -> STOP-RUN)
- **Technology objects with incremental actual values:**
  - Start an "MC_Home" command for active homing
    (After successful completion of the homing operation, the "Homed" status is set again.)
  - Error in the encoder system, or encoder failure
  - Restart of the technology object
  - After POWER OFF → POWER ON of the CPU
  - Memory reset
  - Modification of the encoder configuration
- **Technology objects with absolute actual values:**
  - Errors in the sensor system/encoder failure
  - Replacement of the CPU
  - Modification of the encoder configuration
  - Restoration of the CPU factory settings
  - Transfer of a different project to the controller

Homing an axis

To home the axis, follow these stops:

1. Check the requirements indicated above.
2. Provide the necessary input parameters with values and start the homing operation with a rising edge at input parameter "Execute".
3. If output parameter "Done" and technology object variable <axis name>.StatusBits.HomingDone indicate the value TRUE, homing is complete. The reference position can be taken from the <axis name>.ReferenceMarkPosition variable.

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)
MC_Halt: Stop axis as of V6 (Page 4105)
MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)
MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)
MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)
MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)
MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)
MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)
MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)
MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

MC_Halt (S7-1200)

MC_Halt: Stop axis as of V6 (S7-1200)

Description

The "MC_Halt" Motion Control instruction stops all movements and brings the axis to a standstill with the configured deceleration. The standstill position is not defined.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_Halt command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_Halt command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command

- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Done	OUTPUT	BOOL	FALSE	TRUE	Zero velocity reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)

MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)

MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

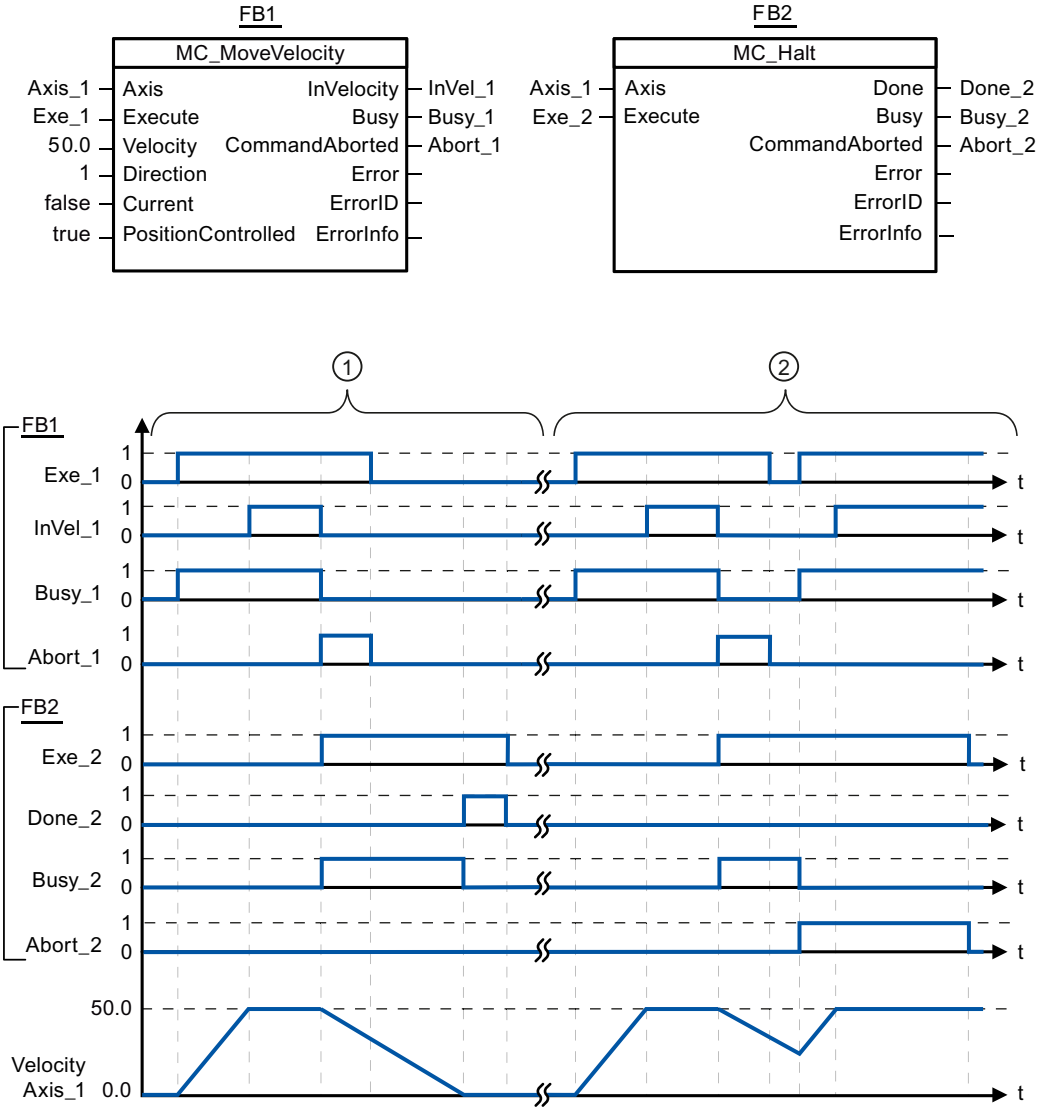
MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)

MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

Move the axis without position control for servicing (Page 6303)

MC_Halt: Function chart as of V6 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 5.0

①	The axis is braked by an MC_Halt command until it comes to a standstill. The axis standstill is signaled via "Done_2".
②	While an MC_Halt command is braking the axis, this command is aborted by another motion command. The abort is signaled via "Abort_2".

MC_MoveAbsolute (S7-1200)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (S7-1200)

Description

The "MC_MoveAbsolute" Motion Control instruction starts an axis positioning motion to move it to an absolute position.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.
- The axis is homed.

Override response

The MC_MoveAbsolute command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveAbsolute command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description
Axis	INPUT	TO_Positionin-gAxis	-	Axis technology object
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge

Parameter	Declaration	Data type	Default value	Description	
Position	INPUT	REAL	0.0	Absolute target position Limit values: $-1.0e^{12} \leq$ Position $\leq 1.0e^{12}$	
Velocity	INPUT	REAL	10.0	Velocity of axis This velocity is not always reached on account of the configured acceleration and deceleration and the target position to be approached. Limit values: Start/stop velocity ≤ Velocity ≤ maximum velocity	
Direction	INPUT	INT	1	Motion direction of the axis Is only evaluated with "modulo" enabled. "Technology object > Configuration > Extended parameters > Modulo > Enable Modulo" Parameter is ignored with PTO axes.	
				0	The sign for the speed ("Velocity" parameter) determines the motion direction.
				1	Positive direction (Target position is approached in a positive direction)
				2	Negative direction (Target position is approached in a negative direction)
				3	Shortest distance (Starting from the current position, the technology selects the shortest distance to the target position)
Done	OUTPUT	BOOL	FALSE	TRUE	Absolute target position reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

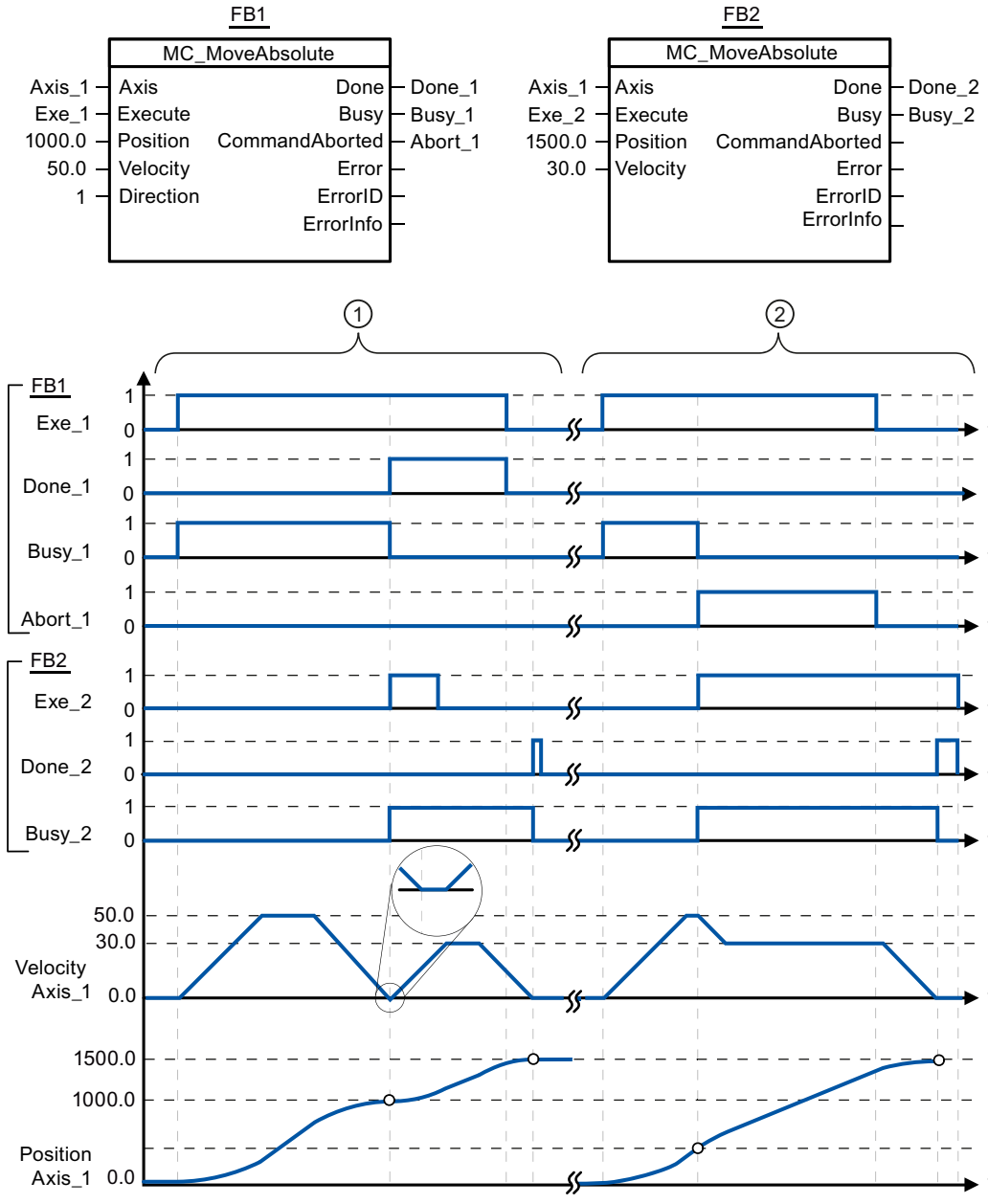
MC_Halt: Stop axis as of V6 (Page 4105)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)
MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)
MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)
MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)
MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)
MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)
Configuration - Modulo (PROFIdrive/analog drive connection only) (Page 6197)

MC_MoveAbsolute: Function chart as of V6 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	An axis is moved to absolute position 1000.0 with an MC_MoveAbsolute command. When the axis reaches the target position, this is signaled via "Done_1". When "Done_1" = TRUE, another MC_MoveAbsolute command, with target position 1500.0, is started. Because of the response times (e.g., cycle time of user program, etc.), the axis comes to a standstill briefly (see zoomed-in detail). When the axis reaches the new target position, this is signaled via "Done_2".
②	An active MC_MoveAbsolute command is aborted by another MC_MoveAbsolute command. The abort is signaled via "Abort_1". The axis is then moved at the new velocity to the new target position 1500.0. When the new target position is reached, this is signaled via "Done_2".

MC_MoveRelative (S7-1200)

MC_MoveRelative: Relative positioning of axis as of V6 (S7-1200)

Description

The "MC_MoveRelative" Motion Control instruction starts a positioning motion relative to the start position.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_MoveRelative command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveRelative command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

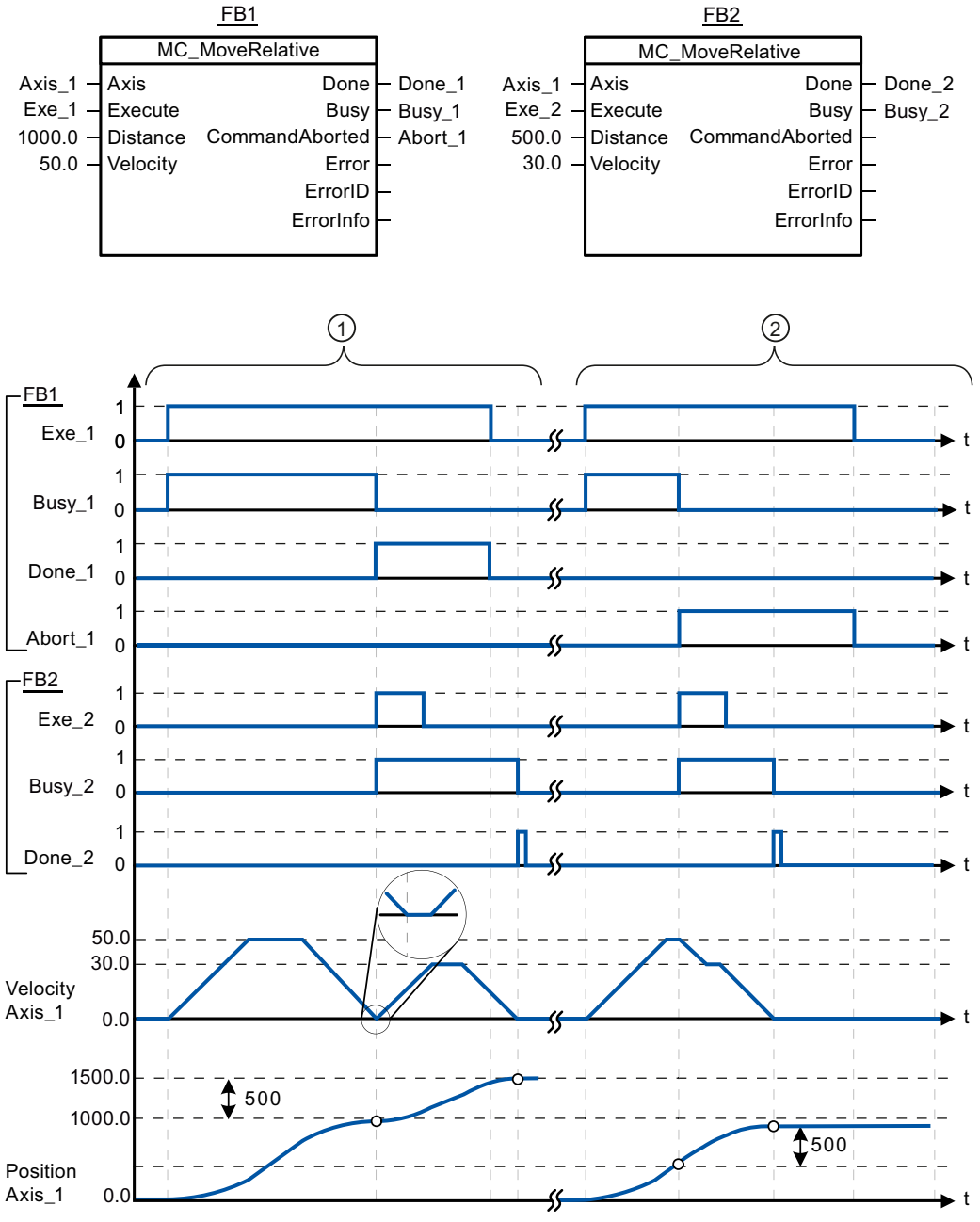
Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Positioning-Axis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Distance	INPUT	REAL	0.0	Travel distance for the positioning operation Limit values: $-1.0e^{12} \leq$ Distance $\leq 1.0e^{12}$	
Velocity	INPUT	REAL	10.0	Velocity of axis This velocity is not always reached on account of the configured acceleration and deceleration and the distance to be traveled. Limit values: Start/stop velocity ≤ Velocity ≤ maximum velocity	
Done	OUTPUT	BOOL	FALSE	TRUE	Target position reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)
MC_Power: Enable, disable axis as of V6 (Page 4094)
MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)
MC_Home: Home axes, set reference point as of V6 (Page 4100)
MC_Halt: Stop axis as of V6 (Page 4105)
MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)
MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)
MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)
MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)
MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)
MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)
MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

MC_MoveRelative: Function chart as of V6 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	The axis is moved by an MC_MoveRelative command by the distance ("Distance") 1000.0. When the axis reaches the target position, this is signaled via "Done_1". When "Done_1" = TRUE, another MC_MoveRelative command, with travel distance 500.0, is started. Because of the response times (e.g., cycle time of user program, etc.), the axis comes to a standstill briefly (see zoomed-in detail). When the axis reaches the new target position, this is signaled via "Done_2".
②	An active MC_MoveRelative command is aborted by another MC_MoveRelative command. The abort is signaled via "Abort_1". The axis is then moved at the new velocity by the new distance ("Distance") 500.0. When the new target position is reached, this is signaled via "Done_2".

MC_MoveVelocity (S7-1200)

MC_MoveVelocity: Move axis at set velocity as of V6 (S7-1200)

Description

Motion control instruction "MC_MoveVelocity" moves the axis constantly at the specified velocity.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.

Override response

MC_MoveVelocity can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveVelocity command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Velocity	INPUT	REAL	10.0	Velocity specification for axis motion Limit values: Start/stop velocity ≤ |Velocity ≤ maximum velocity (Velocity = 0.0 is permitted)	
Direction	INPUT	INT	0	Direction specification	
				0	Direction of rotation corresponds to the sign of the value in parameter "Velocity"
				1	Positive direction of rotation (The sign of the value in parameter "Velocity" is ignored)
				2	Negative direction of rotation (The sign of the value in parameter "Velocity" is ignored)
Current	INPUT	BOOL	FALSE	Maintain current velocity	
				FALSE	"Maintain current velocity" is deactivated. The values of parameters "Velocity" and "Direction" are used.
				TRUE	"Maintain current velocity" is activated. The values in parameters "Velocity" and "Direction" are not taken into account. When the axis resumes motion at the current velocity, the "InVelocity" parameter returns the value TRUE.
PositionControlled	INPUT	BOOL	TRUE	FALSE	Non position-controlled operation
				TRUE	Position-controlled operation
				The parameter applies as long as the MC_MoveVelocity command is executed. After this, the setting of MC_Power applies again. This parameter is ignored when a PTO axis is used.	
InVelocity	OUTPUT	BOOL	FALSE	TRUE	• "Current" = FALSE: The velocity specified in parameter "Velocity" was reached. • "Current" = TRUE: The axis travels at the current velocity at the start time.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".

Parameter	Declaration	Data type	Default value	Description
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"

Note

PLCopen Version 2.0

The Motion Control instruction "MC_MoveVelocity" is compatible to PLCopen Version 2.0 as of V4.

The "InVelocity" and "Busy" parameters show their status regardless of the "Execute" parameter until the command is overridden or stopped by an error. For more information, refer to the section " Tracking active commands.

Behavior with zero setpoint velocity (Velocity = 0.0)

An MC_MoveVelocity command with "Velocity" = 0.0 (such as an MC_Halt command) aborts active motion commands and stops the axis with the configured deceleration.

When the axis comes to a standstill, output parameter "InVelocity" indicates TRUE for at least one program cycle.

"Busy" indicates the value TRUE during the deceleration process and changes to FALSE together with "InVelocity". If parameter "Execute" = TRUE is set, "InVelocity" and "Busy" are latched.

When the "MC_MoveVelocity" command is started, status bit "SpeedCommand" is set in the technology object. Status bit "ConstantVelocity" is set upon axis standstill. Both bits are adapted to the new situation when a new motion command is started.

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

MC_Halt: Stop axis as of V6 (Page 4105)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)

MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)

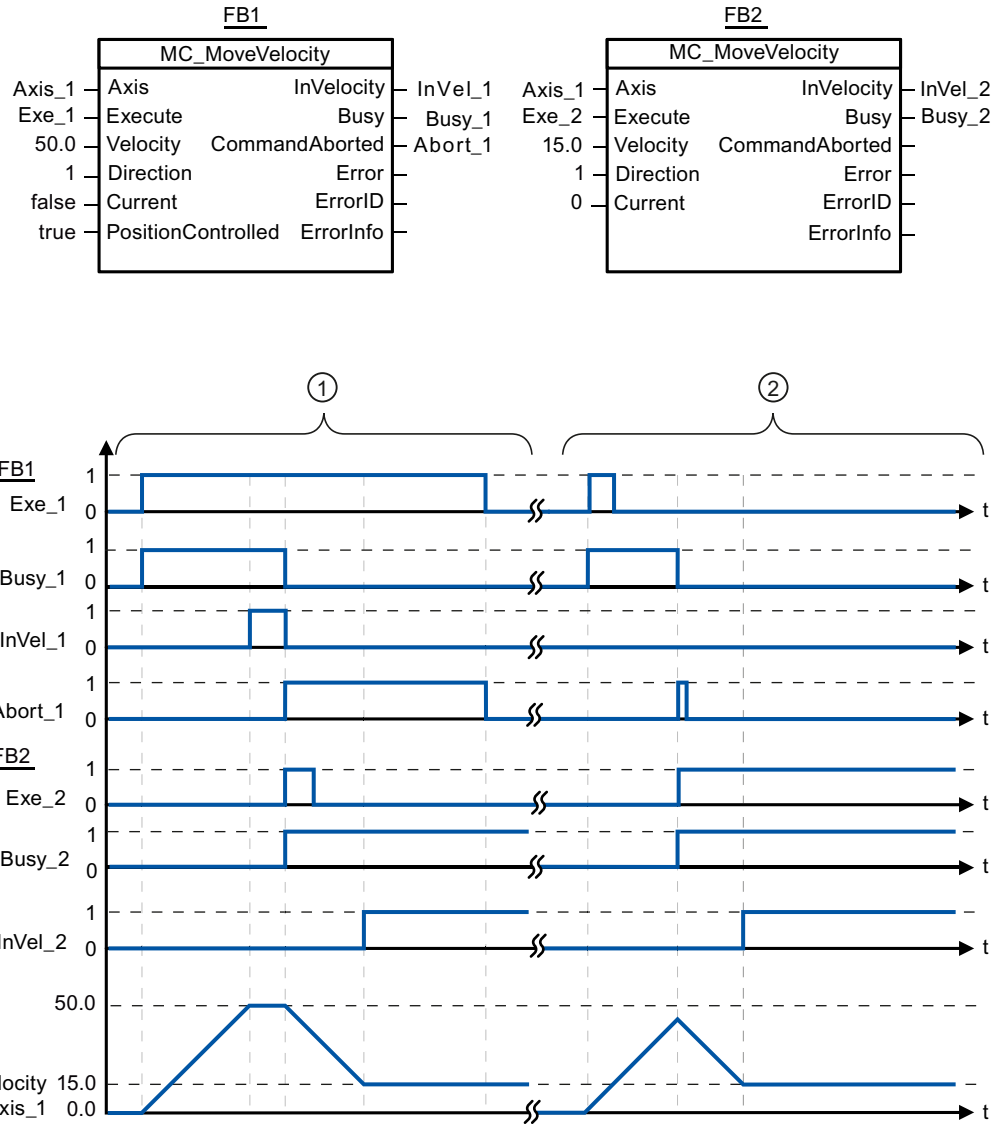
MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)

MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

MC_MoveVelocity: Function chart as of V6 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	An active MC_MoveVelocity command signals via "InVel_1" that its target velocity has been reached. It is then aborted by another MC_MoveVelocity command. The abort is signaled via "Abort_1". When the new target velocity 15.0 is reached, this is signaled via "InVel_2". The axis then continues moving at the new constant velocity.
②	An active MC_MoveVelocity command is aborted by another MC_MoveVelocity command prior to reaching its target velocity. The abort is signaled via "Abort_1". When the new target velocity 15.0 is reached, this is signaled via "InVel_2". The axis then continues moving at the new constant velocity.

MC_MoveJog (S7-1200)

MC_MoveJog: Move axis in jog mode as of V6 (S7-1200)

Description

Motion control instruction "MC_MoveJog" moves the axis constantly at the specified velocity in jog mode. You use this Motion Control instruction, for example, for testing and commissioning purposes.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_MoveJog command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveJog command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
JogForward	INPUT	BOOL	FALSE	As long as the parameter is TRUE, the axis moves in the positive direction at the velocity specified in parameter "Velocity".	
JogBackward	INPUT	BOOL	FALSE	As long as the parameter is TRUE, the axis moves in the negative direction at the velocity specified in parameter "Velocity".	
If both parameters are simultaneously TRUE, the axis stops with the configured deceleration. An error is indicated in parameters "Error", "ErrorID", and "ErrorInfo".					
Velocity	INPUT	REAL	10.0	Preset velocity for jog mode Limit values: Start/stop velocity ≤ velocity ≤ maximum velocity	
PositionControlled	INPUT	BOOL	TRUE	FALSE	Non position-controlled operation
				TRUE	Position-controlled operation
				The parameter applies as long as the MC_MoveJog command is executed. After this, the setting of MC_Power applies again. This parameter is ignored when a PTO axis is used.	
InVelocity	OUTPUT	BOOL	FALSE	TRUE	The velocity specified in parameter "Velocity" was reached.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

MC_Halt: Stop axis as of V6 (Page 4105)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)

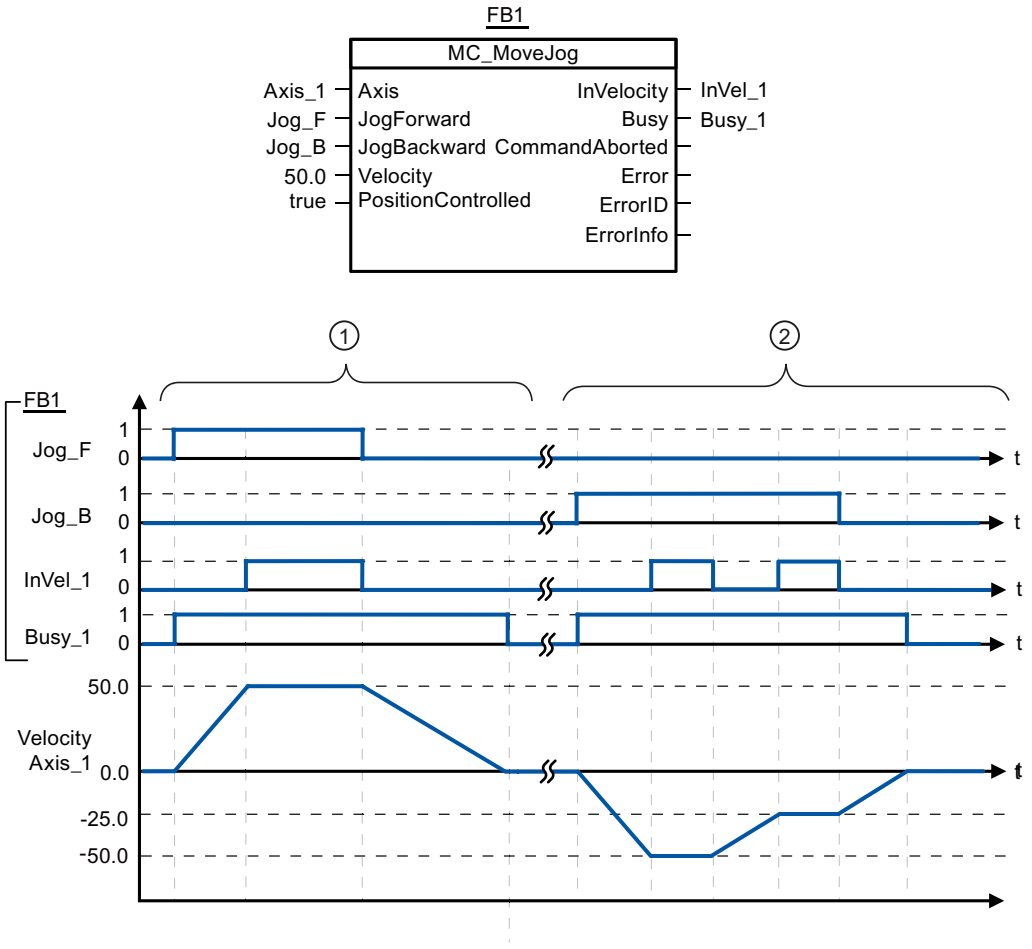
MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)
MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

MC_MoveJog: Function chart as of V6 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 5.0

①	The axis is moved in the positive direction in jog mode via "Jog_F". When the target velocity 50.0 is reached, this is signaled via "InVel_1". After "Jog_F" is reset, the axis is braked to a standstill.
②	The axis is moved in the negative direction in jog mode via "Jog_B". When the target velocity -50.0 is reached, this is signaled via "InVel_1". When "Jog_B" is set, the value at parameter "Velocity" changes to 25.0. "InVel_1" is reset and the axis is braked. When the new target velocity -25.0 is reached, this is signaled via "InVel_1". After "Jog_B" is reset, the axis is braked to a standstill.

MC_CommandTable (S7-1200)

MC_CommandTable: Run axis commands as motion sequence as of V6 (S7-1200)

Description

The Motion Control instruction "MC_CommandTable" combines multiple individual axis control commands in one movement sequence. "MC_CommandTable" is available for axes with drive connection via PTO (Pulse Train Output).

Requirements

- The positioning axis technology object has been inserted and correctly configured.
- The drive is connected via PTO (Pulse Train Output).
- The command table technology object has been inserted and correctly configured.
- The axis is enabled.

Override response

The MC_CommandTable command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_CommandTable command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The active Motion Control command is canceled by the start of the first "Positioning Relative", "Positioning Absolute", "Velocity set point" or "Halt" command.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
CommandTable	INPUT	TO_CommandTable	-	Command table technology object	
Execute	INPUT	BOOL	FALSE	Command table start with positive edge	
StartStep	INPUT	INT	1	Defines the step at which the execution of the command table should begin Limit values: 1 ≤ StartStep ≤ EndStep	
EndStep	INPUT	INT	32	Defines the step up to which the execution of command table should take place Limit values: StartStep ≤ EndStep ≤ 32	
Done	OUTPUT	BOOL	FALSE	TRUE	Command table has been successfully executed
Busy	OUTPU	BOOL	FALSE	TRUE	The command table is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	The command table was canceled by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command table. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	
CurrentStep	OUTPUT	INT	0	Step in command table currently being executed	
StepCode	OUTPUT	WORD	16#0000	User-defined numerical value / bit pattern of the step currently being executed	

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

MC_Halt: Stop axis as of V6 (Page 4105)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)

MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)

MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

MC_ChangeDynamic (S7-1200)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (S7-1200)

Description

Motion Control instruction "MC_ChangeDynamic" allows you to change the following settings of the axis:

- Change the ramp-up time (acceleration) value
- Change the ramp-down time (deceleration) value
- Change the emergency stop ramp-down time (emergency stop deceleration) value
- Change the smoothing time (jerk) value

For the effectiveness of the change, refer to the description of the tag.

Requirements

The positioning axis technology object has been configured correctly.

Override response

A MC_ChangeDynamic command cannot be aborted by any other Motion Control command.

A new MC_ChangeDynamic command does not abort any active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
ChangeRampUp	INPUT	BOOL	FALSE	TRUE	Change ramp-up time in line with input parameter "RampUpTime"
RampUpTime	INPUT	REAL	5.00	Time (in seconds) to accelerate axis from standstill to configured maximum velocity without jerk limit. The change will influence the tag <Axis name>. Config.DynamicDefaults.Acceleration. For the effectiveness of the change, refer to the description of this tag.	
ChangeRampDown	INPUT	BOOL	FALSE	TRUE	Change ramp-down time to correspond to input parameter "RampDownTime"
RampDownTime	INPUT	REAL	5.00	Time (in seconds) to decelerate axis from the configured maximum velocity to standstill without jerk limiter. The change will influence the tag <Axis name>. Config.DynamicDefaults.Deceleration . For the effectiveness of the change, refer to the description of this tag.	
ChangeEmergency	INPUT	BOOL	FALSE	TRUE	Change emergency stop ramp-down time in line with input parameter "EmergencyRampTime"

Parameter	Declaration	Data type	Default value	Description	
EmergencyRampTime	INPUT	REAL	2.00	Time (in seconds) to decelerate the axis from configured maximum velocity to standstill without jerk limiter in emergency stop mode. The change will influence the tag <Axis name>. Config.DynamicDefaults.EmergencyDeceleration . For the effectiveness of the change, refer to the description of this tag.	
ChangeJerkTime	INPUT	BOOL	FALSE	TRUE	Change smoothing time according to the input parameter "JerkTime"
JerkTime	INPUT	REAL	0.25	Smoothing time (in seconds) used for the axis acceleration and deceleration ramps The change will influence the tag <Axis name>. Config.DynamicDefaults.Jerk . For the effectiveness of the change, refer to the description of this tag.	
Done	OUTPUT	BOOL	FALSE	TRUE	The changed values have been written to the technology data block. The description of the tags will show when the change becomes effective.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	

Note

At the input parameters "RampUpTime", "RampDownTime", "EmergencyRampTime" and "JerkTime", values can be entered which exceed the admissible limits of the resulting parameters: "Acceleration", "Deceleration", "Emergency stop deceleration" and "Jerk".

Ensure that your inputs are within the valid range, taking into consideration the equations and limits in section "Dynamic".

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

MC_Halt: Stop axis as of V6 (Page 4105)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)

MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)

MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)

MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)

MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

MC_ReadParam (S7-1200)

MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (S7-1200)

Description

The Motion Control instruction "MC_ReadParam" enables continuous reading of motion data and status messages of an axis. The current value of the corresponding tags is determined at the start of the command.

The following motion data and status messages can be read:

- As of technology version V4:
  - Setpoint position of the axis
  - Setpoint and actual velocity of the axis
  - Current distance of axis from target position
  - Target position of the axis
- Additional as of technology version V5:
  - Actual position of the axis
  - Actual velocity of the axis
  - Current following error
  - Drive status
  - Encoder status
  - Status bits
  - Error bits

Requirements

The positioning axis technology object has been configured correctly.

Override response

A MC_ReadParam command cannot be aborted by any other Motion Control command.

A new MC_ReadParam command does not abort any active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Enable	INPUT	BOOL	FALSE	TRUE	Read the tag specified with the "Parameter" and store the value in the destination address specified with "Value".
				FALSE	Do not update assigned motion data
Parameter	INPUT	VARIANT (REAL)	-	VARIANT pointer to the value to be read. The following tags are permitted: • <Axis name>.Position • <Axis name>.Velocity • <Axis name>.ActualPosition • <Axis name>.ActualVelocity • <Axis name>.StatusPositioning.<Tag name> • <Axis name>.StatusDrive.<Tag name> • <Axis name>.StatusSensor.<Tag name> • <Axis name>.StatusBits.<Tag name> • <Axis name>.ErrorBits.<Tag name> The description of the tags named and the tag structures can be found in the Appendix AUTOHOTSPOT.	
Value	INOUT	VARIANT (REAL)	-	VARIANT pointer to the target tag or destination address to which the read value is to be written.	
Valid	OUTPUT	BOOL	FALSE	TRUE	The read value is valid.
				FALSE	The read value is invalid.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"	

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)
MC_Power: Enable, disable axis as of V6 (Page 4094)
MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)
MC_Home: Home axes, set reference point as of V6 (Page 4100)
MC_Halt: Stop axis as of V6 (Page 4105)
MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)
MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)
MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)
MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)
MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

MC_WriteParam (S7-1200)

MC_WriteParam: Write tag of positioning axis as of V6 (S7-1200)

Description

Motion Control instruction "MC_WriteParam" enables the writing of tags of the positioning axis technology object in the user program. In contrast to the value assignment of the tags in the user program, "MC_WriteParam" can also change values of read-only tags.

You can learn about the tags, the conditions under which they can be written and the time at which they take effect in the description of the technology object tags (Page 6328).

Requirements

- The positioning axis technology object has been configured correctly.
- To write tags that are read-only in the user program, the axis must be disabled.
- Tags whose change requires a restart cannot be written with "MC_WriteParam".

Override response

A MC_WriteParam command cannot be aborted by any other Motion Control command.

A new MC_WriteParam command does not abort any active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Parameter	INPUT	VARIANT (BOOL, INT, DINT, UDINT, REAL)	-	VARIANT pointer to the technology object tags (Page 6328) positioning axis (destination address) to be written	
Value	INPUT	VARIANT (BOOL, INT, DINT, UDINT, REAL)	-	VARIANT pointer to the value to be written (source address)	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Done	OUTPUT	BOOL	FALSE	TRUE	Value was written
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".

Parameter	Declaration	Data type	Default value	Description
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6304) for parameter "Error"
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6304) for parameter "ErrorID"

See also

List of ErrorIDs and ErrorInfos (technology objects as of V6) (Page 6304)

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)

MC_Home: Home axes, set reference point as of V6 (Page 4100)

MC_Halt: Stop axis as of V6 (Page 4105)

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)

MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)

MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)

MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)

Tags of the positioning axis technology object as of V6 (Page 6328)

S7-1200 Motion Control V4...5 (S7-1200)

MC_Power (S7-1200)

MC_Power: Enable, disable axis V4...5 (S7-1200)

Description

The Motion Control instruction "MC_Power" enables or disables an axis.

Requirements

- The positioning axis technology object has been configured correctly.
- There is no pending enable-inhibiting error.

Override response

Execution of "MC_Power" cannot be aborted by a Motion Control command.

Disabling the axis (input parameter "Enable" = FALSE) aborts all Motion Control commands for the associated technology object in accordance with the selected "StopMode".

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis	-	Axis technology object	
Enable	INPUT	BOOL	FALSE	TRUE	The axis is enabled.
				FALSE	All current jobs are interrupted in accordance with the "StopMode" configured. The axis is stopped and disabled.
StopMode	INPUT	INT	0	0	Emergency stop If a request to disable the axis is pending, the axis brakes at the configured emergency deceleration. The axis is disabled after reaching standstill.
				1	Immediate stop If a request to disable the axis is pending, this set-point zero is output and the axis is disabled. The axis is braked depending on the configuration in the drive, and is brought to a standstill. With drive connection via PTO (Pulse Train Output): When you disable the axis, the pulse output is stopped with a frequency-dependent deceleration: • Output frequency ≥ 100 Hz Deceleration: max. 30 ms • Output frequency < 100 Hz Deceleration: 30 ms up to max. 1.5 s at 2 Hz
				2	Emergency stop with jerk control If a request to disable the axis is pending, the axis brakes at the configured emergency deceleration. If the jerk control is activated, the configured jerk is taken into account. The axis is disabled after reaching standstill.

Parameter	Declaration	Data type	Default value	Description	
Status	OUTPUT	BOOL	FALSE	Status of axis enable	
				FALSE	The axis is disabled. The axis does not execute Motion Control commands and does not accept any new commands (exception: MC_Reset command). For drive connection via PTO (Pulse Train Output): The axis is not homed. Upon disabling, the status does not change to FALSE until the axis reaches a standstill.
				TRUE	The axis is enabled. The axis is ready to execute Motion Control commands. Upon axis enabling, the status does not change to TRUE until the signal "Drive ready" is pending. If the "Drive ready" drive interface was not configured in the axis configuration, the status changes to TRUE immediately.
Busy	OUTPUT	BOOL	FALSE	TRUE	"MC_Power" is active.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred in Motion Control instruction "MC_Power" or in the associated technology object. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Note

If the axis is switched off due to an error, it will be enabled again automatically after the error has been eliminated and acknowledged. This requires that the input parameter "Enable" has retained the value TRUE during this process.

Enabling an axis with configured drive interface

To enable the axis, follow these steps:

1. Check the requirements indicated above.
2. Initialize input parameter "StopMode" with the desired value. Set the input parameter "Enable" to TRUE.
   The enable output for "Drive enabled" changes to TRUE to enable the power to the drive.
   The CPU waits for the "Drive ready" signal of the drive.
   When the "Drive ready" signal is available at the configured ready input of the CPU, the axis is enabled. The output parameter "Status" and the tag of the technology object <Axis name>.StatusBits.Enable indicate the value TRUE.

Enabling an axis without configured drive interface

To enable the axis, follow these steps:

1. Check the requirements indicated above.
2. Initialize input parameter "StopMode" with the desired value. Set the input parameter "Enable" to TRUE. The axis is enabled. The output parameter "Status" and the tag of the technology object <Axis name>.StatusBits.Enable indicate the value TRUE.

Disabling an axis

To disable an axis, you can follow the steps described below:

1. Bring the axis to a standstill.
   You can identify when the axis is at a standstill in the tag of the technology object <Axis name>.StatusBits.StandStill.
2. Set input parameter "Enable" to FALSE after standstill is reached.
3. If output parameters "Busy" and "Status" and tag of technology object <Axis name>.StatusBits.Enable indicate the value FALSE, disabling of the axis is complete.

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_Power: Function chart V4...5 (Page 4133)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

MC_Home: Home axis, set home position V4...5 (Page 4136)

MC_Halt: Stop axis V4...5 (Page 4140)

MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)

MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)

MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)

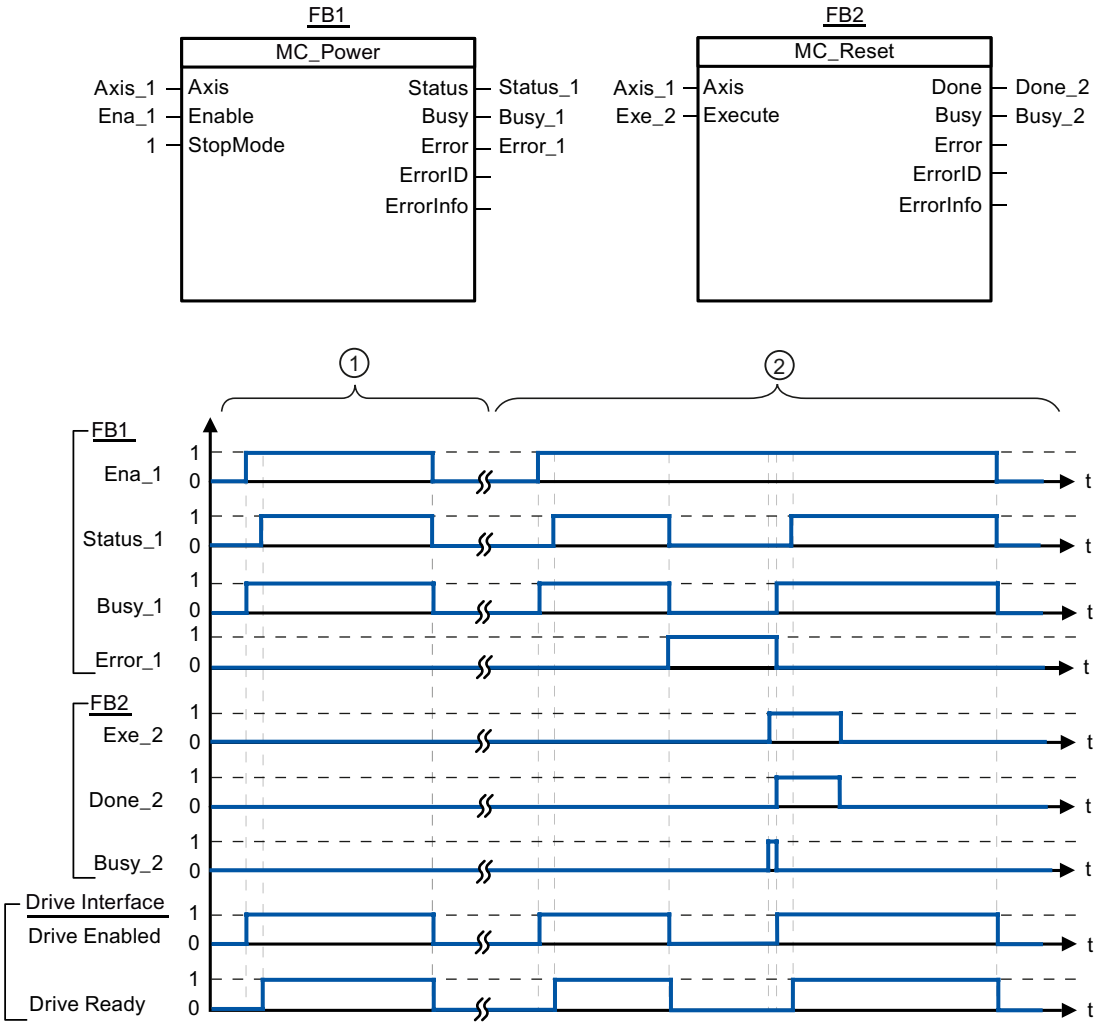
MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)

MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)

S7-1200 Motion Control V1...3 (Page 4165)

MC_Power: Function chart V4...5 (S7-1200)

Function chart



①	An axis is enabled and then disabled again. When the drive has signaled "Drive ready" back to the CPU, the successful enable can be read out via "Status_1".
②	Following an axis enable, an error has occurred that caused the axis to be disabled. The error is eliminated and acknowledged with "MC_Reset". The axis is then enabled again.

See also

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset (S7-1200)

MC_Reset: Acknowledge fault, restart technology object V4...5 (S7-1200)

Description

Motion Control instruction "MC_Reset" can be used to acknowledge "Operating error with axis stop" and "Configuration error". The errors that require acknowledgment can be found in the "List of ErrorIDs and ErrorInfos" under "Remedy".

The axis configuration can be downloaded to the work memory after a download in RUN mode.

Requirements

- The positioning axis technology object has been configured correctly.
- The cause of a pending configuration error requiring acknowledgment has been eliminated (for example, acceleration in positioning axis technology object has been changed to a valid value).

Override response

The MC_Reset command cannot be aborted by any other Motion Control command.

The new MC_Reset command does not abort any other active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Restart	INPUT	BOOL	FALSE	TRUE	Download the axis configuration from the load memory to the work memory. The command can only be executed when the axis is disabled. Refer to the notes on Download to the CPU (Page 6260).
				FALSE	Acknowledges pending errors
Done	OUTPUT	BOOL	FALSE	TRUE	Error has been acknowledged.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Acknowledging an error requiring acknowledgment with MC_Reset

To acknowledge an error, follow these steps:

1. Check the requirements indicated above.
2. Start the acknowledgment of the error with a rising edge at input parameter "Execute".
3. If output parameter "Done" indicates the value TRUE and tag of technology object <Axis name>.StatusBits.Error the value FALSE, the error has been acknowledged.

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

Download to CPU (Page 6260)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Home: Home axis, set home position V4...5 (Page 4136)

MC_Halt: Stop axis V4...5 (Page 4140)

MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)

MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)

MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)

MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)

MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)

S7-1200 Motion Control V1...3 (Page 4165)

MC_Home (S7-1200)

MC_Home: Home axis, set home position V4...5 (S7-1200)

Description

Motion Control instruction "MC_Home" is used to match the axis coordinates to the real, physical drive position. Homing is required for absolute positioning of the axis. The following types of homing can be executed:

- Active homing (Mode = 3)
  The homing procedure is executed automatically.
- Passive homing (Mode = 2)
  During passive homing, the "MC_Home" Motion Control instruction does not carry out any homing motion. The traversing motion required for this must be implemented by the user via other Motion Control instructions. When the homing switch is detected, the axis is homed.
- Direct homing absolute (Mode = 0)
  The current axis position is set to the value of parameter "Position".
- Direct homing relative (Mode = 1)
  The current axis position is offset by the value of parameter "Position".
- Absolute encoder adjustment relative (Mode = 6)
  The current axis position is offset by the value of parameter "Position".
- Absolute encoder adjustment absolute (Mode = 7)
  The current axis position is set to the value of parameter "Position".

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.
- No MC_CommandTable command may be active upon start with Mode = 0, 1, or 2.

Override response

The override response depends on the selected mode:

Mode = 0, 1, 6, 7

The MC_Home command cannot be aborted by any other Motion Control command.

The MC_Home command does not abort any active Motion Control commands. Position-related motion commands are resumed after homing according to the new homing position (value at input parameter: "Position").

Mode = 2

The MC_Home command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 2, 3

The new MC_Home command aborts the following active Motion Control command:

- MC_Home command Mode = 2

Position-related motion commands are resumed after homing according to the new homing position (value at input parameter: "Position").

Mode = 3

The MC_Home command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_Home command aborts the following active Motion Control commands:

- MC_Home command Mode = 2, 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description
Axis	INPUT	TO_Axis	-	Axis technology object
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge
Position	INPUT	REAL	0.0	• Mode = 0, 2, and 3 Absolute position of axis after completion of the homing operation • Mode = 1 Correction value for the current axis position Limit values: $-1.0e^{12} \leq$ Position $\leq 1.0e^{12}$

Parameter	Declaration	Data type	Default value	Description	
Mode	INPUT	INT	0	Homing mode	
				0	Direct homing (absolute) New axis position is the position value of parameter "Position".
				1	Direct homing (relative) New axis position is the current axis position + position value of parameter "Position".
				2	Passive homing Homing according to the axis configuration. Following homing, the value of parameter "Position" is set as the new axis position.
				3	Active homing Homing procedure in accordance with the axis configuration. Following homing, the value of parameter "Position" is set as the new axis position.
				6	Absolute encoder adjustment (relative) The current axis position is offset by the value of parameter "Position". The calculated absolute value offset is stored retentively in the CPU. (<AxisName>.StatusSensor.AbsEncoderOffset)
				7	Absolute encoder adjustment (absolute) The current axis position is set to the value of parameter "Position". The calculated absolute value offset is stored retentively in the CPU. (<AxisName>.StatusSensor.AbsEncoderOffset)
Done	OUTPUT	BOOL	FALSE	TRUE	Command completed
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Resetting the "Homed" status

The "Homed" status of a technology object (<Axis name>.StatusBits.HomingDone) is reset under the following conditions:

- **Drive connection via PTO (Pulse Train Output):**
  - Start an "MC_Home" command for active homing
    (After successful completion of the homing operation, the "Homed" status is set again.)
  - Disabling of axis by the "MC_Power" Motion Control instruction
  - Changeover between automatic mode and manual control
  - After POWER OFF -> POWER ON of the CPU
  - After CPU restart (RUN-STOP -> STOP-RUN)
- **Technology objects with incremental actual values:**
  - Start an "MC_Home" command for active homing
    (After successful completion of the homing operation, the "Homed" status is set again.)
  - Error in the encoder system, or encoder failure
  - Restart of the technology object
  - After POWER OFF → POWER ON of the CPU
  - Memory reset
  - Modification of the encoder configuration
- **Technology objects with absolute actual values:**
  - Errors in the sensor system/encoder failure
  - Replacement of the CPU
  - Modification of the encoder configuration
  - Restoration of the CPU factory settings
  - Transfer of a different project to the controller

Homing an axis

To home the axis, follow these stops:

1. Check the requirements indicated above.
2. Provide the necessary input parameters with values and start the homing operation with a rising edge at input parameter "Execute".
3. If output parameter "Done" and technology object tag <Axis name>.StatusBits.HomingDone indicate the value TRUE, homing is complete.

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)
MC_Halt: Stop axis V4...5 (Page 4140)
MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)
MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)
MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)
MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)
MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)
MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)
MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)
MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)
S7-1200 Motion Control V1...3 (Page 4165)

MC_Halt (S7-1200)

MC_Halt: Stop axis V4...5 (S7-1200)

Description

The "MC_Halt" Motion Control instruction stops all movements and brings the axis to a standstill with the configured deceleration. The standstill position is not defined.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_Halt command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_Halt command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Done	OUTPUT	BOOL	FALSE	TRUE	Zero velocity reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_Halt: Function chart V4...5 (Page 4142)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

MC_Home: Home axis, set home position V4...5 (Page 4136)

MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)

MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)

MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)

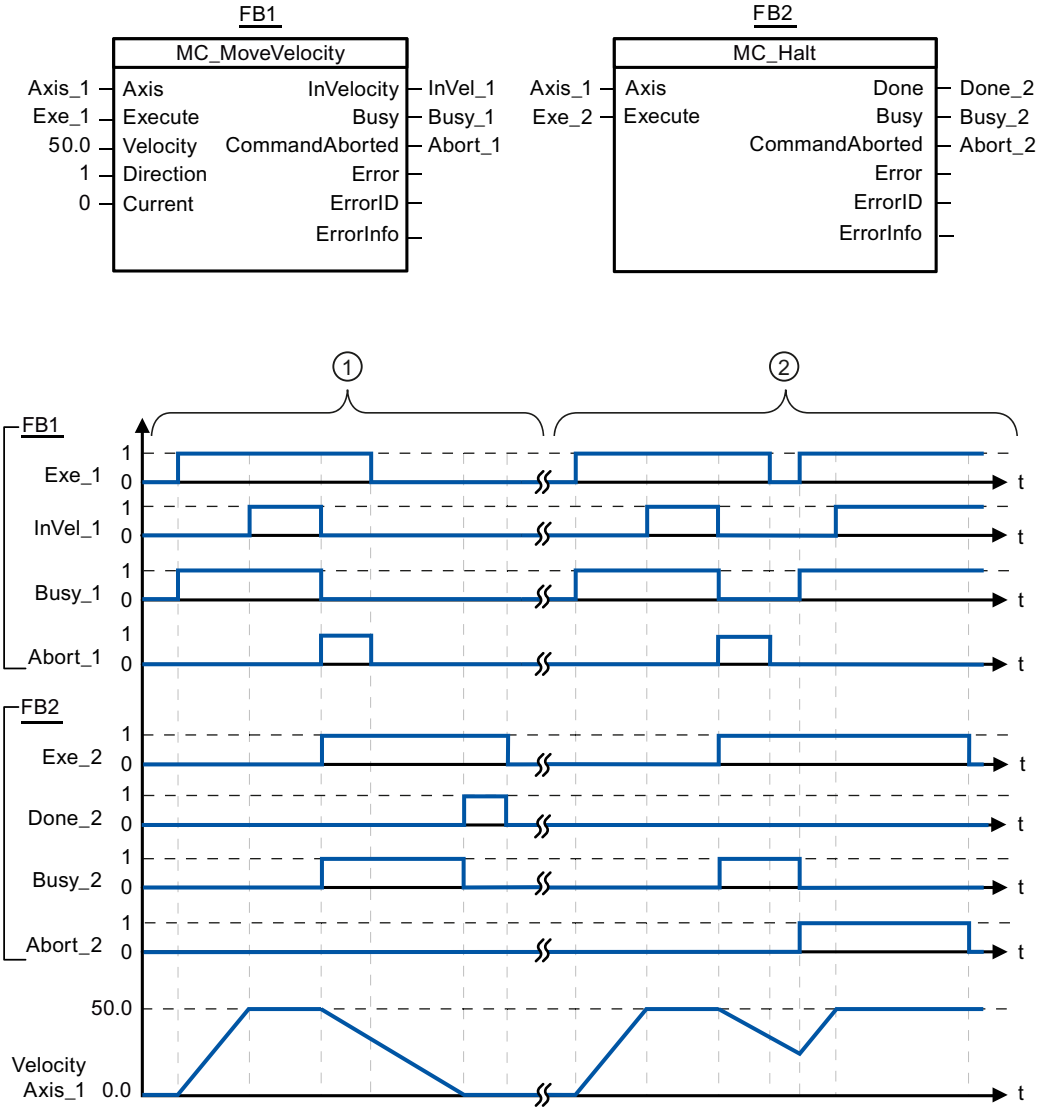
MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)

MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)

MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)
S7-1200 Motion Control V1...3 (Page 4165)

MC_Halt: Function chart V4...5 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 5.0

①	The axis is braked by an MC_Halt command until it comes to a standstill. The axis standstill is signaled via "Done_2".
②	While an MC_Halt command is braking the axis, this command is aborted by another motion command. The abort is signaled via "Abort_2".

See also

MC_Halt: Stop axis V4...5 (Page 4140)

MC_MoveAbsolute (S7-1200)

MC_MoveAbsolute: Absolute positioning of axis V4...5 (S7-1200)

Description

The "MC_MoveAbsolute" Motion Control instruction starts an axis positioning motion to move it to an absolute position.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.
- The axis is homed.

Override response

The MC_MoveAbsolute command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveAbsolute command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command

- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Positioning-Axis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Position	INPUT	REAL	0.0	Absolute target position Limit values: $-1.0e^{12} \leq$ Position $\leq 1.0e^{12}$	
Velocity	INPUT	REAL	10.0	Velocity of axis This velocity is not always reached on account of the configured acceleration and deceleration and the target position to be approached. Limit values: Start/stop velocity ≤ Velocity ≤ maximum velocity	
Done	OUTPUT	BOOL	FALSE	TRUE	Absolute target position reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_MoveAbsolute: Function chart V4...5 (Page 4145)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

MC_Home: Home axis, set home position V4...5 (Page 4136)

MC_Halt: Stop axis V4...5 (Page 4140)

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)

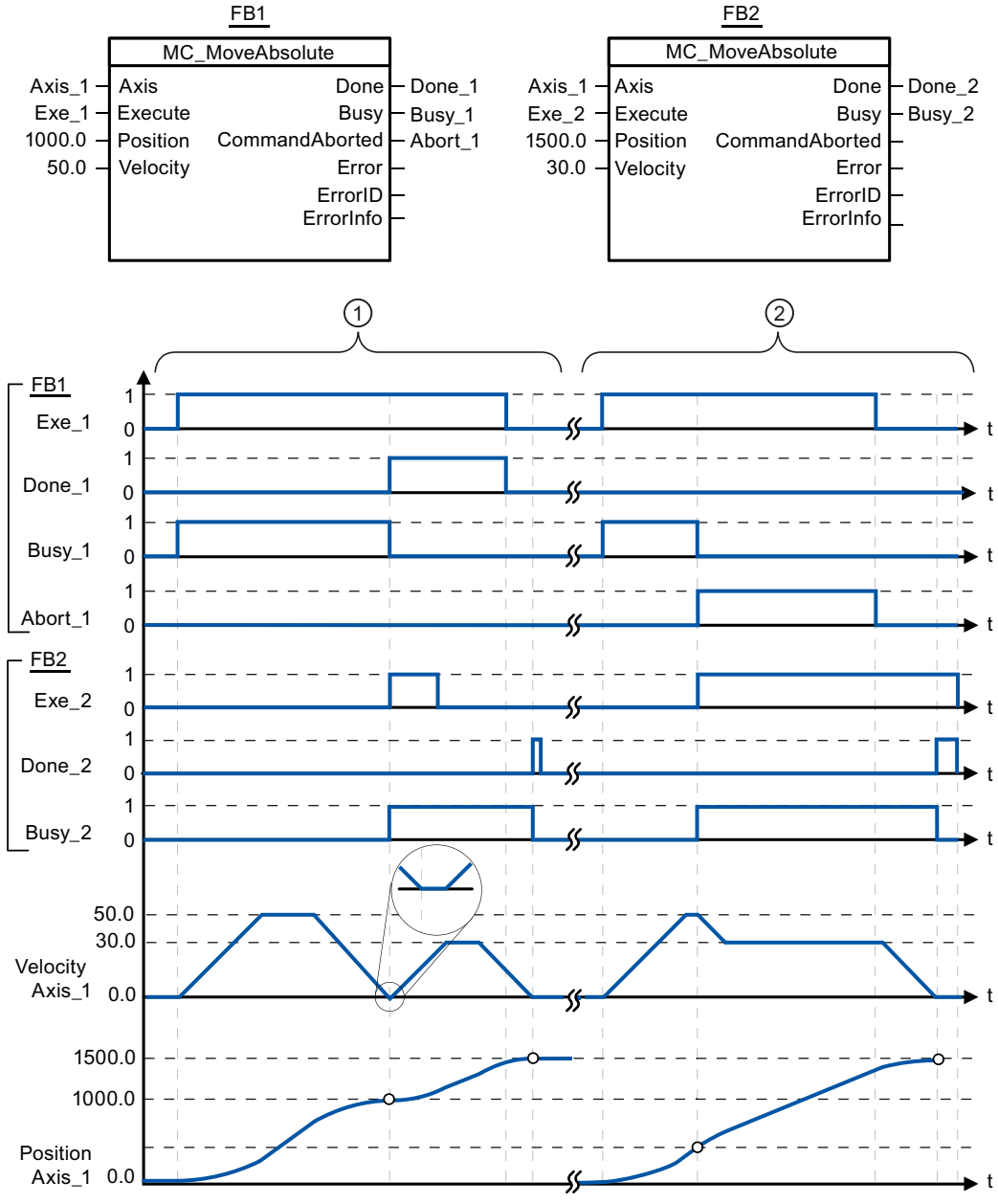
MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)

MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)
MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)
MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)
S7-1200 Motion Control V1...3 (Page 4165)

MC_MoveAbsolute: Function chart V4...5 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	An axis is moved to absolute position 1000.0 with an MC_MoveAbsolute command. When the axis reaches the target position, this is signaled via "Done_1". When "Done_1" = TRUE, another MC_MoveAbsolute command, with target position 1500.0, is started. Because of the response times (e.g., cycle time of user program, etc.), the axis comes to a standstill briefly (see zoomed-in detail). When the axis reaches the new target position, this is signaled via "Done_2".
②	An active MC_MoveAbsolute command is aborted by another MC_MoveAbsolute command. The abort is signaled via "Abort_1". The axis is then moved at the new velocity to the new target position 1500.0. When the new target position is reached, this is signaled via "Done_2".

See also

MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)

MC_MoveRelative (S7-1200)

MC_MoveRelative: Relative positioning of axis V4...5 (S7-1200)

Description

The "MC_MoveRelative" Motion Control instruction starts a positioning motion relative to the start position.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_MoveRelative command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveRelative command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Positioning-Axis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Distance	INPUT	REAL	0.0	Travel distance for the positioning operation Limit values: $-1.0e^{12} \leq$ Distance $\leq 1.0e^{12}$	
Velocity	INPUT	REAL	10.0	Velocity of axis This velocity is not always reached on account of the configured acceleration and deceleration and the distance to be traveled. Limit values: Start/stop velocity ≤ Velocity ≤ maximum velocity	
Done	OUTPUT	BOOL	FALSE	TRUE	Target position reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_MoveRelative: Function chart V4...5 (Page 4149)

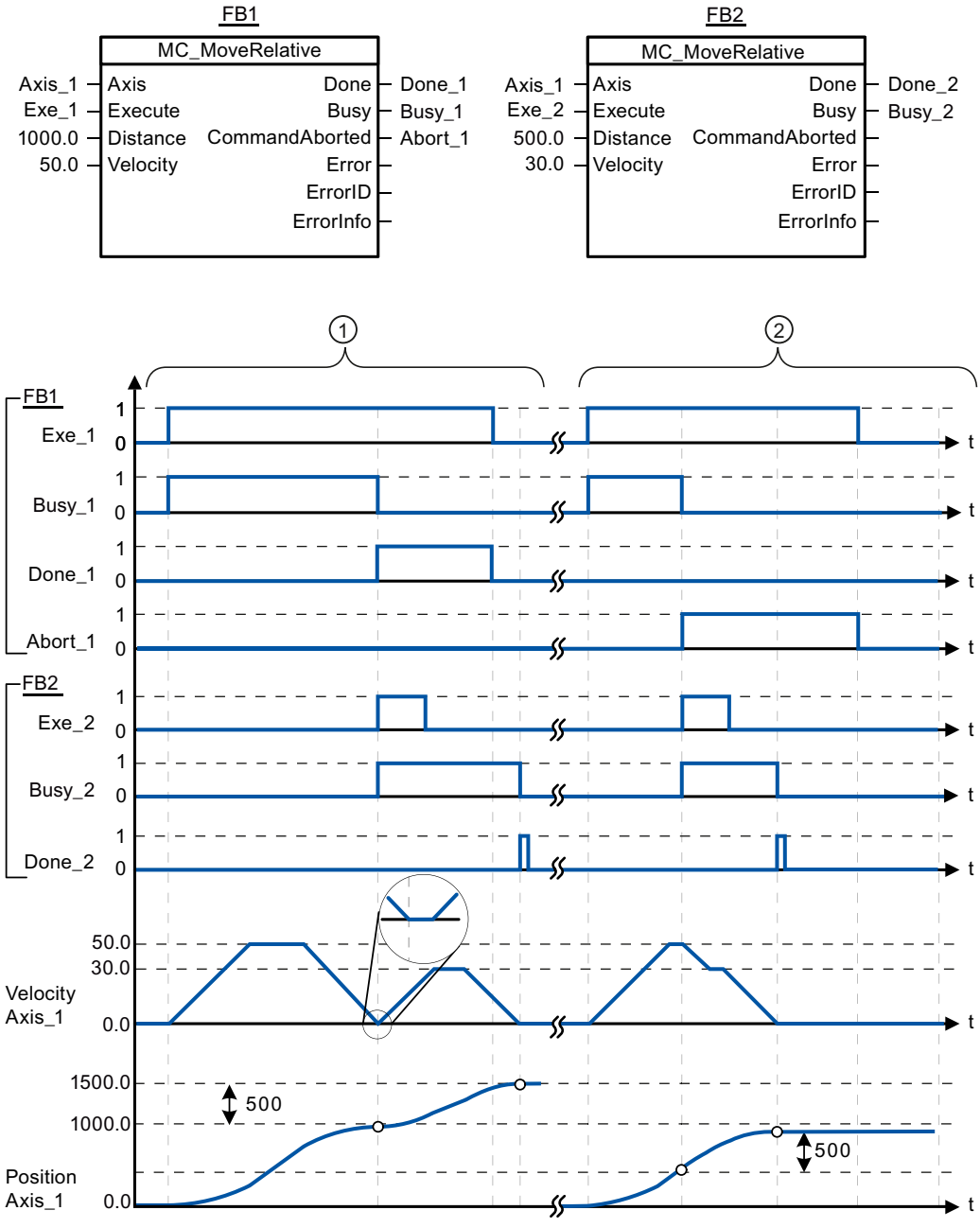
MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

MC_Home: Home axis, set home position V4...5 (Page 4136)
MC_Halt: Stop axis V4...5 (Page 4140)
MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)
MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)
MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)
MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)
MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)
MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)
MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)
S7-1200 Motion Control V1...3 (Page 4165)

MC_MoveRelative: Function chart V4...5 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	The axis is moved by an MC_MoveRelative command by the distance ("Distance") 1000.0. When the axis reaches the target position, this is signaled via "Done_1". When "Done_1" = TRUE, another MC_MoveRelative command, with travel distance 500.0, is started. Because of the response times (e.g., cycle time of user program, etc.), the axis comes to a standstill briefly (see zoomed-in detail). When the axis reaches the new target position, this is signaled via "Done_2".
②	An active MC_MoveRelative command is aborted by another MC_MoveRelative command. The abort is signaled via "Abort_1". The axis is then moved at the new velocity by the new distance ("Distance") 500.0. When the new target position is reached, this is signaled via "Done_2".

See also

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveVelocity (S7-1200)

MC_MoveVelocity: Move axis at set velocity V4...5 (S7-1200)

Description

Motion control instruction "MC_MoveVelocity" moves the axis constantly at the specified velocity.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.

Override response

MC_MoveVelocity can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveVelocity command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command

- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Velocity	INPUT	REAL	10.0	Velocity specification for axis motion Limit values: Start/stop velocity ≤ |Velocity ≤ maximum velocity (Velocity = 0.0 is permitted)	
Direction	INPUT	INT	0	Direction specification	
				0	Direction of rotation corresponds to the sign of the value in parameter "Velocity"
				1	Positive direction of rotation (The sign of the value in parameter "Velocity" is ignored)
				2	Negative direction of rotation (The sign of the value in parameter "Velocity" is ignored)
Current	INPUT	BOOL	FALSE	Maintain current velocity	
				FALSE	"Maintain current velocity" is deactivated. The values of parameters "Velocity" and "Direction" are used.
				TRUE	"Maintain current velocity" is activated. The values in parameters "Velocity" and "Direction" are not taken into account. When the axis resumes motion at the current velocity, the "InVelocity" parameter returns the value TRUE.
InVelocity	OUTPUT	BOOL	FALSE	TRUE	• "Current" = FALSE: The velocity specified in parameter "Velocity" was reached. • "Current" = TRUE: The axis travels at the current velocity at the start time.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Note

PLCopen Version 2.0

The Motion Control instruction "MC_MoveVelocity" is compatible to PLCopen Version 2.0 as of V4.

The "InVelocity" and "Busy" parameters show their status regardless of the "Execute" parameter until the command is overridden or stopped by an error. For more information, refer to the section " Tracking active commands (Page 6276).

Behavior with zero setpoint velocity (Velocity = 0.0)

An MC_MoveVelocity command with "Velocity" = 0.0 (such as an MC_Halt command) aborts active motion commands and stops the axis with the configured deceleration.

When the axis comes to a standstill, output parameter "InVelocity" indicates TRUE for at least one program cycle.

"Busy" indicates the value TRUE during the deceleration process and changes to FALSE together with "InVelocity". If parameter "Execute" = TRUE is set, "InVelocity" and "Busy" are latched.

When the "MC_MoveVelocity" command is started, status bit "SpeedCommand" is set in the technology object. Status bit "ConstantVelocity" is set upon axis standstill. Both bits are adapted to the new situation when a new motion command is started.

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_MoveVelocity: Function chart V4...5 (Page 4153)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

MC_Home: Home axis, set home position V4...5 (Page 4136)

MC_Halt: Stop axis V4...5 (Page 4140)

MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)

MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)

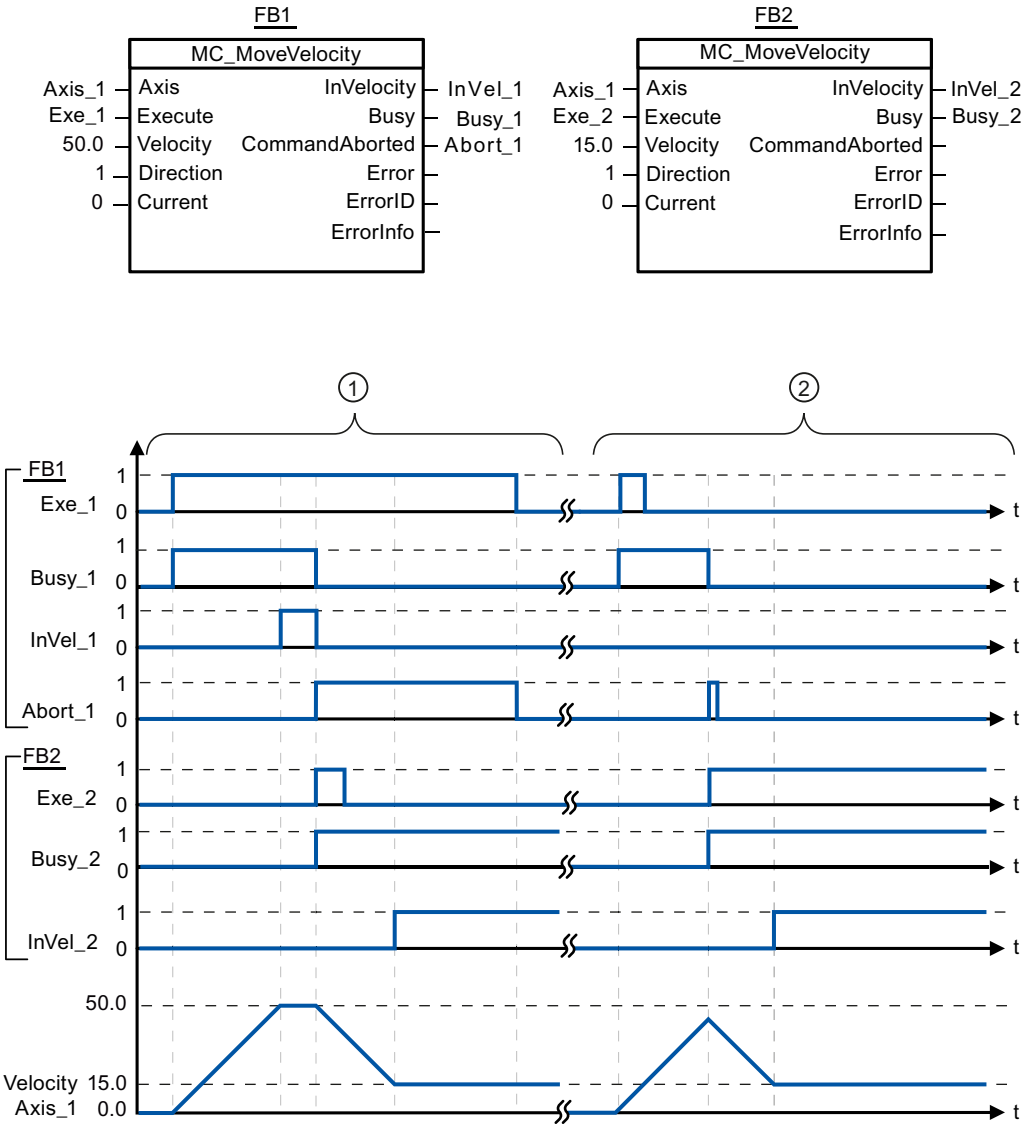
MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)

MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)

S7-1200 Motion Control V1...3 (Page 4165)

MC_MoveVelocity: Function chart V4...5 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	An active MC_MoveVelocity command signals via "InVel_1" that its target velocity has been reached. It is then aborted by another MC_MoveVelocity command. The abort is signaled via "Abort_1". When the new target velocity 15.0 is reached, this is signaled via "InVel_2". The axis then continues moving at the new constant velocity.
②	An active MC_MoveVelocity command is aborted by another MC_MoveVelocity command prior to reaching its target velocity. The abort is signaled via "Abort_1". When the new target velocity 15.0 is reached, this is signaled via "InVel_2". The axis then continues moving at the new constant velocity.

See also

MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)

MC_MoveJog (S7-1200)

MC_MoveJog: Move axis in jog mode V4...5 (S7-1200)

Description

Motion control instruction "MC_MoveJog" moves the axis constantly at the specified velocity in jog mode. You use this Motion Control instruction, for example, for testing and commissioning purposes.

Requirements

- The positioning axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_MoveJog command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveJog command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
JogForward	INPUT	BOOL	FALSE	As long as the parameter is TRUE, the axis moves in the positive direction at the velocity specified in parameter "Velocity".	
JogBackward	INPUT	BOOL	FALSE	As long as the parameter is TRUE, the axis moves in the negative direction at the velocity specified in parameter "Velocity".	
If both parameters are simultaneously TRUE, the axis stops with the configured deceleration. An error is indicated in parameters "Error", "ErrorID", and "ErrorInfo".					
Velocity	INPUT	REAL	10.0	Preset velocity for jog mode Limit values: Start/stop velocity ≤ velocity ≤ maximum velocity	
InVelocity	OUTPUT	BOOL	FALSE	TRUE	The velocity specified in parameter "Velocity" was reached.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_MoveJog: Function chart V4...5 (Page 4156)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

MC_Home: Home axis, set home position V4...5 (Page 4136)

MC_Halt: Stop axis V4...5 (Page 4140)

MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)

MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)

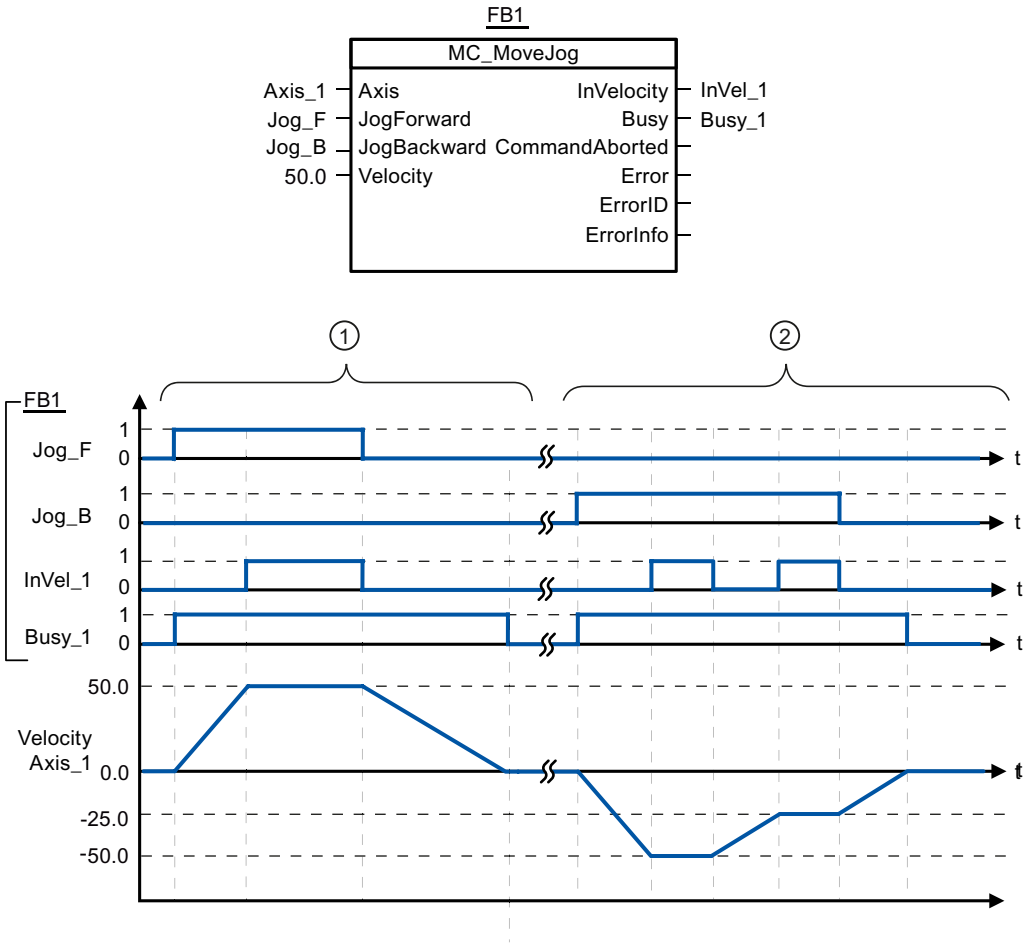
MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)

MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)

MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)
S7-1200 Motion Control V1...3 (Page 4165)

MC_MoveJog: Function chart V4...5 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 5.0

①	The axis is moved in the positive direction in jog mode via "Jog_F". When the target velocity 50.0 is reached, this is signaled via "InVel_1". After "Jog_F" is reset, the axis is braked to a standstill.
②	The axis is moved in the negative direction in jog mode via "Jog_B". When the target velocity -50.0 is reached, this is signaled via "InVel_1". When "Jog_B" is set, the value at parameter "Velocity" changes to 25.0. "InVel_1" is reset and the axis is braked. When the new target velocity -25.0 is reached, this is signaled via "InVel_1". After "Jog_B" is reset, the axis is braked to a standstill.

See also

MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)

MC_CommandTable (S7-1200)

MC_CommandTable: Run axis commands as motion sequence V4...5 (S7-1200)

Description

The Motion Control instruction "MC_CommandTable" combines multiple individual axis control commands in one movement sequence. "MC_CommandTable" is available for axes with drive connection via PTO (Pulse Train Output).

Requirements

- The positioning axis technology object has been inserted and correctly configured.
- The drive is connected via PTO (Pulse Train Output).
- The command table technology object has been inserted and correctly configured.
- The axis is enabled.

Override response

The MC_CommandTable command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_CommandTable command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The active Motion Control command is canceled by the start of the first "Positioning Relative", "Positioning Absolute", "Velocity set point" or "Halt" command.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
CommandTable	INPUT	TO_CommandTable	-	Command table technology object	
Execute	INPUT	BOOL	FALSE	Command table start with positive edge	
StartStep	INPUT	INT	1	Defines the step at which the execution of the command table should begin Limit values: 1 ≤ StartStep ≤ EndStep	
EndStep	INPUT	INT	32	Defines the step up to which the execution of command table should take place Limit values: StartStep ≤ EndStep ≤ 32	
Done	OUTPUT	BOOL	FALSE	TRUE	Command table has been successfully executed
Busy	OUTPU	BOOL	FALSE	TRUE	The command table is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	The command table was canceled by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command table. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	
CurrentStep	OUTPUT	INT	0	Step in command table currently being executed	
StepCode	OUTPUT	WORD	16#0000	User-defined numerical value / bit pattern of the step currently being executed	

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

MC_Home: Home axis, set home position V4...5 (Page 4136)

MC_Halt: Stop axis V4...5 (Page 4140)

MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)

MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)
MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)
MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)
S7-1200 Motion Control V1...3 (Page 4165)

MC_ChangeDynamic (S7-1200)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (S7-1200)

Description

Motion Control instruction "MC_ChangeDynamic" allows you to change the following settings of the axis:

- Change the ramp-up time (acceleration) value
- Change the ramp-down time (deceleration) value
- Change the emergency stop ramp-down time (emergency stop deceleration) value
- Change the smoothing time (jerk) value

For the effectiveness of the change, refer to the description of the tag (Page 6436).

Requirements

The positioning axis technology object has been configured correctly.

Override response

A MC_ChangeDynamic command cannot be aborted by any other Motion Control command.

A new MC_ChangeDynamic command does not abort any active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_SpeedAxis	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
ChangeRampUp	INPUT	BOOL	FALSE	TRUE	Change ramp-up time in line with input parameter "RampUpTime"
RampUpTime	INPUT	REAL	5.00	Time (in seconds) to accelerate axis from standstill to configured maximum velocity without jerk limit. The change will influence the tag <Axis name>. Config.DynamicDefaults.Acceleration. For the effectiveness of the change, refer to the description of this tag.	
ChangeRampDown	INPUT	BOOL	FALSE	TRUE	Change ramp-down time to correspond to input parameter "RampDownTime"

Parameter	Declaration	Data type	Default value	Description	
Ramp-DownTime	INPUT	REAL	5.00	Time (in seconds) to decelerate axis from the configured maximum velocity to standstill without jerk limiter. The change will influence the tag <Axis name>. Config.DynamicDefaults.Deceleration . For the effectiveness of the change, refer to the description of this tag.	
ChangeEmergency	INPUT	BOOL	FALSE	TRUE	Change emergency stop ramp-down time in line with input parameter "EmergencyRampTime"
EmergencyRampTime	INPUT	REAL	2.00	Time (in seconds) to decelerate the axis from configured maximum velocity to standstill without jerk limiter in emergency stop mode. The change will influence the tag <Axis name>. Config.DynamicDefaults.EmergencyDeceleration . For the effectiveness of the change, refer to the description of this tag.	
ChangeJerkTime	INPUT	BOOL	FALSE	TRUE	Change smoothing time according to the input parameter "JerkTime"
JerkTime	INPUT	REAL	0.25	Smoothing time (in seconds) used for the axis acceleration and deceleration ramps The change will influence the tag <Axis name>. Config.DynamicDefaults.Jerk . For the effectiveness of the change, refer to the description of this tag.	
Done	OUTPUT	BOOL	FALSE	TRUE	The changed values have been written to the technology data block. The description of the tags will show when the change becomes effective.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Note

At the input parameters "RampUpTime", "RampDownTime", "EmergencyRampTime" and "JerkTime", values can be entered which exceed the admissible limits of the resulting parameters: "Acceleration", "Deceleration", "Emergency stop deceleration" and "Jerk".

Ensure that your inputs are within the valid range, taking into consideration the equations and limits in section "Dynamic (Page 6203)".

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

Changing the configuration of dynamics in the user program (Page 6209)

Changing the homing configuration in the user program (Page 6217)

Tags of the positioning axis technology object V4...5 (Page 6436)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)
MC_Home: Home axis, set home position V4...5 (Page 4136)
MC_Halt: Stop axis V4...5 (Page 4140)
MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)
MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)
MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)
MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)
MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)
MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)
MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)
S7-1200 Motion Control V1...3 (Page 4165)

MC_ReadParam (S7-1200)

MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (S7-1200)

Description

The Motion Control instruction "MC_ReadParam" enables continuous reading of motion data and status messages of an axis. The current value of the corresponding tags is determined at the start of the command.

The following motion data and status messages can be read:

- As of technology version V4:
  - Setpoint position of the axis
  - Setpoint and actual velocity of the axis
  - Current distance of axis from target position
  - Target position of the axis
- Additional as of technology version V5:
  - Actual position of the axis
  - Actual velocity of the axis
  - Current following error
  - Drive status
  - Encoder status
  - Status bits
  - Error bits

Requirements

The positioning axis technology object has been configured correctly.

Override response

A MC_ReadParam command cannot be aborted by any other Motion Control command.

A new MC_ReadParam command does not abort any active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Enable	INPUT	BOOL	FALSE	TRUE	Read the tag specified with the "Parameter" and store the value in the destination address specified with "Value".
				FALSE	Do not update assigned motion data
Parameter	INPUT	VARIANT (REAL)	-	VARIANT pointer to the value to be read. The following tags are permitted: • <Axis name>.Position • <Axis name>.Velocity • <Axis name>.ActualPosition • <Axis name>.ActualVelocity • <Axis name>.StatusPositioning.<Tag name> • <Axis name>.StatusDrive.<Tag name> • <Axis name>.StatusSensor.<Tag name> • <Axis name>.StatusBits.<Tag name> • <Axis name>.ErrorBits.<Tag name> The description of the tags named and the tag structures can be found in the Appendix Tags of the positioning axis technology object V4...5 (Page 6436).	
Value	INOUT	VARIANT (REAL)	-	VARIANT pointer to the target tag or destination address to which the read value is to be written.	
Valid	OUTPUT	BOOL	FALSE	TRUE	The read value is valid.
				FALSE	The read value is invalid.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_Power: Enable, disable axis V4...5 (Page 4129)
MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)
MC_Home: Home axis, set home position V4...5 (Page 4136)
MC_Halt: Stop axis V4...5 (Page 4140)
MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)
MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)
MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)
MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)
MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)
MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)
MC_WriteParam: Write tag of positioning axis V4...5 (Page 4163)
S7-1200 Motion Control V1...3 (Page 4165)

MC_WriteParam (S7-1200)

MC_WriteParam: Write tag of positioning axis V4...5 (S7-1200)

Description

Motion Control instruction "MC_WriteParam" enables the writing of tags of the positioning axis technology object in the user program. In contrast to the value assignment of the tags in the user program, "MC_WriteParam" can also change values of read-only tags.

You can learn about the tags, the conditions under which they can be written and the time at which they take effect in the description of the technology object tags (Page 6436).

Requirements

- The positioning axis technology object has been configured correctly.
- To write tags that are read-only in the user program, the axis must be disabled.

Override response

A MC_WriteParam command cannot be aborted by any other Motion Control command.

A new MC_WriteParam command does not abort any active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Parameter	INPUT	VARIANT (BOOL, INT, DINT, UDINT*, REAL) *) as of V5	-	VARIANT pointer to the technology object tag (Page 6436) positioning axis (destination address) to be written	
Value	INPUT	VARIANT (BOOL, INT, DINT, UDINT*, REAL) *) as of V5	-	VARIANT pointer to the value to be written (source address)	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Done	OUTPUT	BOOL	FALSE	TRUE	Value was written
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

MC_Power: Enable, disable axis V4...5 (Page 4129)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

MC_Home: Home axis, set home position V4...5 (Page 4136)

MC_Halt: Stop axis V4...5 (Page 4140)

MC_MoveAbsolute: Absolute positioning of axis V4...5 (Page 4143)

MC_MoveRelative: Relative positioning of axis V4...5 (Page 4146)

MC_MoveVelocity: Move axis at set velocity V4...5 (Page 4150)

MC_MoveJog: Move axis in jog mode V4...5 (Page 4154)

MC_CommandTable: Run axis commands as motion sequence V4...5 (Page 4157)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)

MC_ReadParam: Continuously read motion data of a positioning axis V4...5 (Page 4161)

Tags of the positioning axis technology object V4...5 (Page 6436)

S7-1200 Motion Control V1...3 (Page 4165)

S7-1200 Motion Control V1...3 (S7-1200)

MC_Power (S7-1200)

MC_Power: Enable, disable axis V1...3 (S7-1200)

Description

The Motion Control instruction "MC_Power" enables or disables an axis.

Requirements

- The axis technology object has been configured correctly.
- There is no pending enable-inhibiting error.

Override response

Execution of "MC_Power" cannot be aborted by a Motion Control command.

Disabling the axis (input parameter "Enable" = FALSE) aborts all Motion Control commands for the associated technology object in accordance with the selected "StopMode".

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
Enable	INPUT	BOOL	FALSE	TRUE	Motion Control attempts to enable the axis.
				FALSE	All current jobs are interrupted in accordance with the "StopMode" configured. The axis is stopped and disabled.
StopMode	INPUT	INT	0	0	Emergency stop If a request to disable the axis is pending, the axis brakes at the configured emergency stop deceleration. The axis is disabled after reaching standstill.
				1	Immediate stop If a request to disable the axis is pending, this axis is disabled without deceleration. The pulse output is stopped immediately.
				2	Emergency stop with jerk control If a request to disable the axis is pending, the axis brakes at the configured emergency stop deceleration. If the jerk control is activated, the configured jerk is taken into account. The axis is disabled after reaching standstill.

Parameter	Declaration	Data type	Default value	Description	
Status	OUTPUT	BOOL	FALSE	Status of axis enable	
				FALSE	The axis is disabled. The axis does not execute Motion Control commands and does not accept any new commands (exception: MC_Reset command). The axis is not homed. Upon disabling, the status does not change to FALSE until the axis reaches a standstill.
				TRUE	The axis is enabled. The axis is ready to execute Motion Control commands. Upon axis enabling, the status does not change to TRUE until the signal "Drive ready" is pending. If the "Drive ready" drive interface was not configured in the axis configuration, the status changes to TRUE immediately.
Busy	OUTPUT	BOOL	FALSE	TRUE	"MC_Power" is active.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred in Motion Control instruction "MC_Power" or in the associated technology object. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Note

If the axis is switched off due to an error, it will be enabled again automatically after the error has been eliminated and acknowledged. This requires that the input parameter "Enable" has retained the value TRUE during this process.

Enabling an axis with configured drive interface

To enable the axis, follow these steps:

1. Check the requirements indicated above.
2. Initialize input parameter "StopMode" with the desired value. Set the input parameter "Enable" to TRUE.
   The enable output for "Drive enabled" changes to TRUE to enable the power to the drive.
   The CPU waits for the "Drive ready" signal of the drive.
   When the "Drive ready" signal is available at the configured Ready input of the CPU, the axis is enabled. The output parameter "Status" and the tag of the technology object <Axis name>.StatusBits.Enable indicate the value TRUE.

Enabling an axis without configured drive interface

To enable the axis, follow these steps:

1. Check the requirements indicated above.
2. Initialize input parameter "StopMode" with the desired value. Set the input parameter "Enable" to TRUE. The axis is enabled. The output parameter "Status" and the tag of the technology object <Axis name>.StatusBits.Enable indicate the value TRUE.

Disabling an axis

To disable an axis, you can follow the steps described below:

1. Bring the axis to a standstill.
   You can identify when the axis is at a standstill in the tag of the technology object <Axis name>.StatusBits.StandStill.
2. Set input parameter "Enable" to FALSE after standstill is reached.
3. If output parameters "Busy" and "Status" and the tag of technology object <Axis name>.StatusBits.Enable indicate the value FALSE, disabling of the axis is complete.

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

MC_Power: Function chart V1...3 (Page 4168)

MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)

MC_Home: Home axis, set home position V1...3 (Page 4170)

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

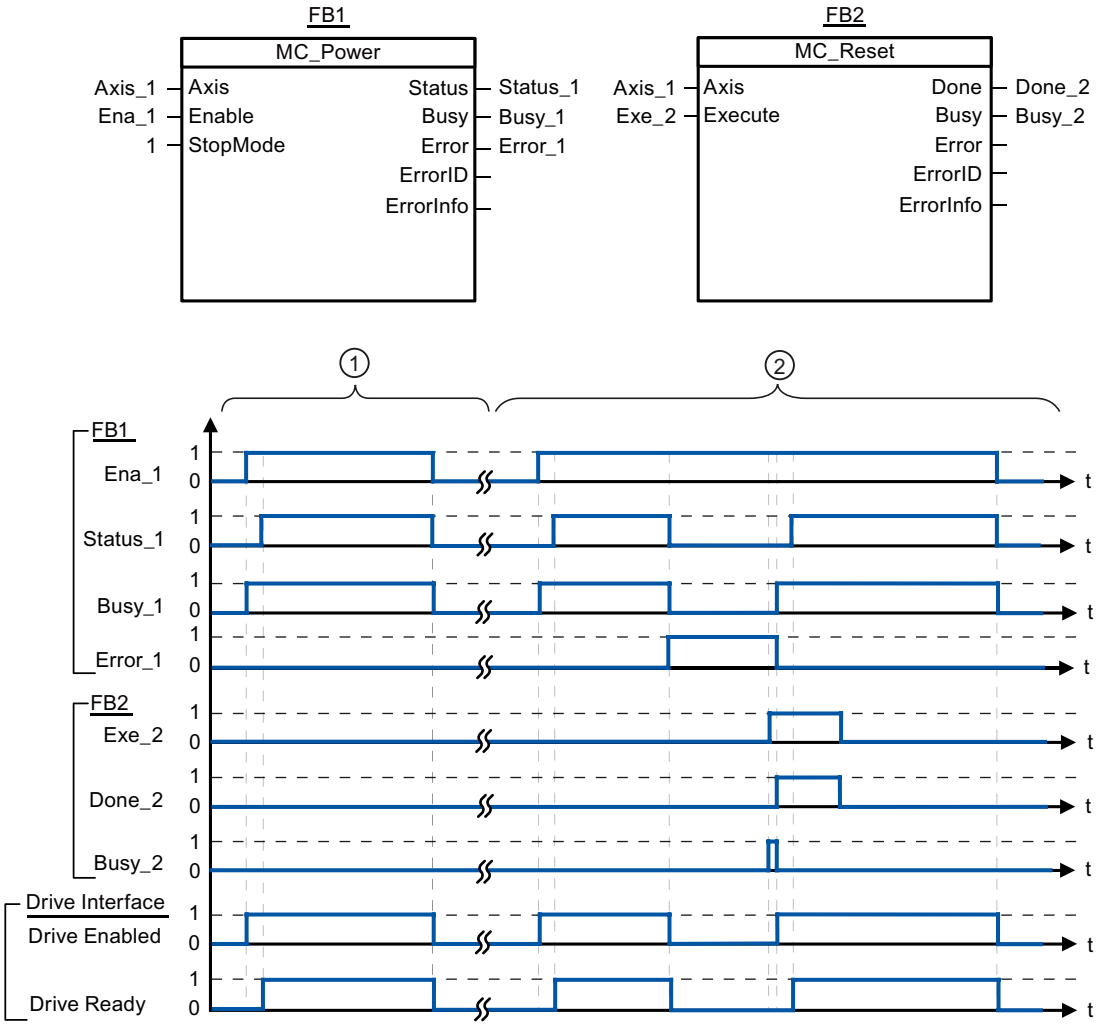
MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)

MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_Power: Function chart V1...3 (S7-1200)

Function chart



①	An axis is enabled and then disabled again. When the drive has signaled "Drive ready" back to the CPU, the successful enable can be read out via "Status_1".
②	Following an axis enable, an error has occurred that caused the axis to be disabled. The error is eliminated and acknowledged with "MC_Reset". The axis is then enabled again.

See also

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Reset (S7-1200)

MC_Reset: Acknowledge fault, restart technology object V1...3 (S7-1200)

Description

Motion Control instruction "MC_Reset" can be used to acknowledge "Operating error with axis stop" and "Configuration error". The errors that require acknowledgement can be found in the "List of ErrorIDs and ErrorInfos" under "Remedy".

From version V3.0, the axis configuration can be downloaded to the work memory in RUN operating mode.

Requirements

- The axis technology object has been configured correctly.
- The cause of a pending configuration error requiring acknowledgement has been eliminated (for example, acceleration in positioning axis technology object has been changed to a valid value).

Override response

The MC_Reset command cannot be aborted by any other Motion Control command.

The new MC_Reset command does not abort any other active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Restart	INPUT	BOOL	FALSE	(From version V3.0)	
				TRUE	Download the axis configuration from the load memory to the work memory. The command can only be executed when the axis is disabled. Refer to the notes on Download to the CPU (Page 6260).
				FALSE	Acknowledges pending errors
Done	OUTPUT	BOOL	FALSE	TRUE	Error has been acknowledged.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Acknowledging an error requiring acknowledgement with MC_Reset

To acknowledge an error, follow these steps:

1. Check the requirements indicated above.
2. Start the acknowledgement of the error with a rising edge at input parameter "Execute".
3. If output parameter "Done" indicates the value TRUE and tag of technology object <Axis name>.StatusBits.Error the value FALSE, the error has been acknowledged.

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

Download to CPU (Page 6260)

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Home: Home axis, set home position V1...3 (Page 4170)

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)

MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_Home (S7-1200)

MC_Home: Home axis, set home position V1...3 (S7-1200)

Description

Motion Control instruction "MC_Home" is used to match the axis coordinates to the real, physical drive position. Homing is required for absolute positioning of the axis. The following types of homing can be executed:

- Active homing (Mode = 3)
  The homing procedure is executed automatically.
- Passive homing (Mode = 2)
  During passive homing, the "MC_Home" Motion Control instruction does not carry out any homing motion. The traversing motion required for this must be implemented by the user via other Motion Control instructions. When the homing switch is detected, the axis is homed.

- Direct homing absolute (Mode = 0)
  The current axis position is set to the value of parameter "Position".
- Direct homing relative (Mode = 1)
  The current axis position is offset by the value of parameter "Position".

Requirements

- The axis technology object has been configured correctly.
- The axis is enabled.
- No MC_CommandTable command may be active upon start with Mode = 0, 1, or 2.

Override response

The override response depends on the selected mode:

Mode = 0, 1

The MC_Home command cannot be aborted by any other Motion Control command.

The MC_Home command does not abort any active Motion Control commands. Position-related motion commands are resumed after homing according to the new homing position (value at input parameter: "Position").

Mode = 2

The MC_Home command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 2, 3

The new MC_Home command aborts the following active Motion Control command:

- MC_Home command Mode = 2

Position-related motion commands are resumed after homing according to the new homing position (value at input parameter: "Position").

Mode = 3

The MC_Home command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_Home command aborts the following active Motion Control commands:

- MC_Home command Mode = 2, 3
- MC_Halt command
- MC_MoveAbsolute command

- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Position	INPUT	REAL	0.0	• Mode = 0, 2, and 3 Absolute position of axis after completion of the homing operation • Mode = 1 Correction value for the current axis position Limit values: -1.0E+12 ≤ Position ≤ 1.0E+12	
Mode	INPUT	INT	0	Homing mode	
				0	Direct homing absolute New axis position is the position value of parameter "Position".
				1	Direct homing relative New axis position is the current axis position + position value of parameter "Position".
				2	Passive homing Homing according to the axis configuration. Following homing, the value of parameter "Position" is set as the new axis position.
				3	Active homing Homing procedure in accordance with the axis configuration. Following homing, the value of parameter "Position" is set as the new axis position.
Done	OUTPUT	BOOL	FALSE	TRUE	Command completed
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Note

Axis homing is lost under the following conditions:

- Disabling of axis by the "MC_Power" Motion Control instruction
- Changeover between automatic mode and manual control
- Upon start of active homing. After successful completion of the homing operation, axis homing is again available.
- After POWER OFF -> POWER ON of the CPU
- After CPU restart (RUN-STOP -> STOP-RUN)

Homing an axis

To home the axis, follow these stops:

1. Check the requirements indicated above.
2. Initialize the necessary input parameters with values, and start the homing operation with a rising edge at input parameter "Execute"
3. If output parameter "Done" and technology object tag <Axis name>.StatusBits.HomingDone indicate the value TRUE, homing is complete.

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)

MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_Halt (S7-1200)

MC_Halt: Stop axis V1...3 (S7-1200)

Description

The "MC_Halt" Motion Control instruction stops all movements and brings the axis to a standstill with the configured deceleration. The standstill position is not defined.

Requirements

- The axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_Halt command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_Halt command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Done	OUTPUT	BOOL	FALSE	TRUE	Zero velocity reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.

Parameter	Declaration	Data type	Default value	Description	
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

MC_Halt: Function chart V1...3 (Page 4176)

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)

MC_Home: Home axis, set home position V1...3 (Page 4170)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

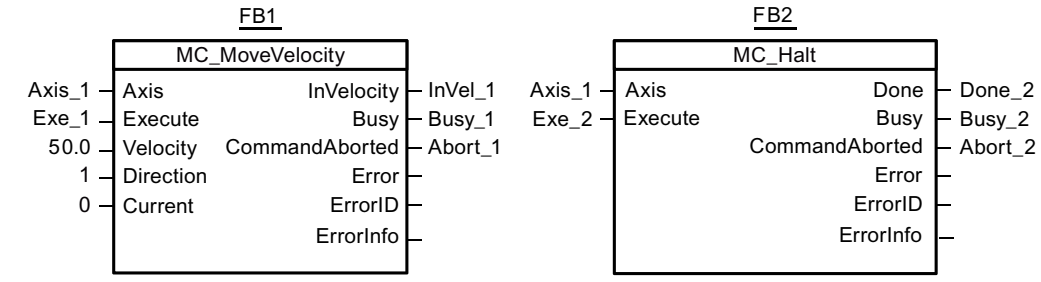
MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

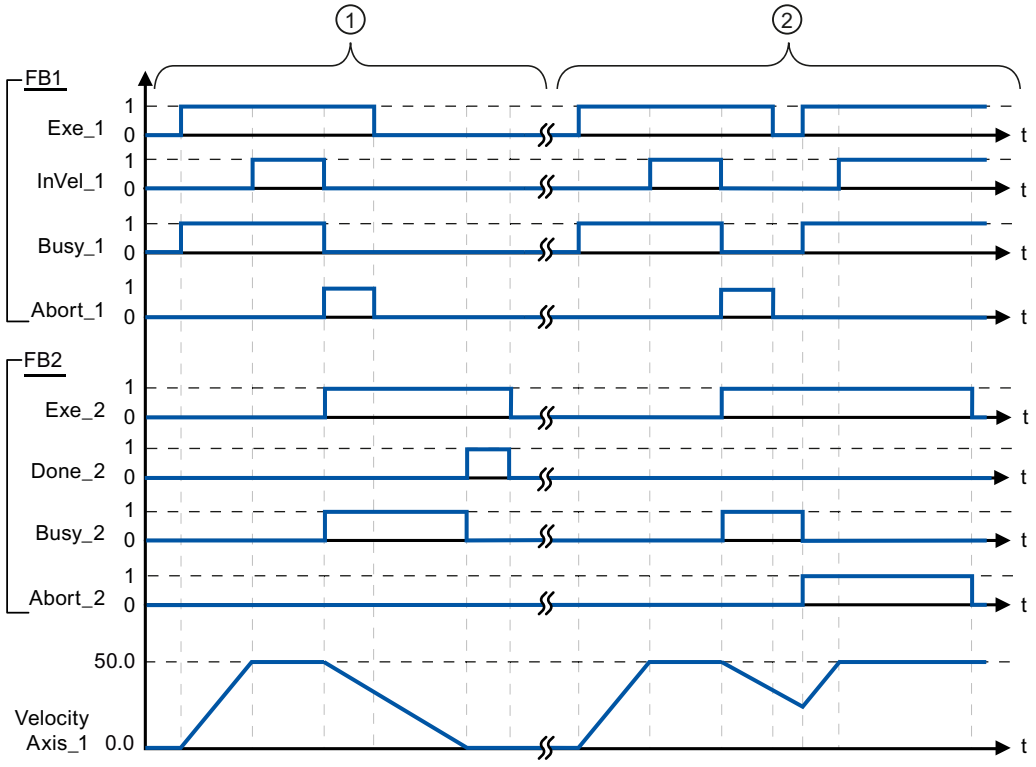
MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)

MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_Halt: Function chart V1...3 (S7-1200)

Function chart





The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 5.0

①	The axis is braked by an MC_Halt command until it comes to a standstill. The axis standstill is signaled via "Done_2".
②	While an MC_Halt command is braking the axis, this command is aborted by another motion command. The abort is signaled via "Abort_2".

See also

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute (S7-1200)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (S7-1200)

Description

The "MC_MoveAbsolute" Motion Control instruction starts an axis positioning motion to move it to an absolute position.

Requirements

- The axis technology object has been configured correctly.
- The axis is enabled.
- The axis is homed.

Override response

The MC_MoveAbsolute command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveAbsolute command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

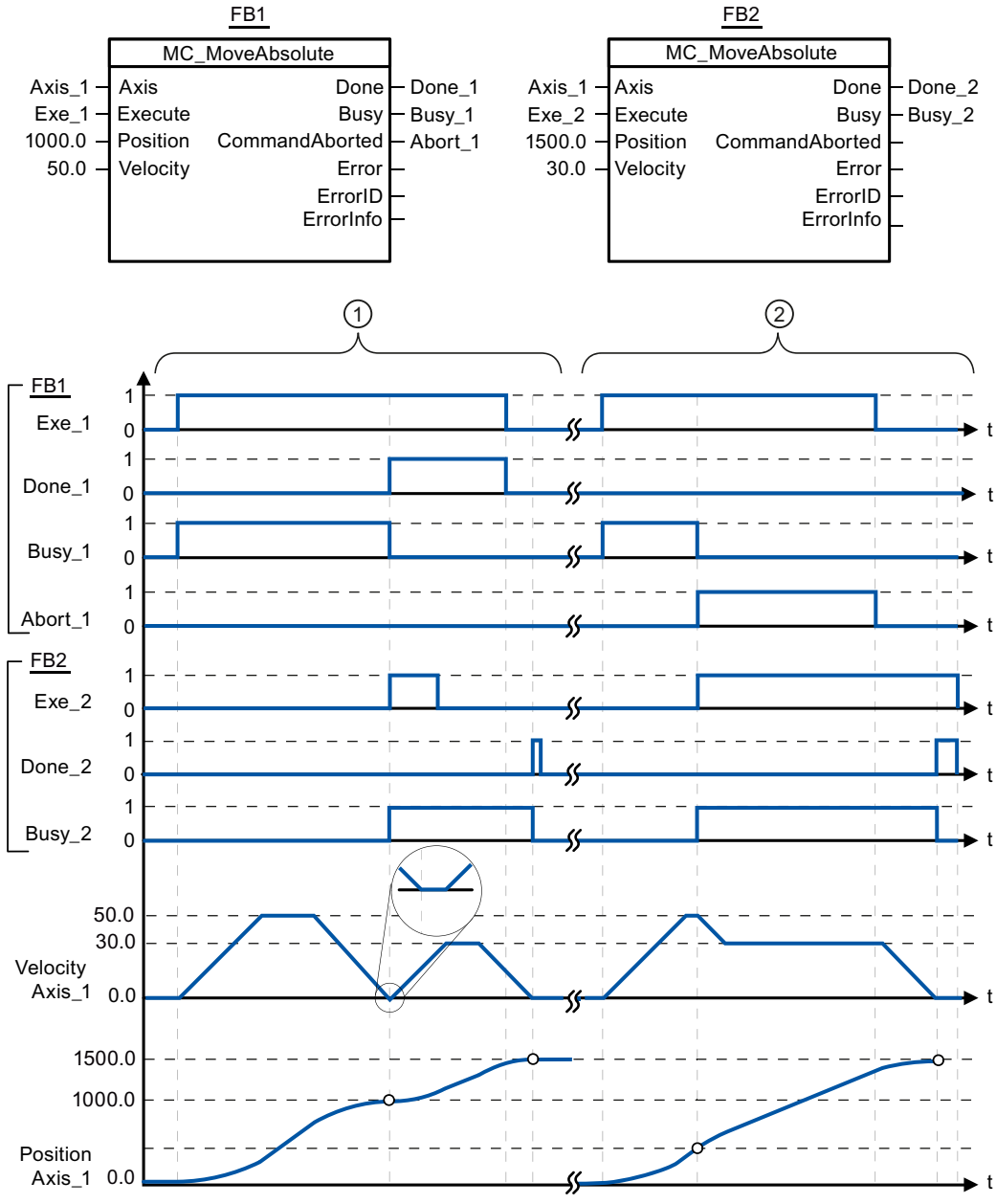
Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Position	INPUT	REAL	0.0	Absolute target position Limit values: $-1.0e^{12} \leq$ Position $\leq 1.0e^{12}$	
Velocity	INPUT	REAL	10.0	Velocity of axis This velocity is not always reached on account of the configured acceleration and deceleration and the target position to be approached. Limit values: Start/stop velocity ≤ Velocity ≤ maximum velocity	
Done	OUTPUT	BOOL	FALSE	TRUE	Absolute target position reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)
S7-1200 Motion Control V4...5 (Page 4129)
ErrorIDs and ErrorInfos (Page 6386)
MC_MoveAbsolute: Function chart V1...3 (Page 4179)
MC_Power: Enable, disable axis V1...3 (Page 4165)
MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)
MC_Home: Home axis, set home position V1...3 (Page 4170)
MC_Halt: Stop axis V1...3 (Page 4174)
MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)
MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)
MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)
MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)
MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_MoveAbsolute: Function chart V1...3 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	An axis is moved to absolute position 1000.0 with an MC_MoveAbsolute command. When the axis reaches the target position, this is signaled via "Done_1". When "Done_1" = TRUE, another MC_MoveAbsolute command, with target position 1500.0, is started. Because of the response times (e.g., cycle time of user program, etc.), the axis comes to a standstill briefly (see zoomed-in detail). When the axis reaches the new target position, this is signaled via "Done_2".
②	An active MC_MoveAbsolute command is aborted by another MC_MoveAbsolute command. The abort is signaled via "Abort_1". The axis is then moved at the new velocity to the new target position 1500.0. When the new target position is reached, this is signaled via "Done_2".

See also

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative (S7-1200)

MC_MoveRelative: Relative positioning of axis V1...3 (S7-1200)

Description

The "MC_MoveRelative" Motion Control instruction starts a positioning motion relative to the start position.

Requirements

- The axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_MoveRelative command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveRelative command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command

- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Distance	INPUT	REAL	0.0	Travel distance for the positioning operation Limit values: $-1.0e^{12} \leq$ Distance $\leq 1.0e^{12}$	
Velocity	INPUT	REAL	10.0	Velocity of axis This velocity is not always reached on account of the configured acceleration and deceleration and the distance to be traveled. Limit values: Start/stop velocity ≤ Velocity ≤ maximum velocity	
Done	OUTPUT	BOOL	FALSE	TRUE	Target position reached
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed.
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

MC_MoveRelative: Function chart V1...3 (Page 4182)

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)

MC_Home: Home axis, set home position V1...3 (Page 4170)

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

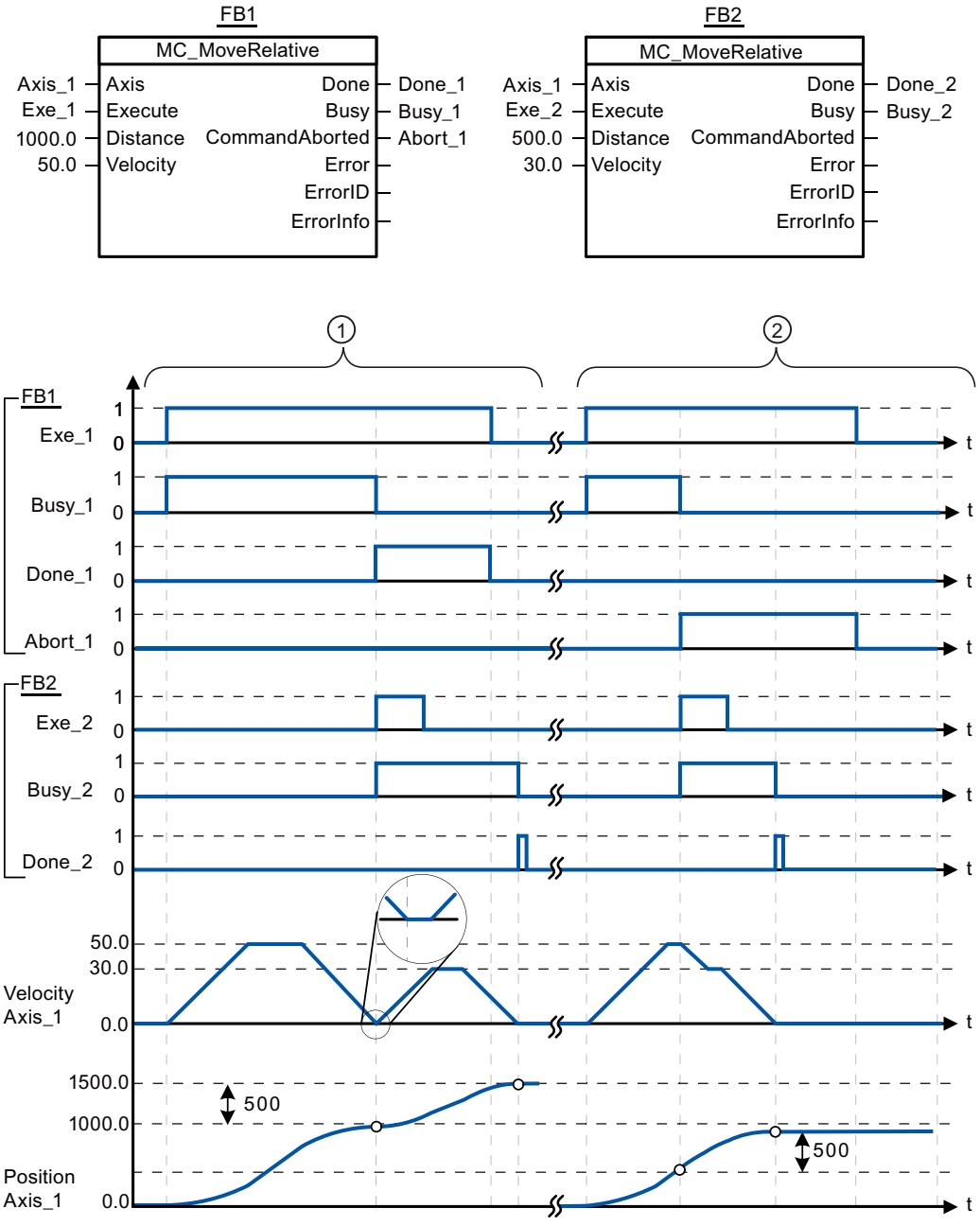
MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)
MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_MoveRelative: Function chart V1...3 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	The axis is moved by an MC_MoveRelative command by the distance ("Distance") 1000.0. When the axis reaches the target position, this is signaled via "Done_1". When "Done_1" = TRUE, another MC_MoveRelative command, with travel distance 500.0, is started. Because of the response times (e.g., cycle time of user program, etc.), the axis comes to a standstill briefly (see zoomed-in detail). When the axis reaches the new target position, this is signaled via "Done_2".
②	An active MC_MoveRelative command is aborted by another MC_MoveRelative command. The abort is signaled via "Abort_1". The axis is then moved at the new velocity by the new distance ("Distance") 500.0. When the new target position is reached, this is signaled via "Done_2".

See also

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

MC_MoveVelocity (S7-1200)

MC_MoveVelocity: Move axis at set velocity V1...3 (S7-1200)

Description

Motion control instruction "MC_MoveVelocity" moves the axis constantly at the specified velocity.

Requirements

- The axis technology object has been configured correctly.
- The axis is enabled.

Override response

MC_MoveVelocity can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveVelocity command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
Velocity	INPUT	REAL	10.0	Velocity specification for axis motion Limit values: Start/stop velocity ≤ |Velocity ≤ maximum velocity (Velocity = 0.0 is permitted)	
Direction	INPUT	INT	0	Direction specification	
				0	Direction of rotation corresponds to the sign of the value in parameter "Velocity"
				1	Positive direction of rotation (The sign of the value in parameter "Velocity" is ignored)
				2	Negative direction of rotation (The sign of the value in parameter "Velocity" is ignored)
Current	INPUT	BOOL	FALSE	Maintain current velocity	
				FALSE	"Maintain current velocity" is deactivated. The values of parameters "Velocity" and "Direction" are used.
				TRUE	"Maintain current velocity" is activated. The values in parameters "Velocity" and "Direction" are not taken into account. When the axis resumes motion at the current velocity, the "InVelocity" parameter returns the value TRUE.
InVelocity	OUTPUT	BOOL	FALSE	TRUE	• "Current" = FALSE: The velocity specified in parameter "Velocity" was reached. • "Current" = TRUE: The axis travels at the current velocity at the start time.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed

Parameter	Declaration	Data type	Default value	Description	
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Behavior with zero setpoint velocity (Velocity = 0.0)

An MC_MoveVelocity command with "Velocity" = 0.0 (such as an MC_Halt command) aborts active motion commands and stops the axis with the configured deceleration.

When the axis comes to a standstill, output parameter "InVelocity" indicates TRUE for at least one program cycle.

"Busy" indicates the value TRUE during the deceleration process and changes to FALSE together with "InVelocity". If parameter "Execute" = TRUE is set, "InVelocity" and "Busy" are latched.

When the "MC_MoveVelocity" command is started, status bit "SpeedCommand" is set in the technology object. Status bit "ConstantVelocity" is set upon axis standstill. Both bits are adapted to the new situation when a new motion command is started.

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

MC_MoveVelocity: Function chart V1...3 (Page 4186)

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)

MC_Home: Home axis, set home position V1...3 (Page 4170)

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

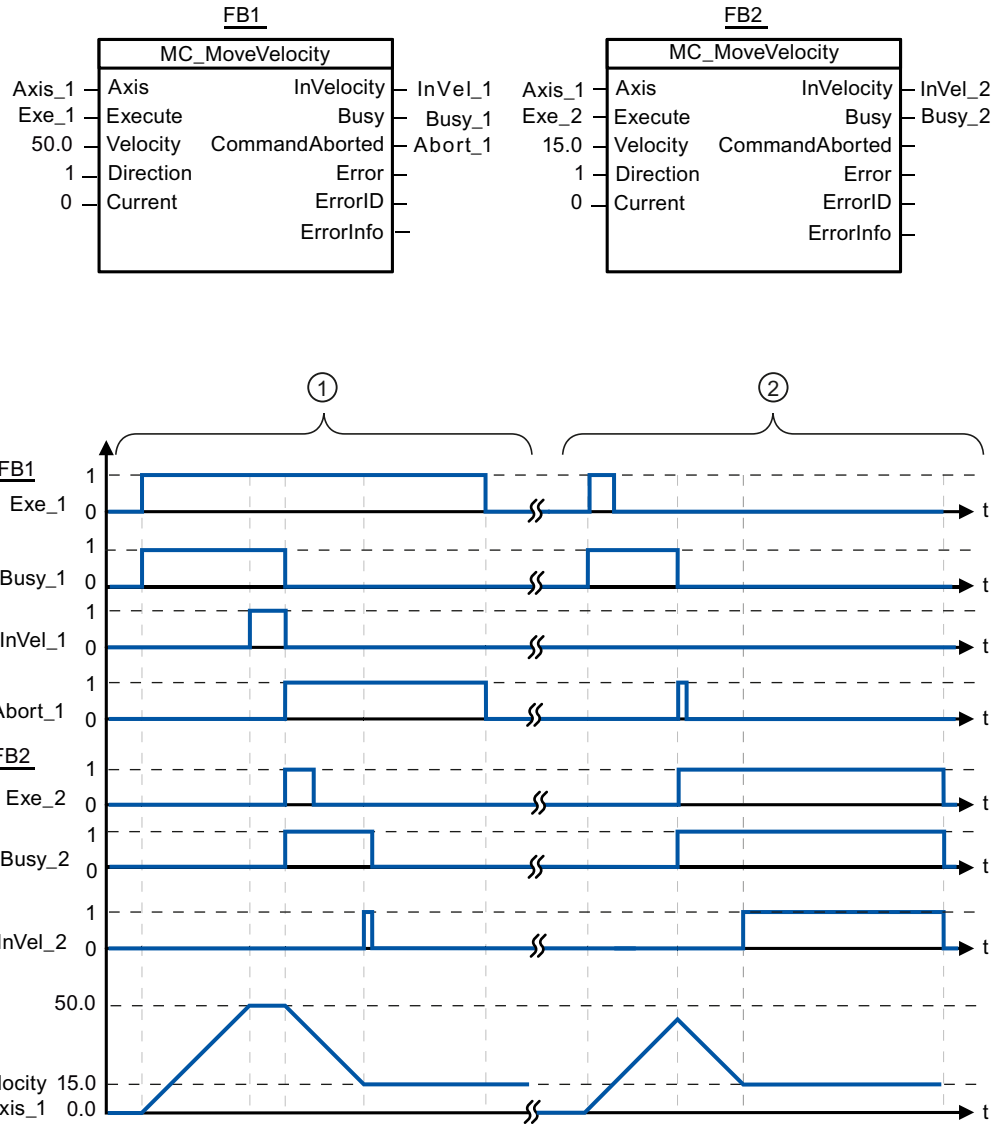
MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)

MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_MoveVelocity: Function chart V1...3 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 10.0

①	An active MC_MoveVelocity command signals via "InVel_1" that its target velocity has been reached. It is then aborted by another MC_MoveVelocity command. The abort is signaled via "Abort_1". When the new target velocity 15.0 is reached, this is signaled via "InVel_2". The axis then continues moving at the new constant velocity.
②	An active MC_MoveVelocity command is aborted by another MC_MoveVelocity command prior to reaching its target velocity. The abort is signaled via "Abort_1". When the new target velocity 15.0 is reached, this is signaled via "InVel_2". The axis then continues moving at the new constant velocity.

See also

MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

MC_MoveJog (S7-1200)

MC_MoveJog: Move axis in jog mode V1...3 (S7-1200)

Description

Motion control instruction "MC_MoveJog" moves the axis constantly at the specified velocity in jog mode. You use this Motion Control instruction, for example, for testing and commissioning purposes.

Requirements

- The axis technology object has been configured correctly.
- The axis is enabled.

Override response

The MC_MoveJog command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_MoveJog command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
JogForward	INPUT	BOOL	FALSE	As long as the parameter is TRUE, the axis moves in the positive direction at the velocity specified in parameter "Velocity".	
JogBackward	INPUT	BOOL	FALSE	As long as the parameter is TRUE, the axis moves in the negative direction at the velocity specified in parameter "Velocity".	
If both parameters are simultaneously TRUE, the axis stops with the configured deceleration. An error is indicated in parameters "Error", "ErrorID", and "ErrorInfo".					
Velocity	INPUT	REAL	10.0	Preset velocity for jog mode	
				Limit values, instruction version V1.0: Start/stop velocity ≤ |Velocity ≤ maximum velocity	
				Limits, instruction version V2.0: Start/stop velocity ≤ velocity ≤ maximum velocity	
InVelocity	OUTPUT	BOOL	FALSE	TRUE	The velocity specified in parameter "Velocity" was reached.
Busy	OUTPUT	BOOL	FALSE	TRUE	The command is being executed
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	During execution, the command was aborted by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

MC_MoveJog: Function chart V1...3 (Page 4189)

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)

MC_Home: Home axis, set home position V1...3 (Page 4170)

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

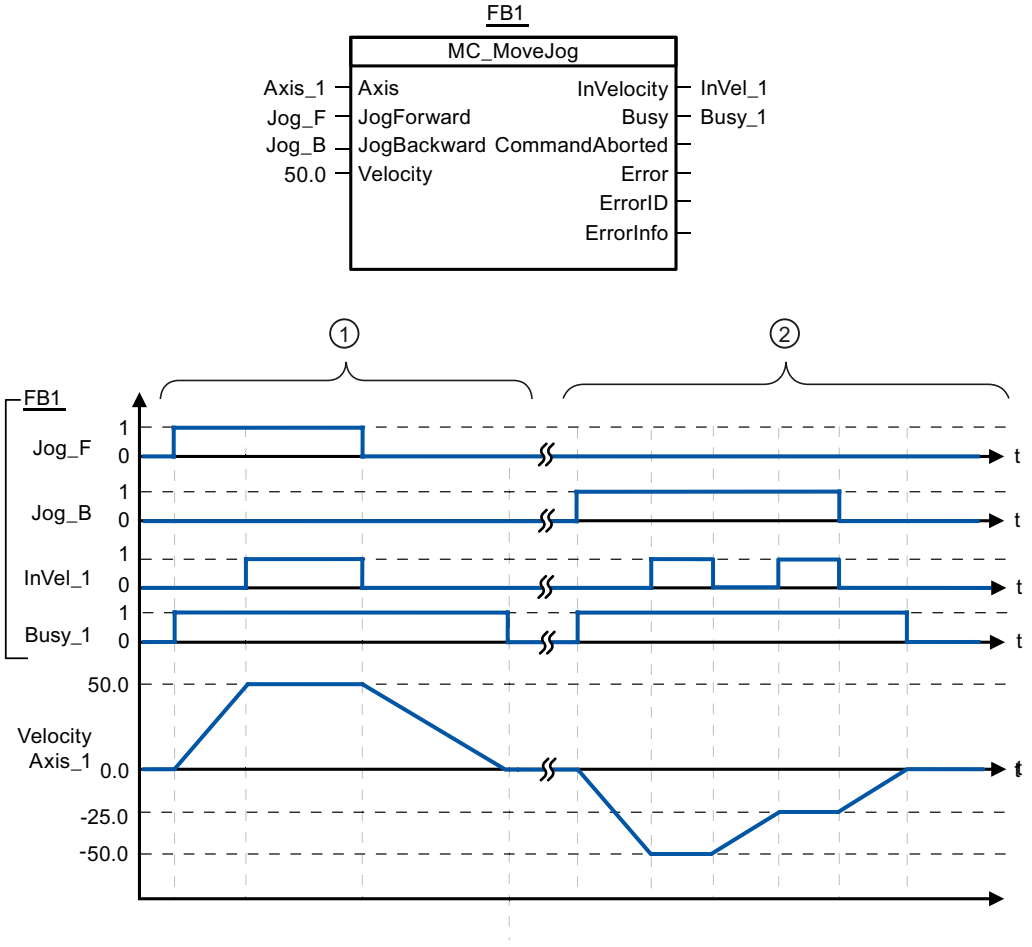
MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)

MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_MoveJog: Function chart V1...3 (S7-1200)

Function chart



The following values were configured in the configuration window **Dynamics > General**:

- Acceleration: 10.0
- Deceleration: 5.0

①	The axis is moved in the positive direction in jog mode via "Jog_F". When the target velocity 50.0 is reached, this is signaled via "InVelo_1". After " Jog_F" is reset, the axis brakes again until it comes to a standstill.
②	The axis is moved in the negative direction in jog mode via "Jog_B". When the target velocity 50.0 is reached, this is signaled via "InVelo_1". After " Jog_B" is reset, the axis brakes again until it comes to a standstill.

See also

MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

MC_CommandTable (S7-1200)

MC_CommandTable: Run axis commands as motion sequence V2...3 (S7-1200)

Description

The Motion Control instruction "MC_CommandTable" combines multiple individual axis control commands in one movement sequence.

Requirements

- The axis technology object has been inserted in V2 and correctly configured.
- The command table technology object has been inserted and correctly configured.
- The axis is enabled.

Override response

The MC_CommandTable command can be aborted by the following Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The new MC_CommandTable command aborts the following active Motion Control commands:

- MC_Home command Mode = 3
- MC_Halt command
- MC_MoveAbsolute command
- MC_MoveRelative command
- MC_MoveVelocity command
- MC_MoveJog command
- MC_CommandTable command

The active Motion Control command is cancelled by the start of the first "Positioning Relative", "Positioning Absolute", "Velocity set point" or "Halt" command.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
CommandTable	INPUT	TO_CommandTable_1	-	Command table technology object	
Execute	INPUT	BOOL	FALSE	Command table start with positive edge	
StartStep	INPUT	INT	1	Defines the step at which the execution of the command table should begin Limit values: 1 ≤ StartStep ≤ EndStep	
EndStep	INPUT	INT	32	Defines the step up to which the execution of command table should take place Limit values: StartStep ≤ EndStep ≤ 32	
Done	OUTPUT	BOOL	FALSE	TRUE	Command table has been successfully executed
Busy	OUTPU	BOOL	FALSE	TRUE	The command table is being executed.
CommandAborted	OUTPUT	BOOL	FALSE	TRUE	The command table was cancelled by another command.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command table. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	
CurrentStep	OUTPUT	INT	0	Step in command table currently being executed	
StepCode	OUTPUT	WORD	16#0000	User-defined numerical value / bit pattern of the step currently being executed	

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)

MC_Home: Home axis, set home position V1...3 (Page 4170)

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

MC_ChangeDynamic: Change dynamic settings of axis V2...3 (Page 4192)

MC_ChangeDynamic (S7-1200)

MC_ChangeDynamic: Change dynamic settings of axis V2...3 (S7-1200)

Description

Motion Control instruction "MC_ChangeDynamic" allows you to change the following settings of the axis:

- Change the ramp-up time (acceleration) value
- Change the ramp-down time (deceleration) value
- Change the emergency stop ramp-down time (emergency stop deceleration) value
- Change the smoothing time (jerk) value

For the effectiveness of the change, refer to the description of the tag (Page 6420).

Requirements

- The axis technology object has been inserted in Version V2.
- The axis technology object has been configured correctly.

Override response

A MC_ChangeDynamic command cannot be aborted by any other Motion Control command.

A new MC_ChangeDynamic command does not abort any active Motion Control commands.

Parameters

Parameter	Declaration	Data type	Default value	Description	
Axis	INPUT	TO_Axis_1	-	Axis technology object	
Execute	INPUT	BOOL	FALSE	Start of the command with a positive edge	
ChangeRampUp	INPUT	BOOL	FALSE	TRUE	Change ramp-up time in line with input parameter "RampUpTime"
RampUpTime	INPUT	REAL	5.00	Time (in seconds) to accelerate axis from standstill to configured maximum velocity without jerk limit. The change will influence the tag <Axis name>. Config.DynamicDefaults.Acceleration. For the effectiveness of the change, refer to the description of this tag.	
ChangeRampDown	INPUT	BOOL	FALSE	TRUE	Change ramp-down time in line with input parameter "RampDownTime"
RampDownTime	INPUT	REAL	5.00	Time (in seconds) to decelerate axis from the configured maximum velocity to standstill without jerk limiter. The change will influence the tag <Axis name>. Config.DynamicDefaults.Deceleration . For the effectiveness of the change, refer to the description of this tag.	

Parameter	Declaration	Data type	Default value	Description	
ChangeEmergency	INPUT	BOOL	FALSE	TRUE	Change emergency stop ramp-down time in line with input parameter "EmergencyRampTime"
EmergencyRampTime	INPUT	REAL	2.00	Time (in seconds) to decelerate the axis from configured maximum velocity to standstill without jerk limiter in emergency stop mode. The change will influence the tag <Axis name>. Config.DynamicDefaults.EmergencyDeceleration . For the effectiveness of the change, refer to the description of this tag.	
ChangeJerkTime	INPUT	BOOL	FALSE	TRUE	Change smoothing time according to the input parameter "JerkTime"
JerkTime	INPUT	REAL	0.25	Smoothing time (in seconds) used for the axis acceleration and deceleration ramps The change will influence the tag <Axis name>. Config.DynamicDefaults.Jerk . For the effectiveness of the change, refer to the description of this tag.	
Done	OUTPUT	BOOL	FALSE	TRUE	The changed values have been written to the technology data block. The description of the tags will show when the change becomes effective.
Error	OUTPUT	BOOL	FALSE	TRUE	An error occurred during execution of the command. The cause of the error can be found in parameters "ErrorID" and "ErrorInfo".
ErrorID	OUTPUT	WORD	16#0000	Error ID (Page 6386) for parameter "Error"	
ErrorInfo	OUTPUT	WORD	16#0000	Error info ID (Page 6386) for parameter "ErrorID"	

Note

At the input parameters "RampUpTime", "RampDownTime", "EmergencyRampTime" and "JerkTime", values can be entered which exceed the admissible limits of the resulting parameters: "Acceleration", "Deceleration", "Emergency stop deceleration" and "Jerk".

Please note the equations and limits in "Axis technology object" -> "Configuring the technology object" -> "Dynamics" and ensure that the values you input are within the valid range.

See also

Overview of the Motion Control statements (Page 6266)

S7-1200 Motion Control V4...5 (Page 4129)

ErrorIDs and ErrorInfos (Page 6386)

Tag of the axis technology object V1...3 (Page 6420)

MC_Power: Enable, disable axis V1...3 (Page 4165)

MC_Reset: Acknowledge fault, restart technology object V1...3 (Page 4169)

MC_Home: Home axis, set home position V1...3 (Page 4170)

MC_Halt: Stop axis V1...3 (Page 4174)

MC_MoveAbsolute: Absolute positioning of axis V1...3 (Page 4177)

MC_MoveRelative: Relative positioning of axis V1...3 (Page 4180)

MC_MoveVelocity: Move axis at set velocity V1...3 (Page 4183)

MC_MoveJog: Move axis in jog mode V1...3 (Page 4187)

MC_CommandTable: Run axis commands as motion sequence V2...3 (Page 4190)

11.4.5.2 High-speed counters (S7-1200)

CTRL_HSC: Control high-speed counters (S7-1200)

Parameters

Parameter	Declaration	Data type	Memory area	Description
EN	INPUT	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	OUTPUT	BOOL	I, Q, M, D, L	Enable output
HSC	INPUT	HW_HSC	I, Q, M or constant	Hardware address of the high-speed counter (HW-ID)
DIR	INPUT	BOOL	I, Q, M, D, L or constant	Enables the new count direction (see NEW_DIR)
CV	INPUT	BOOL	I, Q, M, D, L or constant	Enables the new count value (see NEW_CV)
RV	INPUT	BOOL	I, Q, M, D, L or constant	Enables the new reference value (see NEW_RV)
PERIOD	INPUT	BOOL	I, Q, M, D, L or constant	Enables the new period of a frequency measurement (see NEW_PERIOD)
NEW_DIR	INPUT	INT	I, Q, M, D, L or constant	Count direction loaded when DIR = TRUE.
NEW_CV	INPUT	DINT	I, Q, M, D, L or constant	Count value loaded when CV = TRUE.
NEW_RV	INPUT	DINT	I, Q, M, D, L or constant	Reference value loaded when RV = TRUE.
NEW_PERIOD	INPUT	INT	I, Q, M, D, L or constant	Period of the frequency measurement loaded when PERIOD = TRUE.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	Processing status *
STATUS	OUTPUT	WORD	I, Q, M, D, L	Status of the operation

* With a high-speed counter in the CPU or in the signal board, the BUSY parameter always has the value 0.

Description

With the "Control high-speed counter" instruction, you can make parameter settings and control the high-speed counters supported by the CPU by loading new values into the counter. Execution of the instruction requires that a high-speed counter to be controlled is enabled. You cannot execute multiple "Control high-speed counter" instructions simultaneously in the program for a given high-speed counter.

You can load the following parameter values into a high-speed counter using the "Control high-speed counter" instruction:

- Count direction (NEW_DIR): The count direction defines whether a high-speed counter counts up or down. The count direction is defined by the following values at the NEW_DIR input: 1 = up, -1 = down.
  A change to the count direction with the "Control high-speed counter" instruction is only possible when direction control is set in the parameters by the program. The count direction specified at the NEW_DIR input is loaded into a high-speed counter when the bit at the DIR input is set.
- Count value (NEW_CV): The count value is the initial value at which a high-speed counter starts counting. The count value can be in the range -2147483648 to 2147483647.
  The count value specified at the NEW_CV input is loaded into a high-speed counter when the bit at the CV input is set.
- Reference value (NEW_RV): You can compare the reference value with the current counter value to trigger an alarm. Similar to the counter value, the reference value can be in the range -2147483648 to 2147483647.
  The reference value specified at the NEW_RV input is loaded into a high-speed counter when the bit at the RV input is set.
- Period of the frequency measurement (NEW_PERIOD): The period of the frequency measurement is specified by the following values at the NEW_PERIOD input: 10 = 0.01s, 100 = 0.1s, 1000 = 1s.
  The time period can be updated if the "Measure frequency" function for the specified high-speed counter is configured. The time period specified at the NEW_PERIOD input is loaded into a high-speed counter when the bit at the PERIOD input is set.

The "Control high-speed counter" instruction is only executed if the signal state at the EN input is "1".

The ENO enable output is set only when the EN enable input has signal state "1" and no errors occur during execution of the operation.

When inserting the "Control high-speed counter" instruction, an instance data block is created in which the operation data is saved.

STATUS parameter

At the STATUS output, you can query whether errors occurred during execution of the "Control high-speed counter" instruction. The following table shows the meaning of the values output at the STATUS output:

Error code (hexadecimal)	Description
0	No error
80A1	Hardware identifier of the high-speed counter invalid

Error code (hexadecimal)	Description
80B1	Count direction (NEW_DIR) invalid
80B2	Count value (NEW_CV) invalid
80B3	Reference value (NEW_RV) invalid
80B4	Period of the frequency measurement (NEW_PERIOD) invalid
80C0	Multiple access to the high-speed counter
80D0	The high-speed counter (HSC) is not enabled in the CPU hardware configuration.

CTRL_HSC_EXT: Control high-speed counter (extended) (S7-1200)

Parameters

Parameters	Declaration	Data type	Memory area	Description
EN	INPUT	BOOL	I, Q, M, D, L, T, C	Enable input
ENO	OUTPUT	BOOL	I, Q, M, D, L	Enable output
HSC	INPUT	HW_HSC	I, Q, M or constant	Hardware address of the high-speed counter (HW-ID)
CTRL	INOUT	VARIANT	M, D	Use of a system data type (SDT)
DONE	OUTPUT	BOOL	I, Q, M, D, L	Feedback after successful processing instruction
BUSY	OUTPUT	BOOL	I, Q, M, D, L	Processing status
ERROR	OUTPUT	BOOL	I, Q, M, D, L	Feedback for faulty processing of the instruction
STATUS	OUTPUT	WORD	I, Q, M, D, L	Status of the operation

Description

With the "Control high-speed counter (extended)" instruction, you can control and assign parameters to the high-speed counters supported by the CPU via the software by loading new values into the counters. Execution of the instruction requires that a high-speed counter to be controlled is enabled. You cannot execute multiple "Control high-speed counter (extended)" instructions simultaneously in the program for a given high-speed counter.

The "Control high-speed counter (extended)" instruction is only executed if the signal state at the EN input is "1". As long as the operation is executing, the bit at the BUSY output is set. Once the operation has executed completely, the bit at the BUSY output is reset.

The ENO enable output is set only when the EN enable input has signal state "1" and no errors occur during execution of the operation.

When the "Control high-speed counter (extended)" instruction is inserted, an instance data block is created in which the operation data is saved.

Using the system data type HSC_Period

The "Control high-speed counter (extended)" instruction supports the system data type SDT 381 "HSC_Period" for the period measurement.

The "HSC_Period" data type corresponds to an HSC that is configured for the "Period" operating mode. The CTRL_HSC_EXT instruction offers program access to the number of input pulses across a specified measuring interval. This instruction allows the calculation of the period between the input pulses with a resolution in the nanosecond range.

Byte	Parameters	Declaration	Data type	Description
0 … 3	Elapsed-Time	OUT	UDINT	Time between the rising edges of Edge_Count
4 … 7	EdgeCount	OUT	UDINT	Number of the rising edges within the Elapsed_Time. If Edge_Count = 0, the Elapsed_Time is the time since the last rising edge.
8.0	EnHSC	IN	BOOL	Use as an enable input via gate control: • FALSE: Measurement stopped • TRUE: Measurement enabled
8.6	EnPeriod	IN	BOOL	Update of the period • FALSE: No update • TRUE: Update period
10 .. . 11	NewPeriod	IN	INT	Interval of the period measurement in milliseconds. Permissible values are 10, 100 and 1000.

ElapsedTime specifies the time in nanoseconds between the last counter events of consecutive measuring intervals. If no counter events occurred during a measuring interval, ElapsedTime outputs the accumulated time since the last counter event. ElapsedTime has a range of 0 to 4,294,967,280 nanoseconds (0x0000 0000 to 0xFFFF FFF0). The return value 4,294,967,295 (0xFFFF FFFF) signals that a period overflow has occurred. An overflow indicates that the period between two pulse edges exceeds 4.295 seconds and that the period cannot be calculated with this instruction. The values from 0xFFFF FFF1 to 0xFFFF FFFE are reserved.

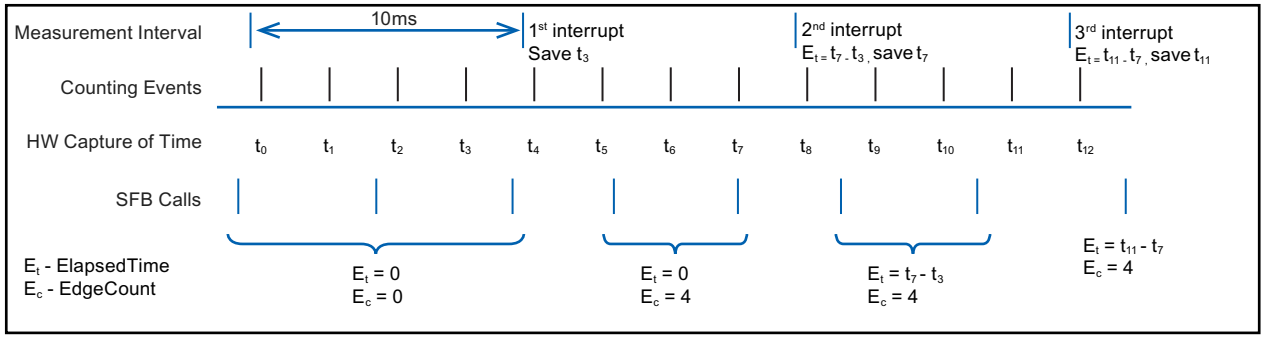
EdgeCount outputs the number of counter events received during the measuring interval. The period can only be calculated if the value of EdgeCount is greater than zero. If ElapsedTime is either 0 (no input pulses received) or OR 0xFFFF FFFF (period overflow), EdgeCount is not valid.

If EdgeCount is valid, calculate the period in nanoseconds by using the following formula: Period = ElapsedTime/EdgeCount

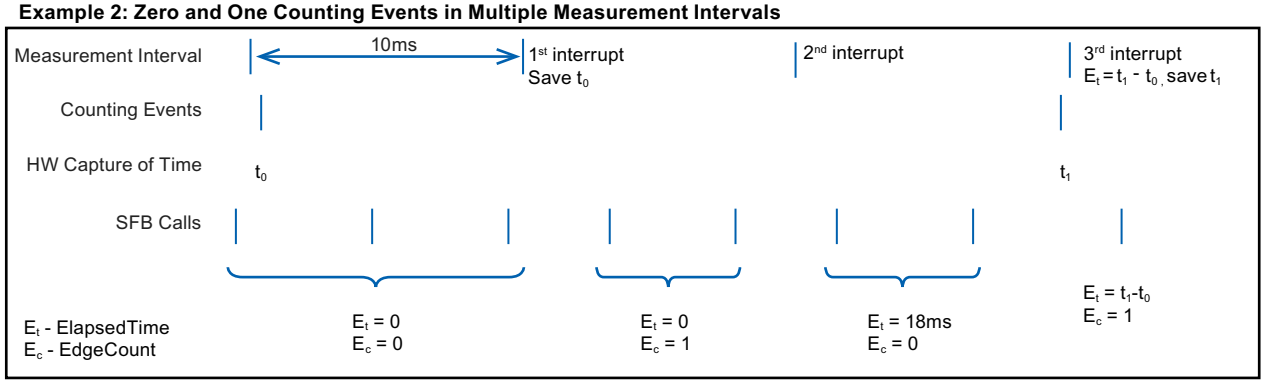
The calculated period value is a mean value from time periods of all the pulses that occur during the measuring interval. If the period of an incoming pulse is greater than the measuring interval (10, 100 or 1000 ms), the period calculation requires several measuring intervals.

The following figures show examples of the period measurement with the instruction:

Example 1: Multiple Counting Events in a Measurement Interval



Example 2: Zero and One Counting Events in Multiple Measurement Intervals



Rules:
1. If $E1_t = 0$, the period is invalid
2. Else, Period = E_t / E_c

STATUS parameter

At the STATUS output, you can query whether errors occurred during execution of the "Control high-speed counter (extended)" instruction. The following table shows the meaning of the values output at the STATUS output:

Error code (hexadecimal)	Description
0	No error
80A1	Hardware identifier of the high-speed counter invalid
80C0	Multiple access to the high-speed counter
80D0	The high-speed counter (HSC) is not enabled in the CPU hardware configuration.

11.4.5.3 PID Control (S7-1200, S7-1500)

PID_Compact (S7-1200, S7-1500)

New features of PID_Compact (S7-1200, S7-1500)

PID_Compact V2.3

- **Response of the output value when switching from "Inactive" operating mode to "Automatic mode"**
  The new option IntegralResetMode = 4 was added and defined as default. With IntegralResetMode = 4, the integral action is automatically pre-assigned when switching from "Inactive" operating mode to "Automatic mode" so that a control deviation results in a jump of the output value with identical sign.
- **Initialization of the integral action in automatic mode**
  The integral action can be initialized in automatic mode with the tags OverwriteInitialOutputValue and PIDCtrl.PIDInit. This simplifies the use of PID_Compact for override controls.

PID_Compact V2.2

- **Use with S7-1200**
  As of PID_Compact V2.2, the instruction with V2 functionality can also be used on S7-1200 with firmware version 4.0 or higher.

PID_Compact V2.0

- **Reaction to error**
  The reaction to error has been completely overhauled. PID_Compact now reacts in a more fault-tolerant manner in the default setting. This reaction is set when copying PID_Compact V1.X from an S7-1200 CPU to an S7-1500 CPU.

NOTICE
Your system may be damaged. If you use the default setting, PID_Compact remains in automatic mode when the process value limits are exceeded. This may damage your system. It is essential to configure how your controlled system reacts in the event of an error to protect your system from damage.

  The Error parameter indicates if an error is pending. When the error is no longer pending, Error = FALSE. The ErrorBits parameter shows which errors have occurred. Use ErrorAck to acknowledge the errors and warnings without restarting the controller or clearing the integral action. Switching operating modes no longer clears errors that are no longer pending.
  You can configure the reaction to error with SetSubstituteOutput and ActivateRecoverMode.
- **Substitute output value**
  You can configure a substitute output value that is to be output if an error occurs.

- **Switching the operating mode**
  You specify the operating mode at the Mode in/out parameter and use a rising edge at ModeActivate to start the operating mode. The sRet.i_Mode tag has been omitted.
- **Multi-instance capability**
  You can call up PID_Compact as multi-instance DB. No technology object is created in this case and no parameter assignment interface or commissioning interface is available. You must assign parameters for PID_Compact directly in the multi-instance DB and commission it via a watch table.
- **Startup characteristics**
  The operating mode specified at the Mode parameter is also started on a falling edge at Reset and during a CPU cold restart, if RunModeByStartup = TRUE.
- **ENO characteristics**
  ENO is set depending on the operating mode.
  If State = 0, then ENO = FALSE.
  If State ≠ 0, then ENO = TRUE.
- **Setpoint value specification during tuning**
  You configure the permitted fluctuation of the setpoint during tuning at the CancelTuningLevel tag.
- **Value range for output value limits**
  The value 0.0 no longer has to fall within the output value limits.
- **Pre-assigning the integral action**
  Using the tags IntegralResetMode and OverwriteInitialOutputValue, you can determine the pre-assignment of the integral action when switching from "Inactive" operating mode to "Automatic mode".
- **Switching a disturbance variable on**
  You can switch a disturbance variable on at the Disturbance parameter.
- **Default value of PID parameters**
  The following default settings have been changed:
  - Proportional action weighting (PWeighting) from 0.0 to 1.0
  - Derivative action weighting (DWeighting) from 0.0 to 1.0
  - Coefficient for derivative delay (TdFiltRatio) from 0.0 to 0.2
- **Renaming tags**
  The static tags have been given new names that are compatible with PID_3Step.

PID_Compact V1.2

- **Manual mode on CPU startup**
  If ManualEnable = TRUE when the CPU starts, PID_Compact starts in manual mode. A rising edge at ManualEnable is not necessary.
- **Pretuning**
  If the CPU is switched off during pretuning, pretuning starts again when the CPU is switched back on.

PID_Compact V1.1

- **Manual mode on CPU startup**
  When the CPU starts up, PID_Compact only switches to manual mode with a rising edge at ManualEnable. Without rising edge, PID_Compact starts in the last operating mode in which ManualEnable was FALSE.
- **Reaction to reset**
  A rising edge at Reset resets the errors and warnings and clears the integral action. A falling edge at Reset triggers a switchover to the most recently active operating mode.
- **Default of process value high limit**
  The default value of r_Pv_Hlm has been changed to 120.0.
- **Monitoring the sampling time**
  - An error is no longer output when the current sampling time is ≥ 1.5 x current mean value or when the current sampling time is ≤ 0.5 x current mean value. The sampling time may deviate much more in automatic mode.
  - PID_Compact is compatible with FW, V2.0 or higher.
- **Access to tags**
  The following tags can now be used in the user program.
  - i_Event_SUT
  - i_Event_TIR
  - r_Ctrl_Ioutv
- **Troubleshooting**
  PID_Compact now outputs the correct pulses when the shortest ON time is not equal to the shortest OFF time.

Compatibility with CPU and FW (S7-1200, S7-1500)

The following table shows which version of PID_Compact can be used on which CPU.

CPU	FW	PID_Compact
S7-1200	V4.2 or higher	V2.3 V2.2 V1.2
	V4.0 to V4.1	V2.2 V1.2
	V3.x	V1.2 V1.1
	V2.x	V1.2 V1.1
	V1.x	V1.0

CPU	FW	PID_Compact
S7-1500	V2.0 or higher	V2.3 V2.2 V2.1 V2.0
	V1.5 to V1.8	V2.2 V2.1 V2.0
	V1.1	V2.1 V2.0
	V1.0	V2.0

CPU processing time and memory requirement PID_Compact V2.x (S7-1200, S7-1500)

CPU processing time

Typical CPU processing times of the PID_Compact technology object as of Version V2.0, depending on CPU type.

CPU	Typ. CPU processing time PID_Compact V2.x
CPU 1211C ≥ V4.0	300 µs
CPU 1215C ≥ V4.0	300 µs
CPU 1217C ≥ V4.0	300 µs
CPU 1505S ≥ V1.0	45 µs
CPU 1510SP-1 PN ≥ V1.6	85 µs
CPU 1511-1 PN ≥ V1.5	85 µs
CPU 1512SP-1 PN ≥ V1.6	85 µs
CPU 1516-3 PN/DP ≥ V1.5	50 µs
CPU 1518-4 PN/DP ≥ V1.5	4 µs

Memory requirement

Memory requirement of an instance DB of the PID_Compact technology object as of Version V2.0.

	Memory requirement of the instance DB of PID_Compact V2.x
Load memory requirement	Approx. 12000 bytes
Total work memory requirement	788 bytes
Retentive work memory requirement	44 bytes

PID_Compact V2 (S7-1200, S7-1500)

Description of PID_Compact V2 (S7-1200, S7-1500)

Description

The PID_Compact instruction provides a PID controller with integrated tuning for actuators with proportional action.

The following operating modes are possible:

- Inactive
- Pretuning
- Fine tuning
- Automatic mode
- Manual mode
- Substitute output value with error monitoring

For a more detailed description of the operating modes, see the State parameter.

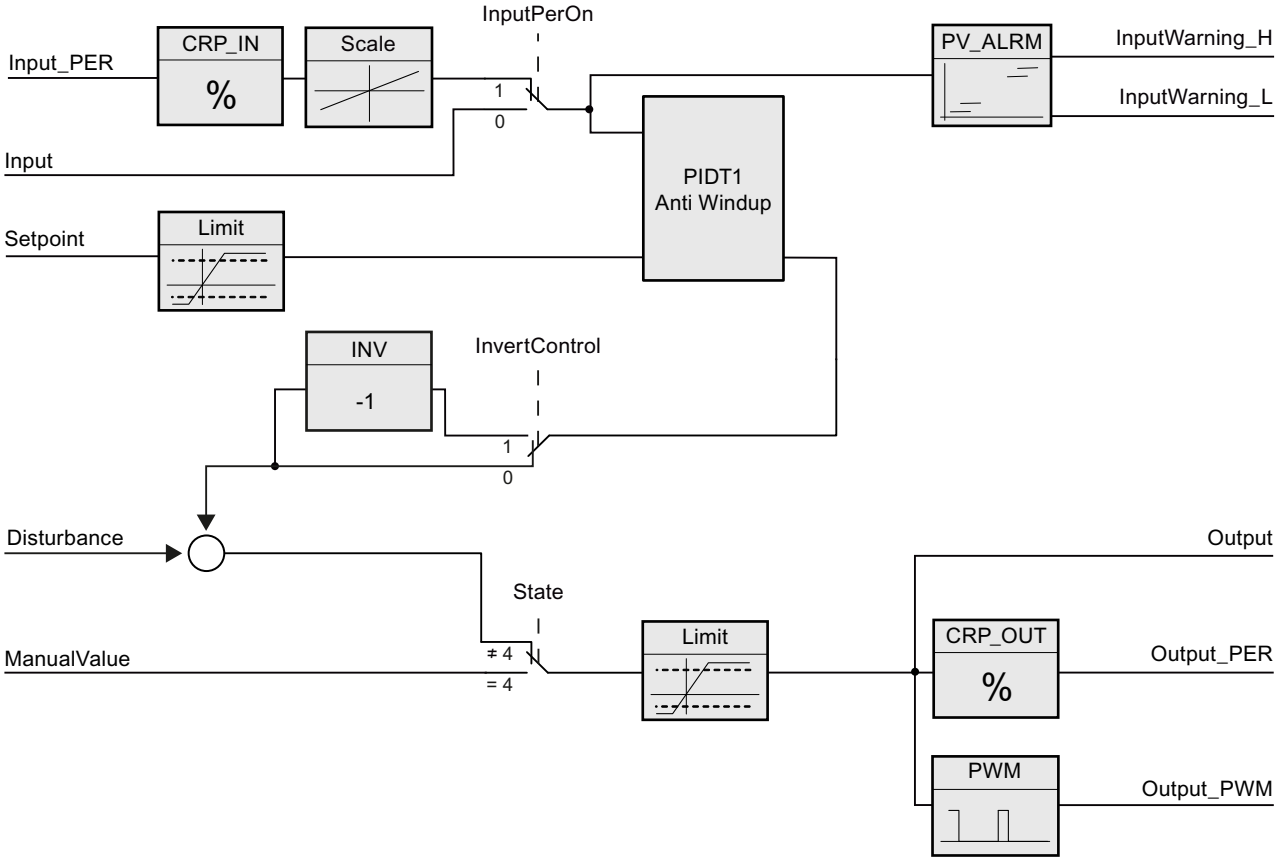
PID algorithm

PID_Compact is a PIDT1 controller with anti-windup and weighting of the proportional and derivative actions. The PID algorithm operates according to the following equation:

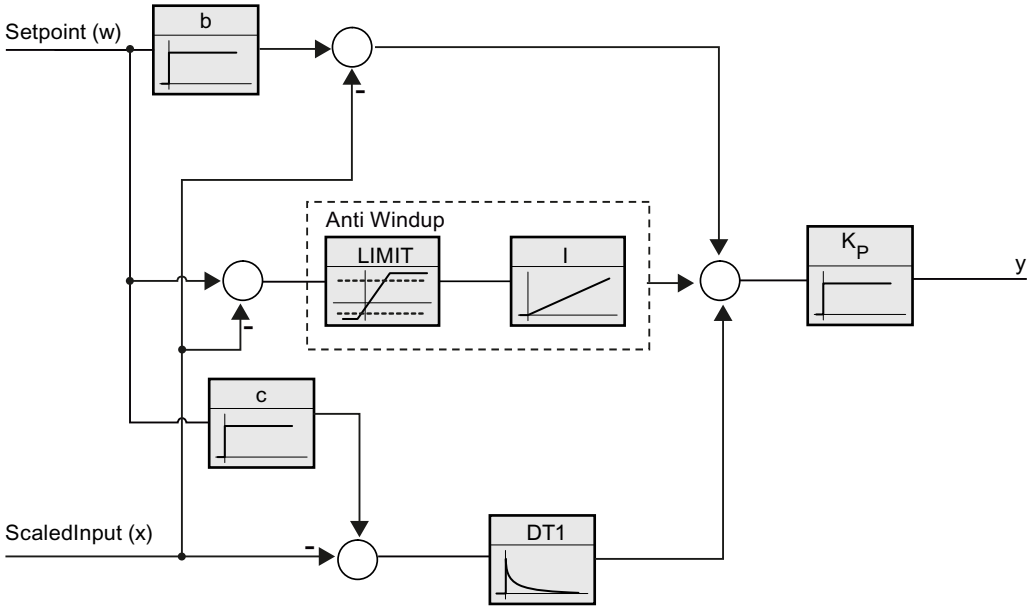
$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
y	Output value of the PID algorithm
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integral action time
T_D	Derivative action time
a	Derivative delay coefficient (derivative delay T1 = a × T_D)
c	Derivative action weighting

Block diagram of PID_Compact



Block diagram of PIDT1 with anti-windup



Call

PID_Compact is called in the constant time scale of a cycle interrupt OB.

If you call PID_Compact as a multi-instance DB, no technology object is created. No parameter assignment interface or commissioning interface is available. You must assign parameters for PID_Compact directly in the multi-instance DB and commission it via a watch table.

Download to device

The actual values of retentive variables are only updated when you download PID_Compact completely.

Downloading technology objects to device (Page 6000)

Startup

When the CPU starts up, PID_Compact starts in the operating mode that is saved in the Mode in/out parameter. To switch to "Inactive" operating mode during startup, set RunModeByStartup = FALSE.

Reaction to error

In automatic mode and during commissioning, the reaction to error depends on the SetSubstituteOutput and ActivateRecoverMode variables. In manual mode, the reaction is independent of SetSubstituteOutput and ActivateRecoverMode. If ActivateRecoverMode = TRUE, the reaction additionally depends on the error that occurred.

SetSubstituteOutput	ActivateRecoverMode	Configuration editor > output value > Set Output to	Reaction
Not relevant	FALSE	Zero (inactive)	Switch to "Inactive" mode (State = 0) The value 0.0 0 is transferred to the actuator.
FALSE	TRUE	Current output value while error is pending	Switch to "Substitute output value with error monitoring" mode (State = 5) The current output value is transferred to the actuator while the error is pending.
TRUE	TRUE	Substitute output value while error is pending	Switch to "Substitute output value with error monitoring" mode (State = 5) The value at SubstituteOutput is transferred to the actuator while the error is pending.

In manual mode, PID_Compact uses ManualValue as output value, unless ManualValue is invalid. If ManualValue is invalid, SubstituteOutput is used. If ManualValue and SubstituteOutput are invalid, Config.OutputLowerLimit is used.

The Error parameter indicates if an error is pending. When the error is no longer pending, Error = FALSE. The ErrorBits parameter shows which errors have occurred. ErrorBits is reset by a rising edge at Reset or ErrorAck.

Mode of operation of PID_Compact V2 (S7-1200, S7-1500)

Monitoring process value limits

You specify the high limit and low limit of the process value in the Config.InputUpperLimit and Config.InputLowerLimit tags. If the process value is outside these limits, an error occurs (ErrorBits = 0001h).

You specify a high and low warning limit of the process value in the Config.InputUpperWarning and Config.InputLowerWarning tags. If the process value is outside these warning limits, a warning occurs (Warning = 0040h), and the InputWarning_H or InputWarning_L output parameter changes to TRUE.

Limiting the setpoint

You specify a high limit and low limit of the setpoint in the Config.SetpointUpperLimit and Config.SetpointLowerLimit tags. PID_Compact automatically limits the setpoint to the process value limits. You can limit the setpoint to a smaller range. PID_Compact checks whether this range falls within the process value limits. If the setpoint is outside these limits, the high or low limit is used as the setpoint, and output parameter SetpointLimit_H or SetpointLimit_L is set to TRUE.

The setpoint is limited in all operating modes.

Limiting the output value

You specify a high limit and low limit of the output value in the Config.OutputUpperLimit and Config.OutputLowerLimit tags. Output, ManualValue and SubstituteOutput are limited to these values. The output value limits must match the control logic.

The valid output value limit values depend on the Output used.

Output	-100.0 to 100.0%
Output_PER	-100.0 to 100.0%
Output_PWM	0.0 to 100.0%

Rule:

OutputUpperLimit > OutputLowerLimit

Note

Use with two or more actuators

PID_Compact is not suitable for use with two or more actuators (for example, in heating/cooling applications), because different actuators need different PID parameters to achieve a good control response. Use PID_Temp for applications with two actuators acting in opposite directions.

Substitute output value

In the event of an error, PID_Compact can output a substitute output value that you define at the SubstituteOutput tag. The substitute output value must be within the output value limits.

Monitoring signal validity

The values of the following parameters are monitored for validity when used:

- Setpoint
- Input
- Input_PER
- Disturbance
- ManualValue
- SubstituteOutput
- Output
- Output_PER
- Output_PWM

Monitoring of the sampling time PID_Compact

Ideally, the sampling time is equivalent to the cycle time of the calling OB. The PID_Compact instruction measures the time interval between two calls. This is the current sampling time. On every switchover of operating mode and during the initial startup, the mean value is formed from the first 10 sampling times. Too great a difference between the current sampling time and this mean value triggers an error (Error = 0800h).

The error occurs during tuning if:

- New mean value >= 1.1 x old mean value
- New mean value <= 0.9 x old mean value

The error occurs in automatic mode if:

- New mean value >= 1.5 x old mean value
- New mean value <= 0.5 x old mean value

If you deactivate the sampling time monitoring (CycleTime.EnMonitoring = FALSE), you can also call PID_Compact in OB1. You must then accept a lower control quality due to the deviating sampling time.

Sampling time of the PID algorithm

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the cycle time. All other functions of PID_Compact are executed at every call.

If you use Output_PWM, the accuracy of the output signal is determined by the ratio of the PID algorithm sampling time to the cycle time of the OB. The cycle time should be at least 10 times the PID algorithm sampling time.

Control logic

An increase of the output value is generally intended to cause an increase in the process value. This is referred to as a normal control logic. For cooling and discharge control systems, it may be necessary to invert the control logic. PID_Compact does not work with negative proportional gain. If InvertControl = TRUE, an increasing control deviation causes a reduction in the output value. The control logic is also taken into account during pretuning and fine tuning.

Input parameters of PID_Compact V2 (S7-1200, S7-1500)

Table 11-67

Parameter	Data type	Default	Description
Setpoint	REAL	0.0	Setpoint of the PID controller in automatic mode
Input	REAL	0.0	A tag of the user program is used as source for the process value. If you are using parameter Input, then Config.InputPerOn = FALSE must be set.
Input_PER	INT	0	An analog input is used as the source of the process value. If you are using parameter Input_PER, then Config.InputPerOn = TRUE must be set.
Disturbance	REAL	0.0	Disturbance variable or precontrol value
ManualEnable	BOOL	FALSE	• A FALSE -> TRUE edge activates "manual mode", while State = 4, Mode remains unchanged. As long as ManualEnable = TRUE, you cannot change the operating mode via a rising edge at ModeActivate or use the commissioning dialog. • A TRUE -> FALSE edge activates the operating mode that is specified by Mode. We recommend that you change the operating mode using ModeActivate only.
ManualValue	REAL	0.0	Manual value This value is used as the output value in manual mode. Values from Config.OutputLowerLimit to Config.OutputUpperLimit are permitted.
ErrorAck	BOOL	FALSE	• FALSE -> TRUE edge ErrorBits and Warning are reset.

Parameter	Data type	Default	Description
Reset	BOOL	FALSE	Restarts the controller. • FALSE -> TRUE edge – Switch to "Inactive" mode – ErrorBits and Warnings are reset. • As long as Reset = TRUE, – PID_Compact remains in "Inactive" mode (State = 0). – You cannot change the operating mode with Mode and ModeActivate or ManualEnable. – You cannot use the commissioning dialog. • TRUE -> FALSE edge – If ManualEnable = FALSE, PID_Compact switches to the operating mode that is saved in Mode. – If Mode = 3, the integral action is treated as configured with the tag IntegralResetMode.
ModeActivate	BOOL	FALSE	• FALSE -> TRUE edge PID_Compact switches to the operating mode that is saved in the Mode parameter.

Output parameters of PID_Compact V2 (S7-1200, S7-1500)

Table 11-68

Parameter	Data type	Default	Description
ScaledInput	REAL	0.0	Scaled process value
The "Output", "Output_PER", and "Output_PWM" outputs can be used concurrently.			
Output	REAL	0.0	Output value in REAL format
Output_PER	INT	0	Analog output value
Output_PWM	BOOL	FALSE	Pulse-width-modulated output value The output value is formed by by variable On and Off times.
SetpointLimit_H	BOOL	FALSE	If SetpointLimit_H = TRUE, the absolute setpoint high limit is reached (Setpoint ≥ Config.SetpointUpperLimit). The setpoint is limited to Config.SetpointUpperLimit .
SetpointLimit_L	BOOL	FALSE	If SetpointLimit_L = TRUE, the absolute setpoint low limit has been reached (Setpoint ≤ Config.SetpointLowerLimit). The setpoint is limited to Config.SetpointLowerLimit .
InputWarning_H	BOOL	FALSE	If InputWarning_H = TRUE, the process value has reached or exceeded the warning high limit.
InputWarning_L	BOOL	FALSE	If InputWarning_L = TRUE, the process value has reached or fallen below the warning low limit.

Parameter	Data type	Default	Description
State	INT	0	The State parameter (Page 4219) shows the current operating mode of the PID controller. You can change the operating mode using the input parameter Mode and a rising edge at ModeActivate. • State = 0: Inactive • State = 1: Pretuning • State = 2: Fine tuning • State = 3: Automatic mode • State = 4: Manual mode • State = 5: Substitute output value with error monitoring
Error	BOOL	FALSE	If Error = TRUE, at least one error message is pending in this cycle.
ErrorBits	DWORD	DW#16#0	The ErrorBits parameter (Page 4223) shows which error messages are pending. ErrorBits is retentive and is reset upon a rising edge at Reset or ErrorAck.

In/out parameters of PID_Compact V2 (S7-1200, S7-1500)

Table 11-69

Parameter	Data type	Default	Description
Mode	INT	4	At Mode, specify the operating mode to which PID_Compact is to switch. Options are: • Mode = 0: Inactive • Mode = 1: Pretuning • Mode = 2: Fine tuning • Mode = 3: Automatic mode • Mode = 4: Manual mode The operating mode is activated by: • Rising edge at ModeActivate • Falling edge at Reset • Falling edge at ManualEnable • Cold restart of CPU if RunModeByStartup = TRUE Mode is retentive. A detailed description of the operating modes can be found in Parameters State and Mode V2 (Page 4219).

See also

Parameters State and Mode V2 (Page 4219)

Static tags of PID_Compact V2 (S7-1200, S7-1500)

You must not change tags that are not listed. These are used for internal purposes only.

Tag	Data type	Default	Description
IntegralResetMode	INT	Up to V2.2: 1, V2.3 or higher: 4	The IntegralResetMode V2 tag (Page 4227) determines how the integral actionPIDCtrl.IntegralSum is pre-assigned when switching from "Inactive" operating mode to "Automatic mode". This setting only works for one cycle. Options are: • IntegralResetMode = 0: Smooth • IntegralResetMode = 1: Delete • IntegralResetMode = 2: Hold • IntegralResetMode = 3: Pre-assign • IntegralResetMode = 4: Like setpoint change (only for PID_Compact with version ≥ 2.3)
OverwriteInitialOutputValue	REAL	0.0	If one of the following conditions is met, the integral action PIDCtrl.IntegralSum is pre-assigned automatically as if Output = OverwriteInitialOutputValue in the previous cycle: • IntegralResetMode = 3 when switching from "Inactive" operating mode to "Automatic mode". • TRUE -> FALSE edge at parameter Reset and parameter Mode = 3 • PIDCtrl.PIDInit = TRUE in "Automatic mode" (available as of PID_Compact version 2.3)
RunModeByStartup	BOOL	TRUE	Activate operating mode at Mode parameter after CPU restart If RunModeByStartup = TRUE, PID_Compact starts in the operating mode saved in the Mode parameter after CPU startup. If RunModeByStartup = FALSE, PID_Compact remains in "Inactive" mode after CPU startup.
LoadBackUp	BOOL	FALSE	If LoadBackUp = TRUE, the last set of PID parameters is reloaded. The set was saved prior to the last tuning. LoadBackUp is automatically set back to FALSE.
PhysicalUnit	INT	0	Unit of measurement of the process value and setpoint, e.g., °C, or °F.
PhysicalQuantity	INT	0	Physical quantity of the process value and setpoint, e.g., temperature.
ActivateRecoverMode	BOOL	TRUE	The Tag ActivateRecoverMode V2 (Page 4225) determines the reaction to error.
Warning	DWORD	0	Tag Warning V2 (Page 4227) shows the warnings since Reset = TRUE or ErrorAck =TRUE. Warning is retentive.
Progress	REAL	0.0	Progress of tuning as a percentage (0.0 - 100.0)
CurrentSetpoint	REAL	0.0	CurrentSetpoint always displays the current setpoint. This value is frozen during tuning.

Tag	Data type	Default	Description
CancelTuningLevel	REAL	10.0	Permissible fluctuation of setpoint during tuning. Tuning is not canceled until: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel
SubstituteOutput	REAL	0.0	Substitute output value When the following conditions are met, the substitute output value is used: • An error has occurred in automatic mode. • SetSubstituteOutput = TRUE • ActivateRecoverMode = TRUE
SetSubstituteOutput	BOOL	TRUE	If SetSubstituteOutput = TRUE and ActivateRecoverMode = TRUE, the substitute output value configured is output as long as an error is pending. If SetSubstituteOutput = FALSE and ActivateRecoverMode = TRUE, the actuator remains at the current output value as long as an error is pending. If ActivateRecoverMode = FALSE, SetSubstituteOutput is not effective. If SubstituteOutput is invalid (ErrorBits = 20000h), the substitute output value cannot be output.
Config.InputPerOn	BOOL	TRUE	If InputPerOn = TRUE, the Input_PER parameter is used. If InputPerOn = FALSE, the Input parameter is used.
Config.InvertControl	BOOL	FALSE	Invert control logic If InvertControl = TRUE, an increasing control deviation causes a reduction in the output value.
Config.InputUpperLimit	REAL	120.0	High limit of the process value Input and Input_PER are monitored to ensure adherence to this limit. At the I/O input, the process value can be a maximum of 18% higher than the standard range (overrange). This pre-assignment ensures that an error is no longer signaled due to a violation of the "Process value high limit". Only a wire-break and a short-circuit are recognized and PID_Compact reacts according to the configured reaction to error. InputUpperLimit > InputLowerLimit
Config.InputLowerLimit	REAL	0.0	Low limit of the process value Input and Input_PER are monitored to ensure adherence to this limit. InputLowerLimit < InputUpperLimit

Tag	Data type	Default	Description
Config.InputUpperWarning	REAL	3.402822e+38	Warning high limit of the process value If you set InputUpperWarning outside the process value limits, the configured absolute process value high limit is used as the warning high limit. If you configure InputUpperWarning within the process value limits, this value is used as the warning high limit. InputUpperWarning > InputLowerWarning InputUpperWarning ≤ InputUpperLimit
Config.InputLowerWarning	REAL	-3.402822e+38	Warning low limit of the process value If you set InputLowerWarning outside the process value limits, the configured absolute process value low limit is used as the warning low limit. If you configure InputLowerWarning within the process value limits, this value is used as the warning low limit. InputLowerWarning < InputUpperWarning InputLowerWarning ≥ InputLowerLimit
Config.OutputUpperLimit	REAL	100.0	High limit of output value For details, see OutputLowerLimit OutputUpperLimit > OutputLowerLimit
Config.OutputLowerLimit	REAL	0.0	Low limit of output value For Output and Output_PER, the range of values from -100.0 to +100.0, including zero, is valid. At -100.0, Output_PER = -27648; at +100.0, Output_PER = 27648. For Output_PWM, the value range 0.0 to +100.0 applies. The output value limits must match the control logic. OutputLowerLimit < OutputUpperLimit
Config.SetpointUpperLimit	REAL	3.402822e+38	High limit of setpoint If you configure SetpointUpperLimit outside the process value limits, the configured process value absolute high limit is used as the setpoint high limit. If you configure SetpointUpperLimit within the process value limits, this value is used as the setpoint high limit.
Config.SetpointLowerLimit	REAL	-3.402822e+38	Low limit of the setpoint If you set SetpointLowerLimit outside the process value limits, the configured process value absolute low limit is used as the setpoint low limit. If you set SetpointLowerLimit within the process value limits, this value is used as the setpoint low limit.
Config.MinimumOnTime	REAL	0.0	The minimum ON time of the pulse width modulation in seconds is rounded to MinimumOnTime = n×CycleTime.Value
Config.MinimumOffTime	REAL	0.0	The minimum OFF time of the pulse width modulation in seconds is rounded to MinimumOffTime = n×CycleTime.Value

Tag	Data type	Default	Description
Config.InputScaling.UpperPointIn	REAL	27648.0	Scaling Input_PER high Input_PER is converted to percent based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn.
Config.InputScaling.LowerPointIn	REAL	0.0	Scaling Input_PER low Input_PER is converted to percent based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn.
Config.InputScaling.UpperPointOut	REAL	100.0	Scaled high process value Input_PER is converted to percent based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn.
Config.InputScaling.LowerPointOut	REAL	0.0	Scaled low process value Input_PER is converted to percent based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn.
CycleTime.StartEstimation	BOOL	TRUE	If CycleTime.StartEstimation = TRUE, the automatic determination of the cycle time is started. CycleTime.StartEstimation = FALSE once measurement is complete.
CycleTime.EnEstimation	BOOL	TRUE	If CycleTime.EnEstimation = TRUE, the PID_Compact sampling time is calculated. If CycleTime.EnEstimation = FALSE, the PID_Compact sampling time is not calculated and you need to correct the configuration of CycleTime.Value manually.
CycleTime.EnMonitoring	BOOL	TRUE	If CycleTime.EnMonitoring = FALSE, the PID_Compact sampling time is not monitored. If it is not possible to execute PID_Compact within the sampling time, no error (ErrorBits=0800h) is output and PID_Compact does not switch to "Inactive" mode.
CycleTime.Value	REAL	0.1	PID_Compact sampling time in seconds CycleTime.Value is determined automatically and is usually equivalent to the cycle time of the calling OB.
CtrlParamsBackUp.Gain	REAL	1.0	Saved proportional gain You can reload values from the CtrlParamsBackUp structure with LoadBackUp = TRUE.
CtrlParamsBackUp.Ti	REAL	20.0	Saved integral action time [s]
CtrlParamsBackUp.Td	REAL	0.0	Saved derivative action time [s]
CtrlParamsBackUp.TdFiltRatio	REAL	0.2	Saved derivative delay coefficient
CtrlParamsBackUp.PWeighting	REAL	1.0	Saved proportional action weighting factor
CtrlParamsBackUp.DWeighting	REAL	1.0	Saved derivative action weighting factor
CtrlParamsBackUp.Cycle	REAL	1.0	Saved sampling time of PID algorithm

Tag	Data type	Default	Description
PIDSelfTune.SUT.CalculateParams	BOOL	FALSE	The properties of the controlled system are saved during tuning. If SUT.CalculateParams = TRUE, the parameters for pretuning are recalculated according to these properties. This enables you to change the parameter calculation method without having to repeat controller tuning. SUT.CalculateParams is set to FALSE after the calculation.
PIDSelfTune.SUT.TuneRule	INT	0	Methods used to calculate parameters during pretuning: • SUT.TuneRule = 0: PID according to Chien, Hrones and Reswick • SUT.TuneRule = 1: PI according to Chien, Hrones and Reswick
PIDSelfTune.SUT.State	INT	0	The SUT.State tag indicates the current phase of pretuning: • State = 0: Initialize pretuning • State = 100: Calculate the standard deviation • State = 200: Find the point of inflection • State = 9900: Pretuning successful • State = 1: Pretuning not successful
PIDSelfTune.TIR.RunIn	BOOL	FALSE	With the RunIn tag, you can specify that fine tuning can also be performed without pretuning. • RunIn = FALSE Pretuning is started when fine tuning is started from inactive or manual mode. If the requirements for pretuning are not met, PID_Compact reacts as if RunIn = TRUE. If fine tuning is started from automatic mode, the system uses the existing PID parameters to control to the setpoint. Only then will fine tuning start. If pretuning is not possible, PID_Compact switches to the mode from which tuning was started. • RunIn = TRUE The pretuning is skipped. PID_Compact tries to reach the setpoint with minimum or maximum output value. This can produce increased overshoot. Fine tuning then starts automatically. RunIn is set to FALSE after fine tuning.
PIDSelfTune.TIR.CalculateParams	BOOL	FALSE	The properties of the controlled system are saved during tuning. If TIR.CalculateParams = TRUE, the parameters for fine tuning are recalculated according to these properties. This enables you to change the parameter calculation method without having to repeat controller tuning. TIR.CalculateParams is set to FALSE after the calculation.

Tag	Data type	Default	Description
PIDSelfTune.TIR.TuneRule	INT	0	Methods used to calculate parameters during fine tuning: • TIR.TuneRule = 0: PID automatic • TIR.TuneRule = 1: PID fast • TIR.TuneRule = 2: PID slow • TIR.TuneRule = 3: Ziegler-Nichols PID • TIR.TuneRule = 4: Ziegler-Nichols PI • TIR.TuneRule = 5: Ziegler-Nichols P
PIDSelfTune.TIR.State	INT	0	The TIR.State tag indicates the current phase of fine tuning: • State = -100 Fine tuning is not possible. Pretuning will be performed first. • State = 0: Initialize fine tuning • State = 200: Calculate the standard deviation • State = 300: Attempt to reach setpoint • State = 400: Attempt to reach setpoint with existing PID parameters (if pretuning was successful) • State = 500: Determine oscillation and calculate parameters • State = 9900: Fine tuning successful • State = 1: Fine tuning not successful
PIDCtrl.IntegralSum	REAL	0.0	Current integral action
PIDCtrl.PIDInit	BOOL	FALSE	PIDCtrl.PIDInit is available as of PID_Compact version 2.3. If PIDCtrl.PIDInit = TRUE in "Automatic mode", the integral action PIDCtrl.IntegralSum is pre-assigned automatically as if Output = OverwriteInitialOutputValue in the previous cycle. This can be used for a Override control with PID_Compact V2 (Page 6040).
Retain.CtrlParams.Gain	REAL	1.0	Active proportional gain To invert the control logic, use the Config.InvertControl tag. Negative values at Gain also invert the control logic. We recommend you use only InvertControl to set the control logic. The control logic is also inverted if InvertControl = TRUE and Gain < 0.0. Gain is retentive.
Retain.CtrlParams.Ti	REAL	20.0	• CtrlParams.Ti > 0.0: Active integral action time • CtrlParams.Ti = 0.0: Integral action is deactivated Ti is retentive.
Retain.CtrlParams.Td	REAL	0.0	• CtrlParams.Td > 0.0: Active derivative action time • CtrlParams.Td = 0.0: Derivative action is deactivated Td is retentive.

Tag	Data type	Default	Description
Retain.CtrlParams.TdFiltRatio	REAL	0.2	Active derivative delay coefficient The derivative delay coefficient delays the effect of the derivative action. Derivative delay = derivative action time × derivative delay coefficient • 0.0: Derivative action is effective for one cycle only and therefore almost not effective. • 0.5: This value has proved useful in practice for controlled systems with **one** dominant time constant. • > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed. TdFiltRatio is retentive.
Retain.CtrlParams.PWeighting	REAL	1.0	Active proportional action weighting The proportional action may weaken with changes to the setpoint. Values from 0.0 to 1.0 are applicable. • 1.0: Proportional action for setpoint change is fully effective • 0.0: Proportional action for setpoint change is not effective The proportional action is always fully effective when the process value is changed. PWeighting is retentive.
Retain.CtrlParams.DWeighting	REAL	1.0	Active derivative action weighting The derivative action may weaken with changes to the setpoint. Values from 0.0 to 1.0 are applicable. • 1.0: Derivative action is fully effective upon setpoint change • 0.0: Derivative action is not effective upon setpoint change The derivative action is always fully effective when the process value is changed. DWeighting is retentive.
Retain.CtrlParams.Cycle	REAL	1.0	Active sampling time of the PID algorithm CtrlParams.Cycle is calculated during tuning and rounded to an integer multiple of CycleTime.Value. CtrlParams.Cycle is used as time period of the pulse width modulation. Cycle is retentive.

Note

Change the tags listed in this table in "Inactive" mode to prevent malfunction of the PID controller.

See also

Tag ActivateRecoverMode V2 (Page 4225)

Tag Warning V2 (Page 4227)

Downloading technology objects to device (Page 6000)

Changing the PID_Compact V2 interface (S7-1200, S7-1500)

The following table shows what has changed in the PID_Compact instruction interface.

PID_Compact V1	PID_Compact V2	Change
Input_PER	Input_PER	Data type from Word to Int
	Disturbance	New
	ErrorAck	New
	ModeActivate	New
Output_PER	Output_PER	Data type from Word to Int
Error	ErrorBits	Renamed
	Error	New
	Mode	New
sb_RunModeByStartup	RunModeByStartup	Function
	IntegralResetMode	
	OverwriteInitialOutputValue	New
	SetSubstituteOutput	New
	CancelTuningLevel	New
	SubstituteOutput	New

The following table shows which variables have been renamed.

PID_Compact V1.x	PID_Compact V2
sb_GetCycleTime	CycleTime.StartEstimation
sb_EnCyclEstimation	CycleTime.EnEstimation
sb_EnCyclMonitoring	CycleTime.EnMonitoring
sb_RunModeByStartup	RunModeByStartup
si_Unit	PhysicalUnit
si_Type	PhysicalQuantity
sd_Warning	Warning
sBackUp.r_Gain	CtrlParamsBackUp.Gain
sBackUp.r_Ti	CtrlParamsBackUp.Ti
sBackUp.r_Td	CtrlParamsBackUp.Td
sBackUp.r_A	CtrlParamsBackUp.TdFiltRatio
sBackUp.r_B	CtrlParamsBackUp.PWeighting
sBackUp.r_C	CtrlParamsBackUp.DWeighting
sBackUp.r_Cycle	CtrlParamsBackUp.Cycle
sPid_Calc.r_Cycle	CycleTime.Value

PID_Compact V1.x	PID_Compact V2
sPid_Calc.b_RunIn	PIDSelfTune.TIR.RunIn
sPid_Calc.b_CalcParamSUT	PIDSelfTune.SUT.CalculateParams
sPid_Calc.b_CalcParamTIR	PIDSelfTune.TIR.CalculateParams
sPid_Calc.i_CtrlTypeSUT	PIDSelfTune.SUT.TuneRule
sPid_Calc.i_CtrlTypeTIR	PIDSelfTune.TIR.TuneRule
sPid_Calc.r_Progress	Progress
sPid_Cmpt.r_Sp_Hlm	Config.SetpointUpperLimit
sPid_Cmpt.r_Sp_Llm	Config.SetpointLowerLimit
sPid_Cmpt.r_Pv_Norm_IN_1	Config.InputScaling.LowerPointIn
sPid_Cmpt.r_Pv_Norm_IN_2	Config.InputScaling.UpperPointIn
sPid_Cmpt.r_Pv_Norm_OUT_1	Config.InputScaling.LowerPointOut
sPid_Cmpt.r_Pv_Norm_OUT_2	Config.InputScaling.UpperPointOut
sPid_Cmpt.r_Lmn_Hlm	Config.OutputUpperLimit
sPid_Cmpt.r_Lmn_Llm	Config.OutputLowerLimit
sPid_Cmpt.b_Input_PER_On	Config.InputPerOn
sPid_Cmpt.b_LoadBackUp	LoadBackUp
sPid_Cmpt.b_InvCtrl	Config.InvertControl
sPid_Cmpt.r_Lmn_Pwm_PPTm	Config.MinimumOnTime
sPid_Cmpt.r_Lmn_Pwm_PBTm	Config.MinimumOffTime
sPid_Cmpt.r_Pv_Hlm	Config.InputUpperLimit
sPid_Cmpt.r_Pv_Llm	Config.InputLowerLimit
sPid_Cmpt.r_Pv_HWrn	Config.InputUpperWarning
sPid_Cmpt.r_Pv_LWrn	Config.InputLowerWarning
sParamCalc.i_Event_SUT	PIDSelfTune.SUT.State
sParamCalc.i_Event_TIR	PIDSelfTune.TIR.State
sRet.i_Mode	sRet.i_Mode has been omitted. The operating mode is changed using Mode and ModeActivate.
sRet.r_Ctrl_Gain	Retain.CtrlParams.Gain
sRet.r_Ctrl_Ti	Retain.CtrlParams.Ti
sRet.r_Ctrl_Td	Retain.CtrlParams.Td
sRet.r_Ctrl_A	Retain.CtrlParams.TdFiltRatio
sRet.r_Ctrl_B	Retain.CtrlParams.PWeighting
sRet.r_Ctrl_C	Retain.CtrlParams.DWeighting
sRet.r_Ctrl_Cycle	Retain.CtrlParams.Cycle

Parameters State and Mode V2 (S7-1200, S7-1500)

Correlation of the parameters

The State parameter shows the current operating mode of the PID controller. You cannot change the State parameter.

With a rising edge at ModeActivate, PID_Compact switches to the operating mode saved in the Mode in-out parameter.

When the CPU is switched on or switches from Stop to RUN mode, PID_Compact starts in the operating mode that is saved in the Mode parameter. To leave PID_Compact in "Inactive" mode, set RunModeByStartup = FALSE.

Meaning of values

State / Mode	Description of operating mode
0	Inactive In "Inactive" operating mode, the output value 0.0 is always output, regardless of Config.OutputUpperLimit and Config.OutputLowerLimit. Pulse width modulation is off.
1	Pretuning The pretuning determines the process response to a jump change of the output value and searches for the point of inflection. The PID parameters are calculated from the maximum rate of rise and dead time of the controlled system. You obtain the best PID parameters when you perform pretuning and fine tuning. Pretuning requirements: • Inactive (State = 0), manual mode (State = 4), or automatic mode (State = 3) • ManualEnable = FALSE • Reset = FALSE • The process value must not be too close to the setpoint. |Setpoint - Input| > 0.3 * | Config.InputUpperLimit - Config.InputLowerLimit| and |Setpoint - Input| > 0.5 * |Setpoint| • The setpoint and the process value lie within the configured limits. The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise. The setpoint is frozen in the CurrentSetpoint tag. Tuning is canceled when: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel Before the PID parameters are recalculated, they are backed up and can be reactivated with LoadBackUp. The controller switches to automatic mode following successful pretuning. If pretuning is unsuccessful, the switchover of the operating mode is dependent on ActivateRecoverMode. The phase of pretuning is indicated with PIDSelfTune.SUT.State.

State / Mode	Description of operating mode
2	Fine tuning Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are recalculated based on the amplitude and frequency of this oscillation. PID parameters from fine tuning usually have better master control and disturbance characteristics than PID parameters from pretuning. You obtain the best PID parameters when you perform pretuning and fine tuning. PID_Compact automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. The setpoint is frozen in the CurrentSetpoint tag. Tuning is canceled when: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel Before the PID parameters are recalculated, they are backed up and can be reactivated with LoadBackUp. Requirements for fine tuning: • No disturbances are expected. • The setpoint and the process value lie within the configured limits. • ManualEnable = FALSE • Reset = FALSE • Automatic (State = 3), inactive (State = 0) or manual (State = 4) mode Fine tuning proceeds as follows when started from: • Automatic mode (State = 3) Start fine tuning from automatic mode if you wish to improve the existing PID parameters through tuning. PID_Compact controls the system using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start. • Inactive (State = 0) or manual mode (State = 4) If the requirements for pretuning are met, pretuning is started. The determined PID parameters will be used for control until the control loop has stabilized and the requirements for fine tuning have been met. If the process value for pretuning is already too near the setpoint or PIDSelfTune.TIR.RunIn = TRUE, an attempt is made to reach the setpoint with the minimum or maximum output value. This can produce increased overshoot. Only then will fine tuning start. The controller switches to automatic mode following successful fine tuning. If fine tuning is unsuccessful, the switchover of the operating mode is dependent on ActivateRecoverMode. The "Fine tuning" phase is indicated with PIDSelfTune.TIR.State.
3	Automatic mode In automatic mode, PID_Compact corrects the controlled system in accordance with the parameters specified. The controller switches to automatic mode if one the following requirements is fulfilled: • Pretuning successfully completed • Fine tuning successfully completed • Changing of the Mode in-out parameter to the value 3 and a rising edge at ModeActivate. The switchover from automatic mode to manual mode is only bumpless if carried out in the commissioning editor. The ActivateRecoverMode tag is taken into consideration in automatic mode.

State / Mode	Description of operating mode
4	Manual mode In manual mode, you specify a manual output value in the ManualValue parameter. You can also activate this operating mode using ManualEnable = TRUE. We recommend that you change the operating mode using Mode and ModeActivate only. The switchover from manual mode to automatic mode is bumpless. Manual mode is also possible when an error is pending.
5	Substitute output value with error monitoring The control algorithm is deactivated. The SetSubstituteOutput tag determines which output value is output in this operating mode. • SetSubstituteOutput = FALSE: Last valid output value • SetSubstituteOutput = TRUE: Substitute output value You cannot activate this operating mode using Mode = 5. In the event of an error, it is activated instead of "Inactive" operating mode if all the following conditions are met: • Automatic mode (Mode = 3) • ActivateRecoverMode = TRUE • One or more errors have occurred in which ActivateRecoverMode is effective. As soon as the errors are no longer pending, PID_Compact switches back to automatic mode.

ENO characteristics

If State = 0, then ENO = FALSE.

If State ≠ 0, then ENO = TRUE.

Automatic switchover of operating mode during commissioning

Automatic mode is activated following successful pretuning or fine tuning. The following table shows how Mode and State change during successful pretuning.

Cycle no.	Mode	State	Action
0	4	4	Set Mode = 1
1	1	4	Set ModeActivate = TRUE
1	4	1	Value of State is saved in Mode parameter Pretuning is started
n	4	1	Pretuning successfully completed
n	3	3	Automatic mode is started

PID_Compact automatically switches the operating mode in the event of an error. The following table shows how Mode and State change during pretuning with errors.

Cycle no.	Mode	State	Action
0	4	4	Set Mode = 1
1	1	4	Set ModeActivate = TRUE
1	4	1	Value of State is saved in Mode parameter Pretuning is started

Cycle no.	Mode	State	Action
n	4	1	Pretuning canceled
n	4	4	Manual mode is started

If ActivateRecoverMode = TRUE, the operating mode that is saved in the Mode parameter is activated. At the start of pretuning or fine tuning, PID_Compact has saved the value of State in the Mode in/out parameter. PID_Compact therefore switches to the operating mode from which tuning was started.

If ActivateRecoverMode = FALSE, the system switches to "Inactive" operating mode.

See also

Output parameters of PID_Compact V2 (Page 4209)

Parameter ErrorBits V2 (S7-1200, S7-1500)

If several errors are pending simultaneously, the values of the ErrorBits are displayed with binary addition. The display of ErrorBits = 0003h, for example, indicates that the errors 0001h and 0002h are pending simultaneously.

In manual mode, PID_Compact uses ManualValue as output value. The exception is Errorbits = 10000h.

ErrorBits (DW#16#...)	Description
0000	There is no error.
0001	The "Input" parameter is outside the process value limits. • Input > Config.InputUpperLimit or • Input < Config.InputLowerLimit If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact remains in automatic mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact switches to the operating mode that is saved in the Mode parameter.
0002	Invalid value at "Input_PER" parameter. Check whether an error is pending at the analog input. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact outputs the configured substitute output value. As soon as the error is no longer pending, PID_Compact switches back to automatic mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact switches to the operating mode that is saved in the Mode parameter.
0004	Error during fine tuning. Oscillation of the process value could not be maintained. If ActivateRecoverMode = TRUE before the error occurred, PID_Compact cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0008	Error at start of pretuning. The process value is too close to the setpoint. Start fine tuning. If ActivateRecoverMode = TRUE before the error occurred, PID_Compact cancels the tuning and switches to the operating mode that is saved in the Mode parameter.

ErrorBits (DW#16#...)	Description
0010	The setpoint was changed during tuning. You can set the permitted fluctuation of the setpoint at the CancelTuningLevel tag. If ActivateRecoverMode = TRUE before the error occurred, PID_Compact cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0020	Pretuning is not permitted during fine tuning. If ActivateRecoverMode = TRUE before the error occurred, PID_Compact remains in fine tuning mode.
0080	Error during pretuning. The output value limits are not configured correctly or the actual value does not react as expected. Check whether the limits of the output value are configured correctly and match the control logic. Also make sure that the actual value does not oscillate strongly before starting pretuning. If ActivateRecoverMode = TRUE before the error occurred, PID_Compact cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0100	Error during fine tuning resulted in invalid parameters. If ActivateRecoverMode = TRUE before the error occurred, PID_Compact cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0200	Invalid value at "Input" parameter: Value has an invalid number format. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact outputs the configured substitute output value. As soon as the error is no longer pending, PID_Compact switches back to automatic mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact switches to the operating mode that is saved in the Mode parameter.
0400	Calculation of output value failed. Check the PID parameters. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact outputs the configured substitute output value. As soon as the error is no longer pending, PID_Compact switches back to automatic mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact switches to the operating mode that is saved in the Mode parameter.
0800	Sampling time error: PID_Compact is not called within the sampling time of the cyclic interrupt OB. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact remains in automatic mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact switches to the operating mode that is saved in the Mode parameter. If this error occurred during simulation with PLCSIM, see the notes under Simulating PID_Compact V2 with PLCSIM (Page 6044).
1000	Invalid value at "Setpoint" parameter: Value has an invalid number format. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact outputs the configured substitute output value. As soon as the error is no longer pending, PID_Compact switches back to automatic mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Compact switches to the operating mode that is saved in the Mode parameter.
10000	Invalid value at ManualValue parameter. Value has an invalid number format. If ActivateRecoverMode = TRUE before an error occurred, PID_Compact uses SubstituteOutput as the output value. As soon as you specify a valid value in ManualValue, PID_Compact uses it as the output value.

ErrorBits (DW#16#...)	Description
20000	Invalid value at SubstituteOutput tag. Value has an invalid number format. PID_Compact uses the output value low limit as the output value. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_Compact switches back to automatic mode.
40000	Invalid value at Disturbance parameter. Value has an invalid number format. If automatic mode was active and ActivateRecoverMode = TRUE before the error occurred, Disturbance is set to zero. PID_Compact remains in automatic mode. If pretuning or fine tuning mode was active and ActivateRecoverMode = TRUE before the error occurred, PID_Compact switches to the operating mode saved in the Mode parameter. If Disturbance in the current phase has no effect on the output value, tuning is not be canceled.

Tag ActivateRecoverMode V2 (S7-1200, S7-1500)

The ActivateRecoverMode tag determines the reaction to error. The Error parameter indicates if an error is pending. When the error is no longer pending, Error = FALSE. The ErrorBits parameter shows which errors have occurred.

Automatic mode

NOTICE
Your system may be damaged. If ActivateRecoverMode = TRUE, PID_Compact remains in automatic mode even if there is an error and the process limit values are exceeded. This may damage your system. It is essential to configure how your controlled system reacts in the event of an error to protect your system from damage.

ActivateRecoverMode	Description
FALSE	PID_Compact automatically switches to "Inactive" mode in the event of an error. The controller is only activated by a falling edge at Reset or a rising edge at ModeActivate.
TRUE	If errors occur frequently in automatic mode, this setting has a negative effect on the control response, because PID_Compact switches between the calculated output value and the substitute output value at each error. In this case, check the ErrorBits parameter and eliminate the cause of the error. If one or more of the following errors occur, PID_Compact stays in automatic mode: • 0001h: The "Input" parameter is outside the process value limits. • 0800h: Sampling time error • 40000h: Invalid value at parameter Disturbance. If one or more of the following errors occur, PID_Compact switches to "Substitute output value with error monitoring" mode: • 0002h: Invalid value at Input_PER parameter. • 0200h: Invalid value at Input parameter. • 0400h: Calculation of output value failed. • 1000h: Invalid value at Setpoint parameter. If the following error occurs, PID_Compact switches to "Substitute output value with error monitoring" mode and moves the actuator to Config.OutputLowerLimit: • 20000h: Invalid value at SubstituteOutput tag. Value has an invalid number format. This characteristics are independent of SetSubstituteOutput. As soon as the errors are no longer pending, PID_Compact switches back to automatic mode.

Pretuning and fine tuning

ActivateRecoverMode	Description
FALSE	PID_Compact automatically switches to "Inactive" mode in the event of an error. The controller is only activated by a falling edge at Reset or a rising edge at ModeActivate.
TRUE	If the following error occurs, PID_Compact remains in the active mode: • 0020h: Pretuning is not permitted during fine tuning. The following errors are ignored: • 10000h: Invalid value at ManualValue parameter. • 20000h: Invalid value at SubstituteOutput tag. When any other error occurs, PID_Compact cancels the tuning and switches to the mode from which tuning was started.

Manual mode

ActivateRecoverMode is not effective in manual mode.

Tag Warning V2 (S7-1200, S7-1500)

If several warnings are pending simultaneously, the values of the Warning tag are displayed with binary addition. The display of warning 0003h, for example, indicates that the warnings 0001h and 0002h are pending simultaneously.

Warning (DW#16#....)	Description
0000	No warning pending.
0001	The point of inflection was not found during pretuning.
0004	The setpoint was limited to the configured limits.
0008	Not all the necessary controlled system properties were defined for the selected method of calculation. Instead, the PID parameters were calculated using the TIR.TuneRule = 3 method.
0010	The operating mode could not be changed because Reset = TRUE or ManualEnable = TRUE.
0020	The cycle time of the calling OB limits the sampling time of the PID algorithm. Improve results by using shorter OB cycle times.
0040	The process value exceeded one of its warning limits.
0080	Invalid value at Mode. The operating mode is not switched.
0100	The manual value was limited to the limits of the controller output.
0200	The specified rule for tuning is not supported. No PID parameters are calculated.
1000	The substitute output value cannot be reached because it is outside the output value limits.

The following warnings are deleted as soon as the cause is eliminated:

- 0001h
- 0004h
- 0008h
- 0040h
- 0100h

All other warnings are cleared with a rising edge at Reset or ErrorAck.

IntegralResetMode V2 tag (S7-1200, S7-1500)

The IntegralResetMode tag determines how the integral action PIDCtrl.IntegralSum is pre-assigned:

- When switching from "Inactive" operating mode to "Automatic mode"
- With edge TRUE -> FALSE at parameter Reset and parameter Mode = 3

This setting only works for one cycle and is only effective if the integral action is activated (Retain.CtrlParams.Ti > 0.0 tag).

IntegralReset-Mode	Description
0	Smooth The value of PIDCtrl.IntegralSum is pre-assigned so that the switchover is bumpless, which means "Automatic mode" starts with the output value = 0.0 (parameter Output) and there is no jump of the output value regardless of the control deviation (setpoint – actual value).
1	Delete We recommend setting the weighting of the proportional action (Retain.CtrlParams.PWeighting) to 1.0 if this option is used. The value of PIDCtrl.IntegralSum is deleted. Any control deviation will cause a jump change of the output value. The direction of the output value jump depends on the configured weighting of the proportional action (Retain.CtrlParams.PWeighting tag) and the control deviation: • Proportional action weighting = 1.0: Output value jump and control deviation have identical signs. Example: If the actual value value is smaller than the setpoint (positive control deviation), the output value jumps to a positive value. • Proportional action weighting < 1.0: For large control deviations, the output value jump and control deviation have identical signs. Example: If the actual value is much smaller than the setpoint (positive control deviation), the output value jumps to a positive value. For small control deviations, the output value jump and control deviation have different signs. Example: If the actual value value is just below the setpoint (positive control deviation), the output value jumps to a negative value. This is usually not desirable, because it results in a temporary increase in the control deviation. The smaller the configured weighting of the proportional action, the greater the control deviation must be to receive an output value jump with identical sign. We recommend setting the weighting of the proportional action (Retain.CtrlParams.PWeighting) to 1.0 when this option is used. Otherwise, you may experience the undesirable behavior described for small control deviations. Alternatively, you can also use IntegralResetMode = 4. This option guarantees identical signs of the output value jump and control deviation independent of the configured weighting of the proportional action and the control deviation.
2	Hold The value of PIDCtrl.IntegralSum is not changed. You can define a new value using the user program.
3	Pre-assign The value of PIDCtrl.IntegralSum is automatically pre-assigned as if Output = OverwriteInitialOutputValue in the last cycle.
4	Like setpoint change (only for PID_Compact with version ≥ 2.3) The value of PIDCtrl.IntegralSum is automatically pre-assigned so that a similar output value jump results as for a PI controller in automatic mode in case of a setpoint change from the current actual value to the current setpoint. Any control deviation will cause a jump of the output value. Output value jump and control deviation have identical signs. Example: If the actual value value is smaller than the setpoint (positive control deviation), the output value jumps to a positive value. This is independent of the configured weighting of the proportional action and the control deviation.

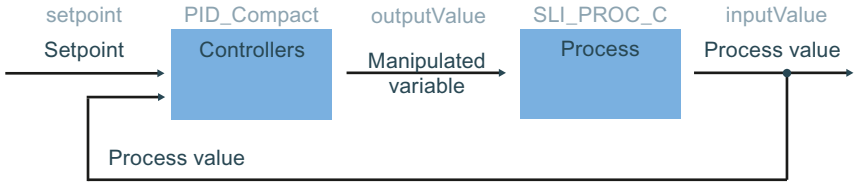
If IntegralResetMode is assigned a value outside the valid value range, PID_Compact behaves as with the pre-assignment of IntegralResetMode:

- PID_Compact up to V2.2: IntegralResetMode = 1
- PID_Compact V2.3 and higher: IntegralResetMode = 4

All statements made above regarding the sign of the output value jump are based on a normal control logic (Config.InvertControl = FALSE tag). With an inverted control logic (Config.InvertControl = TRUE), the output value jump will have a reverse sign.

Sample program for PID_Compact (S7-1200, S7-1500)

In the following example, you are controlling temperature values with the technology object of the instruction "PID_Compact". The temperature values are simulated based on a block which simulates a delay element of the third order (PT3 element). The PID parameters of the technology object can be set automatically via the pretuning.



Data storage

Create seven tags in a global data block for storage of the interconnection data.

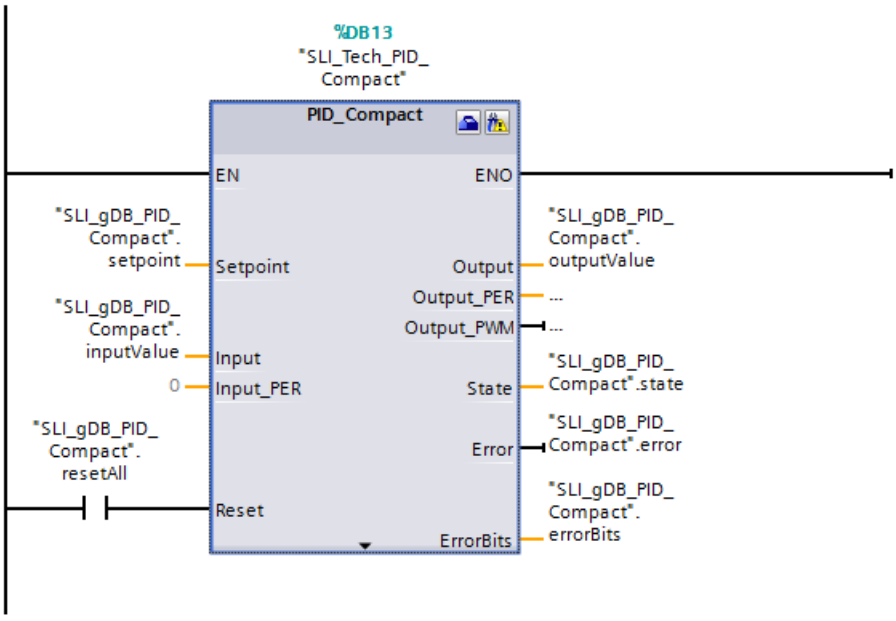
SLI_gDB_PID_Compact

		Name	Data type	Start value
1		Static		
2		setpoint	Real	75.0
3		inputValue	Real	0.0
4		outputValue	Real	0.0
5		state	Int	0
6		error	Bool	false
7		errorBits	DWord	16#0
8		resetAll	Bool	false

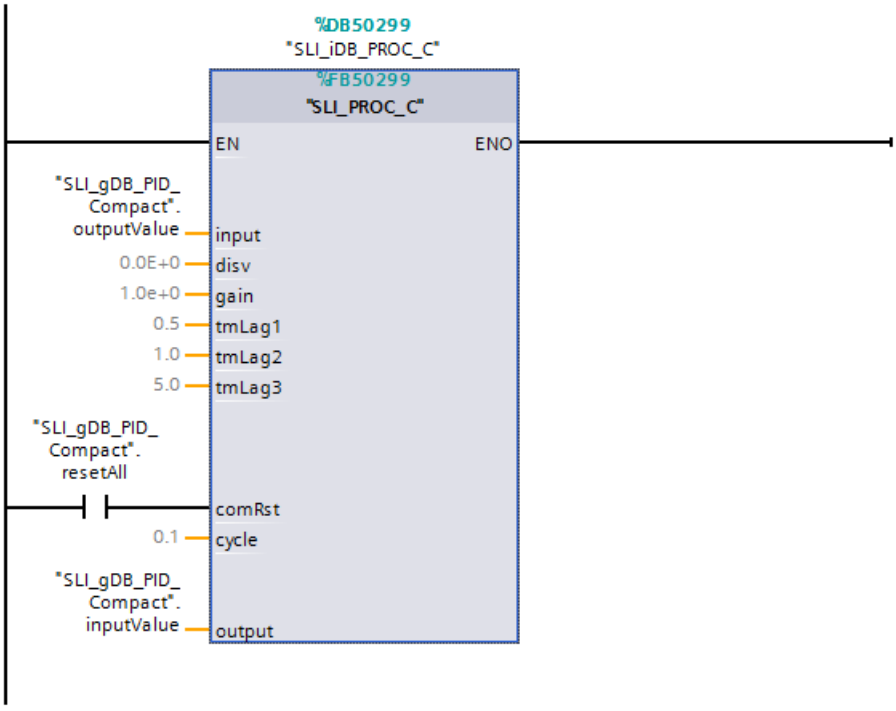
Interconnection of the parameters

You call the following interconnections in a cyclic interrupt OB.

Network 1: You interconnect the parameters of the instruction "PID_Compact" as follows.

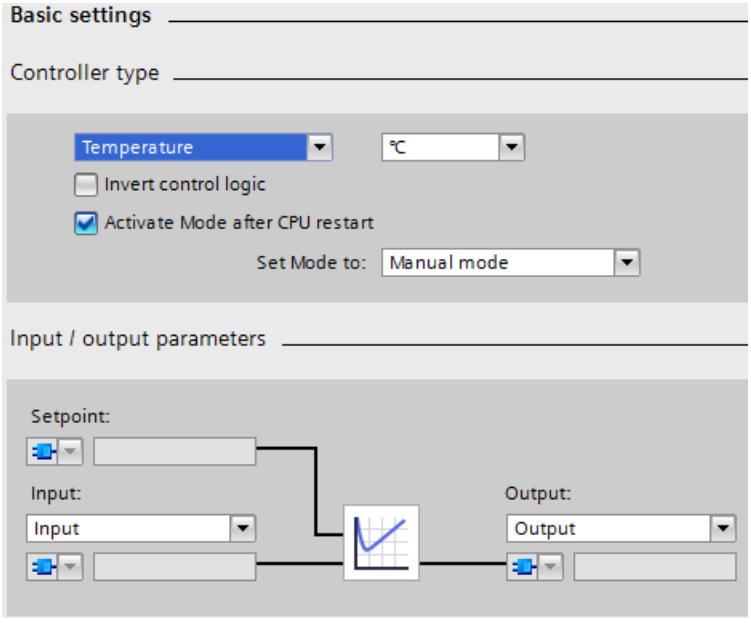


Network 2: You interconnect the parameters of the block simulating the temperature values "SLI_PROC_C" as follows.



Technology object

You configure the technology object with the properties of the instruction "PID_Compact" or by using the path Technology object > Configuration. The controller type and the input/output parameters are important for the example. With the controller type, you make a preselection for the unit of the value to the controlled. In this example, "Temperature" with the unit "°C" is used as controller type. The parameters of the "PID_Compact" are already interconnected with global tags. Therefore, the information on use of the parameters Input and Output is sufficient.



Procedure for starting the control

After the download to the CPU the PID_Compact is in manual mode with manual value 0.0. To start the control, follow these steps:

1. Open the Commissioning of the technology object "SLI_Tech_PID_Compact".
2. Click the "Start" button in the "Measurement" area.



Measurement starts and PID_Compact can be activated.

3. Pretuning is selected.
   Click the "Start" button in the "Tuning mode" area.
   A pretuning is performed. The PID parameters are automatically adjusted to the process. After the completion of the pretuning PID_Compact switches to automatic mode.

Note

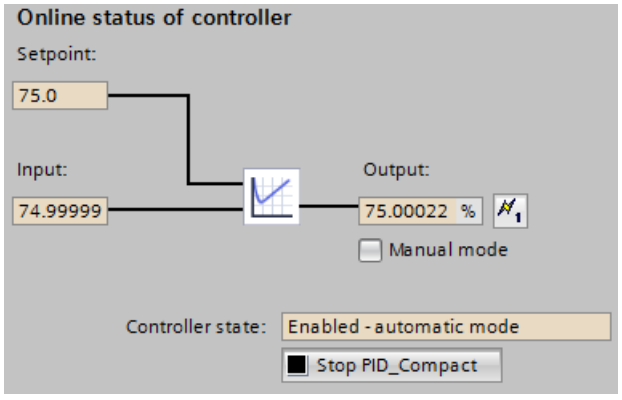
Alternative to start PID_Compact

Alternatively, you can switch PID_Compact to automatic mode in the "Online status of controller" area with the "Stop PID_Compact" / "Start PID_Compact" without pretuning. In this case the controller uses default values for the PID parameters and shows a worse controller behavior for the application case.

Procedure for stopping control

To stop and exit PID_Compact and the program, follow these steps:

1. Click the "Stop PID_Compact" button in the technology object "SLI_Tech_PID_Compact" in the "Online status of controller" area.



The instruction "PID_Compact" exits the control and outputs the value "0.0" as manipulated variable.

2. Click the "Stop" button in the "Measurement" area.
3. To set the process value immediately to the value "0.0", follow these steps:
   In the block "SLI_OB_PID_Compact", set the "resetAll" tag to the value "TRUE", and then to the value "FALSE".

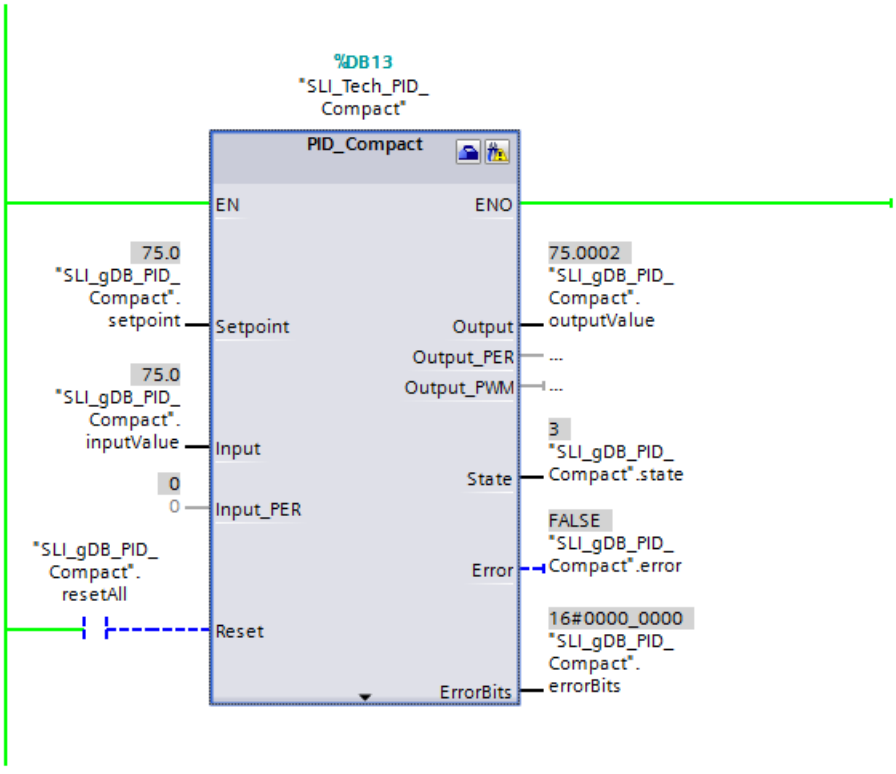
"PID_Compact" instruction

The setpoint for the temperature that is to be controlled is specified at the parameter Setpoint ("setpoint"). The control is started when the instruction "PID_Compact" was started with the technology object. The instruction "PID_Compact" outputs a manipulated variable at the output parameter Output ("outputValue"). The process value of the temperature is transferred to the instruction "PID_Compact" with the input parameter Input ("inputValue").

The instruction "PID_Compact" adjusts the manipulated variable ("outputValue") depending on the history of the difference between setpoint ("setpoint") and process value ("inputValue") . The process is repeated so that the process value ("inputValue") approaches the setpoint ("setpoint") through the manipulated variable ("outputValue").

The current operating mode of the instruction "PID_Compact" is displayed at the output parameter State ("state"). After pretuning (the value of "state" is "1"), the PID_Compact switches to automatic mode (the value is "3").

The output parameter Error ("error") currently shows that no error is pending. The output parameter ErrorBits ("errorBits") provides information on the error type in case of error. If an error occurs, this can be acknowledged in the technology object, in the optimization status area with the "ErrorAck" button.

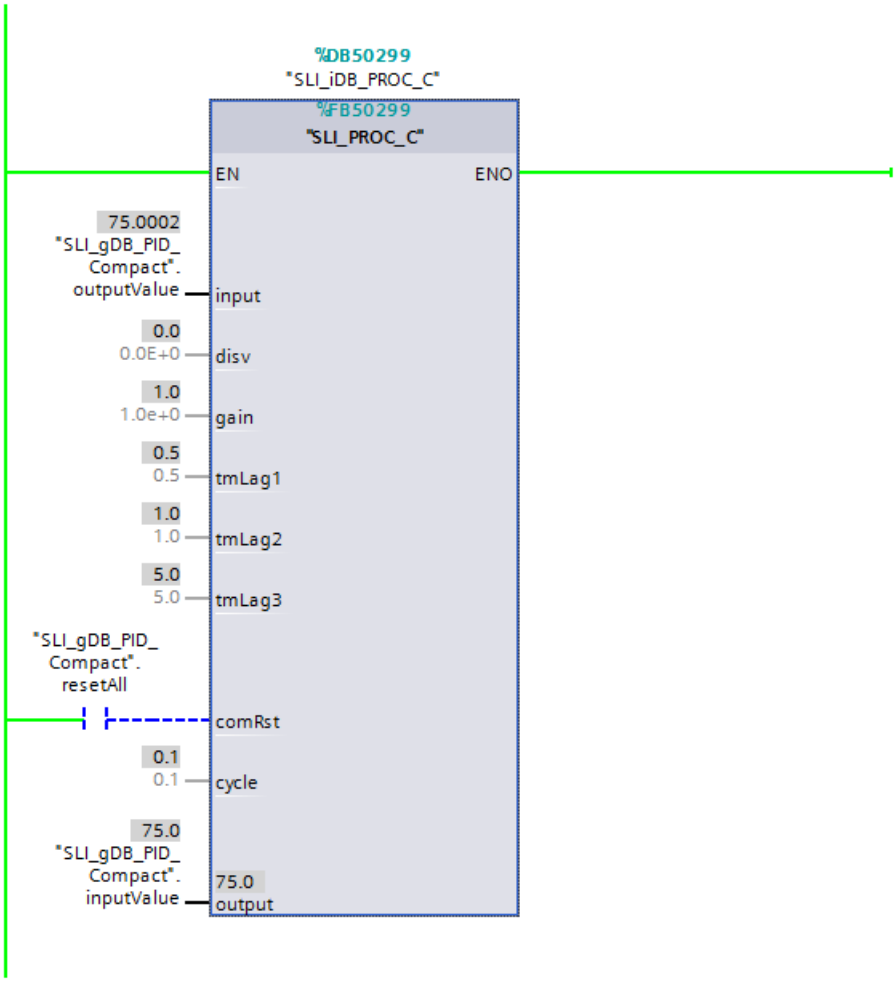


"SLI_PROC_C" block

The "SLI_PROC_C" block simulates the process value ("inputValue") of the rising temperature of a plant. The block "SLI_PROC_C" contains the manipulated variable of the controller ("outputValue) and simulates the temperature behavior of the process. This temperature is fed as process value ("inputValue") into the controller.

A change in the values of the "resetAll" tag (of the comRst parameter) has the following effects:

Parameter comRst ("resetAll")	The instruction "PID_Compact" is running	The instruction "PID_Compact" was stopped
comRst ("resetAll") remains set to the value "FALSE"	The "SLI_PROC_C" block outputs a new process value ("inputValue") based on a manipulated variable ("outputValue").	The "SLI_PROC_C" block does not receive a manipulated variable > "0.0", but it still outputs a new process value > "0.0".
comRst ("resetAll"): Change from "FALSE" to the value "TRUE"	Both manipulated variable ("outputValue") and output process value ("inputValue") are reset to "0.0".	The output process value ("inputValue") / the temperature of the "SLI_PROC_C" block is reset to "0.0".
comRst ("resetAll"): Change from "TRUE" to the value "FALSE"	Temperature control starts again.	The output process value / the temperature ("inputValue") remains "0.0".



Program code

You can find additional information about the program code for the above-named example under the keyword "Sample Library for Instructions".

PID_Compact V1 (S7-1200)

Description of PID_Compact V1 (S7-1200)

Description

The PID_Compact instruction provides a PID controller with integrated tuning for automatic and manual mode.

Call

PID_Compact is called in the constant interval of the cycle time of the calling OB (preferably in a cyclic interrupt OB).

Download to device

The actual values of retentive tags are only updated when you download PID_Compact completely.

Downloading technology objects to device (Page 6000)

Startup

At the startup of the CPU, PID_Compact starts in the operating mode that was last active. To retain PID_ Compact in "Inactive" mode, set sb_RunModeByStartup = FALSE.

Monitoring of the sampling time PID_Compact

Ideally, the sampling time is equivalent to the cycle time of the calling OB. The PID_Compact instruction measures the time interval between two calls. This is the current sampling time. On every switchover of operating mode and during the initial startup, the mean value is formed from the first 10 sampling times. If the current sampling time deviates too much from this mean value, Error = 0800 hex occurs and PID_Compact switches to "Inactive" mode.

PID_Compact, Version 1.1 or higher is set to "Inactive" mode during controller tuning under the following conditions:

- New mean value >= 1.1 x old mean value
- New mean value <= 0.9 x old mean value

In automatic mode, PID_Compact, Version 1.1 or higher, is set to "Inactive" mode under the following conditions:

- New mean value >= 1.5 x old mean value
- New mean value <= 0.5 x old mean value

During controller tuning and in automatic mode, PID_Compact 1.0 is set to "Inactive" operating mode under the following conditions:

- New mean value >= 1.1 x old mean value
- New mean value <= 0.9 x old mean value

- Current sampling time >= 1.5 x current mean value
- Current sampling time <= 0.5 x current mean value

Sampling time of the PID algorithm

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the cycle time. All other functions of PID_Compact are executed at every call.

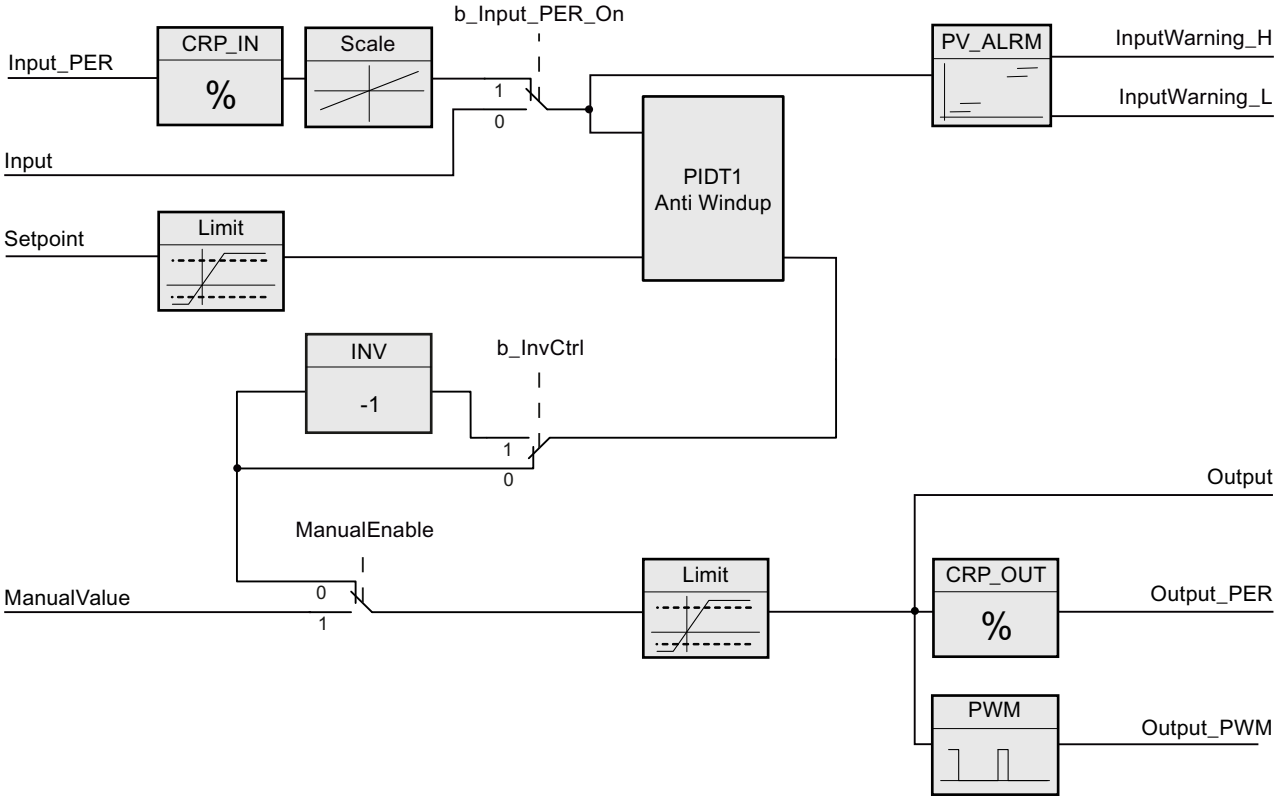
PID algorithm

PID_Compact is a PIDT1 controller with anti-windup and weighting of the proportional and derivative actions. The following equation is used to calculate the output value.

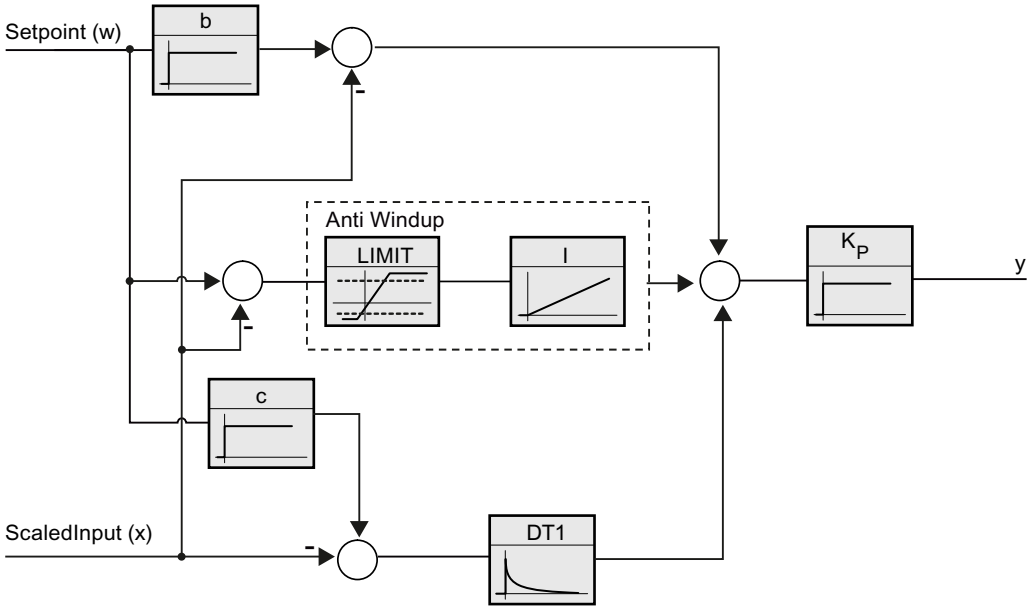
$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
y	Output value
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integral action time
a	Derivative delay coefficient (T1 = a × T_D)
	Derivative action time
c	Derivative action weighting

Block diagram of PID_Compact



Block diagram of PIDT1 with anti-windup



Reaction to error

If errors occur, they are output in parameter Error, and PID_Compact changes to "Inactive" mode. Reset the errors using the Reset parameter.

Control logic

An increase of the output value is generally intended to cause an increase in the process value. This is referred to as a normal control logic. For cooling and discharge control systems, it may be necessary to invert the control logic. PID_Compact does not work with negative proportional gain. If InvertControl = TRUE, an increasing control deviation causes a reduction in the output value. The control logic is also taken into account during pretuning and fine tuning.

See also

Control mode V1 (Page 6045)

Input parameters of PID_Compact V1 (S7-1200)

Table 11-70

Parameter	Data type	Default	Description
Setpoint	REAL	0.0	Setpoint of the PID controller in automatic mode
Input	REAL	0.0	A variable of the user program is used as source for the process value. If you are using parameter Input, then sPid_Cmpt.b_Input_PER_On = FALSE must be set.
Input_PER	WORD	W#16#0	Analog input as the source of the process value If you are using parameter Input_PER, then sPid_Cmpt.b_Input_PER_On = TRUE must be set.
ManualEnable	BOOL	FALSE	• A FALSE -> TRUE edge selects "Manual mode", while State = 4, sRet.i_Mode remains unchanged. • A TRUE -> FALSE edge selects the most recently active operating mode, State =sRet.i_Mode A change of sRet.i_Mode will not take effect during ManualEnable = TRUE. The change of sRet.i_Mode will only be considered upon a TRUE -> FALSE edge at ManualEnable . **PID_Compact V1.2 und PID_Compact V1.0** If at start of the CPU ManualEnable = TRUE, PID_Compact starts in manual mode. A rising edge (FALSE > TRUE) at ManualEnable is not necessary. **PID_Compact V1.1** At the start of the CPU, PID_Compact only switches to manual mode with a rising edge (FALSE->TRUE) at ManualEnable . Without rising edge, PID_Compact starts in the last operating mode in which ManualEnable was FALSE.
ManualValue	REAL	0.0	Manual value This value is used as the output value in manual mode.
Reset	BOOL	FALSE	The Reset parameter (Page 4248) restarts the controller.

Output parameters of PID_Compact V1 (S7-1200)

Table 11-71

Parameter	Data type	Default	Description
ScaledInput	REAL	0.0	Output of the scaled process value
Outputs "Output", "Output_PER", and "Output_PWM" can be used concurrently.			
Output	REAL	0.0	Output value in REAL format
Output_PER	WORD	W#16#0	Analog output value
Output_PWM	BOOL	FALSE	Pulse-width-modulated output value The output value is formed by minimum On and Off times.
SetpointLimit_H	BOOL	FALSE	If SetpointLimit_H = TRUE, the setpoint absolute high limit is reached. The setpoint in the CPU is limited to the configured setpoint absolute high limit. The configured process value absolute high limit is the default for the setpoint high limit. If you set sPid_Cmpt.r_Sp_Hlm to a value within the process value limits, this value is used as the setpoint high limit.
SetpointLimit_L	BOOL	FALSE	If SetpointLimit_L = TRUE, the setpoint absolute low limit has been reached. In the CPU, the setpoint is limited to the configured setpoint absolute low limit. The configured process value absolute low limit is the default setting for the setpoint low limit. If you set sPid_Cmpt.r_Sp_Llm to a value within the process value limits, this value is used as the setpoint low limit.
InputWarning_H	BOOL	FALSE	If InputWarning_H = TRUE, the process value has reached or exceeded the warning high limit.
InputWarning_L	BOOL	FALSE	If InputWarning_L = TRUE, the process value has reached or fallen below the warning low limit.
State	INT	0	The State parameter (Page 4244) shows the current operating mode of the PID controller. To change the operating mode, use variable sRet.i_Mode. • State = 0: Inactive • State = 1: pretuning • State = 2: fine tuning • State = 3: Automatic mode • State = 4: Manual mode
Error	DWORD	W#16#0	The Error parameter (Page 4247) indicates the error messages. Error = 0000: No error pending.

Static tags of PID_Compact V1 (S7-1200)

You must not change tags that are not listed. These are used for internal purposes only.

Table 11-72

Tag	Data type	Default	Description
sb_GetCycleTime	BOOL	TRUE	If sb_GetCycleTime = TRUE, the automatic determination of the cycle time is started. CycleTime.StartEstimation = FALSE once measurement is complete.
sb_EnCyclEstimation	BOOL	TRUE	If sb_EnCyclEstimation = TRUE, the PID_Compact sampling time is not monitored.
sb_EnCyclMonitoring	BOOL	TRUE	If sb_EnCyclMonitoring = FALSE, the PID_Compact sampling time is not monitored. If it is not possible to execute PID_Compact within the sampling time, an 0800 error is not output and PID_Compact does not change to "Inactive" mode.
sb_RunModeByStartup	BOOL	TRUE	Activate Mode after CPU restart If sb_RunModeByStartup = FALSE, the controller will remain inactive after a CPU startup. After a CPU startup and if sb_RunModeByStartup = TRUE, the controller will return to the most recently active operating mode.
si_Unit	INT	0	Unit of measurement of the process value and setpoint, e.g., ºC, or ºF.
si_Type	INT	0	Physical quantity of the process value and setpoint, e.g., temperature.
sd_Warning	DWORD	DW#16#0	Variable sd_warning (Page 4250) displays the warnings generated since the reset, or since the last change of the operating mode.
sBackUp.r_Gain	REAL	1.0	Saved proportional gain You can reload values from the sBackUp structure with sPid_Cmpt.b_LoadBackUp = TRUE.
sBackUp.r_Ti	REAL	20.0	Saved integral action time [s]
sBackUp.r_Td	REAL	0.0	Saved derivative action time [s]
sBackUp.r_A	REAL	0.0	Saved derivative delay coefficient
sBackUp.r_B	REAL	0.0	Saved proportional action weighting factor
sBackUp.r_C	REAL	0.0	Saved derivative action weighting factor
sBackUp.r_Cycle	REAL	1.0	Saved sampling time of PID algorithm
sPid_Calc.r_Cycle	REAL	0.1	Sampling time of the PID_Compact instruction r_Cycle is determined automatically and is usually equivalent to the cycle time of the calling OB.

Tag	Data type	Default	Description
sPid_Calc.b_RunIn	BOOL	FALSE	• b_RunIn = FALSE Pretuning is started when fine tuning is started from inactive or manual mode. If the requirements for pretuning are not met, PID_Compact reacts as if b_RunIn = TRUE. If fine tuning is started from automatic mode, the system uses the existing PID parameters to control to the setpoint. Only then will fine tuning start. If pretuning is not possible, PID_Compact switches to "Inactive" mode. • b_RunIn = TRUE The pretuning is skipped. PID_3Compact tries to reach the setpoint with minimum or maximum output value. This can produce increased overshoot. Fine tuning then starts automatically. b_RunIn is set to FALSE after fine tuning.
sPid_Calc.b_CalcParamSUT	BOOL	FALSE	The parameters for pretuning will be recalculated if b_CalcParamSUT = TRUE. This enables you to change the parameter calculation method without having to repeat controller tuning. b_CalcParamSUT is set to FALSE after the calculation.
sPid_Calc.b_CalcParamTIR	BOOL	FALSE	The parameters for fine tuning will be recalculated if b_CalcParamTIR = TRUE. This enables you to change the parameter calculation method without having to repeat controller tuning. b_CalcParamTIR will be set to FALSE after calculation.
sPid_Calc.i_CtrlTypeSUT	INT	0	Methods used to calculate parameters during pretuning: • i_CtrlTypeSUT = 0: PID according to Chien, Hrones and Reswick • i_CtrlTypeSUT = 1: PI according to Chien, Hrones and Reswick
sPid_Calc.i_CtrlTypeTIR	INT	0	Methods used to calculate parameters during fine tuning: • i_CtrlTypeTIR = 0: PID automatic • i_CtrlTypeTIR = 1: PID fast • i_CtrlTypeTIR = 2: PID slow • i_CtrlTypeTIR = 3: Ziegler-Nichols PID • i_CtrlTypeTIR = 4: Ziegler-Nichols PI • i_CtrlTypeTIR = 5: Ziegler-Nichols P
sPid_Calc.r_Progress	REAL	0.0	Progress of tuning as a percentage (0.0 - 100.0)

Tag	Data type	Default	Description
sPid_Cmpt.r_Sp_Hlm	REAL	+3.402822e +38	High limit of setpoint If you configure sPid_Cmpt.r_Sp_Hlm outside the process value limits, the configured process value absolute high limit is used as the setpoint high limit. If you set sPid_Cmpt.r_Sp_Hlm within the process value limits, this value is used as the setpoint high limit.
sPid_Cmpt.r_Sp_Llm	REAL	-3.402822e +38	Low limit of the setpoint If you set sPid_Cmpt.r_Sp_Llm outside the process value limits, the configured process value absolute low limit is used as the setpoint low limit. If you set sPid_Cmpt.r_Sp_Llm within the process value limits, this value is used as the setpoint low limit.
sPid_Cmpt.r_Pv_Norm_IN_1	REAL	0.0	Scaling Input_PER low Input_PER is converted to percent based on the two value pairs r_Pv_Norm_OUT_1, r_Pv_Norm_IN_1 and r_Pv_Norm_OUT_2, r_Pv_Norm_IN_2 from the sPid_Cmpt structure.
sPid_Cmpt.r_Pv_Norm_IN_2	REAL	27648.0	Scaling Input_PER high Input_PER is converted to percent based on the two value pairs r_Pv_Norm_OUT_1, r_Pv_Norm_IN_1 and r_Pv_Norm_OUT_2, r_Pv_Norm_IN_2 from the sPid_Cmpt structure.
sPid_Cmpt.r_Pv_Norm_OUT_1	REAL	0.0	Scaled low process value Input_PER is converted to percent based on the two value pairs r_Pv_Norm_OUT_1, r_Pv_Norm_IN_1 and r_Pv_Norm_OUT_2, r_Pv_Norm_IN_2 from the sPid_Cmpt structure.
sPid_Cmpt.r_Pv_Norm_OUT_2	REAL	100.0	Scaled high process value Input_PER is converted to percent based on the two value pairs r_Pv_Norm_OUT_1, r_Pv_Norm_IN_1 and r_Pv_Norm_OUT_2, r_Pv_Norm_IN_2 from the sPid_Cmpt structure.
sPid_Cmpt.r_Lmn_Hlm	REAL	100.0	Output value high limit for output parameter "Output"
sPid_Cmpt.r_Lmn_Llm	REAL	0.0	Low output value limit for output parameter "Output"
sPid_Cmpt.b_Input_PER_On	BOOL	TRUE	If b_Input_PER_On = TRUE, then parameter Input_PER is used. If b_Input_PER_On = FALSE, then parameter Input is used.
sPid_Cmpt.b_LoadBackUp	BOOL	FALSE	Activate the back-up parameter set. If an optimization has failed, you can reactivate the previous PID parameters by setting this bit.
sPid_Cmpt.b_InvCtrl	BOOL	FALSE	Invert control logic With b_InvCtrl = TRUE, an increasing control deviation causes a reduction in the output value.
sPid_Cmpt.r_Lmn_Pwm_PPTm	REAL	0.0	The minimum ON time of the pulse width modulation in seconds is rounded to r_Lmn_Pwm_PPTm = r_Cycle or r_Lmn_Pwm_PPTm = n*r_Cycle

Tag	Data type	Default	Description
sPid_Cmpt.r_Lmn_Pwm_PBTm	REAL	0.0	The minimum OFF time of the pulse width modulation in seconds is rounded to r_Lmn_Pwm_PBTm = r_Cycle or r_Lmn_Pwm_PBTm = n*r_Cycle
sPid_Cmpt.r_Pv_Hlm	REAL	120.0	High limit of the process value At the I/O input, the process value can be a maximum of 18% higher than the standard range (overrange). An error is no longer reported for a violation of the "Process value high limit". Only a wire-break and a short-circuit are recognized and the PID_Compact switches to "Inactive" mode. r_Pv_Hlm > r_Pv_Llm
sPid_Cmpt.r_Pv_Llm	REAL	0.0	Low limit of the process value r_Pv_Llm < r_Pv_Hlm
sPid_Cmpt.r_Pv_HWrn	REAL	+3.402822e +38	Warning high limit of the process value If you set r_Pv_HWrn outside the process value limits, the configured process value absolute high limit is used as the warning high limit. If you configure r_Pv_HWrn within the process value limits, this value is used as the warning high limit. r_Pv_HWrn > r_Pv_LWrn r_Pv_HWrn ≤ r_Pv_Hlm
sPid_Cmpt.r_Pv_LWrn	REAL	-3.402822e +38	Warning low limit of the process value If you set r_Pv_LWrn outside the process value limits, the configured process value absolute low limit is used as the warning low limit. If you configure r_Pv_LWrn within the process value limits, this value is used as the warning low limit. r_Pv_LWrn < r_Pv_HWrn r_Pv_LWrn ≥ r_Pv_LWrn
sPidCalc.i_Ctrl_IOutv	REAL	0.0	Current integral action
sParamCalc.i_Event_SUT	INT	0	Variable i_Event_SUT (Page 4251) indicates the current phase of "pretuning":
sParamCalc.i_Event_TIR	INT	0	Variable i_Event_TIR (Page 4251) indicates the current phase of "fine tuning":
sRet.i_Mode	INT	0	The operating mode is changed edge-triggered. The following operating mode is enabled on a change to • i_Mode = 0: "Inactive" mode (controller stop) • i_Mode = 1: "Pretuning" mode • i_Mode = 2: "Fine tuning" mode • i_Mode = 3: "Automatic" mode • i_Mode = 4: "Manual" mode i_Mode is retentive.
sRet.r_Ctrl_Gain	REAL	1.0	Active proportional gain Gain is retentive.

Tag	Data type	Default	Description
sRet.r_Ctrl_Ti	REAL	20.0	• r_Ctrl_Ti > 0.0: Active integral action time • r_Ctrl_Ti = 0.0: Integral action is deactivated r_Ctrl_Ti is retentive.
sRet.r_Ctrl_Td	REAL	0.0	• r_Ctrl_Td > 0.0: Active derivative action time • r_Ctrl_Td = 0.0: Derivative action is deactivated r_Ctrl_Td is retentive.
sRet.r_Ctrl_A	REAL	0.0	Active derivative delay coefficient r_Ctrl_A is retentive.
sRet.r_Ctrl_B	REAL	0.0	Active proportional action weighting r_Ctrl_B is retentive.
sRet.r_Ctrl_C	REAL	0.0	Active derivative action weighting r_Ctrl_C is retentive.
sRet.r_Ctrl_Cycle	REAL	1.0	Active sampling time of the PID algorithm r_Ctrl_Cycle is calculated during tuning and rounded to an integer multiple of r_Cycle. r_Ctrl_Cycle is used as time period of the pulse width modulation. r_Ctrl_Cycle is retentive.

Note

Change the tags listed in this table in "Inactive" mode to prevent malfunction of the PID controller. "Inactive" mode is forced by setting the "sRet.i_Mode" tag to "0".

See also

Downloading technology objects to device (Page 6000)

Parameters State and sRet.i_Mode V1 (S7-1200)

Correlation of the parameters

The State parameter indicates the current operating mode of the PID controller. You cannot modify the State parameter.

You need to modify the sRet.i_Mode tag to change the operating mode. This also applies when the value for the new operating mode is already in sRet.i_Mode. First set sRet.i_Mode = 0 and then sRet.i_Mode = 3. Provided the current operating mode of the controller supports this change, State is set to the value of sRet.i_Mode.

When PID_Compact automatically switches the operating mode, the following applies: State ! = sRet.i_Mode.

Examples:

- Successful pretuning
  State = 3 and sRet.i_Mode = 1
- Error
  State = 0 and sRet.i_Mode remains at the same value, e.g sRet.i_Mode = 3
- ManualEnalbe = TRUE
  State = 4 and sRet.i_Mode remain at the previous value, for example, sRet.i_Mode = 3

Note

You wish to repeat successful fine tuning without exiting automatic mode with i_Mode = 0.

Setting sRet.i_Mode to an invalid value such as 9999 for one cycle has no effect on State. Set Mode = 2 in the next cycle. You can generate a change to sRet.i_Mode without first switching to "inactive" mode.

Meaning of values

State / sRet.i_Mode	Description of the operating mode
0	Inactive The controller is switched off. The controller was in "inactive" mode before pretuning was performed. The PID controller will change to "inactive" mode when running if an error occurs or if the "Deactivate controller" icon is clicked in the commissioning window.
1	Pretuning The pretuning determines the process response to a jump of the output value and searches for the point of inflection. The optimized PID parameters are calculated as a function of the maximum rate of rise and dead time of the controlled system. Pretuning requirements: • The controller is in inactive mode or manual mode • ManualEnable = FALSE • The process value must not be too close to the setpoint. |Setpoint - Input| > 0.3 * |sPid_Cmpt.r_Pv_Hlm - sPid_Cmpt.r_Pv_Llm| and |Setpoint - Input| > 0.5 * |Setpoint| • The setpoint may not be changed during pretuning. The higher the stability of the process value, the easier it is to calculate the PID parameters and increase precision of the result. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise. PID parameters are backed up before they are recalculated and can be reactivated with sPid_Cmpt.b_LoadBackUp. There is a change to automatic mode following successful pretuning and to "inactive" mode following unsuccessful pretuning. The phase of pretuning is indicated with Tag i_Event_SUT V1 (Page 4251).

State / sRet.i_Mode	Description of the operating mode
2	Fine tuning Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are optimized based on the amplitude and frequency of this oscillation. The differences between the process response during pretuning and fine tuning are analyzed. All PID parameters are recalculated on the basis of the findings. PID parameters from fine tuning usually have better master control and disturbance behavior than PID parameters from pretuning. PID_Compact automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. PID parameters are backed up before they are recalculated and can be reactivated with sPid_Cmpt.b_LoadBackUp. Requirements for fine tuning: • No disturbances are expected. • The setpoint and the process value lie within the configured limits. • The setpoint may not be changed during fine tuning. • ManualEnable = FALSE • Automatic (State = 3), inactive (State = 0) or manual (State = 4) mode Fine tuning proceeds as follows when started in: • Automatic mode (State = 3) Start fine tuning in automatic mode if you wish to improve the existing PID parameters using controller tuning. PID_Comact will regulate using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start. • Inactive (State = 0) or manual (State = 4) mode If the requirements for pretuning are met, pretuning is started. The PID parameters established will be used for adjustment until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start. If pretuning is not possible, PID_Compact will change to "Inactive" mode. An attempt is made to reach the setpoint with a minimum or maximum output value if the process value for pretuning is already too near the setpoint or sPid_Calc.b_RunIn = TRUE. This can produce increased overshoot. The controller will change to "automatic mode" after successfully completed "fine tuning" and to "inactive" mode if "fine tuning" has not been successfully completed. The "Fine tuning" phase is indicated with Tag i_Event_TIR V1 (Page 4251).

State / sRet.i_Mode	Description of the operating mode
3	Automatic mode In automatic mode, PID_Compact corrects the controlled system in accordance with the parameters specified. The controller changes to automatic mode if one the following conditions is fulfilled: • Pretuning successfully completed • Fine tuning successfully completed • Change of variable sRet.i_Mode to the value 3. After CPU startup or change from Stop to RUN mode, PID_Compact will start in the most recently active operating mode. To retain PID_Compact in "Inactive" mode, set sb_RunModeByStartup = FALSE.
4	Manual mode In manual mode, you specify a manual output value in the ManualValue parameter. This operating mode is enabled if sRet.i_Mode = 4, or at the rising edge on ManualEnable. If ManualEnable changes to TRUE, only State will change. sRet.i_Mode will retain its current value. PID_Compact will return to the previous operating mode upon a falling edge at ManualEnable. The change to automatic mode is bumpless.

See also

Output parameters of PID_Compact V1 (Page 4239)

Pretuning V1 (Page 6055)

Fine tuning V1 (Page 6057)

"Manual" mode V1 (Page 6058)

Tag i_Event_SUT V1 (Page 4251)

Tag i_Event_TIR V1 (Page 4251)

Parameter Error V1 (S7-1200)

If several errors are pending simultaneously, the values of the error codes are displayed with binary addition. The display of error code 0003, for example, indicates that the errors 0001 and 0002 are pending simultaneously.

Error (DW#16#...)	Description
0000	There is no error.
0001	The "Input" parameter is outside the process value limits. • Input > sPid_Cmpt.r_Pv_Hlm or • Input < sPid_Cmpt.r_Pv_Llm You cannot move the actuator again until you eliminate the error.
0002	Invalid value at "Input_PER" parameter. Check whether an error is pending at the analog input.
0004	Error during fine tuning. Oscillation of the process value could not be maintained.
0008	Error at start of pretuning. The process value is too close to the setpoint. Start fine tuning.
0010	The setpoint was changed during tuning.
0020	Pretuning is not permitted in automatic mode or during fine tuning.

Error (DW#16#...)	Description
0080	Error during pretuning. The output value limits are not configured correctly or the actual value does not react as expected. Check whether the limits of the output value are configured correctly and match the control logic. Also make sure that the actual value does not oscillate strongly before starting pretuning.
0100	Error during tuning resulted in invalid parameters.
0200	Invalid value at "Input" parameter: Value has an invalid number format.
0400	Calculation of output value failed. Check the PID parameters.
0800	Sampling time error: PID_Compact is not called within the sampling time of the cyclic interrupt OB. If this error occurred during simulation with PLCSIM, see the notes under Simulating PID_Compact V1 with PLCSIM (Page 6060).
1000	Invalid value at "Setpoint" parameter: Value has an invalid number format.

See also

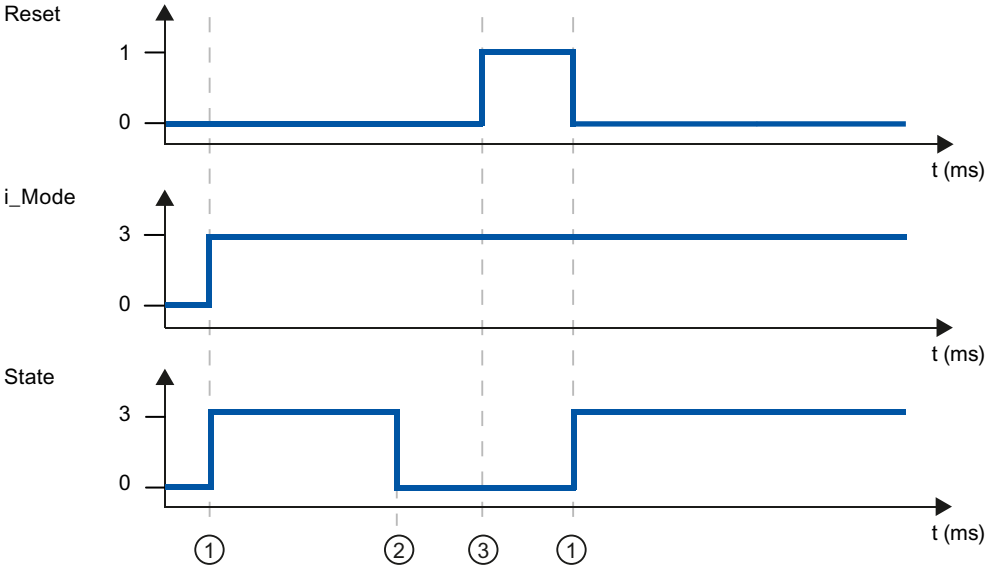
Output parameters of PID_Compact V1 (Page 4239)

Parameter Reset V1 (S7-1200)

The response to Reset = TRUE depends on the version of the PID_Compact instruction.

Reset response PID_Compact V.1.1 or higher

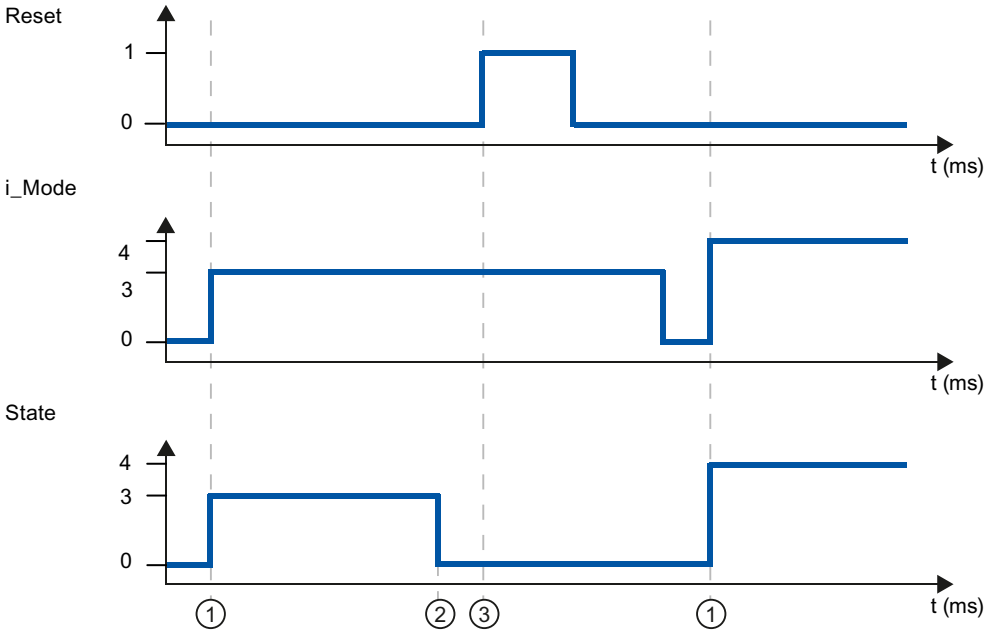
A rising edge at Reset triggers a change to "Inactive" mode; errors and warnings are reset and the integral action is deleted. A falling edge at Reset triggers a change to the most recently active operating mode. If automatic mode was active before, the integral action is pre-assigned in such a way that the switchover is bumpless.



① Activation
② Error
③ Reset

Reset response PID_Compact V.1.0

A rising edge at Reset triggers a change to "Inactive" mode; errors and warnings are reset and the integral action is deleted. The controller is not reactivated until the next edge at i_Mode.



① Activation
② Error
③ Reset

Tag sd_warning V1 (S7-1200)

If several warnings are pending, the values of variable sd_warning are displayed by means of binary addition. The display of warning 0003, for example, indicates that the warnings 0001 and 0002 are also pending.

sd_warning (DW#16#....)	Description
0000	No warning pending.
0001	The point of inflection was not found during pretuning.
0002	Oscillation increased during fine tuning.
0004	The setpoint was outside the set limits.
0008	Not all the necessary controlled system properties were defined for the selected method of calculation. The PID parameters were instead calculated using the "i_CtrlTypeTIR = 3" method.
0010	The operating mode could not be changed because ManualEnable = TRUE.
0020	The cycle time of the calling OB limits the sampling time of the PID algorithm. Improve results by using shorter OB cycle times.
0040	The process value exceeded one of its warning limits.

The following warnings are deleted as soon as the cause is dealt with:

- 0004
- 0020
- 0040

All other warnings are cleared with a rising edge at Reset.

Tag i_Event_SUT V1 (S7-1200)

i_Event_SUT	Name	Description
0	SUT_INIT	Initialize pretuning
100	SUT_STDABW	Calculate the standard deviation
200	SUT_GET_POI	Find the point of inflection
9900	SUT_IO	Pretuning successful
1	SUT_NIO	Pretuning not successful

See also

Static tags of PID_Compact V1 (Page 4240)

Parameters State and sRet.i_Mode V1 (Page 4244)

Tag i_Event_TIR V1 (S7-1200)

i_Event_TIR	Name	Description
-100	TIR_FIRST_SUT	Fine tuning is not possible. Pretuning will be executed first.
0	TIR_INIT	Initialize fine tuning
200	TIR_STDABW	Calculate the standard deviation
300	TIR_RUN_IN	Attempt to reach the setpoint
400	TIR_CTRLN	Attempt to reach the setpoint with the existing PID parameters (if pretuning has been successful)
500	TIR_OSZIL	Determine oscillation and calculate parameters
9900	TIR_IO	Fine tuning successful
1	TIR_NIO	Fine tuning not successful

See also

Static tags of PID_Compact V1 (Page 4240)

Parameters State and sRet.i_Mode V1 (Page 4244)

PID_3Step (S7-1200, S7-1500)

New features of PID_3Step (S7-1200, S7-1500)

PID_3Step V2.3

- As of PID_3Step Version 2.3 the monitoring and limiting of the travel time can be deactivated with Config.VirtualActuatorLimit = 0.0.

PID_3Step V2.2

- **Use with S7-1200**
  As of PID_3Step V2.2, the instruction with V2 functionality can also be used on S7-1200 with firmware version 4.0 or higher.

PID_3Step V2.0

- **Reaction to error**
  The reaction to ActivateRecoverMode = TRUE has been completely overhauled. PID_3Step reacts in a more fault tolerant manner in the default setting.

NOTICE
Your system may be damaged. If you use the default setting, PID_3Step remains in automatic mode even if the process value limits are exceeded. This may damage your system. It is essential to configure how your controlled system reacts in the event of an error to protect your system from damage.

  You use the ErrorAck input parameter to acknowledge the errors and warnings without restarting the controller or clearing the integral action.
  Switching operating modes does not acknowledge errors that are no longer pending.
- **Switching the operating mode**
  You specify the operating mode at the Mode in/out parameter and use a positive edge at ModeActivate to start the operating mode. The Retain.Mode tag has been omitted.
  The transition time measurement can no longer be started with GetTransitTime.Start, but only with Mode = 6 and a positive edge at ModeActivate.
- **Multi-instance capability**
  You can call up PID_3Step as multi-instance DB. No technology object is created in this case and no parameter assignment interface or commissioning interface is available. You must assign parameters for PID_3Step directly in the multi-instance DB and commission it via a watch table.
- **Startup characteristics**
  The operating mode specified at the Mode parameter is also started on a negative edge at Reset and during a CPU cold restart, if RunModeByStartup = TRUE.

- **ENO characteristics**
  ENO is set depending on the operating mode.
  If State = 0, then ENO = FALSE.
  If State ≠ 0, then ENO = TRUE.
- **Manual mode**
  The Manual_UP and Manual_DN input parameters no longer function as edge-triggered parameters. Edge-triggered manual mode continues to be possible using the ManualUpInternal and ManualDnInternal tags.
  In "Manual mode without endstop signals" (Mode = 10), the endstop signals Actuator_H and Actuator_L are ignored even though they are activated.
- **Default value of PID parameters**
  The following default settings have been changed:
  - Proportional action weighting (PWeighting) from 0.0 to 1.0
  - Derivative action weighting (DWeighting) from 0.0 to 1.0
  - Coefficient for derivative delay (TdFiltRatio) from 0.0 to 0.2
- **Limiting of motor transition time**
  You configure the maximum percentage of the motor transition time that the actuator will travel in one direction in the Config.VirtualActuatorLimit tag.
- **Setpoint value specification during tuning**
  You configure the permitted fluctuation of the setpoint during tuning at the CancelTuningLevel tag.
- **Switching a disturbance variable on**
  You can switch a disturbance variable on at the Disturbance parameter.
- **Troubleshooting**
  If the endstop signals are not activated (ActuatorEndStopOn = FALSE), ScaledFeedback is determined without Actuator_H or Actuator_L.

PID_3Step V1.1

- **Manual mode on CPU startup**
  If ManualEnable = TRUE when the CPU starts, PID_3Step starts in manual mode. A positive edge at ManualEnable is not necessary.
- **Reaction to error**
  The ActivateRecoverMode tag is no longer effective in manual mode.
- **Troubleshooting**
  The Progress tag is reset following successful tuning or transition time measurement.

Compatibility with CPU and FW (S7-1200, S7-1500)

The following table shows which version of PID_3Step can be used on which CPU.

CPU	FW	PID_3Step
S7-1200	V4.2 or higher	V2.3 V2.2 V1.1
	V4.0 to V4.1	V2.2 V1.1
	V3.x	V1.1 V1.0
	V2.x	V1.1 V1.0
	V1.x	-
S7-1500	V2.0 or higher	V2.3 V2.2 V2.1 V2.0
	V1.5 to V1.8	V2.2 V2.1 V2.0
	V1.1	V2.1 V2.0
	V1.0	V2.0

CPU processing time and memory requirement PID_3Step V2.x (S7-1200, S7-1500)

CPU processing time

Typical CPU processing times of the PID_3Step technology object as of Version V2.0, depending on CPU type.

CPU	Typ. CPU processing time PID_3Step V2.x
CPU 1211C ≥ V4.0	410 µs
CPU 1215C ≥ V4.0	410 µs
CPU 1217C ≥ V4.0	410 µs
CPU 1505S ≥ V1.0	50 µs
CPU 1510SP-1 PN ≥ V1.6	120 µs
CPU 1511-1 PN ≥ V1.5	120 µs
CPU 1512SP-1 PN ≥ V1.6	120 µs
CPU 1516-3 PN/DP ≥ V1.5	65 µs
CPU 1518-4 PN/DP ≥ V1.5	5 µs

Memory requirement

Memory requirement of an instance DB of the PID_3Step technology object as of Version V2.0.

	Memory requirement of the instance DB of PID_3Step V2.x
Load memory requirement	Approx. 15000 bytes
Total work memory requirement	1040 bytes
Retentive work memory requirement	60 bytes

PID_3Step V2 (S7-1200, S7-1500)

Description of PID_3Step V2 (S7-1200, S7-1500)

Description

You use the PID_3Step instruction to configure a PID controller with self tuning for valves or actuators with integrating behavior.

The following operating modes are possible:

- Inactive
- Pretuning
- Fine tuning
- Automatic mode
- Manual mode
- Approach substitute output value
- Transition time measurement
- Error monitoring
- Approach substitute output value with error monitoring
- Manual mode without endstop signals

For a more detailed description of the operating modes, see the State parameter.

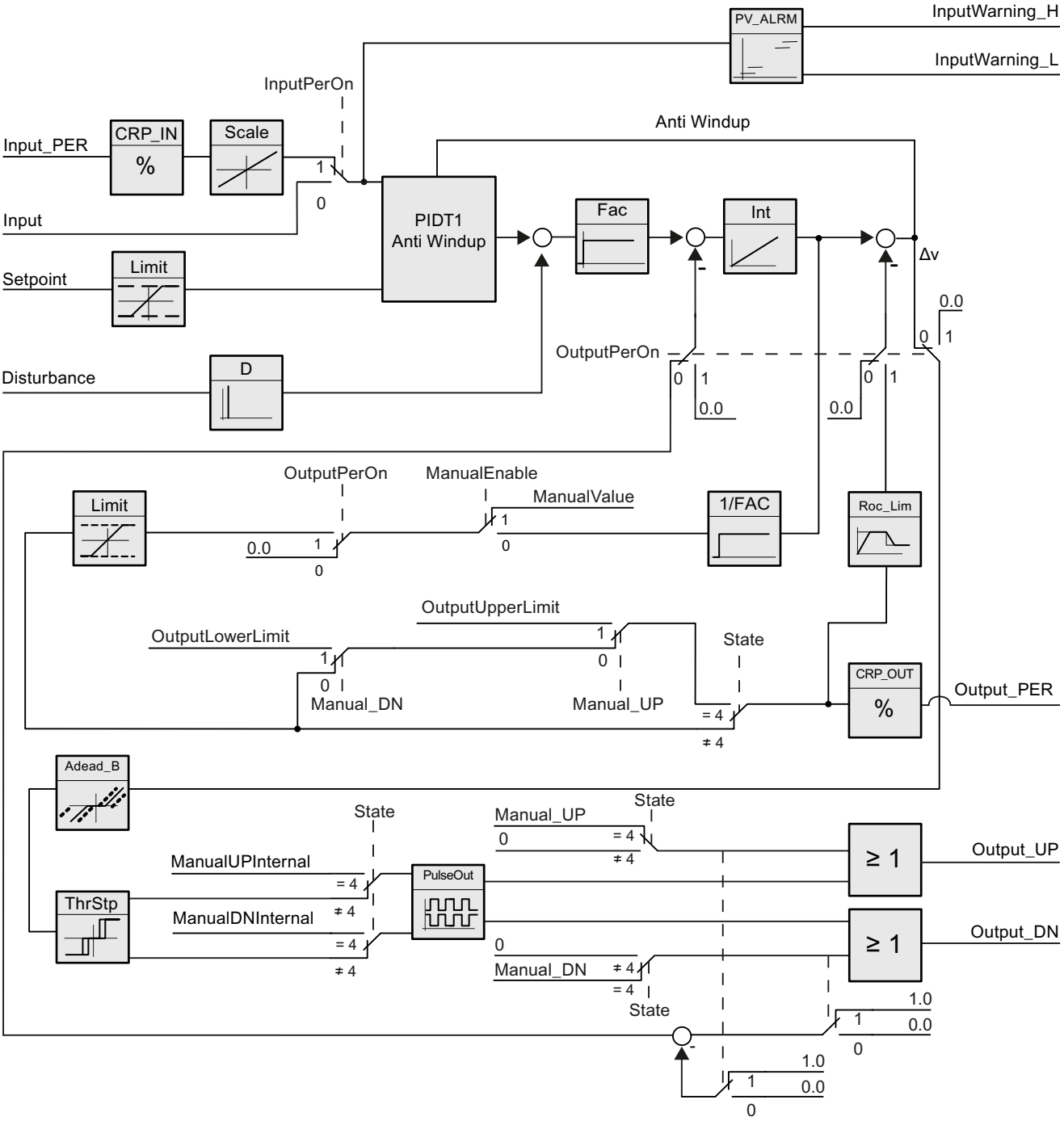
PID algorithm

PID_3Step is a PIDT1 controller with anti-windup and weighting of the proportional and derivative actions. The PID algorithm operates according to the following equation:

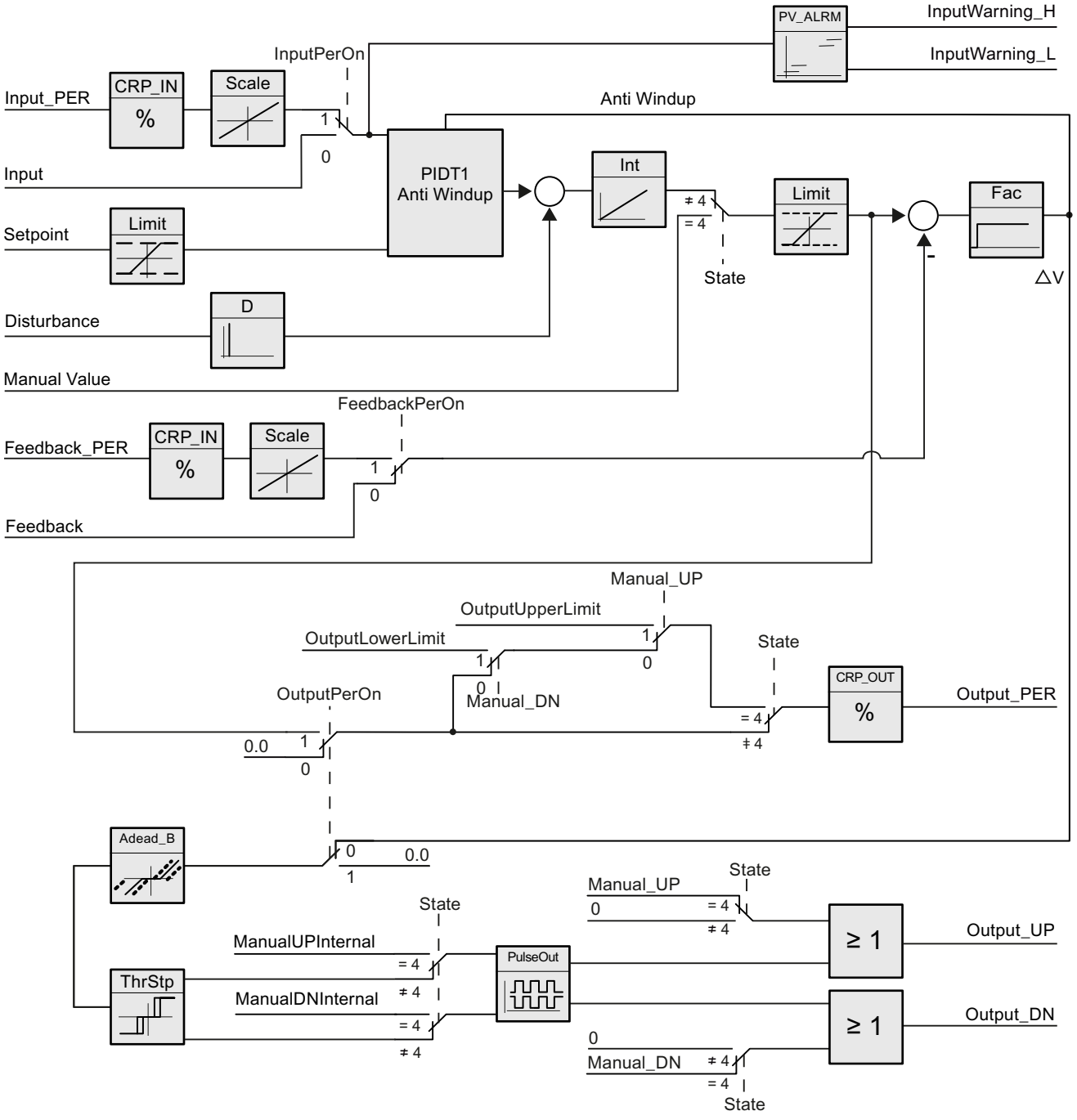
$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
Δy	Output value of the PID algorithm
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integral action time
T_D	Derivative action time
a	Derivative delay coefficient (derivative delay T1 = a × T_D)
c	Derivative action weighting

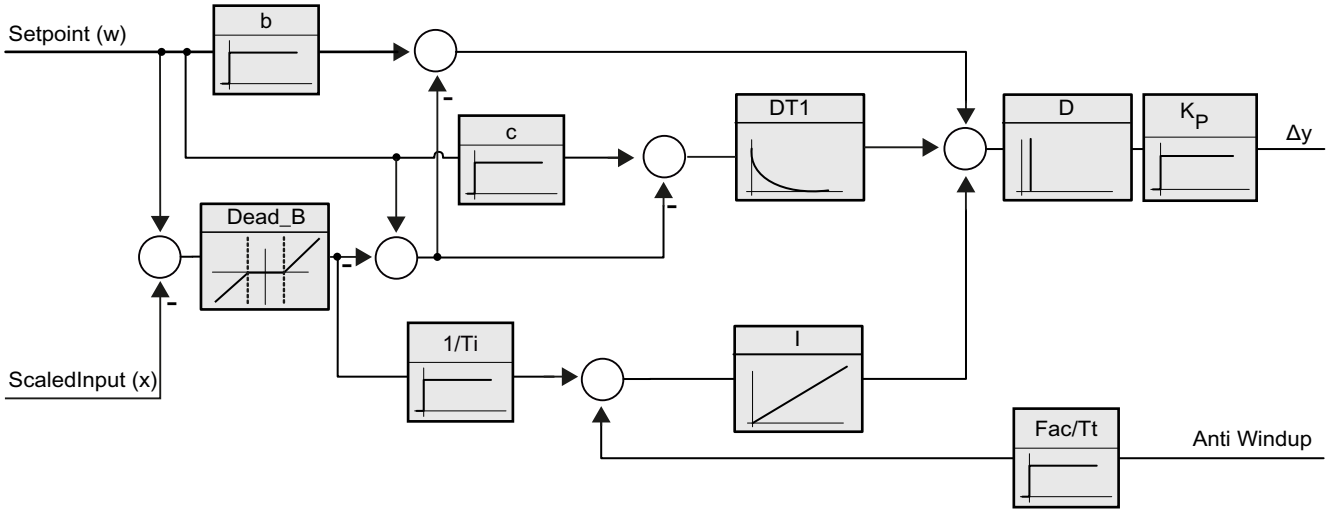
Block diagram without position feedback



Block diagram with position feedback



Block diagram of PIDT1 with anti-windup



Call

PID_3Step is called in the constant time scale of a cycle interrupt OB.

If you call PID_3Step as a multi-instance DB, no technology object is created. No parameter assignment interface or commissioning interface is available. You must assign parameters for PID_3Step directly in the multi-instance DB and commission it via a watch table.

Download to device

The actual values of retentive tags are only updated when you download PID_3Step completely.

Downloading technology objects to device (Page 6000)

Startup

When the CPU starts up, PID_3Step starts in the operating mode that is saved in the Mode in/out parameter. To leave PID_3Step in "Inactive" mode, set RunModeByStartup = FALSE.

Reaction to error

In automatic mode and during commissioning, the reaction to error depends on the ErrorBehaviour and ActivateRecoverMode tags. In manual mode, the reaction is independent of ErrorBehaviour and ActivateRecoverMode. If ActivateRecoverMode = TRUE, the reaction additionally depends on the error that occurred.

ErrorBehaviour	ActivateRecoverMode	Configuration editor > actuator setting > Set Output to	Reaction
FALSE	FALSE	Current output value	Switch to "Inactive" mode (State = 0) The actuator remains in the current position.
FALSE	TRUE	Current output value while error is pending	Switch to "Error monitoring" mode (State = 7) The actuator remains in the current position while the error is pending.
TRUE	FALSE	Substitute output value	Switch to "Approach substitute output value" mode (State = 5) The actuator moves to the configured substitute output value. Switch to "Inactive" mode (State = 0) The actuator remains in the current position.
TRUE	TRUE	Substitute output value while error is pending	Switch to "Approach substitute output value with error monitoring" mode (State = 8) The actuator moves to the configured substitute output value. Switch to "Error monitoring" mode (State = 7)

In manual mode, PID_3Step uses ManualValue as output value, unless the following errors occur:

- 2000h: Invalid value at Feedback_PER parameter.
- 4000h: Invalid value at Feedback parameter.
- 8000h: Error during digital position feedback.

You can only change the position of the actuator with Manual_UP and Manual_DN, not with ManualValue.

The Error parameter indicates whether an error has occurred in this cycle. The ErrorBits parameter shows which errors have occurred. ErrorBits is reset by a rising edge at Reset or ErrorAck.

See also

Parameters State and Mode V2 (Page 4276)

Parameter ErrorBits V2 (Page 4281)

Configuring PID_3Step V2 (Page 6061)

Mode of operation of PID_3Step V2 (S7-1200, S7-1500)

Monitoring process value limits

You specify the high limit and low limit of the process value in the Config.InputUpperLimit and Config.InputLowerLimit tags. If the process value is outside these limits, an error occurs (ErrorBits = 0001h).

You specify a high and low warning limit of the process value in the Config.InputUpperWarning and Config.InputLowerWarning tags. If the process value is outside these warning limits, a warning occurs (Warning = 0040h), and the InputWarning_H or InputWarning_L output parameter changes to TRUE.

Limiting the setpoint

You specify a high limit and low limit of the setpoint in the Config.SetpointUpperLimit and Config.SetpointLowerLimit tags. PID_3Step automatically limits the setpoint to the process value limits. You can limit the setpoint to a smaller range. PID_3Step checks whether this range falls within the process value limits. If the setpoint is outside these limits, the high or low limit is used as the setpoint, and output parameter SetpointLimit_H or SetpointLimit_L is set to TRUE.

The setpoint is limited in all operating modes.

Limiting the output value

You specify a high limit and low limit of the output value in the Config.OutputUpperLimit and Config.OutputLowerLimit tags. The output value limits must be within "Low endstop" and "High endstop".

- High endstop: Config.FeedbackScaling.UpperPointOut
- Low endstop: Config.FeedbackScaling.LowerPointOut

Rule:

UpperPointOut ≥ OutputUpperLimit > OutputLowerLimit ≥ LowerPointOut

The valid values for "High endstop" and "Low endstop" depend upon:

- FeedbackOn
- FeedbackPerOn
- OutputPerOn

OutputPerOn	FeedbackOn	FeedbackPerOn	LowerPointOut	UpperPointOut
FALSE	FALSE	FALSE	Cannot be set (0.0%)	Cannot be set (100.0%)
FALSE	TRUE	FALSE	-100.0% or 0.0%	0.0% or +100.0%
FALSE	TRUE	TRUE	-100.0% or 0.0%	0.0% or +100.0%
TRUE	FALSE	FALSE	Cannot be set (0.0%)	Cannot be set (100.0%)
TRUE	TRUE	FALSE	-100.0% or 0.0%	0.0% or +100.0%
TRUE	TRUE	TRUE	-100.0% or 0.0%	0.0% or +100.0%

If OutputPerOn = FALSE and FeedbackOn = FALSE, you cannot limit the output value. Output_UP and Output_DN are then reset upon Actuator_H = TRUE or Actuator_L = TRUE. If endstop signals are also not present, Output_UP and Output_DN are reset after a travel time of Config.VirtualActuatorLimit × Retain.TransitTime/100. As of PID_3Step Version 2.3 the monitoring and limiting of the travel time can be deactivated with Config.VirtualActuatorLimit = 0.0.

The output value is 27648 at 100% and -27648 at -100%. PID_3Step must be able to completely close the valve.

Note

Use with two or more actuators

PID_3 Step is not suitable for use with two or more actuators (for example, in heating/cooling applications), because different actuators need different PID parameters to achieve a good control response.

Substitute output value

If an error has occurred, PID_3Step can output a substitute output value and move the actuator to a safe position that is specified in the SavePosition tag. The substitute output value must be within the output value limits.

Monitoring signal validity

The values of the following parameters are monitored for validity when used:

- Setpoint
- Input
- Input_PER
- Input_PER
- Feedback
- Feedback_PER
- Disturbance
- ManualValue
- SavePosition
- Output_PER

Monitoring the PID_3Step sampling time

Ideally, the sampling time is equivalent to the cycle time of the calling OB. The PID_3Step instruction measures the time interval between two calls. This is the current sampling time. On every switchover of operating mode and during the initial startup, the mean value is formed from the first 10 sampling times. Too great a difference between the current sampling time and this mean value triggers an error (ErrorBits = 0800h).

The error occurs during tuning if:

- New mean value >= 1.1 x old mean value
- New mean value <= 0.9 x old mean value

The error occurs in automatic mode if:

- New mean value >= 1.5 x old mean value
- New mean value <= 0.5 x old mean value

If you deactivate the sampling time monitoring (CycleTime.EnMonitoring = FALSE), you can also call PID_3Step in OB1. You must then accept a lower control quality due to the deviating sampling time.

Sampling time of the PID algorithm

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the cycle time. All other functions of PID_3Step are executed at every call.

Measuring the motor transition time

The motor transition time is the time in seconds the motor requires to move the actuator from the closed to the opened state. The actuator is moved in one direction for a maximum time of Config.VirtualActuatorLimit × Retain.TransitTime/100. PID_3Step requires the motor transition time to be as accurate as possible for good controller results. The data in the actuator documentation contains average values for this type of actuator. The value for the specific actuator used may differ. You can measure the motor transition time during commissioning. The output value limits are not taken into consideration during the motor transition time measurement. The actuator can travel to the high or the low endstop.

The motor transition time is taken into consideration in the calculation of the analog output value as well as in the calculation of the digital output values. It is mainly required for correct operation during auto-tuning and the anti-windup behavior. You should therefore configure the motor transition time with the value that the motor requires to move the actuator from the closed to the opened state.

If no relevant motor transition time is in effect in your process (e.g. with solenoid valves), so that the output value has a direct and full effect on the process, use PID_Compact instead.

Control logic

An increase of the output value is generally intended to cause an increase in the process value. This is referred to as a normal control logic. For cooling and discharge control systems, it may be necessary to invert the control logic. PID_3Step does not work with negative proportional gain. If InvertControl = TRUE, an increasing control deviation causes a reduction in the output value. The control logic is also taken into account during pretuning and fine tuning.

See also

Configuring PID_3Step V1 (Page 6079)

Changing the PID_3Step V2 interface (S7-1200, S7-1500)

The following table shows what has changed in the PID_3Step instruction interface.

PID_3Step V1	PID_3Step V2	Change
Input_PER	Input_PER	Data type from Word to Int
Feedback_PER	Feedback_PER	Data type from Word to Int
	Disturbance	New
Manual_UP	Manual_UP	Function
Manual_DN	Manual_DN	Function
	ErrorAck	New
	ModeActivate	New
Output_PER	Output_PER	Data type from Word to Int
	ManualUPInternal	New
	ManualDNInternal	New
	CancelTuningLevel	New
	VirtualActuatorLImit	New
Config.Loadbackup	Loadbackup	Renamed
Config.TransitTime	Retain.TransitTime	Renamed and retentivity added
GetTransitTime.Start		Replaced by Mode and ModeActivate
SUT.CalculateSUTPar-ams	SUT.CalculateParams	Renamed
SUT.TuneRuleSUT	SUT.TuneRule	Renamed
TIR.CalculateTIRParams	TIR.CalculateParams	Renamed
TIR.TuneRuleTIR	TIR.TuneRule	Renamed
Retain.Mode	Mode	Function Declaration of static for in-out parameters

Input parameters of PID_3Step V2 (S7-1200, S7-1500)

Table 11-73

Parameter	Data type	Default	Description
Setpoint	REAL	0.0	Setpoint of the PID controller in automatic mode
Input	REAL	0.0	A tag of the user program is used as source for the process value. If you are using parameter Input, then Config.InputPer-On = FALSE must be set.
Input_PER	INT	0	An analog input is used as the source of the process value. If you are using parameter Input_PER, then Config.InputPer-On = TRUE must be set.

Parameter	Data type	Default	Description
Actuator_H	BOOL	FALSE	Digital position feedback of the valve for the high endstop If Actuator_H = TRUE, the valve is at the high endstop and is no longer moved towards this direction.
Actuator_L	BOOL	FALSE	Digital position feedback of the valve for the low endstop If Actuator_L = TRUE, the valve is at the low endstop and is no longer moved towards this direction.
Feedback	REAL	0.0	Position feedback of the valve If you are using parameter Feedback, then Config.Feedback-PerOn = FALSE must be set.
Feedback_PER	INT	0	Analog position feedback of a valve If you are using parameter Feedback_PER, then Config.Feed-backPerOn = TRUE must be set. Feedback_PER is scaled based on the tags: • Config.FeedbackScaling.LowerPointIn • Config.FeedbackScaling.UpperPointIn • Config.FeedbackScaling.LowerPointOut • Config.FeedbackScaling.UpperPointOut
Disturbance	REAL	0.0	Disturbance variable or precontrol value
ManualEnable	BOOL	FALSE	• A FALSE -> TRUE edge activates "manual mode", while State = 4, Mode remains unchanged. As long as ManualEnable = TRUE, you cannot change the operating mode via a rising edge at ModeActivate or use the commissioning dialog. • A TRUE -> FALSE edge activates the operating mode that is specified by Mode. We recommend that you change the operating mode using ModeActivate only.
ManualValue	REAL	0.0	In manual mode, the absolute position of the valve is specified. ManualValue is only evaluated if you are using Output_PER, **or** if position feedback is available.
Manual_UP	BOOL	FALSE	• Manual_UP = TRUE The valve is opened even if you are using Output_PER or a position feedback. The valve is no longer moved if the high endstop has been reached. See also Config.VirtualActuatorLimit • Manual_UP = FALSE If you are using Output_PER or a position feedback, the valve is moved to ManualValue. Otherwise, the valve is no longer moved. If Manual_UP and Manual_DN are set to TRUE simultaneously, the valve is not moved.

Parameter	Data type	Default	Description
Manual_DN	BOOL	FALSE	• Manual_DN = TRUE The valve is closed even if you are using Output_PER or a position feedback. The valve is no longer moved if the low endstop has been reached. See also Config.VirtualActuatorLimit • Manual_DN = FALSE If you are using Output_PER or a position feedback, the valve is moved to ManualValue. Otherwise, the valve is no longer moved.
ErrorAck	BOOL	FALSE	• FALSE -> TRUE edge ErrorBits and Warning are reset.
Reset	BOOL	FALSE	Restarts the controller. • FALSE -> TRUE edge – Switch to "Inactive" mode – ErrorBits and Warnings are reset. • As long as Reset = TRUE, – PID_3Step remains in "Inactive" mode (State = 0). – You cannot change the operating mode with Mode and ModeActivate or ManualEnable. – You cannot use the commissioning dialog. • TRUE -> FALSE edge – If ManualEnable = FALSE, PID_3Step switches to the operating mode that is saved in Mode. – If Mode = 3, the integral action is treated as configured with the tag IntegralResetMode.
ModeActivate	BOOL	FALSE	• FALSE -> TRUE edge PID_3Step switches to the operating mode that is saved in the Mode parameter.

Output parameters of PID_3Step V2 (S7-1200, S7-1500)

Table 11-74

Parameter	Data type	Default	Description
ScaledInput	REAL	0.0	Scaled process value
ScaledFeedback	REAL	0.0	Scaled position feedback For an actuator without position feedback, the position of the actuator indicated by ScaledFeedback is very imprecise. ScaledFeedback may only be used for rough estimation of the current position in this case.
Output_UP	BOOL	FALSE	Digital output value for opening the valve If Config.OutputPerOn = FALSE, the Output_UP parameter is used.

Parameter	Data type	Default	Description
Output_DN	BOOL	FALSE	Digital output value for closing the valve If Config.OutputPerOn = FALSE, the Output_DN parameter is used.
Output_PER	INT	0	Analog output value If Config.OutputPerOn = TRUE, Output_PER is used. Use Output_PER if you are using a valve as actuator which is triggered via an analog output and controlled with a continuous signal, e.g. 0...10 V or 4...20 mA. The value at Output_PER corresponds to the target position of the valve, e.g. Output_PER = 13824, when the valve is to be opened by 50%.
SetpointLimit_H	BOOL	FALSE	If SetpointLimit_H = TRUE, the absolute setpoint high limit is reached (Setpoint ≥ Config.SetpointUpperLimit). The setpoint is limited to Config.SetpointUpperLimit .
SetpointLimit_L	BOOL	FALSE	If SetpointLimit_L = TRUE, the absolute setpoint low limit has been reached (Setpoint ≤ Config.SetpointLowerLimit). The setpoint is limited to Config.SetpointLowerLimit .
InputWarning_H	BOOL	FALSE	If InputWarning_H = TRUE, the process value has reached or exceeded the warning high limit.
InputWarning_L	BOOL	FALSE	If InputWarning_L = TRUE, the process value has reached or fallen below the warning low limit.
State	INT	0	The State parameter (Page 4276) shows the current operating mode of the PID controller. You can change the operating mode using the input parameter Mode and a rising edge at ModeActivate. • State = 0: Inactive • State = 1: Pretuning • State = 2: Fine tuning • State = 3: Automatic mode • State = 4: Manual mode • State = 5: Approach substitute output value • State = 6: Transition time measurement • State = 7: Error monitoring • State = 8: Approach substitute output value with error monitoring • State = 10: Manual mode without endstop signals
Error	BOOL	FALSE	If Error = TRUE, at least one error message is pending in this cycle.
ErrorBits	DWORD	DW#16#0	The ErrorBits parameter (Page 4281) shows which error messages are pending. ErrorBits is retentive and is reset upon a rising edge at Reset or ErrorAck.

See also

Parameters State and Mode V2 (Page 4276)

Parameter ErrorBits V2 (Page 4281)

In/out parameters of PID-3Step V2 (S7-1200, S7-1500)

Table 11-75

Parameter	Data type	Default	Description
Mode	INT	4	At the Mode parameter, you specify the operating mode to which PID_3Step is to switch. Options are: • Mode = 0: Inactive • Mode = 1: Pretuning • Mode = 2: Fine tuning • Mode = 3: Automatic mode • Mode = 4: Manual mode • Mode = 6: Transition time measurement • Mode = 10: Manual mode without endstop signals The operating mode is activated by: • Rising edge at ModeActivate • Falling edge at Reset • Falling edge at ManualEnable • Cold restart of CPU if RunModeByStartup = TRUE Mode is retentive. A detailed description of the operating modes can be found in Parameters State and Mode V2 (Page 4276).

Static tags of PID_3Step V2 (S7-1200, S7-1500)

You must not change tags that are not listed. These are used for internal purposes only.

Tag	Data type	Default	Description
ManualUpInternal	BOOL	FALSE	In manual mode, each rising edge opens the valve by 5% of the total control range or for the duration of the minimum motor transition time. ManualUpInternal is only evaluated if you are not using Output_PER or a position feedback. This tag is used in the commissioning dialog.
ManualDnInternal	BOOL	FALSE	In manual mode, every rising edge closes the valve by 5% of the total control range or for the duration of the minimum motor transition time. ManualDnInternal is only evaluated if you are not using Output_PER or position feedback. This tag is used in the commissioning dialog.
ActivateRecoverMode	BOOL	TRUE	The ActivateRecoverMode V2 (Page 4284) tag determines the reaction to error.
RunModeByStartup	BOOL	TRUE	Activate operating mode at Mode parameter after CPU restart If RunModeByStartup = TRUE, PID_3Step starts in the operating mode saved in the Mode parameter after CPU startup. If RunModeByStartup = FALSE, PID_3Step remains in "Inactive" mode after CPU startup.
LoadBackUp	BOOL	FALSE	If LoadBackUp = TRUE, the last set of PID parameters is reloaded. The set was saved prior to the last tuning. LoadBackUp is automatically set back to FALSE.

Tag	Data type	Default	Description
PhysicalUnit	INT	0	Unit of measurement of the process value and setpoint, e.g., ºC, or ºF.
PhysicalQuantity	INT	0	Physical quantity of the process value and setpoint, e.g., temperature
ErrorBehaviour	BOOL	FALSE	If ErrorBehaviour = FALSE and an error has occurred, the valve stays at its current position and the controller switches directly to "Inactive" or "Error monitoring" mode. If ErrorBehaviour = TRUE and an error occurs, the actuator moves to the substitute output value and only then switches to "Inactive" or "Error monitoring" mode. If the following errors occur, you can no longer move the valve to a configured substitute output value. • 2000h: Invalid value at Feedback_PER parameter. • 4000h: Invalid value at Feedback parameter. • 8000h: Error during digital position feedback. • 20000h: Invalid value at SavePosition tag.
Warning	DWORD	DW#16#0	The Warning tag (Page 4276) shows the warnings since Reset = TRUE or ErrorAck =TRUE. Warning is retentive. Cyclic warnings (for example, process value warning) are shown until the cause of the warning is removed. They are automatically deleted once their cause has gone. Non-cyclic warnings (for example, point of inflection not found) remain and are deleted like errors.
SavePosition	REAL	0.0	Substitute output value If ErrorBehaviour = TRUE, the actuator is moved to a position that is safe for the plant when an error occurs. As soon as the substitute output value has been reached, PID_3Step switches the operating mode according to ActivateRecoverMode.
CurrentSetpoint	REAL	0.0	Currently active setpoint. This value is frozen at the start of tuning.
CancelTuningLevel	REAL	10.0	Permissible fluctuation of setpoint during tuning. Tuning is not canceled until: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel
Progress	REAL	0.0	Progress of tuning as a percentage (0.0 - 100.0)
Config.InputPerOn	BOOL	TRUE	If InputPerOn = TRUE, the Input_PER parameter is used. If InputPerOn = FALSE, the Input parameter is used.
Config.OutputPerOn	BOOL	FALSE	If OutputPerOn = TRUE, the Output_PER parameter is used. If OutputPerOn = FALSE, the Ouput_UP and Output_DN parameters are used.
Config.InvertControl	BOOL	FALSE	Invert control logic If InvertControl = TRUE, an increasing control deviation causes a reduction in the output value.
Config.FeedbackOn	BOOL	FALSE	If FeedbackOn = FALSE, a position feedback is simulated. Position feedback is generally activated when Feedback-On = TRUE.

Tag	Data type	Default	Description
Config.FeedbackPerOn	BOOL	FALSE	FeedbackPerOn is only effective when FeedbackOn = TRUE. If FeedbackPerOn = TRUE, the analog input is used for the position feedback (Feedback_PER parameter). If FeedbackPerOn = FALSE, the Feedback parameter is used for the position feedback.
Config.ActuatorEndStopOn	BOOL	FALSE	If ActuatorEndStopOn = TRUE, the digital position feedback Actuator_L and Actuator_H are taken into consideration.
Config.InputUpperLimit	REAL	120.0	High limit of the process value Input and Input_PER are monitored to ensure adherence to this limit. At the I/O input, the process value can be a maximum of 18% higher than the standard range (overrange). An error is no longer signaled due to a violation of the "Process value high limit". Only a wire-break and a short-circuit are recognized and PID_3Step reacts according to the configured reaction to error. InputUpperLimit > InputLowerLimit
Config.InputLowerLimit	REAL	0.0	Low limit of the process value InputLowerLimit < InputUpperLimit
Config.InputUpperWarning	REAL	+3.402822e+38	Warning high limit of the process value If you set InputUpperWarning outside the process value limits, the configured absolute process value high limit is used as the warning high limit. If you configure InputUpperWarning within the process value limits, this value is used as the warning high limit. InputUpperWarning > InputLowerWarning InputUpperWarning ≤ InputUpperLimit
Config.InputLowerWarning	REAL	-3.402822e+38	Warning low limit of the process value If you set InputLowerWarning outside the process value limits, the configured absolute process value low limit is used as the warning low limit. If you configure InputLowerWarning within the process value limits, this value is used as the warning low limit. InputLowerWarning < InputUpperWarning InputLowerWarning ≥ InputLowerLimit
Config.OutputUpperLimit	REAL	100.0	High limit of output value For details, see OutputLowerLimit
Config.OutputLowerLimit	REAL	0.0	Low limit of output value If OutputPerOn = TRUE or FeedbackOn = TRUE, the range of values from -100% to +100%, including zero, is valid. At -100%, Output = -27648; at +100% Output = 27648 If OutputPerOn = FALSE, the range of values from 0% to 100% is valid. The valve is completely closed at 0% and completely opened at 100%.
Config.SetpointUpperLimit	REAL	+3.402822e+38	High limit of setpoint If you set SetpointUpperLimit outside the process value limits, the configured absolute process value high limit is preassigned as the setpoint high limit. If you configure SetpointUpperLimit within the process value limits, this value is used as the setpoint high limit.

Tag	Data type	Default	Description
Config.SetpointLowerLimit	REAL	-3.402822e+38	Low limit of the setpoint If you set SetpointLowerLimit outside the process value limits, the configured absolute process value low limit is preassigned as the setpoint low limit. If you set SetpointLowerLimit within the process value limits, this value is used as the setpoint low limit.
Config.MinimumOnTime	REAL	0.0	Minimum ON time Minimum time in seconds for which the servo drive must be switched on. Config.MinimumOnTime is only effective if Output_UP and Output_DN are used (Config.OutputPerOn = FALSE).
Config.MinimumOffTime	REAL	0.0	Minimum OFF time Minimum time in seconds for which the servo drive must be switched off. Config.MinimumOffTime is only effective if Output_UP and Output_DN are used (Config.OutputPerOn = FALSE).
Config.VirtualActuatorLimit	REAL	150.0	If all the following conditions have been satisfied, the actuator is moved in one direction for the maximum period of VirtualActuatorLimit × Retain.TransitTime/100 and the warning 2000h is output: • Config.OutputPerOn = FALSE • Config.ActuatorEndStopOn = FALSE • Config.FeedbackOn = FALSE If Config.OutputPerOn = FALSE and Config.ActuatorEndStopOn = TRUE or Config.FeedbackOn = TRUE, only the warning 2000h is output. If Config.OutputPerOn = TRUE, VirtualActuatorLimit is not taken into consideration. As of PID_3Step Version 2.3 the monitoring and limiting of the travel time can be deactivated with Config.VirtualActuatorLimit = 0.0.
Config.InputScaling.UpperPointIn	REAL	27648.0	Scaling Input_PER high Input_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the InputScaling structure.
Config.InputScaling.LowerPointIn	REAL	0.0	Scaling Input_PER low Input_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the InputScaling structure.
Config.InputScaling.UpperPointOut	REAL	100.0	Scaled high process value Input_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the InputScaling structure.
Config.InputScaling.LowerPointOut	REAL	0.0	Scaled low process value Input_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the InputScaling structure.

Tag	Data type	Default	Description
Config.FeedbackScaling.UpperPointIn	REAL	27648.0	Scaling Feedback_PER high Feedback_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the FeedbackScaling structure.
Config.FeedbackScaling.LowerPointIn	REAL	0.0	Scaling Feedback_PER low Feedback_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the FeedbackScaling structure.
Config.FeedbackScaling.UpperPointOut	REAL	100.0	High endstop Feedback_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the FeedbackScaling structure.
Config.FeedbackScaling.LowerPointOut	REAL	0.0	Low endstop Feedback_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the FeedbackScaling structure.
GetTransitTime.InvertDirection	BOOL	FALSE	If InvertDirection = FALSE, the valve is fully opened, closed, and then reopened in order to determine the valve transition time. If InvertDirection = TRUE, the valve is fully closed, opened, and then closed again.
GetTransitTime.SelectFeedback	BOOL	FALSE	If SelectFeedback = TRUE, then Feedback_PER, or Feedback is taken into consideration in the transition time measurement. If SelectFeedback = FALSE, then Actuator_H and Actuator_L are taken into consideration in the transition time measurement.
GetTransitTime.State	INT	0	Current phase of the transition time measurement • State = 0: Inactive • State = 1: Open valve completely • State = 2: Close valve completely • State = 3: Move valve to target position (NewOutput) • State = 4: Transition time measurement successfully completed • State = 5: Transition time measurement canceled
GetTransitTime.NewOutput	REAL	0.0	Target position for transition time measurement with position feedback The target position must be between "High endstop" and "Low endstop". The difference between NewOutput and ScaledFeedback must be at least 50% of the permissible control range.
CycleTime.StartEstimation	BOOL	TRUE	If StartEstimation = TRUE, the measurement of the PID_3Step sampling time is started. CycleTime.StartEstimation = FALSE once measurement is complete.
CycleTime.EnEstimation	BOOL	TRUE	If EnEstimation = TRUE, the PID_3Step sampling time is calculated. If CycleTime.EnEstimation = FALSE, the PID_3Step sampling time is not calculated and you need to correct the configuration of CycleTime.Value manually.

Tag	Data type	Default	Description
CycleTime.EnMonitoring	BOOL	TRUE	If EnMonitoring = TRUE, the PID_3Step sampling time is monitored. If it is not possible to execute PID_3Step within the sampling time, the error 0800h is output and the operating mode is switched. ActivateRecoverMode and ErrorBehaviour determine which operating mode is switched to. If EnMonitoring = FALSE, the PID_3Step sampling time is not monitored, the error 0800h is not output, and the operating mode is not switched.
CycleTime.Value	REAL	0.1	PID_3Step sampling time in seconds CycleTime.Value is determined automatically and is usually equivalent to the cycle time of the calling OB.
CtrlParamsBackUp.SetByUser	BOOL	FALSE	Saved value of Retain.CtrlParams.SetByUser You can reload values from the CtrlParamsBackUp structure with LoadBackUp = TRUE.
CtrlParamsBackUp.Gain	REAL	1.0	Saved proportional gain
CtrlParamsBackUp.Ti	REAL	20.0	Saved integration time in seconds
CtrlParamsBackUp.Td	REAL	0.0	Saved derivative action time in seconds
CtrlParamsBackUp.TdFiltRatio	REAL	0.2	Saved derivative delay coefficient
CtrlParamsBackUp.PWeighting	REAL	1.0	Saved proportional action weighting
CtrlParamsBackUp.DWeighting	REAL	1.0	Saved derivative action weighting
CtrlParamsBackUp.Cycle	REAL	1.0	Saved sampling time of PID algorithm in seconds
CtrlParamsBackUp.InputDeadBand	REAL	0.0	Saved dead band width of the control deviation
PIDSelfTune.SUT.CalculateParams	BOOL	FALSE	The properties of the controlled system are saved during tuning. If CalculateParams = TRUE, the PID parameters are recalculated on the basis of these properties. The PID parameters are calculated using the method set in TuneRule. CalculateParams is set to FALSE following calculation.
PIDSelfTune.SUT.TuneRule	INT	1	Methods used to calculate parameters during pretuning: • SUT.TuneRule = 0: PID fast I • SUT.TuneRule = 1: PID slow I • SUT.TuneRule = 2: Chien, Hrones and Reswick PID • SUT.TuneRule = 3: Chien, Hrones, Reswick PI • SUT.TuneRule = 4: PID fast II • SUT.TuneRule = 5: PID slow II
PIDSelfTune.SUT.State	INT	0	The SUT.State tag indicates the current phase of pretuning: • State = 0: Initialize pretuning • State = 50: Determine start position without position feedback • State = 100: Calculate the standard deviation • State = 200: Find the point of inflection • State = 300: Determine the rise time • State = 9900: Pretuning successful • State = 1: Pretuning not successful

Tag	Data type	Default	Description
PIDSelfTune.TIR.RunIn	BOOL	FALSE	With the RunIn tag, you can specify that fine tuning can also be performed without pretuning. • RunIn = FALSE Pretuning is started when fine tuning is started from inactive or manual mode. If fine tuning is started from automatic mode, the system uses the existing PID parameters to control to the setpoint. Only then will fine tuning start. If pretuning is not possible, PID_3Step switches to the mode from which tuning was started. • RunIn = TRUE The pretuning is skipped. PID_3Step attempts to reach the setpoint with the minimum or maximum output value. This can produce increased overshoot. Only then will fine tuning start. RunIn is set to FALSE after fine tuning.
PIDSelfTune.TIR.Calculate-Params	BOOL	FALSE	The properties of the controlled system are saved during tuning. If CalculateParams = TRUE, the PID parameters are recalculated on the basis of these properties. The PID parameters are calculated using the method set in TuneRule. CalculateParams is set to FALSE following calculation.
PIDSelfTune.TIR.TuneRule	INT	0	Methods used to calculate parameters during fine tuning: • TIR.TuneRule = 0: PID automatic • TIR.TuneRule = 1: PID fast • TIR.TuneRule = 2: PID slow • TIR.TuneRule = 3: Ziegler-Nichols PID • TIR.TuneRule = 4: Ziegler-Nichols PI • TIR.TuneRule = 5: Ziegler-Nichols P
PIDSelfTune.TIR.State	INT	0	The TIR.State tag indicates the current phase of fine tuning: • State = -100 Fine tuning is not possible. Pretuning will be performed first. • State = 0: Initialize fine tuning • State = 200: Calculate the standard deviation • State = 300: Attempt to reach the setpoint with the maximum or minimum output value • State = 400: Attempt to reach the setpoint with existing PID parameters (if pretuning was successful) • State = 500: Determine oscillation and calculate parameters • State = 9900: Fine tuning successful • State = 1: Fine tuning not successful
Retain.TransitTime	REAL	30.0	Motor transition time in seconds Time in seconds the actuating drive requires to move the valve from the closed to the opened state. TransitTime is retentive.

Tag	Data type	Default	Description
Retain.CtrlParams.SetByUser	BOOL	FALSE	If SetByUser = FALSE, the PID parameters are determined automatically and PID_3Step operates with a dead zone at the output value. The dead band width is calculated during tuning on the basis of the standard deviation of the output value and saved in Retain.CtrlParams.OutputDeadBand. If SetByUser = TRUE, the PID parameters are entered manually and PID_3 Step operates without a dead zone at the output value. Retain.CtrlParams.OutputDeadBand = 0.0 SetByUser is retentive.
Retain.CtrlParams.Gain	REAL	1.0	Active proportional gain To invert the control logic, use the Config.InvertControl tag. Negative values at Gain also invert the control logic. We recommend you use only InvertControl to set the control logic. The control logic is also inverted if InvertControl = TRUE and Gain < 0.0. Gain is retentive.
Retain.CtrlParams.Ti	REAL	20.0	• Ti > 0.0: Active integration time in seconds • Ti = 0.0: Integral action is deactivated Ti is retentive.
Retain.CtrlParams.Td	REAL	0.0	• Td > 0.0: Active derivative action time in seconds • Td = 0.0: Derivative action is deactivated Td is retentive.
Retain.CtrlParams.TdFiltRatio	REAL	0.2	Active derivative delay coefficient The derivative delay coefficient delays the effect of the derivative action. Derivative delay = derivative action time × derivative delay coefficient • 0.0: Derivative action is effective for one cycle only and therefore almost not effective. • 0.5: This value has proved useful in practice for controlled systems with **one** dominant time constant. • > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed. TdFiltRatio is retentive.
Retain.CtrlParams.PWeighting	REAL	1.0	Active proportional action weighting The proportional action may weaken with changes to the setpoint. Values from 0.0 to 1.0 are applicable. • 1.0: Proportional action for setpoint change is fully effective • 0.0: Proportional action for setpoint change is not effective The proportional action is always fully effective when the process value is changed. PWeighting is retentive.

Tag	Data type	Default	Description
Retain.CtrlParams.DWeight-ing	REAL	1.0	Active derivative action weighting The derivative action may weaken with changes to the setpoint. Values from 0.0 to 1.0 are applicable. • 1.0: Derivative action is fully effective upon setpoint change • 0.0: Derivative action is not effective upon setpoint change The derivative action is always fully effective when the process value is changed. DWeighting is retentive.
Retain.CtrlParams.Cycle	REAL	1.0	Active sampling time of PID algorithm in seconds, rounded to an integer multiple of the cycle time of the calling OB. Cycle is retentive.
Retain.CtrlParams.InputDead-Band	REAL	0.0	Dead band width of the control deviation InputDeadBand is retentive.

Note

Change the tags listed in this table in "Inactive" mode to prevent malfunction of the PID controller.

See also

Parameters State and Mode V2 (Page 4276)

Tag ActivateRecoverMode V2 (Page 4284)

Downloading technology objects to device (Page 6000)

Parameters State and Mode V2 (S7-1200, S7-1500)

Correlation of the parameters

The State parameter shows the current operating mode of the PID controller. You cannot change the State parameter.

With a rising edge at ModeActivate, PID_3Step switches to the operating mode saved in the Mode in-out parameter.

When the CPU is switched on or switches from Stop to RUN mode, PID_3Step starts in the operating mode that is saved in the Mode parameter. To leave PID_3Step in "Inactive" mode, set RunModeByStartup = FALSE.

Meaning of values

State	Description of operating mode
0	Inactive The controller is switched off and no longer changes the valve position.
1	Pretuning The pretuning determines the process response to a pulse of the output value and searches for the point of inflection. The PID parameters are calculated from the maximum rate of rise and dead time of the controlled system. You obtain the best PID parameters when you perform pretuning and fine tuning. Pretuning requirements: • The motor transition time has been configured or measured. • Inactive (State = 0), manual mode (State = 4), or automatic mode (State = 3) • ManualEnable = FALSE • Reset = FALSE • The setpoint and the process value lie within the configured limits. The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher as compared to the noise. This is most likely the case in operating modes "Inactive" and "manual mode". The setpoint is frozen in the CurrentSetpoint tag. Tuning is canceled when: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel Before the PID parameters are recalculated, they are backed up and can be reactivated with LoadBackUp. The controller switches to automatic mode following successful pretuning. If pretuning is unsuccessful, the switchover of operating mode is dependent on ActivateRecoverMode and ErrorBehaviour. The pretuning phase is indicated with the SUT.State tag.

State	Description of operating mode
2	Fine tuning Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are recalculated based on the amplitude and frequency of this oscillation. PID parameters from fine tuning usually have better master control and disturbance characteristics than PID parameters from pretuning. You obtain the best PID parameters when you perform pretuning and fine tuning. PID_3Step automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. The setpoint is frozen in the CurrentSetpoint tag. Tuning is canceled when: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel The PID parameters are backed up before fine tuning. They can be reactivated with LoadBackUp. Requirements for fine tuning: • The motor transition time has been configured or measured. • The setpoint and the process value lie within the configured limits. • ManualEnable = FALSE • Reset = FALSE • Automatic (State = 3), inactive (State = 0) or manual (State = 4) mode Fine tuning proceeds as follows when started from: • Automatic mode (State = 3) Start fine tuning from automatic mode if you wish to improve the existing PID parameters through tuning. PID_3Step controls the system using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start. • Inactive (State = 0) or manual mode (State = 4) If the requirements for pretuning are met, pretuning is started. The determined PID parameters will be used for control until the control loop has stabilized and the requirements for fine tuning have been met. If PIDSelfTune.TIR.RunIn = TRUE, pretuning is skipped and an attempt is made to reach the setpoint with the minimum or maximum output value. This can produce increased overshoot. Fine tuning then starts automatically. The controller switches to automatic mode following successful fine tuning. If fine tuning is unsuccessful, the switchover of operating mode is dependent on ActivateRecoverMode and ErrorBehaviour. The fine tuning phase is indicated with the TIR.State tag.
3	Automatic mode In automatic mode, PID_3Step controls the controlled system in accordance with the parameters specified. The controller switches to automatic mode if one the following requirements is fulfilled: • Pretuning successfully completed • Fine tuning successfully completed • Changing of the Mode in-out parameter to the value 3 and a rising edge at ModeActivate. The switchover from automatic mode to manual mode is only bumpless if carried out in the commissioning editor. The ActivateRecoverMode tag is taken into consideration in automatic mode.

State	Description of operating mode
4	Manual mode In manual mode, you specify manual output values in the Manual_UP and Manual_DN parameters or ManualValue parameter. Whether or not the actuator can be moved to the output value in the event of an error is described in the ErrorBits parameter. You can also activate this operating mode using ManualEnable = TRUE. We recommend that you change the operating mode using Mode and ModeActivate only. The switchover from manual mode to automatic mode is bumpless. Manual mode is also possible when an error is pending.
5	Approach substitute output value This operating mode is activated in the event of an error, if Errorbehaviour = TRUE and ActivateRecoverMode = FALSE.. PID_3Step moves the actuator to the substitute output value and then switches to "Inactive" mode.
6	Transition time measurement The time that the motor needs to completely open the valve from the closed condition is determined. This operating mode is activated when Mode = 6 and ModeActivate = TRUE is set. If endstop signals are used to measure the transition time, the valve will be opened completely from its current position, closed completely, and opened completely again. If GetTransitTime.InvertDirection = TRUE, this behavior is inverted. If position feedback is used to measure the transition time, the actuator will be moved from its current position to a target position. The output value limits are not taken into consideration during the transition time measurement. The actuator can travel to the high or the low endstop.
7	Error monitoring The control algorithm is switched off and no longer changes the valve position. This operating mode is activated instead of "Inactive" mode in the event of an error. All the following conditions must be met: • Automatic mode (Mode = 3) • Errorbehaviour = FALSE • ActivateRecoverMode = TRUE • One or more errors have occurred in which ActivateRecoverMode (Page 4284) is effective. As soon as the errors are no longer pending, PID_3Step switches back to automatic mode.
8	Approach substitute output value with error monitoring This operating mode is activated instead of "approach substitute output value" mode when an error occurs. PID_3Step moves the actuator to the substitute output value and then switches to "error monitoring" mode. All the following conditions must be met: • Automatic mode (Mode = 3) • Errorbehaviour = TRUE • ActivateRecoverMode = TRUE • One or more errors have occurred in which ActivateRecoverMode (Page 4284) is effective. As soon as the errors are no longer pending, PID_3Step switches back to automatic mode.
10	Manual mode without endstop signals The endstop signals are **not** taken into consideration, even though Config.ActuatorEndStopOn = TRUE. The output value limits are not taken into consideration. Otherwise, PID_3Step behaves the same as in manual mode.

ENO characteristics

If State = 0, then ENO = FALSE.

If State ≠ 0, then ENO = TRUE.

Automatic switchover of operating mode during commissioning

Automatic mode is activated following successful pretuning or fine tuning. The following table shows how Mode and State change during successful pretuning.

Cycle no.	Mode	State	Action
0	4	4	Set Mode = 1
1	1	4	Set ModeActivate = TRUE
1	4	1	Value of State is saved in Mode parameter Pretuning is started
n	4	1	Pretuning successfully completed
n	3	3	Automatic mode is started

PID_3Step automatically switches the operating mode in the event of an error. The following table shows how Mode and State change during pretuning with errors.

Cycle no.	Mode	State	Action
0	4	4	Set Mode = 1
1	1	4	Set ModeActivate = TRUE
1	4	1	Value of State is saved in Mode parameter Pretuning is started
n	4	1	Pretuning canceled
n	4	4	Manual mode is started

If ActivateRecoverMode = TRUE, the operating mode that is saved in the Mode parameter is activated. At the start of transition time measurement, pretuning, or fine tuning, PID_3Step saved the value of State in the Mode in/out parameter. PID_3Step therefore switches to the operating mode from which transition time measurement or tuning was started.

If ActivateRecoverMode = FALSE, "Inactive" or "Approach substitute output value" mode is activated.

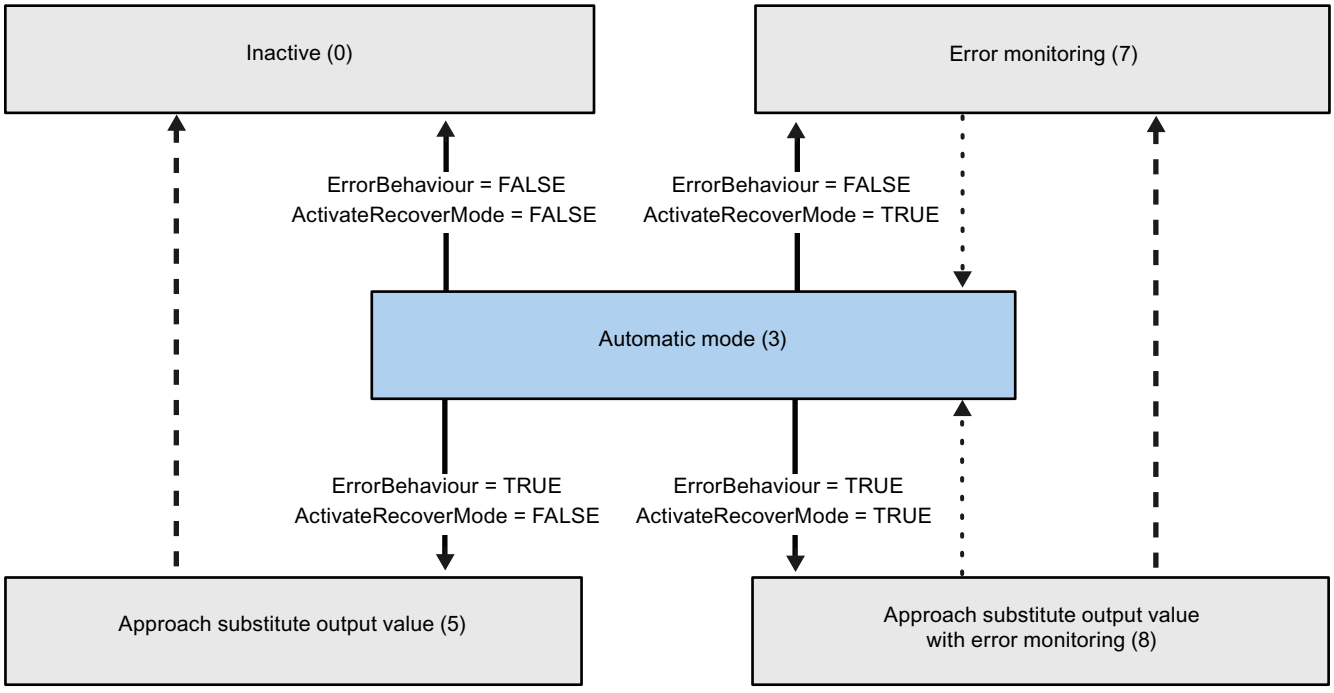
Automatic switchover of operating mode after transition time measurement

If ActivateRecoverMode = TRUE, the operating mode that is saved in the Mode parameter is activated after successful transition time measurement.

If ActivateRecoverMode = FALSE, the system switches to "Inactive" operating mode after successful transition time measurement.

Automatic switchover of operating mode in automatic mode

PID_3Step automatically switches the operating mode in the event of an error. The following diagram illustrates the influence of ErrorBehaviour and ActivateRecoverMode on this switchover of operating mode.



Automatic switchover of operating mode in the event of an error

Automatic switchover of operating mode once the current operation has been completed.

Automatic switchover of operating mode when error is no longer pending.

See also

Tag ActivateRecoverMode V2 (Page 4284)

Parameter ErrorBits V2 (Page 4281)

Parameter ErrorBits V2 (S7-1200, S7-1500)

If several errors are pending simultaneously, the values of the ErrorBits are displayed with binary addition. The display of ErrorBits = 0003h, for example, indicates that the errors 0001h and 0002h are pending simultaneously.

If there is a position feedback, PID_3Step uses ManualValue as output value in manual mode. The exception is Errorbits = 10000h.

ErrorBits (DW#16#...)	Description
0000	There is no error.
0001	The "Input" parameter is outside the process value limits. • Input > Config.InputUpperLimit or • Input < Config.InputLowerLimit If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_3Step remains in automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter.
0002	Invalid value at "Input_PER" parameter. Check whether an error is pending at the analog input. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_3Step switches to "Approach substitute output value with error monitoring" or "Error monitoring" mode. As soon as the error is no longer pending, PID_3Step switches back to automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter.
0004	Error during fine tuning. Oscillation of the process value could not be maintained. If ActivateRecoverMode = TRUE before the error occurred, PID_3Step cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0010	The setpoint was changed during tuning. You can set the permitted fluctuation of the setpoint at the CancelTuningLevel tag. If ActivateRecoverMode = TRUE before the error occurred, PID_3Step cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0020	Pretuning is not permitted during fine tuning. If ActivateRecoverMode = TRUE before the error occurred, PID_3Step remains in fine tuning mode.
0080	Error during pretuning. The output value limits are not configured correctly or the actual value does not react as expected. Check whether the limits of the output value are configured correctly and match the control logic. Also make sure that the actual value does not oscillate strongly before starting pretuning. If ActivateRecoverMode = TRUE before the error occurred, PID_3Step cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0100	Error during fine tuning resulted in invalid parameters. If ActivateRecoverMode = TRUE before the error occurred, PID_3Step cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0200	Invalid value at "Input" parameter: Value has an invalid number format. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_3Step switches to "Approach substitute output value with error monitoring" or "Error monitoring" mode. As soon as the error is no longer pending, PID_3Step switches back to automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter.

ErrorBits (DW#16#...)	Description
0400	Calculation of output value failed. Check the PID parameters. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_3Step switches to "Approach substitute output value with error monitoring" or "Error monitoring" mode. As soon as the error is no longer pending, PID_3Step switches back to automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter.
0800	Sampling time error: PID_3Step is not called within the sampling time of the cyclic interrupt OB. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_3Step remains in automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter. If this error occurred during simulation with PLCSIM, see the notes under Simulating PID_3Step V2 with PLCSIM (Page 6079).
1000	Invalid value at "Setpoint" parameter: Value has an invalid number format. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_3Step switches to "Approach substitute output value with error monitoring" or "Error monitoring" mode. As soon as the error is no longer pending, PID_3Step switches back to automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter.
2000	Invalid value at Feedback_PER parameter. Check whether an error is pending at the analog input. The actuator cannot be moved to the substitute output value and remains in its current position. In manual mode, you can change the position of the actuator only with Manual_UP and Manual_DN, and not with ManualValue. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter.
4000	Invalid value at Feedback parameter. Value has an invalid number format. The actuator cannot be moved to the substitute output value and remains in its current position. In manual mode, you can change the position of the actuator only with Manual_UP and Manual_DN, and not with ManualValue. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter.

ErrorBits (DW#16#...)	Description
8000	Error during digital position feedback. Actuator_H = TRUE and Actuator_L = TRUE. The actuator cannot be moved to the substitute output value and remains in its current position. Manual mode is not possible in this state. In order to move the actuator from this state, you must deactivate the "Actuator endstop" (Config.ActuatorEndStopOn = FALSE) or switch to manual mode without endstop signals (Mode = 10). If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode. If pretuning, fine tuning, or transition time measurement mode and ActivateRecoverMode = TRUE were active before the error occurred, PID_3Step switches to the operating mode that is saved in the Mode parameter.
10000	Invalid value at ManualValue parameter. Value has an invalid number format. The actuator cannot be moved to the manual value and remains in its current position. Specify a valid value in ManualValue or move the actuator in manual mode with Manual_UP and Manual_DN.
20000	Invalid value at SavePosition tag. Value has an invalid number format. The actuator cannot be moved to the substitute output value and remains in its current position.
40000	Invalid value at Disturbance parameter. Value has an invalid number format. If automatic mode was active and ActivateRecoverMode = TRUE before the error occurred, Disturbance is set to zero. PID_3Step remains in automatic mode. If pretuning or fine tuning mode was active and ActivateRecoverMode = TRUE before the error occurred, PID_3Step switches to the operating mode saved in the Mode parameter. If Disturbance in the current phase has no effect on the output value, tuning is not be canceled. The error has no effect during transition time measurement.

Tag ActivateRecoverMode V2 (S7-1200, S7-1500)

The ActivateRecoverMode tag determines the reaction to error. The Error parameter indicates if an error is pending. When the error is no longer pending, Error = FALSE. The ErrorBits parameter shows which errors have occurred.

NOTICE
Your system may be damaged. If ActivateRecoverMode = TRUE, PID_3Step remains in automatic mode even if the process limit values are exceeded. This may damage your system. It is essential to configure how your controlled system reacts in the event of an error to protect your system from damage.

Automatic mode

ActivateRecoverMode	Description
FALSE	In the event of an error, PID_3Step switches to "Inactive" or "Approach substitute output value" mode. The controller is only activated by a falling edge at Reset or a rising edge at ModeActivate.
TRUE	If errors occur frequently in automatic mode, this setting has a negative effect on the control response, because PID_3Step switches between the calculated output value and the substitute output value at each error. In this case, check the ErrorBits parameter and eliminate the cause of the error. If one or more of the following errors occur, PID_3Step stays in automatic mode: • 0001h: The "Input" parameter is outside the process value limits. • 0800h: Sampling time error • 40000h: Invalid value at Disturbance parameter. If one or more of the following errors occur, PID_3Step switches to "Approach substitute output value with error monitoring" or "Error monitoring" mode: • 0002h: Invalid value at Input_PER parameter. • 0200h: Invalid value at Input parameter. • 0400h: Calculation of output value failed. • 1000h: Invalid value at Setpoint parameter. If one or more of the following errors occur, PID_3Step can no longer move the actuator: • 2000h: Invalid value at Feedback_PER parameter. • 4000h: Invalid value at Feedback parameter. • 8000h: Error during digital position feedback. • 20000h: Invalid value at SavePosition tag. Value has an invalid number format. The characteristics are independent of ErrorBehaviour. As soon as the errors are no longer pending, PID_3Step switches back to automatic mode.

Pretuning, fine tuning, and transition time measurement

ActivateRecoverMode	Description
FALSE	In the event of an error, PID_3Step switches to "Inactive" or "Approach substitute output value" mode. The controller is only activated by a falling edge at Reset or a rising edge at ModeActivate. The controller changes to "Inactive" mode after successful transition time measurement.
TRUE	If the following error occurs, PID_3Step remains in the active mode: • 0020h: Pretuning is not permitted during fine tuning. The following errors are ignored: • 10000h: Invalid value at ManualValue parameter. • 20000h: Invalid value at SavePosition tag. When any other error occurs, PID_3Step cancels the tuning and switches to the mode from which tuning was started.

Manual mode

ActivateRecoverMode is not effective in manual mode.

See also

Static tags of PID_3Step V2 (Page 4268)

Parameters State and Mode V2 (Page 4276)

Tag Warning V2 (S7-1200, S7-1500)

If several warnings are pending simultaneously, their values are displayed with binary addition. The display of warning 0005h, for example, indicates that the warnings 0001h and 0004h are pending simultaneously.

Warning (DW#16#...)	Description
0000	No warning pending.
0001	The point of inflection was not found during pretuning.
0004	The setpoint was limited to the configured limits.
0008	Not all the necessary controlled system properties were defined for the selected method of calculation. Instead, the PID parameters were calculated using the TIR.TuneRule = 3 method.
0010	The operating mode could not be changed because Reset = TRUE or ManualEnable = TRUE.
0020	The cycle time of the calling OB limits the sampling time of the PID algorithm. Improve results by using shorter OB cycle times.
0040	The process value exceeded one of its warning limits.
0080	Invalid value at Mode. The operating mode is not switched.
0100	The manual value was limited to the limits of the controller output.
0200	The specified rule for tuning is not supported. No PID parameters are calculated.
0400	The transition time cannot be measured because the actuator settings do not match the selected measuring method.
0800	The difference between the current position and the new output value is too small for transition time measurement. This can produce incorrect results. The difference between the current output value and new output value must be at least 50% of the entire control range.
1000	The substitute output value cannot be reached because it is outside the output value limits.
2000	The actuator was moved in one direction for longer than Config.VirtualActuatorLimit × Retain.TransitTime. Check whether the actuator has reached an endstop signal.

The following warnings are deleted as soon as the cause is eliminated:

- 0001h
- 0004h
- 0008h
- 0040h
- 0100h
- 2000h

All other warnings are cleared with a rising edge at Reset or ErrorAck.

PID_3Step V1 (S7-1200)

Description PID_3Step V1 (S7-1200)

Description

You use the PID_3Step instruction to configure a PID controller with self tuning for valves or actuators with integrating behavior.

The following operating modes are possible:

- Inactive
- Pretuning
- Fine tuning
- Automatic mode
- Manual mode
- Approach substitute output value
- Transition time measurement
- Approach substitute output value with error monitoring
- Error monitoring

For a more detailed description of the operating modes, see the State parameter.

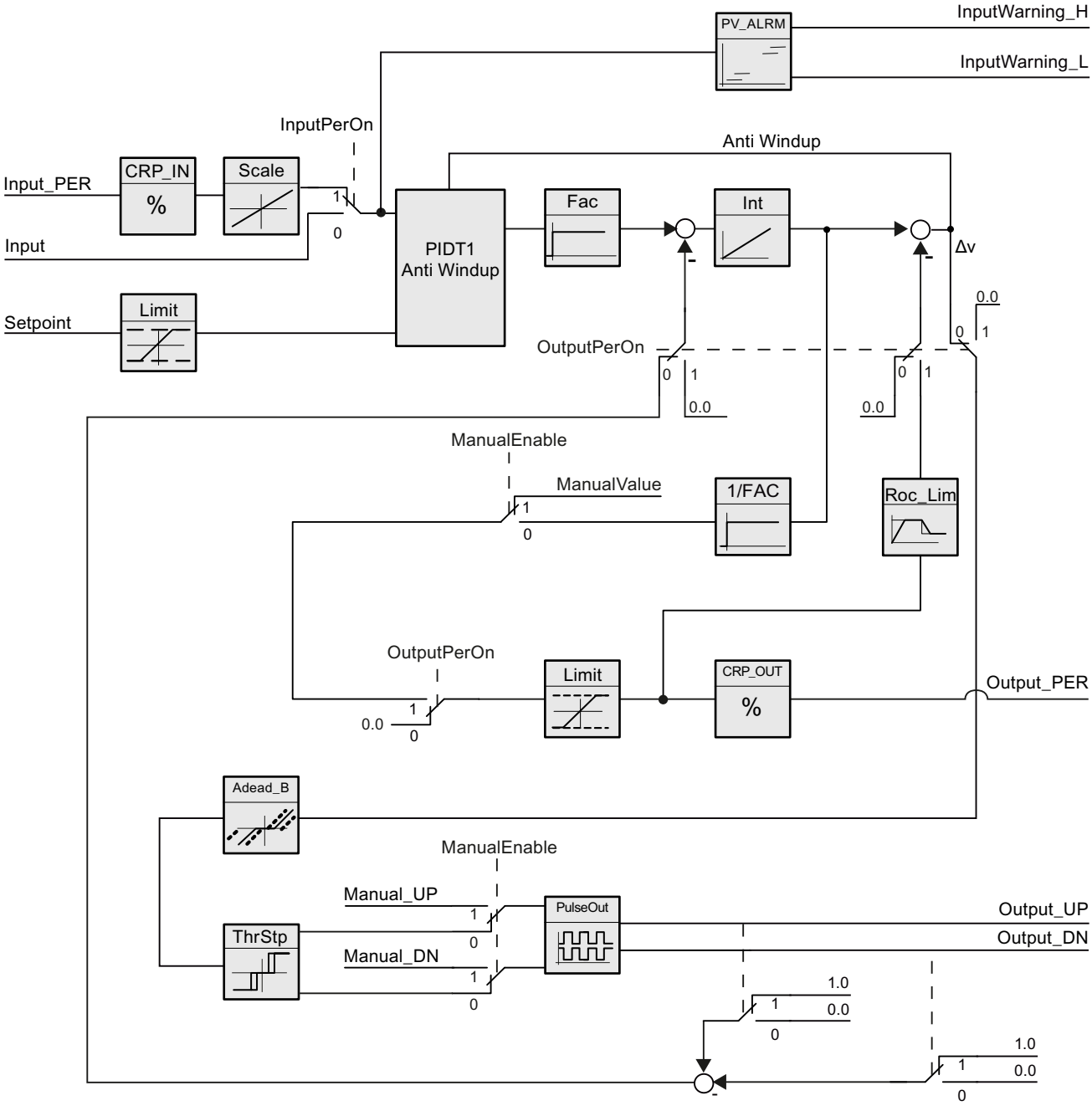
PID algorithm

PID_3Step is a PIDT1 controller with anti-windup and weighting of the proportional and derivative actions. The following equation is used to calculate the output value.

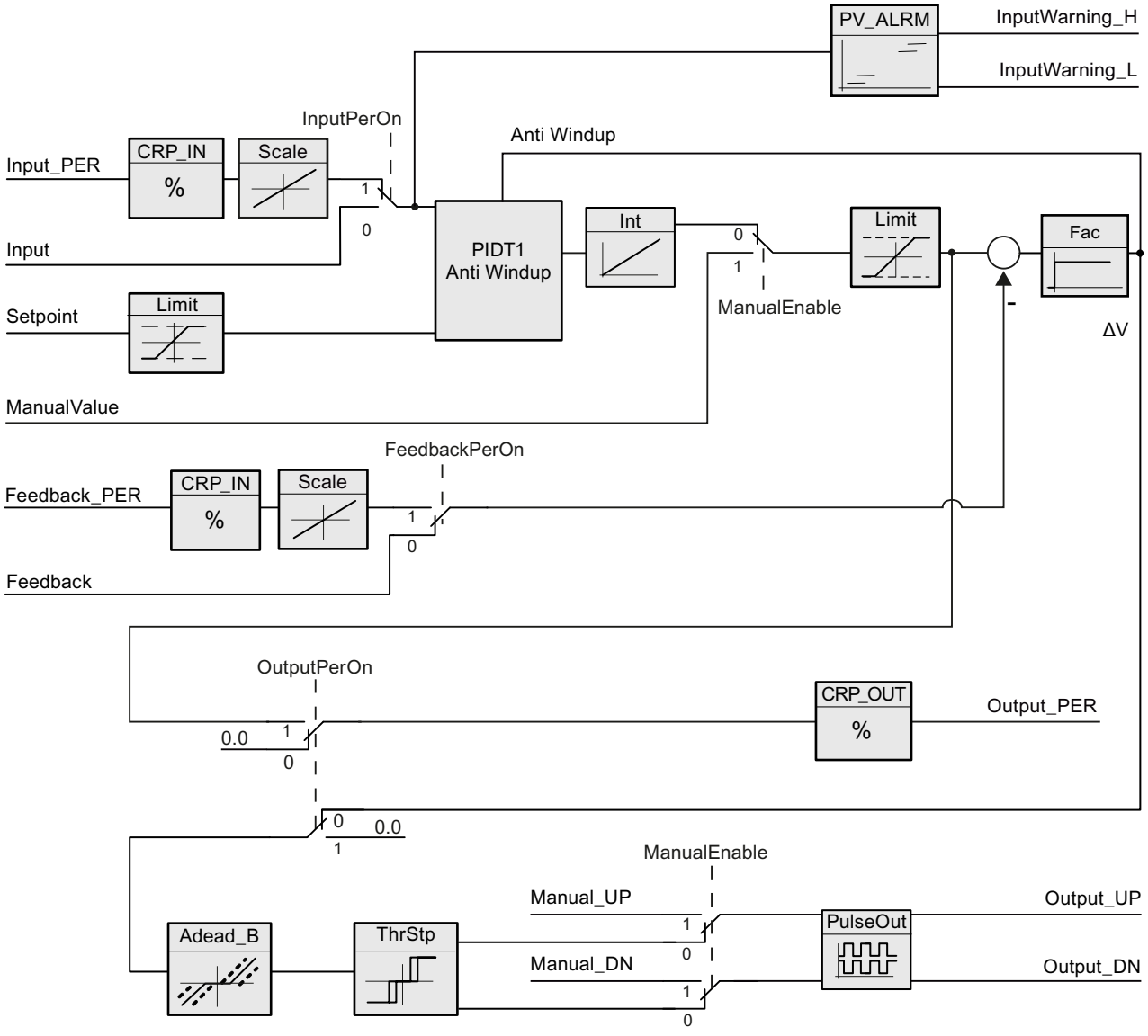
$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
y	Output value
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integral action time
a	Derivative delay coefficient (T1 = a × T_D)
T_D	Derivative action time
c	Derivative action weighting

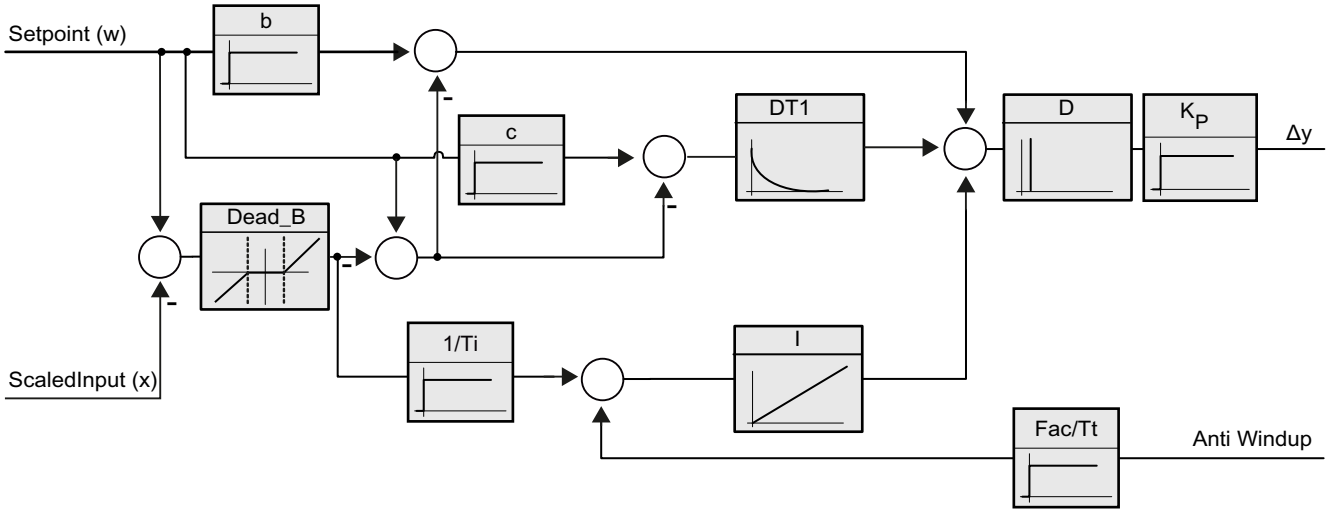
Block diagram without position feedback



Block diagram with position feedback



Block diagram of PIDT1 with anti-windup



Call

PID_3Step is called in a constant time interval of the cycle time of the calling OB (preferably in a cyclic interrupt OB).

Download to device

The actual values of retentive tags are only updated when you download PID_3Step completely.

Downloading technology objects to device (Page 6000)

Startup

At the startup of the CPU, PID_3Step starts in the operating mode that was last active. To leave PID_3Step in "Inactive" mode, set RunModeByStartup = FALSE.

Reaction to error

If errors occur, these are output in the Error parameter. You configure the reaction of PID_3Step using the ErrorBehaviour and ActivateRecoverMode tags.

ErrorBehaviour	ActivateRecoverMode	Actuator setting configuration Set Output to	Reaction
0	FALSE	Current output value	Switch to "Inactive" mode (Mode = 0)
0	TRUE	Current output value while error is pending	Switch to "Error monitoring" mode (Mode = 7)

ErrorBehaviour	ActivateRecoverMode	Actuator setting configuration Set Output to	Reaction
1	FALSE	Substitute output value	Switch to "Approach substitute output value" mode (Mode = 5) Switch to "Inactive" mode (Mode = 0)
1	TRUE	Substitute output value while error is pending	Switch to "Approach substitute output value with error monitoring" mode (Mode = 8) Switch to "Error monitoring" mode (Mode = 7)

The ErrorBits parameter shows which errors have occurred.

See also

Parameter State and Retain.Mode V1 (Page 4305)

Parameter ErrorBits V1 (Page 4312)

Configuring PID_3Step V1 (Page 6079)

Operating principle PID_3Step V1 (S7-1200)

Monitoring process value limits

You specify the high limit and low limit of the process value in the Config.InputUpperLimit and Config.InputLowerLimit tags. If the process value is outside these limits, an error occurs (ErrorBits = 0001hex).

You specify a high and low warning limit of the process value in the Config.InputUpperWarning and Config.InputLowerWarning tags. If the process value is outside these warning limits, a warning occurs (Warnings = 0040hex), and the InputWarning_H or InputWarning_L output parameter changes to TRUE.

Limiting the setpoint

You specify a high limit and low limit of the setpoint in the Config.SetpointUpperLimit and Config.SetpointLowerLimit tags. PID_3Step automatically limits the setpoint to the process value limits. You can limit the setpoint to a smaller range. PID_3Step checks whether this range falls within the process value limits. If the setpoint is outside these limits, the high or low limit is used as the setpoint, and output parameter SetpointLimit_H or SetpointLimit_L is set to TRUE.

The setpoint is limited in all operating modes.

Limiting the output value

You specify a high limit and low limit of the output value in the Config.OutputUpperLimit and Config.OutputLowerLimit tags. The output value limits must be within "Low endstop" and "High endstop".

- High endstop: Config.FeedbackScaling.UpperPointOut
- Low endstop: Config.FeedbackScaling.LowerPointOut

Rule:

UpperPointOut ≥ OutputUpperLimit > OutputLowerLimit ≥ LowerPointOut

The valid values for "High endstop" and "Low endstop" depend upon:

- FeedbackOn
- FeedbackPerOn
- OutputPerOn

OutputPerOn	FeedbackOn	FeedbackPerOn	LowerPointOut	UpperPointOut
FALSE	FALSE	FALSE	Cannot be set (0.0%)	Cannot be set (100.0%)
FALSE	TRUE	FALSE	-100.0% or 0.0%	0.0% or +100.0%
FALSE	TRUE	TRUE	-100.0% or 0.0%	0.0% or +100.0%
TRUE	FALSE	FALSE	Cannot be set (100.0%)	Cannot be set (100.0%)
TRUE	TRUE	FALSE	-100.0% or 0.0%	0.0% or +100.0%
TRUE	TRUE	TRUE	-100.0% or 0.0%	0.0% or +100.0%

If OutputPerOn = FALSE and FeedbackOn = FALSE, you cannot limit the output value. The digital outputs are reset with Actuator_H = TRUE or Actuator_L = TRUE, or after a travel time amounting to 110% of the motor transition time.

The output value is 27648 at 100% and -27648 at -100%. PID_3Step must be able to completely close the valve. Therefore, zero must be included in the output value limits.

Note

Use with two or more actuators

PID_3 Step is not suitable for use with two or more actuators (for example, in heating/cooling applications), because different actuators need different PID parameters to achieve a good control response.

Substitute output value

If an error has occurred, PID_3Step can output a substitute output value and move the actuator to a safe position that is specified in the SavePosition tag. The substitute output value must be within the output value limits.

Monitoring signal validity

The values of the following parameters are monitored for validity:

- Setpoint
- Input
- Input_PER
- Feedback
- Feedback_PER
- Output

Monitoring the PID_3Step sampling time

Ideally, the sampling time is equivalent to the cycle time of the calling OB. The PID_3Step instruction measures the time interval between two calls. This is the current sampling time. On every switchover of operating mode and during the initial startup, the mean value is formed from the first 10 sampling times. Too great a difference between the current sampling time and this mean value triggers an error (ErrorBits = 0800 hex).

PID_3Step is set to "Inactive" mode during tuning under the following conditions:

- New mean value >= 1.1 x old mean value
- New mean value <= 0.9 x old mean value

In automatic mode, PID_3Step is set to "Inactive" mode under the following conditions:

- New mean value >= 1.5 x old mean value
- New mean value <= 0.5 x old mean value

Sampling time of the PID algorithm

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the cycle time. All other functions of PID_3Step are executed at every call.

Measuring the motor transition time

The motor transition time is the time in seconds the motor requires to move the actuator from the closed to the opened state. The maximum time that the actuator is moved in one direction is 110% of the motor transition time. PID_3Step requires the motor transition time to be as accurate as possible for good controller results. The data in the actuator documentation contains average values for this type of actuator. The value for the specific actuator used may differ. You can measure the motor transition time during commissioning. The output value limits are not taken into consideration during the motor transition time measurement. The actuator can travel to the high or the low endstop.

The motor transition time is taken into consideration in the calculation of the analog output value as well as in the calculation of the digital output values. It is mainly required for correct operation during auto-tuning and the anti-windup behavior. You should therefore configure the

motor transition time with the value that the motor requires to move the actuator from the closed to the opened state.

If no relevant motor transition time is in effect in your process (e.g. with solenoid valves), so that the output value has a direct and full effect on the process, use PID_Compact instead.

Control logic

An increase of the output value is generally intended to cause an increase in the process value. This is referred to as a normal control logic. For cooling and discharge control systems, it may be necessary to invert the control logic. PID_3Step does not work with negative proportional gain. If InvertControl = TRUE, an increasing control deviation causes a reduction in the output value. The control logic is also taken into account during pretuning and fine tuning.

See also

Configuring PID_3Step V1 (Page 6079)

PID_3Step V1 input parameters (S7-1200)

Table 11-76

Parameters	Data type	Default	Description
Setpoint	REAL	0.0	Setpoint of the PID controller in automatic mode
Input	REAL	0.0	A tag of the user program is used as source for the process value. If you are using parameter Input, then Config.InputPer-On = FALSE must be set.
Input_PER	WORD	W#16#0	An analog input is used as the source of the process value. If you are using parameter Input_PER, then Config.InputPer-On = TRUE must be set.
Actuator_H	BOOL	FALSE	Digital position feedback of the valve for the high endstop If Actuator_H = TRUE, the valve is at the high endstop and is no longer moved towards this direction.
Actuator_L	BOOL	FALSE	Digital position feedback of the valve for the low endstop If Actuator_L = TRUE, the valve is at the low endstop and is no longer moved towards this direction.
Feedback	REAL	0.0	Position feedback of the valve If you are using parameter Feedback, then Config.Feedback-PerOn = FALSE must be set.
Feedback_PER	WORD	W#16#0	Analog position feedback of a valve If you are using parameter Feedback_PER, then Config.Feed-backPerOn = TRUE must be set. Feedback_PER is scaled based on the tags: • Config.FeedbackScaling.LowerPointIn • Config.FeedbackScaling.UpperPointIn • Config.FeedbackScaling.LowerPointOut • Config.FeedbackScaling.UpperPointOut

Parameters	Data type	Default	Description
ManualEnable	BOOL	FALSE	• A FALSE -> TRUE edge selects "Manual mode", while State = 4, Retain.Mode remains unchanged. • A TRUE -> FALSE edge selects the most recently active operating mode A change of Retain.Mode will not take effect during ManualEnable = TRUE. The change of Retain.Mode will only be considered upon a TRUE -> FALSE edge at ManualEnable . **PID_3Step V1.1**If ManualEnable = TRUE when the CPU starts, PID_3Step starts in manual mode. A rising edge (FALSE > TRUE) at ManualEnable is not necessary. **PID_3Step V1.0** At the start of the CPU, PID_3Step only switches to manual mode with a rising edge (FALSE->TRUE) at ManualEnable . Without rising edge, PID_3Step starts in the last operating mode in which ManualEnable was FALSE.
ManualValue	REAL	0.0	In manual mode, you specify the absolute position of the valve. ManualValue will only be evaluated if you are using OutputPer, **or** if position feedback is available.
Manual_UP	BOOL	FALSE	In manual mode, every rising edge opens the valve by 5% of the total control range, or for the duration of the minimum motor transition time. Manual_UP is evaluated only if you are not using Output_PER and there is no position feedback available.
Manual_DN	BOOL	FALSE	In manual mode, every rising edge closes the valve by 5% of the total control range, or for the duration of the minimum motor transition time. Manual_DN is evaluated only if you are not using Output_PER and there is no position feedback available.
Reset	BOOL	FALSE	Restarts the controller. • FALSE -> TRUE edge – Switch to "Inactive" mode – ErrorBits and Warning are reset – Intermediate controller values are reset (PID parameters are retained) • TRUE -> FALSE edge – Change in most recent active mode – If automatic mode was active before, switchover to automatic mode is bumpless.

PID_3Step V1 output parameters (S7-1200)

Table 11-77

Parameter	Data type	Default	Description
ScaledInput	REAL	0.0	Scaled process value
ScaledFeedback	REAL	0.0	Scaled position feedback For an actuator without position feedback, the position of the actuator indicated by ScaledFeedback is very imprecise. ScaledFeedback may only be used for rough estimation of the current position in this case.
Output_UP	BOOL	FALSE	Digital output value for opening the valve If Config.OutputPerOn = FALSE, the Output_UP parameter is used.
Output_DN	BOOL	FALSE	Digital output value for closing the valve If Config.OutputPerOn = FALSE, the Output_DN parameter is used.
Output_PER	WORD	W#16#0	Analog output value If Config.OutputPerOn = TRUE, Output_PER is used. Use Output_PER if you are using a valve as actuator which is triggered via an analog output and controlled with a continuous signal, e.g. 0...10 V or 4...20 mA. The value at Output_PER corresponds to the target position of the valve, e.g. Output_PER = 13824, when the valve is to be opened by 50%.
SetpointLimit_H	BOOL	FALSE	If SetpointLimit_H = TRUE, the absolute setpoint high limit is reached. In the CPU, the setpoint is limited to the configured absolute setpoint high limit. The configured absolute process value high limit is the default for the setpoint high limit. If you configure Config.SetpointUpperLimit to a value within the process value limits, this value is used as the setpoint high limit.
SetpointLimit_L	BOOL	FALSE	If SetpointLimit_L = TRUE, the absolute setpoint low limit has been reached. In the CPU, the setpoint is limited to the configured absolute setpoint low limit. The configured absolute process value low limit is the default setting for the setpoint low limit. If you configure Config.SetpointLowerLimit to a value within the process value limits, this value is used as the setpoint low limit.
InputWarning_H	BOOL	FALSE	If InputWarning_H = TRUE, the process value has reached or exceeded the warning high limit.
InputWarning_L	BOOL	FALSE	If InputWarning_L = TRUE, the process value has reached or fallen below the warning low limit.

Parameter	Data type	Default	Description
State	INT	0	The State parameter (Page 4305) shows the current operating mode of the PID controller. You change the operating mode with the Retain.Mode tag. • State = 0: Inactive • State = 1: Pretuning • State = 2: Fine tuning • State = 3: Automatic mode • State = 4: Manual mode • State = 5: Approach substitute output value • State = 6: Transition time measurement • State = 7: Error monitoring • State = 8: Approach substitute output value with error monitoring
Error	BOOL	FALSE	If Error = TRUE, at least one error message is pending.
ErrorBits	DWORD	DW#16#0	The ErrorBits parameter (Page 4312) indicates the error messages.

See also

Parameter State and Retain.Mode V1 (Page 4305)

Parameter ErrorBits V1 (Page 4312)

PID_3Step V1 static tags (S7-1200)

You must not change tags that are not listed. These are used for internal purposes only.

Table 11-78

Tag	Data type	Default	Description
ActivateRecoverMode	BOOL	TRUE	The ActivateRecoverMode tag (Page 4314) determines the reaction to error.
RunModeByStartup	BOOL	TRUE	Activate Mode after CPU restart If RunModeByStartup = TRUE, the controller returns to the last active operating mode after a CPU restart. If RunModeByStartup = FALSE, the controller remains inactive after a CPU restart.
PhysicalUnit	INT	0	Unit of measurement of the process value and setpoint, e.g., °C, or °F.
PhysicalQuantity	INT	0	Physical quantity of the process value and setpoint, e.g., temperature.

Tag	Data type	Default	Description
ErrorBehaviour	INT	0	If ErrorBehaviour = 0 and an error has occurred, the valve stays at its current position and the controller switches directly to "Inactive" or "Error monitoring" mode. If ErrorBehaviour = 1 and an error occurs, the actuator moves to the substitute output value and only then switches to "Inactive" or "Error monitoring" mode. If the following errors occur, you can no longer move the valve to a configured substitute output value. • 2000h: Invalid value at Feedback_PER parameter. • 4000h: Invalid value at Feedback parameter. • 8000h: Error during digital position feedback.
Warning	DWORD	DW#16#0	The Warning tag (Page 4305) displays the warnings generated since a Reset or since the last operating mode switchover. Cyclic warnings (for example, process value warning) are shown until the cause of the warning is removed. They are automatically deleted once their cause has gone. Non-cyclic warnings (for example, point of inflection not found) remain and are deleted like errors.
SavePosition	REAL	0.0	Substitute output value If ErrorBehaviour = 1 and an error occurs, the actuator moves to a safe position for the plant and only then switches to "Inactive" mode.
CurrentSetpoint	REAL	0.0	Currently active setpoint. This value is frozen at the start of tuning.
Progress	REAL	0.0	Progress of tuning as a percentage (0.0 - 100.0)
Config.InputPerOn	BOOL	TRUE	If InputPerOn = TRUE, the Input_PER parameter is used. If InputPerOn = FALSE, the Input parameter is used.
Config.OutputPerOn	BOOL	FALSE	If OutputPerOn = TRUE, the Output_PER parameter is used. If OutputPerOn = FALSE, the Ouput_UP and Output_DN parameters are used.
Config.LoadBackUp	BOOL	FALSE	If LoadBackUp = TRUE, the last set of PID parameters is reloaded. The set was saved prior to the last tuning. LoadBackUp is automatically set back to FALSE.
Config.InvertControl	BOOL	FALSE	Invert control logic If InvertControl = TRUE, an increasing control deviation causes a reduction in the output value.
Config.FeedbackOn	BOOL	FALSE	If FeedbackOn = FALSE, a position feedback is simulated. Position feedback is generally activated when FeedbackOn = TRUE.
Config.FeedbackPerOn	BOOL	FALSE	FeedbackPerOn is only effective when FeedbackOn = TRUE. If FeedbackPerOn = TRUE, the analog input is used for the position feedback (Feedback_PER parameter). If FeedbackPerOn = FALSE, the Feedback parameter is used for the position feedback.
Config.ActuatorEndStopOn	BOOL	FALSE	If ActuatorEndStopOn = TRUE, the digital position feedback Actuator_L and Actuator_H are taken into consideration.

Tag	Data type	Default	Description
Config.InputUpperLimit	REAL	120.0	High limit of the process value At the I/O input, the process value can be a maximum of 18% higher than the standard range (overrange). An error is no longer signaled due to a violation of the "Process value high limit". Only a wire-break and a short-circuit are recognized and PID_3Step reacts according to the configured reaction to error. InputUpperLimit > InputLowerLimit
Config.InputLowerLimit	REAL	0.0	Low limit of the process value InputLowerLimit < InputUpperLimit
Config.InputUpperWarning	REAL	+3.402822e+38	Warning high limit of the process value If you set InputUpperWarning outside the process value limits, the configured absolute process value high limit is used as the warning high limit. If you configure InputUpperWarning within the process value limits, this value is used as the warning high limit. InputUpperWarning > InputLowerWarning InputUpperWarning ≤ InputUpperLimit
Config.InputLowerWarning	REAL	-3.402822e+38	Warning low limit of the process value If you set InputLowerWarning outside the process value limits, the configured absolute process value low limit is used as the warning low limit. If you configure InputLowerWarning within the process value limits, this value is used as the warning low limit. InputLowerWarning < InputUpperWarning InputLowerWarning ≥ InputLowerLimit
Config.OutputUpperLimit	REAL	100.0	High limit of output value For details, see OutputLowerLimit
Config.OutputLowerLimit	REAL	0.0	Low limit of output value If OutputPerOn = TRUE or FeedbackOn = TRUE, the range of values from -100% to +100%, including zero, is valid. At -100%, Output = -27648; at +100%, Output = 27648 If OutputPerOn = FALSE, the range of values from 0% to 100% is valid. The valve is completely closed at 0% and completely opened at 100%.
Config.SetpointUpperLimit	REAL	+3.402822e+38	High limit of setpoint If you set SetpointUpperLimit outside the process value limits, the configured absolute process value high limit is preassigned as the setpoint high limit. If you configure SetpointUpperLimit within the process value limits, this value is used as the setpoint high limit.
Config.SetpointLowerLimit	REAL	-3.402822e+38	Low limit of the setpoint If you set SetpointLowerLimit outside the process value limits, the configured absolute process value low limit is preassigned as the setpoint low limit. If you set SetpointLowerLimit within the process value limits, this value is used as the setpoint low limit.

Tag	Data type	Default	Description
Config.MinimumOnTime	REAL	0.0	Minimum ON time Minimum time in seconds for which the servo drive must be switched on. Config.MinimumOnTime is only effective if Output_UP and Output_DN are used (Config.OutputPerOn = FALSE).
Config.MinimumOffTime	REAL	0.0	Minimum OFF time Minimum time in seconds for which the servo drive must be switched off. Config.MinimumOffTime is only effective if Output_UP and Output_DN are used (Config.OutputPerOn = FALSE).
Config.TransitTime	REAL	30.0	Motor transition time Time in seconds the actuating drive requires to move the valve from the closed to the opened state.
Config.InputScaling.UpperPointIn	REAL	27648.0	Scaling Input_PER high Input_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the InputScaling structure.
Config.InputScaling.LowerPointIn	REAL	0.0	Scaling Input_PER low Input_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the InputScaling structure.
Config.InputScaling.UpperPointOut	REAL	100.0	Scaled high process value Input_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the InputScaling structure.
Config.InputScaling.LowerPointOut	REAL	0.0	Scaled low process value Input_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the InputScaling structure.
Config.FeedbackScaling.UpperPointIn	REAL	27648.0	Scaling Feedback_PER high Feedback_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the FeedbackScaling structure.
Config.FeedbackScaling.LowerPointIn	REAL	0.0	Scaling Feedback_PER low Feedback_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the FeedbackScaling structure.
Config.FeedbackScaling.UpperPointOut	REAL	100.0	High endstop Feedback_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the FeedbackScaling structure.
Config.FeedbackScaling.LowerPointOut	REAL	0.0	Low endstop Feedback_PER is converted to a percentage based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn of the FeedbackScaling structure.

Tag	Data type	Default	Description
GetTransitTime.InvertDirection	BOOL	FALSE	If InvertDirection = FALSE, the valve is fully opened, closed, and then reopened in order to determine the valve transition time. If InvertDirection = TRUE, the valve is fully closed, opened, and then closed again.
GetTransitTime.SelectFeedback	BOOL	FALSE	If SelectFeedback = TRUE, then Feedback_PER, or Feedback is taken into consideration in the transition time measurement. If SelectFeedback = FALSE, then Actuator_H and Actuator_L are taken into consideration in the transition time measurement.
GetTransitTime.Start	BOOL	FALSE	If Start = TRUE, the transition time measurement is started.
GetTransitTime.State	INT	0	Current phase of the transition time measurement • State = 0: Inactive • State = 1: Open valve completely • State = 2: Close valve completely • State = 3: Move valve to target position (NewOutput) • State = 4: Transition time measurement successfully completed • State = 5: Transition time measurement canceled
GetTransitTime.NewOutput	REAL	0.0	Target position for transition time measurement with position feedback The target position must be between "High endstop" and "Low endstop". The difference between NewOutput and ScaledFeedback must be at least 50% of the permissible control range.
CycleTime.StartEstimation	BOOL	TRUE	If StartEstimation = TRUE, the measurement of the PID_3Step sampling time is started. CycleTime.StartEstimation = FALSE once measurement is complete.
CycleTime.EnEstimation	BOOL	TRUE	If EnEstimation = TRUE, the PID_3Step sampling time is calculated.
CycleTime.EnMonitoring	BOOL	TRUE	If EnMonitoring = TRUE, the PID_3Step sampling time is monitored. If it is not possible to execute PID_3Step within the sampling time, the error 0800h is output and the operating mode is switched. ActivateRecoverMode and ErrorBehaviour determine which operating mode is switched to. If EnMonitoring = FALSE, the PID_3Step sampling time is not monitored, the error 0800h is not output, and the operating mode is not switched.
CycleTime.Value	REAL	0.1	PID_3Step sampling time in seconds CycleTime.Value is determined automatically and is usually equivalent to the cycle time of the calling OB.
CtrlParamsBackUp.SetByUser	BOOL	FALSE	Saved value of Retain.CtrlParams.SetByUser. You can reload values from the CtrlParamsBackUp structure with Config.LoadBackUp = TRUE.
CtrlParamsBackUp.Gain	REAL	1.0	Saved proportional gain
CtrlParamsBackUp.Ti	REAL	20.0	Saved integral action time
CtrlParamsBackUp.Td	REAL	0.0	Saved derivative action time
CtrlParamsBackUp.TdFiltRatio	REAL	0.0	Saved derivative delay coefficient
CtrlParamsBackUp.PWeighting	REAL	0.0	Saved proportional action weighting

Tag	Data type	Default	Description
CtrlParamsBackUp.DWeighting	REAL	0.0	Saved derivative action weighting
CtrlParamsBackUp.Cycle	REAL	1.0	Saved sampling time of PID algorithm
CtrlParamsBackUp.InputDeadBand	REAL	0.0	Saved dead zone width of the control deviation
PIDSelfTune.SUT.CalculateSUTParams	BOOL	FALSE	The properties of the controlled system are saved during tuning. If CalculateSUTParams = TRUE, the PID parameters are recalculated on the basis of these properties. The PID parameters are calculated using the method set in TuneRuleSUT. CalculateSUTParams is set to FALSE following calculation.
PIDSelfTune.SUT.TuneRuleSUT	INT	1	Methods used to calculate parameters during pretuning: • TuneRuleSUT = 0: PID fast I • TuneRuleSUT = 1: PID slow I • TuneRuleSUT = 2: Chien, Hrones and Reswick PID • TuneRuleSUT = 3: Chien, Hrones, Reswick PI • TuneRuleSUT = 4: PID fast II • TuneRuleSUT = 5: PID slow II
PIDSelfTune.SUT.State	INT	0	The SUT.State tag indicates the current phase of pretuning:
PIDSelfTune.TIR.RunIn	BOOL	FALSE	• RunIn = FALSE Pretuning is started when fine tuning is started from inactive or manual mode. If fine tuning is started from automatic mode, the system uses the existing PID parameters to control to the setpoint. Only then will fine tuning start. If pretuning is not possible, PID_3Step switches to "Inactive" mode. • RunIn = TRUE The pretuning is skipped. PID_3Step attempts to reach the setpoint with the minimum or maximum output value. This can produce increased overshoot. Only then will fine tuning start. RunIn is set to FALSE after fine tuning.
PIDSelfTune.TIR.CalculateTIRParams	BOOL	FALSE	The properties of the controlled system are saved during tuning. If CalculateTIRParams = TRUE, the PID parameters are recalculated on the basis of these properties. The PID parameters are calculated using the method set in TuneRuleTIR. CalculateTIRParams is set to FALSE following calculation.
PIDSelfTune.TIR.TuneRuleTIR	INT	0	Methods used to calculate parameters during fine tuning: • TuneRuleTIR = 0: PID automatic • TuneRuleTIR = 1: PID fast • TuneRuleTIR = 2: PID slow • TuneRuleTIR = 3: Ziegler-Nichols PID • TuneRuleTIR = 4: Ziegler-Nichols PI • TuneRuleTIR = 5: Ziegler-Nichols P
PIDSelfTune.TIR.State	INT	0	The TIR.State tag indicates the current phase of "fine tuning":

Tag	Data type	Default	Description
Retain.Mode	INT	0	A change to the value of Retain.Mode initiates a switch to another operating mode. The following operating mode is enabled upon a change of Mode to: • Mode = 0: Inactive • Mode = 1: Pretuning • Mode = 2: Fine tuning • Mode = 3: Automatic mode • Mode = 4: Manual mode • Mode = 5: Approach substitute output value • Mode = 6: Transition time measurement • Mode = 7: Error monitoring • Mode = 8: Approach substitute output value with error monitoring Mode is retentive.
Retain.CtrlParams.SetByUser	BOOL	FALSE	If SetByUser = FALSE, the PID parameters are determined automatically and PID_3Step operates with a dead zone at the output value. The dead zone width is calculated during tuning on the basis of the standard deviation of the output value and saved in Retain.CtrlParams.OutputDeadBand. If SetByUser = TRUE, the PID parameters are entered manually and PID_3 Step operates without a dead zone at the output value. Retain.CtrlParams.OutputDeadBand = 0.0 SetByUser is retentive.
Retain.CtrlParams.Gain	REAL	1.0	Active proportional gain Gain is retentive.
Retain.CtrlParams.Ti	REAL	20.0	• Ti > 0.0: Active integral action time • Ti = 0.0: Integral action is deactivated Ti is retentive.
Retain.CtrlParams.Td	REAL	0.0	• Td > 0.0: Active derivative action time • Td = 0.0: Derivative action is deactivated Td is retentive.
Retain.CtrlParams.TdFiltRatio	REAL	0.0	Active derivative delay coefficient TdFiltRatio is retentive.
Retain.CtrlParams.PWeighting	REAL	0.0	Active proportional action weighting PWeighting is retentive.
Retain.CtrlParams.DWeighting	REAL	0.0	Active derivative action weighting DWeighting is retentive.
Retain.CtrlParams.Cycle	REAL	1.0	Active sampling time of PID algorithm in seconds, rounded to an integer multiple of the cycle time of the calling OB. Cycle is retentive.
Retain.CtrlParams.InputDeadBand	REAL	0.0	Dead zone width of the control deviation InputDeadBand is retentive.

Note

Change the tags listed in this table in "Inactive" mode to prevent malfunction of the PID controller. "Inactive" mode is forced by setting the "Retain.Mode" tag to "0".

See also

Parameter State and Retain.Mode V1 (Page 4305)

Tag ActivateRecoverMode V1 (Page 4314)

Downloading technology objects to device (Page 6000)

Parameter State and Retain.Mode V1 (S7-1200)

Correlation of the parameters

The State parameter shows the current operating mode of the PID controller. You cannot change the State parameter.

To switch from one operating mode to another, you must change the Retain.Mode tag. This also applies when the value for the new operating mode is already in Retain.Mode. For example, set Retain.Mode = 0 first and then Retain.Mode = 3. Provided the current operating mode of the controller permits this switchover, State will be set to the value of Retain.Mode.

When PID_3Step automatically switches from one operating mode to another, the following applies: State != Retain.Mode.

Examples:

- After successful pretuning
  State = 3 and Retain.Mode = 1
- In the event of an error
  State = 0 and Retain.Mode remain at the previous value, for example, Retain.Mode = 3
- ManualEnalbe = TRUE
  State = 4 and Retain.Mode remain at the previous value, e.g., Retain.Mode = 3

Note

You want, for example, to repeat successful fine tuning without exiting automatic mode with Mode = 0.

Setting Retain.Mode to an invalid value such as 9999 for one cycle has no effect on State. Set Mode = 2 in the next cycle. In this way, you can generate a change to Retain.Mode without first switching to "Inactive" mode.

Meaning of values

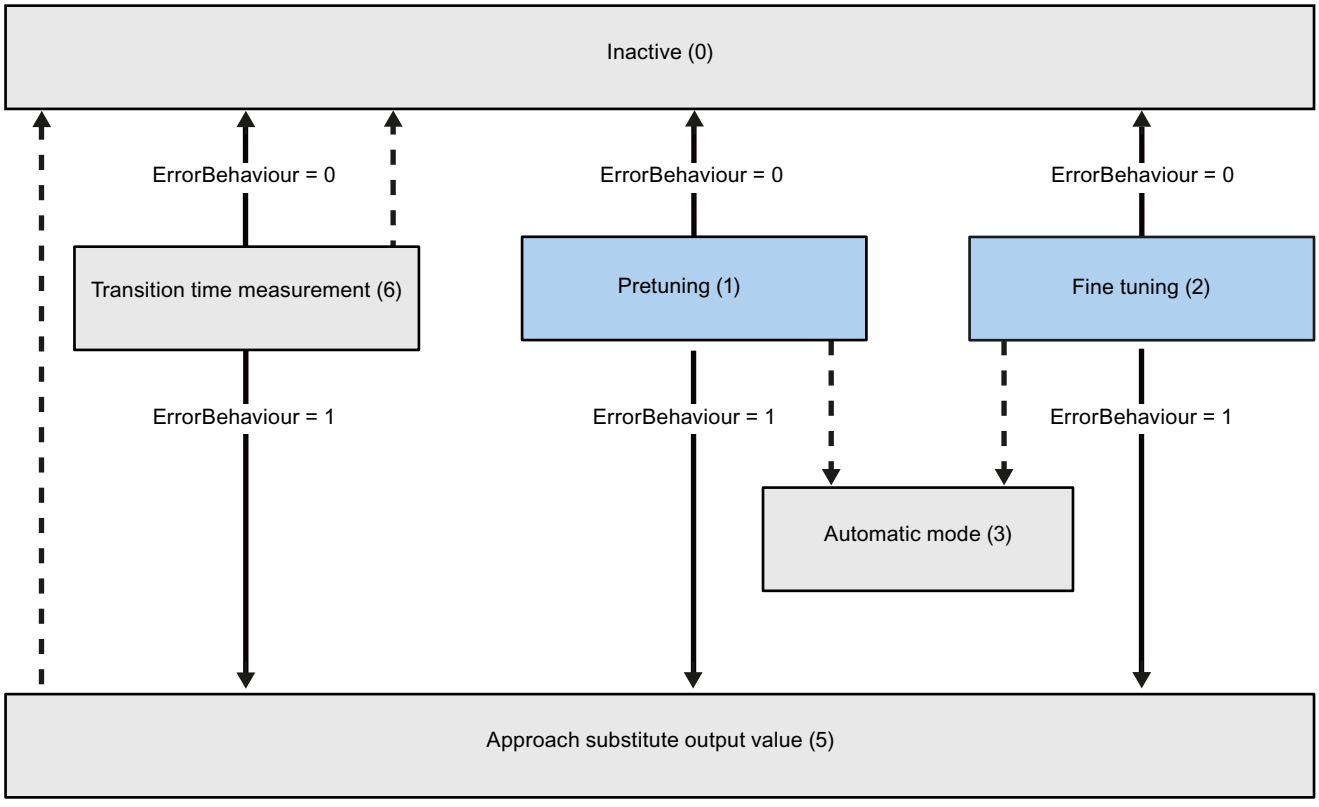
State / Retain.Mode	Description
0	Inactive The controller is switched off and no longer changes the valve position.
1	Pretuning The pretuning determines the process response to a pulse of the output value and searches for the point of inflection. The optimized PID parameters are calculated as a function of the maximum rate of rise and dead time of the controlled system. Pretuning requirements: • State = 0 or State = 4 • ManualEnable = FALSE • The motor transition time has been configured or measured. • The setpoint and the process value lie within the configured limits. The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher as compared to the noise. Before the PID parameters are recalculated, they are backed up and can be reactivated with Config.LoadBackUp. The setpoint is frozen in the CurrentSetpoint tag. The controller switches to automatic mode following successful pretuning and to "Inactive" mode following unsuccessful pretuning. The pretuning phase is indicated with the SUT.State tag.

State / Retain.Mode	Description
2	Fine tuning Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are tuned based on the amplitude and frequency of this oscillation. The differences between the process response during pretuning and fine tuning are analyzed. All PID parameters are recalculated from the results. PID parameters from fine tuning usually have better master control and disturbance characteristics than PID parameters from pretuning. PID_3Step automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. The PID parameters are backed up before fine tuning. They can be reactivated with Config.LoadBackUp. The setpoint is frozen in the CurrentSetpoint tag. Requirements for fine tuning: • The motor transition time has been configured or measured. • The setpoint and the process value lie within the configured limits. • ManualEnable = FALSE • Automatic (State = 3), inactive (State = 0) or manual (State = 4) mode Fine tuning proceeds as follows when started from: • Automatic mode (State = 3) Start fine tuning from automatic mode if you wish to improve the existing PID parameters through tuning. PID_3Step controls the system using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start. • Inactive (State = 0) or manual mode (State = 4) Pretuning is always started first. The determined PID parameters will be used for control until the control loop has stabilized and the requirements for fine tuning have been met. If PIDSelfTune.TIR.RunIn = TRUE, pretuning is skipped and an attempt is made to reach the setpoint with the minimum or maximum output value. This can produce increased overshoot. Fine tuning then starts automatically. The controller switches to automatic mode following successful fine tuning. If fine tuning was not successful, the controller switches to "Inactive" mode. The fine tuning phase is indicated with the TIR.State tag.
3	Automatic mode In automatic mode, PID_3Step controls the controlled system in accordance with the parameters specified. The controller switches to automatic mode if one the following requirements is fulfilled: • Pretuning successfully completed • Fine tuning successfully completed • Changing the Retain.Mode tag to the value 3. When the CPU is switched on or switches from Stop to RUN mode, PID_3Step starts in the most recently active operating mode. To leave PID_3Step in "Inactive" mode, set RunModeByStartup = FALSE. The ActivateRecoverMode tag is taken into consideration in automatic mode.

State / Retain.Mode	Description
4	Manual mode In manual mode, you specify manual output values in the Manual_UP and Manual_DN parameters or ManualValue parameter. Whether or not the actuator can be moved to the output value in the event of an error is described in the ErrorBits parameter. This operating mode is enabled if Retain.Mode = 4, or on a rising edge at ManualEnable. If ManualEnable changes to TRUE, only State changes. Retain.Mode retains its current value. On a falling edge at ManualEnable, PID_3Step returns to the previous operating mode. The switchover to automatic mode is bumpless. **PID_3Step V1.1** Manual mode is always possible in the event of an error. **PID_3Step V1.0** Manual mode is dependent on the ActivateRecoverMode tag in the event of an error.
5	Approach substitute output value This operating mode is activated in the event of an error or when Reset = TRUE if Errorbehaviour = 1 and ActivateRecoverMode = FALSE.. PID_3Step moves the actuator to the substitute output value and then switches to "Inactive" mode.
6	Transition time measurement The time that the motor needs to completely open the valve from the closed condition is determined. This operating mode is activated when GetTransitTime.Start = TRUE is set. If endstop signals are used to measure the transition time, the valve will be opened completely from its current position, closed completely, and opened completely again. If GetTransitTime.InvertDirection = TRUE, this behavior is inverted. If position feedback is used to measure the transition time, the actuator will be moved from its current position to a target position. The output value limits are not taken into consideration during the transition time measurement. The actuator can travel to the high or the low endstop.
7	Error monitoring The control algorithm is switched off and no longer changes the valve position. This operating mode is activated instead of "Inactive" mode in the event of an error. All the following conditions must be met: • Mode = 3 (automatic mode) • Errorbehaviour = 0 • ActivateRecoverMode = TRUE • One or more errors have occurred in which ActivateRecoverMode (Page 4314) is effective. As soon as the errors are no longer pending, PID_3Step switches back to automatic mode.
8	Approach substitute output value with error monitoring This operating mode is activated instead of "Approach substitute output value" mode in the event of an error. PID_3Step moves the actuator to the substitute output value and then switches to "Error monitoring" mode. All the following conditions must be met: • Mode = 3 (automatic mode) • Errorbehaviour = 1 • ActivateRecoverMode = TRUE • One or more errors have occurred in which ActivateRecoverMode (Page 4314) is effective. As soon as the errors are no longer pending, PID_3Step switches back to automatic mode.

Automatic switchover of operating mode during commissioning

PID_3Step automatically switches the operating mode in the event of an error. The following diagram illustrates the influence of ErrorBehaviour on the switchover of operating mode from transition time measurement, pretuning, and fine tuning modes.

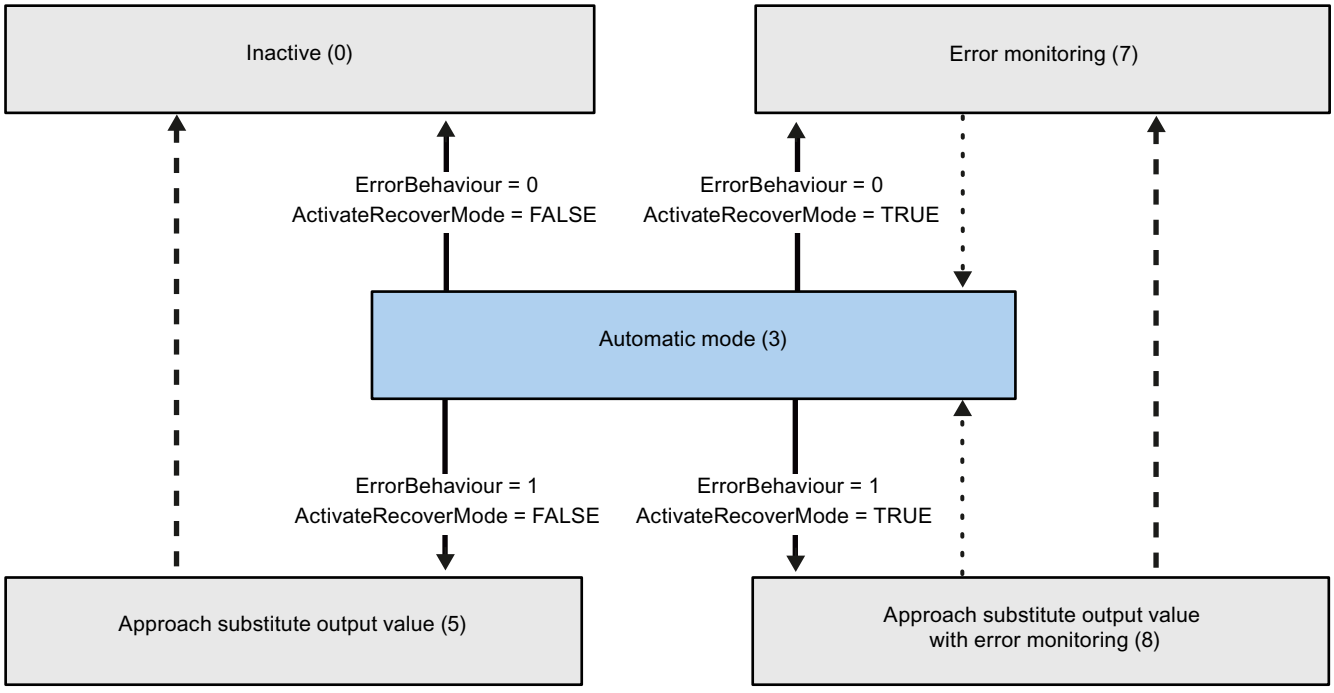


Automatic switchover of operating mode in the event of an error

Automatic switchover of operating mode once the current operation has been completed.

Automatic switchover of operating mode in automatic mode (PID_3Step V1.1)

PID_3Step automatically switches the operating mode in the event of an error. The following diagram illustrates the influence of ErrorBehaviour and ActivateRecoverMode on this switchover of operating mode.

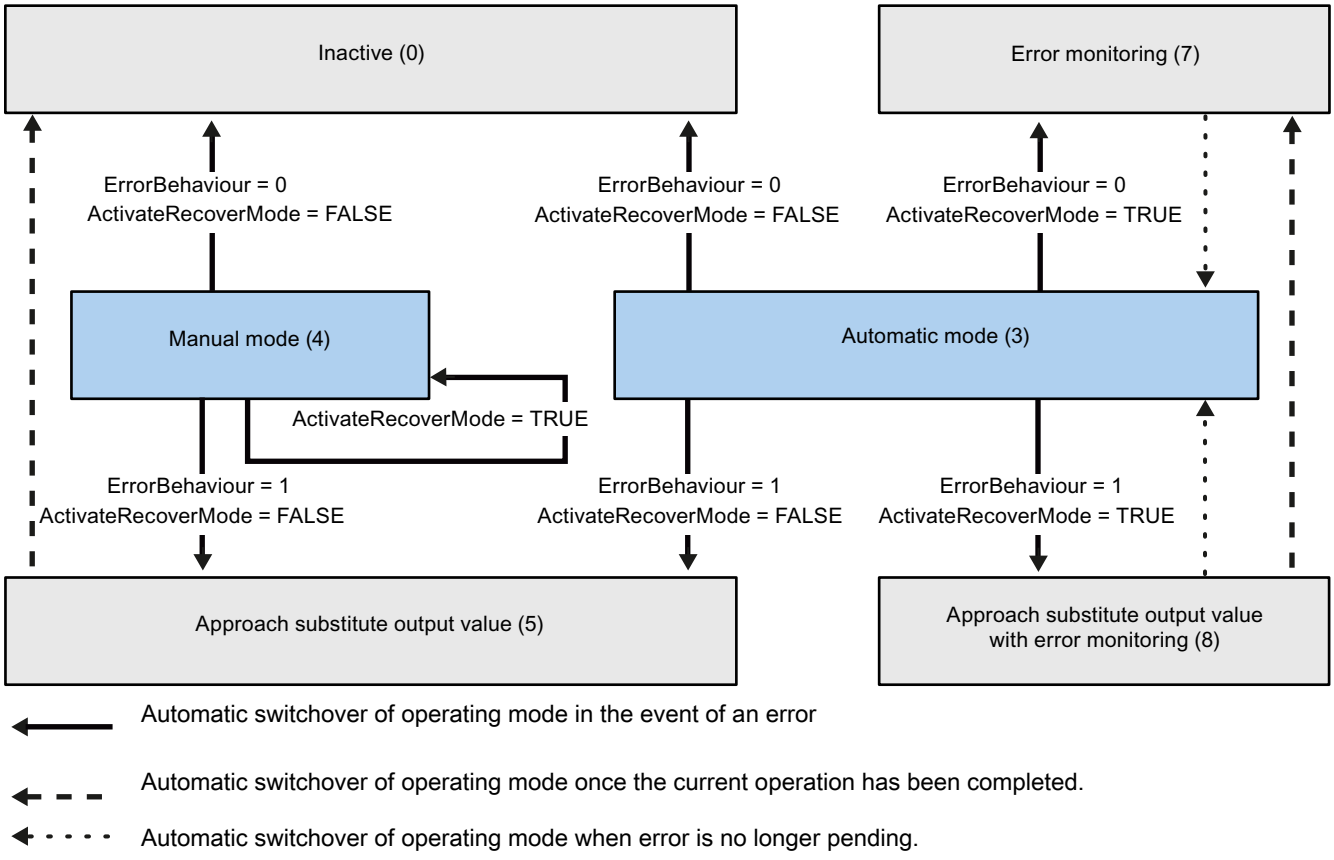


Automatic switchover of operating mode in the event of an error

Automatic switchover of operating mode once the current operation has been completed.

Automatic switchover of operating mode in automatic and manual modes (PID_3Step V1.0)

PID_3Step automatically switches the operating mode in the event of an error. The following diagram illustrates the influence of ErrorBehaviour and ActivateRecoverMode on this switchover of operating mode.



See also

Tag ActivateRecoverMode V1 (Page 4314)

Parameter ErrorBits V1 (Page 4312)

Parameter ErrorBits V1 (S7-1200)

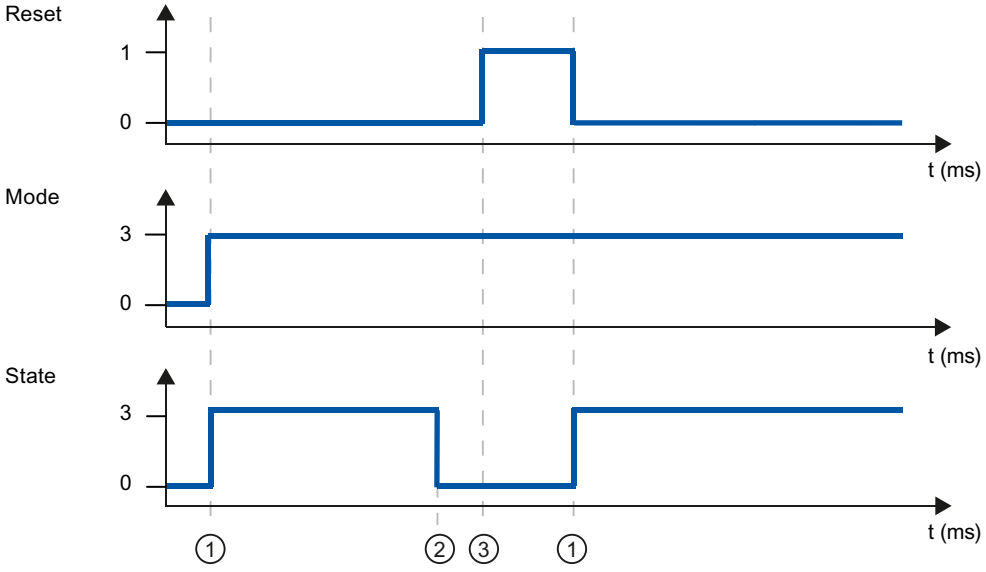
If several errors are pending simultaneously, the values of the error codes are displayed with binary addition. The display of error code 0003, for example, indicates that the errors 0001 and 0002 are pending simultaneously.

ErrorBits (DW#16#...)	Description
0000	There is no error.
0001	The "Input" parameter is outside the process value limits. • Input > Config.InputUpperLimit or • Input < Config.InputLowerLimit If ActivateRecoverMode = TRUE and ErrorBehaviour = 1, the actuator moves to the substitute output value. If ActivateRecoverMode = TRUE and ErrorBehaviour = 0, the actuator stops in its current position. If ActivateRecoverMode = FALSE, the actuator stops in its current position. PID_3Step V1.1 You can move the actuator in manual mode. PID_3Step V1.0 Manual mode is not possible in this state. You cannot move the actuator again until you eliminate the error.
0002	Invalid value at "Input_PER" parameter. Check whether an error is pending at the analog input. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode.
0004	Error during fine tuning. Oscillation of the process value could not be maintained.
0020	Pretuning is not permitted in automatic mode or during fine tuning.
0080	Error during pretuning. The output value limits are not configured correctly or the actual value does not react as expected. Check whether the limits of the output value are configured correctly and match the control logic. Also make sure that the actual value does not oscillate strongly before starting pretuning.
0100	Error during fine tuning resulted in invalid parameters.
0200	Invalid value at "Input" parameter: Value has an invalid number format. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode.
0400	Calculation of output value failed. Check the PID parameters.
0800	Sampling time error: PID_3Step is not called within the sampling time of the cyclic interrupt OB. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode. If this error occurred during simulation with PLCSIM, see the notes under Simulating PID_3Step V1 with PLCSIM (Page 6096).
1000	Invalid value at "Setpoint" parameter: Value has an invalid number format. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode.

ErrorBits (DW#16#...)	Description
2000	Invalid value at Feedback_PER parameter. Check whether an error is pending at the analog input. The actuator cannot be moved to the substitute output value and remains in its current position. Manual mode is not possible in this state. You must deactivate position feedback (Config. FeedbackOn = FALSE) to move the actuator from this state. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode.
4000	Invalid value at Feedback parameter. Value has an invalid number format. The actuator cannot be moved to the substitute output value and remains in its current position. Manual mode is not possible in this state. You must deactivate position feedback (Config. FeedbackOn = FALSE) to move the actuator from this state. If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode.
8000	Error during digital position feedback. Actuator_H = TRUE and Actuator_L = TRUE. The actuator cannot be moved to the substitute output value and remains in its current position. Manual mode is not possible in this state. In order to move the actuator from this state, you must deactivate the "Actuator endstop" (Config.ActuatorEndStopOn = FALSE). If automatic mode was active before the error occurred, ActivateRecoverMode = TRUE, and the error is no longer pending, PID_3Step switches back to automatic mode.

Parameter Reset V1 (S7-1200)

A rising edge at Reset triggers a change to "Inactive" mode, and errors and warnings are reset. A falling edge at Reset triggers a change to the most recently active operating mode. If automatic mode was active before, switchover to automatic mode is bumpless.



① Activation
② Error
③ Reset

Tag ActivateRecoverMode V1 (S7-1200)

The effect of the ActivateRecoverMode variable depends on the version of the PID_3Step.

Behavior in version 1.1

The ActivateRecoverMode variable determines the behavior in the event of an error in automatic mode. ActivateRecoverMode is not effective during pretuning, fine tuning and transition time measurement.

ActivateRecoverMode	Description
FALSE	In the event of an error, PID_3Step switches to "Inactive" or "Approach substitute output value" operating mode. The controller is activated by a reset or a change in Retain.Mode.
TRUE	If errors occur frequently in automatic mode, this setting has a negative effect on the control response. In this case, check the ErrorBits parameter and eliminate the cause of the error. If one or more errors occur, PID_3Step switches to "Approach substitute output value with error monitoring" or "Error monitoring" mode: • 0002h: Invalid value at parameter Input_PER. • 0200h: Invalid value at parameter Input. • 0800h: Sampling time error • 1000h: Invalid value at parameter Setpoint. • 2000h: Invalid value at parameter Feedback_PER. • 4000h: Invalid value at parameter Feedback. • 8000h: Error in digital position feedback. With errors 2000h, 4000h and 8000h, PID_3Step **cannot** approach the configured substitute output value. As soon as the errors are no longer pending, PID_3Step switches back to automatic mode.

Behavior in version 1.0

The ActivateRecoverMode variable determines the behavior in the event of an error in automatic and manual mode. ActivateRecoverMode is not effective during pretuning, fine tuning and transition time measurement.

ActivateRecoverMode	Description
FALSE	In the event of an error, PID_3Step switches to "Inactive" or "Approach substitute output value" operating mode. The controller is activated by a reset or a change in Retain.Mode.
TRUE	**Errors in automatic mode** If errors occur frequently in automatic mode, this setting has a negative effect on the control response. In this case, check the ErrorBits parameter and eliminate the cause of the error. If one or more errors occur, PID_3Step switches to "Approach substitute output value with error monitoring" or "Error monitoring" mode: • 0002h: Invalid value at parameter Input_PER. • 0200h: Invalid value at parameter Input. • 0800h: Sampling time error • 1000h: Invalid value at parameter Setpoint. • 2000h: Invalid value at parameter Feedback_PER. • 4000h: Invalid value at parameter Feedback. • 8000h: Error in digital position feedback. With errors 2000h, 4000h and 8000h, PID_3Step **cannot** approach the configured substitute output value. As soon as the errors are no longer pending, PID_3Step switches back to automatic mode. **Errors in manual mode** If one or more of the following errors occur, PID_3Step stays in manual mode: • 0002h: Invalid value at parameter Input_PER. • 0200h: Invalid value at parameter Input. • 0800h: Sampling time error • 1000h: Invalid value at parameter Setpoint. • 2000h: Invalid value at parameter Feedback_PER. • 4000h: Invalid value at parameter Feedback. • 8000h: Error in digital position feedback. With errors 2000h, 4000h and 8000h, you **cannot** move the valve to a suitable position.

See also

PID_3Step V1 static tags (Page 4298)

Parameter State and Retain.Mode V1 (Page 4305)

Tag Warning V1 (S7-1200)

If several warnings are pending simultaneously, their values are displayed with binary addition. The display of warning 0003, for example, indicates that the warnings 0001 and 0002 are pending simultaneously.

Warning (DW#16#...)	Description
0000	No warning pending.
0001	The point of inflection was not found during pretuning.
0002	Oscillation increased during fine tuning.
0004	The setpoint was limited to the configured limits.
0008	Not all the necessary controlled system properties were defined for the selected method of calculation. The PID parameters were instead calculated using the TuneRuleTIR = 3 method.
0010	The operating mode could not be changed because ManualEnable = TRUE.
0020	The cycle time of the calling OB limits the sampling time of the PID algorithm. Improve results by using shorter OB cycle times.
0040	The process value exceeded one of its warning limits.
0080	Invalid value at Retain.Mode. The operating mode is not switched.
0100	The manual value was limited to the limits of the controller output.
0200	The rule used for tuning produces an incorrect result, or is not supported.
0400	Method selected for transition time measurement not suitable for actuator. The transition time cannot be measured because the actuator settings do not match the selected measuring method.
0800	The difference between the current position and the new output value is too small for transition time measurement. This can produce incorrect results. The difference between the current output value and new output value must be at least 50% of the entire control range.
1000	The substitute output value cannot be reached because it is outside the output value limits.

The following warnings are deleted as soon as the cause is eliminated:

- 0004
- 0020
- 0040
- 0100

All other warnings are cleared with a rising edge at Reset.

Tag SUT.State V1 (S7-1200)

SUT.State	Name	Description
0	SUT_INIT	Initialize pretuning
50	SUT_TPDN	Determine start position without position feedback
100	SUT_STDABW	Calculate the standard deviation
200	SUT_GET_POI	Find the point of inflection
300	SUT_GET_RISETM	Determine the rise time

SUT.State	Name	Description
9900	SUT_IO	Pretuning successful
1	SUT_NIO	Pretuning not successful

Tag TIR.State V1 (S7-1200)

TIR.State	Name	Description
-100	TIR_FIRST_SUT	Fine tuning is not possible. Pretuning will be executed first.
0	TIR_INIT	Initialize fine tuning
200	TIR_STDABW	Calculate the standard deviation
300	TIR_RUN_IN	Attempt to reach the setpoint with the maximum or minimum output value
400	TIR_CTRLN	Attempt to reach the setpoint with the existing PID parameters (if pretuning has been successful)
500	TIR_OSZIL	Determine oscillation and calculate parameters
9900	TIR_IO	Fine tuning successful
1	TIR_NIO	Fine tuning not successful

PID_Temp (S7-1200, S7-1500)

New features of PID_Temp (S7-1200, S7-1500)

PID_Temp V1.1

- **Response of the output value on switchover from "Inactive" operating mode to "Automatic mode"**
  The new option IntegralResetMode = 4 was added and defined as default. With IntegralResetMode = 4, the integral action is automatically pre-assigned when switching from "Inactive" operating mode to "Automatic mode" so that a control deviation results in a jump of the PID output value with identical sign.
- **Initialization of the integral action in automatic mode**
  The integral action can be initialized in automatic mode with the tags OverwriteInitialOutputValue and PIDCtrl.PIDInit. This simplifies the use of PID_Temp for override controls.

Compatibility with CPU and FW (S7-1200, S7-1500)

The following table shows which version of PID_Temp can be used on which CPU.

CPU	FW	PID_Temp
S7-1200	V4.2 or higher	V1.1 V1.0
	V4.1	V1.0

CPU	FW	PID_Temp
S7-1500	V2.0 or higher	V1.1 V1.0
	V1.7 to V1.8	V1.0

CPU processing time and memory requirement PID_Temp V1 (S7-1200, S7-1500)

CPU processing time

Typical CPU processing times of the PID_Temp technology object as of Version 1.0, depending on CPU type.

CPU	Typ. CPU processing time PID_Temp V1
CPU 1211C ≥ V4.1	580 µs
CPU 1215C ≥ V4.1	580 µs
CPU 1217C ≥ V4.1	580 µs
CPU 1505S ≥ V1.0	50 µs
CPU 1510SP-1 PN ≥ V1.7	130 µs
CPU 1511-1 PN ≥ V1.7	130 µs
CPU 1512SP-1 PN ≥ V1.7	130 µs
CPU 1516-3 PN/DP ≥ V1.7	75 µs
CPU 1518-4 PN/DP ≥ V1.7	6 µs

Memory requirement

Memory requirement of an instance DB of the PID_Temp technology object as of Version V1.0.

	Memory requirement of the instance DB of PID_Temp V1
Load memory requirement	Approx. 17000 bytes
Total work memory requirement	1280 bytes
Retentive work memory requirement	100 bytes

PID_Temp (S7-1200, S7-1500)

Description of PID_Temp (S7-1200, S7-1500)

Description

The PID_Temp instruction provides a PID controller with integrated tuning for temperature processes. PID_Temp can be used for pure heating or heating/cooling applications.

The following operating modes are possible:

- Inactive
- Pretuning
- Fine tuning
- Automatic mode
- Manual mode
- Substitute output value with error monitoring

For a more detailed description of the operating modes, see the State parameter.

PID algorithm

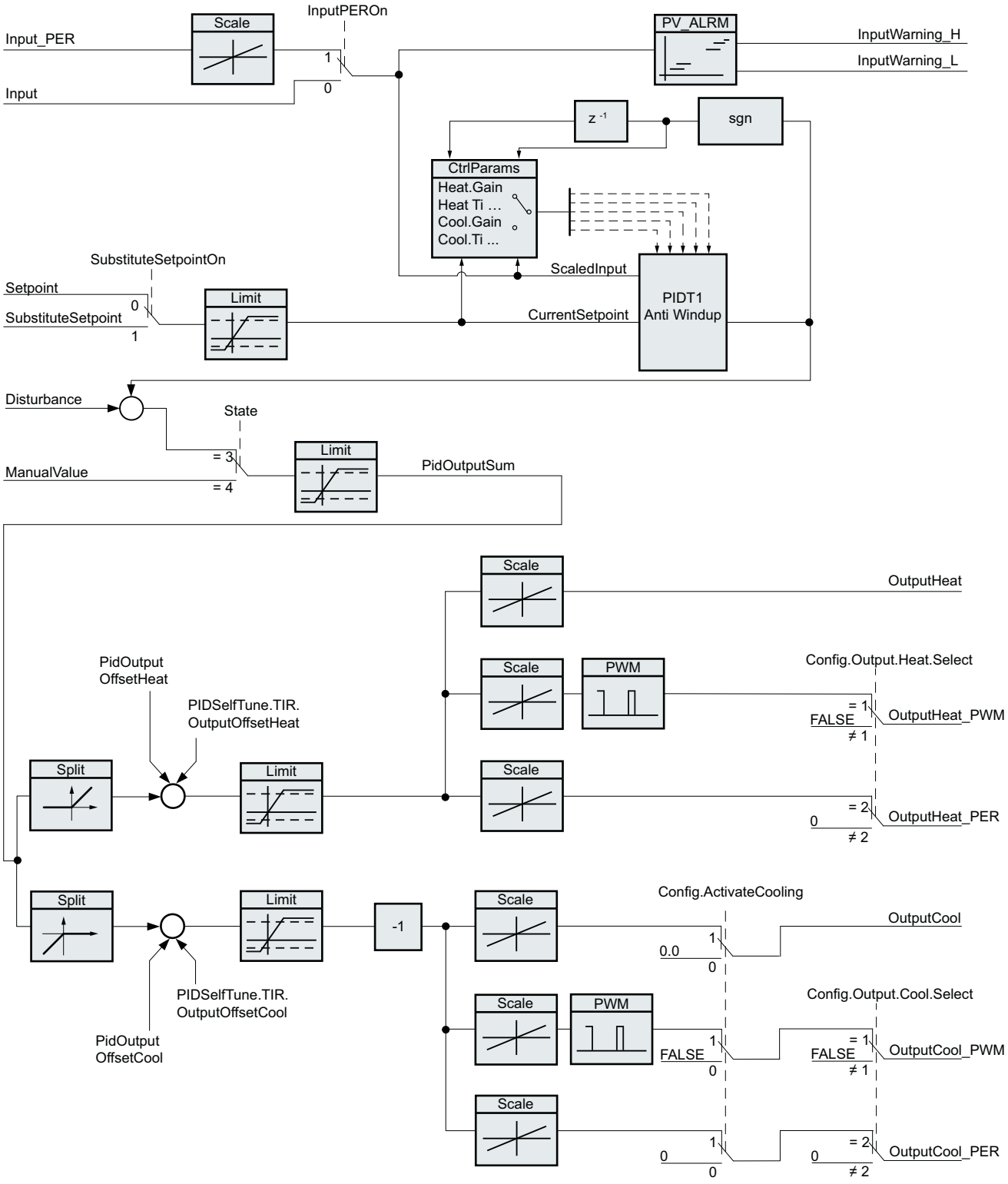
PID_Temp is a PIDT1 controller with anti-windup and weighting of the proportional and derivative actions. The PID algorithm operates according to the following equation (control zone and dead zone deactivated):

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

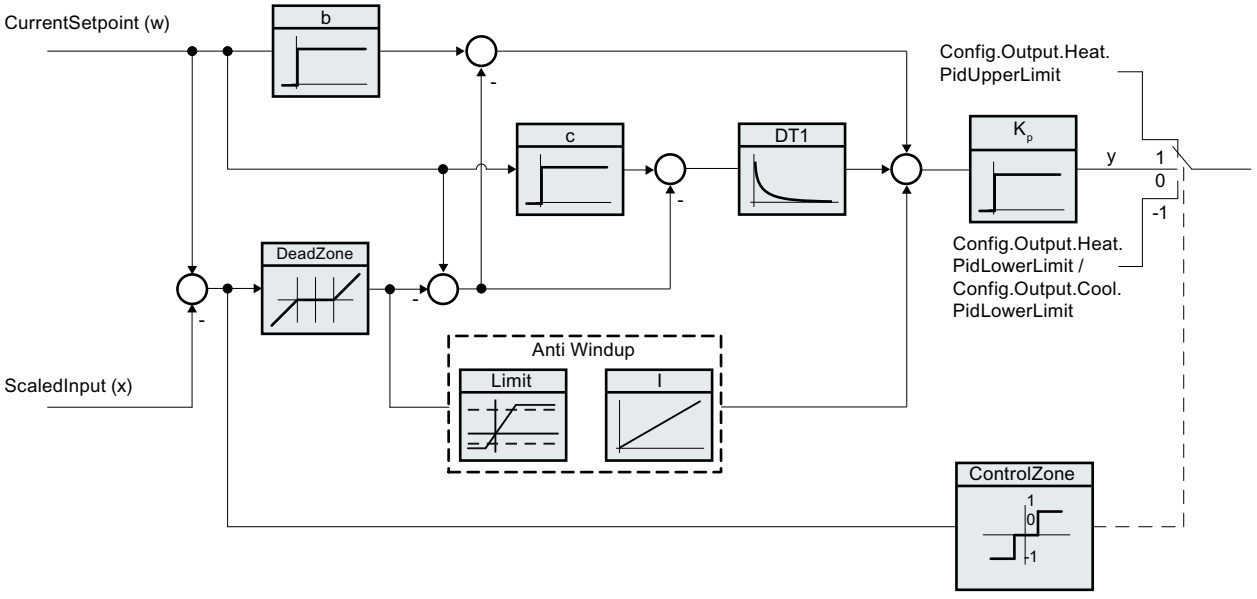
The table below shows the meaning of the icons used in the equation and in the subsequent figures.

Icon	Description	Associated parameters of the PID_Temp instruction
y	Output value of the PID algorithm	-
K_p	Proportional gain	Retain.CtrlParams.Heat.Gain Retain.CtrlParams.Cool.Gain CoolFactor
s	Laplace operator	-
b	Proportional action weighting	Retain.CtrlParams.Heat.PWeighting Retain.CtrlParams.Cool.PWeighting
w	Setpoint	CurrentSetpoint
x	Process value	ScaledInput
T_I	Integral action time	Retain.CtrlParams.Heat.Ti Retain.CtrlParams.Cool.Ti
T_D	Derivative action time	Retain.CtrlParams.Heat.Td Retain.CtrlParams.Cool.Td
a	Derivative delay coefficient (derivative delay T1 = a × T_D)	Retain.CtrlParams.Heat.TdFiltRatio Retain.CtrlParams.Cool.TdFiltRatio
c	Derivative action weighting	Retain.CtrlParams.Heat.DWeighting Retain.CtrlParams.Cool.DWeighting
DeadZone	Dead zone width	Retain.CtrlParams.Heat.DeadZone Retain.CtrlParams.Cool.DeadZone
ControlZone	Control zone width	Retain.CtrlParams.Heat.ControlZone Retain.CtrlParams.Cool.ControlZone

PID_Temp block diagram



Block diagram of PIDT1 with anti-windup



Call

PID_Temp is called in the constant time scale of a cyclic interrupt OB.

If you call PID_Temp as a multi-instance DB, no technology object is created. No parameter assignment interface or commissioning interface is available. You must assign parameters for PID_Temp directly in the multi-instance DB and commission it via a watch table.

Download to device

The process values of retentive tags are only updated when you download PID_Temp completely.

Download technology object to device (Page 6000)

Startup

When the CPU starts up, PID_Temp starts in the operating mode that is saved in the Mode in/out parameter. To switch to "Inactive" operating mode during startup, set RunModeByStartup = FALSE.

Reaction to error

The behavior in the case of an error is determined by the tags SetSubstituteOutput and ActivateRecoverMode. If ActivateRecoverMode = TRUE, the behavior also depends on the error that occurred.

SetSubstitu-teOutput	ActivateReco-verMode	Configuration editor > Basic settings of output > Set PidOutputSum to	Reaction
Not relevant	FALSE	Zero (inactive)	Switch to "Inactive" (State = 0) mode The output value of the PID algorithm and all outputs for heating and cooling are set to 0. The scaling of the outputs for heating and cooling is not active.
FALSE	TRUE	Current value for error while error is pending	Switch to "Substitute output value with error monitoring" mode (State = 5) The current output value is transferred to the actuator while the error is pending.
TRUE	TRUE	Substitute output value while error is pending	Switch to "Substitute output value with error monitoring" mode (State = 5) The value at SubstituteOutput is transferred to the actuator while the error is pending.

In manual mode, PID_Temp uses ManualValue as output value, unless ManualValue is invalid.

- If ManualValue is invalid, SubstituteOutput is used.
- If ManualValue and SubstituteOutput are invalid, Config.Output.Heat.PidLowerLimit is used.

The Error parameter indicates if an error is pending. When the error is no longer pending, Error = FALSE. The ErrorBits parameter shows which errors have occurred. ErrorBits is reset by a rising edge at Reset or ErrorAck.

Mode of operation of PID_Temp (S7-1200, S7-1500)

Monitoring process value limits

You specify the high limit and low limit of the process value in the Config.InputUpperLimit and Config.InputLowerLimit tags. If the process value is outside these limits, an error occurs (ErrorBits = 0000001h).

You specify a high and low warning limit of the process value in the Config.InputUpperWarning and Config.InputLowerWarning tags. If the process value is outside these warning limits, a warning occurs (Warning = 0000040h), and the InputWarning_H or InputWarning_L output parameter changes to TRUE.

Limiting the setpoint

You specify a high limit and low limit of the setpoint in the Config.SetpointUpperLimit and Config.SetpointLowerLimit tags. PID_Temp automatically limits the setpoint to the process value limits. You can limit the setpoint to a smaller area. PID_Temp checks whether this area is within the process value limits. If the setpoint is outside these limits, the high or low limit is used as the setpoint, and output parameter SetpointLimit_H or SetpointLimit_L is set to TRUE.

The setpoint is limited in all operating modes.

Substitute setpoint

You can specify a substitute setpoint at the SubstituteSetpoint tag and activate it with SubstituteSetpointOn = TRUE. In this way, you can temporarily specify the setpoint directly, for example for a slave controller in a cascade, without having to change the user program. The limits set for the setpoint also apply to the substitute setpoint.

Heating and cooling

With the default setting, PID_Temp only uses the outputs for heating (OutputHeat, OutputHeat_PWM, OutputHeat_PER). The output value of the PID algorithm (PidOutputSum) is scaled and output at the outputs for heating. You specify with Config.Output.Heat.Select if OutputHeat_PWM or OutputHeat_PER is calculated. OutputHeat is always calculated.

With Config.ActivateCooling = TRUE, you can also activate the outputs for cooling (OutputCool, OutputCool_PWM, OutputCool_PER). Positive output values of the PID algorithm (PidOutputSum) are scaled and output at the outputs for heating. Negative output values of the PID algorithm are scaled and output at the outputs for cooling. You specify with Config.Output.Cool.Select if OutputCool_PWM or OutputCool_PER is calculated. OutputCool is always calculated.

Two methods are available to calculate the PID output value with activated cooling:

- Cooling factor (Config.AdvancedCooling = FALSE):
  The output value calculation for cooling takes place with the PID parameters for heating, taking into consideration the configurable cooling factor Config.CoolFactor. This method is suitable if the heating and cooling actuators have a similar time response but different gains. If this method is selected, pretuning and fine tuning for cooling as well as the PID parameter set for cooling are not available. You can only execute the tuning for heating.
- PID parameter switching (Config.AdvancedCooling = TRUE):
  The output value calculation for cooling takes place by means of a separate PID parameter set. Based on the calculated output value and the control deviation, the PID algorithm decides whether the PID parameter for heating or cooling is used. This method is suitable if the heating and cooling actuator have different time responses and different gains. Pretuning and fine tuning for cooling are only available if this method is selected.

Output value limits and scaling

Depending on the operating mode, the PID output value (PidOutputSum) is calculated automatically by the PID algorithm or defined by the manual value (ManualValue) or the configured substitute output value (SubstituteOutput).

The PID output value is limited according to the configuration:

- If cooling is deactivated (Config.ActivateCooling = FALSE), Config.Output.Heat.PidUpperLimit is the high limit and Config.Output.Heat.PidLowerLimit the low limit.
- If cooling is activated (Config.ActivateCooling = TRUE), Config.Output.Heat.PidUpperLimit is the high limit and Config.Output.Cool.PidLowerLimit the low limit.

The PID output value is scaled and output at the outputs for heating and cooling. Scaling can be defined separately for each output and is specified in the structures Config.Output.Heat or Config.Output.Cool with 2 value pairs each:

Output	Value pair	Parameter
OutputHeat	Value pair 1	PID output value high limit (heating) Config.Output.Heat.PidUpperLimit, Scaled high output value (heating) Config.Output.Heat.UpperScaling
	Value pair 2	PID output value low limit (heating) Config.Output.Heat.PidLowerLimit, Scaled low output value (heating) Config.Output.Heat.LowerScaling
OutputHeat_PWM	Value pair 1	PID output value high limit (heating) Config.Output.Heat.PidUpperLimit, Scaled high PWM output value (heating) Config.Output.Heat.PwmUpperScaling
	Value pair 2	PID output value low limit (heating) Config.Output.Heat.PidLowerLimit, Scaled low PWM output value (heating) Config.Output.Heat.PwmLowerScaling
OutputHeat_PER	Value pair 1	PID output value high limit (heating) Config.Output.Heat.PidUpperLimit, Scaled high analog output value (heating) Config.Output.Heat.PerUpperScaling
	Value pair 2	PID output value low limit (heating) Config.Output.Heat.PidLowerLimit, Scaled low analog output value (heating) Config.Output.Heat.PerLowerScaling
OutputCool	Value pair 1	PID output value low limit (cooling) Config.Output.Cool.PidLowerLimit, Scaled high output value (cooling) Config.Output.Cool.UpperScaling
	Value pair 2	PID output value high limit (cooling) Config.Output.Cool.PidUpperLimit, Scaled low output value (cooling) Config.Output.Cool.LowerScaling

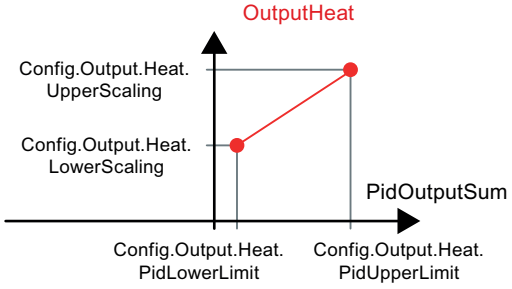
Output	Value pair	Parameter
OutputCool_PWM	Value pair 1	PID output value low limit (cooling) Config.Output.Cool.PidLowerLimit, Scaled high PWM output value (cooling) Config.Output.Cool.PwmUpperScaling
	Value pair 2	PID output value high limit (cooling) Config.Output.Cool.PidUpperLimit, Scaled low PWM output value (cooling) Config.Output.Cool.PwmLowerScaling
OutputCool_PER	Value pair 1	PID output value low limit (cooling) Config.Output.Cool.PidLowerLimit, Scaled high analog output value (cooling) Config.Output.Cool.PerUpperScaling
	Value pair 2	PID output value high limit (cooling) Config.Output.Cool.PidUpperLimit, Scaled low analog output value (cooling) Config.Output.Cool.PerLowerScaling

If cooling is activated (Config.ActivateCooling = TRUE), Config.Output.Heat.PidLowerLimit must have the value 0.0.

Config.Output.Cool.PidUpperLimit must always have the value 0.0.

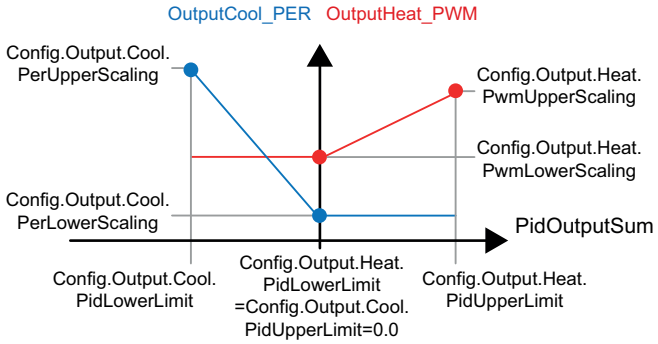
Example:

Output scaling when using output OutputHeat (cooling deactivated; Config.Output.Heat.PidLowerLimit may be unequal to 0.0):



Example:

Output scaling when using output OutputHeat_PWM and OutputCool_PER (cooling activated; Config.Output.Heat.PidLowerLimit must be 0.0):



With the exception of the "Inactive" operating mode, the value at an output is always located between its scaled high output value and scaled low output value, for example, for OutputHeat always between Config.Output.Heat.UpperScaling and Config.Output.Heat.LowerScaling.

If you want to limit the value at the associated output, you therefore have to adapt these scaling values as well.

Cascading

PID_Temp supports you when you use cascade control (see: Program creation (Page 6128)).

Substitute output value

In the event of an error, PID_Temp can output a substitute output value that you define at the SubstituteOutput tag. The substitute output value must be within the limits for the PID output value. The values at the outputs for heating and cooling resulting from the substitute output value are the result of the configured output scaling.

Monitoring signal validity

The values of the following parameters are monitored for validity when used:

- Setpoint
- SubstituteSetpoint
- Input
- Input_PER
- Disturbance
- ManualValue
- SubstituteOutput
- PID parameters in the structures Retain.CtrlParams.Heat and Retain.CtrlParams.Cool.

Monitoring the sampling time PID_Temp

Ideally, the sampling time is equivalent to the cycle time of the cyclic interrupt OB. The PID_Temp instruction measures the time interval between two calls. This is the current sampling time. On every switchover of operating mode and during the initial startup, the mean value is formed from the first 10 sampling times. Too great a difference between the current sampling time and this mean value triggers an error (Error = 0000800h).

The error occurs during tuning if:

- New mean value >= 1.1 x old mean value
- New mean value <= 0.9 x old mean value

The error occurs in automatic mode if:

- New mean value >= 1.5 x old mean value
- New mean value <= 0.5 x old mean value

If you deactivate the sampling time monitoring (CycleTime.EnMonitoring = FALSE), you can also call PID_Temp in OB1. You must then accept a lower control quality due to the deviating sampling time.

Sampling time of the PID algorithm

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the cycle time of the cyclic interrupt OB (sampling time PID_Temp). All other functions of the PID_Temp are executed at every call.

If cooling and PID parameter switching are activated, PID_Temp uses a separate sampling time of the PID algorithm for heating and cooling. In all other configurations, only the sampling time of the PID algorithm for heating is used.

If you use OutputHeat_PWM or OutputCool_PWM, the sampling time of the PID algorithm is used as time period of the pulse width modulation. The accuracy of the output signal is determined by the ratio of the PID algorithm sampling time to the cycle time of the OB. The cycle time should be no more than a tenth of the PID algorithm sampling time.

If the PID algorithm sampling time and thus the time period of the pulse width modulation is very high when you use OutputHeat_PWM or OutputCool_PWM, you can define a deviating shorter period duration at the Config.Output.Heat.PwmPeriode or Config.Output.Cool.PwmPeriode parameters to improve the smoothness of the process value.

Control logic

PID_Temp can be used for heating or heating/cooling applications and always works with normal control logic.

An increase of the PID output value (PidOutputSum) is intended to increase the process value. The values at the outputs for heating and cooling resulting from the PID output value are the result of the configured output scaling.

An inverted control logic or negative proportional gain are not supported.

If you only need an output value for your application in which an increase is to reduce the process value (for example, discharge control), you can use PID_Compact with inverted control logic.

Input parameters of PID_Temp (S7-1200, S7-1500)

Parameter	Data type	Default	Description
Setpoint	REAL	0.0	Setpoint of the PID controller in automatic mode Valid range of values: Config.SetpointUpperLimit ≥ Setpoint ≥ Config.SetpointLowerLimit Config.InputUpperLimit ≥ Setpoint ≥ Config.InputLowerLimit
Input	REAL	0.0	A tag of the user program is used as source for the process value. If you are using the Input parameter, Config.InputPerOn = FALSE must be set.
Input_PER	INT	0	An analog input is used as the source of the process value. If you are using the Input_PER parameter, Config.InputPerOn = TRUE must be set.
Disturbance	REAL	0.0	Disturbance variable or precontrol value
ManualEnable	BOOL	FALSE	• A FALSE -> TRUE edge activates "Manual mode", whileState = 4, Mode remains unchanged. As long as ManualEnable = TRUE, you cannot change the operating mode via a rising edge at ModeActivate or use the commissioning dialog. • A TRUE -> FALSE edge activates the operating mode that is specified by Mode. We recommend that you change the operating mode using Mode and ModeActivate only.
ManualValue	REAL	0.0	Manual value This value is used in manual mode as PID output value (PidOutputSum). The values at the outputs for heating and cooling resulting from this manual value are the result of the configured output scaling (structures Config.Output.Heat and Config.Output.Cool). For controllers with activated cooling output (Config.ActivateCooling = TRUE), define: • a positive manual value to output the value at the outputs for heating • a negative manual value to output the value at the outputs for cooling The permitted value range is determined by the configuration. • Cooling output deactivated (Config.ActivateCooling = FALSE): Config.Output.Heat.PidUpperLimit ≥ ManualValue ≥ Config.Output.Heat.PidLowerLimit • Cooling output activated (Config.ActivateCooling = TRUE): Config.Output.Heat.PidUpperLimit ≥ ManualValue ≥ Config.Output.Cool.PidLowerLimit
ErrorAck	BOOL	FALSE	• FALSE -> TRUE edge ErrorBits and Warning are reset.

Parameter	Data type	Default	Description
Reset	BOOL	FALSE	Restarts the controller. • FALSE -> TRUE edge – Switch to "Inactive" mode – ErrorBits and Warning are reset. • As long as Reset = TRUE, – PID_Temp remains in "Inactive" mode (State = 0). – you cannot change the operating mode with Mode and ModeActivate or ManualEnable – You cannot use the commissioning dialog. • TRUE -> FALSE edge – If ManualEnable = FALSE, PID_Temp switches to the operating mode that is saved in Mode. – If Mode = 3 (automatic mode), the integral action is treated as configured with the tag IntegralResetMode.
ModeActivate	BOOL	FALSE	• FALSE -> TRUE edge PID_Temp switches to the operating mode that is saved at the Mode input.

Output parameters of PID_Temp (S7-1200, S7-1500)

Parameter	Data type	Default	Description
ScaledInput	REAL	0.0	Scaled process value
OutputHeat	REAL	0.0	Output value (heating) in REAL format The PID output value (PidOutputSum) is scaled with the two value pairs Config.Output.Heat.PidUpperLimit, Config.Output.Heat.UpperScaling and Config.Output.Heat.PidLowerLimit, Config.Output.Heat.LowerScaling and output in REAL format at OutputHeat. OutputHeat is always calculated.
OutputCool	REAL	0.0	Output value (cooling) in REAL format The PID output value (PidOutputSum) is scaled with the two value pairs Config.Output.Cool.PidUpperLimit, Config.Output.Cool.LowerScaling and Config.Output.Cool.PidLowerLimit, Config.Output.Cool.UpperScaling and output in REAL format at OutputCool. OutputCool is only calculated if the cooling output is activated (Config.ActivateCooling = TRUE).
OutputHeat_PER	INT	0	Analog output value (heating) The PID output value (PidOutputSum) is scaled with the two value pairs Config.Output.Heat.PidUpperLimit, Config.Output.Heat.PerUpperScaling and Config.Output.Heat.PidLowerLimit, Config.Output.Heat.PerLowerScaling and output as analog value at OutputHeat_PER. OutputHeat_PER is only calculated if Config.Output.Heat.Select = 2.

Parameter	Data type	Default	Description
Output-Cool_PER	INT	0	Analog output value (cooling) The PID output value (PidOutputSum) is scaled with the two value pairs Config.Output.Cool.PidUpperLimit, Config.Output.Cool.PerLowerScaling and Config.Output.Cool.PidLowerLimit, Config.Output.Cool.PerUpperScaling and output as analog value at OutputCool_PER. OutputCool_PER is only calculated if the cooling output is activated (Config.ActivateCooling = TRUE) and Config.Output.Cool.Select = 2.
OutputHeat_PWM	BOOL	FALSE	Pulse-width modulated output value (heating) The PID output value (PidOutputSum) is scaled with the two value pairs Config.Output.Heat.PidUpperLimit, Config.Output.Heat.PwmUpperScaling and Config.Output.Heat.PidLowerLimit, Config.Output.Heat.PwmLowerScaling and output as pulse-width modulated value (variable switch on and switch off times) at OutputHeat_PWM. OutputHeat_PWM is only calculated if Config.Output.Heat.Select = 1.
Output-Cool_PWM	BOOL	FALSE	Pulse-width modulated output value (cooling) The PID output value (PidOutputSum) is scaled with the two value pairs Config.Output.Cool.PidUpperLimit, Config.Output.Cool.PwmLowerScaling and Config.Output.Cool.PidLowerLimit, Config.Output.Cool.PwmUpperScaling and output as pulse-width modulated value (variable switch on and switch off times) at OutputCool_PWM. OutputCool_PWM is only calculated if the cooling output is activated (Config.ActivateCooling = TRUE) and Config.Output.Cool.Select = 1.
SetpointLimit_H	BOOL	FALSE	If SetpointLimit_H = TRUE, the absolute setpoint high limit is reached (Setpoint ≥ Config.SetpointUpperLimit) or Setpoint ≥ Config.InputUpperLimit. The setpoint high limit is the minimum of Config.SetpointUpperLimit and Config.InputUpperLimit.
SetpointLimit_L	BOOL	FALSE	If SetpointLimit_L = TRUE, the absolute setpoint low limit is reached (Setpoint ≤ Config.SetpointLowerLimit) or Setpoint ≤ Config.InputLowerLimit. The setpoint low limit is the maximum of Config.SetpointLowerLimit and Config.InputLowerLimit.
InputWarning_H	BOOL	FALSE	If InputWarning_H = TRUE, the process value has reached or exceeded the warning high limit (ScaledInput ≥ Config.InputUpperWarning).
InputWarning_L	BOOL	FALSE	If InputWarning_L = TRUE, the process value has reached or fallen below the warning low limit (ScaledInput ≤ Config.InputLowerWarning).
State	INT	0	The PID_Temp state and mode parameters (Page 4364) shows the current operating mode of the PID controller. You can change the operating mode using the input parameter Mode and a rising edge at ModeActivate. For pretuning and fine tuning, you specify with Heat.EnableTuning and Cool.EnableTuning whether tuning takes place for heating or cooling. • State = 0: Inactive • State = 1: Pretuning • State = 2: Fine tuning • State = 3: Automatic mode • State = 4: Manual mode • State = 5: Substitute output value with error monitoring

Parameter	Data type	Default	Description
Error	BOOL	FALSE	If Error = TRUE, at least one error message is pending in this cycle.
ErrorBits	DWORD	DW#16#0	The PID_Temp ErrorBits parameter (Page 4373) shows the pending error messages. ErrorBits is retentive and is reset with a rising edge at Reset or ErrorAck.

In/out parameters of PID_Temp V2 (S7-1200, S7-1500)

Parameter	Data type	Default	Description
Mode	INT	4	At Mode, specify the operating mode to which PID_Temp is to switch. Options are: • Mode = 0: Inactive • Mode = 1: Pretuning • Mode = 2: Fine tuning • Mode = 3: Automatic mode • Mode = 4: Manual mode The operating mode is activated by: • Rising edge at ModeActivate • Falling edge at Reset • Falling edge at ManualEnable • Cold restart of CPU if RunModeByStartup = TRUE For pretuning and fine tuning, you specify with Heat.EnableTuning and Cool.EnableTuning whether tuning takes place for heating or cooling. Mode is retentive. A detailed description of the operating modes can be found in State and Mode parameters (Page 4364).
Master	DWORD	DW#16#0	Interface for cascade control If this PID_Temp instance is used as slave controller in a cascade (Config.Cascade.IsSlave = TRUE), assign the Master parameter at the instruction call with the Slave parameter of the master controller. Example: Call of a slave controller "PID_Temp_2" with master controller "PID_Temp_1" in SCL: --- `"PID_Temp_2"(Master := "PID_Temp_1".Slave, Setpoint := "PID_Temp_1".OutputHeat);` --- You use this interface to exchange slave controller information about operating mode, limit and substitute setpoint with your master controller. Keep in mind that the call of the master controller has to take place before the call of the slave controller in the same cyclic interrupt OB. Assignment: • Bits 0 to 15: Unassigned • Bits 16 to 23 – Limit counter: A slave controller whose output value is limited increments this counter. Depending on the configured number of slaves (Config.Cascade.CountSlaves) and of the anti-windup mode (Config.Cascade.AntiWindUpMode), the master controller reacts accordingly. • Bit 24 – Automatic mode of the slave controllers: TRUE, if all slave controllers are in automatic mode • Bit 25 – Substitute setpoint of the slave controllers: TRUE, if a slave controller has activated the substitute setpoint (SubstituteSetpointOn = TRUE)

Parameter	Data type	Default	Description
Slave	DWORD	DW#16#0	Interface for cascade control You use this interface to exchange slave controller information about operating mode, limit and substitute setpoint with your master controller. See description of Master parameter

See also

PID_Temp state and mode parameters (Page 4364)

Program creation (Page 6128)

Cascade control with PID_Temp (Page 6126)

PID_Temp static tags (S7-1200, S7-1500)

You must not change tags that are not listed. These are used for internal purposes only.

Tag	Data type	Default	Description
IntegralResetMode	Int	V1.0: 1, V1.1 or higher: 4	The IntegralResetMode tag (Page 4381) determines how the integral action PIDCtrl.IOutputOld is pre-assigned when switching from "Inactive" operating mode to "Automatic mode". This setting only works for one cycle. • IntegralResetMode = 0: Smooth • IntegralResetMode = 1: Delete • IntegralResetMode = 2: Hold • IntegralResetMode = 3: Pre-assign • IntegralResetMode = 4: Like setpoint change (only for PID_Temp with version ≥ 1.1)
OverwriteInitialOutputValue	REAL	0.0	If one of the following conditions is met, the integral action PIDCtrl.IOutputOld is pre-assigned automatically as if PIDOutputSum = OverwriteInitialOutputValue in the previous cycle: • IntegralResetMode = 3 when switching from "Inactive" operating mode to "Automatic mode" • TRUE -> FALSE edge at parameter Reset and parameter Mode = 3 • PIDCtrl.PIDInit = TRUE in "Automatic mode" (available as of PID_Temp version 1.1)
RunModeByStartup	BOOL	TRUE	Activate operating mode at Mode parameter after CPU restart • If RunModeByStartup = TRUE, PID_Temp starts in the operating mode saved in the Mode parameter after CPU startup. • If RunModeByStartup = FALSE, PID_Temp remains in "Inactive" mode after CPU startup.
LoadBackUp	BOOL	FALSE	If LoadBackUp = TRUE, the last set of PID parameters is reloaded from the CtrlParamsBackUp structure. The set was saved prior to the last tuning. LoadBackUp is automatically set back to FALSE. The acceptance is bumpless.

Tag	Data type	Default	Description
SetSubstituteOutput	BOOL	TRUE	Selection of the output value while an error is pending (State = 5): • If SetSubstituteOutput = TRUE and ActivateRecoverMode = TRUE, the configured substitute output value SubstituteOutput is output as PID output value as long as an error is pending. • If SetSubstituteOutput = FALSE and ActivateRecoverMode = TRUE, the actuator remains at the current PID output value as long as an error is pending. • If ActivateRecoverMode = FALSE, SetSubstituteOutput is not effective. • If SubstituteOutput is invalid (ErrorBits = 0020000h), the substitute output value cannot be output. In this case, the low limit of the PID output value for heating (Config.Output.Heat.PidLowerLimit) is used as PID output value.
PhysicalUnit	INT	0	Unit of measurement of the process value and setpoint, e.g., ºC, or ºF. This parameter is used for display in the editors and does not influence the control algorithm.
PhysicalQuantity	INT	0	Physical quantity of the process value and setpoint, e.g., temperature. This parameter is used for display in the editors and does not influence the control algorithm.
ActivateRecoverMode	BOOL	TRUE	The ActivateRecoverMode tag determines the reaction to error.
Warning	DWORD	0	The Warning tag shows the warnings since Reset = TRUE or ErrorAck =TRUE. Warning is retentive.
Progress	REAL	0.0	Progress of current tuning phase as a percentage (0.0 - 100.0)
CurrentSetpoint	REAL	0.0	CurrentSetpoint always displays the currently effective setpoint. This value is frozen during tuning.
CancelTuningLevel	REAL	10.0	Permissible fluctuation of setpoint during tuning. Tuning is not canceled until: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel

Tag	Data type	Default	Description
SubstituteOutput	REAL	0.0	The substitute output value is used as PID output value as long as the following conditions are met: • One or more errors are pending in automatic mode for which ActivateRecoverMode is in effect • SetSubstituteOutput = TRUE • ActivateRecoverMode = TRUE The values at the outputs for heating and cooling resulting from the substitute output value are the result of the configured output scaling (structures Config.Output.Heat and Config.Output.Cool). For controllers with activated cooling output (Config.ActivateCooling = TRUE), define: • a positive substitute output value to output the value at the outputs for heating • a negative substitute output value to output the value at the outputs for cooling The permitted value range is determined by the configuration. • Cooling output deactivated (Config.ActivateCooling = FALSE): Config.Output.Heat.PidUpperLimit ≥ SubstituteOutput ≥ Config.Output.Heat.PidLowerLimit • Cooling output activated (Config.ActivateCooling = TRUE): Config.Output.Heat.PidUpperLimit ≥ SubstituteOutput ≥ Config.Output.Cool.PidLowerLimit
PidOutputSum	REAL	0.0	PID output value PidOutputSum displays the output value of the PID algorithm. Depending on the operating mode, it is either calculated automatically or defined by the manual value or the configured substitute output value. The values at the outputs for heating and cooling resulting from the PID output value are the result of the configured output scaling (structures Config.Output.Heat and Config.Output.Cool). The PidOutputSum is limited as defined in the configuration. • Cooling output deactivated (Config.ActivateCooling = FALSE): Config.Output.Heat.PidUpperLimit ≥ PidOutputSum ≥ Config.Output.Heat.PidLowerLimit • Cooling output activated (ConfigActivateCooling = TRUE): Config.Output.Heat.PidUpperLimit ≥ PidOutputSum ≥ Config.Output.Cool.PidLowerLimit

Tag	Data type	Default	Description
PidOutputOffsetHeat	REAL	0.0	Offset of the PID output value heating PidOutputOffsetHeat is added to the value that results from PidOutputSum for the heating branch. Enter a positive value for PidOutputOffsetHeat to receive a positive offset at the outputs for heating. The resulting values at the outputs for heating are the result of the configured output scaling (Config.Output.Heat structure). This offset can be used for actuators which need a fixed minimum value, for example, fans with minimum speed.
PidOutputOffsetCool	REAL	0.0	Offset of the PID output value cooling PidOutputOffsetCool is added to the value that results from PidOutputSum for the cooling branch. Enter a negative value for PidOutputOffsetCool to receive a positive offset at the outputs for cooling. The resulting values at the outputs for cooling are the result of the configured output scaling (Config.Output.Cool structure). This offset can be used for actuators which need a fixed minimum value, for example, fans with minimum speed.
SubstituteSetpointOn	BOOL	FALSE	Activates the substitute setpoint as controller setpoint. • FALSE = the Setpoint parameter is used. • TRUE = the SubstituteSetpoint parameter is used as setpoint SubstituteSetpointOn can be used to specify the setpoint of a slave controller in a cascade directly without having to change the user program.
SubstituteSetpoint	REAL	0.0	Substitute setpoint If SubstituteSetpointOn = TRUE, the SubstituteSetpoint parameter is used as setpoint. Valid range of values: Config.SetpointUpperLimit ≥ SubstituteSetpoint ≥ Config.SetpointLowerLimit, Config.InputUpperLimit ≥ SubstituteSetpoint ≥ Config.InputLowerLimit
DisableCooling	BOOL	FALSE	DisableCooling = TRUE deactivates the cooling branch for heating/cooling controllers (Config.ActivateCooling = TRUE) in Automatic mode by setting PidOutputSum to 0.0 as low limit. PidOutputOffsetCool and the output scaling for the cooling outputs remain active. DisableCooling can be used for tuning of multi-zone applications to temporarily deactivate the cooling branch as long as all controllers have not completed their tuning yet. This parameter is set/reset by the user manually and is not automatically reset by the PID_Temp instruction.

Tag	Data type	Default	Description
AllSlaveAutomaticState	BOOL	FALSE	If this PID_Temp instance is used as master controller in a cascade (Config.Cascade.IsMaster = TRUE), AllSlaveAutomaticState = TRUE indicates that all slave controllers are in automatic mode. Tuning, manual mode or automatic mode of the master controller can only be executed accurately if all slave controllers are in automatic mode. AllSlaveAutomaticState is only determined if you interconnect the master controller and slave controller with the Master and Slave parameters. For details, see the Master parameter.
NoSlaveSubstituteSetpoint	BOOL	FALSE	If this PID_Temp instance is used as master controller in a cascade (Config.Cascade.IsMaster = TRUE), NoSlaveSubstituteSetpoint = TRUE indicates that no slave controller has activated its substitute setpoint. Tuning, manual mode or automatic mode of the master controller can only be executed accurately if no slave controller has activated its substitute setpoint. NoSlaveSubstituteSetpoint is only determined if you interconnect the master controller and slave controller with the Master and Slave parameters. For details, see the Master parameter.
Heat.EnableTuning	BOOL	TRUE	Enabling of tuning for heating Heat.EnableTuning must be set for the following tunings (at the same time or prior to the start with Mode and ModeActivate): • Pretuning heating • Pretuning heating and cooling • Fine tuning heating This parameter is not automatically reset by the PID_Temp instruction.
Cool.EnableTuning	BOOL	FALSE	Enabling of tuning for cooling Cool.EnableTuning must be set for the following tunings (simultaneously with or prior to the start with Mode and ModeActivate): • Pretuning cooling • Pretuning heating and cooling • Fine tuning cooling Only effective if the cooling output and PID parameter switching are activated ("Config.ActivateCooling" = TRUE and "Config.AdvancedCooling" = TRUE). This parameter is not automatically reset by the PID_Temp instruction.
Config.InputPerOn	BOOL	TRUE	If InputPerOn = TRUE, the Input_PER parameter is used for detecting the process value. If InputPerOn = FALSE, the Input parameter is used.

Tag	Data type	Default	Description
Config.InputUpperLimit	REAL	120.0	High limit of the process value Input and Input_PER are monitored to ensure adherence to this limit. If the limit is exceeded, an error is output and the reaction is determined by ActivateRecoverMode. At the I/O input, the process value can be a maximum of 18% higher than the nominal range (overrange). This means the limit cannot be exceeded when you use an I/O input with the pre-setting for high limit and process value scaling. When pretuning is started, the difference between high and low limit of the process value is checked to determine whether the distance between setpoint and process value meets the necessary requirements. InputUpperLimit > InputLowerLimit
Config.InputLowerLimit	REAL	0.0	Low limit of the process value Input and Input_PER are monitored to ensure adherence to this limit. If the limit is undershot, an error is output and the reaction is determined by ActivateRecoverMode. InputLowerLimit < InputUpperLimit
Config.InputUpperWarning	REAL	3.402822e+38	Warning high limit of the process value Input and Input_PER are monitored to ensure adherence to this limit. If the limit is exceeded, a warning is output at the Warning parameter. • If you set InputUpperWarning outside the process value limits, the configured absolute process value high limit is used as the warning high limit. • If you configure InputUpperWarning within the process value limits, this value is used as the warning high limit. InputUpperWarning > InputLowerWarning
Config.InputLowerWarning	REAL	-3.402822e+38	Warning low limit of the process value Input and Input_PER are monitored to ensure adherence to this limit. If the limit is undershot, a warning is output at the Warning parameter. • If you set InputLowerWarning outside the process value limits, the configured absolute process value low limit is used as the warning low limit. • If you configure InputLowerWarning within the process value limits, this value is used as the warning low limit. InputLowerWarning < InputUpperWarning

Tag	Data type	Default	Description
Config.SetpointUpperLimit	REAL	3.402822e +38	High limit of setpoint Setpoint and SubstituteSetpoint are monitored to ensure adherence to this limit. If the limit is exceeded, a warning is output at the Warning parameter. • If you configure SetpointUpperLimit outside the process value limits, the configured absolute process value high limit is used as the setpoint high limit. • If you configure SetpointUpperLimit within the process value limits, this value is used as the setpoint high limit. SetpointUpperLimit > SetpointLowerLimit
Config.SetpointLowerLimit	REAL	-3.402822e +38	Low limit of the setpoint Setpoint and SubstituteSetpoint are monitored to ensure adherence to this limit. If the limit is undershot, a warning is output at the Warning parameter. • If you set SetpointLowerLimit outside the process value limits, the configured process value absolute low limit is used as the setpoint low limit. • If you configure SetpointLowerLimit within the process value limits, this value is used as the setpoint low limit. SetpointLowerLimit < SetpointUpperLimit
Config.ActivateCooling	BOOL	FALSE	Activate cooling output • Config.ActivateCooling = FALSE Only the outputs for heating are used. • Config.ActivateCooling = TRUE The outputs for heating and cooling are used. If you are using the cooling output, the controller must not be configured as master controller (Config.Cascade.IsMaster must be FALSE) .

Tag	Data type	Default	Description
Config.AdvancedCooling	BOOL	TRUE	Method for heating/cooling • Cooling factor (Config.AdvancedCooling = FALSE) The output value calculation for cooling takes place with the PID parameters for heating (Retain.CtrlParams.Heat structure) taking into consideration the configurable cooling factor Config.CoolFactor. This method is suitable if the heating and cooling actuators have a similar time response but different gains. Pretuning and fine tuning for cooling are not available when you select this method. You can only execute the tuning for heating. • PID parameter switching (Config.AdvancedCooling = TRUE) The output value calculation for cooling takes place by means of a separate PID parameter set (Retain.CtrlParams.Cool structure). This method is suitable if the heating and cooling actuator have different time responses and different gains. Pretuning and fine tuning for cooling are only available when you select this method (Mode = 1 or 2, Cool.EnableTuning = TRUE). Config.AdvancedCooling is only calculated if the cooling output is activated (Config.ActivateCooling = TRUE).
Config.CoolFactor	REAL	1.0	Cooling factor If Config.AdvancedCooling = FALSE, Config.CoolFactor is considered as factor in the calculation of the output value for cooling. This allows different gains of heating and cooling actuators to be taken into account. Config.CoolFactor is not set automatically or adjusted during tuning. You must correctly configure Config.CoolFactor manually with the ratio "heating actuator gain/cooling actuator gain". Example: Config.CoolFactor = 2.0 means that the gain of the heating actuator is twice as high as the gain of the cooling actuator. Config.CoolFactor is only effective if the cooling output is activated (Config.ActivateCooling = TRUE) and cooling factor is selected as method for heating/cooling (Config.AdvancedCooling = FALSE). Config.CoolFactor > 0.0
Config.InputScaling.UpperPointIn	REAL	27648.0	Scaling Input_PER high Input_PER is scaled based on the two value pairs UpperPointOut, UpperPointIn and LowerPointOut, LowerPointIn. Only effective if Input_PER is used for process value detection (Config.InputPerOn = TRUE). UpperPointIn > LowerPointIn

Tag	Data type	Default	Description
Config.InputScaling.LowerPointIn	REAL	0.0	Scaling Input_PER low Input_PER is scaled based on the two value pairs UpperPoint-Out, UpperPointIn and LowerPointOut, LowerPointIn. Only effective if Input_PER is used for process value detection (Config.InputPerOn = TRUE). LowerPointIn < UpperPointIn
Config.InputScaling.UpperPoint-Out	REAL	100.0	Scaled high process value Input_PER is scaled based on the two value pairs UpperPoint-Out, UpperPointIn and LowerPointOut, LowerPointIn. Only effective if Input_PER is used for process value detection (Config.InputPerOn = TRUE). UpperPointOut > LowerPointOut
Config.InputScaling.LowerPoint-Out	REAL	0.0	Scaled low process value Input_PER is scaled based on the two value pairs UpperPoint-Out, UpperPointIn and LowerPointOut, LowerPointIn. Only effective if Input_PER is used for process value detection (Config.InputPerOn = TRUE). LowerPointOut < UpperPointOut
Config.Output.Heat.Select	INT	1	Selecting the output value for heating Config.Output.Heat.Select specifies which outputs are used for heating: • Heat.Select = 0 - OutputHeat is used • Heat.Select = 1 - OutputHeat and OutputHeat_PWM are used • Heat.Select = 2 -OutputHeat and OutputHeat_PER are used Outputs that are not used are not calculated and remain at their default value.

Tag	Data type	Default	Description
Config.Output.Heat.PwmPeriode	REAL	0.0	Period duration of the pulse width modulation (PWM) for heating (OutputHeat_PWM output) in seconds: • Heat.PwmPeriode = 0.0 The sampling time of the PID algorithm for heating (Retain.CtrlParams.Heat.Cycle) is used as period duration of the PWM. • Heat.PwmPeriode > 0.0 The value is rounded off to an integer multiple of the PID_Temp sampling time (CycleTime.Value) and used as period duration of the PWM. This setting can be used to improve the smoothing of the process value with a long sampling time of the PID algorithm. The value must meet the following conditions: – Heat.PwmPeriode ≤ Retain.CtrlParams.Heat.Cycle, – Heat.PwmPeriode > Config.Output.Heat.MinimumOnTime – Heat.PwmPeriode > Config.Output.Heat.MinimumOffTime
Config.Output.Heat.PidUpperLimit	REAL	100.0	High limit of the PID output value for heating The PID output value (PidOutputSum) is limited to the high limit. Heat.PidUpperLimit forms a value pair together with the following parameters for scaling of the PID output value (PidOutputSum) to the outputs for heating: • Heat.UpperScaling for OutputHeat • Heat.PwmUpperScaling for OutputHeat_PWM • Heat.PerUpperScaling for OutputHeat_PER If you want to limit the value at the associated output, you must also adjust these scaling values. Heat.PidUpperLimit > Heat.PidLowerLimit

Tag	Data type	Default	Description
Config.Output.Heat.PidLowerLimit	REAL	0.0	Low limit of the PID output value for heating For controllers with **deactivated** cooling output (Config.ActivateCooling = FALSE), the PID output value (PidOutputSum) is limited to this low limit. For controllers with **activated** cooling output (Config.ActivateCooling = TRUE), the value must be 0.0. Heat.PidLowerLimit forms a value pair together with the following parameters for scaling of the PID output value (PidOutputSum) to the outputs for heating: • Heat.LowerScaling for OutputHeat • Heat.PwmLowerScaling for OutputHeat_PWM • Heat.PerLowerScaling for OutputHeat_PER If you want to limit the value at the associated output, you must also adjust these scaling values. The permitted value range is determined by the configuration. • Cooling output deactivated (Config.ActivateCooling = FALSE): Heat.PidLowerLimit < Heat.PidUpperLimit • Cooling output activated (Config.ActivateCooling = TRUE): Heat.PidLowerLimit = 0.0
Config.Output.Heat.UpperScaling	REAL	100.0	Scaled high output value for heating Heat.UpperScaling and Heat.PidUpperLimit form a value pair for scaling of the PID output value (PidOutputSum) to the output value for heating (OutputHeat). The OutputHeat value is always located between Heat.UpperScaling and Heat.LowerScaling. Heat.UpperScaling ≠ Heat.LowerScaling
Config.Output.Heat.LowerScaling	REAL	0.0	Scaled low output value for heating Heat.LowerScaling and Heat.PidLowerLimit form a value pair for scaling of the PID output value (PidOutputSum) to the output value for heating (OutputHeat). The OutputHeat value is always located between Heat.UpperScaling and Heat.LowerScaling. Heat.UpperScaling ≠ Heat.LowerScaling
Config.Output.Heat.PwmUpperScaling	REAL	100.0	Scaled high PWM output value for heating Heat.PwmUpperScaling and Heat.PidUpperLimit form a value pair for scaling of the PID output value (PidOutputSum) to the pulse-width modulated output value for heating (OutputHeat_PWM). The OutputHeat_PWM value is always located between Heat.PwmUpperScaling and Heat.PWMLowerScaling. Heat.PwmUpperScaling is only effective if OutputHeat_PWM is selected as output for heating (Heat.Select = 1) 100.0 ≥ Heat.PwmUpperScaling ≥ 0.0 Heat.PwmUpperScaling ≠ Heat.PwmLowerScaling

Tag	Data type	Default	Description
Config.Output.Heat.PwmLowerScaling	REAL	0.0	Scaled low PWM output value for heating Heat.PwmLowerScaling and Heat.PidLowerLimit form a value pair for scaling of the PID output value (PidOutputSum) to the pulse-width modulated output value for heating (OutputHeat_PWM). The OutputHeat_PWM value is always located between Heat.PwmUpperScaling and Heat.PwmLowerScaling. Heat.PwmLowerScaling is only effective if OutputHeat_PWM is selected as output for heating (Heat.Select = 1) 100.0 ≥ Heat.PwmLowerScaling ≥ 0.0 Heat.PwmUpperScaling ≠ Heat.PwmLowerScaling
Config.Output.Heat.PerUpperScaling	REAL	27648.0	Scaled high analog output value for heating Heat.PerUpperScaling and Heat.PidUpperLimit form a value pair for scaling of the PID output value (PidOutputSum) to the analog output value for heating (OutputHeat_PER). The OutputHeat_PER value is always located between Heat.PerUpperScaling and Heat.PerLowerScaling. Heat.PerUpperScaling is only effective if OutputHeat_PER is selected as output for heating (Heat.Select = 2) 32511.0 ≥ Heat.PerUpperScaling ≥ -32512.0 Heat.PerUpperScaling ≠ Heat.PerLowerScaling
Config.Output.Heat.PerLowerScaling	REAL	0.0	Scaled low analog output value for heating Heat.PerLowerScaling and Heat.PidLowerLimit form a value pair for scaling of the PID output value (PidOutputSum) to the analog output value for heating (OutputHeat_PER). The OutputHeat_PER value is always located between Heat.PerUpperScaling and Heat.PerLowerScaling. Heat.PerLowerScaling is only effective if OutputHeat_PER is selected as output for heating (Heat.Select = 2) 32511.0 ≥ Heat.PerLowerScaling ≥ -32512.0 Heat.PerUpperScaling ≠ Heat.PerLowerScaling
Config.Output.Heat.MinimumOnTime	REAL	0.0	Minimum on time of the pulse width modulation for heating (OutputHeat_PWM output) A PWM pulse is never shorter than this value. The value is rounded off to: Heat.MinimumOnTime = n × CycleTime.Value Heat.MinimumOnTime is only effective if the output for heating OutputHeat_PWM is selected (Heat.Select = 1)". 100000.0 ≥ Heat.MinimumOnTime ≥ 0.0

Tag	Data type	Default	Description
Config.Output.Heat.MinimumOff-Time	REAL	0.0	Minimum off time of the pulse width modulation for heating (OutputHeat_PWM output) A PWM pause is never shorter than this value. The value is rounded off to: Heat.MinimumOffTime = n × CycleTime.Value Heat.MinimumOffTime is only effective if the output for heating OutputHeat_PWM is selected (Heat.Select = 1)". 100000.0 ≥ Heat.MinimumOffTime ≥ 0.0
Config.Output.Cool.Select	INT	1	Selecting the output value for cooling Config.Output.Cool.Select specifies which outputs are used for cooling: • Cool.Select = 0 - OutputCool is used • Cool.Select = 1 -OutputCool and OutputCool_PWM are used • Cool.Select = 2 - OutputCool and OutputCool_PER are used Outputs that are not used are not calculated and remain at their default value. Only effective if the cooling output is activated (Config.Activa-teCooling = TRUE).

Tag	Data type	Default	Description
Config.Output.Cool.PwmPeriode	REAL	0.0	Period duration of the pulse width modulation for cooling (OutputCool_PWM output) in seconds: • Cool.PwmPeriode = 0.0 and Config.AdvancedCooling = FALSE: sampling time of the PID algorithm for heating (Retain.CtrlParams.Heat.Cycle) is used as period duration of the PWM. • Cool.PwmPeriode = 0.0 and Config.AdvancedCooling = TRUE: The sampling time of the PID algorithm for cooling (Retain.CtrlParams.Cool.Cycle) is used as period duration of the PWM. • Cool.PwmPeriode > 0.0: The value is rounded off to an integer multiple of the PID_Temp sampling time (CycleTime.Value) and used as period duration of the PWM. This setting can be used to improve the smoothing of the process value with a long sampling time of the PID algorithm. The value must meet the following conditions: – Cool.PwmPeriode ≤ Retain.CtrlParams.Cool.Cycle or Retain.CtrlParams.Heat.Cycle – Cool.PwmPeriode > Config.Output.Cool.MinimumOnTime – Cool.PwmPeriode > Config.Output.Cool.MinimumOffTime Only effective if the cooling output is activated (Config.ActivateCooling = TRUE).
Config.Output.Cool.PidUpperLimit	REAL	0.0	High limit of the PID output value for cooling The value must be 0.0. Cool.PidUpperLimit forms a value pair together with the following parameters for scaling of the PID output value (PidOutputSum) to the outputs for cooling: • Cool.LowerScaling for OutputCool • Cool.PwmLowerScaling for OutputCool_PWM • Cool.PerLowerScaling for OutputCool_PER If you want to limit the value at the associated output, you must also adjust these scaling values. Only effective if the cooling output is activated (Config.ActivateCooling = TRUE). Cool.PidUpperLimit = 0.0

Tag	Data type	Default	Description
Config.Output.Cool.PidLowerLimit	REAL	-100.0	Low limit of the PID output value for cooling For controllers with activated cooling output (Config.ActivateCooling = TRUE), the PID output value (PidOutputSum) is limited to this low limit. Cool.PidLowerLimit forms a value pair together with the following parameters for scaling of the PID output value (PidOutputSum) to the outputs for cooling: • Cool.UpperScaling for OutputCool • Cool.PwmUpperScaling for OutputCool_PWM • Cool.PerUpperScaling for OutputCool_PER If you want to limit the value at the associated output, you must also adjust these scaling values. Only effective if the cooling output is activated (Config.ActivateCooling = TRUE). Cool.PidLowerLimit < Cool.PidUpperLimit
Config.Output.Cool.UpperScaling	REAL	100.0	Scaled high output value for cooling Cool.UpperScaling and Cool.PidLowerLimit form a value pair for scaling of the PID output value (PidOutputSum) to the output value for cooling (OutputCool). The OutputCool value is always located between Cool.UpperScaling and Cool.LowerScaling. Only effective if the cooling output is activated (Config.ActivateCooling = TRUE). Cool.UpperScaling ≠ Cool.LowerScaling
Config.Output.Cool.LowerScaling	REAL	0.0	Scaled low output value for cooling Cool.LowerScaling and Cool.PidUpperLimit form a value pair for scaling of the PID output value (PidOutputSum) to the output value for cooling (OutputCool). The OutputCool value is always located between Cool.UpperScaling and Cool.LowerScaling. Only effective if the cooling output is activated (Config.ActivateCooling = TRUE). Cool.UpperScaling ≠ Cool.LowerScaling
Config.Output.Cool.PwmUpperScaling	REAL	100.0	Scaled high PWM output value for cooling Cool.PwmUpperScaling and Cool.PidLowerLimit form a value pair for scaling of the PID output value (PidOutputSum) to the pulse-width modulated output value for cooling (OutputCool_PWM). The OutputCool_PWM value is always located between Cool.PwmUpperScaling and Cool.PwmLowerScaling. Cool.PwmUpperScaling is only effective if the cooling output is activated (Config.ActivateCooling = TRUE) and OutputCool_PWM is selected as output for cooling (Cool.Select = 1). 100.0 ≥ Cool.PwmUpperScaling ≥ 0.0 Cool.PwmUpperScaling ≠ Cool.PwmLowerScaling

Tag	Data type	Default	Description
Config.Output.Cool.PwmLowerScaling	REAL	0.0	Scaled low PWM output value for cooling Cool.PwmLowerScaling and Cool.PidUpperLimit form a value pair for scaling of the PID output value (PidOutputSum) to the pulse-width modulated output value for cooling (OutputCool_PWM). The OutputCool_PWM value is always located between Cool.PwmUpperScaling and CoolPwm.LowerScaling. Cool.PwmLowerScaling is only effective if the cooling output is activated (Config.ActivateCooling = TRUE) and OutputCool_PWM is selected as output for cooling (Cool.Select = 1). 100.0 ≥ Cool.PwmLowerScaling ≥ 0.0 Cool.PwmUpperScaling ≠ Cool.PwmLowerScaling
Config.Output.Cool.PerUpperScaling	REAL	27648.0	Scaled high analog output value for cooling Cool.PerUpperScaling and Cool.PidLowerLimit form a value pair for scaling of the PID output value (PidOutputSum) to the analog output value for cooling (OutputCool_PER). The OutputCool_PER value is always located between Cool.PerUpperScaling and Cool.PerLowerScaling. Cool.PerUpperScaling is only effective if the cooling output is activated (Config.ActivateCooling = TRUE) and OutputCool_PER is selected as output for cooling (Cool.Select = 2). 32511.0 ≥ Cool.PerUpperScaling ≥ -32512.0 Cool.PerUpperScaling ≠ Cool.PerLowerScaling
Config.Output.Cool.PerLowerScaling	REAL	0.0	Scaled low analog output value for cooling Cool.PerLowerScaling and Cool.PidUpperLimit form a value pair for scaling of the PID output value (PidOutputSum) to the analog output value for cooling (OutputCool_PER). The OutputCool_PER value is always located between Cool.PerUpperScaling and Cool.PerLowerScaling. Cool.PerLowerScaling is only effective if the cooling output is activated (Config.ActivateCooling = TRUE) and OutputCool_PER is selected as output for cooling (Cool.Select = 2). 32511.0 ≥ Cool.PerLowerScaling ≥ -32512.0 Cool.PerUpperScaling ≠ Cool.PerLowerScaling
Config.Output.Cool.MinimumOnTime	REAL	0.0	Minimum on time of the pulse width modulation for cooling (OutputCool_PWM output) A PWM pulse is never shorter than this value. The value is rounded off to: Cool.MinimumOnTime = n × CycleTime.Value Cool.MinimumOnTime is only effective if the output for cooling OutputCool_PWM is selected (Cool.Select = 1). Only effective if the cooling output is activated (Config.ActivateCooling = TRUE). 100000.0 ≥ Cool.MinimumOnTime ≥ 0.0

Tag	Data type	Default	Description
Config.Output.Cool.MinimumOff-Time	REAL	0.0	Minimum off time of the pulse width modulation for cooling (OutputCool_PWM output) A PWM pause is never shorter than this value. The value is rounded off to: Cool.MinimumOffTime = n × CycleTime.Value Cool.MinimumOffTime is only effective if the output for cooling OutputCool_PWM is selected (Cool.Select = 1). Only effective if the cooling output is activated (Config.Activa-teCooling = TRUE). 100000.0 ≥ Cool.MinimumOffTime ≥ 0.0
If you are using PID_Temp in a cascade, the master controller and slave controller exchange information via the Master and Slave parameters. You need to make the interconnection. For details, see the Master parameter.			
Config.Cascade.IsMaster	BOOL	FALSE	The controller is master in a cascade and provides the slave setpoint. Set IsMaster = TRUE if you are using this PID_Temp instance as master controller in a cascade. A master controller defines the setpoint of a slave controller with its output. A PID_Temp instance can be master controller and slave controller at the same time. If the controller is used as master controller, the cooling output must be deactivated (Config.ActivateCooling = FALSE).
Config.Cascade.IsSlave	BOOL	FALSE	The controller is slave in a cascade and receives its setpoint from the master. Set IsSlave = TRUE if you are using this PID_Temp instance as slave controller in a cascade. A slave controller receives its setpoint (Setpoint parameter) from the output of its master controller (OutputHeat parame-ter). A PID_Temp instance can be master controller and slave controller at the same time.
Config.Cascade.AntiWindUpMode	INT	1	Anti-windup behavior in the cascade Options are: • Anti-windup = 0 The AntiWindUp functionality is deactivated. The master controller does not respond to the limit of its slave controllers. • Anti-windup = 1 The integral action of the master controller is reduced in the ratio "Slaves in limit" to "Number of slaves" ("CountSlaves" parameter). This reduces the effects of the limitation on the control behavior. • Anti-windup = 2 The integral action of the master controller is held as soon as a slave controller is in the limit. Only effective if the controller is configured as master controller (Config.Cascade.IsMaster = TRUE).

Tag	Data type	Default	Description
Config.Cascade.CountSlaves	INT	1	Number of subordinate slaves Here you enter the number of directly subordinate slave controllers which receive their setpoint from this master controller. Only effective if the controller is configured as master controller (Config.Cascade.IsMaster = TRUE). 255 ≥ CountSlaves ≥ 1
CycleTime.StartEstimation	BOOL	TRUE	If CycleTime.EnEstimation = TRUE, CycleTime.StartEstimation = TRUE starts automatic determination of the PID_Temp sampling time (cycle time of the calling OB). CycleTime.StartEstimation = FALSE is set once measurement is complete.
CycleTime.EnEstimation	BOOL	TRUE	If CycleTime.EnEstimation = TRUE, the PID_Temp sampling time is determined automatically. If CycleTime.EnEstimation = FALSE, the sampling time PID_Temp is not determined automatically and must be configured correctly manually with CycleTime.Value.
CycleTime.EnMonitoring	BOOL	TRUE	If CycleTime.EnMonitoring = FALSE, the PID_Temp sampling time is not monitored. If PID_Temp cannot be executed within the sampling time, no error (ErrorBits=0000800h) is output and PID_Temp does not respond as configured with ActivateRecoverMode.
CycleTime.Value	REAL	0.1	PID_Temp sampling time (cycle time of the calling OB) in seconds CycleTime.Value is determined automatically and is usually equivalent to the cycle time of the calling OB.
You can reload values from the CtrlParamsBackUp structure with LoadBackUp = TRUE.			
CtrlParamsBackUp.SetByUser	BOOL	FALSE	Saved value of Retain.CtrlParams.SetByUser
CtrlParamsBackUp.Heat.Gain	REAL	1.0	Saved proportional gain for heating
CtrlParamsBackUp.Heat.Ti	REAL	20.0	Saved integral action time for heating in seconds
CtrlParamsBackUp.Heat.Td	REAL	0.0	Saved derivative action time for heating in seconds
CtrlParamsBackUp.Heat.TdFiltRatio	REAL	0.2	Saved derivative delay coefficient for heating
CtrlParamsBackUp.Heat.PWeighting	REAL	1.0	Saved weighting of the proportional action for heating
CtrlParamsBackUp.Heat.DWeighting	REAL	1.0	Saved weighting of the derivative action for heating
CtrlParamsBackUp.Heat.Cycle	REAL	1.0	Saved sampling time of the PID algorithm for heating in seconds
CtrlParamsBackUp.Heat.ControlZone	REAL	3.402822e+38	Saved control zone width for heating
CtrlParamsBackUp.Heat.DeadZone	REAL	0.0	Saved dead zone width for heating
CtrlParamsBackUp.Cool.Gain	REAL	1.0	Saved proportional gain for cooling
CtrlParamsBackUp.Cool.Ti	REAL	20.0	Saved integral action time for cooling in seconds
CtrlParamsBackUp.Cool.Td	REAL	0.0	Saved derivative action time for cooling in seconds
CtrlParamsBackUp.Cool.TdFiltRatio	REAL	0.2	Saved derivative delay coefficient for cooling

Tag	Data type	Default	Description
CtrlParamsBackUp.Cool.PWeighting	REAL	1.0	Saved proportional action weighting factor for cooling
CtrlParamsBackUp.Cool.DWeighting	REAL	1.0	Saved derivative action weighting factor for cooling
CtrlParamsBackUp.Cool.Cycle	REAL	1.0	Saved sampling time of the PID algorithm for cooling in seconds
CtrlParamsBackUp.Cool.ControlZone	REAL	3.402822e+38	Saved control zone width for cooling
CtrlParamsBackUp.Cool.DeadZone	REAL	0.0	Saved dead zone width for cooling
PIDSelfTune.SUT.CalculateParamsHeat	BOOL	FALSE	The properties of the heating branch of the controlled system are saved during pretuning for heating. If SUT.CalculateParamsHeat = TRUE, the PID parameters for heating (Retain.CtrlParams.Heat structure) are recalculated on the basis of these properties. This enables you to change the parameter calculation method (PIDSelfTune.SUT.TuneRuleHeat parameter) without having to repeat the tuning. SUT.CalculateParamsHeat is set to FALSE after the calculation. Only possible if the pretuning was successful (SUT.ProcParHeatOk = TRUE).
PIDSelfTune.SUT.CalculateParamsCool	BOOL	FALSE	The properties of the cooling branch of the controlled system are saved during tuning for cooling. If SUT.CalculateParamsCool = TRUE, the PID parameters for cooling (Retain.CtrlParams.Cool structure) are recalculated on the basis of these properties. This enables you to change the parameter calculation method (PIDSelfTune.SUT.TuneRuleCool parameter) without having to repeat the tuning. SUT.CalculateParamsCool is set to FALSE after the calculation. Only possible if the pretuning was successful (SUT.ProcParCoolOk = TRUE). Only effective if Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE.
PIDSelfTune.SUT.TuneRuleHeat	INT	2	Method for PID parameter calculation with pretuning for heating Options are: • SUT.TuneRuleHeat = 0: PID according to CHR • SUT.TuneRuleHeat = 1: PI according to CHR • SUT.TuneRuleHeat = 2: PID for temperature processes according to CHR (results in a slower and rather asymptomatic control response with lower overshoot than SUT.TuneRuleHeat = 0) (CHR = Chien, Hrones and Reswick) Only with SUT.TuneRuleHeat = 2 is the control zone Retain.CtrlParams.Heat.ControlZone automatically set during pretuning for heating.

Tag	Data type	Default	Description
PIDSelfTune.SUT.TuneRuleCool	INT	2	Method for PID parameter calculation with pretuning for cooling Options are: • SUT.TuneRuleCool = 0: PID according to CHR • SUT.TuneRuleCool = 1: PI according to CHR • SUT.TuneRuleCool = 2: PID for temperature processes according to CHR (results in a slower and rather asymptomatic control response with lower overshoot than SUT.TuneRuleCool = 0) (CHR = Chien, Hrones and Reswick) Only with SUT.TuneRuleCool = 2 is the control zone Retain.CtrlParams.Cool.ControlZone automatically set during pretuning for cooling. SUT.TuneRuleCool is only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE).
PIDSelfTune.SUT.State	INT	0	The SUT.State tag indicates the current phase of pretuning: • State = 0: Initialize pretuning • State = 100: Calculate standard deviation for heating • State = 200: Calculate standard deviation for cooling • State = 300: Determine point of inflection for heating • State = 400: Determine point of inflection for cooling • State = 500: Set heating to setpoint after reaching point of inflection • State = 600: Set cooling to setpoint after reaching point of inflection • State = 700: Compare efficiency of the heating actuator and cooling actuator • State = 800: Heating and cooling activated • State = 900: Cooling activated • State = 1000: Determine delay time after switching off heating • State = 9900: Pretuning successful • State = 1: Pretuning not successful
PIDSelfTune.SUT.ProcParHeatOk	BOOL	FALSE	TRUE: The calculation of the process parameters for pretuning heating was successful. This tag is set during tuning. It must be TRUE for calculation of the PID parameters for heating.
PIDSelfTune.SUT.ProcParCoolOk	BOOL	FALSE	TRUE: The calculation of the process parameters for pretuning cooling was successful. This tag is set during tuning. It must be TRUE for calculation of the PID parameters for cooling.

Tag	Data type	Default	Description
PIDSelfTune.SUT.AdaptDelay-Time	INT	0	The AdaptDelayTime tag determines the adaptation of the delay time for heating at the operating point (for "Pretuning heating" and "Pretuning heating and cooling"). Options are: • SUT.AdaptDelayTime = 0: No adaptation of delay time. The SUT.State = 1000 phase is skipped. This option results in a shorter tuning time than with SUT.AdaptDelayTime = 1. • SUT.AdaptDelayTime = 1: Adaptation of the delay time to the setpoint in SUT.State = 1000 phase by switching off heating temporarily. This option results in a longer tuning time than withSUT.AdaptDelayTime = 0. It can improve the control response if the process behavior depends significantly on the operating point (non-linearity). This option should not be used for multi-zone applications with strong thermal connections.
PIDSelfTune.SUT.CoolingMode	INT	0	The CoolingMode tag determines the manipulated variable output to determine the cooling parameters (for pretuning heating and cooling). Options are: • SUT.CoolingMode = 0: Switch off heating and switch on cooling after reaching the setpoint. The SUT.State = 700 phase is skipped. Phase SUT.State = 500 is followed by phase SUT.State = 900. This option can improve the control response if the gain of the cooling actuator is low compared to the gain of the heating actuator. It results in a shorter tuning time than with SUT.CoolingMode = 1 or 2. • SUT.CoolingMode = 1: Switch on cooling in addition to heating after reaching the setpoint. The SUT.State = 700 phase is skipped. Phase SUT.State = 500 is followed by phase SUT.State = 800. This option can improve the control response if the gain of the cooling actuator is high compared to the gain of the heating actuator. • SUT.CoolingMode = 2: After heating up to the setpoint, a decision is automatically made in phase SUT.State = 700 as to whether heating is switched off. Phase SUT.State = 500 is followed by phase SUT.State = 700 and then SUT.State = 800 or SUT.State = 900. This option requires more time than options 0 and 1.

Tag	Data type	Default	Description
PIDSelfTune.TIR.RunIn	BOOL	FALSE	Use the RunIn tag to specify the sequence of fine tuning during start from automatic mode. • RunIn = FALSE If fine tuning is started from automatic mode, the system uses the existing PID parameters to control to the setpoint (TIR.State = 500 or 600). Only then will fine tuning start. • RunIn = TRUE PID_Temp tries to reach the setpoint with minimum or maximum output value (TIR.State = 300 or 400). This can produce increased overshoot. Fine tuning then starts automatically. RunIn is set to FALSE after fine tuning. During start of fine tuning from Inactive or Manual mode, PID_Temp reacts as described under RunIn = TRUE.
PIDSelfTune.TIR.CalculateParamsHeat	BOOL	FALSE	The properties of the heating branch of the controlled system are saved during fine tuning for heating. If TIR.CalculateParamsHeat= TRUE, the PID parameters for heating (Retain.CtrlParams.Heat structure) are recalculated on the basis of these properties. This enables you to change the parameter calculation method (PIDSelfTune.TIR.TuneRuleHeat parameter) without having to repeat the tuning. TIR.CalculateParamsHeat is set to FALSE after the calculation. Only possible if fine tuning heating was successful beforehand (TIR.ProcParHeatOk = TRUE).
PIDSelfTune.TIR.CalculateParamsCool	BOOL	FALSE	The properties of the cooling branch of the controlled system are saved during fine tuning for cooling. If TIR.CalculateParamsCool= TRUE, the PID parameters for cooling (Retain.CtrlParams.Cool structure) are recalculated on the basis of these properties. This enables you to change the parameter calculation method (PIDSelfTune.TIR.TuneRuleCool parameter) without having to repeat the tuning. TIR.CalculateParamsCool is set to FALSE after the calculation. Only possible if fine tuning cooling was successful beforehand (TIR.ProcParCoolOk = TRUE). Only effective if Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE

Tag	Data type	Default	Description
PIDSelfTune.TIR.TuneRuleHeat	INT	0	Method for parameter calculation during fine tuning for heating Options are: • TIR.TuneRuleHeat = 0: PID automatic • TIR.TuneRuleHeat = 1: PID fast (faster control response with higher amplitudes of the output value than with TIR.TuneRuleHeat = 2) • TIR.TuneRuleHeat = 2: PID slow (slower control response with lower amplitudes of the output value than with TIR.TuneRuleHeat = 1) • TIR.TuneRuleHeat = 3: ZN PID • TIR.TuneRuleHeat = 4: ZN PI • TIR.TuneRuleHeat = 5: ZN P (ZN=Ziegler-Nichols) To be able to repeat the calculation of the PID parameters for heating with TIR.CalculateParamsHeat and TIR.TuneRuleHeat = 0, 1 or 2, the previous fine tuning also has to have been executed with TIR.TuneRuleHeat = 0, 1 or 2. If this is not the case, TIR.TuneRuleHeat = 3 is used. The recalculation of the PID parameters for heating with TIR.CalculateParamsHeat and TIR.TuneRuleHeat = 3, 4 or 5 is always possible.
PIDSelfTune.TIR.TuneRuleCool	INT	0	Method for parameter calculation during fine tuning for cooling Options are: • TIR.TuneRuleCool = 0: PID automatic • TIR.TuneRuleCool = 1: PID fast (faster control response with higher amplitudes of the output value than with TIR.TuneRuleCool = 2) • TIR.TuneRuleCool = 2: PID slow (slower control response with lower amplitudes of the output value than with TIR.TuneRuleCool = 1) • TIR.TuneRuleCool = 3: ZN PID • TIR.TuneRuleCool = 4: ZN PI • TIR.TuneRuleCool = 5: ZN P (ZN=Ziegler-Nichols) To be able to repeat the calculation of the PID parameters for cooling with TIR.CalculateParamsCool and TIR.TuneRuleCool = 0, 1 or 2, the previous fine tuning also has to have been executed with TIR.TuneRuleCool = 0, 1 or 2. If this is not the case, TIR.TuneRuleCool = 3 is used. The recalculation of the PID parameters for cooling with TIR.CalculateParamsCool and TIR.TuneRuleCool = 3, 4 or 5 is always possible. Only effective if the cooling output and PID parameter switching are activated (ConfigActivateCooling = TRUE and Config.AdvancedCooling = TRUE).

Tag	Data type	Default	Description
PIDSelfTune.TIR.State	INT	0	The TIR.State tag indicates the current phase of "fine tuning": • State = 0: Initialize fine tuning • State = 100: Calculate standard deviation for heating • State = 200: Calculate standard deviation for cooling • State = 300: Attempting to reach setpoint for heating with two-step control using heating • State = 400: Attempting to reach setpoint for cooling with two-step control using cooling • State = 500: Attempting to reach setpoint for heating with PID control • State = 600: Attempting to reach setpoint for cooling with PID control • State = 700: Calculate standard deviation for heating • State = 800: Calculate standard deviation for cooling • State = 900: Determine oscillation and calculate parameters for heating • State = 1000: Determine oscillation and calculate parameters for cooling • State = 9900: Fine tuning successful • State = 1: Fine tuning not successful
PIDSelfTune.TIR.ProcParHeatOk	BOOL	FALSE	TRUE: The calculation of the process parameters for fine tuning heating was successful. This tag is set during tuning. It must be met for calculation of the PID parameters for heating.
PIDSelfTune.TIR.ProcParCoolOk	BOOL	FALSE	TRUE: The calculation of the process parameters for fine tuning cooling was successful. This tag is set during tuning. It must be met for calculation of the PID parameters for cooling.

Tag	Data type	Default	Description
PIDSelfTune.TIR.OutputOffsetHeat	REAL	0.0	Tuning offset heating of the PID output value TIR.OutputOffsetHeat is added to the value that results from PidOutputSum for the heating branch. To receive a positive offset at the outputs for heating, define a positive value for TIR.OutputOffsetHeat. The resulting values at the outputs for heating are the result of the configured output scaling (Struktur Config.Output.Heat). This tuning offset can be used in controllers with activated cooling output and PID parameter switching (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE) for fine tuning cooling. If the outputs for cooling are not active at the setpoint that is to be tuned (PidOutputSum > 0.0), fine tuning cooling is not possible. In this case, define a positive tuning offset heating which is greater than the PID output value (PidOutputSum) at the setpoint in the steady state before you start tuning. This step increases the values at the outputs for heating and activates the outputs for cooling (PidOutputSum < 0.0). Fine tuning cooling is now possible. When fine tuning is complete, TIR.OutputOffsetHeat is reset to 0.0. Major changes at TIR.OutputOffsetHeat in one step can result in temporary overshoots. Config.Output.Heat.PidUpperLimit ≥ PIDSelfTune.TIR.OutputOffsetHeat ≥ Config.Output.Heat.PidLowerLimit
PIDSelfTune.TIR.OutputOffsetCool	REAL	0.0	Tuning offset cooling of the PID output value TIR.OutputOffsetCool is added to the value that results from PidOutputSum for the cooling branch. To receive a positive offset at the outputs for cooling, define a negative value for TIR.OutputOffsetCool. The resulting values at the outputs for cooling are the result of the configured output scaling (Struktur Config.Output.Coool). This tuning offset can be used in controllers with activated cooling output (Config.ActivateCooling = TRUE) for fine tuning heating. If the outputs for heating are not active at the setpoint that is to be tuned (PidOutputSum < 0.0), fine tuning heating is not possible. In this case, define a negative tuning offset cooling which is less than the PID output value (PidOutputSum) at the setpoint in the steady state before you start tuning. This step increases the values at the outputs for cooling and activates the outputs for heating (PidOutputSum > 0.0). Fine tuning heating is now possible. When fine tuning is complete, TIR.OutputOffsetCool is reset to 0.0. Major changes at TIR.OutputOffsetCool in one step can result in temporary overshoots. Config.Output.Cool.PidUpperLimit ≥ PIDSelfTune.TIR.OutputOffsetCool ≥ Config.Output.Cool.PidLowerLimit

Tag	Data type	Default	Description
PIDSelfTune.TIR.WaitForControlIn	BOOL	FALSE	Waiting with fine tuning after reaching the setpoint If TIR.WaitForControlIn = TRUE, fine tuning waits in between reaching the setpoint (TIR.State = 500 or 600) and calculation of the standard deviation (TIR.State = 700 or 800) until a FALSE -> TRUE edge is given at TIR.FinishControlIn. TIR.WaitForControlIn can be used for simultaneous fine tuning of several controllers in multi-zone applications to synchronize tuning of the individual zones. It ensures that all zones have reached their setpoints before the actual tuning starts. The influence of thermal connections between the zones on tuning can be reduced in this way. TIR.WaitForControlIn is only effective if fine tuning is started from automatic mode with PIDSelfTune.TIR.RunIn = FALSE.
PIDSelfTune.TIR.ControlInReady	BOOL	FALSE	If TIR.WaitForControlIn = TRUE, PID_Temp sets TIR.ControlInReady = TRUE as soon as the setpoint has been reached and waits with additional tuning steps until a FALSE -> TRUE edge is given at TIR.FinishControlIn.
PIDSelfTune.TIR.FinishControlIn	BOOL	FALSE	If TIR.ControlInReady = TRUE, a FALSE -> TRUE edge at TIR.FinishControlIn stops the wait and fine tuning resumes.
PIDCtrl.IOutputOld	REAL	0.0	Integral action in last cycle
PIDCtrl.PIDInit	BOOL	FALSE	PIDCtrl.PIDInit is available as of PID_Temp version 1.1. If PIDCtrl.PIDInit = TRUE in "Automatic mode", the integral action PIDCtrl.IOutputOld is pre-assigned automatically as if PidOutputSum = OverwriteInitialOutputValue in the previous cycle. This can be used for a Override control with PID_Temp (Page 6135).
Retain.CtrlParams.SetByUser	BOOL	FALSE	If the PID parameters are entered manually in the configuration editor, SetByUser = TRUE. This parameter is used for display in the editors and does not influence the control algorithm. SetByUser is retentive.
Retain.CtrlParams.Heat.Gain	REAL	1.0	Active proportional gain for heating Heat.Gain is retentive. Heat.Gain ≥ 0.0
Retain..CtrlParams.Heat.Ti	REAL	20.0	Active integral action time for heating in seconds The integral action for heating is switched off with Heat.CtrlParams.Ti = 0.0. Heat.Ti is retentive. 100000.0 ≥ Heat.Ti ≥ 0.0
Retain.CtrlParams.Heat.Td	REAL	0.0	Active derivative action time for heating in seconds The derivative action for heating is switched off with Heat.CtrlParams.Td = 0.0. Heat.Td is retentive. 100000.0 ≥ Heat.Td ≥ 0.0

Tag	Data type	Default	Description
Retain.CtrlParams.Heat.TdFiltRatio	REAL	0.2	Active derivative delay coefficient for heating The derivative delay coefficient delays the effect of the derivative action. Derivative delay = derivative action time × derivative delay coefficient • 0.0: Derivative action is effective for one cycle only and therefore almost not effective. • 0.5: This value has proved useful in practice for controlled systems with **one** dominant time constant. • > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed. Heat.TdFiltRatio is retentive. Heat.TdFiltRatio ≥ 0.0
Retain.CtrlParams.Heat.PWeighting	REAL	1.0	Active weighting of the proportional action for heating The proportional action may weaken with changes to the setpoint. Values from 0.0 to 1.0 are applicable. • 1.0: Proportional action for setpoint change is fully effective • 0.0: Proportional action for setpoint change is not effective The proportional action is always fully effective when the process value is changed. Heat.PWeighting is retentive. 1.0 ≥ Heat.PWeighting ≥ 0.0
Retain.CtrlParams.Heat.DWeighting	REAL	1.0	Active weighting of the derivative action for heating The derivative action may weaken with changes to the setpoint. Values from 0.0 to 1.0 are applicable. • 1.0: Derivative action is fully effective upon setpoint change • 0.0: Derivative action is not effective upon setpoint change The derivative action is always fully effective when the process value is changed. Heat.DWeighting is retentive. 1.0 ≥ Heat.DWeighting ≥ 0.0
Retain.CtrlParams.Heat.Cycle	REAL	1.0	Active sampling time of the PID algorithm for heating in seconds CtrlParams.Heat.Cycle is calculated during tuning and rounded to an integer multiple of CycleTime.Value. If Config.Output.Heat.PwmPeriode = 0.0, Heat.Cycle is used as period duration of the pulse width modulation for heating. If Config.Output.Cool.PwmPeriode = 0.0 and Config.AdvancedCooling = FALSE, Heat.Cycle is used as period duration of the pulse width modulation for cooling. Heat.Cycle is retentive. 100000.0 ≥ Heat.Cycle > 0.0

Tag	Data type	Default	Description
Retain.CtrlParams.Heat.Control-Zone	REAL	3.402822e +38	Active control zone width for heating The control zone for heating is switched off with Heat.Control-Zone = 3.402822e+38. Heat.ControlZone is only set automatically during pretuning heating or pretuning heating and cooling if PIDSelf-Tune.SUT.TuneRuleHeat = 2 is selected as method of the parameter calculation. For controllers with deactivated cooling output (Config.Activa-teCooling = FALSE) or controllers with activated cooling output and cooling factor (Config.AdvancedCooling = FALSE), the control zone is symmetrically located between Setpoint – Heat.ControlZone and Setpoint + Heat.ControlZone. For controllers with activated cooling output and PID parameter switching (Config.ActivateCooling = TRUE, Config.Advanced-Cooling = TRUE), the control zone is located between Setpoint – Heat.ControlZone and Setpoint + Cool.ControlZone. Heat.ControlZone is retentive. Heat.ControlZone > 0.0
Retain.CtrlParams.Heat.Dead-Zone	REAL	0.0	Active dead zone width for heating (see PID parameters (Page 6112)) The dead zone for heating is switched off with Heat.DeadZone = 0.0. Heat.DeadZone is not set automatically or adjusted during tuning. You must correctly configure Heat.DeadZone manually. When the dead zone is switched on, the result can be a permanent control deviation (deviation between setpoint and process value). This can have a negative effect on fine tuning. For controllers with deactivated cooling output (Config.Activa-teCooling = FALSE) or controllers with activated cooling output and cooling factor (Config.AdvancedCooling = FALSE), the dead zone is symmetrically located between Setpoint – Heat.DeadZone and Setpoint + Heat.DeadZone. For controllers with activated cooling output and PID parameter switching (Config.ActivateCooling = TRUE, Config.Advanced-Cooling = TRUE), the dead zone is located between Setpoint – Heat.DeadZone and Setpoint + Cool.DeadZone. Heat.DeadZone is retentive. Heat.DeadZone ≥ 0.0
Retain.CtrlParams.Cool.Gain	REAL	1.0	Active proportional gain for cooling Cool.Gain is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). Cool.Gain ≥ 0.0

Tag	Data type	Default	Description
Retain.CtrlParams.Cool.Ti	REAL	20.0	Active integral action time for cooling in seconds The integral action for cooling is switched off with Cool.CtrlParams.Ti = 0.0. Cool.Ti is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). 100000.0 ≥ Cool.Ti ≥ 0.0
Retain.CtrlParams.Cool.Td	REAL	0.0	Active derivative action time for cooling in seconds The derivative action for cooling is switched off with Cool.CtrlParams.Td = 0.0. Cool.Td is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). 100000.0 ≥ Cool.Td ≥ 0.0
Retain.CtrlParams.Cool.TdFiltRatio	REAL	0.2	Active derivative delay coefficient for cooling The derivative delay coefficient delays the effect of the derivative action. Derivative delay = derivative action time × derivative delay coefficient • 0.0: Derivative action is effective for one cycle only and therefore almost not effective. • 0.5: This value has proved useful in practice for controlled systems with one dominant time constant. • > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed. Cool.TdFiltRatio is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). Cool.TdFiltRatio ≥ 0.0
Retain.CtrlParams.Cool.PWeighting	REAL	1.0	Active weighting of the proportional action for cooling The proportional action may weaken with changes to the setpoint. Values from 0.0 to 1.0 are applicable. • 1.0: Proportional action for setpoint change is fully effective • 0.0: Proportional action for setpoint change is not effective The proportional action is always fully effective when the process value is changed. Cool.PWeighting is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). 1.0 ≥ Cool.PWeighting ≥ 0.0

Tag	Data type	Default	Description
Retain.CtrlParams.Cool.DWeighting	REAL	1.0	Active weighting of the derivative action for cooling The derivative action may weaken with changes to the setpoint. Values from 0.0 to 1.0 are applicable. • 1.0: Derivative action is fully effective upon setpoint change • 0.0: Derivative action is not effective upon setpoint change The derivative action is always fully effective when the process value is changed. Cool.DWeighting is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). 1.0 ≥ Cool.DWeighting ≥ 0.0
Retain.CtrlParams.Cool.Cycle	REAL	1.0	Active sampling time of the PID algorithm for cooling in seconds CtrlParams.Cool.Cycle is calculated during tuning and rounded off to an integer multiple of CycleTime.. If Config.Output.Cool.PwmPeriode = 0.0 and Config.AdvancedCooling = TRUE, Cool.Cycle is used as period duration of the pulse width modulation for cooling. If Config.Output.Cool.PwmPeriode = 0.0 and Config.AdvancedCooling = FALSE, Heat.Cycle is used as period duration of the pulse width modulation for cooling. Cool.Cycle is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). 100000.0 ≥ Cool.Cycle > 0.0

Tag	Data type	Default	Description
Retain.CtrlParams.Cool.Control-Zone	REAL	3.402822e+38	Active control zone width for cooling The control zone for cooling is switched off with Cool.Control-Zone = 3.402822e+38. Cool.ControlZone is only set automatically during pretuning cooling or pretuning heating and cooling if PIDSelf-Tune.SUT.TuneRuleCool = 2 is selected as method of the parameter calculation. Cool.ControlZone is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). Cool.ControlZone > 0.0
Retain.CtrlParams.Cool.Dead-Zone	REAL	0.0	Active dead zone width for cooling (see PID parameters (Page 6112)) The dead zone for cooling is switched off with Cool.DeadZone = 0.0. Cool.DeadZone is not set automatically or adjusted during tuning. You must correctly configure Cool.DeadZone manually. When the dead zone is switched on, the result can be a permanent control deviation (deviation between setpoint and process value). This can have a negative effect on fine tuning. Cool.DeadZone is retentive. Only effective if the cooling output and PID parameter switching are activated (Config.ActivateCooling = TRUE and Config.AdvancedCooling = TRUE). Cool.DeadZone ≥ 0.0

Note

Change the tags listed in this table in "Inactive" mode to prevent malfunction of the PID controller.

See also

PID_Temp ActivateRecoverMode tag (Page 4375)

PID_Temp Warning tag (Page 4378)

Multi-zone controlling with PID_Temp (Page 6133)

PID_Temp state and mode parameters (S7-1200, S7-1500)

Correlation of the parameters

The State parameter shows the current operating mode of the PID controller. You cannot change the State parameter.

With a rising edge at ModeActivate, PID_Temp switches to the operating mode saved in the Mode in-out parameter.

Heat.EnableTuning and Cool.EnableTuning specify for pretuning and fine tuning, if tuning takes place for heating or cooling.

If the CPU is switched on or switches from Stop to RUN mode, PID_Temp starts in the operating mode that is saved in the Mode parameter. To leave PID_Temp in "Inactive" mode, set RunModeByStartup = FALSE.

Meaning of values

State / Mode	Description of operating mode
0	**Inactive** The following output values are output in "Inactive" mode: • 0.0 as PID output value (PidOutputSum) • 0.0 as output value for heating (OutputHeat) and output value for cooling (OutputCool) • 0 as analog output value for heating (OutputHeat_PER) and analog output value for cooling (OutputCool_PER) • FALSE as PWM output value for heating (OutputHeat_PWM) and PWM output value for cooling (OutputCool_PWM) This does not depend on the configured output value limits and scaling in the structures Config.Output.Heat and Config.Output.Cool.

State / Mode	Description of operating mode
1	**Pretuning** The pretuning determines the process response to a jump change of the output value and searches for the point of inflection. The PID parameters are calculated from the maximum rate of rise and dead time of the controlled system. You obtain the best PID parameters when you perform pretuning and fine tuning. PID_Temp offers different pretuning types depending on the configuration: • Pretuning heating: A jump change is output at the output value heating, the PID parameters for heating are calculated (Retain.CtrlParams.Heat structure), and control to the setpoint then takes place in automatic mode. If the process behavior strongly depends on the operating point, an adaptation of the delay time can be activated at the setpoint with PIDSelfTune.SUT.AdaptDelayTime. • Pretuning heating and cooling: A jump is output at the output value heating. As soon as the process value is close to the setpoint, a jump change is output at the output value cooling. The PID parameters for heating (Retain.CtrlParams.Heat structure) and cooling (Retain.CtrlParams.Cool structure) are calculated. Then, control to the setpoint takes place in automatic mode. If the process behavior strongly depends on the operating point, an adaptation of the delay time can be activated at the setpoint with PIDSelfTune.SUT.AdaptDelayTime. Depending on the effect of the cooling actuator compared to the heating actuator, the quality of tuning can be influenced by whether or not the heating and cooling outputs are operated simultaneously during tuning. You can specify this with PIDSelfTune.SUT.CoolingMode. • Pretuning cooling: A jump change is output at the output value cooling and the PID parameters for cooling are calculated (Struktur Retain.CtrlParams.Cool). Then, control to the setpoint takes place in automatic mode. If you want to tune the PID parameters for heating and cooling, you can expect a better control response with "Pretuning heating" followed by "Pretuning cooling" rather than with "Pretuning heating and cooling". However, carrying out pretuning in two steps takes more time. **General requirements for pretuning:** • The PID_Temp instruction is called in a cyclic interrupt OB. • Inactive (State = 0), manual mode (State = 4), or automatic mode (State = 3) • ManualEnable = FALSE • Reset = FALSE • The setpoint and the process value lie within the configured limits. **Requirements for pretuning heating:** • Heat.EnableTuning = TRUE • Cool.EnableTuning = FALSE • The process value must not be too close to the setpoint. |Setpoint - Input| > 0.3 * |Config.InputUpperLimit - Config.InputLowerLimit| and |Setpoint - Input| > 0.5 * |Setpoint| • The setpoint is greater than the process value.

State / Mode	Description of operating mode
	Setpoint > Input **Requirements for pretuning heating and cooling:** • Heat.EnableTuning = TRUE· • Cool.EnableTuning = TRUE • The cooling output is activated (Config.ActivateCooling = TRUE). • The PID parameter switching is activated (Config.AdvancedCooling = TRUE). • The process value must not be too close to the setpoint. |Setpoint - Input| > 0.3 * |Config.InputUpperLimit - Config.InputLowerLimit| and |Setpoint - Input| > 0.5 * |Setpoint| • The setpoint is greater than the process value. Setpoint > Input **Requirements for pretuning cooling:** • Heat.EnableTuning = FALSE· • Cool.EnableTuning = TRUE· • The cooling output is activated (Config.ActivateCooling = TRUE). • The PID parameter switching is activated (Config.AdvancedCooling = TRUE). • A "pretuning heating" or "pretuning heating and cooling" has been successful (PIDSelfTune.SUT.ProcParHeatOk = TRUE), if possible at the same setpoint. • The process value must be close to the setpoint. |Setpoint - Input| < 0.05 * |Config.InputUpperLimit - Config.InputLowerLimit| The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise. This is most likely the case in operating modes "Inactive" or "Manual mode". The setpoint is frozen in the CurrentSetpoint tag. Tuning is canceled when: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel The method for calculation of the PID parameters can be specified separately for heating and cooling with PIDSelfTune.SUT.TuneRuleHeat and PIDSelfTune.SUT.TuneRuleCool. Before the PID parameters are recalculated, they are backed up in the CtrlParamsBackUp structure and can be reactivated with LoadBackUp. After successful pretuning, the switch is made to automatic mode. After unsuccessful pretuning, the switch to the mode is determined by ActivateRecoverMode. The phase of pretuning is indicated with PIDSelfTune.SUT.State.

State / Mode	Description of operating mode
2	**Fine tuning** Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are tuned for the operating point from the amplitude and frequency of this oscillation. PID parameters from fine tuning usually have better master control and disturbance characteristics than PID parameters from pretuning. You obtain the best PID parameters when you perform pretuning and fine tuning. PID_Temp automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. PID_Temp offers different fine tuning types depending on the configuration: • Fine tuning heating: PID_Temp generates an oscillation of the process value with periodic changes at the output value heating and calculates the PID parameters for heating (Struktur Retain.CtrlParams.Heat). • Fine tuning cooling: PID_Temp generates an oscillation of the process value with periodic changes at the output value cooling and calculates the PID parameters for cooling (Struktur Retain.CtrlParams.Cool). **Temporary tuning offset for heating/cooling controllers** If PID_Temp is used as heating/cooling controller (Config.ActivateCooling = TRUE), the PID output value (PidOutputSum) at the setpoint must meet the following requirements for a process value oscillation to be generated and fine tuning to be successful: • Positive PID output value for fine tuning heating • Negative PID output value for fine tuning cooling If this requirement is not met, you can define a temporary offset for fine tuning which is output at the output with the opposite effect: • Offset for cooling output (PIDSelfTune.TIR.OutputOffsetCool) with fine tuning heating. Define a negative tuning offset cooling which is less than the PID output value (PidOutputSum) at the setpoint in the steady state before you start tuning. • Offset for heating output (PIDSelfTune.TIR.OutputOffsetHeat) with fine tuning cooling. Define a positive tuning offset heating which is greater than the PID output value (PidOutputSum) at the setpoint in the steady state before you start tuning. The defined offset is balanced by the PID algorithm so that the process value remains at the setpoint. The height of the offset allows the PID output value to be adapted correspondingly so that it fulfills the requirement mentioned above. To avoid larger overshoots of the process value when defining the offset, it can also be increased in several steps. If PID_Temp exits the fine tuning mode, the tuning offset is reset. Example for definition of an offset for fine tuning cooling: • Without offset: – Setpoint = Process value (ScaledInput) = 80°C – PID output value (PidOutputSum) = 30.0 – Output value heating (OutputHeat) = 30.0 – Output value cooling (OutputCool) = 0.0 Oscillation of the process value around the setpoint cannot be generated with the cooling output alone. Fine tuning would fail here. • With definition of an offset for heating output (PIDSelfTune.TIR.OutputOffsetHeat) = 80.0 – Setpoint = process value (ScaledInput) = 80°C – PID output value (PidOutputSum) = -50.0

State / Mode	Description of operating mode
	– Output value heating (OutputHeat) = 80.0 – Output value cooling (OutputCool) = -50.0 By defining an offset for the heating output, the cooling output can now create an oscillation of the process value around the setpoint. Fine tuning can now be carried out successfully. **General requirements for fine tuning:** • The PID_Temp instruction is called in a cyclic interrupt OB. • No disturbances are expected. • The setpoint and the process value lie within the configured limits. • The control loop has stabilized at the operating point. The operating point is reached when the process value corresponds to the setpoint. When the dead zone is switched on, the result can be a permanent control deviation (deviation between setpoint and actual value). This can have a negative effect on fine tuning. • ManualEnable = FALSE • Reset = FALSE • Automatic (State = 3), inactive (State = 0) or manual (State = 4) mode **Requirements for fine tuning heating:** • Heat.EnableTuning = TRUE • Cool.EnableTuning = FALSE • If PID_Temp is configured as heating/cooling controller (Config.ActivateCooling = TRUE), the heating output must be active at the operating point at which tuning is to take place (PidOutputSum > 0.0 (see tuning offset)). **Requirements for fine tuning cooling:** • Heat.EnableTuning = FALSE • Cool.EnableTuning = TRUE • The cooling output is activated (Config.ActivateCooling = TRUE). • The PID parameter switching is activated (Config.AdvancedCooling = TRUE) • The cooling output must be active at the operating point at which tuning is to take place (PidOutputSum < 0.0 (see tuning offset)). **The course of fine tuning is determined by the mode from which it is started:** • Automatic mode (State = 3) with PIDSelfTune.TIR.RunIn = FALSE (default) Start fine tuning from automatic mode if you wish to improve the existing PID parameters through tuning. PID_Temp controls the system using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start. • Inactive (State = 0), manual mode (State = 4), or automatic mode (State = 3) with PIDSelfTune.TIR.RunIn = TRUE Attempts are made to reach the setpoint with the minimum or maximum output value: – with minimum or maximum output value heating for fine tuning heating – With minimum or maximum output value cooling for fine tuning cooling. This can produce increased overshoot. Fine tuning starts when the setpoint is reached.

State / Mode	Description of operating mode
	If the setpoint cannot be reached, PID_Temp does not automatically abort tuning. The setpoint is frozen in the CurrentSetpoint tag. Tuning is canceled when: • Setpoint > CurrentSetpoint + CancelTuningLevel or • Setpoint < CurrentSetpoint - CancelTuningLevel The method for calculation of the PID parameters can be specified separately for heating and cooling with PIDSelfTune.TIR.TuneRuleHeat and PIDSelfTune.TIR.TuneRuleCool. Before the PID parameters are recalculated, they are backed up in the CtrlParamsBackUp structure and can be reactivated with LoadBackUp. The controller changes to automatic mode after successful fine tuning. After unsuccessful fine tuning, the switch to the mode is determined by ActivateRecoverMode. The "Fine tuning" phase is indicated with PIDSelfTune.TIR.State.
3	**Automatic mode** In automatic mode, PID_Temp corrects the controlled system in accordance with the parameters specified. The controller switches to automatic mode if one the following requirements is met: • Pretuning successfully completed • Fine tuning successfully completed • Changing of the Mode in-out parameter to the value 3 and a rising edge at ModeActivate. The switchover from automatic mode to manual mode is only bumpless if carried out in the commissioning editor. The ActivateRecoverMode tag is taken into consideration in automatic mode.
4	**Manual mode** In manual mode, you specify a manual PID output value in the ManualValue parameter. The values at the outputs for heating and cooling resulting from this manual value are the result of the configured output scaling. You can also activate this operating mode using ManualEnable = TRUE. We recommend that you change the operating mode using Mode and ModeActivate only. The switchover from manual mode to automatic mode is bumpless. The ActivateRecoverMode tag is taken into consideration in manual mode.
5	**Substitute output value with error monitoring** The control algorithm is deactivated. The SetSubstituteOutput tag determines which PID output value (PidOutputSum) is output in this operating mode. • SetSubstituteOutput = FALSE: Last valid PID output value • SetSubstituteOutput = TRUE: Substitute output value (SubstituteOutput) You cannot activate this operating mode using Mode = 5. In the event of an error, it is activated instead of "Inactive" operating mode if all the following conditions are met: • Automatic mode (State = 3) • ActivateRecoverMode = TRUE • One or more errors have occurred in which ActivateRecoverMode is effective. As soon as the errors are no longer pending, PID_Temp switches back to automatic mode.

ENO characteristics

If State = 0, then ENO = FALSE.

If State ≠ 0, then ENO = TRUE.

Automatic switchover of operating mode during commissioning

Automatic mode is activated following successful pretuning or fine tuning. The following table shows how Mode and State change during successful pretuning.

Cycle no.	Mode	State	Action
0	4	4	Set Mode = 1
1	1	4	Set ModeActivate = TRUE
1	4	1	Value of State is saved in Mode parameter Pretuning is started
n	4	1	Pretuning successfully completed
n	3	3	Automatic mode is started

PID_Temp automatically switches the operating mode in the event of an error.

The following table shows how Mode and State change during pretuning with errors.

Cycle no.	Mode	State	Action
0	4	4	Set Mode = 1
1	1	4	Set ModeActivate = TRUE
1	4	1	Value of State is saved in Mode parameter Pretuning is started
n	4	1	Pretuning canceled
n	4	4	Manual mode is started

If ActivateRecoverMode = TRUE, the operating mode that is saved in the Mode parameter is activated. When you start pretuning or fine tuning, PID_Temp has saved the value of State in the Mode in-out parameter. This means PID_Temp switches to the mode from which tuning was started.

If ActivateRecoverMode = FALSE, the system switches to "Inactive" operating mode.

See also

Output parameters of PID_Temp (Page 4330)

In/out parameters of PID_Temp V2 (Page 4333)

PID_Temp ErrorBits parameter (S7-1200, S7-1500)

If several errors are pending simultaneously, the values of the ErrorBits are displayed with binary addition. The display of ErrorBits = 0000003h, for example, indicates that the errors 0000001h and 0000002h are pending simultaneously.

ErrorBits (DW#16#...)	Description
0000000	There is no error.
0000001	The "Input" parameter is outside the process value limits. • Input > Config.InputUpperLimit or • Input < Config.InputLowerLimit If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in automatic mode. If manual mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in manual mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp switches to the operating mode that is saved in the Mode parameter.
0000002	Invalid value at "Input_PER" parameter. Check whether an error is pending at the analog input. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp outputs the configured substitute output value. As soon as the error is no longer pending, PID_Temp switches back to automatic mode. If manual mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in manual mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp switches to the operating mode that is saved in the Mode parameter.
0000004	Error during fine tuning. Oscillation of the process value could not be maintained. If PID_Temp is used as heating-cooling controller (Config.ActivateCooling = TRUE), to be able to generate actual value oscillation, the PID output value (PidOutputSum) at the setpoint must be • positive for fine tuning heating • negative for fine tuning cooling If this requirement is not met, use the tuning offsets (PIDSelfTune.TIR.OutputOffsetCool and PIDSelf-Tune.TIR.OutputOffsetHeat tags), see Fine tuning (Page 6121). If ActivateRecoverMode was = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0000008	Error at start of pretuning. The process value is too close to the setpoint or greater than the setpoint. Start fine tuning. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0000010	The setpoint was changed during tuning. You can set the permitted fluctuation of the setpoint at the CancelTuningLevel tag. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0000020	Pretuning is not permitted during fine tuning. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp remains in fine tuning mode.
0000040	Error during pretuning. Cooling could not reduce the process value. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.

ErrorBits (DW#16#...)	Description
0000100	Error during fine tuning resulted in invalid parameters. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0000200	Invalid value at "Input" parameter: Value has an invalid number format. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp outputs the configured substitute output value. As soon as the error is no longer pending, PID_Temp switches back to automatic mode. If manual mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in manual mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp switches to the operating mode that is saved in the Mode parameter.
0000400	Calculation of output value failed. Check the PID parameters. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp outputs the configured substitute output value. As soon as the error is no longer pending, PID_Temp switches back to automatic mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp switches to the operating mode that is saved in the Mode parameter.
0000800	Sampling time error: PID_Temp is not called within the sampling time of the cyclic interrupt OB. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in automatic mode. If manual mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in manual mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp switches to the operating mode that is saved in the Mode parameter. If this error occurred during simulation with PLCSIM, see the notes under Simulating PID_Temp with PLCSIM (Page 6139).
0001000	Invalid value at "Setpoint" parameter or "SubstituteSetpoint": Value has an invalid number format. If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp outputs the configured substitute output value. As soon as the error is no longer pending, PID_Temp switches back to automatic mode. If manual mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in manual mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp switches to the operating mode that is saved in the Mode parameter.
0010000	Invalid value at ManualValue parameter. Value has an invalid number format. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp remains in manual mode and uses SubstituteOutput as PID output value. As soon as you specify a valid value in ManualValue, PID_Temp uses it as the PID output value.
0020000	Invalid value at SubstituteOutput tag. Value has an invalid number format. PID_Temp remains in the "Substitute output value with error monitoring" mode or manual mode and uses the low limit of the PID output value for heating (Config.Output.Heat.PidLowerLimit) as PID output value. As soon as you specify a valid value in SubstituteOutput, PID_Temp uses it as the PID output value.

ErrorBits (DW#16#...)	Description
0040000	Invalid value at Disturbance parameter. Value has an invalid number format. If automatic mode was active and ActivateRecoverMode = TRUE before the error occurred, Disturbance is set to zero. PID_Temp remains in automatic mode. If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp switches to the operating mode that is saved in the Mode parameter. If Disturbance in the current phase has no effect on the output value, tuning is not be canceled.
0200000	Error in master in the cascade: Slaves are not in automatic mode or have activated substitute setpoint and prevent tuning of the master. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0400000	Pretuning heating is not permitted while cooling is active. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
0800000	The process value must be close to the setpoint to start pretuning cooling. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
1000000	Error at start of tuning: Heat.EnableTuning and Cool.EnableTuning are not set or do not match the configuration. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
2000000	Pretuning cooling requires successful pretuning heating. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
4000000	Error at start of fine tuning: Heat.EnableTuning and Cool.EnableTuning must not be set simultaneously. If ActivateRecoverMode = TRUE before the error occurred, PID_Temp cancels the tuning and switches to the operating mode that is saved in the Mode parameter.
8000000	Error during calculation of the PID parameters resulted in invalid parameters. The invalid parameters are discarded and the original PID parameters are retained unchanged. We can distinguish between the following cases: • If automatic mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in automatic mode. • If manual mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp remains in manual mode. • If pretuning or fine tuning mode was active before the error occurred and ActivateRecoverMode = TRUE, PID_Temp switches to the operating mode that is saved in the Mode parameter.

PID_Temp ActivateRecoverMode tag (S7-1200, S7-1500)

The ActivateRecoverMode tag determines the reaction to error. The Error parameter indicates if an error is pending. When the error is no longer pending, Error = FALSE. The ErrorBits parameter shows which errors have occurred.

Automatic mode and manual mode

NOTICE
Your system may be damaged. If ActivateRecoverMode = TRUE, PID_Temp remains in automatic mode or in manual mode even if there is an error and the process limit values are exceeded. This may damage your system. It is essential to configure how your controlled system reacts in the event of an error to protect your system from damage.

ActivateRecoverMode	Description
FALSE	PID_Temp switches to "Inactive" mode in the event of an error. The controller is only activated by a falling edge at Reset or a rising edge at ModeActivate.
TRUE	**Automatic mode** If errors occur frequently in automatic mode, this setting has a negative effect on the control response, because PID_Temp switches between the calculated PID output value and the substitute output value at each error. In this case, check the ErrorBits parameter and eliminate the cause of the error. If one or several of the following errors occur and automatic mode was active before the error occurred, PID_Temp remains in automatic mode: • 0000001h: The "Input" parameter is outside the process value limits. • 0000800h: Sampling time error • 0040000h: Invalid value at Disturbance parameter. • 8000000h: Error during calculation of the PID parameters If one or several of the following errors occur and automatic mode was active before the error occurred, PID_Temp switches to "Substitute output value with error monitoring" mode: • 0000002h: Invalid value at Input_PER parameter. • 0000200h: Invalid value at Input parameter. • 0000400h: Calculation of output value failed. • 0001000h: Invalid value at Setpoint parameter or SubstituteSetpoint. As soon as the errors are no longer pending, PID_Temp switches back to automatic mode. If the following error occurs in "Substitute output value with error monitoring" mode, PID_Temp sets the PID output value to Config.Output.Heat.PidLowerLimit as long as this error is pending: • 0020000h: Invalid value at SubstituteOutput tag. Value has an invalid number format. This behavior is independent of SetSubstituteOutput.
	Manual mode If one or several errors occur and manual mode was active before the error occurred, PID_Temp remains in manual mode. If the following error occurs in manual mode, as long as this error is pending, PID_Temp sets the PID output value to SubstituteOutput: • 0010000h: Invalid value at ManualValue parameter. Value has an invalid number format. If the error 0010000h is pending in manual mode and the following error occurs, PID_Temp sets the PID output value to Config.Output.Heat.PidLowerLimit as long as this error is pending: • 0020000h: Invalid value at SubstituteOutput tag. Value has an invalid number format. This behavior is independent of SetSubstituteOutput.

Pretuning and fine tuning

ActivateRecoverMode	Description
FALSE	PID_Temp switches to "Inactive" mode in the event of an error. The controller is only activated by a falling edge at Reset or a rising edge at ModeActivate.
TRUE	If the following error occurs, PID_Temp remains in the active mode: • 0000020h: Pretuning is not permitted during fine tuning. The following errors are ignored: • 0010000h: Invalid value at ManualValue parameter. • 0020000h: Invalid value at SubstituteOutput tag. When any other error occurs, PID_Temp cancels the tuning and switches to the mode from which tuning was started.

PID_Temp Warning tag (S7-1200, S7-1500)

If several warnings are pending simultaneously, the values of the Warning tag are displayed with binary addition. If the warning 0000003h is displayed, for example, the warnings 0000001h and 0000002h are pending simultaneously.

Warning (DW#16#....)	Description
0000000	No warning pending.
0000001	The point of inflection was not found during pretuning.
0000004	The setpoint was limited to the configured limits.
0000008	Not all the necessary controlled system properties were defined for the selected method of calculation. Instead, the PID parameters were calculated using the method TIR.TuneRuleHeat = 3 or TIR.TuneRuleCool = 3.
0000010	The operating mode could not be changed because Reset = TRUE or ManualEnable = TRUE.
0000020	The cycle time of the calling OB limits the sampling time of the PID algorithm. Improve results by using shorter OB cycle times.
0000040	The process value exceeded one of its warning limits.
0000080	Invalid value at Mode. The operating mode is not switched.
0000100	The manual value was limited to the limits of the PID output value.
0000200	The specified rule for tuning is not supported. No PID parameters are calculated.
0001000	The substitute output value cannot be reached because it is outside the output value limits.
0004000	The specified selection of the output value for heating and/or cooling is not supported. Only the output OutputHeat or OutputCool is used.
0008000	Invalid value at PIDSelfTune.SUT.AdaptDelayTime. The default value 0 is used.
0010000	Invalid value at PIDSelfTune.SUT.CoolingMode. The default value 0 is used.
0020000	The activation of cooling (Config.ActivateCooling tag) is not supported by the controller that is used as master (Config.Cascade.IsMaster tag). PID_Temp works as heating controller. Set the Config.ActivateCooling tag to FALSE.
0040000	Invalid value at Retain.CtrlParams.Heat.Gain, Retain.CtrlParams.Cool.Gain oder Config.CoolFactor. PID_Temp supports only positive values for proportional gain (heating and cooling) and cooling factor. Automatic mode remains active with PID output value 0.0. The integral component is stopped.

The following warnings are deleted as soon as the cause has been remedied or you repeat the action with valid parameters:

- 0000001h
- 0000004h
- 0000008h
- 0000040h
- 0000100h

All other warnings are cleared with a rising edge at Reset or ErrorAck.

PwmPeriode tag (S7-1200, S7-1500)

If the PID algorithm sampling time (Retain.CtrlParams.Heat.Cycle or Retain.CtrlParams.Heat.Cycle) and thus the time period of the pulse width modulation is very high when you use OutputHeat_PWM or OutputCool_PWM, you can define a deviating shorter time period at the Config.Output.Heat.PwmPeriode or Config.Output.Cool.PwmPeriode parameters to improve the smoothness of the process value.

Time period of the pulse width modulation at OutputHeat_PWM

Time period of the PWM at output OutputHeat_PWM depending on Config.Output.Heat.PwmPeriode:

- Heat.PwmPeriode = 0.0 (default)
  The sampling time of the PID algorithm for heating (Retain.CtrlParams.Heat.Cycle) is used as time period of the PWM.
- Heat.PwmPeriode > 0.0
  The value is rounded off to an integer multiple of the PID_Temp sampling time (CycleTime.Value) and used as time period of the PWM.
  The value must meet the following conditions:
  - Heat.PwmPeriode ≤ Retain.CtrlParams.Heat.Cycle
  - Heat.PwmPeriode > Config.Output.Heat.MinimumOnTime
  - Heat.PwmPeriode > Config.Output.Heat.MinimumOffTime

Time period of the pulse width modulation at OutputCool_PWM

Time period of the PWM at output OutputCool_PWM depending on Config.Output.Cool.PwmPeriode and the method for heating/cooling:

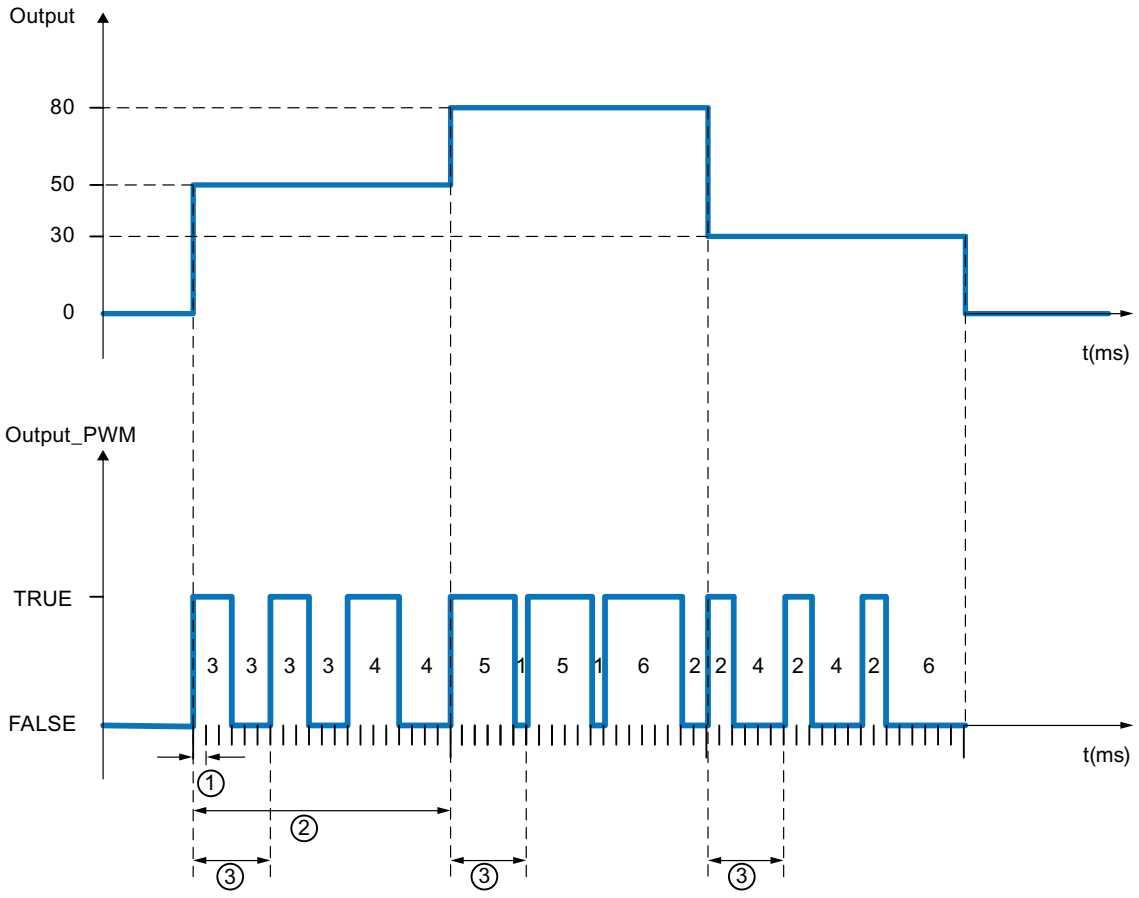
- Cool.PwmPeriode = 0.0 and cooling factor (Config.AdvancedCooling = FALSE):
  The sampling time of the PID algorithm for heating (Retain.CtrlParams.Heat.Cycle) is used as time period of the PWM.
- Cool.PwmPeriode = 0.0 and PID parameter switching (Config.AdvancedCooling = TRUE):
  The sampling time of the PID algorithm for cooling (Retain.CtrlParams.Cool.Cycle) is used as time period of the PWM.
- Cool.PwmPeriode > 0.0:
  The value is rounded off to an integer multiple of the PID_Temp sampling time (CycleTime.Value) and used as time period of the PWM.
  The value must meet the following conditions:
  - Cool.PwmPeriode ≤ Retain.CtrlParams.Cool.Cycle or Retain.CtrlParams.Heat.Cycle
  - Cool.PwmPeriode > Config.Output.Cool.MinimumOnTime
  - Cool.PwmPeriode > Config.Output.Cool.MinimumOffTime

Config.Output.Cool.PwmPeriode is only effective if the cooling output is activated (Config.ActivateCooling =TRUE).

When you use PwmPeriode, the accuracy of the PWM output signal is determined by the relationship of PwmPeriode to the PID_Temp sampling time (cycle time of the OB). PwmPeriode should be at least 10 times the PID_Temp sampling time.

If the sampling time of the PID algorithm is not an integer multiple of PwmPeriode, each last period of the PWM within the sampling time of the PID algorithm is extended accordingly.

Example for OutputHeat_PWM



① PID_Temp sampling time = 100.0 ms (cycle time of the calling cyclic interrupt OB, CycleTime.Value tag)

② PID algorithm sampling time = 2000.0 ms (Retain.CtrlParams.Heat.Cycle tag)

③ Time period of the PWM for heating = 600.0 ms (Config.Output.Heat.PwmPeriode tag)

IntegralResetMode tag (S7-1200, S7-1500)

The IntegralResetMode tag determines how the integral action PIDCtrl.IOutputOld is pre-assigned:

- When switching from "Inactive" operating mode to "Automatic mode"
- With edge TRUE -> FALSE at parameter Reset and parameter Mode = 3

This setting only works for one cycle and is only effective if the integral action is activated (Retain.CtrlParams.Heat.Ti and Retain.CtrlParams.Cool.Ti > 0.0 tags).

IntegralReset-Mode	Description
0	Smooth The value of PIDCtrl.IOutputOld is pre-assigned so that the switchover is bumpless, which means "Automatic mode" starts with the output value = 0.0 (parameter PidOutputSum) and there is no jump of the output value regardless of the control deviation (setpoint – process value).
1	Delete We recommend setting the weighting of the proportional action (Retain.CtrlParams.Heat.PWeighting and Retain.CtrlParams.Cool.PWeighting tags) to 1.0 if this option is used. The value of PIDCtrl.IOutputOld is deleted. Any control deviation will cause a jump of the PID output value. The direction of the output value jump depends on the active weighting of the proportional action (Retain.CtrlParams.Heat.PWeighting and Retain.CtrlParams.Cool.PWeighting tags) and the control deviation: • Active proportional action weighting = 1.0: Output value jump and control deviation have identical signs. Example: If the process value value is smaller than the setpoint (positive control deviation), the PID output value jumps to a positive value. • Active proportional action weighting < 1.0: For large control deviations, the PID output value jump and control deviation have identical signs. Example: If the process value is much smaller than the setpoint (positive control deviation), the PID output value jumps to a positive value. For small control deviations, the PID output value jump and control deviation have different signs. Example: If the process value is just below the setpoint (positive control deviation), the PID output value jumps to a negative value. This is usually not desirable, because it results in a temporary increase in the control deviation. The smaller the configured weighting of the proportional action, the greater the control deviation must be to receive a PID output value jump with identical sign. We recommend setting the weighting of the proportional action (Retain.CtrlParams.Heat.PWeighting and Retain.CtrlParams.Cool.PWeighting tags) to 1.0 when this option is used. Otherwise, you may experience the undesirable behavior described for small control deviations. Alternatively, you can also use IntegralResetMode = 4. This option guarantees identical signs of the PID output value jump and control deviation independent of the configured weighting of the proportional action and the control deviation.
2	Hold The value of PIDCtrl.IOutputOld is not changed. You can define a new value using the user program.
3	Pre-assign The value of PIDCtrl.IOutputOld is automatically pre-assigned as if PidOutputSum = OverwriteInitialOutputValue in the last cycle.
4	Like setpoint change (only for PID_Temp with version ≥ 1.1) The value of PIDCtrl.IOutputOld is automatically pre-assigned so that a similar PID output value jump results as for a PI controller in automatic mode in case of a setpoint change from the current process value to the current setpoint. Any control deviation will cause a jump of the PID output value. The PID output value jump and control deviation have identical signs. Example: If the process value value is smaller than the setpoint (positive control deviation), the PID output value jumps to a positive value. This is independent of the configured weighting of the proportional action and the control deviation.

If IntegralResetMode is assigned a value outside the valid value range, PID_Temp behaves as with the pre-assignment of IntegralResetMode:

- PID_Temp up to V1.0: IntegralResetMode = 1
- PID_Temp as of V1.1: IntegralResetMode = 4

11.4.6 Communication (S7-1200, S7-1500)

11.4.6.1 S7 communication (S7-1200, S7-1500)

Data consistency (S7-1200, S7-1500)

Definition

A data block which cannot be modified by concurrent processes is called consistent data area. A data block which is larger than the consistent data area can thus be falsified as a whole during transmission. This means that a data block which belongs together and which is larger than consistent data area can consist in part of new and of old consistent data at the same time.

Example

An inconsistency can occur when an instruction for communication is interrupted, for example, by a hardware interrupt OB with higher priority. If the user program in this OB now changes the data that has already been partly processed by the instruction, the transferred data originates:

- In part from the time before the hardware interrupt was processed
- And in part from the time after the hardware interrupt was processed

This means that these data are inconsistent (not coherent).

Ensuring data consistency

If the communication process can be interrupted by an interrupt OB, you must ensure that the data is transferred consistently. Make sure that only an image of the data and not the data to be transferred is not changed directly by the interrupt OB. Copy the image of the data to the transfer area of the communication instruction prior to the next data transfer.

- If the user program contains a communication instruction that accesses common data, access to this data area can be coordinated, for example, by means of the DONE parameter. The data consistency of the communication areas which are transferred locally with a communication instruction can therefore be ensured in the user program.
- In the case of S7 communication instructions "PUT (Page 4393)"/"GET (Page 4388)", the size of the consistent data areas must already be taken into consideration during programming or configuration, because a communication block is not available in the user program of the target device (server) to synchronize communication data to the user program.
- For the S7-300 and C7-300 (exception: CPU 318-2 DP) the communication data are copied consistently into the user memory in blocks of 32 bytes in the cycle control point of the operating system. Data consistency is not guaranteed for larger data areas. If a defined data consistency is required, the communication data in the user program may not exceed 32 bytes (maximum of 8 bytes, depending on the version).
- In the S7-400 and S7-1500, on the other hand, the communication data in 462-byte blocks is not processed at the cycle control point, but in fixed time slices during the program cycle. The consistency of a tag is ensured by the system. These communication areas can then be accessed consistently using the "PUT (Page 4393)" / "GET (Page 4388)" instructions or when reading/writing tags, for example by an OP or an OS.

Note

Additional information on data consistency is provided in the description of the specific instructions.

In the S7-1500 module series, you can change parameters during ongoing operation with the instructions of S7 communication. Changed parameters are used immediately, even if a job is still active. Both can lead to data inconsistency! To avoid data inconsistencies, you should never change parameters while a job is running.

Effect on interrupt reaction times

The interrupt reaction times of the CPU are slightly extended by copying the data. The greater the quantity of data which must be transferred with absolute consistency, the longer the interrupt reaction time of a system.

Common parameters of instructions for S7 communication (S7-1200, S7-1500)

Classification

The parameters of the instructions for S7 communication can be divided into the following five categories according to their functions:

1. Control parameters are used to activate an instruction.
2. Addressing parameters are used to address the remote communication partner.
3. Send parameters point to the data areas that are to be sent to the remote partner.
4. Receive parameters point to the data areas where the data received from remote partners will be entered.
5. Status parameters are used to monitor whether the instruction has completed its task without error or for the analysis of any errors that have occurred.

Control parameter

Data exchange will only be activated if the appropriate control parameters have a defined value (for example, are set) when the instruction is called or when the value has undergone a specific change since the previous call (for example, a positive edge).

Addressing parameters

Parameter	Description
ID	Reference to the local connection description (specified by the configuration of the connection).
R_ID	With the R_ID parameter, you specify that a send and a receive instruction belong together: The R_ID parameter must match in the instruction at the sending end and the instruction at the receiving end. This allows the communication of several instruction pairs via the same logic connection. • R_ID must be specified in the form DW#16#wxyzWXYZ. • The instruction pairs of a logical connection specified in R_ID must be unique for this connection.

Note

Addressing parameters ID and R_ID

You can reassign the addressing parameters ID and R_ID during runtime. The new parameters are validated with each new job after the previous job has been closed.

You can use the following options to reduce the number of instance DBs and therefore the work memory required:

1. With tag IDs, you can use several connections via one data instance block.
2. With tag R_IDs, you can define several pairs of sending and receiving instructions for one job with a single instance.
3. You can combine cases 1 and 2.

Please note that the new parameters are valid after the last job is executed. When you activate the sending operation, the R_ID parameter in the instruction at the sending end must match its counterpart at the receiving end.

Status parameter

With the status parameters, you monitor whether the instruction has completed its task correctly or whether it is still active. The status parameters also indicate errors.

Note

The status parameters are valid for one cycle only, namely from the first command following the call until the next call. As a result, you must evaluate these parameters after each instruction cycle.

Send and receive parameters

For communication instructions configured at both ends

- The number of the SD_i and RD_i parameters used must match at the send and receive end.
- The data types of the SD_i and RD_i parameters that belong together must match at the send and receive end.
- The amount of data to be sent according to the SD_i parameter must not exceed the range made available by the corresponding RD_i (does not apply to "BSEND (Page 4407)" / "BRCV (Page 4410)"). The RD_i parameters must (with exception of "BSEND"/"BRCV") have the identical data size.

If you do not keep to the rules above, this is indicated by ERROR = 1 and STATUS = 4.

Note

Supplying the send and receive parameters

With the VARIANT data type, send and receive parameters must always be supplied any time a communication instruction is called. It is not possible, for example, to supply the send parameters of the communication instructions at startup and to only trigger the send job in cyclic operation.

User data size

With the "USEND (Page 4399)", "URCV (Page 4401)", "GET (Page 4388)" and "PUT (Page 4393)" instructions, the amount of data to be transferred must not exceed a defined user data size. The maximum user data size depends on:

- The instruction used
- The communication partner.

The guaranteed minimum size of the user data for an instruction with 1-4 tags is listed in the following table:

Instruction	Partner: S7-300	Partner: S7-400	Partner: S7-1200	Partner: S7-1500
PUT / GET	160 bytes	400 bytes	160 bytes	880 bytes
USEND / URCV	160 bytes	440 bytes	-	920 bytes
BSEND / BRCV	32768/65534 bytes	65534 bytes	-	• 65534 bytes with standard access • 65535 bytes with optimized access

Further information on the user data size can be found in the technical data of the respective CPU.

Exact user data size

If the user data size specified above is insufficient you can determine the maximum byte length of the user data as follows:

First read the data block size valid for communication from the following table:

Local CPU	Remote CPU	Data block size in bytes
S7-1200	Any	240
S7-1500	S7-300	240
	S7-400	480
	S7-1200	240
	S7-1500	960

Use this value in the following table to read the maximum possible user data length in bytes as total of the parameters used. It applies for even lengths of the areas SD_i, RD_i, and ADDR_i.

For each range of uneven length the maximum possible user data length is reduced by one byte.

		Number of SD_i, RD_i, ADDR_i parameters used			
Data block size	**Instruction**	**1**	**2**	**3**	**4**
240 (S7-300)	PUT/GET/ USEND	160	-	-	-
240 (S7-300 via integrated interface)	PUT	212	-	-	-
	GET	222	-	-	-
	USEND	212	-	-	-
240 (S7-400)	PUT	212	196	180	164
	GET	222	218	214	210
	USEND	212	-	-	-
480 (S7-400)	PUT	452	436	420	404
	GET	462	458	454	450
	USEND	452	448	444	440
240 (S7-1200)	PUT	212	196	180	164
	GET	222	218	214	210
960 (S7-1500)	PUT	932	916	900	884
	GET	942	938	934	930
	USEND	932	928	924	920

GET: Read data from a remote CPU (S7-1200, S7-1500)

Description

With the instruction "GET", you can read data from a remote CPU.

The instruction is started on a positive edge at control input REQ:

- The relevant pointers to the areas to be read out (ADDR_i) are then sent to the partner CPU. The partner CPU can be in RUN or STOP mode.
- The partner CPU returns the data:
  - If the reply exceeds the maximum user data length, this is displayed with error code "2" at the STATUS parameter.
  - The received data is copied to the configured receive areas (RD_i) at the next call.
- Completion of this action is indicated by the status parameter NDR having the value "1".

Reading can only be activated again after the previous reading process has been completed. Errors and warnings are output via ERROR and STATUS if access problems occurred while the data was being read or if the data type check results in an error.

Changes in the data areas addressed on the partner CPU are not registered by the "GET" instruction.

Requirements for using the instruction

- The "Permit access with PUT/GET communication from remote partner" function was activated in the properties of the partner CPU under "Protection".
- The blocks which you access with the "GET" instruction were created with the access type "standard".
- Make sure that the areas defined with the parameters ADDR_i and SD_i match in terms of number, length and data type.
- The area to be read (ADDR_i parameter) cannot be larger than the area for data storage (RD_i parameter).

Parameters

The following table shows the parameters of the "GET" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Control parameter request, activates the data exchange on a positive edge.
ID	Input	WORD	I, Q, M, D, L or constant	Addressing parameters for specifying the connection to the partner CPU.
NDR	Output	BOOL	I, Q, M, D, L	Status parameter NDR: • 0: Job not yet started or still running. • 1: Job successfully completed.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameters ERROR and STATUS, error code: • ERROR=0 STATUS has the value: – 0000H: Neither warning nor error – <> 0000H: Warning, STATUS supplies detailed information. • ERROR=1 An error has occurred. STATUS supplies detailed information on the type of error.
STATUS	Output	WORD	I, Q, M, D, L	
ADDR_1	InOut	REMOTE	I, Q, M, D	Pointers to the areas on the partner CPU that are to be read. When the REMOTE pointer accesses a DB, the DB must always be specified. Example: P#DB10.DBX5.0 Byte 10.
ADDR_2	InOut	REMOTE		
ADDR_3	InOut	REMOTE		
ADDR_4	InOut	REMOTE		
RD_1	InOut	VARIANT	I, Q, M, D, L	Pointers to the areas on the local CPU in which the read data is entered.
RD_2	InOut	VARIANT		
RD_3	InOut	VARIANT		
RD_4	InOut	VARIANT		

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameters ERROR and STATUS

The following table contains all specific error information for the "GET" instruction that can be output via the ERROR and STATUS parameters.

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New job not active because the previous job is still busy.
0	25	Communication has started. The job is being processed.
1	1	Communication problems, for example • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	2	• Negative acknowledgment from the partner device. The function cannot be executed. • Response from the remote station exceeds the maximum user data length (see: Common parameters of instructions for S7 communication (Page 4385)). • Access protection is activated on the partner CPU. Deactivate access protection in the CPU settings.
1	4	Error in the pointers to the data storage RD_i: • Data types of the parameters RD_i and ADDR_i are not compatible with each other. • The length of the area RD_i is smaller than the length of the data of the ADDR_i parameter that is to be read.
1	8	Access error on the partner CPU.
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	20	• Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with low priority (first call).
1	W#16#80C3	(only with S7-1500) • Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with low priority (first call).

Note

Data consistency

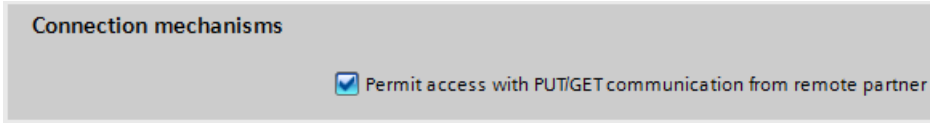
Data is received consistently if you read the part of the receive area RD_i currently being used completely before initiating another job.

Example

In the following example use an S7 connection and in CPU 1 read out a data record of CPU 2. The data record to be sent is of the data type INT.

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. The connection is not yet configured.
- A low protection level under “<CPU> > Properties > Protection“ ensures that read and write access is permitted for the CPUs.
- Access with the PUT/GET instruction is permitted.



Program of CPU 1

Create six tags in a global data block for storing the data of “GET“.

SLI_gDB_GET

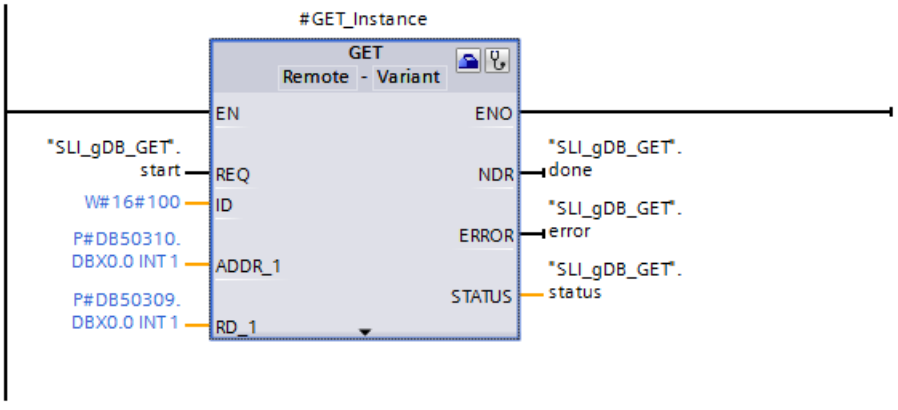
		Name	Data type	Start value
1		Static		
2		start	Bool	false
3		done	Bool	false
4		memDoneStat	Word	16#0
5		error	Bool	false
6		status	Word	16#0
7		memErrStatus	Word	16#0

Create one tag in a global data block for storing the data record to be received.

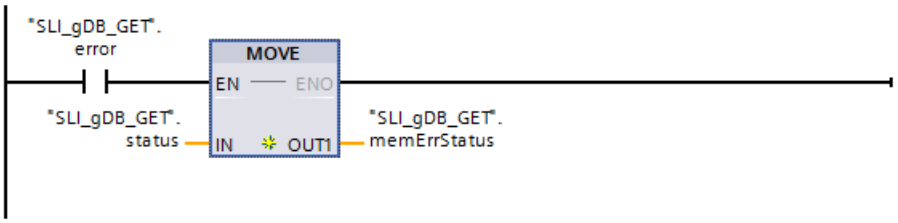
SLI_gDB_getData

		Name	Data type	Offset	Start value
1		Static			
2		readValue	Int	0.0	0

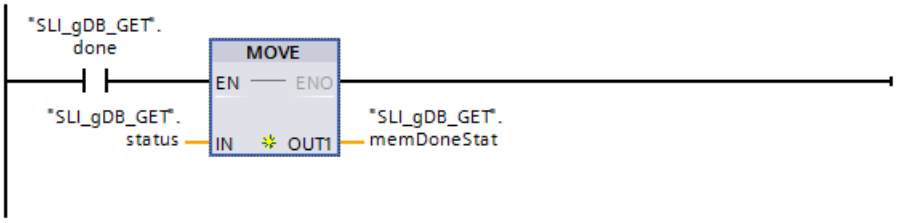
Network 1: Interconnect the parameters of the "GET" instruction as follows.



Network 2: Save the status for an error of GET as follows.



Network 3: Save the success status of GET as follows.



Configuration of GET

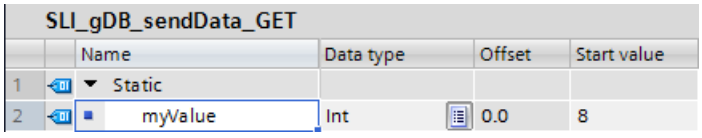
To interconnect the input parameter ID, open the wizard of the "GET" instruction with “Properties > Configuration“.

Make the following settings for the connection:

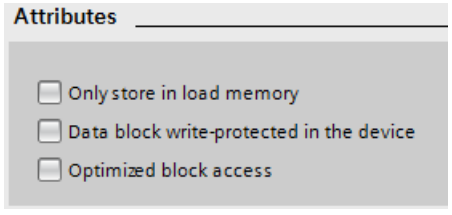
Section	Setting
End point	Select the communication partner. The remaining connection data will be entered automatically. An S7 connection is created automatically and its identifier entered at the input parameter ID.
Active	Make sure that the active connection goes out from CPU 1.

Program of CPU 2

Create one tag in a global data block for storing the data record to be sent.



In “Properties > Attributes“ of the data block disable the optimized block access.



Behavior of GET

When the input parameter REQ ("start") supplies the signal state "TRUE", the "GET" instruction is started. The instruction calls up the connection data and makes contact with the

communications partner of the S7 connection being used. To achieve this, the identifier of the S7 connection is stored at the input parameter ID.

Via the parameter ADDR_1 the storage location of the data record of CPU 2 to be sent (“myValue“) is identified. The data record (“myValue“) is read out and at the parameter RD_1 the identified storage location (“readValue“) is entered.

SLI_gDB_getData

		Name	Data type	Offset	Start value	Monitor value
1		Static				
2		readValue	Int	0.0	0	8

Successful processing is indicated by the output parameter NDR ("done") with the signal state "TRUE" and the output parameter STATUS ("status") with the value "16#0000". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "memDoneStat". The output parameter ERROR ("error") or the tag "memErrStatus" indicates that processing took place without errors.

SLI_gDB_GET

		Name	Data type	Start value	Monitor value
1		Static			
2		start	Bool	false	TRUE
3		done	Bool	false	FALSE
4		memDoneStat	Word	16#0	16#0000
5		error	Bool	false	FALSE
6		status	Word	16#0	16#0000
7		memErrStatus	Word	16#0	16#0000

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

PUT: Write data to a remote CPU (S7-1200, S7-1500)

Description

You can write data to a remote CPU with the instruction "PUT".

The instruction is started on a positive edge at control input REQ:

- The pointers to the areas to be written (ADDR_i) and the data (SD_i) are then sent to the partner CPU. The partner CPU can be in RUN or STOP mode.
- The data to be sent is copied from the configured send areas ((SD_i). The partner CPU saves the sent data under the addresses supplied with the data and returns an execution acknowledgment.
- If no errors occur, this is indicated at the next instruction call with status parameter DONE = "1". The writing process can only be activated again after the last job is complete.

Errors and warnings are output via ERROR and STATUS if access problems occurred while the data was being written or if the execution check results in an error.

Requirements for using the instruction

- The "Permit access with PUT/GET communication from remote partner" function was activated in the properties of the partner CPU under "Protection".
- The blocks which you access with the "PUT" instruction were created with the access type "standard".
- Make sure that the areas defined with the parameters ADDR_i and SD_i match in terms of number, length and data type.
- The area to be written (ADDR_i parameter) must be as large as the send area (SD_i parameter).

Parameters

The following table shows the parameters of the "PUT" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Control parameter request, activates the data exchange on a positive edge.
ID	Input	WORD	I, Q, M, D, L or constant	Addressing parameters for specifying the connection to the partner CPU.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter DONE: • 0: Job not yet started or still executing • 1: Job executed without errors.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameters ERROR and STATUS, error code: • ERROR=0 STATUS has the value: – 0000H: Neither warning nor error – <> 0000H: Warning, STATUS supplies detailed information. • ERROR=1 An error has occurred, STATUS supplies detailed information on the type of error.
STATUS	Output	WORD	I, Q, M, D, L	
ADDR_1	InOut	REMOTE	I, Q, M, D	Pointers to the areas on the partner CPU to which the data will be written. When the REMOTE pointer accesses a DB, the DB must always be specified. Example: P#DB10.DBX5.0 Byte 10. When transferring data structures (e.g., Struct) the data type CHAR must be used at the ADDR_i parameters.
ADDR_2	InOut	REMOTE		
ADDR_3	InOut	REMOTE		
ADDR_4	InOut	REMOTE		

Parameter	Declaration	Data type	Memory area	Description
SD_1	InOut	VARIANT	I, Q, M, D, L	Pointers to the areas on the local CPU which contain the data to be sent. Only the data types BOOL, BYTE, CHAR, WORD, INT, DWORD, DINT, REAL are permitted. When transferring data structures (e.g., Struct) the data type CHAR must be used at the SD_i parameters.
SD_2	InOut	VARIANT		
SD_3	InOut	VARIANT		
SD_4	InOut	VARIANT		

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

ERROR and STATUS parameters

The following table contains all specific error information for the "PUT" instruction that can be output via the ERROR and STATUS parameters.

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New job not active because the previous job is still busy.
0	25	Communication has started. The job is being processed.
1	1	Communication problems, for example • Connection description not loaded (local or remote). • Connection interrupted (for example, cable, CPU off, CP in STOP mode). • Connection to partner not yet established.
1	2	• Negative acknowledgment of partner CPU. The function cannot be executed. • Access on the partner CPU was not granted. Activate the access in the CPU settings.
1	4	Error in the pointers to the data storage: • Data types of the parameters SD_i and ADDR_i are not compatible with each other. • The length of the area SD_i is greater than the length of the data of the ADDR_i parameter that is to be written. • Not possible to access SD_i. • Maximum user data size exceeded. • Number of the parameters SD_i and ADDR_i does not match.
1	8	Access error with the partner CPU (e.g. DB not loaded or write-protected).
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	20	• Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with low priority (first call).
1	W#16#80C3	(only with S7-1500) • Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with low priority (first call).

Data consistency

When a send operation is activated (positive edge at REQ), the data to be sent from the send area SD_i is copied from the user program. After the block call, you can write to these areas without corrupting the current send data.

Note

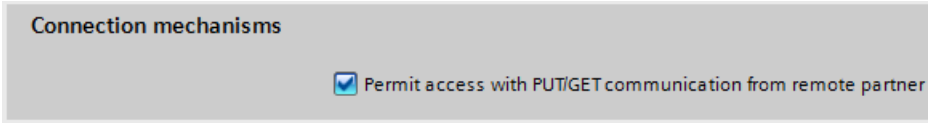
The send operation is only complete when the DONE status parameter has the value "1".

Example

In the following example use an S7 connection and transfer a data record from CPU 1 to CPU 2. The data record to be sent is of the data type INT.

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. The connection is not yet configured.
- A low protection level under “<CPU> > Properties > Protection“ ensures that read and write access is permitted for the CPUs.
- Access with the PUT/GET instruction is permitted.



Program of CPU 1

Create six tags in a global data block for storing the data of “PUT“.

SLI_gDB_PUT

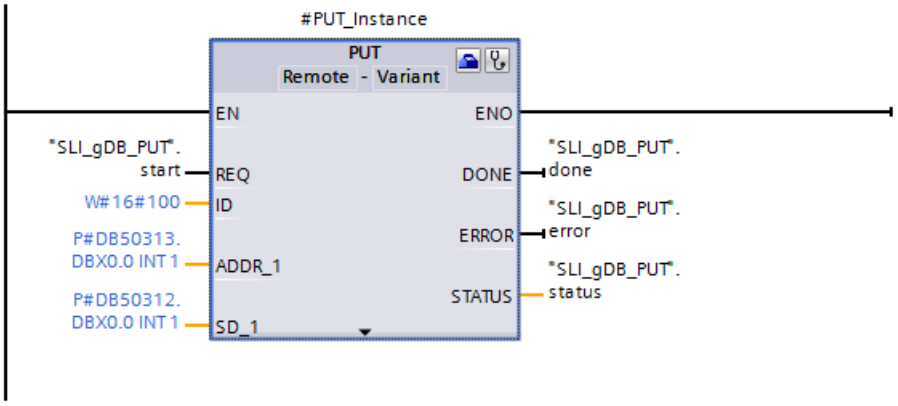
		Name	Data type	Start value
1		Static		
2		start	Bool	false
3		done	Bool	false
4		memDoneStat	Word	16#0
5		error	Bool	false
6		status	Word	16#0
7		memErrStatus	Word	16#0

Create one tag in a global data block for storing the data record to be sent.

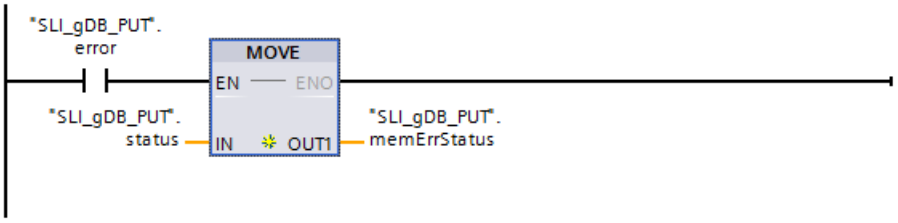
SLI_gDB_putData

		Name	Data type	Offset	Start value
1		Static			
2		writeValue	Int	0.0	8

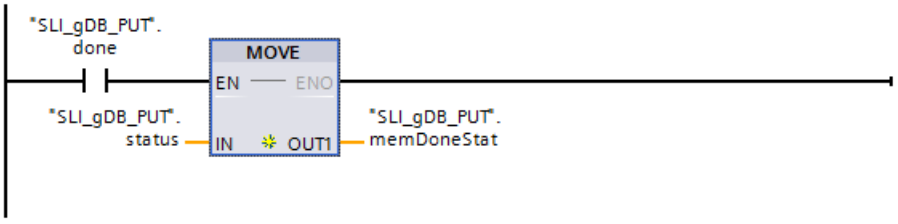
Network 1: Interconnect the parameters of the "PUT" instruction as follows.



Network 2: Save the status for an error of PUT as follows.



Network 3: Save the success status of PUT as follows.



Configuration of PUT

To interconnect the input parameter ID, open the wizard of the "PUT" instruction with “Properties > Configuration“.

Make the following settings for the connection:

Section	Setting
End point	Select the communication partner. The remaining connection data will be entered automatically. An S7 connection is created automatically and its identifier entered at the input parameter ID.
Active	Make sure that the active connection goes out from CPU 1.

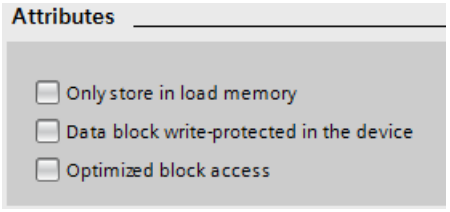
Program of CPU 2

Create one tag in a global data block for storing the data record to be received.

SLI_gDB_rcvData_PUT

	Name	Data type	Offset	Start value
1	Static			
2	myValue	Int	0.0	0

In “Properties > Attributes“ of the data block disable the optimized block access.



Behavior of PUT

When the input parameter REQ ("start") returns the signal state "TRUE", the "PUT" instruction is started. The instruction calls up the connection data and makes contact with the communications partner of the S7 connection being used. To achieve this, the identifier of the S7 connection is stored at the input parameter ID.

At the parameter SD_1 the storage location of the data record of CPU 1 to be sent (“writeValue“) is identified. At the parameter ADDR_1 the storage location (“myValue“) of the data record in CPU 2 is identified. The data record (“writeValue“) is transferred and indicated at the storage location of CPU 2 (“myValue“).

SLI_gDB_rcvData_PUT

	Name	Data type	Offset	Start value	Monitor value
1	Static				
2	myValue	Int	0.0	0	8

Successful processing is indicated by the output parameter DONE ("done") with the signal state "TRUE" and the output parameter STATUS ("status") with the value "16#0000". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "memDoneStat". The output parameter ERROR ("error") or the tag "memErrStatus" indicates that processing took place without errors.

SLI_gDB_PUT

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	done	Bool	false	FALSE
4	memDoneStat	Word	16#0	16#0000
5	error	Bool	false	FALSE
6	status	Word	16#0	16#0000
7	memErrStatus	Word	16#0	16#0000

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

Other (S7-1200, S7-1500)

USEND: Send data uncoordinated (S7-1500)

Description

The "USEND" instruction sends data to a remote partner instruction of type "URCV (Page 4401)". The sending process is carried out without coordination with the partner instruction. This means that the data transfer is carried out without acknowledgment by the partner instruction.

When a send operation is activated (positive edge at REQ), the data to be sent from the send areas SD_i are copied from the user program. After the instruction call, you can write to these send areas again without corrupting the current send data.

Successful completion of the send operation is indicated by the value of the DONE status parameter being set to "1".

Parameters

The following table shows the parameters of the instruction "USEND":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Control parameter request, activates the data exchange on a positive edge.
ID	Input	CONN_PRG	I, Q, M, D, L, P or constant	Addressing parameters for specifying the connection to the partner CPU.
R_ID	Input	CONN_R_ID	I, Q, M, D, L or constant	Addressing parameter R_ID for defining the instruction pairs "USEND" and "URCV". See also: Common parameters of instructions for S7 communication (Page 4385)
DONE	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Job not yet started or still executing. • 1: Job executed without errors.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Neither warning nor error. • 1: An error has occurred. STATUS supplies detailed information on the type of error.

Parameter	Declaration	Data type	Memory area	Description
STATUS	Output	WORD	I, Q, M, D, L	Status parameter See "ERROR and STATUS parameters" table.
SD_i (1≤ i ≤4)	InOut	VARIANT	I, Q, M, D, L	Pointer on the i-th send area. Only the BOOL data types are permitted (not permitted: bit array), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL or STRUCT. The maximum user data size for SD_i parameters depends on the partner CPU ("URCV" instruction) and on the number of parameters used. Additional information is available in: Common parameters of instructions for S7 communication (Page 4385)

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameters ERROR and STATUS

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New job not active because the previous job is still busy.
0	25	Communication has started. The job is being processed.
1	1	Communication problem has occurred. Possible causes: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	4	• Errors in the send area pointers SD_i relating to the data length or the data type. • Maximum user data length was exceeded.
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	18	Value at R_ID parameter already exists in the connection specified at the ID parameter (R_ID value must be unique for the connection).
1	20	• Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with lower priority (first call).
1	W#16#80C3	(only with S7-1500) • Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with low priority (first call).

Example

You can find the example here: Program example for USEND & URCV (Page 4403).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

URCV: Receive data uncoordinated (S7-1500)

Description

The "URCV" instruction receives data asynchronously from a remote partner instruction of type "USEND (Page 4399)" and copies this data into the configured receive areas.

The instruction is ready to receive if there is a logical 1 at the EN_R input. An active job can be canceled with EN_R=0.

The receive data areas are referenced by the parameters RD_1 to RD_4. Make sure that the areas defined by the parameters RD_i / RD_1 and SD_i / SD_1 (with the associated partner instruction "USEND (Page 4399)") match in terms of number and length.

Successful completion of the copy operation is indicated when the value of the NDR status parameter is set to logical "1". Once the status parameter NDR has changed to "1", there will be new receive data in your receive areas (RD_i). A new block call may cause these data to be overwritten with new receive data. If you want to prevent this, call "URCV" (for example, during cyclic block processing) with the value 0 at EN_R until you have finished processing the received data.

Parameters

The following table shows the parameters of the instruction "URCV":

Parameter	Declaration	Data type	Memory area	Description
EN_R	Input	BOOL	I, Q, M, D, L or constant	Control parameter enabled to receive, signals ready to receive if the input is set.
ID	Input	CONN_PRG	I, Q, M, D, L, P or constant	Addressing parameters for specifying the connection to the partner CPU.
R_ID	Input	CONN_R_ID	I, Q, M, D, L or constant	Addressing parameter for defining the instruction pairs "USEND" and "URCV". See also: Common parameters of instructions for S7 communication (Page 4385)
NDR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Job not yet started or still running. • 1: Job successfully completed.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Neither warning nor error • 1: An error has occurred, STATUS supplies detailed information on the type of error.

Parameter	Declaration	Data type	Memory area	Description
STATUS	Output	WORD	I, Q, M, D, L	Status parameter See "ERROR and STATUS parameters" table.
RD_i (1≤ i ≤4)	InOut	VARIANT	I, Q, M, D, L	Pointer to the i-th receive area: Only the BOOL data types are permitted (not permitted: bit array), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL or STRUCT. Additional information is available in: Common parameters of instructions for S7 communication (Page 4385)

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameters ERROR and STATUS

ERROR	STATUS (decimal)	Explanation
0	9	Warning: older received data were overwritten by newer received data.
0	11	Warning: The received data is already being processed in a priority class with lower priority (error can occur when copying data to the receive area).
0	25	Communication has started. The job is being processed.
1	1	Communication problem has occurred. Possible causes: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	4	Errors in the receive area pointers RD_i relating to the data length or the data type.
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	18	Value at R_ID parameter already exists in the connection specified at the ID parameter (R_ID value must be unique for the connection).
1	19	The associated "USEND (Page 4399)" instruction sends data faster than it can be copied to the receive areas by "URCV".
1	20	• Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with lower priority (first call).
1	W#16#80C3	(only with S7-1500) • Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with low priority (first call).

Example

You can find the example here: Program example for USEND & URCV (Page 4403).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

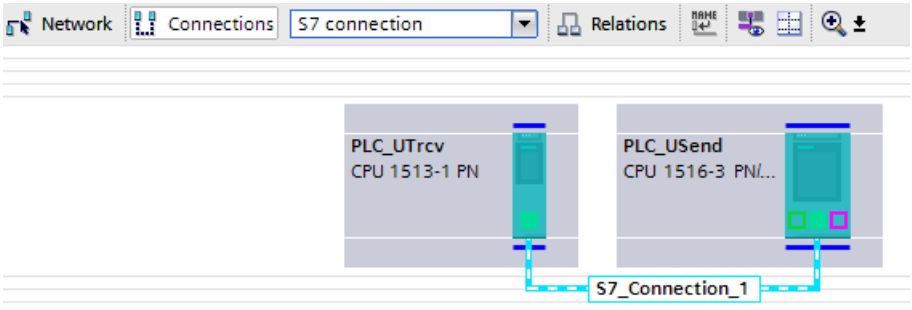
Program example for USEND & URCV (S7-1200, S7-1500)

Introduction

In the following example you create an S7 connection between two CPUs of the S7-1500 series. You send a character string (data type STRING) from CPU 1 to CPU 2. The data transfer is carried out without acknowledgement by the communication partner.

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. An S7 connection is configured.



- A low protection level under "<CPU> > Properties > Protection" ensures that read and write access is permitted for the CPUs.

Program of CPU 1

Create the following tags for storing the data of USEND in a global data block.

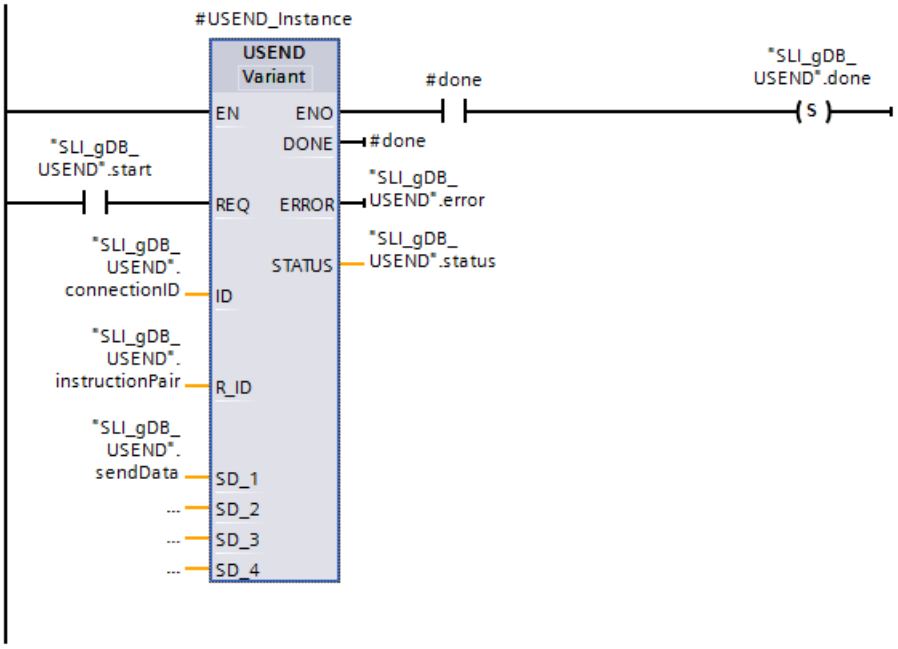
SLI_gDB_USEND

	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	connectionID	CONN_PRG	16#0100
4	instructionPair	CONN_R_ID	16#01
5	done	Bool	false
6	error	Bool	false
7	status	Word	16#0
8	memErrStatus	Word	16#0
9	sendData	String	'HelloData'

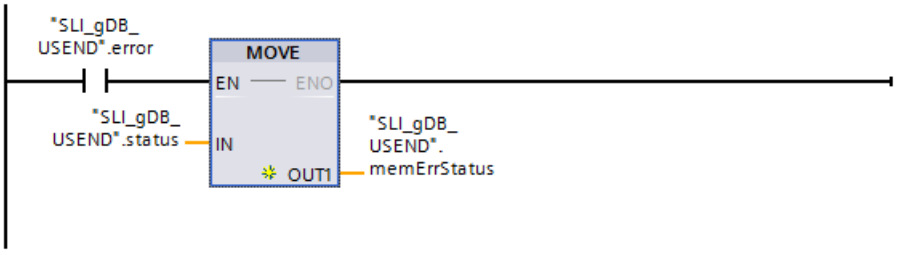
Create an FB "SLI_FB_USEND". Create the following local tags in it.

7	Static			
8	USEND_Instance	USEND		
9	done	Bool	false	Non-ret...

Network 1: Interconnect the parameter of the instruction "USEND" as follows:



Network 2: In case of error of USEND, save the status as follows.



Program of CPU 2

Create the following tags for storing the data of URCV in a global data block.

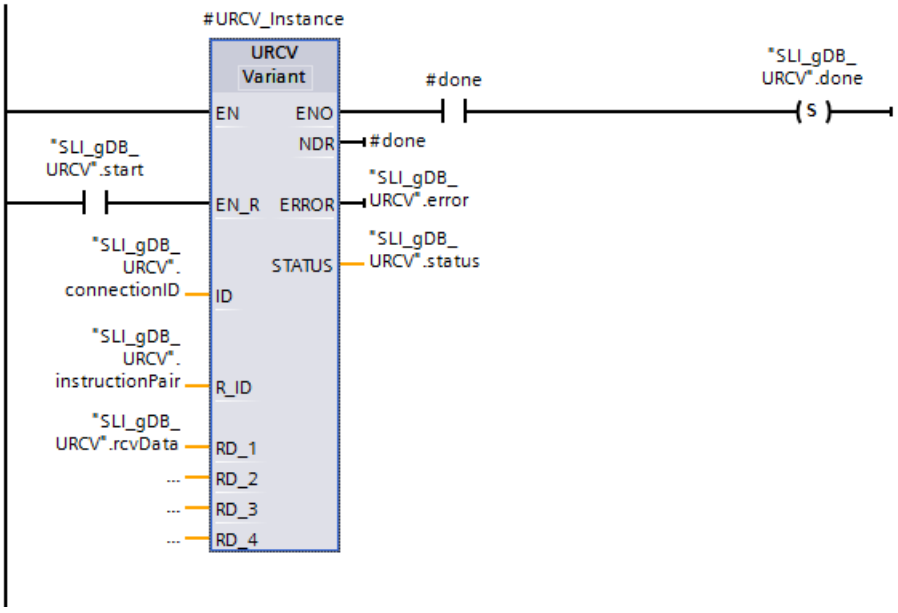
SLI_gDB_URCV

	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	connectionID	CONN_PRG	16#0100
4	instructionPair	CONN_R_ID	16#01
5	done	Bool	false
6	error	Bool	false
7	status	Word	16#0
8	memErrStatus	Word	16#0
9	rcvData	String	''

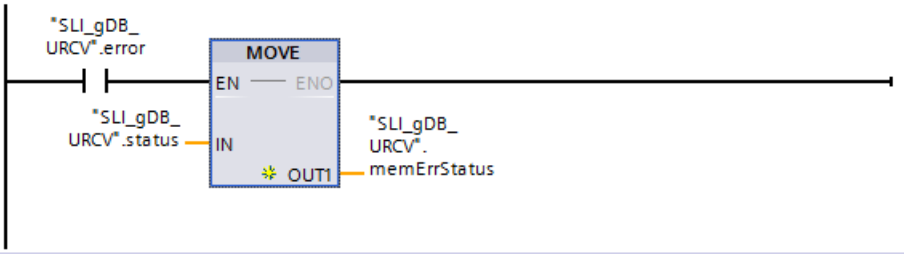
Create an FB "SLI_FB_URCV". Create the following local tags in it.



Network 1: Interconnect the parameter of the instruction "URCV" as follows:



Network 2: In case of error of URCV, save the status as follows.



Assigning communication connection

The addressing parameters for connecting the two CPUs and for defining the instruction pair must be uniform.

- In each case store the hexadecimal value of the hardware ID of the configured S7 connection at the input parameter ID ("connectionID").
  You can find the hardware identifier in the "Network view" under "Connections".

Network overview | Connections | Relations | I/O communication | VPN

Local connection n...	Local end point	Local ID (hex)	Partner ID (hex)	Partner	Connection type
S7_Connection_1	PLC_UTrcv	100	100	PLC_USend	S7 connection
S7_Connection_1	PLC_USend	100	100	PLC_UTrcv	S7 connection

- In each case store a self-selected identifier (as hex value, in byte size) for the instruction pair at the input parameter R_ID ("instructionPair"). The identifier must not be already assigned for another instruction pair.

Behavior of CPU 1

When the input parameter REQ ("start") supplies the signal state "TRUE", the "USEND" instruction is started. The "USEND" instruction hereby copies the data record detected at input parameter SD_1 ("sendData") to the work memory of the CPU. Successful copying of the data record is indicated by the output parameter DONE ("#done") with the signal state "TRUE". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "done".

Corresponding to the input parameter ID ("connectionID") and R_ID ("instructionPair"), the instruction "URCV" of the CPU 2 as receiver of the data record ("sendData" is addressed with the value "HelloData"). Data transfer via S7 connection continues without further participation by USEND ("start" may be set to "FALSE").

The output parameter ERROR ("error") or the tag "memErrStatus" indicates that processing of USEND in the example is taking place without errors.

SLI_gDB_USEND

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	connectionID	CONN_PRG	16#0100	16#0100
4	instructionPair	CONN_R_ID	16#01	16#0000_0001
5	done	Bool	false	TRUE
6	error	Bool	false	FALSE
7	status	Word	16#0	16#0000
8	memErrStatus	Word	16#0	16#0000
9	sendData	String	'HelloData'	'HelloData'

Behavior of CPU 2

CPU 2 receives a data record of CPU 1 via the S7 connection. Corresponding to the input parameter ID ("connectionID") and R_ID ("instructionPair"), the instruction "URCV" knows the receiver of the data record.

When the input parameter EN_R ("start") supplies the signal state "TRUE", the "URCV" instruction is started and is ready to receive. The instruction "URCV" reads out the transferred data and copies the data record into the value area at the input parameter RD_1 ("rcvData").

Successful copying of the data record is indicated by the output parameter NDR ("#done") with the signal state "TRUE". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "done".

The output parameter ERROR ("error") or the tag "memErrStatus" indicates that processing of URCV in the example is taking place without errors.

SLI_gDB_URCV

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	connectionID	CONN_PRG	16#0100	16#0100
4	instructionPair	CONN_R_ID	16#01	16#0000_0001
5	done	Bool	false	TRUE
6	error	Bool	false	FALSE
7	status	Word	16#0	16#0000
8	memErrStatus	Word	16#0	16#0000
9	rcvData	String	''	'HelloData'

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

USEND: Send data uncoordinated (Page 4399)

URCV: Receive data uncoordinated (Page 4401)

BSEND: Send data in segments (S7-1500)

Description

The "BSEND" instruction sends data to a remote partner instruction of type "BRCV (Page 4410)". With this type of data transfer, more data can be transported between the communication partners than is possible with all other communication instructions for configured S7 connections. In each case, the maximum data volume amounts to 65534 bytes (standard access) or 65535 bytes (optimized access) for the integrated interface and for SIMATIC Net CP.

Functional description

You specify the instruction pair "BSEND" and "BRCV" with the input parameter R_ID. The R_ID parameter must be identical in the related instructions.

The send job is activated after the instruction is called and a positive edge is detected at control input REQ. After the call, "BSEND" is not processed in the background, which means the data can only be read within the user program.

The data area to be transmitted is segmented. Each segment is sent individually to the partner. Each segment is acknowledged by the partner after the application of this segment by "BRCV (Page 4410)". If the data is segmented, "BSEND" must be called multiple times until all segments have been transferred.

The data area of the data to be sent is specified by SD_1. To ensure data consistency, you may only write to the part of the currently used sending area SD_1 after the current send operation is complete. This is the case when the value of the status parameter DONE changes to "1".

The length of the send data is specified by LEN based on the job. With LEN = "0", all data addressed with the SD_1 parameter is sent.

If there is a positive edge at control input R, the current sending process is canceled.

Due to the asynchronous data transfer, a new data transfer can only be initiated if the previous data has been fetched by a partner instruction call. When the data has been collected, the status parameter "NDR" is set in the partner instruction "BRCV".

Note

Migration of S7-400 user programs

An S7-400 CPU interprets the parameter SD_1 as pointer and not as data area.

With S7-1500, LEN may not exceed the area of SD1. This was permitted with S7-400. Recommendation: Use the maximum size for the LEN parameter (65534 bytes for the integrated interface) as size of the data area at the SD_1 parameter.

Parameters

The following table shows the parameters of the instruction "BSEND":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Control parameter request, activates the data exchange at a positive edge
R	Input	BOOL	I, Q, M, D, L or constant	Control parameter reset, activates an abort of the current data exchange at a positive edge
ID	Input	CONN_PRG	I, Q, M, D, L, P or constant	Addressing parameters for specifying the connection to the partner CPU.
R_ID	Input	CONN_R_ID	I, Q, M, D, L or constant	Addressing parameter for defining the instruction pairs "BSEND" and "BRCV (Page 4410)". See also: Common parameters of instructions for S7 communication (Page 4385)
SD_1	InOut	VARIANT	I, Q, M, D, L	Pointer to send area When transferring structures, the structures must be identical at the sending and receiving end.

Parameter	Declaration	Data type	Memory area	Description
LEN	InOut	WORD	I, Q, M, D, L	Length of the block of data to be sent in bytes. With LEN = "0", all data is sent by SD_1.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Job not yet started or still executing. • 1: Job executed without errors.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Neither warning nor error. • 1: An error has occurred, STATUS supplies detailed information on the type of error.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter See "ERROR and STATUS parameters" table.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameters ERROR and STATUS

The following table contains all specific error information for "BSEND" that can be output via the ERROR and STATUS parameters.

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New job not active because the previous job is still busy.
0	25	Communication has started. The job is being processed.
1	1	Communication problem has occurred. Possible causes: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	2	Negative acknowledgment of partner instruction. The instruction cannot be executed.
1	3	R_ID is unknown on the connection specified by the ID or the receive block has not yet been called.
1	4	• Error in the data length or data type in the send area pointer SD_1. • Value for LEN is greater than area SD_1.
1	5	Reset request was executed.
1	6	Partner instruction is in DISABLED status (EN_R has the value "0"). Also check the input parameters of "BRCV (Page 4410)" for consistency with "BSEND".
1	7	"BRCV (Page 4410)" partner instruction not called since the last data transfer.
1	8	Access to remote object in the user memory was rejected: The target range for the associated "BRCV (Page 4410)" is too small. ERROR = 1, STATUS = 4 or ERROR = 1, STATUS = 10 reported at the output parameters of "BRCV (Page 4410)".
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	18	R_ID already exists in the connection.

ERROR	STATUS (decimal)	Explanation
1	20	• Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with lower priority (first call).
1	W#16#80C3	(only with S7-1500) • Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with low priority (first call).

Example

You can find the example here: Program example for BSEND & BRCV (Page 4412).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

BRCV: Receive data in segments (S7-1500)

Description

The "BRCV" instruction receives data from a remote partner instruction of type "BSEND (Page 4407)". The R_ID parameter must be identical in the related instructions.

The instruction is ready to receive data (STATUS = 25) after it is called with the value "1" at control input EN_R. An active job can be canceled with EN_R=0.

The maximum receive area is specified by RD_1. Data will be received consistently if you fully evaluate the part of the RD_1 receive area currently in use before calling the block again with value 1 at control input EN_R.

After each received data segment, an acknowledgment is sent to the partner instruction. If there are multiple segments it will be necessary to call "BRCV" several times until all of the segments have been received. Asynchronous data receipt is indicated by STATUS = 17. The amount of data currently received is indicated at the parameter LEN. The RD_1 parameter must remain constant during the operation.

Value "1" at the NDR status parameter indicates that all data segments have been received without error. The received data remains unchanged until the next call with EN_R=1.

Parameters

The following table shows the parameters of the instruction "BRCV":

Parameter	Declaration	Data type	Memory area	Description
EN_R	Input	BOOL	I, Q, M, D, L or constant	Control parameter enabled to receive, signals ready to receive if the input is set.
ID	Input	CONN_PRG	I, Q, M, D, L, P or constant	Addressing parameters for specifying the connection to the partner CPU.
R_ID	Input	CONN_R_ID	I, Q, M, D, L or constant	Addressing parameter for defining the instruction pairs "BSEND (Page 4407)" and "BRCV". See also: Common parameters of instructions for S7 communication (Page 4385)
RD_1	InOut	VARIANT	I, Q, M, D, L	Pointer to the receive area. When transferring structures, the structures must be identical at the sending and receiving end.
LEN	InOut	WORD	I, Q, M, D, L	Length of the data already received in bytes.
NDR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Job not yet started or still running. • 1: Job successfully completed.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter • 0: Neither warning nor error • 1: An error has occurred, STATUS supplies detailed information on the type of error.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter See "ERROR and STATUS parameters" table.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameters ERROR and STATUS

The following table contains all specific error information for "BRCV" that can be output via the ERROR and STATUS parameters.

ERROR	STATUS (decimal)	Explanation
0	17	Warning: Instruction receives data asynchronously. The LEN parameter shows the amount of data already received in bytes.
0	25	Communication has started. The job is being processed.
1	1	Communication problem has occurred. Possible causes: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	2	Function cannot be executed (protocol error)
1	4	Error in the receive area pointer RD_1 relating to the data length or data type. The block of data sent is longer than the receive area.
1	5	Reset request received, incomplete transfer.

ERROR	STATUS (decimal)	Explanation
1	8	Access error in associated "BSEND (Page 4407)": After the last valid data segment has been sent, ERROR = 1 and STATUS = 4 or ERROR = 1 and STATUS = 10 is reported.
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	18	Value at R_ID parameter already exists in the connection specified at the ID parameter (R_ID value must be unique for the connection).
1	20	• Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with lower priority (first call).
1	W#16#80C3	(only with S7-1500) • Maximum number of parallel jobs exceeded. • The job is currently being processed in a priority class with low priority (first call).

Example

You can find the example here: Program example for BSEND & BRCV (Page 4412).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

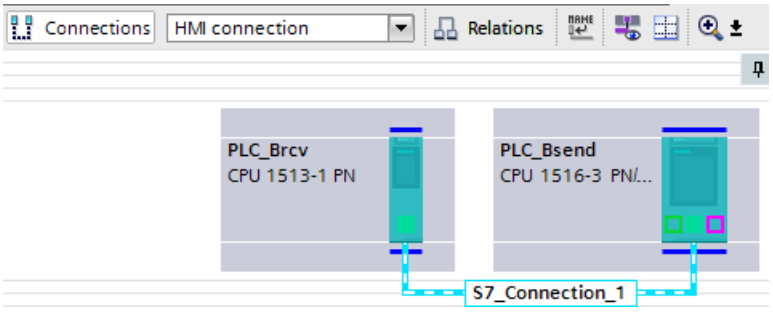
Program example for BSEND & BRCV (S7-1200, S7-1500)

Introduction

In the following example you create an S7 connection between two CPUs of the S7-1500 series. You send a data record from CPU 1 to CPU 2 using BSEND und BRCV. The data transfer is carried out with acknowledgement by BRCV.

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. An S7 connection is configured.



- A low protection level under "<CPU> > Properties > Protection" ensures that read and write access is permitted for the CPUs.

Program of CPU 1

Create the following tags for storing the data of BSEND in a global data block.

SLI_gDB_BSEND

	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	reset	Bool	false
4	connectionID	CONN_PRG	16#0100
5	instructionPair	CONN_R_ID	16#02
6	maxLength	UInt	0
7	done	Bool	false
8	error	Bool	false
9	status	Word	16#0
10	memErrStatus	Word	16#0

For the data record, create the following PLC data type.

BSEND_User

	Name	Data type	Default value
1	prename	String	''
2	lastname	String	''
3	userId	UInt	0

For the data transfer, create the following data block based on the created PLC data type ("BSEND_User").

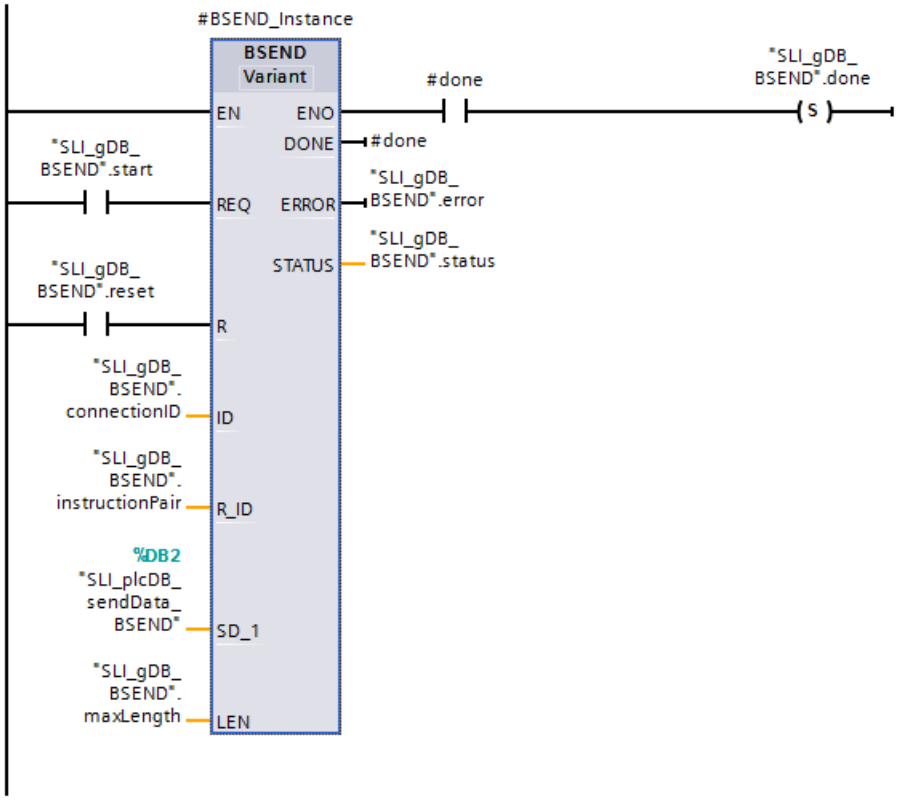
SLI_plcDB_sendData_BSEND

	Name	Data type	Start value
1	Static		
2	prename	String	'Simon'
3	lastname	String	'Sample'
4	userId	UInt	1

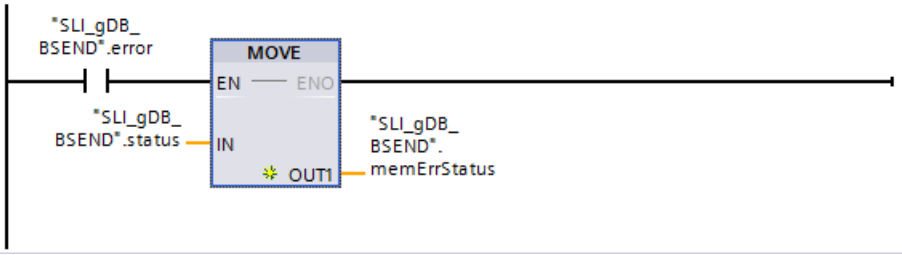
Create an FB "SLI_FB_BSEND". Create the following local tags in it.



Network 1: Interconnect the parameter of the instruction "BSEND" as follows:



Network 2: In case of error of BSEND, save the status as follows.



Program of CPU 2

Create the following tags for storing the data of BRCV in a global data block.

SLI_gDB_BRCV

	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	connectionID	CONN_PRG	16#0100
4	instructionPair	CONN_R_ID	16#02
5	done	Bool	false
6	error	Bool	false
7	status	Word	16#0
8	memErrStatus	Word	16#0
9	readLength	UInt	0

For the data record, create the following PLC data type.

BSEND_User

	Name	Data type	Default value
1	prename	String	''
2	lastname	String	''
3	userId	UInt	0

For the data transfer, create the following data block based on the created PLC data type ("BSEND_User").

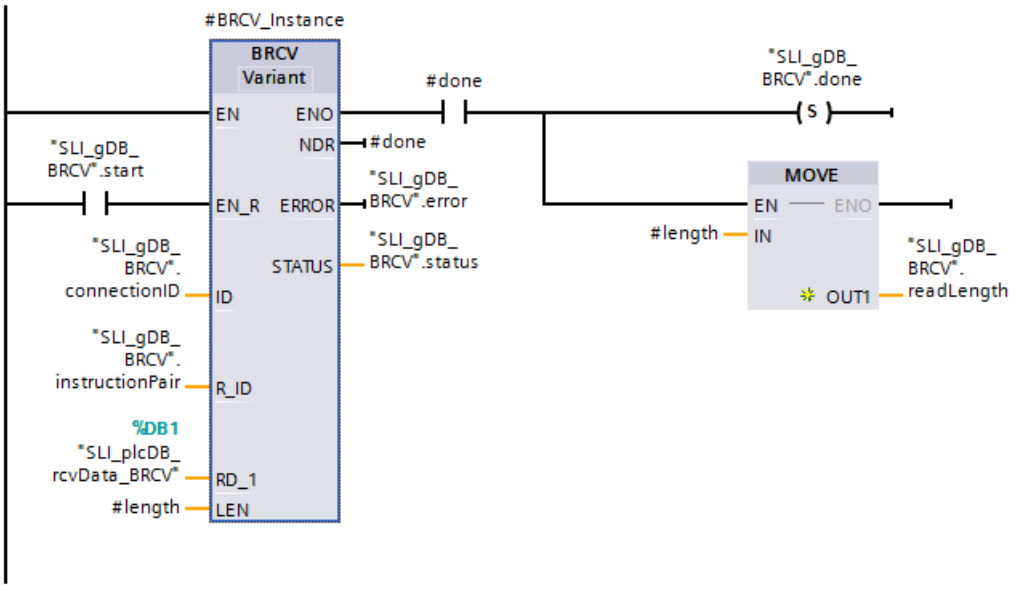
SLI_plcDB_rcvData_BRCV

	Name	Data type	Start value
1	Static		
2	prename	String	''
3	lastname	String	''
4	userId	UInt	0

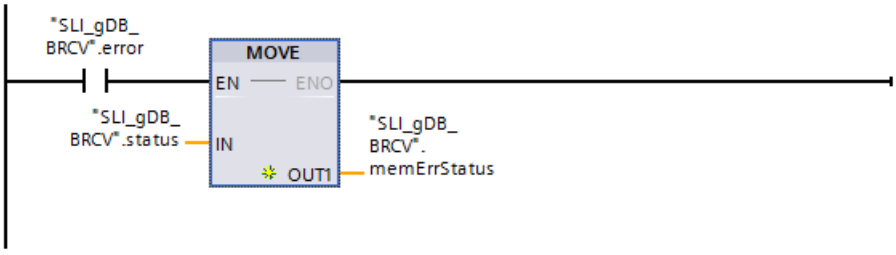
Create an FB "SLI_FB_BSEND". Create the following local tags in it.

7	Static			
8	done	Bool	false	Non-ret...
9	length	UInt	0	Non-retain
10	BRCV_Instance	BRCV		

Network 1: Interconnect the parameter of the instruction "BRCV" as follows:



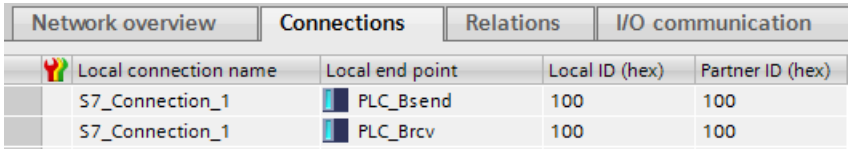
Network 2: In case of error of BRCV, save the status as follows.



Assigning communication connection

The addressing parameters for connecting the two CPUs and for defining the instruction pair must be uniform.

- In each case store the hexadecimal value of the hardware ID of the configured S7 connection at the input parameter ID ("connectionID").
  You can find the hardware identifier in the "Network view" under "Connections".



- In each case store a self-selected identifier (as hex value, in byte size) for the instruction pair at the input parameter R_ID ("instructionPair"). The identifier must not be already assigned for another instruction pair.

Behavior of CPU 1

Corresponding to the input parameter ID ("connectionID") and R_ID ("instructionPair"), the instruction "BRCV" of the CPU 2 is addressed as receiver of the data record ("SLI_plcDB_sendData_BSEND").

When the input parameter REQ ("start") supplies the signal state "TRUE", the "BSEND" instruction is started. For multiple calls, the "BSEND" instruction transmits the data record detected at input parameter SD_1 ("SLI_plcDB_sendData_BSEND") in segments to the CPU 2. According to the value "0" of the input parameter LEN ("maxLength"), is the length of the data record.

Successful transfer of the data record () is indicated by the output parameter DONE ("#done") with the signal state "TRUE". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "done".

The output parameter ERROR ("error") or the tag "memErrStatus" indicates that processing in the example is taking place without errors.

SLI_gDB_BSEND

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	reset	Bool	false	FALSE
4	connectionID	CONN_PRG	16#0100	16#0100
5	instructionPair	CONN_R_ID	16#02	16#0000_0002
6	maxLength	UInt	0	0
7	done	Bool	false	TRUE
8	error	Bool	false	FALSE
9	status	Word	16#0	16#0000
10	memErrStatus	Word	16#0	16#0000

A new data transfer can only be initiated if the previous data has been fetched by calling of BRCV.

Behavior of CPU 2

CPU 2 receives a data record (in segments) of CPU 1 via the S7 connection. Corresponding to the input parameter ID ("connectionID") and R_ID ("instructionPair"), the instruction "BRCV" knows the receiver of the data record.

When the input parameter EN_R ("start") supplies the signal state "TRUE", the "BRCV" instruction is started and is ready to receive. Over several calls the "BRCV" instruction reads out the data record in segments and saves the data record at the input parameter RD_1 ("SLI_plcDB_rcvData_BRCV").

SLI_plcDB_rcvData_BRCV

	Name	Data type	Start value	Monitor value
1	Static			
2	prename	String	''	'Simon'
3	lastname	String	''	'Sample'
4	userId	UInt	0	1

Successful receipt of the entire data record is indicated by the output parameter NDR ("#done") with the signal state "TRUE". The length in BYTES of the actually transmitted data record is

detected by the output parameter LEN ("#length"). This value is only displayed during the success status. Then "0" is detected. Because the values of the output parameters are only displayed for the moment of their validity, save the success status of "#done" in the tag "done" and the length in BYTES in the tag "readLength".

The output parameter ERROR ("error") or the tag "memErrStatus" indicates that processing in the example is taking place without errors.

SLI_gDB_BRCV

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	connectionID	CONN_PRG	16#0100	16#0100
4	instructionPair	CONN_R_ID	16#02	16#0000_0002
5	done	Bool	false	TRUE
6	error	Bool	false	FALSE
7	status	Word	16#0	16#0019
8	memErrStatus	Word	16#0	16#0000
9	readLength	UInt	0	514

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

BSEND: Send data in segments (Page 4407)

BRCV: Receive data in segments (Page 4410)

11.4.6.2 Open User Communication (S7-1200, S7-1500)

Data consistency (S7-1200, S7-1500)

Ensuring data consistency

In the S7-1500 module series, you can change parameters during ongoing operation with the instructions of the Open User Communication. Changed parameters are used immediately, even if a job is still active. Both can lead to data inconsistency! To avoid data inconsistencies, you should never change parameters while a job is running.

TSEND_C: Establishing a connection and sending data (S7-1200, S7-1500)

TSEND_C: Send data via Ethernet (S7-1200)

Validity

The following description of the "TSEND_C" instruction applies to S7-1200 CPUs with firmware version < V4.0.

Description

The "TSEND_C" instruction sets up and establishes a TCP or ISO-on-TCP communication connection. Once the connection has been set up and established, it is automatically maintained and monitored by the CPU. The connection description specified at the CONNECT parameter is used to set up the communication connection.

The instruction is executed asynchronously and has the following functions:

- A communication connection is set up and established:
  The communication connection is set up and established with CONT=1. Once the connection is successfully established, the DONE parameter is set to "1" for one cycle. An existing connection is terminated and the connection which has been set up is removed when the CPU goes into STOP mode. To set up and establish the connection again, you must execute "TSEND_C" again. For information on the number of possible communication connections, refer to the technical specifications for your CPU.
- Sending data via an existing communication connection:
  You specify the send area with the DATA parameter. This includes the address and the length of the data to be sent. Do not use a data area with the data type BOOL or Array of BOOL at the DATA parameter. If you use purely symbolic values at the DATA parameter, the LEN parameter must have the value "0".
- The send job is executed when a rising edge is detected at the REQ parameter. With the LEN parameter, you specify the maximum number of bytes sent with a send job. When sending data (rising edge at the REQ parameter), the CONT parameter must have the value "1" in order to establish or maintain a connection. The data to be sent must not be edited until the send job is completed. If the send job executes successfully, the DONE parameter is set to "1". Signal state "1" at the DONE parameter is not confirmation that the data sent has already been read by the communication partner.
- Terminating the communication connection:
  The communication connection is terminated when the CONT parameter is set to "0" even if an ongoing data transfer is not complete yet. This does not apply if you are using an already configured connection for "TSEND_C".

When setting the COM_RST parameter to "1", the currently established connection or a current data transmission can be reset at any time. This terminates the existing communication connection and a new connection is established. If data is being transferred when it executes again, this can lead to a loss of data.

To enable "TSEND_C" again after the execution (DONE = 1), call the instruction once with REQ = 0.

Note

Support when programming connections

If you select an instruction for communication TCON, TSEND_C or TRCV_C in a program block and create connections of the type TCP, UDP or ISO-on-TCP and want to assign parameters to them, you can use the support of the connection parameter assignment.

You can find the connection parameter assignment the Inspector window of the program editor.

Parameters

The following table shows the parameters of the "TSEND_C" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L	Starts the send job on a rising edge.
CONT	Input	BOOL	I, Q, M, D, L	Controls the communication connection: • 0: Disconnect the communication connection • 1: Establish and maintain the communication connection When sending data (rising edge at the REQ parameter), the CONT parameter must have the value TRUE in order to establish or maintain a connection.
LEN	Input	UINT	I, Q, M, D, L or constant	Maximum number of bytes to be sent with the job. If you use purely symbolic values at the DATA parameter, the LEN parameter must have the value "0".
CONNECT	InOut	TCON_Param	D	Pointer to the connection description See also: AUTOHOTSPOT
DATA	InOut	VARIANT	I, Q, M, D, L	Pointer to the send area containing the address and the length of the data to be sent (maximum length: 8192 bytes). When transferring structures, the structures must be identical at the sending and receiving end.
COM_RST	InOut	BOOL	I, Q, M, D, L	Restarts the instruction: • 0: Irrelevant • 1: Complete restart of the instruction causing an existing connection to be terminated and a new connection to be established.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started.

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

BUSY, DONE and ERROR parameters

You can check the status of the job with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. With the DONE parameter, you can check whether or not a job executed successfully. The ERROR parameter is set when errors occurred during execution of "TSEND_C". The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

BUSY	DONE	ERROR	Description
1	-	-	The job is being processed.
0	1	0	The job was completed successfully.
0	0	1	The job ended with an error. The cause of the error is specified in the STATUS parameter.
0	0	0	No new job was assigned.

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Description
0	0000	Job completed without error.
0	0001	The connection establishment is complete.
0	0003	The connection termination is complete.
0	7000	No job processing active.
0	7001	• Start execution of the job • Establish connection • Wait for connection partner
0	7002	Data was sent.
0	7003	Connection is being terminated.
0	7004	Connection established and monitored, no job processing active.
1	80A0	Group error for error codes 80A1 and 80A2.
1	80A1	• Connection or port already being used by user. • Communication error: – The specified connection has not yet been established. – The specified connection is being terminated. Transfer via this connection is not possible. – The interface is being re-initialized.

ERROR	STATUS* (W#16#...)	Description
1	80A2	Local or remote port is being used by the system.
1	80A3	Attempt being made to terminate a non-existent connection.
1	80A4	IP address of the remote endpoint of the connection is invalid, in other words, it matches the IP address of the local partner.
1	80A7	Communication error: You called the instruction with COM_RST = 1 before the send job was complete.
1	80AA	A connection is currently being established with the same connection ID by another block. Repeat the job with a new rising edge at the REQ parameter.
1	80B2	The CONNECT parameter points to a data block that was generated with the attribute "Only store in load memory".
1	80B3	Inconsistent parameter assignment: Group error for error codes 80A0 to 80A2, 80A4, 80B4 to 80B9.
1	80B4	When using the protocol variant ISO on TCP (connection_type = B#16#12) for the passive establishment of a connection (active_est = FALSE) one or both of the following conditions was violated: "local_tsap_id_len >= B#16#02" and/or "local_tsap_id[1] = B#16#E0".
1	80B5	Only passive connection establishment is permitted for connection type 13 = UDP.
1	80B6	Parameter assignment error in the connection_type parameter of the data block for connection description.
1	80B7	Error in one of the following parameters of the data block for connection description: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len.
1	8085	The LEN parameter is larger than the highest permitted value.
1	8086	The ID parameter within the CONNECT parameter is outside the permitted range.
1	8087	Maximum number of connections reached; no additional connection possible.
1	8088	The value at the LEN parameter does not correspond to the receive area set at the DATA parameter.
1	8089	The CONNECT parameter does not point to a data block.
1	8091	Maximum nesting depth exceeded.
1	809A	The CONNECT parameter points to a field that does not correspond to the length of the connection description.
1	809B	The ID of the local device in the connection description does not correspond to the CPU.
1	80C3	• All connection resources are in use. • A block with this ID is already being processed in a different priority group.
1	80C4	Temporary communication error: • The connection cannot be established at this time. • The connection cannot be established because the firewalls on the connection path are not open for the required ports. • The interface is receiving new parameters or the connection is being established. • The configured connection is currently being removed by a "TDISCON" instruction. • The connection used is being terminated by a call with COM_RST= 1.
1	80C6	Remote network error. Remote partner cannot be reached.
1	8722	Parameter CONNECT: The source area is invalid. The area does not exist in the DB.
1	873A	Parameter CONNECT: Access to the connection description is not possible (for example, DB does not exist).
1	877F	Parameter CONNECT: Internal error.

ERROR	STATUS* (W#16#...)	Description
1	8822	Parameter DATA: Invalid source area, the area does not exist in the DB.
1	8824	Parameter DATA: Area error in the VARIANT pointer.
1	8832	Parameter DATA: The DB number is too high.
1	883A	Parameter CONNECT: Access to specified connection data not possible (e.g. because the DB does not exist).
1	887F	Parameter DATA: Internal error, e.g. invalid VARIANT reference.
1	893A	Parameter DATA: Access to send area not possible (e.g. because the DB does not exist).
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Note

Error messages of the instructions "TCON", "TSEND", "T_DIAG" and "TDISCON"

Internally, the "TSEND_C" instruction uses the instructions "TCON (Page 4474)", "TSEND (Page 4490)", "T_DIAG (Page 4538)", "T_RESET (Page 4533)" and "TDISCON (Page 4484)". The error messages of these instructions can also be output at the STATUS parameter. The meaning of the error codes is described in the corresponding instructions. In the event of identical error codes for internally used instructions with different meanings, the instance data block of "TSEND_C" can be used to determine which instruction output the error.

See also

Difference between synchronous and asynchronous instructions (Page 2571)

TSEND_C: Establishing a connection and sending data (S7-1200, S7-1500)

Validity

The following description of the "TSEND_C" instruction is valid for:

- Ethernet
  - CPU S7-1200 with firmware version ≥ V4.0 and CPU S7-1500
  - CPU S7-1500 firmware version V2.1 and higher: UDP multicast communication
- PROFIBUS
  FDL connections of the S7-1500 with CM 1542-5 as of V2.0 with the system data type TCON_FDL

Description

The "TSEND_C" instruction sets up and establishes a communications connection. Once the connection has been set up and established, it is automatically maintained and monitored by the CPU.

The instruction is executed asynchronously and has the following functions:

- Setting up and establishing a communication connection
- Sending data via an existing communication connection
- Terminating or resetting the communication connection

Internally, the instruction "TSEND_C" uses the communication instructions TCON", "TSEND", "T_DIAG", "T_RESET" and "TDISCON".

Note

Support when programming connections

If you select an instruction for communication TCON, TSEND_C or TRCV_C in a program block and wish to create and assign parameters to connections of the type TCP, UDP, ISO-on-TCP or FDL, you can use the support of the connection parameter assignment. UDP multicast connections via the integrated PROFINET interfaces are possible for S7-1500-CPUs as of firmware version V2.1 and higher.

You can find the connection parameter assignment in the Inspector window of the program editor.

Setting up and establishing a communication connection

The communication connection is set up and established with CONT=1. For information on the number of possible communication connections, refer to the technical specifications for your CPU. The connection description specified at the CONNECT parameter is used to set up the communication connection. The following connection types can be used:

- Programmed connections (structure of connection via "TCON"):
  - TCP / UDP: Connection description via the TCON_IP_v4 system data type
  - ISO-on-TCP: Connection description via the TCON_IP_RFC system data type
  - ISO: Connection description via the TCON_ISOnative system data type (for CP1543-1 only)
  - Telecontrol connections to SMS clients: Connection description via the TCON_PHONE system data type
    For this connection type, the station requires access to the mobile network via a mobile network CP.
  - FDL connections of the S7-1500 with CM 1542-5 as of V2.0 with the system data type TCON_FDL
- Configured connections
  - Specify an existing connection in the TCON_Configured system data type.

An existing connection is terminated and the connection which has been set up is removed when the CPU goes into STOP mode. To set up and establish the connection again, you must execute "TSEND_C" again.

Sending data via an existing communication connection

The send job is executed when a rising edge is detected at the REQ parameter. As described above, the communication connection is established first.

You specify the send area with the DATA parameter. This includes the address and the length of the data to be sent. Do not use a data area with the data type BOOL or Array of BOOL at the DATA parameter. With the LEN parameter, you specify the maximum number of bytes sent with a send job. If you use a send area with optimized access at the DATA parameter, the LEN parameter must have the value "0".

The data to be sent must not be edited until the send job is completed.

Terminating and resetting the communication connection

The communication connection is terminated when the CONT parameter is set to "0" even if an ongoing data transfer is not complete yet. This does not apply if you are using a configured connection for "TSEND_C".

The connection can be reset at any time by setting the parameter COM_RST to "1". This terminates the existing communication connection and a new connection is established. If data is being transferred at this time, this can lead to data loss.

Parameters

The following table shows the parameters of the "TSEND_C" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, T, C or constant	Starts the send job on a rising edge.
CONT	Input	BOOL	I, Q, M, D, L	Controls the communication connection: • 0: Disconnect the communication connection. • 1: Establish and maintain the communication connection.
LEN	Input	UDINT	I, Q, M, D, L or constant	Optional parameter (hidden) Maximum number of bytes to be sent with the job. If you use a send area with optimized access at the DATA parameter, the value "0" must be used at the LEN parameter. For FDL connections of the CM 1542-5, the maximum length is 240 bytes. In this regard, note the maximum lengths that can be processed by the connection partner.

Parameter	Declaration	Data type	Memory area	Description
CONNECT	InOut	VARIANT	D	Pointer to the structure of the connection description: • Programmed connection: – For TCP or UDP, use the TCON_IP_v4 system data type. For a description, refer toAUTOHOTSPOT. – For ISO-on-TCP, use the TCON_IP_RFC system data type. For a description, refer to AUTOHOTSPOT. – For ISO, use the TCON_ISOnative system data type (for CP 1543-1 only). For description, refer to instruction "TCON (Page 4478)". – For connections to SMS clients, use the TCON_PHONE system data type. For a description, refer to AUTOHOTSPOT. – For FDL connections of the CM 1542-5, use the system data type TCON_FDL; see AUTOHOTSPOT. • Configured connection: – For existing connections, use the TCON_Configured system data type. For a description, see "System data type for configured connections" below.
DATA	InOut	VARIANT	I, Q, M, D, L	Pointer to the send area containing the address and the length of the data to be sent. When transferring structures, the structures must be identical at the sending and receiving end.
ADDR	InOut	VARIANT	D	Hidden parameter that needs to be used, however, with UDP. In this case it contains a pointer to the system data type TADDR_Param. Store the address information of the recipient (IP address and port number) in a data block with the system data type TADDR_Param. See also: Structure of the address information of the remote partner with UDP (Page 4523)
COM_RST	InOut	BOOL	I, Q, M, D, L	Optional parameter (hidden) Resets the connection: • 0: Irrelevant • 1: The existing connection is reset. The COM_RST parameter is reset after evaluation by the "TSEND_C" instruction and should not, therefore, be interconnected statically.

Parameter	Declaration	Data type	Memory area	Description
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Send job not yet started or still in progress. • 1: Send job executed without error. This state is only displayed for one cycle. The output parameter DONE is set if an intermediate step was completed successfully during processing (connection establishment, sending, connection termination) and if the execution of "TSEND_C" was completed successfully.
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Send job not yet started or already completed. • 1: Send job not yet completed. A new send job cannot be started.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred during connection establishment, data transfer or connection termination. The output parameter ERROR can be set due to an error in the "TSEND_C" instruction or the communication instructions used internally.
STATUS	Output	WORD	I, Q, M, D, L	Status of instruction (see the "ERROR and STATUS" parameters" description).

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

REQ, CONT and COM_RST parameters

The parameter CONT controls the connection establishment of the "TSEND_C" instruction regardless of the REQ parameter. The behavior of the CONT parameter partially depends on whether a programmed or a configured connection is used:

- With CONT = "0": No data is sent (regardless of whether a programmed or a configured connection is used).
- When changing CONT = "0" to "1":
  - With a programmed connection, it is established with "TCON".
  - With a configured connection, it is checked with "T_DIAG".
- With CONT = "1":
  - As long as no data is sent (REQ="0"), the connection is checked with "T_DIAG".
  - If the internally used communication instructions signal that no connection end point exists, the connection is automatically reestablished with "TCON".
- When changing CONT = "1" to "0":
  - With a programmed connection, it is terminated with "TDISCON".
  - With a configured connection, it is reset with "T_RESET".

The parameter COM_RST resets the connection when changing from "0" to "1":

- If a connection is established, it is reset with "T_RESET" (regardless of whether a programmed or configured connection is used).
- If no connection is established, the setting of the parameter has no effect.

The REQ and COM_RST parameters only have an effect if CONT has been set to "1". The following table shows the relationship between the REQ, CONT and COM_RST parameters:

REQ	CONT	COM_ RST	Status of the instruction	Description
Irrelevant	0	Irrelevant	Not yet executed	No job active (STATUS = 7000).
Irrelevant	0	Irrelevant	Initialization	Connection is being terminated. The instruction is being reset.
Irrelevant	0 > 1	Irrelevant	Connection establishment	Connection is being established. Data is not being transferred yet.
0	1	0	Connection established	The connection is established and is monitored with the instruction "T_DIAG".
Irrelevant	1	0 > 1	Connection established	The connection is interrupted by "T_RESET" briefly and reset.
0 > 1	1	0	Connection established	Instruction starts sending.
Irrelevant	1	0 > 1	Data is being sent	Data transfer is interrupted. The connection is being reset.

System data type for configured connections

For configured connections at the CONNECT parameter, use the following structure for connection description to TCON_Configured:

Byte	Parameter	Data type	Start value	Description
0 … 1	InterfaceID	HW_ANY	-	Hardware identifier of the local interface (value range: 0 to 65535).
2 … 3	ID	CONN_OUC	-	Reference to the connection (value range: 1 to 4095). Enter the connection ID of the existing connection.
4	ConnectionType	BYTE	-	Connection type Select 254 (decimal) for a configured connection.

Note

For reasons of compatibility, the parameters InterfaceID and ConnectionType are part of the structure for the connection description to TCON_Configured. These parameters do not have an effect on the connection parameter assignment; only the parameter ID for the connection ID is evaluated in the connection parameter assignment.

BUSY, DONE and ERROR parameters

You can check the status of the job with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. With the DONE parameter, you can check whether or not a send job executed successfully. The ERROR parameter is set when errors occurred during execution of "TSEND_C". The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

DONE	BUSY	ERROR	Description
0	0	0	The instruction has not been executed yet (no rising edge at REQ parameter).
0	1	0	The instruction is being executed and calls the internally used communication instructions.
1	0	0	The send job was completed successfully. "0000" is output at the STATUS parameter. DONE = "1" is only displayed for one cycle.
0	0	1	The execution of the instruction or an intermediate step during processing was terminated with an error. If there is a subsequent error due to an internally used communication instruction, the error that occurred first during processing is displayed. This state is only displayed for one cycle.

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Description
0	0000	Send job was executed without error.
0	0001	Communication connection established.
0	0003	Communication connection closed.
0	7000	No active send job execution; no communication connection established.
0	7001	Initial call for establishing a connection.
0	7002	Second call for establishing a connection
0	7003	Communication connection is being terminated.
0	7004	Communication connection has been established and is being monitored. No send job execution active.
0	7005	Data transfer in progress.
1	80A1	• Connection or port already being used by user. • Communication error: – The specified connection has not yet been established. – The specified connection is being terminated. Transfer via this connection is not possible. – The interface is being re-initialized.
1	80A3	The nested "T_DIAG" instruction has reported that the connection has closed.
1	80A4	IP address of the remote endpoint of the connection is invalid or it matches the IP address of the local partner.
1	80A7	Communication error: You called the instruction with COM_RST = 1 before the send job was complete.
1	80AA	A connection is currently being established with the same connection ID by another block. Repeat the job with a new rising edge at the REQ parameter.

ERROR	STATUS* (W#16#...)	Description
1	80B3	• When using the protocol variant UDP the ADDR parameter does not contain any data. • Error in the connection description • The local port is already being used in a different connection description.
1	80B4	You have violated one or both of the following conditions for passive connection establishment (active_est = FALSE) when using the ISO-on-TCP protocol variant (connection_type = B#16#12): • local_tsap_id_len >= B#16#02 • local_tsap_id[1] = B#16#E0
1	80B5	Only passive connection establishment is permitted for connection type 13 = UDP.
1	80B6	Parameter assignment error in the connection_type parameter of the data block for connection description.
1	80B7	• For system data type TCON_Param: Error in one of the following parameters of the data block for connection description: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len. • For system data types TCON_IP_V4 and TCON_IP_RFC: IP address of the partner end point was set to 0.0.0.0.
1	8085	The LEN parameter is larger than the highest permitted value.
1	8086	The ID parameter within the CONNECT parameter is outside the permitted range.
1	8087	Maximum number of connections reached; no additional connection possible.
1	8088	The value at the LEN parameter does not correspond to the receive area set at the DATA parameter.
1	8089	• The CONNECT parameter does not point to a data block. • The CONNECT parameter does not point to a connection description. • The manually created connection description has an incorrect structure for the selected connection type.
1	8091	Maximum nesting depth exceeded.
1	809A	The CONNECT parameter points to a field that does not correspond to the length of the connection description.
1	809B	InterfaceID is invalid. It is either zero or it does not point to a local CPU interface or a CP.
1	80C3	• All connection resources are in use. • A block with this ID is already being processed in a different priority group.
1	80C4	Temporary communication error: • The connection cannot be established at this time. • The connection cannot be established because the firewalls on the connection path are not open for the required ports. • The interface is receiving new parameters or the connection is being established. • The configured connection is currently being removed by a "TDISCON (Page 4484)" instruction. • The connection used is being terminated by a call with COM_RST = 1. • Temporarily no receive resources available at the connection partner. The connection partner is not ready to receive.
1	80C5	• Connection terminated by the communication partner. • LSAP of the remote connection partner is not released

ERROR	STATUS* (W#16#...)	Description
1	80C6	Network error: • Remote partner cannot be reached. • Physical interruption on PROFIBUS
1	8722	Parameter CONNECT: The source range is invalid. The area does not exist in the DB.
1	873A	Parameter CONNECT: Access to the connection description is not possible (for example, because the DB is not available).
1	877F	Parameter CONNECT: Internal error.
1	8822	Parameter DATA: Invalid source range, the range does not exist in the DB.
1	8824	Parameter DATA: Area error in the VARIANT pointer.
1	8832	Parameter DATA: The DB number is too high.
1	883A	Parameter CONNECT: Access to specified connection data not possible (e.g. because the DB does not exist).
1	887F	Parameter DATA: Internal error, e.g. invalid VARIANT reference.
1	893A	Parameter DATA: Access to send area not possible (e.g. because the DB does not exist).
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Note

Error messages of the instructions "TCON", "TSEND", "T_DIAG", "T_RESET" and "TDISCON"

Internally, the "TSEND_C" instruction uses the instructions "TCON (Page 4478)", "TSEND (Page 4490)", "T_DIAG (Page 4538)", "T_RESET (Page 4533)" and "TDISCON (Page 4484)". The error messages of these instructions can also be output at the STATUS parameter. The meaning of the error codes is described in the corresponding instructions. In the event of identical error codes for internally used instructions with different meanings, the instance data block of "TSEND_C" can be used to determine which instruction output the error.

Example

You can find the example here: Program example for send functions (Page 4446).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

TRCV_C: Establishing a connection and receiving data (S7-1200, S7-1500)

TRCV_C: Receive data via Ethernet (S7-1200)

Validity

The following description of the "TRCV_C" instruction applies to S7-1200 CPUs with firmware version < V4.0.

Description

The "TRCV_C" instruction is executed asynchronously and has the following functions:

1. **Setting up and establishing a communication connection:**
   "TRCV_C" sets up and establishes a TCP or ISO-on-TCP communication connection. Once the connection has been set up and established, it is automatically maintained and monitored by the CPU.
   The connection description specified at the CONNECT parameter is used to set up the communication connection. To establish a connection, the CONT parameter must be set to the value "1". Once the connection is successfully established, the DONE parameter is set to "1".
   An existing connection is terminated and the connection which has been set up is removed when the CPU goes into STOP mode. To set up and establish the connection again, you must execute "TRCV_C" again.
   For information on the number of possible communication connections, refer to the technical specifications for your CPU.
2. **Receiving data via an existing communication connection:**
   If the EN_R parameter is set to the value "1", receipt of data is enabled. When receiving data (rising edge at the EN_R parameter), the CONT parameter must have the value TRUE in order to establish or maintain a connection.
   The received data is entered in a receive area. You specify the length of the receive area either with the LEN parameter (if LEN <> 0) or with the length information of the DATA parameter (if LEN = 0) in accordance with the protocol variant used. If you use purely symbolic values at the DATA parameter, the LEN parameter must have the value "0".
   After data has been received successfully, the signal state at the DONE parameter is "1". If errors occur in the data transfer, the DONE parameter is set to "0".
3. **Terminating the communication connection:**
   The communication connection is terminated immediately when the CONT parameter is set to "0".

TRCV_C is executed again when the COM_RST parameter is set. This terminates the existing communication connection and a new connection is established. If data is being received when it executes again, this can lead to a loss of data.

Note

Support when programming connections

If you select an instruction for communication TCON, TSEND_C or TRCV_C in a program block and create connections of the type TCP, UDP or ISO-on-TCP and want to assign parameters to them, you can use the support of the connection parameter assignment.

You can find the connection parameter assignment the Inspector window of the program editor.

Receive modes of TRCV_C

The following table shows how the received data is entered in the receive area.

Protocol variant	Availability of data in the receive area	connection_type parameter of the connection description	LEN parameter
TCP (Ad-hoc mode)	The data is immediately available.	Hexadecimal value: B#16#11 Integer value: 17	0
TCP (Receipt of data with specified length)	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#11 Integer value: 17	1 to 8192
ISO on TCP (Message-oriented data transfer)	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#12 Integer value: 18	• 1 to 1452, if a CP is used. • 1 to 8192, if no CP is used.

TCP (Ad-hoc mode)

The ad-hoc mode is only available with the TCP protocol variant. You use the ad-hoc mode to receive data with dynamic length with the "TRCV" instruction.

You set ad-hoc mode by assigning the value "0" to the LEN parameter. All data types can be used for data blocks with standard access when you use ad-hoc mode. Only ARRAY of BYTE or data types with a length of 8 bits can be used for data blocks with optimized access (e.g., CHAR, USINT, SINT, etc.). The data length actually received is output at the RCVD_LEN parameter.

TCP (Receipt of data with specified length)

You use the value of the LEN parameter to specify the length for the data receipt. The data receipt is not complete until the length of data specified at the LEN parameter has been completely received. Only then is the data available in the receive area (DATA parameter). The actually received data length in bytes at the RCVD_LEN parameter corresponds to the data length at the LEN parameter after receipt.

ISO on TCP (Message-oriented data transfer)

Complete message blocks are sent via a connection with the protocol variant ISO on TCP; these are recognized as such by the recipient. When using ISO on TCP, "TRCV_C" signals data receipt as soon as the message block has been completely received. The receive area is defined by the LEN and DATA parameters. If the receive buffer (DATA parameter) is too small for the sent data, "TRCV_C" signals an error. The actually received data length in bytes at the RCVD_LEN parameter corresponds to the data length at the LEN parameter after receipt.

Parameters

The following table shows the parameters of the "TRCV_C" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN_R	Input	BOOL	I, Q, M, D, L	Receive enable
CONT	Input	BOOL	I, Q, M, D, L	Controls the communication connection: • 0: Disconnect the communication connection • 1: Establish and maintain the communication connection When receiving data (rising edge at the EN_R parameter), the CONT parameter must have the value TRUE in order to establish or maintain a connection.
LEN	Input	UINT	I, Q, M, D, L or constant	Maximum length of the data to be received (maximum value: 8192 bytes). If you use purely symbolic values at the DATA parameter, the LEN parameter must have the value "0".
CONNECT	InOut	TCON_Param	D	Pointer to the connection description See also: AUTOHOTSPOT
DATA	InOut	VARIANT	I, Q, M, D, L	Pointer to the receive area When transferring structures, the structures must be identical at the sending and receiving end.
COM_RST	InOut	BOOL	I, Q, M, D, L	Restarts the instruction: • 0: Irrelevant • 1: Complete restart of the instruction causing an existing connection to be terminated
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter ERROR: • 0: No error • 1: Error occurred

Parameter	Declaration	Data type	Memory area	Description
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction
RCVD_LEN	Output	UINT	I, Q, M, D, L	Amount of data actually received in bytes

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

BUSY, DONE and ERROR parameters

You can check the status of the job with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. With the DONE parameter, you can check whether or not a job executed successfully. The ERROR parameter is set when errors occurred during execution of "TRCV_C". The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

BUSY	DONE	ERROR	Description
1	-	-	The job is being processed.
0	1	0	The job was completed successfully.
0	0	1	The job ended with an error. The cause of the error is output at the STATUS parameter.
0	0	0	No new job was assigned.

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Description
0	0000	Job completed without error.
0	0001	The connection establishment is complete.
0	0003	The connection termination is complete.
0	7000	No job processing active.
0	7001	• Start execution of the job • Establish connection • Wait for connection partner
0	7002	Data is being received.
0	7003	Connection is being terminated.
0	7004	• Connection established and monitored • No job processing active
0	7006	Data is currently being received.
1	8085	• The LEN parameter is larger than the highest permitted value. • The value at the LEN or DATA parameter was changed after the first call.
1	8086	The ID parameter is outside the permitted range.
1	8087	Maximum number of connections reached; no additional connection possible
1	8088	The value at the LEN parameter does not correspond to the receive area set at the DATA parameter.

ERROR	STATUS* (W#16#...)	Description
1	8089	The CONNECT parameter does not point to a data block.
1	8091	Maximum nesting depth exceeded.
1	809A	The CONNECT parameter points to a field that does not correspond to the length of the connection description.
1	809B	The ID of the local device (local_device_id) in the connection description does not correspond to the CPU.
1	80A0	Group error for error codes W#16#80A1 and W#16#80A2.
1	80A1	• Connection or port already being used by user. • Communication error: – The specified connection has not yet been established. – The specified connection is being terminated. Transfer via this connection is not possible. – The interface is being re-initialized.
1	80A2	Local or remote port is being used by the system.
1	80A3	• Attempt being made to re-establish an existing connection. • Attempt being made to terminate a non-existent connection.
1	80A4	IP address of the remote endpoint of the connection is invalid, in other words, it matches the IP address of the local partner.
1	80A7	Communication error: You called the instruction with COM_RST = 1 before the send job was complete.
1	80B2	The CONNECT parameter points to a data block that was generated with the attribute "Only store in load memory".
1	80B3	Inconsistent parameter assignment: Group error for error codes W#16#80A0 to W#16#80A2, W#16#80A4, W#16#80B4 to W#16#80B9.
1	80B4	When using the protocol variant ISO on TCP (connection_type = B#16#12) for the passive establishment of a connection (active_est = FALSE) one or both of the following conditions was violated: "local_tsap_id_len >= B#16#02" and/or "local_tsap_id[1] = B#16#E0".
1	80B5	Only passive connection establishment is permitted for connection type 13 = UDP.
1	80B6	Parameter assignment error in the connection_type parameter of the data block for connection description.
1	80B7	Error in one of the following parameters of the data block for connection description: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len.
1	80C3	• All connection resources are in use. • A block with this ID is already being processed in a different priority group.
1	80C4	Temporary communication error: • The connection cannot be established at this time. • The connection cannot be established because the firewalls on the connection path are not open for the required ports. • The interface is receiving new parameters or the connection is being established. • The configured connection is currently being removed by a "TDISCON" instruction. • The connection used is being terminated by a call with COM_RST= 1.
1	80C6	The remote partner cannot be reached (network error).
1	8722	Error in the CONNECT parameter: Invalid source area (area not declared in data block).

ERROR	STATUS* (W#16#...)	Description
1	873A	Error in the CONNECT parameter: Access to connection description is not possible (no access to data block).
1	877F	Error in the CONNECT parameter: Internal error
1	8922	Parameter DATA: Invalid destination area; the area does not exist in the DB.
1	8924	Parameter DATA: Area error in the VARIANT pointer.
1	8932	Parameter DATA: The DB number is too high.
1	893A	Parameter CONNECT: Access to specified connection data not possible (e.g. because the DB does not exist).
1	897F	Parameter DATA: Internal error, e.g. invalid VARIANT reference.
1	8A3A	Parameter DATA: No access to the data area, for example because the data block does not exist.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Note

Error messages of the instructions "TCON", "TRCV" and "TDISCON"

Internally, the TRV_C instruction uses the "TCON (Page 4474)", "TRCV (Page 4493)" and "TDISCON (Page 4484)" instructions. The error messages of these instructions are contained in the respective descriptions.

See also

Difference between synchronous and asynchronous instructions (Page 2571)

TRCV_C: Establishing a connection and receiving data (S7-1200, S7-1500)

Validity

The following description of the "TRCV_C" instruction is valid for:

- Ethernet:
  - CPU S7-1200 with firmware version V4.0 or higher and CPU S7-1500
  - CPU S7-1500 firmware version V2.1 and higher: UDP multicast communication
- PROFIBUS: FDL connections of the S7-1500 with CM 1542-5 as of V2.0 with the system data type TCON_FDL

Description

The "TRCV_C" instruction is executed asynchronously and implements the following functions in sequence:

- Setting up and establishing a communication connection
- Receiving data via an existing communication connection
- Terminating or resetting the communication connection

Internally, the instruction "TRCV_C" uses the communication instructions TCON", "TRCV", "T_DIAG", "T_RESET" and "TDISCON".

Note

Support when programming connections

If you select an instruction for communication TCON, TSEND_C or TRCV_C in a program block and wish to create and assign parameters to connections of the type TCP, UDP, ISO-on-TCP or FDL, you can use the support of the connection parameter assignment. UDP multicast connections via the integrated PROFINET interfaces are possible for S7-1500-CPUs as of firmware version V2.1 and higher.

You can find the connection parameter assignment in the Inspector window of the program editor.

Setting up and establishing a communication connection

The communication connection is set up and established with CONT=1. For information on the number of possible communication connections, refer to the technical specifications for your CPU. The connection description specified at the CONNECT parameter is used to set up the communication connection. The following connection types can be used:

- Programmed connections (structure of connection via "TCON"):
  - TCP / UDP: Connection description via the TCON_IP_v4 system data type
  - ISO-on-TCP: Connection description via the TCON_IP_RFC system data type
  - ISO: Connection description via the TCON_ISOnative system data type (for CP1543-1 only)
  - Telecontrol connections to SMS clients: Connection description via the TCON_PHONE system data type
    For this connection type, the station requires access to the mobile network via a mobile network CP.
  - FDL connections of the S7-1500 with CM 1542-5 as of V2.0 with the system data type TCON_FDL
- Configured connections
  - Specify an existing connection in the TCON_Configured system data type.

An existing connection is terminated and the connection which has been set up is removed when the CPU goes into STOP mode. To set up and establish the connection again, you must execute "TRCV_C" again.

Receiving data via an existing communication connection

Receipt of data is enabled when the EN_R parameter is set to the value "1". The received data is entered in a receive area. You specify the length of the receive area either with the LEN parameter (if LEN <> 0) or with the length information of the DATA parameter (if LEN = 0), depending on the protocol variant being used. If you use purely symbolic values at the DATA parameter, the LEN parameter must have the value "0".

Receive modes of TRCV_C:

- **TCP (Ad-hoc mode)**
  The ad-hoc mode is only available with the TCP protocol variant. You use the ad-hoc mode to receive data with dynamic length with the "TRCV_C" instruction.
  You set ad-hoc mode by assigning the value "1" to the ADHOC parameter. All data types can be used for data blocks with standard access when you use ad-hoc mode. Only ARRAY of BYTE or data types with a length of 8 bits can be used for data blocks with optimized access (e.g., CHAR, USINT, SINT, etc.). The data length actually received is output at the RCVD_LEN parameter.
- **TCP (Receipt of data with specified length)**
  Assign the value "0" to the ADHOC parameter for receipt of data with specified length. If ad-hoc mode is disabled, the data reception is not complete until the length of data specified at the LEN parameter has been completely received. Only then is the data available in the receive area (DATA parameter). The actually received data length in bytes at the RCVD_LEN parameter corresponds to the data length at the LEN parameter after receipt.
- **ISO-on-TCP (Message-oriented data transfer)**
  Complete message blocks are sent via a connection with the protocol variant ISO-on-TCP; these are recognized as such by the recipient. The receive area is defined by the LEN and DATA parameters. If the receive buffer (DATA parameter) is too small for the sent data, "TRCV_C" signals an error. The actually received data length in bytes at the RCVD_LEN parameter corresponds to the data length at the LEN parameter after receipt.

The following table shows how the received data is entered in the receive area.

Protocol variant	Availability of data in the receive area	connection_type parameter of the connection description	Parameter LEN
TCP (Ad-hoc mode)	The data is immediately available.	Hexadecimal value: B#16#11 Integer value: 17	1 up to maximum length (depending on the CPU)
TCP (Receipt of data with specified length)	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#11 Integer value: 17	1 to 8192
ISO-on-TCP (Message-oriented data transfer)	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#12 Integer value: 18	1 to 8192
FDL	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#15 Decimal: 21	1 to 240

Terminating the communication connection

The communication connection is terminated when the CONT parameter is set to "0" even if an ongoing data transfer is not complete yet. This does not apply if you are using a configured connection, however.

The connection can be reset at any time by setting the parameter COM_RST to "1". This terminates the existing communication connection and a new connection is established. If data is being transferred at this time, this can lead to data loss.

Parameters

The following table shows the parameters of the "TRCV_C" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN_R	Input	BOOL	I, Q, M, D, L, T, C or constant	Receive enable
CONT	Input	BOOL	I, Q, M, D, L	Controls the communication connection: • 0: Disconnect the communication connection. • 1: Establish communication connection and maintain after receipt of data.
LEN	Input	UDINT	I, Q, M, D, L or constant	Maximum length of the data to be received. If you use a receive area with optimized access at the DATA parameter, the value "0" must be used at the LEN parameter.
ADHOC	Input	BOOL	I, Q, M, D, L or constant	Optional parameter (hidden) Use ad-hoc mode for the TCP protocol variant.

Parameter	Declaration	Data type	Memory area	Description
CONNECT	InOut	VARIANT	D	Pointer to the connection description • Programmed connection: – For TCP or UDP, use the structure TCON_IP_v4 For description, refer to:AUTOHOTSPOT – For ISO-on-TCP, use the structure TCON_IP_RFC For a description, refer to: AUTOHOTSPOT – For ISO, use the structure TCON_ISOnative (for CP 1543-1 only) For description, refer to instruction "TCON (Page 4478)": "Structure of the connection description according to TCON_ISOnative" – For connections to SMS clients, use the TCON_PHONE system data type. For a description, refer to AUTOHOTSPOT. – For FDL connections of the CM 1542-5, use the system data type TCON_FDL; see AUTOHOTSPOT. • Configured connection: – For existing connections, use the TCON_Configured system data type. For a description, see "System data type for configured connections" below.
DATA	InOut	VARIANT	I, Q, M, D, L	Pointer to the receive area. When transferring structures, the structures must be identical at the sending and receiving end.
ADDR	InOut	VARIANT	D	Hidden parameter that needs to be used, however, with UDP. In this case it contains a pointer to the system data type TADDR_Param. Store the address information of the sender (IP address and port number) in a data block with the system data type TADDR_Param. See also: Structure of the address information of the remote partner with UDP (Page 4523)
COM_RST	InOut	BOOL	I, Q, M, D, L	Optional parameter (hidden) Resets the connection: • 0: Irrelevant • 1: The existing connection is reset. The COM_RST parameter is reset after evaluation by the "TRCV_C" instruction and should not, therefore, be interconnected statically.

Parameter	Declaration	Data type	Memory area	Description
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Receipt not yet started or still in progress. • 1: Receipt executed without error. This state is only displayed for one cycle. The output parameter DONE is set if an intermediate step was completed successfully during processing (connection establishment, receipt, connection termination) and if the execution of "TRCV_C" was completed successfully.
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Receipt not yet started or already completed. • 1: Receipt not yet completed. A new send job cannot be started.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred during connection establishment, data reception or connection termination. The output parameter ERROR can be set due to an error in the "TRCV_C" instruction or the communication instructions used internally.
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction
RCVD_LEN	Output	UDINT	I, Q, M, D, L	Amount of data actually received in bytes

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

EN_R, CONT and COM_RST parameters

The parameter CONT controls the connection establishment of the "TRCV_C" instruction regardless of the EN_R parameter. The behavior of the CONT parameter partially depends on whether a programmed or a configured connection is used:

- With CONT = "0": No data is received (regardless of whether a programmed or a configured connection is used).
- When changing CONT = "0" to "1":
  - With a programmed connection, it is established with "TCON".
  - With a configured connection, it is checked with "T_DIAG".
- With CONT = "1":
  - As long as no data is received (EN_R="0"), the connection is checked with "T_DIAG".
  - If the internally used communication instructions signal that no connection end point exists, the connection is automatically reestablished with "TCON".
- When changing CONT = "1" to "0":
  - With a programmed connection, it is terminated with "TDISCON".
  - With a configured connection, it is reset with "T_RESET".

The parameter COM_RST resets the connection when changing from "0" to "1":

- If a connection is established, it is reset with "T_RESET" (regardless of whether a programmed or configured connection is used).
- If no connection is established, the setting of the parameter has no effect.

The EN_R and COM_RST parameters only have an effect if CONT has been set to "1". The following table shows the relationship between the EN_R, CONT and COM_RST parameters:

EN_R	CONT	COM_RST	Status of the instruction	Description
Irrelevant	0	Irrelevant	Not yet executed	No job active (STATUS = 7000).
Irrelevant	0	Irrelevant	Initialization	Connection is being terminated. The instruction is being reset.
Irrelevant	0 > 1	Irrelevant	Connection establishment	Connection is being established. Data is not being transferred yet.
0	1	0	Connection established	The connection is established and is monitored with the instruction "T_DIAG".
Irrelevant	1	0 > 1	Connection established	The connection is interrupted by "T_RESET" briefly and reset.
0 > 1	1	0	Connection established	Instruction starts receiving.
Irrelevant	1	0 > 1	Data is being received	Data transfer is interrupted. The connection is being reset.

System data type for configured connections

For configured connections at the CONNECT parameter, use the following structure for connection description to TCON_Configured:

Byte	Parameter	Data type	Start value	Description
0 … 1	InterfaceID	HW_ANY	-	Hardware identifier of the local interface (value range: 0 to 65535).
2 … 3	ID	CONN_OUC	-	Reference to the connection (value range: 1 to 4095). Enter the connection ID of the existing connection.
4	ConnectionType	BYTE	-	Connection type Select 254 (decimal) for a configured connection.

BUSY, DONE and ERROR parameters

You can check the status of the job with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. With the DONE parameter, you can check whether or not a job executed successfully. The ERROR parameter is set when errors occurred during execution of "TRCV_C". The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

DONE	BUSY	ERROR	Description
0	0	0	The instruction has not been executed yet (no rising edge at EN_R parameter).
0	1	0	The instruction is being executed and calls the internally used communication instructions.
1	0	0	The receipt has been completed successfully. "0000" is output at the parameter STATUS. DONE = "1" is only displayed for one cycle.
0	0	1	The execution of the instruction or an intermediate step during processing was terminated with an error. If there is a subsequent error due to an internally used communication instruction, the error that occurred first during processing is displayed. This state is only displayed for one cycle.

ERROR and STATUS parameters

ERROR	STATUS (W#16#...)	Description
0	0000	Receive job executed without error.
0	0001	Communication connection established.
0	0003	Communication connection closed.
0	7000	No job processing active.
0	7001	Initial call for establishing a connection.
0	7002	Second call for establishing a connection.
0	7003	Communication connection is being terminated.
0	7004	Communication connection has been established and is being monitored. No receive job processing active.
0	7006	Data is currently being received.
1	8085	• The LEN parameter is larger than the highest permitted value. • The value at the LEN or DATA parameter was changed after the first call.
1	8086	The ID parameter is outside the permitted range.
1	8087	Maximum number of connections reached; no additional connection possible
1	8088	The value at the LEN parameter does not correspond to the receive area set at the DATA parameter.
1	8089	• The CONNECT parameter does not point to a data block. • The CONNECT parameter does not point to a connection description. • The manually created connection description has an incorrect structure for the selected connection type.
1	8091	Maximum nesting depth exceeded.
1	809A	The CONNECT parameter points to a field that does not correspond to the length of the connection description.
1	809B	InterfaceID is invalid. It is either zero or it does not point to a local CPU interface or a CP.
1	80A1	• Connection or port already being used by user. • Communication error: – The specified connection has not yet been established. – The specified connection is being terminated. Transfer via this connection is not possible. – The interface is being re-initialized.

ERROR	STATUS (W#16#...)	Description
1	80A3	The nested "T_DIAG" instruction has reported that the connection has closed.
1	80A4	IP address of the remote endpoint of the connection is invalid or it matches the IP address of the local partner.
1	80A7	Communication error: You called the instruction with COM_RST = 1 before the send job was complete.
1	80AA	A connection is currently being established with the same connection ID by another block. Repeat the job with a new rising edge at the REQ parameter.
1	80B3	• When using the protocol variant UDP the ADDR parameter does not contain any data. • Error in the connection description • The local port is already being used in a different connection description.
1	80B4	When using the protocol variant ISO on TCP (connection_type = B#16#12) for the passive establishment of a connection (active_est = FALSE) one or both of the following conditions was violated: "local_tsap_id_len >= B#16#02" and/or "local_tsap_id[1] = B#16#E0".
1	80B5	Only passive connection establishment is permitted for connection type 13 = UDP.
1	80B6	Parameter assignment error in the connection_type parameter of the data block for connection description.
1	80B7	• For system data type TCON_Param: Error in one of the following parameters of the data block for connection description: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len. • For system data types TCON_IP_V4 and TCON_IP_RFC: IP address of the partner end point was set to 0.0.0.0.
1	80C3	• All connection resources are in use. • A block with this ID is already being processed in a different priority group.
1	80C4	Temporary communication error: • The connection cannot be established at this time. • The connection cannot be established because the firewalls on the connection path are not open for the required ports. • The interface is receiving new parameters or the connection is being established. • The configured connection is currently being removed by a "TDISCON" instruction. • The connection used is being terminated by a call with COM_RST = 1.
1	80C6	The remote partner cannot be reached (network error).
1	8722	Error in the CONNECT parameter: Invalid source range (range not declared in data block).
1	873A	Error in the CONNECT parameter: Access to connection description is not possible (no access to data block).
1	877F	Error in the CONNECT parameter: Internal error
1	8922	Parameter DATA: Invalid target range; the area does not exist in the DB.
1	8924	Parameter DATA: Area error in the VARIANT pointer.
1	8932	Parameter DATA: The DB number is too high.
1	893A	Parameter CONNECT: Access to specified connection data not possible (e.g. because the DB does not exist).
1	897F	Parameter DATA: Internal error, e.g. invalid VARIANT reference.
1	8A3A	Parameter DATA: No access to the data area, for example because the data block does not exist.

Note

Error messages of the instructions "TCON", "TRCV" and "TDISCON"

Internally, the TRV_C instruction uses the "TCON (Page 4478)", "TRCV (Page 4493)" and "TDISCON (Page 4484)" instructions. The error messages of these instructions are contained in the respective descriptions.

Example

You can find the example here: Program example for send functions (Page 4446).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

TSEND_C: Establishing a connection and sending data (Page 4423)

Difference between synchronous and asynchronous instructions (Page 2571)

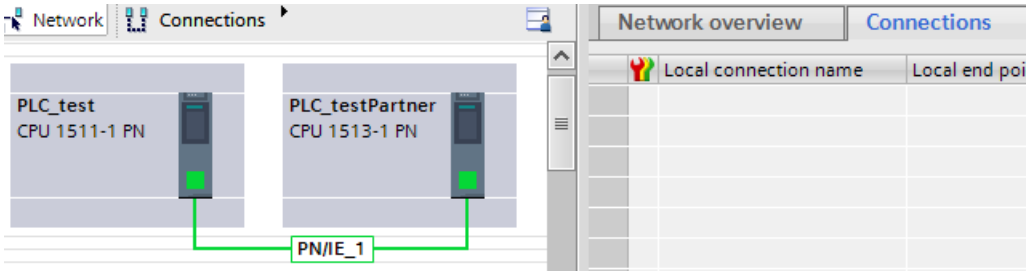
Program example for send functions (S7-1200, S7-1500)

Introduction

In the following example, you create a programmed connection between two CPUs of the S7-1500 series and send a character string from CPU 1 to CPU 2. The character string to be sent is of the data type STRING.

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. Their connection is not yet configured.



- A low protection level under Properties > Protection ensures that read and write access is permitted for each CPU.

Program of CPU 1

Create nine tags in a global data block for storing the data of "TSEND_C".

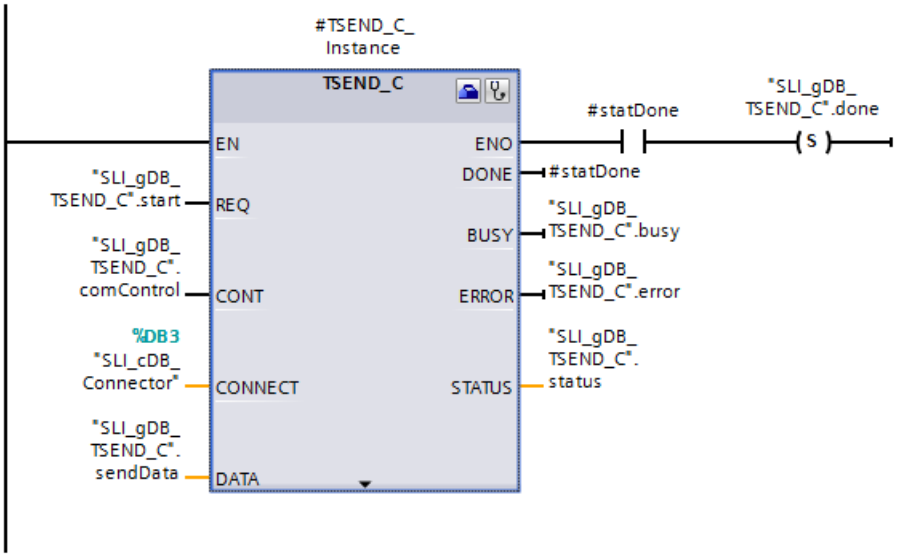
SLI_gDB_TSEND_C

	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	comControl	Bool	false
4	done	Bool	false
5	busy	Bool	false
6	error	Bool	false
7	status	Word	16#0
8	memErrStatus	Word	16#0
9	sendData	String	'ProcessXYZ'

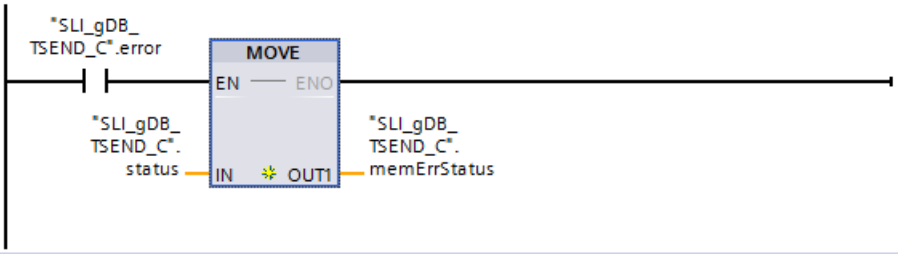
Create an FB "SLI_FB_TSEND_C" and create the following local tags in the FB.

7	Static			
8	TSEND_C_Instance	TSEND_C		
9	done	Bool	false	Non-retain

Network 1: Interconnect the parameters of the "TSEND_C" instruction as follows. Do not interconnect the parameter CONNECT yet.



Network 2: Save the status for an error of TSEND_C as follows.

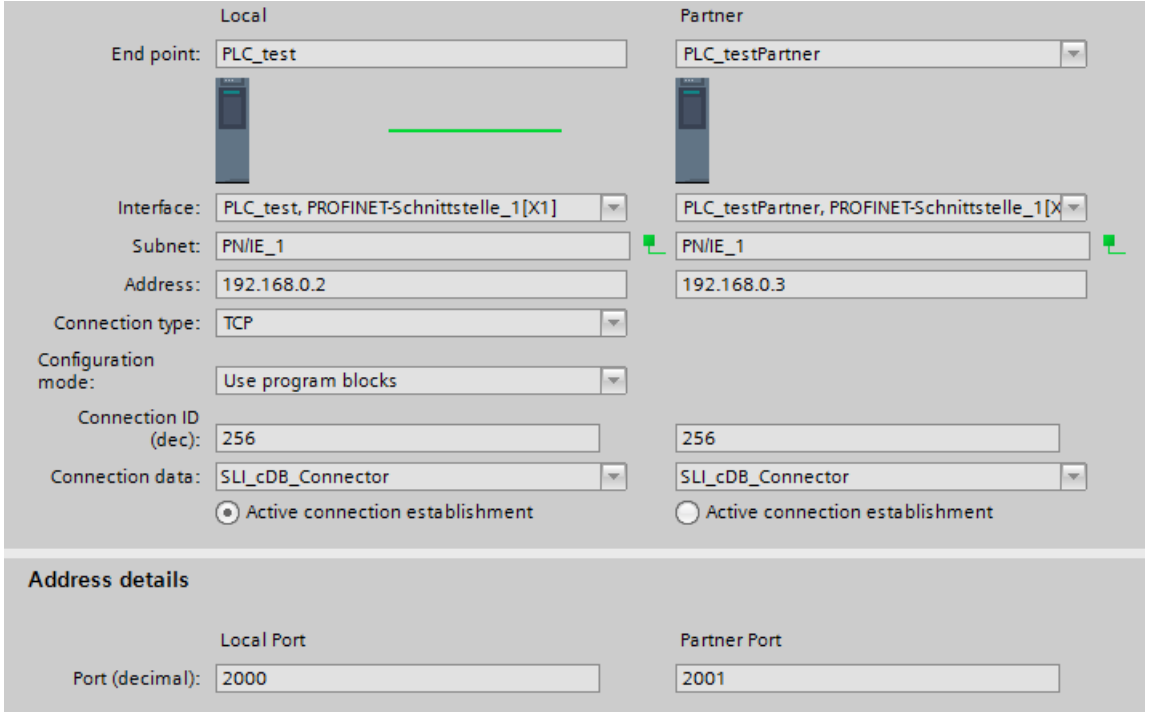


Configuration of TSEND_C

To interconnect the parameter CONNECT, open the wizard of the "TSEND_C" instruction with Properties > Configuration.

Make the following settings for the configuration of TSEND_C:

Input field	Entry
End point	Select a CPU for sender and receiver using the drop-down list. Interface, subnet and address are entered automatically.
Connection data	Use the drop-down list and the selection "New" to create one data block per CPU. This data block is required for storing the connection data. The name of the data block can be freely selected. The active connection establishment is enabled for CPU 1 (local CPU).
Connection type	Select the entry "TCP". This means an Ethernet connection with the "TCP" protocol is used for connection establishment.
Configuration mode	Select the entry "Use program block". This means connection establishment takes place by means of a programmed connection.
Connection ID	You enter a freely selected connection ID for the communication connection. The connection ID must not already be assigned in the project.
Partner port	You enter a value for the partner port. The value must be >=2000.



Program of CPU 2

Create ten tags in a global data block for storing the data of "TRCV_C".

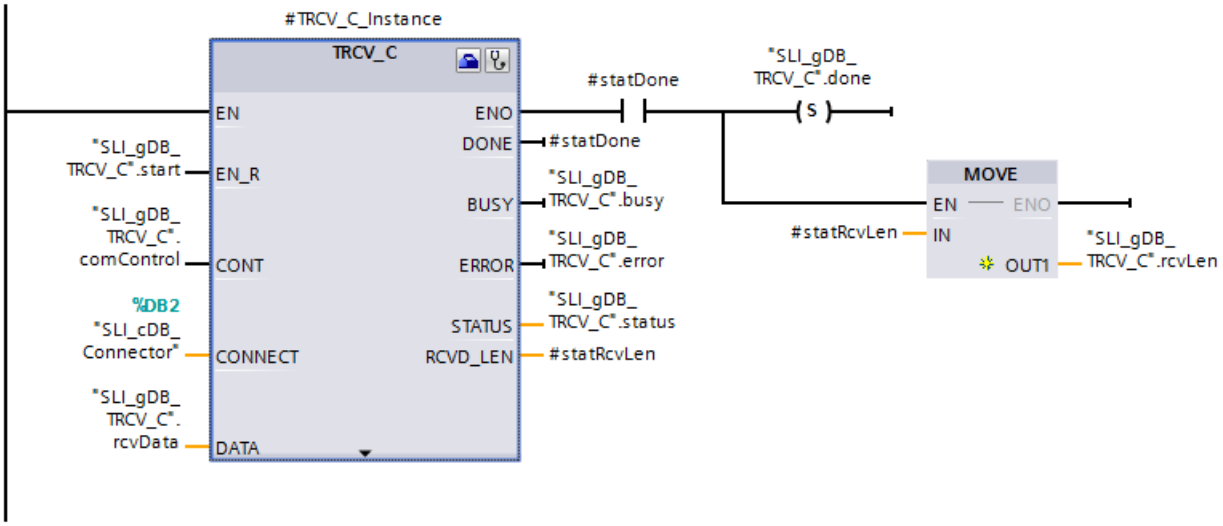
SLI_gDB_TRCV_C

	Name	Data type	Start value
1	Static		
2	start	Bool	false
3	comControl	Bool	false
4	done	Bool	false
5	busy	Bool	false
6	error	Bool	false
7	status	Word	16#0
8	memErrStatus	Word	16#0
9	rcvLen	UDInt	0
10	rcvData	String	''

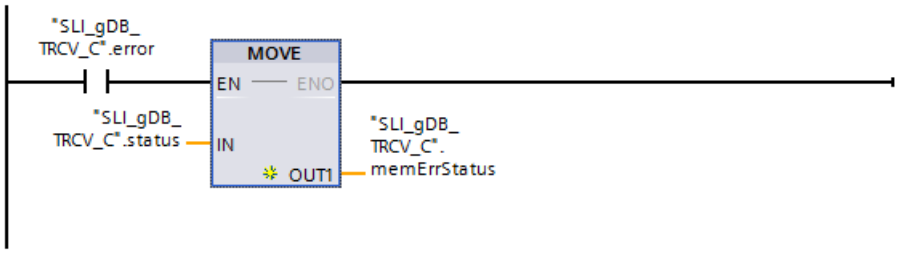
Create an FB "SLI_FB_TRCV_C" and create the following local tags in the FB.

7	Static			
8	TRCV_C_Instance	TRCV_C		
9	statDone	Bool	false	Non-retain
10	statRcvLen	UDInt	0	Non-retain

Network 1: Interconnect the parameters of the "TRCV_C" instruction as follows. Do not interconnect the parameter CONNECT yet.



Network 2: Save the status for an error of TRCV_C as follows.



Configuration of TRCV_C

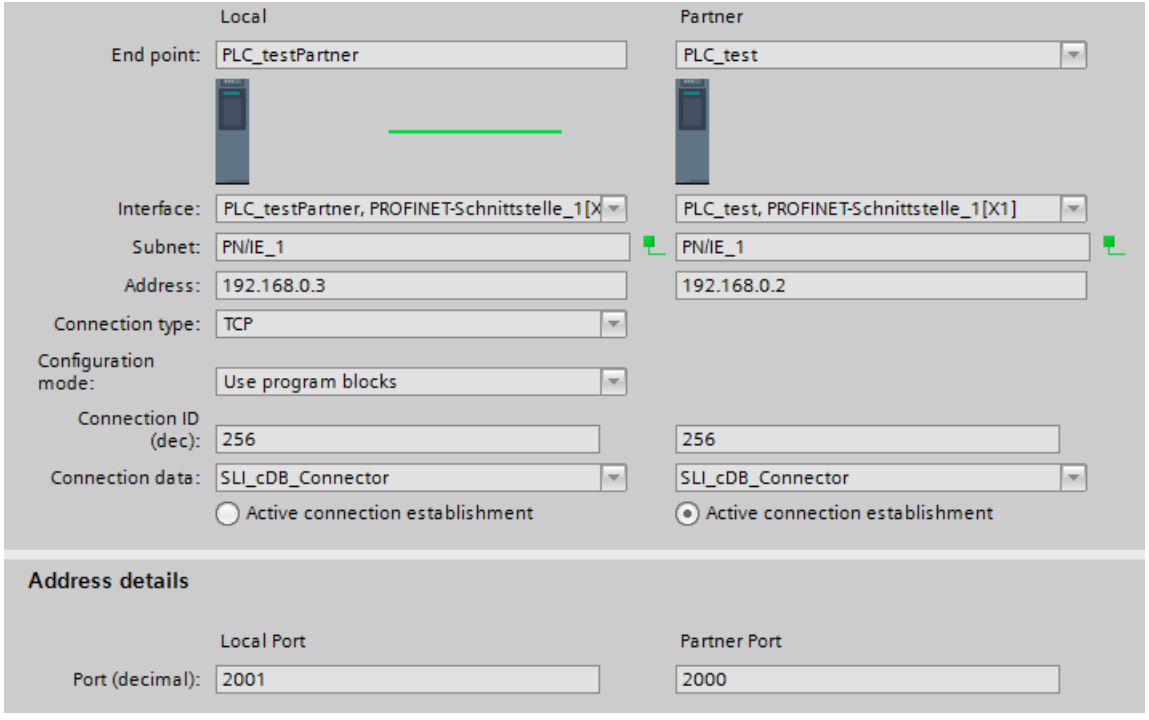
Make the following settings for the configuration of TRCV_C:

Note

Connection data

By configuring TSEND_C, you have already created a data block ("SLI_cDB_Connector") with the stored connection data for each CPU. Instead of creating new data blocks and entering the connection data, you can simply use the created data blocks ("SLI_cDB_Connector").

For the created data blocks to be available for selection, they must still be available in the project tree under Program blocks > System blocks.



Procedure for connection establishment

To establish the communication connection in RUN mode, follow these steps:

1. Change the EN_R parameter to "1" for TRCV_C.
2. Change the CONT parameter to "1" for TRCV_C.
3. Change the REQ parameter to "1" for TSEND_C.
4. Change the CONT parameter to "1" for TSEND_C.

Procedure for connection termination

To terminate the communication connection in RUN mode, follow these steps:

1. Change the CONT parameter to "0" for TSEND_C.
2. Change the REQ parameter to "0" for TSEND_C.
3. Change the CONT parameter to "0" for TRCV_C.
4. Change the EN_R parameter to "0" for TRCV_C.

Behavior of CPU 1

When the input parameter REQ ("start") returns the signal state "TRUE", the "TSEND_C" instruction is started. When the input parameter CONT ("comControl") supplies the signal state "TRUE", the "TSEND_C" instruction creates a communication connection between CPU 1 and CPU 2. To do this, the connection data is retrieved with the input parameter CONNECT (or the data block "SLI_cDB_Connector").

For multiple calls, the "TSEND_C" instruction transmits the data record detected at input parameter DATA ("sendData"). Successful transfer of the data record is indicated by the output parameter DONE ("#statDone") with the signal state "TRUE" and the output parameter STATUS ("status") with the value "0000". Because the values of the output parameters are only displayed for the moment of their validity, save the success status of DONE ("#statDone") in the tag "done".

After transmitting the data record to CPU 2, the communication connection continues to be monitored (status "7004"). The output parameter ERROR ("error") or the tag "memErrStatus" indicates that processing in the example is taking place without errors.

SLI_gDB_TSEND_C

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	comControl	Bool	false	TRUE
4	done	Bool	false	TRUE
5	busy	Bool	false	FALSE
6	error	Bool	false	FALSE
7	status	Word	16#0	16#7004
8	memErrStatus	Word	16#0	16#0000
9	sendData	String	'ProcessXYZ'	'ProcessXYZ'

Behavior of CPU 2

When the input parameter EN_R ("start") returns the signal state "TRUE", the "TRCV_C" instruction is started. When the input parameter CONT ("comControl") supplies the signal state "TRUE", the "TRCV_C" instruction creates a communication connection between CPU 2 and CPU 1. To do this, the connection data is retrieved with the input parameter CONNECT (or the data block "SLI_cDB_Connector").

For multiple calls, the "TRCV_C" instruction receives the transmitted data record. The data record is recorded at the parameter DATA ("rcvData"). Successful transfer of the data record is indicated by the output parameter DONE ("done") with the signal state "TRUE" and the output parameter STATUS ("status") with the value "0000". The length in BYTE of the actually transmitted data record is detected with the output parameter RCVD_LEN ("#statRcvLen"). This value is only displayed during the success status. Then "0" is detected.

Because the values of the output parameters are only displayed for the moment of their validity, do the following:

- Save the value of "#statRcvLen" in "rcvLen".
- Save the value of "#statDone" in "done".
- Save the value of "#status" in "memErrStatus".

After the data record has been received by CPU 2, the communication connection continues to be monitored (status "7006"). The output parameter ERROR ("error") or the tag "memErrStatus" indicates that processing of the example is taking place without errors.

SLI_gDB_TRCV_C

	Name	Data type	Start value	Monitor value
1	Static			
2	start	Bool	false	TRUE
3	comControl	Bool	false	TRUE
4	done	Bool	false	TRUE
5	busy	Bool	false	TRUE
6	error	Bool	false	FALSE
7	status	Word	16#0	16#7006
8	memErrStatus	Word	16#0	16#0000
9	rcvLen	UDInt	0	256
10	rcvData	String	''	'ProcessXYZ'

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

TMAIL_C: Transfer email (S7-1200, S7-1500)

Description of TMAIL_C (S7-1200, S7-1500)

Description

You can use the "TMAIL_C" instruction to send an e-mail via the Ethernet interface of the S7-1500 CPU or S7-1200 > V4.0, a communication module (CM), or a communication processor (CP).

The instruction can only be used once the hardware has been configured and if the network infrastructure allows for a communication connection to the mail server.

You define the content of the e-mail, and the connection data, using the following parameters:

- You define the recipient addresses with the parameters TO_S and CC.
- You define the content of the e-mail with the parameters SUBJECT and TEXT.
- You can define an attachment using VARIANT pointers at the ATTACHMENT and ATTACHMENT_NAME parameters.
- The connection data is defined, and addressing and authentication for the mail server executed, using the system data type TMail_V4, TMail_V6 or TMail_FQDN at the MAIL_ADDR_PARAM parameter.
  - If you are using the interface of the S7-1500 CPU, the system data type TMail_V4 must be used. In this case, the e-mail can only be sent via SMTP.
  - Any of the system data types can be used if you are using the interface of a CM/CP. The e-mail can then also be sent via SMTPS.
- You start the sending of an e-mail with an edge change from "0" to "1" for the REQ parameter.
- The job status is indicated by the output parameters "BUSY", "DONE", "ERROR" and "STATUS".

You cannot send an SMS directly with the "TMAIL_C" instruction. Whether or not the e-mail can be forwarded by the mail server as an SMS depends on your telecommunications provider.

Note

Number of e-mails to be sent

You can send more than one e-mail simultaneously using a PLC. When a CP 1243-8 or CP 1543-1 is used, you can only send one e-mail for each CP. When two CPs are used, parallel sending of two e-mails is possible.

Operation of the instruction

The "TMAIL_C" instruction works asynchronously, which means its execution extends over multiple calls. You must specify an instance when you call the instruction "TMAIL_C".

In the following cases, the connection to the mail server will be lost:

- If the CPU switches to STOP while "TMAIL_C" is active.
- If communication problems occur at the Industrial Ethernet bus.

In this case, the transfer of the e-mail will be interrupted and it will not reach its recipient. The connection is also canceled once the instruction has been successfully executed and the e-mail sent.

NOTICE
Changing user programs You can change the parts of your user program that directly affect calls of "TMAIL_C" only: • The CPU is in "STOP" mode. • No e-mail is being sent (REQ = 0 and BUSY = 0). This relates, in particular, to deleting and replacing program blocks that contain "TMAIL_C" calls or calls for the instance of "TMAIL_C". Ignoring this restriction can tie up connection resources. The automation system can change to an undefined status with the TCP/IP communication functions via Industrial Ethernet. A warm or cold restart of the CPU is required after the changes are transferred.

Data consistency

The TO_S, CC, SUBJECT, TEXT, ATTACHMENT and MAIL_ADDR_PARAM parameters are applied by the "TMAIL_C" instruction while it is running, which means that they may only be changed after the job has been completed (BUSY = 0).

SMTP authentication

Authentication refers here to a procedure for verifying identity, for example, with a password query.

If you are using the S7-1500 CPU interface, the instruction "TMAIL_C" supports the SMTP authentication procedure AUTH-LOGIN which is required by most mail servers. For information about the authentication procedure of your mail server, please refer to your mail server manual or the website of your Internet service provider.

- Before you can use the AUTH-LOGIN authentication procedure, the "TMAIL_C" instruction requires the user name with which it is to log on to the mail server. This user name corresponds to the user name with which you set up a mail account on your mail server. It is transferred via the UserName parameter to the structure at parameter MAIL_ADDR_PARAM.
  If no user name is specified at the MAIL_ADDR_PARAM parameter, the AUTH-LOGIN authentication procedure is not used. The e-mail is then sent without authentication.
- To log on, the "TMAIL_C" instruction also requires the associated password. This password corresponds to the password you specified when you set up your mail account. It is transferred via the PassWord parameter to the structure at parameter MAIL_ADDR_PARAM.

Parameters

The following table shows the parameters of the "TMAIL_C" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L, T, C or constant	Control parameter REQUEST: Activates the sending of an e-mail upon a rising edge.
TO_S (Page 4458)	Input	STRING	D, L or constant	Recipient addresses STRING with a maximum length of 240 characters (bytes). For the e-mail address format, see the example in the parameter description.
CC (Page 4458)	Input	STRING	D, L or constant	CC recipient addresses (optional) STRING with a maximum length of 240 characters (bytes). Same e-mail address format as for the TO_S parameter. If an empty string is assigned here, the e-mail is not sent to a CC recipient.
SUBJECT	Input	STRING	D, L or constant	Subject of the e-mail STRING with a maximum length of 240 characters (bytes).
TEXT	Input	STRING	D, L or constant	Text of the e-mail (optional) STRING with a maximum length of 240 characters (bytes). If an empty string is assigned at this parameter, the e-mail is sent without text.
ATTACHMENT	Input	VARIANT	D	E-mail attachment (optional) Reference to a byte/word/double word field (ArrayOfByte, ArrayOfWord or ArrayOfDWord) with a maximum length of 64 Kb. If no value is assigned, the e-mail is sent without an attachment.
ATTACHMENT_NAME	Input	STRING	D, L or constant	E-mail attachment name (optional) Reference to a character string with a maximum length of 50 characters (bytes) to define the file name of the attachment. If an empty string is assigned at this parameter, the attachment of the e-mail is received with a file name assigned by the e-mail receiving program. It is therefore recommended to use a defined file name.
MAIL_ADDR_PARAM (Page 4459)	Input	VARIANT	D	Connection parameter and address of the e-mail server To define the connection parameters, use the structure TMail_V4, TMail_V6 or TMail_FQDN (see parameter description).

Parameter	Declaration	Data type	Memory area	Description
DONE (Page 4468)	Output	BOOL	I, Q, M, D, L	Status parameter • DONE = 0: Job not yet started or still executing. • DONE = 1: Job executed without errors.
BUSY (Page 4468)	Output	BOOL	I, Q, M, D, L	Status parameter • BUSY = 0: Processing of "TMAIL_C" was terminated • BUSY = 1: Sending of an e-mail is not yet complete.
ERROR (Page 4468)	Output	BOOL	I, Q, M, D, L	Status parameter • ERROR = 0: No error occurred. • ERROR = 1: An error occurred during processing. STATUS supplies detailed information on the type of error.
STATUS (Page 4469)	Output	WORD	I, Q, M, D, L	Status parameter Return value or error information of the "TMAIL_C" instruction (see parameter description).

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

Optional parameters

The optional parameters CC, TEXT, and ATTACHMENT are only sent with the e-mail if the corresponding parameters contain a string of length > 0.

Example

You will find an example of sending e-mails with the TMAIL_CExample: Sending an e-mail with TMAIL_C (Page 4472) instruction with the following link:

See also

Difference between synchronous and asynchronous instructions (Page 2571)

Description TMAIL_C as of version V4.0 (S7-1200, S7-1500)

Introduction

The instruction TMAIL_C as of version V4.0 is described below. This description does not include all details, but instead explains the differences as compared to the instruction versions < V4.0.

The handling of the required certificates is also not described here.

Description

You use the instruction TMAIL_C as of version V4.0 to send an e-mail using Secure Communication via the Ethernet interface of the following modules:

- CP 1543-1 V2.0
- CP 1542SP-1 IRC V1.0
- CP 1543SP-1 V1.0
- CP 1242-7 GPRS V2, Firmware version V2.1
- CP 1243-7 LTE V2.1
- CP 1243-8 V2.1

The files required for the sending process including the TCP port of the e-mail server are defined with one of the system data types TMAIL_V4_SEC, TMAIL_V6_SEC or TMAIL_QDN_SEC at the parameter MAIL_ADDR_PARAM.

Note

Compatibility with the TMAIL_C versions < V4.0

For reasons of compatibility, you can also use the existing system data types TMAIL_V4 and TMAIL_FQDN. In this case, you can also send an email via the Ethernet interfaces of the CPU and the Ethernet interface of the CP 1543-1.

Restrictions for the CP

You must observe the following rules:

- You may use only one CP as email client for each TMAIL_C instance.
- All email servers to which emails are to be sent must be reachable via the IP network from a CP that is used as email client.
- If an email server is to be reached via its fully qualified domain name, the DNS servers, which are configured in the CP, must also be reachable in the IP network.

Parameters

Parameter	Declaration	Data type	Memory area	Description
...				
MAIL_ADDR_PARAM (Page 4462)	Input	VARIANT	D	The data required for the send process (including the connection parameters, the address of the email server and the TCP port used) Select one of the following system data types for this: • TMail_V4_SEC • TMail_V6_SEC • TMail_QDN_SEC
...				

Diagnostic options when an error occurs

You have the following two diagnostic options when an error occurs:

- Evaluation of the STATUS parameter
  The same values can occur as with the instruction-versions < V4.0.
  Only the value W#16#8015 receives an extension. It also occurs if you specify the hardware identifier of a CPU interface at the parameter InterfaceId.
- Evaluation of the MS_STATUS tag in the instance data of TMAIL_C. This tag has been added in the version V4.0 of TMAIL_C. It contains both the protocol codes and their associated textual descriptions as they are defined in RFC 3207 "SMTP Service Extension for Secure SMTP over Transport Layer Security".
  Examples of diagnostic information in MS_STATUS:
  - 550 SMTP AUTH required before submission
  - 550 StartTLS required
  - 550 Requested action not taken

TO_S and CC parameters (S7-1200, S7-1500)

Description

The TO_S and CC parameters are strings, for example, with the following content:

- <wenna@mydomain.com>, <ruby@mydomain.com>
- <admin@mydomain.com>, <judy@mydomain.com>

Note the following rules when entering the parameters:

- A space and an opening pointed bracket "<" must be entered before each address.
- A closing pointed bracket ">" must be entered after each address.
- A comma must be entered between the addresses in TO and CC.

For runtime and memory space reasons, the "TMAIL_C" instruction does not perform a syntax check of parameter TO_S or CC.

MAIL_ADDR_PARAM parameter (S7-1200, S7-1500)

Description

At the MAIL_ADDR_PARAM parameter, you define the connection for sending the e-mail in the structure TMail_V4, TMail_V6 or Tmail_FQDN, and save the e-mail server address and login details.

The structure you use at the MAIL_ADDR_PARAM parameter depends on the format in which the e-mail server is to be addressed:

- TMail_V4: Addressing by IP address to IPv4.
- TMail_V6: Addressing by IP address to IPv6.
- TMail_FQDN: Addressing by fully qualified domain name (FQDN).

Which structure you can use depends on the interface addressed at the InterfaceId parameter:

- If you want to use the "TMAIL_C" instruction with the internal interface, the TMail_V4 structure must be used at the MAIL_ADDR_PARAM parameter.
- If you are using a communication processor (CP) or communication module (CM), you can use all three addressing options (IPv4, IPv6 and FQDN).

Table 11-79 **TMail_V4: Addressing the mail server by IP address (IPv4)**

Parameter	Data type	Description
TMail_V4	Struct	
InterfaceId	LADDR	Hardware identifier of the interface
ID	CONN_OUC	Connection ID
ConnectionType	BYTE	Connection type. Select 16#20 as the connection type for IPv4.
ActiveEstablished	BOOL	Status bit. Set to "1" once the connection is established.
CertIndex	BYTE	• =0: SMTP used (**S**imple **M**ail **T**ransfer **P**rotocol). SMTP must be used if the e-mail is being sent via the interface of an S7-1500 CPU. • ≠0: SMTPS used to secure the connection before it is established (with CPs/CMs).
WatchDogTime	TIME	Execution watchdog. Use this parameter to define the maximum execution time for the send operation. Note: Connection establishment can take longer (approx. one minute) if the connection is slow. When you specify the WATCH_DOG_TIME parameter, remember to allow for the time required to establish the connection. The connection is terminated once the specified time has elapsed.
MailServerAddress	IP_V4	IP address of the mail server. IPv4 in the following format: XXX.XXX.XXX.XXX (decimal). Example: 192.142.131.237.
UserName	STRING[254]	Mail server login name
PassWord	STRING[254]	Mail server password
From	EMAIL_ADDR	E-mail sender address, which is defined using the following two STRING parameters. For example: "myname@mymailserver.com".
LocalPartPlusAtSign	STRING[64]	Local part of sender address, including @ sign. Example: "myname@"
FullQualifiedDomainName	STRING[254]	Fully Qualified Domain Name (abbreviated to FQDN) of the mail server. Example: "mymailserver.com"

Table 11-80 **TMail_V6: Addressing the mail server by IP address to IPv6**

Parameter	Data type	Description
TMail_V6	Struct	
InterfaceId	LADDR	Hardware identifier of the interface
ID	CONN_OUC	Connection ID
ConnectionType	BYTE	Connection type. Select 16#21 as the connection type for IPv6.
ActiveEstablished	BOOL	Status bit. Set to "1" once the connection is established.
CertIndex	BYTE	• =0: SMTP used (**S**imple **M**ail **T**ransfer **P**rotocol). SMTP must be used if the e-mail is being sent via the interface of an S7-1500 CPU. • ≠0: SMTPS used to secure the connection before it is established (with CPs/CMs). You use the CertIndex parameter to specify the certificate to be used (see "Project navigation" > "Global security settings" > "Certificate manager").
WatchDogTime	TIME	Execution watchdog. Use this parameter to define the maximum execution time for the send operation. Note: Connection establishment can take longer (approx. one minute) if the connection is slow. When you specify the WATCH_DOG_TIME parameter, remember to allow for the time required to establish the connection. The connection is terminated once the specified time has elapsed.
MailServerAddress	IP_V6	IP address of the mail server (IPv6) in the following format: XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX (hexadecimal). The address is divided into 8 blocks of 2 bytes each (16 bytes in total). Example: 2001:db8:1f11:08d3:290:27ff:0370:2093
UserName	STRING[254]	Mail server login name
PassWord	STRING[254]	Mail server password
From	EMAIL_ADDR	E-mail sender address, which is defined using the following two STRING parameters. For example: "myname@mymailserver.com".
LocalPartPlusAtSign	STRING[64]	Local part of sender address, including @ sign. Example: "myname@"
FullQualifiedDomainName	STRING[254]	Fully Qualified Domain Name (abbreviated to FQDN) of the mail server. Example: "mymailserver.com"

Table 11-81 **TMail_FQDN: Addressing the mail server via the FQDN**

Parameter	Data type	Description
TMail_FQDN	Struct	
InterfaceId	LADDR	Hardware identifier of the interface
ID	CONN_OUC	Connection ID
ConnectionType	BYTE	Connection type. Select 16#22 as the connection type for FQDN.
ActiveEstablished	BOOL	Status bit. Set to "1" once the connection is established.
CertIndex	BYTE	• =0: SMTP used (**S**imple **M**ail **T**ransfer **P**rotocol). SMTP must be used if the e-mail is being sent via the interface of an S7-1500 CPU. • ≠0: SMTPS used to secure the connection before it is established (with CPs/CMs). You use the CertIndex parameter to specify the certificate to be used (see "Project navigation" > "Global security settings" > "Certificate manager").
WatchDogTime	TIME	Execution watchdog. Use this parameter to define the maximum execution time for the send operation. Note: Connection establishment can take longer (approx. one minute) if the connection is slow. When you specify the WATCH_DOG_TIME parameter, remember to allow for the time required to establish the connection. The connection is terminated once the specified time has elapsed.
MailServerAddress	STRING[254]	FQDN (**F**ully **Q**ualified **D**omain **N**ame) of the mail server. The mail server is addressed using the fully qualified domain name. Example: "www.mymailserver.com."
UserName	STRING[254]	Mail server login name
PassWord	STRING[254]	Mail server password
From	Struct	E-mail sender address, which is defined using the following two STRING parameters. For example: "myname@mymailserver.com".
LocalPartPlusAtSign	STRING[64]	Local part of sender address, including @ sign. Example: "myname@"
FullQualifiedDomainName	STRING[254]	Fully Qualified Domain Name (abbreviated to FQDN) of the mail server. Example: "mymailserver.com"

Parameter MAIL_ADDR_PARAM as of Version 4.0 of TMAIL_C (S7-1200, S7-1500)

Description

You use the MAIL_ADDR_PARAM parameter to specify the data required for the send process, for example the connection parameters, the address of the email server and the TCP port used.

Depending on the format in which the e-mail server is to be addressed, you use one of the following structures at the MAIL_ADDR_PARAM parameter:

- TMail_V4_SEC: Addressing via the IP address in IPv4 format
- TMail_V6_SEC: Addressing via the IP address in IPv6 format
- TMail_QDN_SEC: Addressing via the fully qualified host name (FQDN)

TMail_V4_SEC: Addressing of the mail server via the IP address in IPv4 format

Parameter	Data type	Description
TMail_V4_SEC	Struct	
InterfaceId	LADDR	Hardware identifier of the Ethernet interface
ID	CONN_OUC	Connection ID
ConnectionType	BYTE	Connection type. Select 16#20 as the connection type for IPv4.
ActiveEstablishment	BOOL	Active/passive connection establishment. Because the CP is always the SMTP client, this parameter must be set to "1".
WatchDogTime	TIME	Execution watchdog. Use this parameter to define the maximum execution time for the send operation. Note: Connection establishment can take longer (approx. one minute) if the connection is slow. When you specify the WatchDog-Time parameter, remember to allow for the time required to establish the connection. The connection is terminated once the specified time has elapsed.
MailServerAddress	IP_V4	IP address of the mail server intrinsic in IPv4 format: XXX.XXX.XXX.XXX (decimal place). Example: 192.142.131.237.
UserName	STRING[254]	User name. This is required if users wants to access their email inbox in order to identify themselves as the owner of the email inbox to the email provider.
PassWord	STRING[254]	User password. This is required if users wants to access their email inbox in order to identify themselves as the owner of the email inbox to the email provider.
From	EMAIL_ADDR	E-mail sender address, which is defined using the following two STRING parameters. For example: "myname@mymailserver.com".
LocalPartPlusAtSign	STRING[64]	Local part of sender address, including @ sign. Example: "myname@"

Parameter		Data type	Description
	FullQualifiedDomainName	STRING[254]	Fully Qualified Domain Name (abbreviated to FQDN) of the mail server. Example: "mymailserver.com"
RemotePort		UINT	TCP port of the mail server
ActivateSecureConn		BOOL	• 1: Secure SMTP connection • 0: SMTP connection (non-secure). In this case, the following parameters are irrelevant.
ExtTLSCapabilities		BYTE	Not currently used
TLSServerCertRef		UDINT	Reference to the X.509 V3 (CA) certificate of the mail server, which is used by the TLS client to validate the authentication of the TLS server.

TMail_V6_SEC: Addressing of the e-mail server via the IP address in IPv6 format

Parameter	Data type	Description
TMail_V6_SEC	Struct	
InterfaceId	LADDR	Hardware identifier of the Ethernet interface
ID	CONN_OUC	Connection ID
ConnectionType	BYTE	Connection type. Select 16#21 as the connection type for IPv6.
ActiveEstablishment	BOOL	Active/passive connection establishment. Because the CP is always the SMTP client, this parameter must be set to "1".
WatchDogTime	TIME	Execution watchdog. Use this parameter to define the maximum execution time for the send operation. Note: Connection establishment can take longer (approx. one minute) if the connection is slow. When you specify the WatchDog-Time parameter, remember to allow for the time required to establish the connection. The connection is terminated once the specified time has elapsed.
MailServerAddress	IP_V6	IP address of the mail server in IPv6 format: XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX (hexadecimal). The address is divided into 8 blocks of 2 bytes each (16 bytes in total). Example: 2001:db8:1f11:08d3:290:27ff:0370:2093
UserName	STRING[254]	User name. This is required if users wants to access their e-mail inbox in order to identify themselves as the owner of the e-mail inbox to the e-mail provider.
PassWord	STRING[254]	User password. This is required if users wants to access their e-mail inbox in order to identify themselves as the owner of the e-mail inbox to the e-mail provider.
From	EMAIL_ADDR	E-mail sender address, which is defined using the following two STRING parameters. For example: "myname@mymailserver.com".
LocalPartPlusAtSign	STRING[64]	Local part of sender address, including @ sign. Example: "myname@"
FullQualifiedDomainName	STRING[254]	Fully Qualified Domain Name (abbreviated to FQDN) of the mail server. Example: "mymailserver.com"

Parameter	Data type	Description
RemotePort	UINT	TCP port of the mail server
ActivateSecureConn	BOOL	• 1: Secure SMTP connection • 0: SMTP connection (non-secure). In this case, the following parameters are irrelevant.
ExtTLSCapabilities	BYTE	Not currently used
TLSServerCertRef	UDINT	Reference to the X.509 V3 (CA) certificate of the mail server, which is used by the TLS client to validate the authentication of the TLS server.

TMail_QDN_SEC: Addressing of the mail server via FQDN

Parameter	Data type	Description
TMail_QDN_SEC	Struct	
InterfaceId	LADDR	Hardware identifier of the Ethernet interface
ID	CONN_OUC	Connection ID
ConnectionType	BYTE	Connection type. Select 16#22 as the connection type for FQDN.
ActiveEstablishment	BOOL	Active/passive connection establishment. Because the S7-CP is always the SMTP client, this parameter must be set to "1".
WatchDogTime	TIME	Execution watchdog. Use this parameter to define the maximum execution time for the send operation. Note: Connection establishment can take longer (approx. one minute) if the connection is slow. When you specify the WatchDogTime parameter, remember to allow for the time required to establish the connection. The connection is terminated once the specified time has elapsed.
MailServerQDN	STRING[254]	FQDN (Fully Qualified Domain Name) of the mail server. The mail server is addressed using the fully qualified domain name, which must be end with "." Example: "www.mymailserver.com"
UserName	STRING[254]	User name This is required if users wants to access their email inbox in order to identify themselves as the owner of the email inbox to the email provider.
PassWord	STRING[254]	User password This is required if users wants to access their email inbox in order to identify themselves as the owner of the email inbox to the email provider.
From	EMAIL_ADDR	E-mail sender address, which is defined using the following two STRING parameters. For example: "myname@mymailserver.com".
LocalPartPlusAtSign	STRING[64]	Local part of sender address, including @ sign. Example: "myname@"
FullQualifiedDomainName	STRING[254]	Fully Qualified Domain Name (abbreviated to FQDN) of the mail server. Example: "mymailserver.com"

Parameter	Data type	Description
RemotePort	UINT	TCP port of the mail server
ActivateSecureConn	BOOL	• 1: Secure SMTP connection • 0: Unsecured SMTP connection In this case, the following parameters are irrelevant.
ExtTLSCapabilities	BYTE	Not currently used
TLSServerCertRef	UDINT	Reference to the X.509 V3 (CA) certificate of the mail server, which is used by the TLS client to validate the authentication of the TLS server.

DONE, BUSY and ERROR parameters (S7-1200, S7-1500)

Description

The output parameters DONE, BUSY and ERROR are each displayed for only one cycle if the status of the BUSY output parameter changes from "1" to "0".

The following table shows the relationship between DONE, BUSY, and ERROR. Using this table, you can determine the current status of the instruction "TMAIL_C and when the sending of the e-mail is complete.

DONE	BUSY	ER-ROR	Description
0	1	0	The job is being processed.
1	0	0	Job successfully completed.
0	0	1	The job ended with an error. The cause of the error can be found in the STATUS (Page 4469) parameter.
0	0	0	The "TMAIL_C" instruction was not assigned a (new) job.

STATUS parameter (S7-1200, S7-1500)

Description

The following table shows the return values of "TMAIL_C" at the STATUS parameter:

Return value STATUS* (W#16#...):	Explanation	Notes
0000	The processing of "TMAIL_C" was completed without errors.	Error-free completion of "TMAIL_C" does not mean that the e-mail sent will necessarily arrive. Incorrectly entering the recipient addresses does not generate a status error of the "TMAIL_C" instruction. In this case, there is no guarantee that the e-mail will be sent to other recipients, even if these were entered correctly.
7001	"TMAIL_C" is active (BUSY = 1).	First call: Job triggered.
7002	"TMAIL_C" is active (BUSY = 1).	Intermediate call: Job already active.
8xxx	The processing of "TMAIL_C" was completed with an error code of the communication instructions called internally.	For detailed information, please refer to the STATUS parameter descriptions for the "TCON (Page 4478)", "TDISCON (Page 4484)", "TSEND (Page 4490)" and "TRCV (Page 4493)" communication instructions.
8010	Error during connection establishment	You can find further information on evaluation in the SFB_STATUS parameter of the instance data block. The error code displayed at the SFB_STATUS parameter is explained in the STATUS parameter description for the "TCON (Page 4478)" instruction.
8011	Error sending the data	You can find further information on evaluation in the SFB_STATUS parameter of the instance data block. The error code displayed at the SFB_STATUS parameter is explained in the STATUS parameter description for the "TSEND (Page 4490)" instruction.
8012	Error receiving the data	You can find further information on evaluation in the SFB_STATUS parameter of the instance data block. The error code displayed at the SFB_STATUS parameter is explained in the STATUS parameter description for the "TRCV (Page 4493)" instruction.
8013	Error during connection establishment	You can find further information on evaluation in the SFB_STATUS parameter of the instance data block. The error code displayed at the SFB_STATUS parameter is explained in the STATUS parameter description for the TCON (Page 4478)" and "TDISCON (Page 4484)" instructions.

Return value STATUS* (W#16#...):	Explanation	Notes
8014	Establishment of a connection is not possible.	You may have entered an incorrect mail server IP address (MailServerAddress (Page 4459)) or too short a time span (WatchDogTime (Page 4459)) for connection establishment. It is also possible that the CPU has no connection to the network or that the CPU configuration is incorrect.
8015	Incorrect data type for MAIL_ADDR_PARAM	The only valid data types are the system data types (structures) TMail_V4, TMail_V6 and TMail_FQDN.
8016	Incorrect data type for the ATTACHMENT parameter	The only valid data types are ArrayOfByte, ArrayOfWord and ArrayOfDWord.
8017	Incorrect data length for the ATTACHMENT parameter	Data length must be <= 65534 bytes.
8401	No channels available. Possible cause: There is already an e-mail connection via the CP. A second connection cannot be established in parallel.	Specific error for CP 1543
8403	A TCP/IP connection to the mail server was not possible.	Specific error for CP 1543
8405	Server has denied login request.	Specific error for CP 1543
8406	SMTP client has detected an internal SSL error or a problem with the certificate structure.	Specific error for CP 1543
8407	Request to use SSL denied.	Specific error for CP 1543
8408	Client unable to identify socket for establishing a TCP/IP connection to the mail server.	Specific error for CP 1543
8409	Writing via the connection not possible. Possible cause: The communication partner has reset the connection or the connection has been lost.	Specific error for CP 1543
8410	Reading via the connection not possible. Possible cause: The communication partner has reset the connection or the connection has been lost.	Specific error for CP 1543
8411	E-mail could not be sent. Cause: Insufficient memory space to execute the send operation.	Specific error for CP 1543
8412	Configured DNS server was unable to resolve the specified domain name.	Specific error for CP 1543
8413	An internal error in the DNS subsystem prevented domain name resolution.	Specific error for CP 1543
8414	Empty character string entered as domain name.	Specific error for CP 1543
8415	An internal error has occurred in the cURL module. Execution was stopped.	Specific error for CP 1543

Return value STATUS* (W#16#...):	Explanation	Notes
8416	An internal error has occurred in the SMTP module. Execution was stopped.	Specific error for CP 1543
8417	Request to SMTP on a channel already in use or invalid channel ID. Execution was stopped.	Specific error for CP 1543
8418	E-mail send job aborted. Possible causes: Execution time exceeded (WatchDogTime parameter) or CP transition from start to stop.	Specific error for CP 1543
8419	Channel interrupted and cannot be used before the connection is closed.	Specific error for CP 1543
8420	Server certificate string could not be verified with CP root certificate.	Specific error for CP 1543
8421	Internal error occurred. Execution was stopped.	Specific error for CP 1543
82xx, 84xx, or 85xx	The error message originates from the mail server and corresponds, except for the "8", to the error number of the SMTP protocol. The following lines list several error codes that can occur.	You can find more detailed information on the SMTP error code and other error codes in the SMTP protocol on the Internet or in the error documentation of the mail server. You can also view the most recent error message from the mail server in your instance DB in the BUFFER1 parameter. You can find the last data sent by the "TMAIL_C" instruction under DATEN in the instance DB.
8450	Action not executed: Mailbox not available/cannot be reached	Try again later.
8451	Action aborted: Local processing error	Try again later.
8500	Syntax error: Error not recognized. This also includes the error when a command string is too long. This can also occur when the e-mail server does not support the LOGIN authentication procedure.	Check the parameters of "TMAIL_C". Try to send an e-mail without authentication. To do this, replace the content of the UserName parameter with an empty string. If no user name is specified, the LOGIN authentication procedure is not used.
8501	Syntax error: Incorrect input at a parameter	Possible cause: Incorrect address at the parameters TO_S or CC (see also: TO_S and CC parameters (Page 4458)).
8502	Command unknown or not implemented	Check your entries, in particular the FROM parameter. It may be incomplete and you may have forgotten the "@" or "." (see also: TO_S and CC parameters (Page 4458)).
8504	Command parameters not implemented	Check your entries, in particular the FROM parameter. It may be incomplete and you may have forgotten the "@" (see also: TO_S and CC parameters (Page 4458)).
8535	SMTP authentication incomplete	You have possibly entered an incorrect user name or incorrect password.

Return value STATUS* (W#16#...):	Explanation	Notes
8550	Mail server cannot be reached. You have no access rights.	You may have entered an incorrect user name or password, or the mail server may not support your login. Another cause of error could be an incorrect entry of the domain name after the "@" or a missing "." at the TO_S, CC and FROM parameters (see also: TO_S and CC parameters (Page 4458)).
8552	Action aborted: Assigned memory size has been exceeded	Try again later.
8554	Transfer failed	Try again later.
8555	Error due to incorrect entry of the e-mail address.	If multiple addresses were entered, these need to be separated by a comma.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Example: Sending an e-mail with TMAIL_C (S7-1200, S7-1500)

Introduction

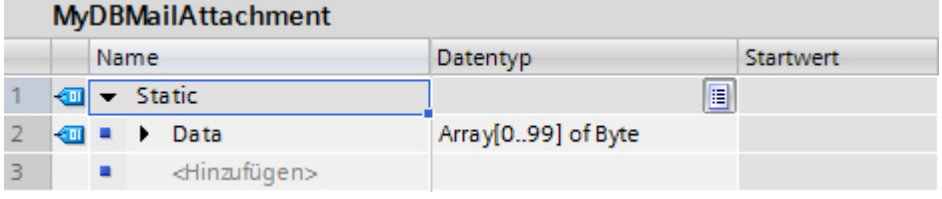
The following call example shows you how the TMAIL_C instruction works.

Requirements

Hardware must be configured beforehand for sending e-mail and the infrastructure of the network must allow a communication connection to the mail server. In addition, you need to create two structures to supply the input parameters of TMAIL_C: One for the e-mail attachment, the other for the connection information and address data.

E-mail attachment

The e-mail should be sent with an attachment. Create an Array of Byte with the name "Data" in a global data block as a source for the attachment. The array will be interconnected with the ATTACHMENT input parameter at a later time.



Connection information and address data

You enter the connection information of the CPU and address data of the mail server in the TMail_V4 system data type:

- Create a "Par1" tag with the "TMail_V4" data type in a global data block.
- In the following structure, enter the parameters of your CPU based on the configuration and the connection data of your mail server. For more information, refer to the description of the MAIL_ADDR_PARAM (Page 4459) input parameter.

The "Par1" tag will be interconnected with the MAIL_ADDR_PARAM input parameter later.

MyDBSendMail

	Name	Datentyp	Startwert
1	Static		
2	Par1	TMail_V4	
3	InterfaceId	HW_ANY	64
4	ID	CONN_OUC	100
5	ConnectionType	Byte	16#20
6	ActiveEstablished	Bool	false
7	CertIndex	Byte	16#0
8	WatchDogTime	Time	T#5s
9	MailServerAddress	IP_V4	
10	ADDR	Array[1..4] of Byte	
11	ADDR[1]	Byte	192
12	ADDR[2]	Byte	168
13	ADDR[3]	Byte	100
14	ADDR[4]	Byte	10
15	UserName	String[254]	'myusername'
16	PassWord	String[254]	'mypassword'
17	From	EMAIL_ADDR	
18	LocalPartPlusAt...	String[64]	'station1@'
19	FullQualifiedD...	String[254]	'mycpu.com'

Calling the instruction

Create a new function block in the SCL language with the following interfaces:

- InOut
  - Name: SendEMail
  - Data type: Bool
- Static
  - Name: STATUS
  - Data type: Word

Copy the following source code with the call for TMAIL_C to the programming window:

Call TMAIL_C

```
BEGIN
    IF #SendEMail = true THEN
     "TMAIL_C_DB"(REQ := NOT "TMAIL_C_DB".BUSY,
                  TEXT := 'The cpu switched to run',
                  ATTACHMENT := "MyDBMailAttachment".Data,
                  MAIL_ADDR_PARAM := "MyDBSendMail".Par1);
     IF ("TMAIL_C_DB".BUSY = false) AND ("TMAIL_C_DB".DONE = false)
AND
     ("TMAIL_C_DB".ERROR = false) THEN
         #SendEMail := false;
     END_IF;
     IF ("TMAIL_C_DB".DONE = false) OR ("TMAIL_C_DB".ERROR = true) THEN
         #STATUS := "TMAIL_C_DB".STATUS;
       END_IF;
    END_IF;
```

Result

This instruction must be called once with SendEMail = true. In the following calls, SendEMail should not be allocated when calling. SendEMail is reset by the instruction shown, as soon as an e-mail has been sent or an error has occurred during sending.

If the email was not sent successfully, the STATUS parameter contains the corresponding value of the STATUS parameter from the TMAIL_C instruction.

See also

Description of TMAIL_C (Page 4453)

Other (S7-1200, S7-1500)

TCON: Establishing a communication connection (S7-1200, S7-1500)

TCON: Establish communication connection (S7-1200)

Validity

The following description of the "TCON" instruction applies to S7-1200 CPUs with firmware version < V4.0.

Description

You use the "TCON" instruction to set up and establish a communication connection. Once the connection has been set up and established, it is automatically maintained and monitored by the CPU. "TCON" executes asynchronously.

The connection data specified for the CONNECT and ID parameters are used to set up the communication connection. To establish the connection, a rising edge must be detected at the REQ parameter. Once the connection is successfully established, the DONE parameter is set to "1".

Note

Support when programming connections

If you select an instruction for communication TCON, TSEND_C or TRCV_C in a program block and create connections of the type TCP, UDP or ISO-on-TCP and want to assign parameters to them, you can use the support of the connection parameter assignment.

You can find the connection parameter assignment the Inspector window of the program editor.

Number of possible connections

For information on the number of possible communication connections, refer to the technical specifications for your CPU.

TCP and ISO-on-TCP connections

Both communication partners call the "TCON" instruction to set up and establish the communication connection. During parameter assignment, you specify which partner is the active communication end point and which is the passive one.

If the connection aborts, for example due to a line break or due to the remote communication partner, the active partner attempts to reestablish the configured connection. You do not have to call "TCON" again. This applies only if "TCON" has been executed successfully once (DONE = 1).

An existing connection is terminated and the connection set up is removed when the "TDISCON (Page 4484)" instruction is executed or when the CPU changes to STOP mode. To set up and establish the connection again, you will need to execute "TCON" again.

Parameters

The following table shows the parameters of the "TCON" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Starts the job to establish the connection specified in the ID upon a rising edge.
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the assigned connection. Value range: W#16#0001 to W#16#0FFF
CONNECT	InOut	TCON_Param	D	Pointer to the connection description See also: AUTOHOTSPOT

Parameter	Declaration	Data type	Memory area	Description
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter ERROR: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

BUSY, DONE and ERROR parameters

You can check the status of the job with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. You use the DONE parameter to check whether or not a job has been executed successfully. The ERROR parameter is set when errors occurred during execution of "TCON". The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

BUSY	DONE	ERROR	Description
1	0	0	The job is being processed.
0	1	0	The job was completed successfully.
0	0	1	The job ended with an error. The cause of the error is output at the STATUS parameter.
0	0	0	No new job was assigned.

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Explanation
0	0000	Connection successfully established.
0	7000	No job processing active
0	7001	Start job execution, establish connection.
0	7002	Connection is being established (REQ irrelevant).
1	8085	Connection ID (ID parameter) is already being used by a configured connection.
1	8086	The ID parameter is outside the valid range.
1	8087	Maximum number of connections reached; no additional connection possible
1	8089	The CONNECT parameter does not point to a data block.
1	809A	The structure at the CONNECT parameter is not supported on an integrated interface or the length is invalid or an incorrect "InterfaceID" is specified in the connection description (SDT).

ERROR	STATUS* (W#16#...)	Explanation
1	809B	The element InterfaceId within the TCON_xxx structure does not reference a hardware identifier of a CPU or CM/CP interface or has the value "0".
1	80A0	Group error for error codes W#16#80A1 and W#16#80A2.
1	80A1	The specified connection or the port is already being used.
1	80A2	Local or remote port is being used by the system.
1	80A3	Attempt being made to re-establish an existing connection.
1	80A4	IP address of the remote endpoint of the connection is invalid, in other words, it matches the IP address of the local partner.
1	80A5	Connection ID is already in use.
1	80A7	Communication error: You executed "TDISCON (Page 4484)" before "TCON" had completed.
1	80B2	The CONNECT parameter points to a data block that was generated with the attribute "Only store in load memory".
1	80B4	You have violated one or more of the following conditions for passive connection establishment with the ISO-on-TCP protocol variant (connection_type = B#16#12): • local_tsap_id_len >= B#16#02 • local_tsap_id[1] = B#16#E0 • With local_tsap_id_len >= B#16#03, local_tsap_id[1] is an ASCII character. • local_tsap_id[1] is an ASCII character and local_tsap_id_len >= B#16#03.
1	80B5	Only passive connection establishment is permitted for connection type 13 = UDP.
1	80B6	Parameter assignment error at the connection_type parameter of the SDT TCON_Param.
1	80B7	Error in one of the following parameters of the data block for connection description: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len. Note: If you call TCON in TCP for the passive end, local_tsap_id_len must have the value 2 and rem_tsap_id_len the value 0.
1	80B8	Connection description of the structure element ID and the block parameter ID are not identical.
1	80C3	All connection resources are in use.
1	80C4	Temporary communication error: • The connection cannot be established at this time. • The connection cannot be established because the firewalls on the connection path are not open for the required ports. • The interface is currently receiving new parameters. • The configured connection is currently being removed by a "TDISCON (Page 4484)" instruction.
* The error codes in the program editor can be displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

See also

Difference between synchronous and asynchronous instructions (Page 2571)

TCON: Establish communication connection (S7-1200, S7-1500)

Description

You use the "TCON" instruction to set up and establish a communication connection. Once the connection has been set up and established, it is automatically maintained and monitored by the CPU. "TCON" executes asynchronously.

For S7-1500-CPUs as of firmware version V2.0 and higher, you can set up an access point for IPv4 multicast communication with the instruction "TCON" via the integrated PROFINET interfaces.

The connection data specified for the CONNECT and ID parameters are used to set up the communication connection. At the CONNECT parameter, if possible, only use the predefined structures as created by the connection parameter assignment in the Inspector window of the program editor.

To establish the connection, a rising edge must be detected at the REQ parameter. Once the connection is successfully established, the DONE parameter is set to "1".

Note

Support when programming connections

If you select an instruction for communication TCON, TSEND_C or TRCV_C in a program block and wish to create and assign parameters to connections of the type TCP, UDP, ISO-on-TCP or FDL, you can use the support of the connection parameter assignment.

You can find the connection parameter assignment in the Inspector window of the program editor.

Number of possible connections

For information on the number of possible communication connections, refer to the technical specifications for your CPU.

Connection with TCP and ISO-on-TCP

Both communication partners call the "TCON" instruction to set up and establish the communication connection. During parameter assignment, you specify which partner is the active communication end point and which is the passive one.

If the connection aborts, for example due to a line break or due to the remote communication partner, the active partner attempts to reestablish the configured connection. You do not have to call "TCON" again. This applies only if "TCON" has been executed successfully once (DONE = 1).

An existing connection is terminated and the connection set up is removed when the "TDISCON (Page 4484)" instruction is executed or when the CPU changes to STOP mode. To set up and establish the connection again, you will need to execute "TCON" again.

Telecontrol connections between CP and SMS client

Use the TCON_PHONE system data type for connection description in the CONNECT parameter.

For this connection type, the station requires access to the mobile network via a mobile network CP.

FDL connections

To set up the FDL connections of the CM 1542-5, see AUTOHOTSPOT.

Parameter

The following table shows the parameters of the instruction "TCON":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Starts the job to establish the connection specified on a rising edge.
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the assigned connection. Value range: W#16#0001 to W#16#0FFF
CONNECT	InOut	VARIANT	D	Pointer to the connection description • For TCP or UDP, use the structure TCON_IP_v4 or TCON_QDN For description, refer to: AUTOHOTSPOT or AUTOHOTSPOT • For TCP or UDP using secure communication, use the structure TCON_IP_V4_SEC or TCON_QDN_SEC For description, refer to: AUTOHOTSPOT or AUTOHOTSPOT • For ISO-on-TCP, use the structure TCON_IP_RFC For a description, refer to: AUTOHOTSPOT • For ISO connections of the CP 1543-1, use the structure TCON_ISOnative. For description, refer to: "Structure of the connection description according to TCON_ISOnative" • For connections to SMS clients, use the TCON_PHONE system data type. For a description, refer to AUTOHOTSPOT. • For FDL connections of the CM 1542-5, use the system data type TCON_FDL; see AUTOHOTSPOT.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or still in progress • 1: Job completed without error

Parameter	Declaration	Data type	Memory area	Description
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter ERROR: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

BUSY, DONE and ERROR parameters

You can check the status of the job with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. You use the DONE parameter to check whether or not a job has been executed successfully. The ERROR parameter is set if errors occur during execution of "TCON". The error information is output at the STATUS parameter.

The instruction "TCON" generates an error message in version 3.0 if an active connection establishment to a remote partner fails. Create a rising edge at the REQ parameter to establish a connection once again.

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

BUSY	DONE	ERROR	Description
1	0	0	The job is being processed.
0	1	0	The job was completed successfully.
0	0	1	The job ended with an error. The cause of the error is output at the STATUS parameter.
0	0	0	No new job was assigned.

Structure of the connection description according to TCON_ISOnative

A connection description DB with a structure according to TCON_ISOnative is used to assign the communication connection parameters for ISO. The fixed data structure of the TCON_ISOnative contains all parameters that are required to establish the connection.

Byte	Parameter	Data type	Start value	Description
0 … 1	InterfaceId	HW_ANY	0	Hardware identifier of the CP interface
2 … 3	ID	CONN_OUC	1	Reference to this connection (unique ID in the value range: 1 to 4095).
4	ConnectionType	BYTE	16#16	Connection type: ISO
5	ActiveEstablished	BOOL	TRUE	ID for the manner in which the connection is established: • FALSE: Passive connection establishment • TRUE: Active connection establishment

Byte	Parameter	Data type	Start value	Description
8 to 13	RemoteMacAddress	ARRAY [1..6] of BYTE	-	MAC address of the partner end point, for example, for 00-74-41-FD-AE-84: • MacAddr[1] = 00 • MacAddr[2] = 74 • MacAddr[3] = 41 • MacAddr[4] = FD • MacAddr[5] = AE • MacAddr[6] = 84
14 - 19	LocalMacAddress	ARRAY [1..6] of BYTE	-	MAC address of the local end point, for example, for 00-74-41-FD-AE-84: • MacAddr[1] = 00 • MacAddr[2] = 74 • MacAddr[3] = 41 • MacAddr[4] = FD • MacAddr[5] = AE • MacAddr[6] = 84
20 - 53	RemoteTSelector	Struct	-	TSelector of the remote connection partner: • Bytes 20 to 21 = TSelLength • Bytes 22 to 53 = TSel[1-32]
	TSelLength	UINT	-	Value range from 0 to 32
	TSel	ARRAY [1..32] of BYTE	-	Value range in each case 0 to 255 in bytes
54 - 87	LocalTSelector	Struct	-	TSelector of the remote connection partner: • Bytes 20 to 21 = TSelLength • Bytes 22 to 53 = TSel[1-32]
	TSelLength	UINT		Value range from 0 to 32
	TSel	ARRAY [1..32] of BYTE	-	Value range in each case 0 to 255 in bytes
88 - 89	CrRetransmitionTime	UINT	-	Duration until connection attempt is repeated in seconds.
90 - 91	DataRetransmition-Time	UINT	100 ms	Duration until data transfer is repeated in milliseconds.
92 - 93	MaxRetransmition-Count	UINT	-	Maximum number of repetitions.
94 - 95	InactivityTime	UINT	-	in seconds
96 - 97	WindowTime	UINT		in seconds

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Explanation
0	0000	Connection successfully established.
0	7000	No job processing active
0	7001	Start job execution, establish connection.
0	7002	Connection is being established (REQ irrelevant).
1	8085	Connection ID (ID parameter) is already being used by a configured connection.
1	8086	The ID parameter is outside the valid range.
1	8087	Maximum number of connections reached; no additional connection possible
1	8089	The CONNECT parameter does not point to a connection description or the connection description was created manually.
1	809A	The structure at the CONNECT parameter is not supported on an integrated interface or the length is invalid.
1	809B	The element InterfaceId within the TCON_xxx structure does not reference a hardware identifier of a CPU or CM/CP interface or has the value "0".
1	80A1	The specified connection or the port is already being used.
1	80A2	Local or remote port is being used by the system. The following ports are reserved locally: 20, 21, 80, 102, 135, 161, 162, 443, 34962, 34963, 34964 as well as the area 49152 to 65535.
1	80A3	ID is used by a connection created by the user program, which uses the same connection description at the CONNECT parameter.
1	80A4	IP address of the remote endpoint of the connection is invalid or it corresponds to the IP address of the local partner.
1	80A7	Communication error: You executed "TDISCON (Page 4484)" before "TCON" had completed.
1	80B4	Only with TCON_IP_RFC: The local T selector was not specified or the first byte does not contain the value 0x0E (only with a length of T selector = 2) or the local T selector starts with "SIMATIC-".
1	80B5	Only passive connection establishment is permitted for connection type 13 = UDP (parameter ActiveEstablished of the structure TCON_IP_v4 / TCON_PARAM has the value TRUE).
1	80B6	Parameter assignment error in the ConnectionType parameter of the data block for connection description. • Only valid with TCON_IP_v4: 0x11, 0x0B and 0x13. • Only valid with TCON_IP_RFC: 0x0C and 0x12
1	80B7	With TCON_IP_v4: • TCP (active connection establishment): Remote port is "0". • TCP (passive connection establishment): Local port is "0". • UDP: Local port is "0". • IP address of the partner end point was set to 0.0.0.0. With TCON_IP_RFC: • Local (LocalTSelector) or remote (RemoteTSelector) T selector was specified with a length of more than 32 bytes. • For TSelLength of the T selector (local or remote), a length greater than 32 was entered. • Error in the length of the IP address of the specific connection partner. • IP address of the partner end point was set to 0.0.0.0.
1	80B8	Parameter ID in the local connection description (structure at CONNECT parameter) and parameter ID of the instruction are different.

ERROR	STATUS* (W#16#...)	Explanation
1	80C3	All connection resources are assigned, or ports may be dynamically used by other applications or connections.
1	80C4	Temporary communication error: • The connection cannot be established at this time. • The connection cannot be established because the firewalls on the connection path are not open for the required ports. • The interface is currently receiving new parameters. • The configured connection is currently being removed by a "TDISCON (Page 4484)" instruction.
1	80C5	The connection partner refuses to establish the connection, has terminated the connection or actively ended it.
1	80C6	The connection partner cannot be reached (network error).
1	80C7	Execution timeout.
1	80C8	Value at the ID parameter is already being used by a connection that was created using the user program. The connection uses the identical ID, but different connection settings at the parameter CONNECT.
1	80C9	Validation of the connection partner failed. The connection partner that wants to establish the connection does not match the defined partner of the structure at the CONNECT parameter.
1	80CE	The IP address of the local interface is 0.0.0.0.
1	80D0	In connection with TCP and the active connection end point: The remote_qdn parameter is an empty string. In this case, no connection can be established.
1	80D1	The remote_qdn parameter is not a fully qualified domain name. The period at the end may be missing.
1	80D2	No DNS server address is configured.
1	80D3	The fully qualified domain name could not be resolved. Possible causes: • The DNS server is not reachable, for example, because it has been shut down or the remote port is not reachable. • An error occurred during communication with the DNS server. • The DNS server returned a valid DNS answer, but the answer contained no IPv4 address.
1	80E0	Unsuitable or poor message was received.
1	80E1	Error during the handshake. Possible causes: • Abort by the user • Security not high enough • Renewed negotiation is not supported • SSL/TLS version is not supported • Validation of the host name failed
1	80E2	Not supported / invalid certificate Possible cause: The time-of-day of the module concerned is not set or the module is not synchronized. Example: The default setting for the date of the module is 1/1/2012 and it was not set during commissioning. The validity period of the certificate starts on 20 August 2016 and ends on 20 August 2024. In this case, the date of the module is outside the validity period of the certificate; the certificate is invalid for the module.
1	80E3	Certificate was discarded.

ERROR	STATUS* (W#16#...)	Explanation
1	80E4	No valid certification authority found.
1	80E5	Certificate expired.
1	80E6	Integrity errors in the Transport Layer Security Protocol
1	80E7	Not supported extension in X.509-V3 certificate
1	80E9	TLS server without server certificate is not supported.
1	80EA	DTLS (UDP) protocol is not supported.
1	80EB	A client cannot request a client certificate.
1	80EC	The server cannot perform validation based on the subjectAlternateName (only clients can do this).
1	80ED	TLSServerCertRef_m-ID invalid
* The error codes in the program editor can be displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".		

Example

You can find the example here: Program example for TCON, TDISCON, TSEND & TRCV (Page 4502).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

TDISCON: Terminate communication connection (S7-1200, S7-1500)

Description

The "TDISCON" instruction terminates a communications connection from the CPU to a connection partner.

Functional description

"TDISCON" is an instruction that works asynchronously, which means its job processing extends over multiple calls. You start the job for terminating a connection by calling the "TDISCON" instruction with REQ = 1.

After "TDISCON" has been successfully executed, the ID specified for "TCON" is no longer valid and cannot be used for sending or receiving.

The job status is indicated by the output parameters BUSY and STATUS. Here, STATUS corresponds to the output parameter RET_VAL of the asynchronous instructions (see also: Difference between synchronous and asynchronous instructions (Page 2571)).

The following table shows the relationship between BUSY, DONE and ERROR. Using this table, you can recognize the current status of "TDISCON" or when the establishment of the connection is completed.

BUSY	DONE	ERROR	Description
1	0	0	The job is being processed.
0	1	0	Job successfully completed.
0	0	1	The job ended with an error. The cause of the error can be found in the STATUS parameter.
0	0	0	The instruction was not assigned a (new) job.

Parameters

The following table shows the parameters of the instruction "TDISCON":

Parameter	Declaration	Data type	Memory area		Description
			S7-1200	S7-1500	
REQ	Input	BOOL	I, Q, M, D, L or constant	I, Q, M, D, L or constant	Control parameter REQUEST starts the job for terminating the connection specified by ID. The job starts on a rising edge.
ID	Input	CONN_OUC (WORD)	D, L or constant	I, Q, M, D, L or constant	Reference to the connection to be terminated (connection ID) Value range: W#16#0001 to W#16#0FFF
DONE	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Status parameter: • 0: Job not yet started or still executing. • 1: Job executed without errors
BUSY	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Status parameter: • BUSY = 1: The job is not yet complete. • BUSY = 0: The job has been completed or has not yet started.
ERROR	Output	BOOL	I, Q, M, D, L	I, Q, M, D, L	Status parameter: • ERROR = 0: No error. • ERROR=1: An error occurred during processing. STATUS supplies detailed information on the type of error
STATUS	Output	WORD	I, Q, M, D, L	I, Q, M, D, L	Status parameter: Error information (see "Parameters ERROR and STATUS")

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Explanation
0	0000	Connection terminated successfully.
0	7000	No job processing active.
0	7001	Start of job processing, connection being terminated.
0	7002	Intermediate call (REQ irrelevant), connection being terminated.
1	8086	The ID parameter is not in the permitted value range.
1	80A3	Attempt is being made to terminate a non-existent connection or the connection is already terminated.
1	80C4	Temporary communication error: New parameters are being assigned to the interface or the connection is currently being established.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Example

You can find the example here: Program example for TCON, TDISCON, TSEND & TRCV (Page 4502).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

TSEND: Send data via communication connection (S7-1200, S7-1500)

TSEND: Send data via communication connection (S7-1200)

Description

The following description of the "TSEND" instruction is valid for the CPU S7-1200 to version 3.0.

You use the "TSEND" instruction to send data over an existing communication connection. TSEND executes asynchronously.

You specify the send area with the DATA parameter. This includes the address and the length of the data to be sent. All data types except BOOL and Array of BOOL can be used for the data to be sent.

The send job is executed when a rising edge is detected at the REQ parameter.

With the LEN parameter, you specify the maximum number of bytes sent with a send job.

- When data is transferred with TCP (streaming protocol), the "TSEND" instruction provides no information about the length of the data sent to "TRCV (Page 4493)".
- When data is transferred with ISO-on-TCP (message-oriented protocol), the length of the data sent is communicated to "TRCV (Page 4493)". The amount of data sent with "TSEND" as packet must also be received again at the receiving end ("TRCV (Page 4493)"):
  - If the receive buffer is too small for the sent data, an error occurs at the receiving end.
  - If the receive buffer is sufficiently large, "TRCV" returns with DONE=1 as soon as the data packet is received.

The data to be sent must not be edited until the send job is completed. If the send job executes successfully, the DONE parameter is set to "1". Signal state "1" at the DONE parameter is not confirmation that the data sent has already been read by the communication partner.

Parameter

The following table shows the parameters of the "TSEND" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Starts the send job on a rising edge.
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the connection established with "TCON". Value range: W#16#0001 to W#16#0FFF
LEN	Input	UINT	I, Q, M, D, L or constant	Maximum number of bytes to be sent with the job.
DATA	InOut	VARIANT	I, Q, M, D	Pointer to the send area containing the address and the length of the data to be sent. The address references: • The process image of the inputs • The process image of the outputs • A bit memory • A data block When transferring structures, the structures must be identical at the sending and receiving end.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or still in progress • 1: Job completed without error
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

LEN and DATA parameters

- With LEN = 0 , all the data specified with the DATA parameter is sent.
- If the number of bytes at the LEN parameter is larger than the length of the data to be sent that was defined with the DATA parameter, the error code 8088 is output at the STATUS parameter (see description of the STATUS parameter below).
- If a structure (Struct) is referenced via the DATA parameter, LEN can be shorter than the structure. In this case, only the data up to the length of the LEN parameter is transferred.
- With data types STRING and WSTRING, all data is transferred if the parameter LEN = 0. When LEN > 0, the length must be at least the maximum number of bytes plus two additional bytes which contain the length information. You will find more detailed information on the structure of the data types in: "Overview of the valid data types (Page 2239)".
- The maximum number of bytes that can be transmitted is 65534.
- When you use structured tags from optimized DBs, the address of the structured tag should be interconnected at the DATA parameter and the parameter LEN = 0. This guarantees type-safe transmission of the entire structure if the same structure is used at the receiving end.

BUSY, DONE and ERROR parameters

You can check the status of the job with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. With the DONE parameter, you can check whether or not a job executed successfully. The ERROR parameter is set when errors occurred during execution of "TSEND". The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

BUSY	DONE	ERROR	Description
1	0	0	The job is being processed.
0	1	0	The job was completed successfully.
0	0	1	The job ended with an error. The cause of the error is specified in the STATUS parameter.
0	0	0	No new job was assigned.

Note

Because "TSEND" is executed asynchronously, you must keep the data in the send area consistent until the DONE parameter or the ERROR parameter changes to the value "1".

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Description
0	0000	Send job completed without error.
0	7000	No job processing active.
0	7001	Start of job execution, data is being sent. When processing this job, the operating system accesses the data in the DATA send area.
0	7002	Job executing (REQ irrelevant). When processing this job, the operating system accesses the data in the DATA send area.
1	8085	• The LEN parameter is greater than the highest permitted value (65536). • DATA and LEN parameters both have the value "0".
1	8086	The ID parameter is outside the permitted address range (1..0xFFF).
1	8088	The LEN parameter is greater than the area specified in DATA.
1	80A1	Communication error: • The specified connection has not yet been established. • The specified connection is being terminated. Transfer via this connection is not possible. • The interface is being re-initialized.
1	80B3	The protocol variant (ConnectionType parameter in the connection description) is set to UDP. Use the instruction "TUSEND" with a UDP connection.
1	80C3	• A block with this ID is already being processed in a different priority group. • Internal lack of resources.
1	80C4	Temporary communication error: • The connection cannot be established to the partner at this time. • The interface is receiving new parameter settings or the connection is being established.
1	80C5	Connection terminated by the communication partner.
1	80C6	Network error. Communication partner cannot be reached.
1	80C7	Execution timeout.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

See also

Difference between synchronous and asynchronous instructions (Page 2571)

TSEND: Send data via communication connection (S7-1200, S7-1500)

Description

The following description of the "TSEND" instruction is valid for:

- Ethernet
  CPU S7-1200 with firmware version ≥ V4.0 and CPU S7-1500
- PROFIBUS
  FDL connections of the S7-1500 with CM 1542-5 as of V2.0 with the system data type TCON_FDL

You use the "TSEND" instruction to send data over an existing communication connection. TSEND executes asynchronously.

You specify the send area with the DATA parameter. This includes the address and the length of the data to be sent. All data types except BOOL and Array of BOOL can be used for the data to be sent.

The send job is executed when a rising edge is detected at the REQ parameter.

With the LEN parameter, you specify the maximum number of bytes sent with a send job.

- When data is transferred with TCP (streaming protocol), the "TSEND" instruction provides no information about the length of the data sent to "TRCV (Page 4493)".
- When data is transferred with ISO-on-TCP (message-oriented protocol), the length of the data sent is communicated to "TRCV (Page 4493)". The amount of data sent with "TSEND" as packet must also be received again at the receiving end ("TRCV (Page 4493)"):
  - If the receive buffer is too small for the sent data, an error occurs at the receiving end.
  - If the receive buffer is sufficiently large, "TRCV" returns with DONE=1 as soon as the data packet is received.

The data to be sent must not be edited until the send job is completed. If the send job executes successfully, the DONE parameter is set to "1". Signal state "1" at the DONE parameter is not confirmation that the data sent has already been read by the communication partner.

Parameter

The following table shows the parameters of the "TSEND" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Starts the send job on a rising edge.
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the connection established with "TCON". Value range: W#16#0001 to W#16#0FFF
LEN	Input	UDINT	I, Q, M, D, L or constant	Maximum number of bytes that are sent with the job (maximum permissible value for S7-1200: 8192, maximum permissible value for S7-1500: 65536). For FDL connections of the CM 1542-5, the maximum length is 240 bytes. In this regard, note the maximum lengths that can be processed by the connection partner.

Parameter	Declaration	Data type	Memory area	Description
DATA	InOut	VARIANT	I, Q, M, D, L	Pointer to the send area containing the address and the length of the data to be sent. The address references: • The process image of the inputs • The process image of the outputs • A bit memory • A data block • Local data When transferring structures, the structures must be identical at the sending and receiving end.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or still in progress • 1: Job completed without error
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

LEN and DATA parameters

- With LEN = 0 , all the data specified with the DATA parameter is sent.
- If the number of bytes at the LEN parameter is larger than the length of the data to be sent that was defined with the DATA parameter, the error code 8088 is output at the STATUS parameter (see description of the STATUS parameter below).
- If a structure (Struct) is referenced via the DATA parameter, LEN can be shorter than the structure. In this case, only the data up to the length of the LEN parameter is transferred.
- With data types STRING and WSTRING, all data is transferred if the parameter LEN = 0. When LEN > 0, the length must be at least the maximum number of bytes plus two additional bytes which contain the length information. You will find more detailed information on the structure of the data types in: "Overview of the valid data types (Page 2239)".
- The maximum number of bytes that can be transmitted depends on the device.
- When you use structured tags from optimized DBs, the address of the structured tag should be interconnected at the DATA parameter and the parameter LEN = 0. This guarantees type-safe transmission of the entire structure if the same structure is used at the receiving end.

BUSY, DONE and ERROR parameters

You can check the status of the job with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. With the DONE parameter, you can check whether or not a job executed successfully. The ERROR parameter is set if errors occur during execution of "TSEND". The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

BUSY	DONE	ERROR	Description
1	0	0	The job is being processed.
0	1	0	The job was completed successfully.
0	0	1	The job ended with an error. The cause of the error is specified in the STATUS parameter.
0	0	0	No new job was assigned.

Note

Because "TSEND" is executed asynchronously, you must keep the data in the send area consistent until the DONE parameter or the ERROR parameter changes to the value "1".

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Description
0	0000	Send job completed without error.
0	7000	No job processing active.
0	7001	Start of job execution, data is being sent. When processing this job, the operating system accesses the data in the DATA send area.
0	7002	Job executing (REQ irrelevant). When processing this job, the operating system accesses the data in the DATA send area.
1	8085	• The LEN parameter is larger than the highest permitted value. For S7-1200: 8192; for S7-1500 (TCP): 65536; for S7-1500 (FDL): 240 / 236 • DATA and LEN parameters both have the value "0".
1	8086	The ID parameter is outside the permitted address range (1..0xFFF).
1	8088	The LEN parameter is greater than the area specified in DATA.
1	80A1	Communication error: • The specified connection has not yet been established. • The specified connection is being terminated. Transfer via this connection is not possible. • The interface is being re-initialized.
1	80B1	You changed the DATA parameter before the current job finished.
1	80B3	The protocol variant (ConnectionType parameter in the connection description) is set to UDP. Use the instruction "TUSEND" with a UDP connection.
1	80C3	• A block with this ID is already being processed in a different priority group. • Internal lack of resources.

ERROR	STATUS* (W#16#...)	Description
1	80C4	Temporary communication error: • The connection cannot be established to the partner at this time. • The interface is receiving new parameter settings or the connection is being established. • Temporarily no receive resources available at the connection partner. The connection partner is not ready to receive.
1	80C5	• Connection terminated by the communication partner. • LSAP of the remote connection partner is not released
1	80C6	Network error: • Remote partner cannot be reached. • Physical interruption on PROFIBUS
1	80C7	Execution timeout.
1	80EE	Handshake not yet completed
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Example

You can find the example here: Program example for TCON, TDISCON, TSEND & TRCV (Page 4502).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

TRCV: Receive data via communication connection (S7-1200, S7-1500)

TRCV: Receive data via communication connection (S7-1200)

Description

The following description of the "TRCV" instruction is valid for the CPU S7-1200 up to version 3.0.

You use the "TRCV" instruction to receive data over an existing communication connection. "TRCV" executes asynchronously.

Receipt of data is enabled when the EN_R parameter is set to the value "1". The received data is entered in a receive area. You specify the length of the receive area either with the LEN parameter (if LEN <> 0) or with the length information of the DATA parameter (if LEN = 0), depending on the protocol variant being used.

While data is being received, you cannot make changes to the DATA parameter or the defined receive area to ensure consistency of the received data.

After successful receipt of data, the NDR parameter is set to the value "1". You can query the amount of data actually received at the RCVD_LEN parameter.

Receive modes of "TRCV"

The following table shows how the received data is entered in the receive area.

Protocol variant	Availability of data in the receive area	connection_type* parameter of the connection description	Parameter LEN
TCP (Ad-hoc mode)	The data is immediately available.	Hexadecimal value: B#16#11 Integer value: 17	0
TCP (Receipt of data with specified length)	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#11 Integer value: 17	1 to 8192
ISO on TCP (Message-oriented data transfer)	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#12 Integer value: 18	• 1 to 1452, if a CP is used. • 1 to 8192, if no CP is used.
* See "AUTOHOTSPOT".			

TCP (Ad-hoc mode)

The ad-hoc mode is only available with the TCP protocol variant. You use the ad-hoc mode to receive data with dynamic length with the "TRCV" instruction.

You set ad-hoc mode by assigning the value "0" to the LEN parameter. All data types can be used for data blocks with standard access when you use ad-hoc mode. Only ARRAY of BYTE or data types with a length of 8 bits can be used for data blocks with optimized access (e.g., CHAR, USINT, SINT, etc.). When ad-hoc mode is active, receipt of data is already signaled after a received byte at the NDR parameter.

TCP (Receipt of data with specified length)

For receipt of data with a specified length, enter the length of the data at the LEN parameter. The data receipt is not complete until the length of data specified at the LEN parameter has been completely received. Only then is the data available in the receive area (DATA parameter). The receipt of data is signaled by the NDR output parameter. The actually received data length in bytes at the RCVD_LEN parameter corresponds to the data length at the LEN parameter after receipt.

ISO on TCP (Message-oriented data transfer)

Complete message blocks are sent via a connection with the protocol variant ISO on TCP; these are recognized as such by the recipient. When using ISO on TCP, "TRCV" signals data receipt as soon as the message block has been completely received. The receive area is defined by the LEN and DATA parameters. If the receive buffer (DATA parameter) is too small for the sent data, "TRCV" signals an error. The successful receipt of data is signaled by the NDR output parameter. The actually received data length in bytes at the RCVD_LEN parameter corresponds to the data length at the LEN parameter after receipt.

Parameter

The following table shows the parameters of the "TRCV" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN_R	Input	BOOL	I, Q, M, D, L or constant	Receive enable
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the connection established with "TCON (Page 4474)". Value range: W#16#0001 (1) to W#16#0FFF (4095)
LEN	Input	UINT	I, Q, M, D, L or constant	Length of the receive area in bytes (hidden). If you use a memory area with optimized access at the DATA parameter, the LEN parameter must have the value "0".
DATA	InOut	VARIANT	I, Q, M, D	Pointer to the receive area When transferring structures, the structures must be identical at the sending and receiving end.
NDR	Output	BOOL	I, Q, M, D, L	Status parameter (New Data Received): • 0: Job not yet started or still in progress • 1: Job completed without error
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: No error. • 1: An error occurred during execution of the instruction. Detailed information is output via the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter: Output of status and error information.
RCVD_LEN	Output	UINT	I, Q, M, D, L	Amount of data actually received in bytes

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

LEN, DATA and RCVD_LEN parameters

- If LEN = 0, the received data is saved in the receive area specified at the DATA parameter. The number of bytes received is indicated at the RCVD_LEN parameter.
  With LEN = 0 (default setting of the LEN parameter), the length of the data to be received is defined by the DATA parameter. It is recommended that the receive area (DATA parameter) has the same size as the data to be transmitted by TSEND.
  If LEN has the value 0 and the sent data was transferred in segments that are smaller than the DATA receive area, the following applies. It is recommended to keep EN_R set until the associated TSEND instruction has sent all data. STATUS shows the value 7002 as long as the size of the data sent by TSEND is not equal to the size of the DATA receive area. EN_R must be set until the amount of the received data is equal to the size of the DATA receive area. If you pulse EN_R, you must do so until BUSY=0 or ERROR <> 0.
  The data in the DATA receive area is only valid when BUSY has the value 0.
- If the length specified at the LEN parameter is greater than the length of the data received at the DATA parameter, the error code 8088 is output at the STATUS parameter (see description of the STATUS) parameter in the following.
- If a structure (Struct) is referenced via the DATA parameter, LEN can be shorter than the structure. In this case, only the data up to the length of the LEN parameter is transferred.
- If the DATA parameter references a data block with optimized access, the LEN parameter must be set to "0".
- If a STRING data type is referenced via the DATA parameter, the length specified at the LEN parameter must be 0 or >=2 (LEN = 1 is not permitted).
- If a WSTRING data type is referenced via the DATA parameter, the length specified at the LEN parameter must be 0 or >=5.

BUSY, NDR and ERROR parameters

You can check the status of the job with the BUSY, NDR, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. With the NDR parameter, you can check whether or not a job executed successfully. The ERROR parameter is set when errors occurred during execution of TRCV. The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, NDR and ERROR parameters:

BUSY	NDR	ERROR	Description
1	-	-	The job is being processed.
0	1	0	The job was completed successfully.
0	0	1	The job ended with an error. The cause of the error is output at the STATUS parameter.
0	0	0	No new job was assigned.

Note

Because "TRCV" is executed asynchronously, the data in the receive area is only consistent when the NDR parameter is set to the value "1".

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Explanation
0	0000	Job completed successfully. The current length of the received data is output at the RCVD_LEN parameter.
0	7000	Block not ready to receive.
0	7001	Block is ready to receive, receive job was activated.
0	7002	Interim call, the receive job is executing. Note: While the job is being processed, data is written to the receive area. Access to the receive area during this time may provide inconsistent data.
1	8085	• The LEN parameter is larger than the highest permitted value. • The value of the LEN or DATA parameter was changed after the first call. • Both LEN parameters and the DATA parameter have the value "0" or LEN is longer than the maximum permitted value (65536).
1	8086	The ID parameter is outside the permitted address range (1 .. 0x0FFF).
1	8088	• Receive area is too small. • The value at the LEN parameter is larger than the receive area set at the DATA parameter.
1	80A1	Communication error: • The specified connection has not yet been established. • The specified connection is being terminated. Receive job over this connection is not possible. • The connection is being re-initialized.
1	80B3	The protocol variant (connection_type parameter in the connection description) is set to UDP. Use the instruction "TURCV" with a UDP connection.
1	80C3	• A block with this ID is already being processed in a different priority group. • Internal lack of resources.
1	80C4	Temporary communication error: • The connection cannot be established to the partner at this time. • The interface is receiving new parameter settings or the connection is being established.
1	80C5	The remote partner has terminated the connection.
1	80C6	The remote partner cannot be reached (network error).
1	80C7	Execution timeout.
1	80C9	The length of the receive area is smaller than the length of the sent data.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

See also

Difference between synchronous and asynchronous instructions (Page 2571)

TRCV: Receive data via communication connection (S7-1200, S7-1500)

Description

The following description of the "TRCV" instruction is valid for:

- Ethernet
  CPU S7-1200 with firmware version ≥ V4.0 and CPU S7-1500
- PROFIBUS
  FDL connections of the S7-1500 with CM 1542-5 as of V2.0 with the system data type TCON_FDL

You use the "TRCV" instruction to receive data over an existing communication connection. "TRCV" executes asynchronously.

Receipt of data is enabled when the EN_R parameter is set to the value "1". The received data is entered in a receive area. You specify the length of the receive area either with the LEN parameter (if LEN <> 0) or with the length information of the DATA parameter (if LEN = 0), depending on the protocol variant being used.

While data is being received, you cannot make changes to the DATA parameter or the defined receive area to ensure consistency of the received data.

After successful receipt of data, the NDR parameter is set to the value "1". You can query the amount of data actually received at the RCVD_LEN parameter.

Receive modes of "TRCV"

The following table shows how the received data is entered in the receive area.

Protocol variant	Parameter ADHOC	Availability of data in the receive area	connection_type parameter of the connection description	Parameter LEN
TCP (Ad-hoc mode)	1 (ad-hoc activated)	The data is immediately available.	Hexadecimal value: B#16#11 Integer value: 17	As much data will be read as is currently available; not exceeding the data length specified by LEN. LEN = 0 (recommended): Length of the receive area referenced by DATA
TCP (Receipt of data with specified length)	0 (ad-hoc deactivated)	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#11 Integer value: 17	• S7-1200: 1 to 8192 • S7-1500: 1 to 65536
ISO on TCP (Message-oriented data transfer)	-	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#12 Integer value: 18	• S7-1200: 1 to 8192 • S7-1500: 1 to 65536
FDL	-	The data is available as soon as the data length specified at the LEN parameter has been fully received.	Hexadecimal value: B#16#15 Decimal: 21	1 to 240

TCP (Ad-hoc mode)

The ad-hoc mode is only available with the TCP protocol variant. You use the ad-hoc mode to receive data with dynamic length with the "TRCV" instruction.

You set ad-hoc mode by assigning the value "1" to the ADHOC parameter. When ad-hoc mode is active, receipt of data is already signaled after a received byte at the NDR parameter. All data types can be used for data blocks with standard access when you use ad-hoc mode. Only ARRAY of BYTE or data types with a length of 8 bits can be used for data blocks with optimized access (e.g., CHAR, USINT, SINT, etc.).

TCP (Receipt of data with specified length)

Assign the value "0" to the ADHOC parameter for receipt of data with specified length. If ad-hoc mode is disabled, the data reception is not complete until the length of data specified at the LEN parameter has been completely received. Only then is the data available in the receive area (DATA parameter). The successful receipt of data is signaled by the NDR output parameter. The actually received data length in bytes at the RCVD_LEN parameter corresponds to the data length at the LEN parameter after receipt.

ISO on TCP (Message-oriented data transfer)

Complete message blocks are sent via a connection with the protocol variant ISO on TCP; these are recognized as such by the recipient. When using ISO on TCP, "TRCV" signals data receipt as soon as the message block has been completely received. The receive area is defined by the LEN and DATA parameters. If the receive buffer (DATA parameter) is too small for the sent data, "TRCV" signals an error. The successful receipt of data is signaled by the NDR output parameter. The actually received data length in bytes at the RCVD_LEN parameter corresponds to the data length at the LEN parameter after receipt.

Parameter

The following table shows the parameters of the instruction "TRCV":

Parameter	Declaration	Data type	Memory area	Description
EN_R	Input	BOOL	I, Q, M, D, L or constant	Receive enable
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the connection established with "TCON". Value range: W#16#0001 to W#16#0FFF
LEN	Input	UDINT	I, Q, M, D, L or constant	Length of the receive area in bytes (hidden) Maximum value for S7-1200: 8192; maximum value for S7-1500: 65536; maximum value for CM 1542-5 (FDL): 240. If you use a receive area with optimized access at the DATA parameter, the LEN parameter must have the value "0".
ADHOC	Input	BOOL	I, Q, M, D, L or constant	Use ad-hoc mode for the TCP protocol variant (hidden).

Parameter	Declaration	Data type	Memory area	Description
DATA	InOut	VARIANT	I, Q, M, D, L	Pointer to the receive area When transferring structures, the structures must be identical at the sending and receiving end.
NDR	Output	BOOL	I, Q, M, D, L	Status parameter (New Data Received): • 0: Job not yet started or still in progress • 1: New data received
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or already completed • 1: Job not yet completed. A new job cannot be started
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter ERROR: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction
RCVD_LEN	Output	UDINT	I, Q, M, D, L	Amount of data actually received in bytes

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

LEN, DATA and RCVD_LEN parameters

- If LEN = 0, the received data is saved in the receive area specified at the DATA parameter. The number of bytes received is indicated at the RCVD_LEN parameter.
  With LEN = 0 (default setting of the LEN parameter), the length of the data to be received is defined by the DATA parameter. It is recommended that the receive area (DATA parameter) has the same size as the data to be transmitted by TSEND.
  If LEN has the value 0 and the sent data was transferred in segments that are smaller than the DATA receive area, the following applies. It is recommended to keep EN_R set until the associated TSEND instruction has sent all data. STATUS shows the value 7002 as long as the size of the data sent by TSEND is not equal to the size of the DATA receive area. EN_R must be set until the amount of the received data is equal to the size of the DATA receive area. If you pulse EN_R, you must do so until BUSY=0 or ERROR <> 0.
  The data in the DATA receive area is only valid when BUSY has the value 0.
- If the length specified at the LEN parameter is greater than the length of the data received at the DATA parameter, the error code 8088 is output at the STATUS parameter (see description of the STATUS) parameter in the following.
- If a structure (Struct) is referenced via the DATA parameter, LEN can be shorter than the structure. In this case, only the data up to the length of the LEN parameter is transferred.
- If the DATA parameter references a data block with optimized access, the LEN parameter must be set to "0". If the length of the data does not match for elementary data types, the data will not be received and the error code 8088 is output at the STATUS parameter.
- If a STRING data type is referenced via the DATA parameter, the length specified at the LEN parameter must be 0 or >=2 (LEN = 1 is not permitted).
- If a WSTRING data type is referenced via the DATA parameter, the length specified at the LEN parameter must be 0 or >=5.

BUSY, NDR and ERROR parameters

You can check the status of the job with the BUSY, NDR, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. With the NDR parameter, you can check whether or not a job executed successfully. The ERROR parameter is set when errors occurred during execution of TRCV. The error information is output at the STATUS parameter.

The following table shows the relationship between the BUSY, NDR and ERROR parameters:

BUSY	NDR	ERROR	Description
1	-	-	The job is being processed.
0	1	0	The job was completed successfully.
0	0	1	The job ended with an error. The cause of the error is output at the STATUS parameter.
0	0	0	No new job was assigned.

Note

Because "TRCV" is executed asynchronously, the data in the receive area is only consistent when the NDR parameter is set to the value "1".

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Explanation
0	0000	Job completed successfully. The current length of the received data is output at the RCVD_LEN parameter.
0	7000	Block not ready to receive.
0	7001	Block is ready to receive, receive job was activated.
0	7002	Interim call, the receive job is executing. Note: While the job is being processed, data is written to the receive area. Access to the receive area during this time may provide inconsistent data.
1	8085	• The LEN parameter is greater than the maximum permissible value (for S7-1200: 8192 bytes, for S7-1500: 65536 bytes). • The value of the LEN or DATA parameter was changed after the first call. • Both LEN parameters and the DATA parameter have the value "0" or LEN is longer than the maximum permissible value (for S7-1200): 8192 bytes, for S7-1500: 65536 bytes).
1	8086	The ID parameter is outside the permitted value range (1 .. 0x0FFF).
1	8088	• Receive area is too small. • The value at the LEN parameter is larger than the receive area set at the DATA parameter.
1	80A1	Communication error: • The specified connection has not yet been established. • The specified connection is being terminated. Receive job over this connection is not possible. • The connection is being re-initialized.
1	80B1	You changed the DATA parameter before the current job finished.
1	80B3	The protocol variant (connection_type parameter in the connection description) is set to UDP. Use the instruction "TURCV" with a UDP connection.

ERROR	STATUS* (W#16#...)	Explanation
1	80C3	• A block with this ID is already being processed in a different priority group. • Internal lack of resources.
1	80C4	Temporary communication error: • The connection cannot be established to the partner at this time. • The interface is receiving new parameter settings or the connection is being established.
1	80C5	The remote partner has terminated the connection.
1	80C6	The remote partner cannot be reached (network error).
1	80C7	Execution timeout.
1	80C9	The length of the receive area is smaller than the length of the sent data.
1	80EE	Handshake not yet completed
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Example

You can find the example here: Program example for TCON, TDISCON, TSEND & TRCV (Page 4502).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

TRCV: Receive data via communication connection (Page 4493)

TCON: Establish communication connection (Page 4478)

Difference between synchronous and asynchronous instructions (Page 2571)

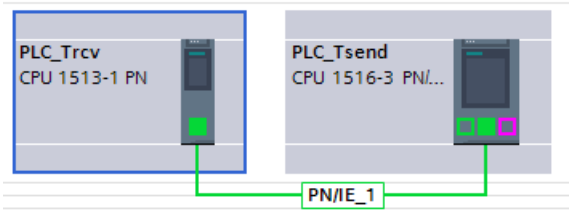
Program example for TCON, TDISCON, TSEND & TRCV (S7-1200, S7-1500)

Introduction

In the following example, you create a programmed connection between two CPUs of the S7-1500 series and send a data record from CPU 1 to CPU 2.

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. Their connection is not yet configured.



- A low protection level under Properties > Protection ensures that read and write access is permitted for each CPU.

Program of CPU 1: Storage of data

For the data record, create the following PLC data type.

TSEND_User

	Name	Data type	Default value
1	prename	String	''
2	lastname	String	''
3	userId	UInt	0

For the data transfer, create the following data block ("SLI_plcDB_sendData_TSEND") based on the created PLC data type.

SLI_plcDB_sendData_TSEND

	Name	Data type	Start value
1	Static		
2	prename	String	'Simon'
3	lastname	String	'Sample'
4	userId	UInt	1

To store the data, create a global data block ("SLI_gDB_TSEND") with the following structures and tags.

SLI_gDB_TSEND

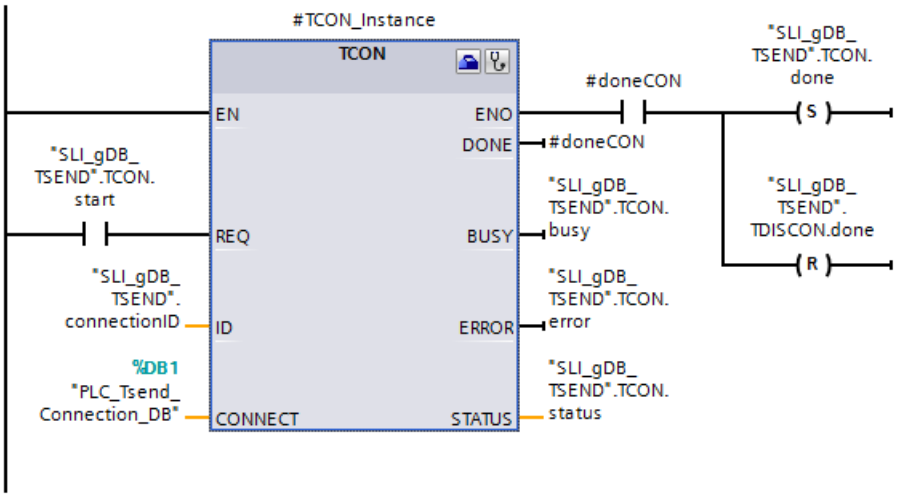
	Name	Data type	Start value
1	Static		
2	connectionID	CONN_OUC	1
3	TCON	Struct	
4	start	Bool	false
5	busy	Bool	false
6	done	Bool	false
7	error	Bool	false
8	status	Word	16#0
9	memErrStatus	Word	16#0
10	TSEND	Struct	
11	start	Bool	false
12	maxLength	UInt	0
13	busy	Bool	false
14	done	Bool	false
15	error	Bool	false
16	status	Word	16#0
17	memErrStatus	Word	16#0
18	TDISCON	Struct	
19	start	Bool	false
20	busy	Bool	false
21	done	Bool	false
22	error	Bool	false
23	status	Word	16#0
24	memErrStatus	Word	16#0

Program of CPU 1: Interconnect parameters

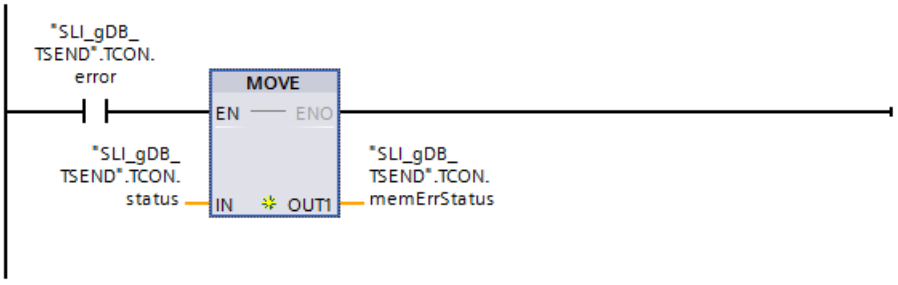
Create the FB "SLI_FB_TSEND" Create the following local tags in it.

7	Static			
8	TCON_Instance	TCON		
9	TSEND_Instance	TSEND		
10	TDISCON_Instance	TDISCON		
11	doneCON	Bool	false	Non-retain
12	doneSEND	Bool	false	Non-retain
13	doneDISC	Bool	false	Non-retain

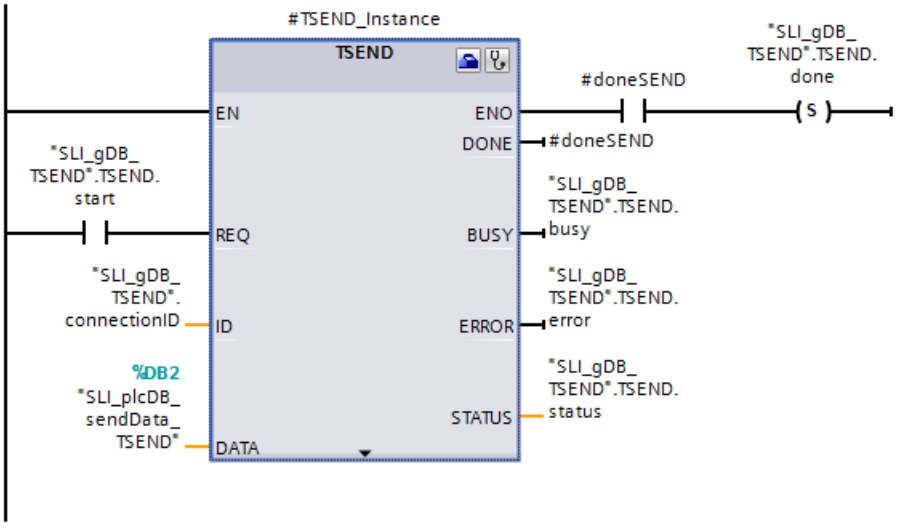
Network 1: Interconnect the parameter of the instruction "TCON" as follows:



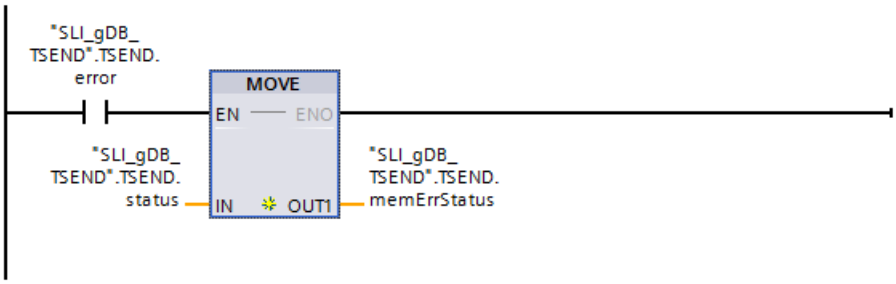
Network 2: In case of error of TCON, save the status as follows.



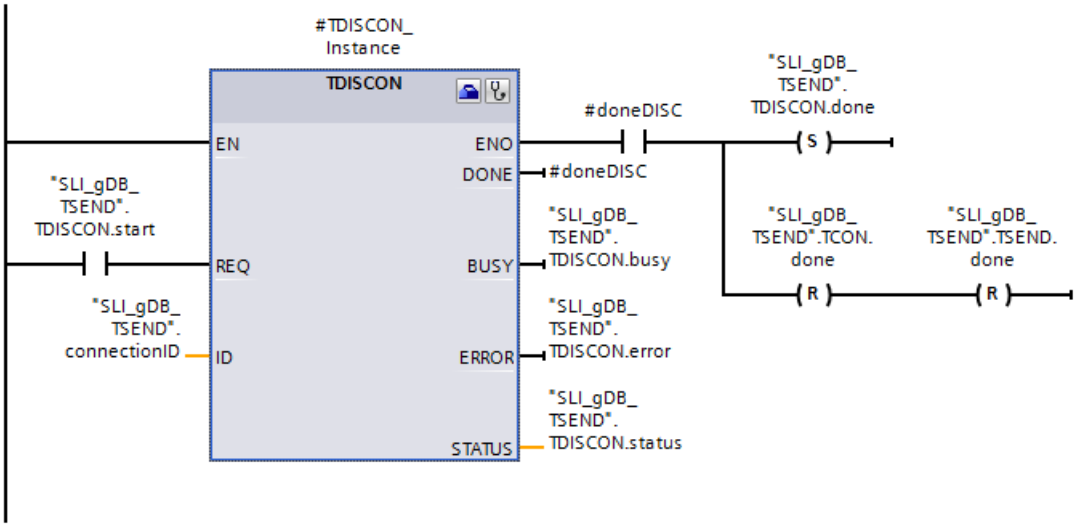
Network 3: Interconnect the parameter of the instruction "TSEND" as follows:



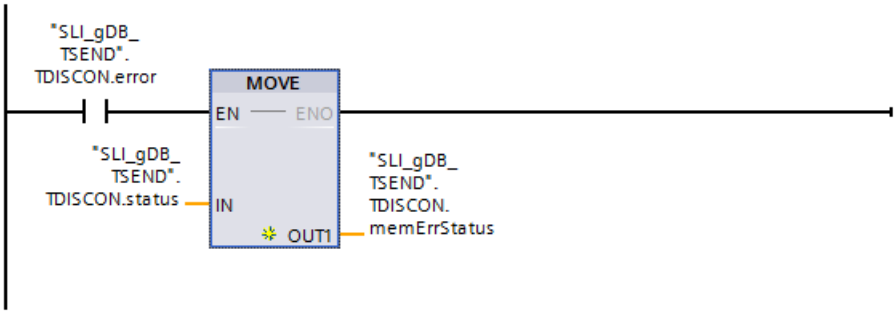
Network 4: In case of error of TSEND, save the status as follows.



Network 5: Interconnect the parameter of the instruction "TDISCON" as follows:



Network 6: In case of error of TDISCON, save the status as follows.

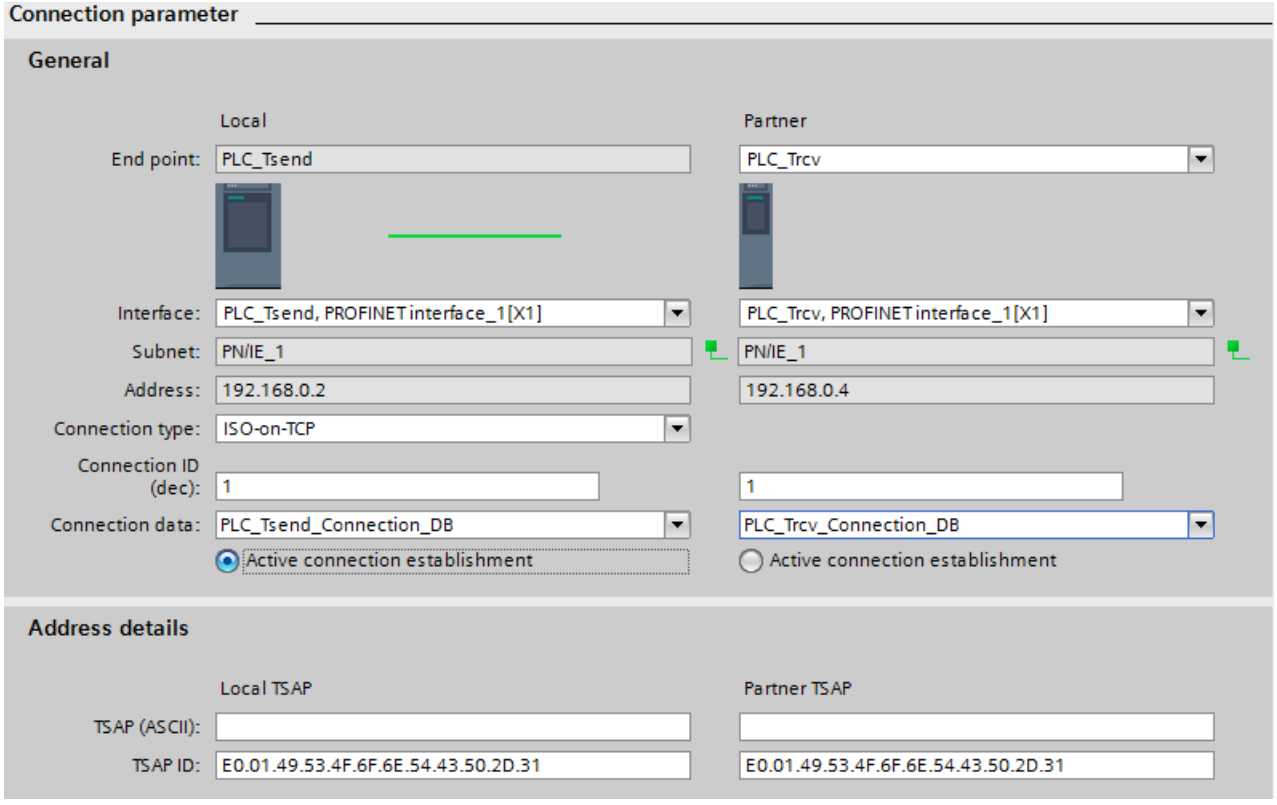


Program of CPU 1: Configuration of TCON

To interconnect the input parameter CONNECT, open the wizard of the "TCON" instruction with its Properties > Configuration.

Make the following settings for the configuration of TCON:

Input field	Entry
End point	Select a CPU for sender and receiver using the drop-down list. Interface, subnet and address are entered automatically.
Connection data	Use the drop-down list and the selection "New" to create one data block per CPU. This data block is required for storing the connection data. The name of the data block can be freely selected. The active connection establishment is enabled for CPU 1 (local CPU).
Connection type	Select the entry "ISO-on-TCP". This means an Ethernet connection with the "ISO-on-TCP" protocol is used for connection setup.
Connection ID	You enter a freely selected connection ID for the communication connection. The connection ID must not already be assigned in the project. Ensure that the value matches the value of the "connectionID" tag that is used.
TSAP ID	The address details are automatically entered during the selection of the connection type "ISO-on-TCP".



Program of CPU 2: Storage of data

For the data record, create the following PLC data type.

TSEND_User

	Name	Data type	Default value
1	prename	String	''
2	lastname	String	''
3	userId	UInt	0

For the data transfer, create the following data block ("SLI_plcDB_rcvData_TRCV") based on the created PLC data type.

SLI_plcDB_rcvData_TRCV

	Name	Data type	Start value
1	Static		
2	prename	String	''
3	lastname	String	''
4	userId	UInt	0

To store the data, create a global data block ("SLI_gDB_TRCV") with the following structures and tags.

SLI_gDB_TRCV

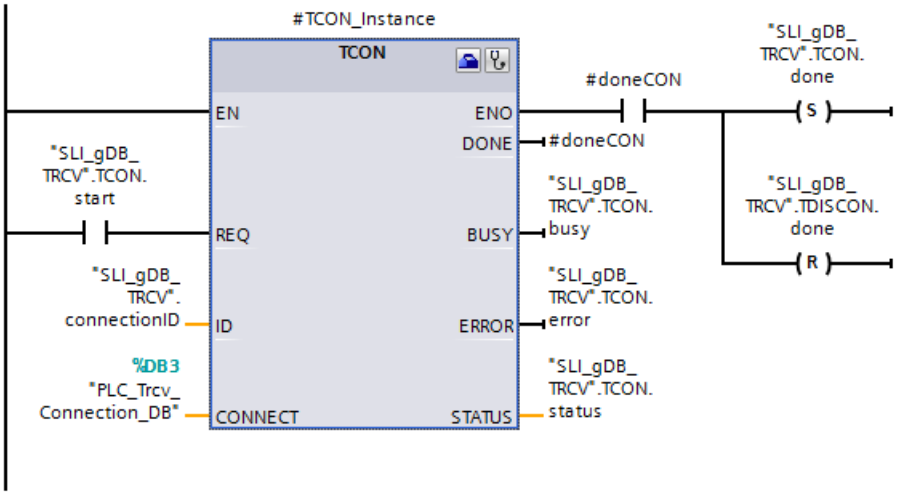
	Name	Data type	Start value
1	Static		
2	connectionID	CONN_OUC	1
3	TCON	Struct	
4	start	Bool	false
5	busy	Bool	false
6	done	Bool	false
7	error	Bool	false
8	status	Word	16#0
9	memErrStatus	Word	16#0
10	TRCV	Struct	
11	start	Bool	false
12	busy	Bool	false
13	done	Bool	false
14	error	Bool	false
15	status	Word	16#0
16	memErrStatus	Word	16#0
17	readLength	UInt	0
18	TDISCON	Struct	
19	start	Bool	false
20	busy	Bool	false
21	done	Bool	false
22	error	Bool	false
23	status	Word	16#0
24	memErrStatus	Word	16#0

Program of CPU 2: Interconnect parameters

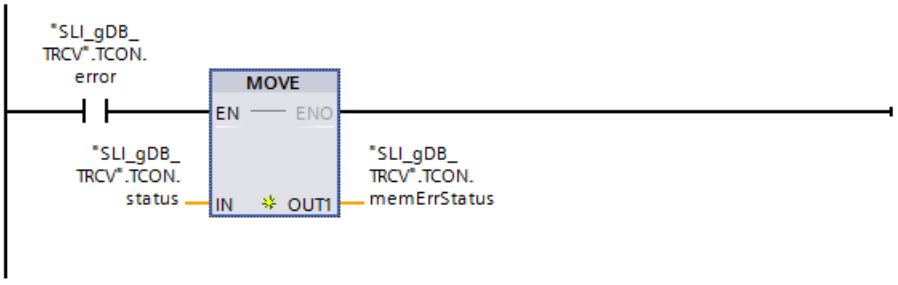
Create the FB "SLI_FB_TRCV" Create the following local tags in it.

7		Static			
8		doneCON	Bool	false	Non-ret...
9		doneRCV	Bool	false	Non-retain
10		doneDISC	Bool	false	Non-retain
11		length	UInt	0	Non-retain
12		TCON_Instance	TCON		
13		TRCV_Instance	TRCV		
14		TDISCON_Instance	TDISCON		

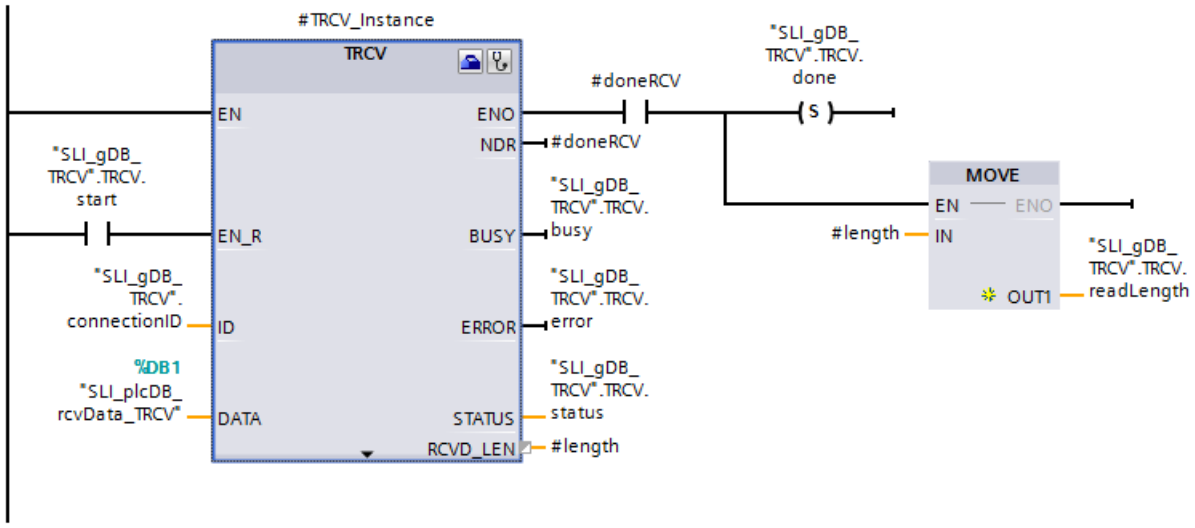
Network 1: Interconnect the parameter of the instruction "TCON" as follows:



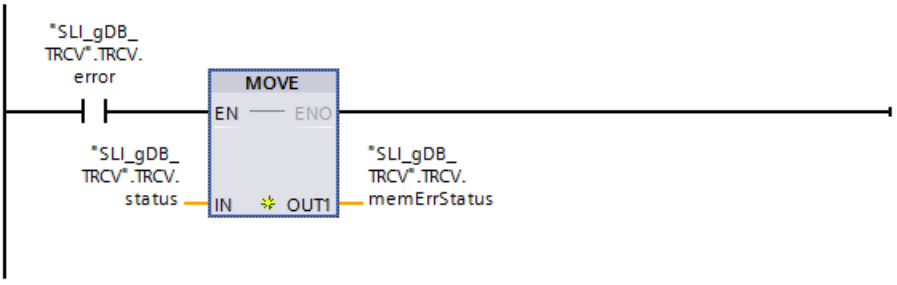
Network 2: In case of error of TCON, save the status as follows.



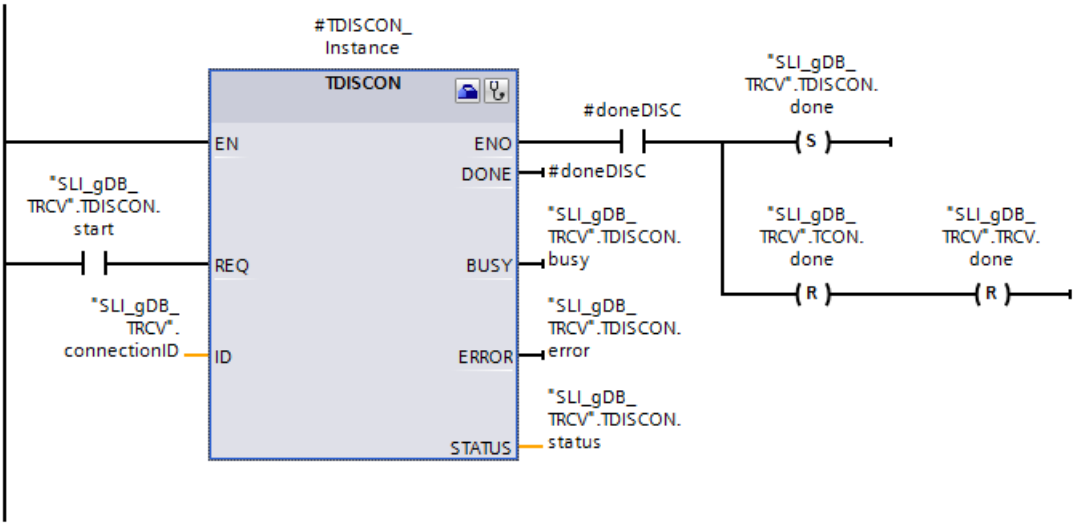
Network 3: Interconnect the parameter of the instruction "TRCV" as follows:



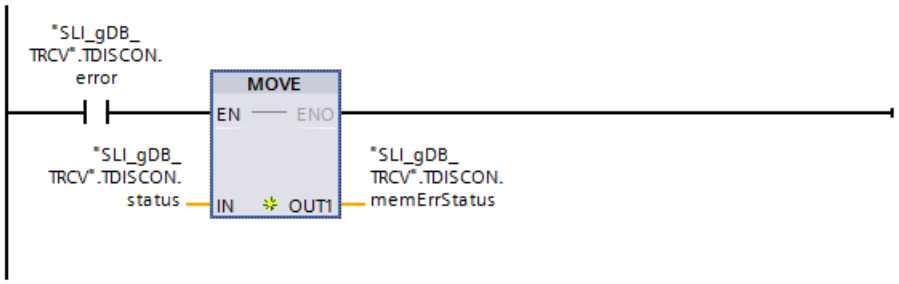
Network 4: In case of error of TRCV, save the status as follows.



Network 5: Interconnect the parameter of the instruction "TDISCON" as follows:



Network 6: In case of error of TDISCON, save the status as follows.



Program of CPU 2: Configuration of TCON

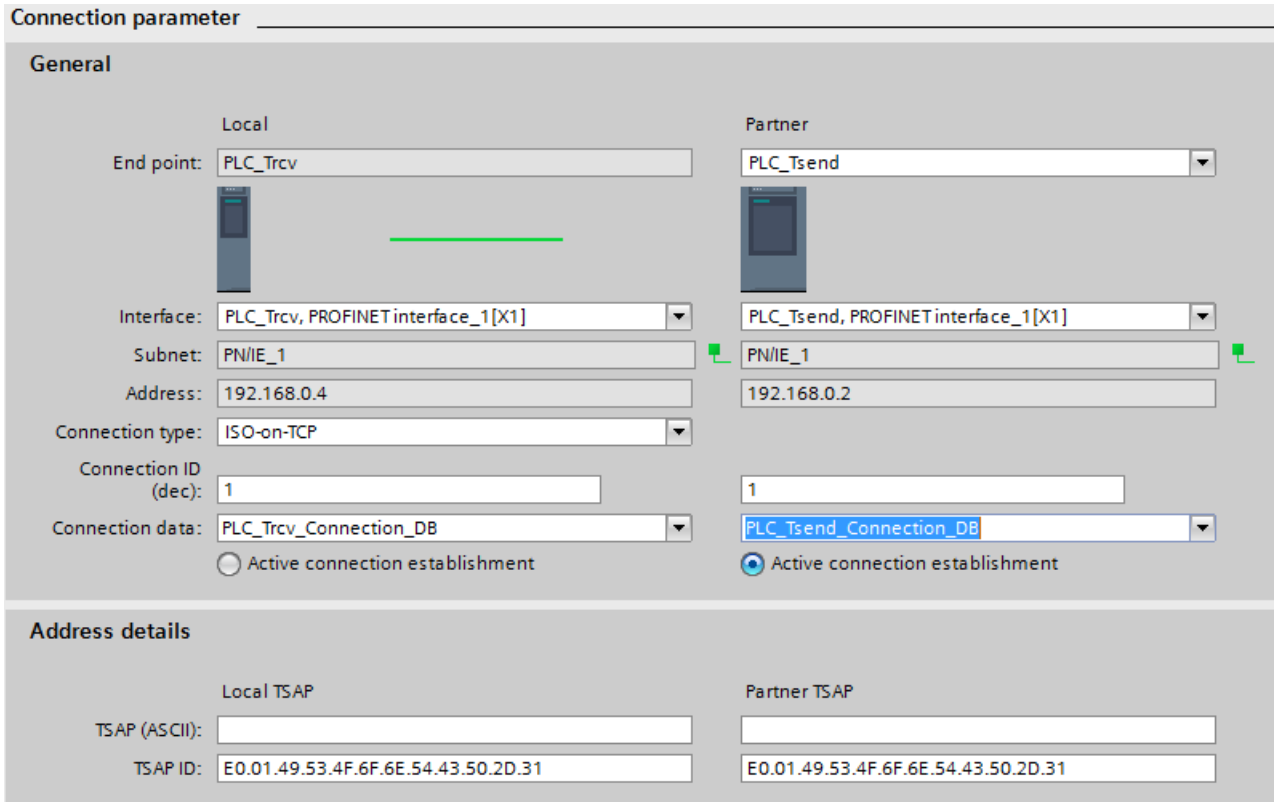
You make the following settings for the connection of CPU 2 / TCON according to the example of CPU 1 / TCON:

Note

Connection data

By configuring CPU 1 / TCON, you have already created a data block ("PLC_Tsend_Connection_DB", "PLC_Trcv_Connection_DB") with the stored connection data for each CPU. Instead of creating new data blocks and entering the connection data, you can simply use the created data blocks ("PLC_Tsend_Connection_DB", "PLC_Trcv_Connection_DB").

For the created data blocks to be available for selection, they must still be available in the project tree under Program blocks > System blocks.



Procedure for establishing connection

Observe the order of the action steps:

1. Change the parameter REQ to "TRUE" for CPU 2 / TCON.
2. Change the parameter REQ to "TRUE" for CPU 1 / TCON.
3. Change the parameter REQ to "FALSE" for CPU 2 / TCON.
4. Change the parameter REQ to "FALSE" for CPU 1 / TCON.

Procedure for terminating connection

Observe the order of the action steps:

1. Change the parameter REQ to "TRUE" for CPU 1 / TDISCON.
2. Change the parameter REQ to "TRUE" for CPU 2 / TDISCON.
3. Change the parameter REQ to "FALSE" for CPU 1 / TDISCON.
4. Change the parameter REQ to "FALSE" for CPU 2 / TDISCON.

Behavior of CPU 1

Network 1 (TCON):

When the input parameter REQ ("TCON.start") supplies the signal state "TRUE", the "TCON" instruction is started. For multiple calls, the "TCON" instruction creates a communication connection to the partner CPU. To do this, the connection data is retrieved with the input parameter CONNECT (data block "PLC_Trcv_Connection_DB").

Successful connection setup is indicated by the output parameter DONE ("#doneCON") with the signal state "TRUE" and the output parameter STATUS ("TCON.status") with the value "0000". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "TCON.done". Reset the success status of any previous connection termination ("TDISCON.done").

The output parameter ERROR ("TCON.error") or the tag "TCON.memErrStatus" indicates that processing in the example is taking place without errors.

3		TCON	Struct		
4		start	Bool	false	TRUE
5		busy	Bool	false	FALSE
6		done	Bool	false	TRUE
7		error	Bool	false	FALSE
8		status	Word	16#0	16#7000
9		memErrStatus	Word	16#0	16#0000

Network 2 (TCON):

When TCON signals an error ("TCON.error" is "TRUE"), the reported status ("TCON.status") is stored permanently ("TCON.memErrStatus").

Network 3 (TSEND):

Based on the input parameter ID ("connectionID"), the "TSEND" instruction knows the communication connection to be used.

When the input parameter REQ ("TSEND.start") supplies the signal state "TRUE", the "TSEND" instruction is started. For multiple calls, the "TSEND" instruction transmits the data record detected at input parameter DATA ("SLI_plcDB_sendData_TSEND").

Successful transfer of the data record is indicated by the output parameter DONE ("#doneSEND") with the signal state "TRUE" and the output parameter STATUS ("TSEND.status") with the value "0000". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "TSEND.done".

The output parameter ERROR ("TSEND.error") or the tag "TSEND.memErrStatus" indicates that processing in the example is taking place without errors.

10		TSEND	Struct		
11		start	Bool	false	TRUE
12		maxLength	UInt	0	0
13		busy	Bool	false	FALSE
14		done	Bool	false	TRUE
15		error	Bool	false	FALSE
16		status	Word	16#0	16#7000
17		memErrStatus	Word	16#0	16#0000

Network 4 (TSEND):

When TSEND signals an error ("TSEND.error" is "TRUE"), the reported status ("TSEND.status") is stored permanently ("TSEND.memErrStatus").

Network 5 (TDISCON):

When the input parameter REQ ("TDISCON.start") supplies the signal state "TRUE", the "TDISCON" instruction is started. For multiple calls, the instruction "TDISCON" terminates the communication connection. The communication connection used at the input parameter ID ("connectionID") is retrieved for this purpose.

Successful connection termination is indicated by the output parameter DONE ("#doneDISC") with the signal state "TRUE" and the output parameter STATUS ("TDISCON.status") with the value "0000". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "TDISCON.done". Reset the success messages for connection setup ("TCON.done") and for data transfer ("TSEND.done").

The output parameter ERROR ("TDISCON.error") or the tag "TDISCON.memErrStatus" indicates that processing in the example is taking place without errors.

18		TDISCON	Struct		
19		start	Bool	false	TRUE
20		busy	Bool	false	FALSE
21		done	Bool	false	TRUE
22		error	Bool	false	FALSE
23		status	Word	16#0	16#7000
24		memErrStatus	Word	16#0	16#0000

Network 6 (TDISCON):

When TDISCON signals an error ("TDISCON.error" is "TRUE"), the reported status ("TDISCON.status") is stored permanently ("TDISCON.memErrStatus").

Behavior of CPU 2

For TCON and TDISCON, the behavior is the same as that for CPU 1.

3		TCON	Struct		
4		start	Bool	false	TRUE
5		busy	Bool	false	FALSE
6		done	Bool	false	TRUE
7		error	Bool	false	FALSE
8		status	Word	16#0	16#7000
9		memErrStatus	Word	16#0	16#0000

18		TDISCON	Struct		
19		start	Bool	false	TRUE
20		busy	Bool	false	FALSE
21		done	Bool	false	TRUE
22		error	Bool	false	FALSE
23		status	Word	16#0	16#7000
24		memErrStatus	Word	16#0	16#0000

Network 3 (TRCV):

Based on the input parameter ID ("connectionID"), the "TRCV" instruction knows the communication connection to be used.

When the input parameter EN_R ("TRCV.start") supplies the signal state "TRUE", the "TRCV" instruction is started. For multiple calls, the "TRCV" instruction receives the transmitted data record. The data record is detected at the input parameter DATA ("SLI_plcDB_rcvData_TRCV").

SLI_plcDB_rcvData_TRCV

	Name	Data type	Start value	Monitor value
1	Static			
2	prename	String	''	'Simon'
3	lastname	String	''	'Sample'
4	userId	UInt	0	1

The length in BYTE of the actually transmitted data record is detected with the output parameter LEN ("#length"). This value is only displayed during the success status. Then "0" is detected. Successful receipt of the data record is indicated by the output parameter DONE ("#doneRCV") with the signal state "TRUE" and the output parameter STATUS ("TRCV.status") with the value "0000".

Because the values of the output parameters are only displayed for the moment of their validity, do the following:

- Save the success status in the tag "TRCV.done"
- Save the length in BYTE in the tag "TRCV.readLength".

The output parameter ERROR ("TRCV.error") or the tag "TRCV.memErrStatus" indicates that processing in the example is taking place without errors.

10	TRCV	Struct		
11	start	Bool	false	TRUE
12	busy	Bool	false	TRUE
13	done	Bool	false	TRUE
14	error	Bool	false	FALSE
15	status	Word	16#0	16#7002
16	memErrStatus	Word	16#0	16#0000
17	readLength	UInt	0	514

Network 4 (TRCV):

When TRCV signals an error ("TRCV.error" is "TRUE"), the reported status ("TRCV.status") is stored permanently ("TRCV.memErrStatus").

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

TCON: Establish communication connection (Page 4478)

TDISCON: Terminate communication connection (Page 4484)

TSEND: Send data via communication connection (Page 4490)

TRCV: Receive data via communication connection (Page 4498)

Sending and receiving data via Ethernet (UDP) or FDL (S7-1200, S7-1500)

TUSEND: Sending data (S7-1200, S7-1500)

Description

TUSEND supports data transfer via UDP (for S7-1500-CPUs as of firmware version V2.0 and higher via the integrated PROFINET interfaces also as multicast communication) and via FDL connections of the S7-1500 with CM 1542-5 as of V2.0 with the system data type TCON_FDL.

The "TUSEND" instruction sends data to the communication partner addressed by the ADDR parameter.

WARNING

Data transfer via UDP

When transferring data with UDP according to RFC 768, the data is transferred to the partner without acknowledgement and therefore non-securely. This means that data can be lost without this being indicated by the block.

Note

For sequential send operations to different partners, you only need to adjust the ADDR parameter when calling "TUSEND". You do not need to call the "TCON (Page 4474)" and "TDISCON (Page 4484)" instructions again. The UDP port must be set up at the specific partner in order for this partner to receive data.

Functional description

The TUSEND" instruction works asynchronously, which means its job processing extends over multiple calls. Create a rising edge at the REQ parameter to establish a connection once again.

The output parameters BUSY, DONE, ERROR and STATUS indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571).

The following table shows the relationship between BUSY, DONE and ERROR. Using this table, you can determine the current status of "TUSEND" or when the send process is concluded.

BUSY	DONE	ERROR	Description
TRUE	FALSE	FALSE	The job is being processed.
FALSE	TRUE	FALSE	Job successfully completed.
FALSE	FALSE	TRUE	The job ended with an error. The cause of the error can be found in the STATUS parameter.
FALSE	FALSE	FALSE	The instruction was not assigned a (new) job.

Note

Due to the asynchronous operation of "TUSEND", make sure the data in the send area remains consistent until the DONE parameter or the ERROR parameter has the value TRUE.

Parameter

The following table shows the parameters of the instruction "TUSEND":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Control parameter REQUEST starts the send job on a rising edge. The data is transferred from the area specified by DATA and LEN.
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the associated connection between the user program and the communication layer of the operating system. ID must be identical to the associated parameter ID in the local connection description with the .TCON instruction: Value range: W#16#0001 to W#16#0FFF
LEN	Input	UDINT	I, Q, M, D, L or constant	Number of bytes to be sent with the job Value range for Ethernet/UDP: 1 to 1472 For FDL connections of the CM 1542-5, the maximum length is 240 bytes. In this regard, note the maximum lengths that can be processed by the connection partner.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter DONE: • 0: Job not yet started or still executing. • 1: Job executed without errors. This value is only displayed for one cycle.
BUSY	Output	BOOL	I, Q, M, D, L	• BUSY = 1: The job is not yet complete. A new job cannot be triggered. • BUSY = 0: Job is complete.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter ERROR: • ERROR = 1: An error occurred during processing. STATUS supplies detailed information on the type of error

Parameter	Declaration	Data type	Memory area	Description
STATUS	Output	WORD	I, Q, M, D, L	Status parameter STATUS: Error information
DATA	InOut	VARIANT	I, Q, M, D	Send area, contains address and length The address refers to: • The process image of the inputs • The process image of the outputs • A bit memory • A data block When transferring structures, the structures must be identical at the sending and receiving end.
ADDR	InOut	VARIANT	D	Address information of the communication partner See also: Addressing the remote partner via TADDR_Param (Page 4523) and Addressing the remote partner via TADDR_SEND_QDN and TADDR_RCV_IP (Page 4525) and AUTOHOTSPOT

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Explanation
0	0000	Send job completed without error
0	7000	No job processing active
0	7001	Start of job processing, data being sent Note: During this processing phase, the operating system accesses the data in the DATA send area.
0	7002	Intermediate call (REQ irrelevant), job is being processed Note: During this processing phase, the operating system accesses the data in the DATA send area.
1	8085	The LEN parameter has the value "0" or is greater than the highest permitted value.
1	8086	The ID parameter is not in the permitted value range.
0	8088	The LEN parameter is greater than the memory area specified in DATA.
1	8089	The parameter ADDR does not point to a data block with the structure TADDR_Param or TADDR_SEND_QDN.
1	80A1	Communication error: • The specified connection between user program and communication layer of the operating system has not yet been established. • The specified connection between the user program and the communication layer of the operating system is currently being terminated. Transmission over this connection is not possible. • The interface is being reinitialized.
1	80B1	You changed the DATA parameter before the current job finished.

ERROR	STATUS* (W#16#...)	Explanation
1	80A4	IP address (at the ADDR parameter) of the remote connection end point is invalid; it may correspond to the local partner's own IP address.
1	80B3	• The protocol variant (connection_type parameter in the connection description) is not set to UDP. Please use "TSEND (Page 4490)". • Parameter ADDR: Invalid information for port no.
1	80B7	The length of the structure referenced by the parameter ADDR is not 8 bytes.
1	80C3	• A block with this ID is already being processed in a different priority class. • Internal lack of resources.
1	80C4	Temporary communication error: • The connection between the user program and the communication layer of the operating system cannot be established at this time. • New parameter settings are being assigned to the interface. • Temporarily no receive resources available at the connection partner. The connection partner is not ready to receive.
1	80C5	• Connection terminated by the communication partner. • LSAP of the remote connection partner is not released
1	80C6	Network error: • Remote partner cannot be reached. • Physical interruption on PROFIBUS
1	80C7	Execution timeout.
1	80D3	The fully qualified domain name could not be resolved. Possible causes: • The DNS server is not reachable, for example, because it has been shut down or the remote port is not reachable. • An error occurred during communication with the DNS server. • The DNS server returned a valid DNS answer, but the answer contained no IPv4 address.
1	80EA	DTLS (UDP) protocol is not supported
-	General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Example

You can find the example here: Program example for TUSEND & TURCV (Page 4526).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

TURCV: Receiving data (S7-1200, S7-1500)

Description

The "TURCV" instruction receives data via UDP (for S7-1500-CPUs as of firmware version V2.0 and higher via the integrated PROFINET interfaces also as multicast communication) and via FDL connections of the S7-1500 with CM 1542-5 as of V2.0. After successful completion of "TURCV", the ADDR parameter will show you the address of the remote partner (the sender).

⚠ WARNING

Unsecured data transfer

When transferring data with UDP according to RFC 768, the data is transferred to the remote partner without acknowledgement and is therefore unreliable. This means that data can be lost without this being indicated by the block.

Functional description

"TURCV" is an instruction that works asynchronously, which means its job processing extends over multiple calls. You start the receive job by calling "TURCV" with EN_R = 1.

The output parameters BUSY, DONE, ERROR and STATUS indicate the status of the job.

See also: Difference between synchronous and asynchronous instructions (Page 2571).

The following table shows the relationship between BUSY, NDR and ERROR. Using this table, you can determine the current status of TURCV or when the receive process is concluded.

BUSY	NDR	ERROR	Description
TRUE	FALSE	FALSE	The job is being processed.
FALSE	TRUE	FALSE	Job successfully completed. New data was received.
FALSE	FALSE	TRUE	The job ended with an error. The cause of the error can be found in the STATUS parameter.
FALSE	FALSE	FALSE	The instruction was not assigned a (new) job.

Note

Due to the asynchronous operation of "TURCV", the data in the receive area is only consistent when the NDR parameter has the value TRUE.

Parameter

The following table shows the parameters of the instruction "TURCV":

Parameter	Declaration	Data type	Memory area	Description
EN_R	Input	BOOL	I, Q, M, D, L or constant	Control parameter enabled to receive When EN_R = 1, "TURCV" is ready to receive. The receive job is being processed.
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the associated connection between the user program and the communication layer of the operating system. ID must be identical to the associated ID parameter in the local connection description. Value range: W#16#0001 to W#16#0FFF
LEN	Input	UDINT	I, Q, M, D, L or constant	Length of the receive area in bytes: 0 (recommended) or 1 to 1472 Maximum value for CM 1542-5 (FDL): 240
NDR	Output	BOOL	I, Q, M, D, L	Status parameter NDR: • NDR = 0: Job not yet started or still running. • NDR = 1: Job successfully completed
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter ERROR: • ERROR=1: An error occurred during processing. STATUS supplies detailed information on the type of error
BUSY	Output	BOOL	I, Q, M, D, L	• BUSY = 1: The job is not yet complete. A new job cannot be triggered. • BUSY = 0: Job is complete.
STATUS	Output	WORD	I, Q, M, D, L	Status parameter STATUS: Error information
RCVD_LEN	Output	UDINT	I, Q, M, D, L	Amount of data actually received in bytes
DATA	InOut	VARIANT	I, Q, M, D, L	Receive area The address references: • The process image of the inputs • The process image of the outputs • A bit memory • A data block When transferring structures, the structures must be identical at the sending and receiving end.
ADDR	InOut	VARIANT	D	Address information of the remote partner See also: Addressing the remote partner via TADDR_Param (Page 4523) and Addressing the remote partner via TADDR_SEND_QDN and TADDR_RCV_IP (Page 4525) and AUTOHOTSPOT

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Explanation
0	0000	New data was accepted. The current length of the received data is shown in RCVD_LEN.
0	7000	Block not ready to receive
0	7001	Block is ready to receive, receive job was activated
0	7002	Intermediate call, receive job being processed Note: During this processing phase, "TURCV" writes data to the receive area. For this reason, an error could result in inconsistent data in the receive area.
1	8085	The LEN parameter is greater than the largest permitted value, or you changed the value of the LEN or DATA parameter since the first call
1	8086	The ID parameter is not in the permitted value range
1	8088	• Receive area is too small • Value in LEN is higher than the receive area specified by DATA
1	8089	The parameter ADDR does not point to a data block with the structure TADDR_Param or TADDR_RCV_IP.
1	80A1	Communication error: • The specified connection between user program and communication layer of the operating system has not yet been established. • The specified connection between the user program and the communication layer of the operating system is currently being terminated. A receive job over this connection is not possible. • New parameter settings are being assigned to the interface.
1	80B1	You changed the DATA parameter before the current job finished.
1	80B3	The protocol variant (connection_type parameter in the connection description) is not set to UDP. Please use "TRCV (Page 4493)".
1	80B7	The length at ADDR parameter does not correspond to 8 bytes.
1	80C3	• A block with this ID is already being processed in a different priority class. • Internal lack of resources.
1	80C4	Temporary communication error: New parameter settings are being assigned to the interface.
1	80C5	The remote partner has terminated the connection.
1	80C7	Execution timeout.
1	80C9	With RFC1006 / UDP: The received data is longer than expected (size of receive buffer exceeded).
1	80EA	DTLS (UDP) protocol is not supported
-	General error information	See also: GET_ERR_ID: Get error ID locally (Page 2840)
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Example

You can find the example here: Program example for TUSEND & TURCV (Page 4526).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

TCON: Establish communication connection (Page 4474)

TDISCON: Terminate communication connection (Page 4484)

Addressing the remote partner via TADDR_Param (S7-1200, S7-1500)

Overview

When using a UDP connection, you can store the address information for the remote partner in the system data type TADDR_Param:

- With the "TUSEND (Page 4516)" instruction, you transfer the address information of the recipient at the parameter ADDR via TADDR_Param.
  The stored address data of the remote partner are read from the system data type by the instruction.

  Note

  Supporting a limited broadcast

  For S7-1500 CPUs as of firmware version V2.0, the following applies to the instruction "TUSEND": When addressing the remote partner via the "TADDR_Param" system data type, a limited broadcast (255.255.255.255 as the IP address of the remote partner) is supported with consistent compliance to IP routing rules. Accordingly, limited broadcasts are only sent over an interface through which a default router is accessible according to configuration.

  It is recommended to use of directed broadcast addresses because they - according to the IP standards - are not subject to this restriction.

- With the "TURCV (Page 4520)" instruction, you receive the address of the sender at the parameter ADDR via TADDR_Param.
  The address data is written by the instruction in the system data type.

Structure of the address information according to TADDR_Param

The system data type TADDR_Param contains the address information of the remote partner and consists of the IP address and the port number.

The system data type TADDR_Param has the following structure:

Byte	Parameter	Data type	Start value	Description
0 to 3	rem_ip_addr	ARRAY [1..4] of USINT	B#16#00 ...	• IP address of the remote partner, e.g. 192.168.002.003: – rem_ip_addr[1] = B#16#C0 (192) – rem_ip_addr[2] = B#16#A8 (168) – rem_ip_addr[3] = B#16#02 (002) – rem_ip_addr[4] = B#16#03 (003) The IP address can be taken from the Devices & networks view in the interface properties of the remote partner. As an alternative these are also displayed in the properties of the UDP connection under Address details. • Multicast address of an IPv4 multicast group (for the "TUSEND" instruction with S7-1500-CPUs as of firmware version V2.0 and higher. Note: For the "TURCV" instruction, the IP address of the sender must also be specified in case of multicast communication.)
4 to 5	rem_port_nr	UINT	B#16#00 ...	Remote port-number (for possible values, see: AUTOHOT-SPOT): • rem_port_nr[1] = high byte of the port no. in hexadecimal notation • rem_port_nr[2] = low byte of port no. in hexadecimal notation The port number can be taken from the Devices & networks view in the UDP connection properties. The port number is indicated under Address details as a decimal value. Example: Port number = 2000 (decimal) / W#16#07D0 (hexadecimal) • rem_port_nr[1] = 07 (high byte) • rem_port_nr[2] = D0 (low byte)
6 to 7	reserved	WORD	B#16#00 ...	Not used. Leave the value "0" at this parameter.

Creating TADDR_Param in a data block

You have the following options for creating TADDR_Param:

- Create a new data block and select TADDR_Param as type in the dialog "Add new data block".
- Open an existing data block create a new tag and enter in the column Data type TADDR_Param.

A data block can contain several system data types TADDR_Param.

Addressing the remote partner via TADDR_SEND_QDN and TADDR_RCV_IP (S7-1500)

Overview

For S7-1500 CPUs as of firmware version V2.0, you can address the recipient with its fully qualified domain name when sending data via UDP. With the instruction TUSEND at the parameter ADDR, you hereby reference a structure of the type TADDR_SEND_QDN.

The receiver can return an IPv4 or an IPv6 address. With the TURCV instruction at the ADDR parameter, you therefore reference a structure of the type TADDR_RCV_IP. Only this can include both IP address types.

Structure of the address information according to TADDR_SEND_QDN

The system data type TADDR_SEND_QDN contains the address information of the remote partner (of the receiver) and consists of the fully qualified domain name and port number.

It has the following structure:

Byte	Parameter	Data type	Start value	Description
0 to 255	RemoteQDN	ARRAY of STRING[1..254]	-	Fully qualified domain name of the partner end point, which must finish with "." Please note that the name including the concluding period must not exceed 254 characters in a SIMATIC environment.
256 to 257	RemotePort	UINT	B#16#00 ...	Remote port-number (for possible values, see: AUTOHOTSPOT): • RemotePort[1] = high byte of the port no. in hexadecimal notation • RemotePort[2] = low byte of port no. in hexadecimal notation Example: Port number = 2000 (decimal) / W#16#07D0 (hexadecimal) • RemotePort[1] = 07 (high byte) • RemotePort[2] = D0 (low byte)
258 to 259	reserved	WORD	W#16#00	Not used. Leave the value "0" at this parameter.

Note

Runtime of the instruction TUSEND

With each call of TUSEND, the fully qualified domain name is resolved in the IP address. As a result, the execution time of TUSEND is longer than if you address the receiver directly via its IPv4 address.

Structure of the address information according to TADDR_RCV_IP

The system data type TADDR_RCV_IP contains the address information of the remote partner (of the sender) and consists of the IP address, the port number and the information on whether the IP address is an IPv4 or IPv6 address.

It has the following structure:

Byte	Parameter	Data type	Start value	Description
0 to 15	RemoteAd-dress	ARRAY [1..16] of BYTE	B#16#00 ...	IP address of the remote partner, e.g. for the IPv4 address 192.168.002.003: • RemoteAddress[1] = B#16#C0 (192) • RemoteAddress[2] = B#16#A8 (168) • RemoteAddress[3] = B#16#02 (002) • RemoteAddress[4] = B#16#03 (003) The following applies to the IPv6 address.
16 to 17	RemotePort	UINT	B#16#00 ...	Remote port-number (for possible values, see: AUTOHOT-SPOT): • RemotePort[1] = high byte of the port no. in hexadecimal notation • RemotePort[2] = low byte of port no. in hexadecimal notation Example: Port number = 2000 (decimal) / W#16#07D0 (hex-adecimal) • RemotePort[1] = 07 (high byte) • RemotePort[2] = D0 (low byte)
18	RemAddr-Type	BYTE	B#16#00	Type of the remote address: • B#16#03: IPv4 • B#16#04: IPv6
19	reserved	BYTE	B#16#00	Not used. Leave the value "0" at this parameter.

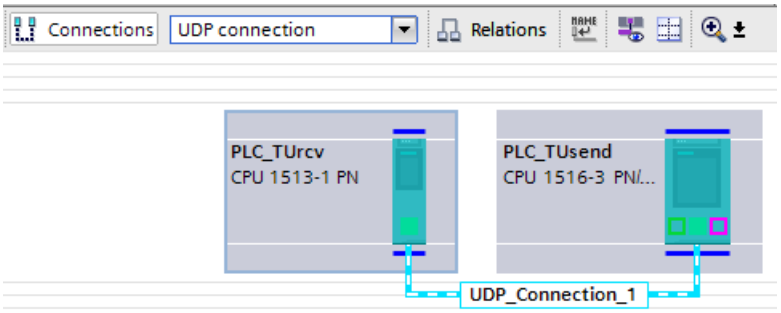
Program example for TUSEND & TURCV (S7-1200, S7-1500)

Introduction

In the following example you create a configured UDP connection between two CPUs of the S7-1500 series. Transmit a data record from CPU1 to CPU2 using the instructions "TUSEND" and "TURCV".

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. A UDP connection is configured.



- A low protection level under "<CPU> > Properties > Protection" ensures that read and write access is permitted for the CPUs.

Program of CPU 1

For the data record, create the following PLC data type "TUSEND_User".

TUSEND_User

	Name	Data type	Default value
1	prename	String	''
2	lastname	String	''
3	userId	UInt	0

For the data transfer, create the following data block ("SLI_plcDB_sendData_TUSEND") based on the PLC data type "TUSEND_User".

SLI_plcDB_sendData_TUSEND

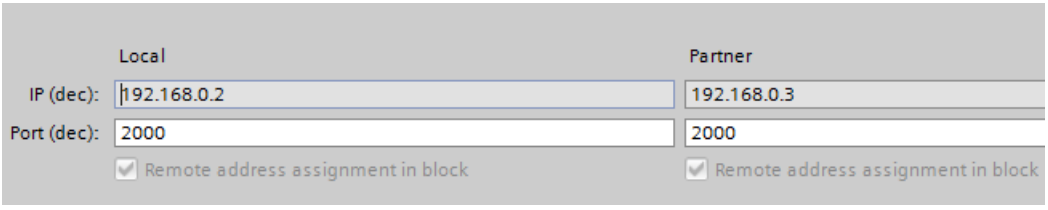
	Name	Data type	Start value
1	Static		
2	prename	String	'Simon'
3	lastname	String	'Sample'
4	userId	UInt	1

For addressing the communication partner, create the following data block ("SLI_plcDB_taddr_param_TUSEND") based on the system data type "TADDR_Param.

SLI_plcDB_taddr_param_TUSEND

	Name	Data type	Start value
1	Static		
2	REM_IP_ADDR	Array[1..4] of USInt	
3	REM_IP_ADDR[1]	USInt	192
4	REM_IP_ADDR[2]	USInt	168
5	REM_IP_ADDR[3]	USInt	0
6	REM_IP_ADDR[4]	USInt	3
7	REM_PORT_NR	UInt	2000
8	RESERVED	Word	16#0

You can find the address date in the "Network view" under "Connections > Properties > Address details". The data of the CPU that had been selected previously are specified in the address details under "Local". In the following figure, CPU 1 "TUsend") is selected.



To store the data, create a global data block ("SLI_gDB_TUSEND") with the following structures and tags.

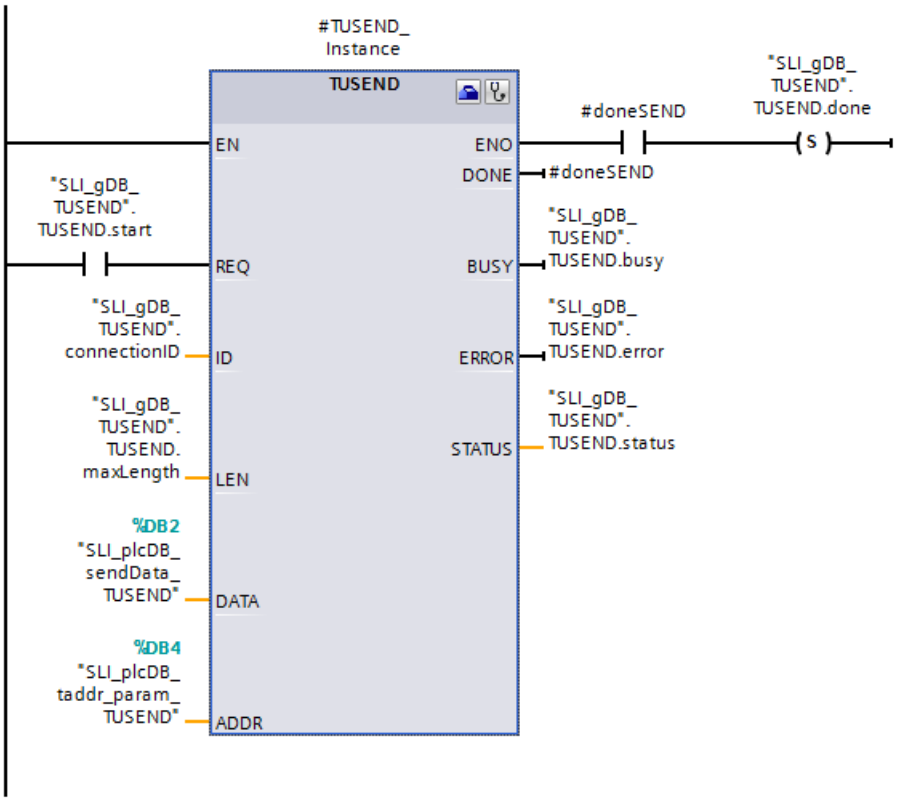
SLI_gDB_TUSEND

	Name	Data type	Start value
1	Static		
2	connectionID	CONN_OUC	16#0100
3	TUSEND	Struct	
4	start	Bool	false
5	maxLength	UDInt	0
6	busy	Bool	false
7	done	Bool	false
8	error	Bool	false
9	status	Word	16#0
10	memErrStat	Word	16#0

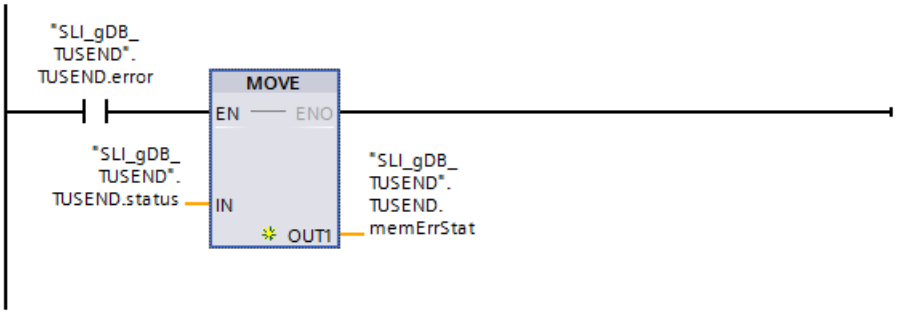
Create the FB "SLI_FB_TUSEND" Create the following local tags in it.

7	Static			
8	doneSEND	Bool	false	Non-ret...
9	TUSEND_Instance	TUSEND		

Network 1: Interconnect the parameter of the instruction "TUSEND" as follows:



Network 2: In case of error of TUSEND, save the status as follows.



Program of CPU 2

For the data record, create the PLC data type "TUSEND_User" based on the example of CPU 1.

For receiving data, create the following data block ("SLI_plcDB_rcvData_TURCV") based on the created PLC data type ("TUSEND_User").

SLI_plcDB_rcvData_TURCV

	Name	Data type	Start value
1	Static		
2	prename	String	''
3	lastname	String	''
4	userId	UInt	0

For addressing the communication partner, create the following data block ("SLI_plcDB_taddr_param_TURCV") based on the system data type "TADDR_Param.

SLI_plcDB_taddr_param_TURCV

	Name	Data type	Start value
1	Static		
2	REM_IP_ADDR	Array[1..4] of USInt	
3	REM_IP_ADDR[1]	USInt	192
4	REM_IP_ADDR[2]	USInt	168
5	REM_IP_ADDR[3]	USInt	0
6	REM_IP_ADDR[4]	USInt	2
7	REM_PORT_NR	UInt	2000
8	RESERVED	Word	16#0

To store the data, create a global data block ("SLI_gDB_TURCV") with the following structures and tags.

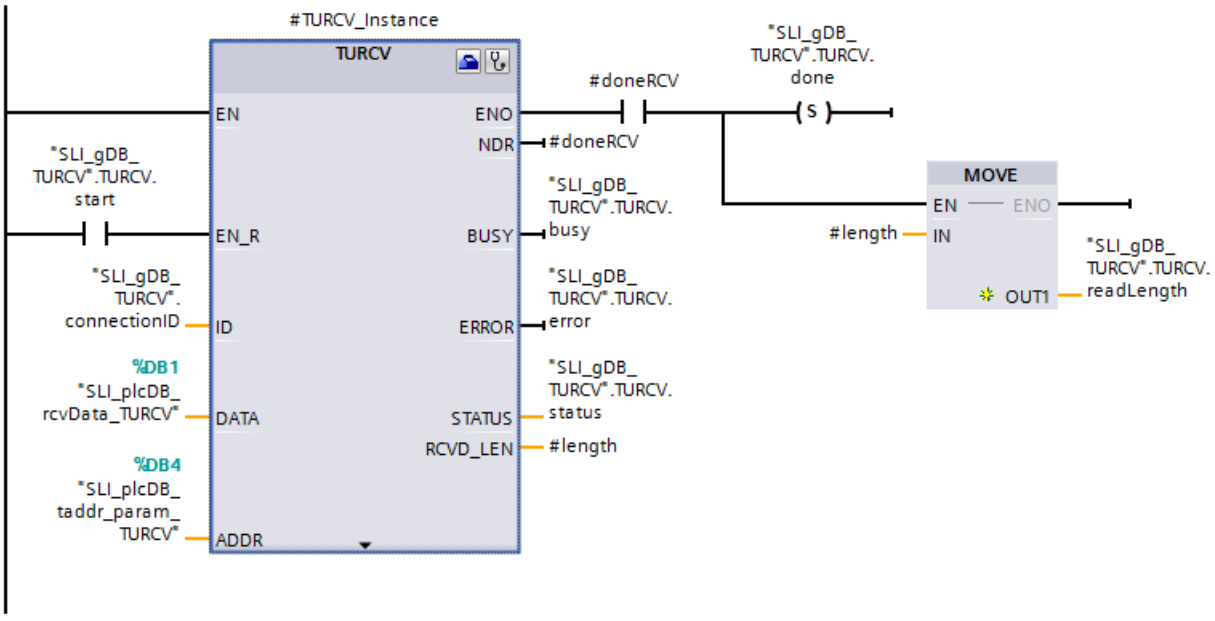
SLI_gDB_TURCV

	Name	Data type	Start value
1	Static		
2	connectionID	CONN_OUC	16#0100
3	TURCV	Struct	
4	start	Bool	false
5	busy	Bool	false
6	done	Bool	false
7	error	Bool	false
8	status	Word	16#0
9	memErrStat	Word	16#0
10	readLength	UDInt	0

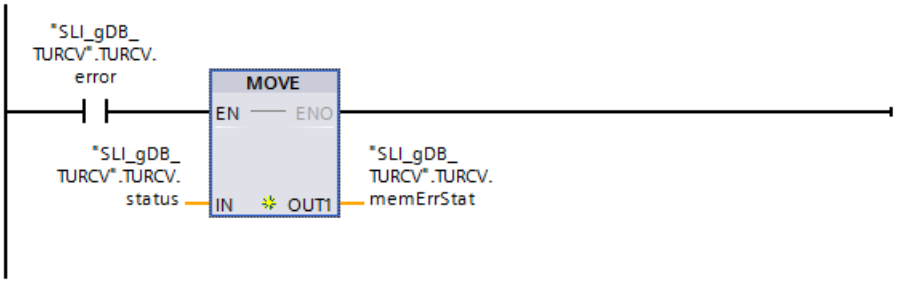
Create the FB "SLI_FB_TURCV" Create the following local tags in it.

7	Static			
8	doneRCV	Bool	false	Non-ret...
9	length	UDInt	0	Non-retain
10	TURCV_Instance	TURCV		

Network 1: Interconnect the parameter of the instruction "TURCV" as follows:



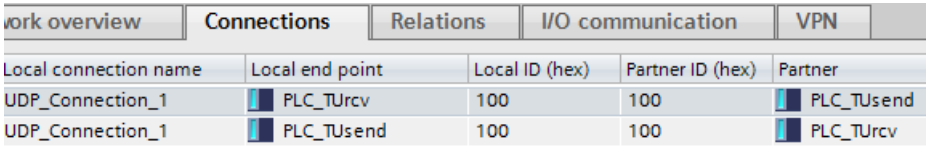
Network 2: In case of error of TURCV, save the status as follows.



Assigning communication connection

The address parameters for the connection of the two CPUs must be uniform.

- In each case store the hexadecimal value of the hardware identifier of the configured UDP connection at the input parameter ID ("connectionID").
  You can find the hardware identifier in the "Network view" under "Connections".



ork overview | Connections | Relations | I/O communication | VPN

Local connection name	Local end point	Local ID (hex)	Partner ID (hex)	Partner
UDP_Connection_1	PLC_TUrcv	100	100	PLC_TUsend
UDP_Connection_1	PLC_TUsend	100	100	PLC_TUrcv

Result of CPU 1

Network 1 (TUSEND):

Based on the input parameter ID ("connectionID"), the "TUSEND" instruction knows the communication connection to be used. Based on the input parameter ADDR

("SLI_plcDB_taddr_param_TUSEND"), the communication partner (CPU 2) can be addressed.

When the input parameter REQ ("TUSEND.start") supplies the signal state "TRUE", the "TUSEND" instruction is started. For multiple calls, the "TUSEND" instruction transmits the data record detected at input parameter DATA ("SLI_plcDB_sendData_TUSEND"). The size of the data record to be transmitted is restricted by the value of the input parameter LEN ("TUSEND.maxLength").

Successful transfer of the data record is indicated by the output parameter DONE ("#doneSEND") with the signal state "TRUE" and the output parameter STATUS ("TUSEND.status") with the value "0000". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "TUSEND.done".

The output parameter ERROR ("TUSEND.error") or the tag "TUSEND.memErrStatus" indicates that processing in the example is taking place without errors.

SLI_gDB_TUSEND

	Name	Data type	Start value	Monitor value
1	Static			
2	connectionID	CONN_OUC	16#0100	16#0100
3	TUSEND	Struct		
4	start	Bool	false	TRUE
5	maxLength	UDInt	0	0
6	busy	Bool	false	FALSE
7	done	Bool	false	TRUE
8	error	Bool	false	FALSE
9	status	Word	16#0	16#7000
10	memErrStat	Word	16#0	16#0000

Network 2 (TUSEND):

When TUSEND signals an error ("TUSEND.error" is "TRUE"), the reported status ("TUSEND.status") is stored permanently ("TUSEND.memErrStatus").

Result of CPU 2

Network 1 (TURCV):

Based on the input parameter ID ("connectionID"), the "TURCV" instruction knows the communication connection to be used. The communication partner (CPU 1) can be addressed based on the input parameter ADDR ("SLI_plcDB_taddr_param_TURCV").

When the input parameter EN_R ("TURCV.start") supplies the signal state "TRUE", the "TURCV" instruction is started. For multiple calls, the "TURCV" instruction receives the transmitted data record. The data record is detected at the input parameter DATA ("SLI_plcDB_rcvData_TURCV").

SLI_plcDB_rcvData_TURCV

	Name	Data type	Start value	Monitor value
1	Static			
2	prename	String	''	'Simon'
3	lastname	String	''	'Sample'
4	userId	UInt	0	1

The length in BYTES of the actually transmitted data record is detected with the output parameter LEN ("#length"). This value is only displayed during the success status. Then "0" is detected. Successful receipt of the data record is indicated by the output parameter DONE ("#doneRCV") with the signal state "TRUE" and the output parameter STATUS ("TURCV.status") with the value "0000".

Because the values of the output parameters are only displayed for the moment of their validity, do the following:

- Save the success status in the tag "TURCV.done"
- Save the length in BYTES in the tag "TURCV.readLength".

The output parameter ERROR ("TURCV.error") or the tag "TURCV.memErrStatus" indicates that processing in the example is taking place without errors.

SLI_gDB_TURCV

	Name	Data type	Start value	Monitor value
1	Static			
2	connectionID	CONN_OUC	16#0100	16#0100
3	TURCV	Struct		
4	start	Bool	false	TRUE
5	busy	Bool	false	TRUE
6	done	Bool	false	TRUE
7	error	Bool	false	FALSE
8	status	Word	16#0	16#7002
9	memErrStat	Word	16#0	16#0000
10	readLength	UDInt	0	514

Network 2 (TURCV):

When TURCV signals an error ("TURCV.error" is "TRUE"), the reported status ("TURCV.status") is stored permanently ("TURCV.memErrStatus").

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

TUSEND: Sending data (Page 4516)

TURCV: Receiving data (Page 4520)

T_RESET: Resetting the connection (S7-1200, S7-1500)

Description

The "T_RESET" instruction terminates and then reestablishes an existing connection.

The local end points of the connection are retained. They are generated automatically:

- If a connection has been configured and loaded to the CPU.
- If a connection has been generated by the user program, for example, by calling the instruction "TCON (Page 4474)".

The instruction "T_RESET" can be executed for all connection types (TCP, UDP, ISO-on-TCP, etc.). It does not matter whether the local interface of the CPU or the interface of a CM/CP was used for the connection.

Once the instruction "T_RESET" has been called using the REQ parameter, the connection specified with the ID parameter is terminated and, if necessary, the data send and receive buffer cleared. Canceling the connection also cancels any data transfer in progress. There is therefore a risk of losing data if data transfer is in progress. The CPU defined as the active connection partner will then automatically attempt to restore the interrupted communication connection. You therefore do not need to call the "TCON (Page 4474)" instruction to reestablish the communication connection.

The output parameters DONE, BUSY and STATUS indicate the status of the job.

Parameter

The following table shows the parameters of the instruction "T_RESET":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Control parameter REQUEST starts the job for terminating the connection specified by ID. The job starts on a rising edge.
ID	Input	CONN_OUC	I, Q, M, D, L or constant	Reference to the connection to the passive partner ID being interrupted must be identical to the corresponding parameter ID in the local connection description. Value range: W#16#0001 to W#16#0FFF
DONE	Output	BOOL	I, Q, M, D, L	Status parameter DONE • 0: Job not yet started or still executing. • 1: Job executed without errors
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter BUSY • 0: Job is complete. • 1: The job is not yet complete.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter ERROR • 0: No error occurred. • 1: Error occurred during processing. The STATUS parameter supplies detailed information on the type of error
STATUS	Output	WORD	I, Q, M, D, L	Status parameter STATUS Error information (see "STATUS parameter" table).

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS

STATUS* (W#16#...)	Explanation
0000	No error.
0001	Connection has not been established.
7000	No job processing active.
7001	Connection termination launched.
7002	Connection being terminated.
8081	Unknown connection specified at the ID parameter.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Example

You can find the example here: Program example for T_RESET (Page 4535).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

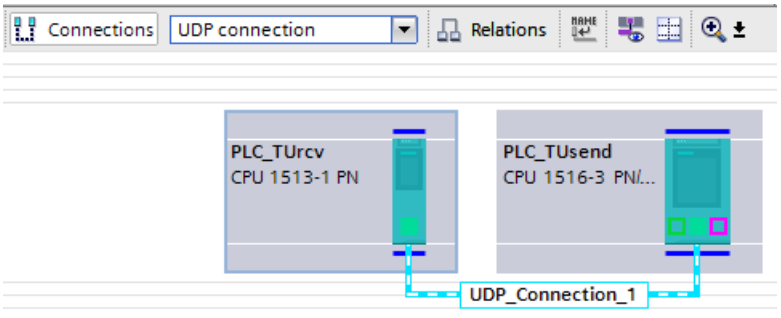
Program example for T_RESET (S7-1200, S7-1500)

Introduction

In the following example you use a configured connection (by UDP for example) between two CPUs (S7-1500). Reset the connection using instruction "T_RESET".

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. A UDP connection is configured.



- A low protection level under "<CPU> > Properties > Protection" ensures that read and write access is permitted for the CPUs.

Create tags and interconnect parameters (program of CPU 1)

To store the data, create a global data block ("SLI_gDB_T_RESET") with the following structures and tags.

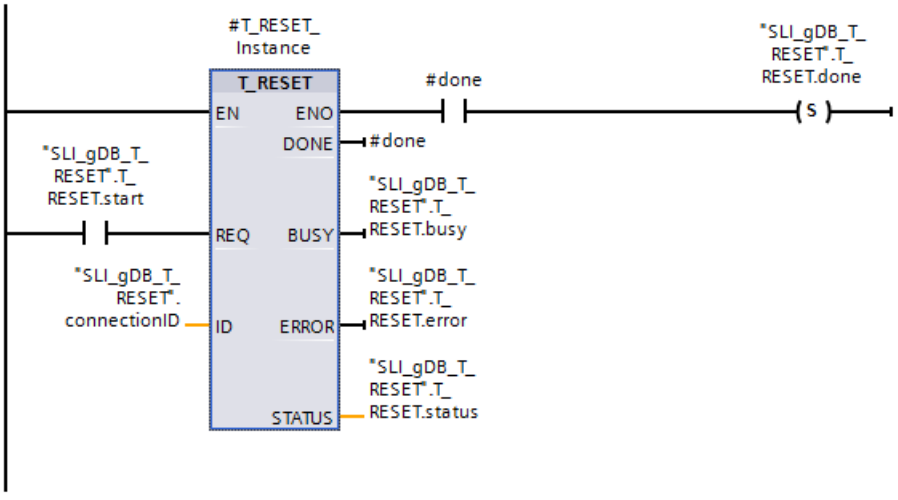
SLI_gDB_T_RESET

	Name	Data type	Start value
1	Static		
2	connectionID	CONN_OUC	16#0100
3	T_RESET	Struct	
4	start	Bool	false
5	busy	Bool	false
6	done	Bool	false
7	error	Bool	false
8	status	Word	16#0
9	memErrStatus	Word	16#0

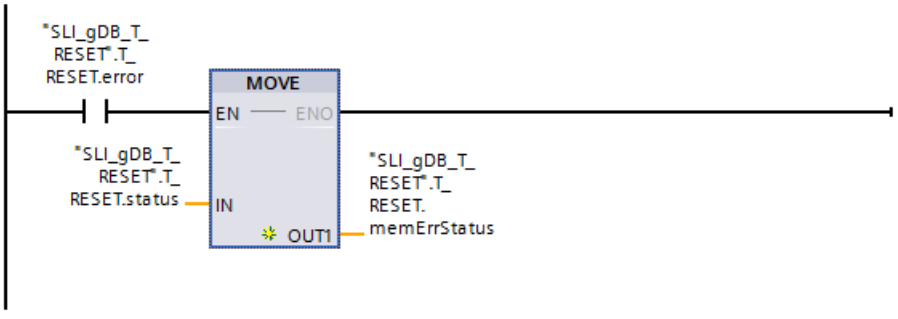
Create the FB "SLI_FB_T_RESET" Create the following local tags in it.

7	Static			
8	done	Bool	false	Non-ret...
9	T_RESET_Instance	T_RESET		

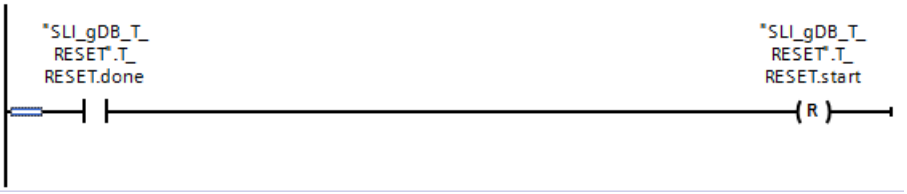
Network 1: Interconnect the parameter of the instruction "T_RESET" as follows:



Network 2: In case of error of T_RESET, save the status as follows.



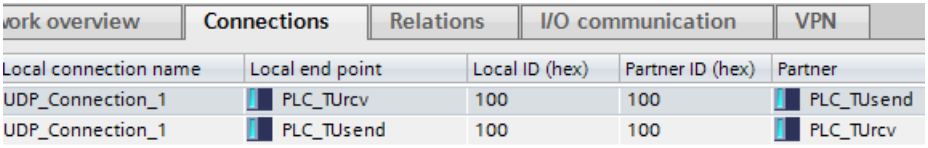
Network 3: To reset the "REQ" parameter, create the following interconnections.



Assigning communication connection

The addressing parameters must be adapted for the connection.

- Store the hexadecimal value of the hardware identifier of the configured UDP connection at the input parameter ID ("connectionID").
  You can find the hardware identifier in the "Network view" under "Connections".



Local connection name	Local end point	Local ID (hex)	Partner ID (hex)	Partner
UDP_Connection_1	PLC_TUrcv	100	100	PLC_TUsend
UDP_Connection_1	PLC_TUsend	100	100	PLC_TUrcv

Result

Network 1:

Based on the input parameter ID ("connectionID"), the "T_RESET" instruction knows the communication connection to be used.

When the input parameter REQ ("T_RESET.start") supplies the signal state "TRUE", the "T_RESET" instruction is started. The "T_RESET" instruction terminates the specified communication connection. All jobs that use the connection are terminated. The active connection partner (CPU 1) then re-establishes the connection automatically.

Successful transfer of the data record is indicated by the output parameter DONE ("#done") with the signal state "TRUE" and the output parameter STATUS ("T_RESET.status") with the value "0000".
Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "T_RESET.done".

The output parameter ERROR ("T_RESET.error") or the tag "T_RESET.memErrStatus" indicates that processing in the example is taking place without errors.

SLI_gDB_T_RESET

	Name	Data type	Start value	Monitor value
1	Static			
2	connectionID	CONN_OUC	16#0100	16#0100
3	T_RESET	Struct		
4	start	Bool	false	FALSE
5	busy	Bool	false	FALSE
6	done	Bool	false	TRUE
7	error	Bool	false	FALSE
8	status	Word	16#0	16#7000
9	memErrStatus	Word	16#0	16#0000

Network 2:

When T_RESET signals an error ("T_RESET.error" is "TRUE"), the reported status ("T_RESET.status") is stored permanently ("T_RESET.memErrStatus").

Network 3:

After the successful processing of T_RESET ("T_RESET.done"), T_RESET is stopped.

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

T_RESET: Resetting the connection (Page 4533)

T_DIAG: Checking the connection (S7-1200, S7-1500)

Description

You use the "T_DIAG" instruction to check the status of a connection and read further information on the local end point of this connection.

- The connection is referenced by the ID parameter. You can read both connection end points configured in the connection editor and programmed connection end points (e.g. with the "TCON" instruction).
  Temporary connection end points (for example end points created when you connect to an engineering station) cannot be diagnosed, as no connection ID is generated in this process.
- The connection information read is stored in a structure referenced by the RESULT parameter.
- The output parameter STATUS indicates whether it was possible to read the connection information. The connection information in the structure at the RESULT parameter is only valid if the "T_DIAG" instruction has been completed with STATUS = W#16#0000 and ERROR = FALSE.
  Connection information cannot be evaluated if an error occurs.

Possible connection information

Two different structures can be used to read the connection information at the RESULT parameter:

- The "TDiag_Status" structure only contains the most important information about a connection end point, for example the protocol used, the connection status and the number of data bytes sent or received.
- The structure "TDiag_StatusExt" supplies not just the most important information, but also the number of connection attempts, the reason for a connection abort, etc.

The structure and parameters of the two structures are described below (see the "TDIAG_Status and TDIAG_StatusExt structures" table).

Parameters

The following table shows the parameters of the instruction "T_DIAG":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Starts the instruction to check the connection specified in the ID parameter when there is a positive edge.
ID	Input	CONN_OUC (WORD)	I, Q, M, D, L or constant	Reference to the assigned connection. Value range: W#16#0001 to W#16#0FFF
RESULT	InOut	VARIANT	D	Pointer to the structure in which the connection information is stored. The structure TDiag_Status or TDiag_StatusExt can be used at the RESULT parameter (for a description, see the "TDIAG_Status and TDIAG_StatusExt structures" table).
DONE	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: Instruction not yet started or still in progress. • 1: Instruction executed without errors.
BUSY	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: Instruction not yet started or already completed. • 1: Instruction not yet completed. A new job cannot be started.
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: No error. • 1: Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

BUSY, DONE and ERROR parameters

You can check the status of "T_DIAG" instruction execution with the BUSY, DONE, ERROR and STATUS parameters. The BUSY parameter indicates the processing status. You use the DONE parameter to check whether or not an instruction has been executed successfully. The ERROR parameter is set if errors occur during execution of "T_DIAG".

The following table shows the relationship between the BUSY, DONE and ERROR parameters:

BUSY	DONE	ER-ROR	Description
1	-	-	The instruction is being processed.
0	1	0	The instruction has been executed successfully. The data in the structure referenced by RESULT are only valid if this is the case.
0	0	1	Instruction completed with an error. The cause of the error is output at the STATUS parameter.
0	0	0	No new instruction has been assigned.

Parameter STATUS

The following table shows the meaning of the values at the STATUS parameter:

STA-TUS* (W#16#...)	Explanation
0000	The instruction "T_DIAG" has been executed successfully. The data in the structure referenced at the RESULT parameter can be evaluated.
7000	No instruction processing active.
7001	Instruction processing launched.
7002	Connection information is being read (REQ parameter irrelevant).
8086	The value at the ID parameter is outside the valid range (W#16#0001 ... W#16#0FFF).
8089	The RESULT parameter points to an invalid data type (structures TDIAG_Status and TDIAG_StatusExt only).
80A3	The ID parameter references a connection end point which does not exist. With programmed connections, this error can also occur after the "TDISCON" instruction is called.
80B1	The RESULT parameter was changed before the execution of T_DIAG was completed. A change of RESULT is not permitted during execution of T_DIAG.
80C4	Internal error. Access to the connection end point is temporarily unavailable.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Structures TDIAG_Status and TDIAG_StatusExt

The table below details the form of the TDIAG_Status and TDIAG_StatusExt structures:

- From the InterfaceID to the ReceivedBytes parameter, the "TDIAG_StatusExt" and "TDIAG_Status" structures are identical.
- The structure TDIAG_StatusExt also includes the parameters ConnTrials to LastDisconnTimeStamp.

The value of each element is only valid if the instruction has been executed without errors. If an error occurs, the content of the parameters will not change.

Name	Data type	Description
The following parameters are in both the TDIAG_Status and the TDIAG_StatusExt structure:		
InterfaceID	HW_ANY	Interface ID (LADDR) of the CPU or the CM/CP.
ID	CONN_OUC	ID of the connection diagnosed. Following a successful call, the value of this element is identical to the parameter ID of the "T_DIAG" instruction.
ConnectionType	BYTE	Protocol type used for connection: • 0x01: Not used. • ... • 0x0B: TCP protocol (IP_v4) • 0x0C: ISO-on-TCP protocol (RFC1006) • 0x0D: TCP protocol (DNS) • 0x0E: Dial-in protocol • 0x0F: WDC protocol • 0x10: SMTP protocol • 0x11: TCP protocol • 0x12: TCP and ISO-on-TCP protocol (RFC1006) • 0x13: UDP protocol • 0x14: Reserved • 0x15: PROFIBUS bus access protocol (FDL) • 0x16: ISO 8073 transport protocol (ISO native) • ... • 0x20: SMTP or SMTPS protocol - based on IPv4 • 0x21 SMTP or SMTPS protocol - based on IPv6 • 0x22: SMTP or SMTPS protocol based on FQDN (**F**ully **Q**ualified **D**omain **N**ame) • ... • 0x70: S7 connection • Other: Reserved
ActiveEstablished	BOOL	• FALSE: Locally, the passive connection end point • TRUE: Locally, the active connection end point

Name	Data type	Description
State	BYTE	Current status of the connection end point • 0x00: Not used. • 0x01: Connection terminated. Temporary status, for example, after the "T_RESET" instruction is called. The system then automatically attempts to reestablish the connection. • 0x02: The active connection end point is attempting to establish a connection to the remote communication partner. • 0x03: The passive connection end point is waiting for establishment of the connection to the remote communication partner. • 0x04: Connection established. • 0x05: The connection is being terminated. This may be because the "T_RESET" or "T_DISCON" instruction has been called. Other possible reasons are protocol errors and line breaks. • 0x06..0xFF: Not used.
Kind	BYTE	Mode of the connection end point: • 0x00: Not used. • 0x01: Configured, static connection which has been configured and loaded to the CPU. • 0x02: Configured, dynamic connection which has been configured and loaded to the CPU (not currently supported). • 0x03: Programmed connection generated in the user program with the instruction "TCON". A call of the instruction "TDISCON" or a transition to CPU STOP status has destroyed the connection end point. • 0x04: Temporary, dynamic connection established by the engineering station (ES) or operator station (OS) (this connection currently not be diagnosed due to the missing ID). • 0x05..0xFF: Not used.
SentBytes	UDINT	Number of data bytes sent.
ReceivedBytes	UDINT	Number of data bytes received.
The following parameters only occur in the TDiag_StatusExt structure:		
ConnTrials	UDINT	Number of connection attempts. Once a connection has been successfully established, ConnTrials has the value 0. If this element is not equal to 0, this indicates that there are connection problems. Note: With a passive connection end point, this value is never greater than 1.
ConnTrialsSuccess	UDINT	Number of successful connection attempts. This element is never reset during the life cycle of a connection end point, and returns to 0 after reaching 0xFFFF FFFF. Note: This parameter is 1 if there has never been a problem in this connection.

Name	Data type	Description
LastConnErrReason	UDINT	Error ID output during the last connection attempt with errors (the error messages are identical to those at the LastDisconnReason parameter): • 0x4F01: Remote connection end point cannot be reached (this error usually occurs during connection establishment). • 0x4F02: Connection terminated locally. • 0x4F03: Connection terminated by remote communication partner. • 0x4F04: Connection terminated by a protocol error. • 0x4F05: Connection terminated by a network problem detected locally. • 0x4F06: Connection terminated by a network problem detected remotely. • 0x4F07: Connected terminated due to timeout in protocol. • 0x4F08: Incorrect parameter assignment: Connection to be established to local partner's own address. • 0x4F09: Connection temporarily reset by a call of the instruction "T_RESET". • 0x4F0A: Insufficient connection resources available (quantity exceeded) • 0x4F0B: Internal error: Incorrect addressing parameters • 0x4F0C: Internal CPU communication error • 0x4F0D: Internal AS communication error between CPU and CM/CP • 0x4F0E: The local TCP/UDP port (or RFC1006-T selector) specified is already in use.
LastConnErrTimeStamp	LDT	Time of the last connection attempt with errors.
LastDisconnReason	UDINT	Error ID which led to the last connection termination (the error messages are identical to those at the LastConnErrReason parameter): • 0x4F01: Remote connection end point cannot be reached (this error usually occurs during connection establishment). • 0x4F02: Connection terminated locally. • 0x4F03: Connection terminated by remote communication partner. • 0x4F04: Connection terminated by a protocol error. • 0x4F05: Connection terminated by a network problem detected locally. • 0x4F06: Connection terminated by a network problem detected remotely. • 0x4F07: Connected terminated due to timeout in protocol. • 0x4F08: Incorrect parameter assignment: Connection to be established to local partner's own address. • 0x4F09: Connection temporarily reset by a call of the instruction "T_RESET". • 0x4F0A: Insufficient connection resources available (quantity exceeded) • 0x4F0B: Internal error: Incorrect addressing parameters • 0x4F0C: Internal CPU communication error • 0x4F0D: Internal AS communication error between CPU and CM/CP • 0x4F0E: The local TCP/UDP port (or RFC1006-T selector) specified is already in use.
LastDisconnTimeStamp	LDT	Time of the last connection termination.

Example

You can find the example here: Program example for T_DIAG (Page 4544).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

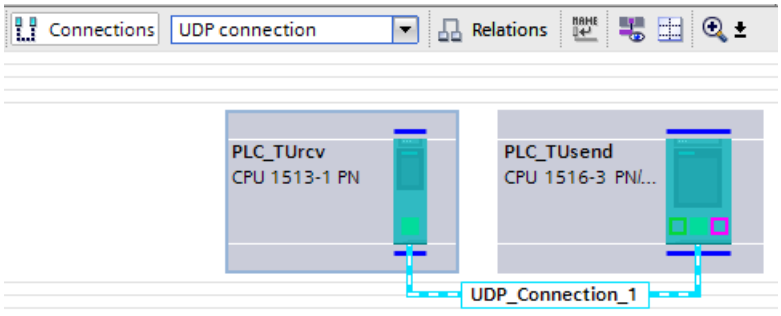
Program example for T_DIAG (S7-1200, S7-1500)

Introduction

In the following example you use a configured connection (by UDP for example) between two CPUs (S7-1500). The instruction "T_DIAG" is used to diagnose the connection and read information from the local end point of the connection.

Requirement

- Two CPUs of the S7-1500 series are available and connected to each other over PROFINET. A UDP connection is configured.



- A low protection level under "<CPU> > Properties > Protection" ensures that read and write access is permitted for the CPUs.

Create tags and interconnect parameters (program of CPU 1)

To store the data, create a global data block ("SLI_gDB_T_DIAG") with the following structures and tags.

Note: "TDiag_StatusExt" is a system data type for recorded the diagnostics data.

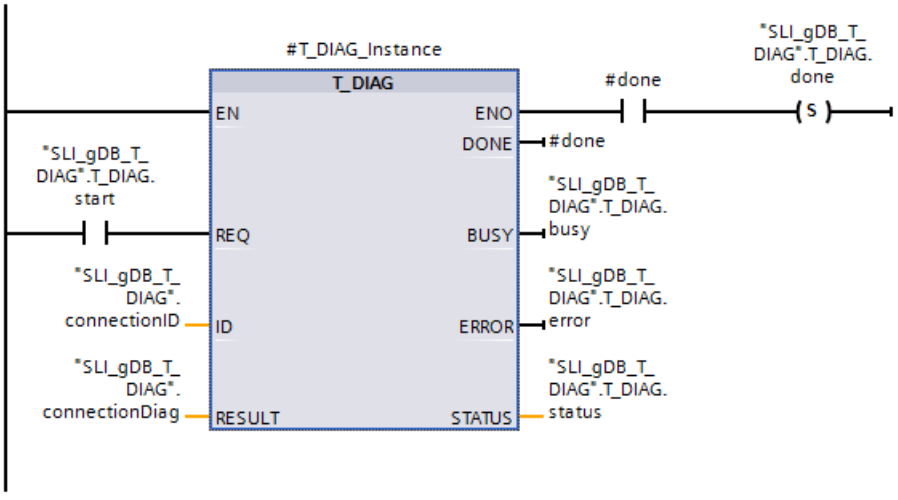
SLI_gDB_T_DIAG

	Name	Data type	Start value
1	Static		
2	connectionID	CONN_OUC	16#0100
3	T_DIAG	Struct	
4	start	Bool	false
5	busy	Bool	false
6	done	Bool	false
7	error	Bool	false
8	status	Word	16#0
9	memErrStatus	Word	16#0
10	connectionDiag	TDiag_StatusExt	

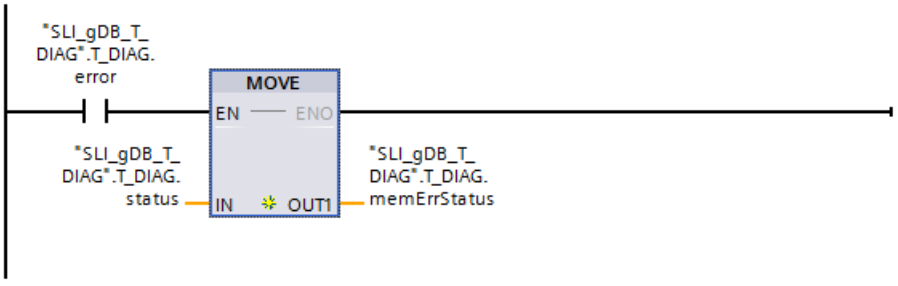
Create the FB "SLI_FB_T_DIAG" Create the following local tags in it.

7	Static			
8	T_DIAG_Instance	T_DIAG		
9	done	Bool	false	Non-retain

Network 1: Interconnect the parameter of the instruction "T_DIAG" as follows:



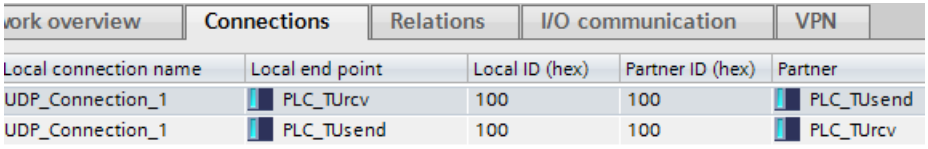
Network 2: In case of error of T_DIAG, save the status as follows.



Assigning communication connection

The addressing parameters must be adapted for the connection.

- Store the hexadecimal value of the hardware identifier of the configured UDP connection at the input parameter ID ("connectionID").
  You can find the hardware identifier in the "Network view" under "Connections".



Result

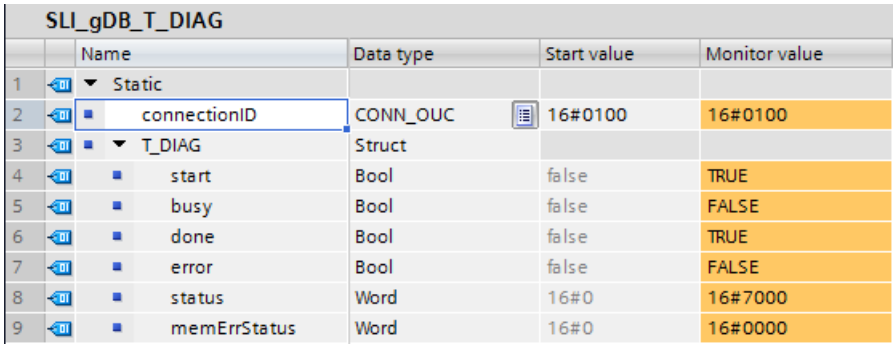
Network 1:

Based on the input parameter ID ("connectionID"), the "T_DIAG" instruction knows the communication connection to be used.

When the input parameter REQ ("T_DIAG.start") supplies the signal state "TRUE", the "T_DIAG" instruction is started. The instruction "T_DIAG" reads the status of the connection and the status of the local end point of the connection.

Successful transfer of the data record is indicated by the output parameter DONE ("#done") with the signal state "TRUE" and the output parameter STATUS ("T_DIAG.status") with the value "0000". Because the values of the output parameters are only displayed for the moment of their validity, save the success status in the tag "T_DIAG.done".

The output parameter ERROR ("T_DIAG.error") or the tag "T_DIAG.memErrStatus" indicates that processing in the example is taking place without errors.



The diagnostics data are recorded at the RESULT parameter ("TDiag_StatusExt").

10	connectionDiag	TDiag_StatusExt		
11	InterfaceId	HW_ANY	16#0	16#0040
12	ID	CONN_OUC	16#0	16#0100
13	ConnectionType	Byte	0	16#13
14	ActiveEstablished	Bool	false	FALSE
15	State	Byte	0	16#04
16	Kind	Byte	0	16#01
17	SentBytes	UDInt	0	0
18	ReceivedBytes	UDInt	0	0
19	ConnTrials	UDInt	0	0
20	ConnTrialsSuccess	UDInt	0	2
21	LastConnErrReason	UDInt	0	0
22	LastConnErrTimeSt..	LDT	LDT#1970-01-01-0	LDT#1970-01-01-0...
23	LastDisconnReason	UDInt	0	20226
24	LastDisconnTimeS...	LDT	LDT#1970-01-01-0	LDT#2012-01-21-0...

Network 2:

When T_DIAG signals an error ("T_DIAG.error" is "TRUE"), the reported status ("T_DIAG.status") is stored permanently ("T_DIAG.memErrStatus").

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

T_DIAG: Checking the connection (Page 4538)

T_CONFIG: Configure interface (S7-1200, S7-1500)

Description T_CONFIG (S7-1200, S7-1500)

Description

The "T_CONFIG" instruction is used for the program-controlled configuration of the integrated PROFINET interfaces of the CPU or the interface of a CP/CM.

Configuration of the integrated PROFINET interfaces of the CPU

With the "T_CONFIG" instruction, you can change the Ethernet address, the PROFINET device name or the IP addresses of the NTP servers for time-of-day synchronization from within the user program. The existing configuration data is overwritten.

You can make the following changes:

- Settings for the IP protocol
  - IP address
  - Subnet mask
  - Router address
- Settings for PROFINET
  - Assignment of the PROFINET device name

  Note

  Dependency between device names and IP protocol

  The IP protocol must also be set when a new device name is assigned.
- Settings for time-of-day synchronization
  - Assignment of the IP addresses of the NTP servers for time-of-day synchronization (only with S7-1500 and there only for the PROFINET interface [X1} of a modular CPU)

These settings correspond to the configuration options under "IP protocol" and "PROFINET" in the "Ethernet addresses " dialog or the "NTP mode" in the "Time synchronization" dialog. This is displayed in the "Devices & Networks" view under the properties of the PROFINET interface.

⚠ WARNING

Restart of CPU after execution of the "T_CONFIG" instruction (only S7-1200 CPUs with firmware versions V2.0 to V4.1.2)

The CPU is restarted after you have executed the instruction to change an IP parameter. The CPU goes to STOP mode, a warm restart is carried out and the CPU starts up again (RUN mode).

Make sure that the control process is in a secure operating mode after the CPU has been restarted following execution of the instruction. Uncontrolled operation can result in serious material damage or personal injury due to malfunctions or programming errors, for example. Non-retentive data could be lost.

Configuration of the PROFINET interface of a CP 1543-1

With the "T_CONFIG" instruction you can change the IPv6 address, the MAC address or the IP addresses of up to two DNS servers of the PROFINET interface of a CP 1543-1 from within the user program. The existing configuration data is overwritten.

Requirement

In order to use the instruction must be explicitly specified in the hardware configuration that the assignment of IP address parameters and device name must be made by the user program.

- For this purpose open the properties of the PROFINET interface in the device view. In the "Ethernet addresses" or "Time synchronization" dialog, enable the following options:
  - In order to change the IP address parameters with "T_CONFIG": Select the setting "IP address is set directly at the device" under "IP protocol".
  - In order to change the PROFINET device name with "T_CONFIG":
    Select the setting "PROFINET device name is set directly at the device" under "PROFINET".
    Also select the setting "IP address is set directly at the device" under "IP protocol".
  - To change the IP addresses of the NTP servers with "T_CONFIG". Select "Enabled time synchronization via NTP server" and specify the IP address of at least one NTP server.
- The configuration data must be stored in the following system data types and assigned at the parameter CONF_DATA (Page 4551):
  - Store the IP address, subnet mask, and the router address in the system data type IF_CONF_V4.
  - Store the device name in the system type IF_CONF_NOS. Observe the restrictions that exist for assigning the name of the device (see parameter CONF_DATA (Page 4551)).
  - Use store the IP addresses for the NTP time synchronization in the system data type IF_CONF_NTP.
  - Store the IPv6 address of the PROFINET interface of a CP 1543-1 in the system data type IF_CONF_IPV6.
  - Store the MAC address of the PROFINET interface of a CP 1543-1 in the system data type IF_CONF_MAC.
  - Store the IP addresses of up to two DNS servers for the PROFINET interface of a CP 1543-1 in the system data type IF_CONF_DNS.

Functional description

The "T_CONFIG" instruction works asynchronously, which means its execution extends over multiple calls. The configuration process is started by calling "T_CONFIG" with REQ = 1. Only one job can be active at any time.

The block is edge triggered which means that after BUSY= FALSE the block must be re-called with REQ=FALSE to enable the instance.

Parameter

The following table shows the parameters of the instruction "T_CONFIG":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Calling the instruction with REQ = 1 starts processing of the instruction.
INTERFACE	Input	HW_INTERFACE	I, Q, M, D, L or constant	Hardware identifier of the interface The hardware ID is displayed in the properties of the interface in the device view, and in the system constants of the PLC tags.
CONF_DATA (Page 4551)	Input	VARIANT	D, L	Pointer to the higher-level structure that contains the system data types IF_CONF_HEADER, IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC and IF_CONF_DNS (see description of the parameters CONF_DATA).
DONE (Page 4556)	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: Processing not yet complete. • 1: Processing of instruction finished successfully.
BUSY (Page 4556)	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: Processing of the instruction has not started, completed or canceled yet. • 1: Processing of instruction is in progress
ERROR (Page 4556)	Output	BOOL	I, Q, M, D, L	Status parameter: • 0: No error • 1: Error
STATUS (Page 4556)	Output	DWORD	I, Q, M, D, L	Detailed status information: Detailed error and status information in the form of an error code are output at the parameter STATUS.
ERR_LOC (Page 4556)	Output	DWORD	I, Q, M, D, L	Error location: • 0: Error during execution of the instruction or when assigning parameters. • > 0: Errors in the structure or content of the configuration data at the parameter CONF_DATA.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Example

You will find the example here: Program example for T_CONFIG (Page 4558).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Difference between synchronous and asynchronous instructions (Page 2571)

Parameter CONF_DATA (S7-1200, S7-1500)

Structure of the configuration data

The configuration data at the parameter CONF_DATA can be stored in a global data block or in the section "Static" of the block interface.

The configuration data must be stored according to the following structure:

Name	Data type	Description
ConfData	Struct	Parent structure that is assigned at the parameter CONF_DATA.
Header	IF_CONF_HEADER	The header is used to define the number of these system data types. The system data type IF_CONF_HEADER must always be included.
IPData	IF_CONF_V4	The IP address, subnet mask and router address are stored in this system data type.
NoS	IF_CONF_NOS	The PROFINET device names are stored in this system data type. Only create IF_CONF_NOS if you also want to change the device names with "T_CONFIG".
NTP	IF_CONF_NTP	In this system data type, use store the IP addresses of the NTP servers for time synchronization.
IP_V6	IF_CONF_V6	Store the IPv6 address of a CP 1543-1 in this system data type.
MAC	IF_CONF_MAC	Store the MAC address of a CP 1543-1 in this system data type.
DNS	IF_CONF_DNS	Store the IP addresses of up to two DNS servers for a CP 1543-1 in this system data type .

Create the system data types IF_CONF_HEADER, IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC and IF_CONF_DNS be entering the name of the system data type in the "Data type " column of the data block or the block interface. The name for the system data types can be freely assigned. The system data types can be combined in any order.

System data type IF_CONF_Header

Use the system data type IF_CONF_Header to specify the number of system data types IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC and IF_CONF_DNS that are used when executing "T_CONFIG".

Byte	Parameter	Data type	Start value	Description
0 ... 1	FieldType	UINT	0	Field type: Must always have the value "0".
2 ... 3	FieldId	UINT	0	Error ID: Must always have the value "0".
4 ... 5	SubfieldCount	UINT	0	Number of system data types IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC and IF_CONF_DNS used

System data type IF_CONF_V4

The IP address, subnet mask and router address are defined with the system data type IF_CONF_V4.

Byte	Parameter	Data type	Start value	Description
0 ... 1	Id	UINT	30	System data type ID The start value of this parameter must not be changed.
2 ... 3	Length	UINT	18	Length of the system data type IF_CONF_V4 The start value must be used for the length specification since the parameters of IF_CONF_V4 have a fixed length and structure.
4 ... 5	Mode	UINT	0	Validity of addressing: • 1: Permanent validity of the configuration data • 2: Temporary validity of the configuration data including the deletion of existing permanent configuration data
6 ... 9	InterfaceAddress	IP_V4 *	0.0.0.0	IP address
10 - 12	SubnetMask	IP_V4 *	0.0.0.0	Subnet mask
14 - 16	DefaultRouter	IP_V4 *	0.0.0.0	Router address
* The data type IP_V4 is a structure of 4 BYTE, which includes the respective address of the respective parameter (e.g. at parameter SubnetMask the four-digit address of the subnet mask of the IP protocol).				

System data type IF_CONF_NOS

With the system data type IF_CONF_NOS you specify the station name to be assigned when the "T_CONFIG" instruction executes.

Byte	Parameter	Data type	Start value	Description
0 ... 1	Id	UINT	40	System data type ID The start value of this parameter must not be changed.
2 ... 3	Length	UINT	246	Length of the system data type IF_CONF_NOS in bytes. • For an absolute length, the value for the Lenght parameter consists of: – 6 bytes for the parameter Id, Length and Mode. – up to 240 bytes for the device name (parameter NOS). Example: An overall length of 10 is the result for the device name "plc1" with a length of 4 characters (=4 bytes). • Use the default start value 246 at the Lenght parameter for a dynamic length. Make sure that you enter the value "0" after the name (see description for NOS parameter).

Byte	Parameter	Data type	Start value	Description
4 ... 5	Mode	UINT	0	Validity of the device name change: • 1: Permanent validity of the device name. • 2: Temporary validity of the device name.
6 ... 244	NOS	ARRAY [1...240] of Byte	0	Device name (Name of Station) • You must occupy the ARRAY from the first byte. The station name is deleted if "0" is assigned as the first byte. • The minimum length for the name is 2 bytes. The maximum length for a name is 240 bytes. • If the device name is shorter than specified at the parameter Length then a zero byte (16#0 hex) must be entered after the actual station name (according to IEC 61185-6-10). Otherwise NOS will be rejected and the instruction "T_CONFIG" outputs the error codeDW#16#C0809400 at the parameter STATUS. • If the device name is longer than specified at the parameter Length then the device name will only be written up to to specified length. The following restrictions apply for the device name: • The name must be specified in ASCII code. • Only lowercase letters, numbers, hyphen or dots may be used for the name. – The name may not begin or end with a hyphen. – The name may not begin with numbers. – The name may not have the form n.n.n.n (n = 0, ... 999). – The name may not begin with the string "port-xyz" or "port-xyz-abcde" (a, b, c, d, e, x, y, z = 0, ... 9). • A name component between two points may be up to 63 characters long. • No special characters such as umlauts, brackets, underscore, slash, blank etc. The error code C080_9400 is output at the parameter STATUS if an invalid character is used.

System data type IF_CONF_NTP

Using the system data type IF_CONF_NTP you specify the IP addresses of the NTP servers for time synchronization.

Byte	Parameter	Data type	Start value	Description
0 ... 1	Id	UINT	17	System data type ID The start value of this parameter must not be changed.
2 ... 3	Length	UINT	22	Length of the system data type IF_CONF_NTP in bytes The start value must be used for the length specification since the parameters of IF_CONF_NTP have a fixed length and structure.
4 ... 5	Mode	UINT	2	Validity: • 1: Permanent validity (not permitted) • 2: Temporary validity
6 ... 9	NTP_IP[1]	ARRAY [1...4] of IP_V4		IP address of NTP server 1
...	...			IP address of NTP server 2 ... 3
18 ... 21	NTP_IP[4]	ARRAY [1...4] of IP_V4		IP address of NTP server 4

System data type IF_CONF_V6 (only when using a CP 1543-1)

With the system data type IF_CONF_V6 you define an IPv6 address for the interface selected with INTERFACE.

Byte	Parameter	Data type	Start value	Description
0 ... 1	Id	UINT	22	System data type ID The start value of this parameter must not be changed.
2 ... 3	Length	UINT	22	Length of the system data type IF_CONF_V6 in bytes The start value must be used for the length specification since the parameters of IF_CONF_V6 have a fixed length and structure.
4 ... 5	Mode	UINT	0	Validity of the IPv6 address: • 1: Permanent validity • 2: Temporary validity
6 ... 21	InterfaceAddress	IP_V6 *		IPv6 address
* The data type IP_V6 is a structure of 16 bytes.				

System data type IF_CONF_MAC (only when using a CP 1543-1)

With the system data type IF_CONF_MAC you set a MAC address for the interface selected with INTERFACE.

Byte	Parameter	Data type	Start value	Description
0 ... 1	Id	UINT	3	System data type ID The start value of this parameter must not be changed.
2 ... 3	Length	UINT	12	Length of the system data type IF_CONF_MAC in bytes The start value must be used for the length specification since the parameters of IF_CONF_MAC have a fixed length and structure.
4 ... 5	Mode	UINT	0	Validity of MAC address: • 1: Permanent validity • 2: Temporary validity
6 ... 11	Mac	ARRAY [1...6] of Byte		MAC address Note that by changing the MAC address, connections can abort.

System data type IF_CONF_DNS (only when using a CP 1543-1)

With the system data type IF_CONF_DNS you set IP addresses of up to two DNS servers for the interface selected with INTERFACE.

Byte	Parameter	Data type	Start value	Description
0 ... 1	Id	UINT	16	System data type ID The start value of this parameter must not be changed.
2 ... 3	Length	UINT	14	Length of the system data type IF_CONF_DNS in bytes The start value must be used for the length specification since the parameters of IF_CONF_DNS have a fixed length and structure.
4 ... 5	Mode	UINT	0	Validity: • 1: Permanent validity • 2: Temporary validity
6 ... 9	DNS_IP1	IP_V4		IP address of DNS server 1
10 ... 13	DNS_IP2	IP_V4		IP address of DNS server 2 If only one address is to be set, the IP address of DNS server 2 must be set to 0.0.0.0.

DONE, BUSY and ERROR parameters (S7-1200, S7-1500)

Description

The following table shows the relationship between BUSY, DONE and ERROR. Using this table, you can determine the current status of the instruction and when the transfer of configuration data is concluded.

BUSY	DONE	ERROR	Description
TRUE	FALSE	FALSE	The job is being processed.
FALSE	TRUE	FALSE	Job successfully completed.
FALSE	FALSE	TRUE	The job ended with an error. The cause of the error can be found in the parameter STATUS (Page 4556).
FALSE	FALSE	FALSE	The instruction was not assigned a (new) job.

Parameter STATUS and ERR_LOC (S7-1200, S7-1500)

Description

The status and error messages of the instruction "T_CONFIG" are output at the parameters STATUS and ERR_LOC.

- The cause of the error is output at the parameter STATUS.
- The location of the error that occurred is output at the parameter ERR_LOC. The following options are available here:
  - 16#0000_0000: Error when calling the instruction (e.g. errors when assigning parameters to the instruction or in communication with the PROFINET interface).
  - 16#0001_0000: Error with the configuration data in the parameters of the system data type IF_CONF_HEADER.
  - 16#0001_000x: Error in the configuration data in the parameters of system data type IF_CONF_V4 or IF_CONF_NOS or IF_CONF_NTP or IF_CONF_V6 or IF_CONF_MAC or IF_CONF_DNS (x specifies the position of the bad sub block in the T_CONFIG structure. If the T_CONFIG structure contains, for example a sub block for the IP address and a sub block for the station name and the error is located in the sub block for the station name, ERR_LOC has the value 0001_0002.)

The following table shows the possible values for the parameters STATUS and ERR_LOC:

STATUS*	ERR_LOC*	Explanation
0000_0000	0000_0000	Order processing completed without errors.
0070_0000	0000_0000	No job processing active.
0070_0100	0000_0000	Start of the order processing.
0070_0200	0000_0000	Intermediate call (REQ irrelevant).
C08x_yy00	0000_0000	General error information. See also: GET_ERR_ID: Get error ID locally (Page 2840)
C080_8000	0000_0000	Error at call of the instruction: The hardware ID at the parameter Interface is invalid.

STATUS*	ERR_LOC*	Explanation
C080_8100	0000_0000	Error at call of the instruction: The hardware ID at the parameter Interface does not address a PROFINET interface.
C080_8700	0000_0000	Error at call of the instruction: Incorrect length of the data block at the parameter CONF_DATA.
C080_8800	0001_0000	Error in the system data type IF_CONF_HEADER: The parameter FieldType has an invalid value. Use the value "0" for FieldType.
C080_8900	0001_0000	Error in the system data type IF_CONF_HEADER: The parameter FieldId has an invalid value. Use the value "0" for FieldId.
C080_8A00	0001_0000	Error in the system data type IF_CONF_HEADER: Incorrect number at the parameter SubfieldCount. Enter the correct number of system data types IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC or IF_CONF_DNS being used.
C080_8B00	0001_000x	Error in the system data type IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC or IF_CONF_DNS: The parameter Id has an invalid value. For IF_CONF_V4 use "30", for IF_CONF_NOS "40", for IF_CONF_NTP "17", for IF_CONF_V6 "22", for IF_CONF_MAC "3", for IF_CONF_DNS "16". Note: IF_CONF_NTP is only permitted if T_CONFIG is used on the first interface [X1] of an S7-1500 modular CPU.
C080_8C00	0001_000x	Error in the system data type IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC or IF_CONF_DNS: Incorrect data type system used, wrong order or multiple use of a system data type.
C080_8D00	0001_000x	Error in the system data type IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC or IF_CONF_DNS: The parameter Length has an incorrect or invalid value.
C080_8E00	0001_000x	Error in the system data type IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC or IF_CONF_DNS: The parameter Mode has an incorrect or invalid value. • With IF_CONF_V4 and IF_CONF_NOS only the values "1" (permanent) or "2" (temporary) are permitted. • With IF_CONF_NTP only the value "2" (temporary) is permitted.
C080_9000	0001_000x	Error in the system data type IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC or IF_CONF_DNS: Configuration data cannot be applied. Possible cause: • With IF_CONF_V4: In the hardware configuration, the setting "Set IP address on the device" was not selected.. • With IF_CONF_NOS: In the hardware configuration, the setting "Set PROFINET device name on the device" was not selected.. • With IF_CONF_NTP: In the hardware configuration, the setting "Enable time synchronization via NTP server " was not selected and no IP address was set for NTP servers..
C080_9400	0001_000x	Error in the system data type IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC or IF_CONF_DNS: A parameter value is undefined or invalid.
C080_9500	0001_000x	Error in the system data type IF_CONF_V4, IF_CONF_NOS, IF_CONF_NTP, IF_CONF_V6, IF_CONF_MAC or IF_CONF_DNS: The values of two parameters are inconsistent.

STATUS*	ERR_LOC*	Explanation
C080_C200	0000_0000	Error at call of the instruction: The configuration data can not be transferred. Possible cause: The PROFINET interface is not accessible.
C080_C300	0000_0000	Error at call of the instruction: Insufficient resources (e.g., multiple calling of "T_CONFIG" with different parameters).
C080_C400	0000_0000	Error at call of the instruction: Temporary communication error. Time indication for change to daylight saving time.
C080_D200	0000_0000	Error at call of the instruction: Call not possible. Instruction is not supported by the selected PROFINET interface.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Program example for T_CONFIG (S7-1200, S7-1500)

Introduction

In the following example you use a configured connection (by UDP for example) between two CPUs. Configure the IP address and the PROFINET device name of CPU 1 using the instruction "T_CONFIG".

Requirements

Set connection:

- Two CPUs (for example, S7-1513-1 PN) are available and connected to each other via PROFINET.
- A configured connection is not required.

Setting up PROFINET of CPU 1:

1. Open the device view of the CPU 1 "> Properties > PROFINET interface > Ethernet addresses > PROFINET".
2. To set the device names, select the following options:
   - "IP address is set directly at the device".
   - "PROFINET device name is set directly at the device".

Adjust the value of "hwid":

- Change the value of "hwid" according to the hardware identifier of the PROFINET interface of your local device (CPU 1).

Note

Open "PLC tags > Show all tags > System constants". Find the entry "<Local~PROFINET_interface_1>", with the data type "Hw_Interface". The "value" cell contains the hardware identifier.

Create tags and interconnect parameters (program of CPU 1)

To store the data, create a global data block ("SLI_gDB_T_CONFIG") with the following structures and tags.

SLI_gDB_T_CONFIG

	Name	Data type	Start value
1	Static		
2	T_CONFIG	Struct	
3	start	Bool	false
4	hwid	HW_INTERFACE	64
5	busy	Bool	false
6	done	Bool	false
7	error	Bool	false
8	errorLocation	DWord	16#0
9	status	DWord	16#0
10	memErrStat	DWord	16#0
11	configData	Struct	
12	header	IF_CONF_Header	
13	deviceIP	IF_CONF_v4	
14	deviceName	IF_CONF_NOS	

In the structure "configData": Assign parameters for the system data type "IF_CONF_Header" for specifying the size of the PROFINET data, as follows:

12	header	IF_CONF_Header	
13	FieldType	UInt	0
14	FieldId	UInt	0
15	SubfieldCount	UInt	2

In the structure "configData": Assign parameters for the system data type "IF_CONF_v4" for defining the IP address, as follows:

16	deviceIP	IF_CONF_v4	
17	Id	UInt	30
18	Length	UInt	18
19	Mode	UInt	1
20	InterfaceAddress	IP_V4	
21	ADDR	Array[1..4] of Byte	
22	ADDR[1]	Byte	192
23	ADDR[2]	Byte	168
24	ADDR[3]	Byte	0
25	ADDR[4]	Byte	3
26	SubnetMask	IP_V4	
27	ADDR	Array[1..4] of Byte	
28	ADDR[1]	Byte	255
29	ADDR[2]	Byte	255
30	ADDR[3]	Byte	255
31	ADDR[4]	Byte	0
32	DefaultRouter	IP_V4	
33	ADDR	Array[1..4] of Byte	
34	ADDR[1]	Byte	16#0
35	ADDR[2]	Byte	16#0
36	ADDR[3]	Byte	16#0
37	ADDR[4]	Byte	16#0

In the structure "configData": Assign parameters for the system data type "IF_CONF_NOS" for defining the PROFINET device name, as follows:

38	deviceName	IF_CONF_NOS	
39	Id	UInt	40
40	Length	UInt	11
41	Mode	UInt	1
42	NOS	Array[1..240] of Byte	
43	NOS[1]	Byte	'm'
44	NOS[2]	Byte	'y'
45	NOS[3]	Byte	'p'
46	NOS[4]	Byte	'l'
47	NOS[5]	Byte	'c'

Note

Structure of "configData"

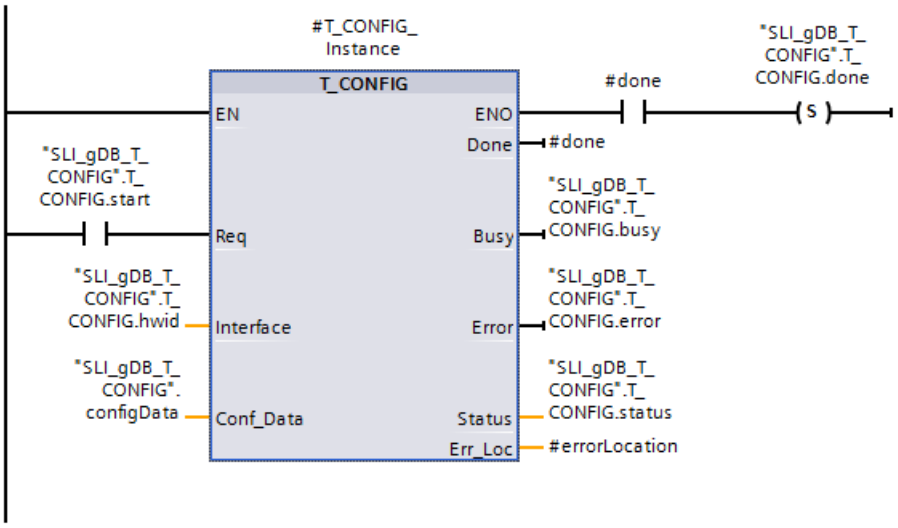
The information of the structure "configData" can be interpreted as follows:

- In the Header (IF_CONF_Header).
  SubfieldCount = 2: The value means: Two additional structures ("deviceIP", "deviceName") (*) are used below.
  *Make sure to maintain the order of both structures.
- For In the structure "deviceName" (system data type IF_CONF_NOS)
  - Lenght = 11 (*). The value corresponds to the overall length of the structure NOS (5 bytes for the device name"myplc" + 6 bytes for the parameters Id, Length and Mode)
    *Instead of an absolute length, you can use the default start value (Lenght = 0) for a dynamic length.
  - Mode = 1. The value causes a permanent change of the device name in "myplc".
  - NOS[1] ... NOS[5]. The NOS array contains the new device name (1 character / byte).

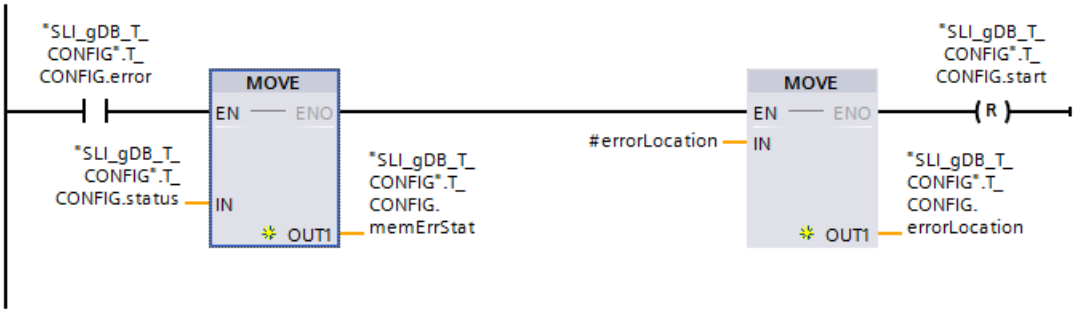
Create the FB "SLI_FB_T_CONFIG" Create the following local tags in it.

7	Static			
8	T_CONFIG_Instance	T_CONFIG		
9	done	Bool	false	Non-retain
10	errorLocation	DWord	16#0	Non-retain

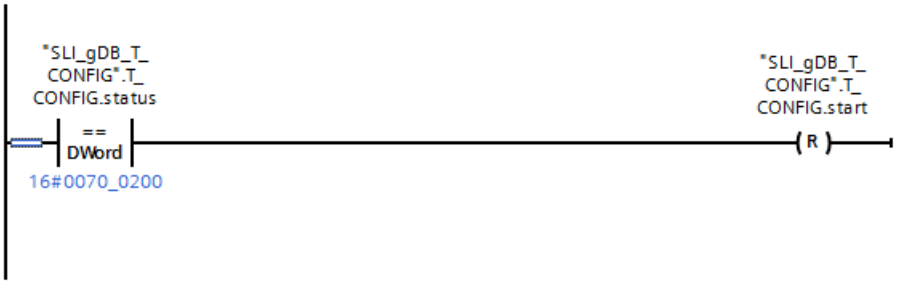
Network 1: Interconnect the parameter of the instruction "T_CONFIG" as follows:



Network 2: In case of error of T_CONFIG, save the status as follows.



Network 3: To exit T_CONFIG, create the following interconnections.



Result

Network 1:

When the input parameter REQ ("T_CONFIG.start") supplies the signal state "TRUE", the "T_CONFIG" instruction is started. The instruction "T_CONFIG" configures the integrated PROFINET interface of the CPU for multiple calls. Based on the input parameter INTERFACE ("T_CONFIG.hwid"), the "T_CONFIG" instruction knows the interface to be used.

Successful transfer of the data record () is indicated by the output parameter DONE ("#done") with the signal state "TRUE". At the same time the value "0000_0000" is displayed at the output

parameter STATUS ("T_CONFIG.status"). Because the values of the output parameters are only displayed for the moment of their validity, save the success message in the tag "T_CONFIG.done".

The output parameter ERROR ("T_CONFIG.error") or the tag "T_CONFIG.memErrStat" indicates that processing in the example is taking place without errors.

SLI_gDB_T_CONFIG

	Name	Data type	Start value	Monitor value
1	Static			
2	T_CONFIG	Struct		
3	start	Bool	false	FALSE
4	hwid	HW_INTERFACE	64	16#0040
5	busy	Bool	false	FALSE
6	done	Bool	false	TRUE
7	error	Bool	false	FALSE
8	errorLocation	DWord	16#0	16#0000_0000
9	status	DWord	16#0	16#0070_0000
10	memErrStat	DWord	16#0	16#0000_0000
11	configData	Struct		

Network 2:

If T_CONFIG signals an error ("T_CONFIG.error", "TRUE"), save the alarm as follows:

- Save the status ("T_CONFIG.status") in the "T_CONFIG.memErrStat" tag.
- Save the error location, output at the output parameter ERR_LOC ("#errorLocation") from T_CONFIG, in the tag "T_CONFIG.errorLocation".

Online & diagnostics

To check whether the PROFINET data were changed, open the path "Online & Diagnostics > Functions > Assign name" in the project tree.

Accessible devices in the network:

IP address	MAC address	Device	PROFINET device name
192.168.0.2	28-63-36-8D-BB-49	S7-1500	simaticplc
192.168.0.3	00-1B-1B-22-63-D0	S7-1500	myplc

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

See also

Description T_CONFIG (Page 4547)

Changes to the communications instructions (S7-1200, S7-1500)

Differences between the versions <= V3.x and >= V4.1 of the OUC library (S7-1200, S7-1500)

Introduction

CPU S7-1200 as of firmware version 4.1 supports new instruction versions for Open User Communication (OUC). The new instructions are included in the library with version 4.1.

If you want to use this library, changes must be made to the user program, since some of the new instructions for Open User Communication behave differently.

This section describes the differences in detail, especially in the call behavior of the instructions.

Note

Description only relevant when upgrading from the CPU S7-1200 ≤ V4.0 to S7-1200 ≥ V4.1

If you are using an S7-1500 CPU, changes between the library versions are not relevant. The same applies if you are using an S7-1200 ≥ Version 4.1 and do not convert the library for the Open User Communication to versvon <4.1.

Differences between the versions <= V3.x and >= V4.1 of the Open User Communication library

The following table shows the instruction that have differences between the versions <= V3.x and >= V4.1 of the Open User Communication library Click on the name of the instruction for detailed information.

Instruction	Version in library <= V3.x (CPU FW ≤ V4.0)	Version in library >= V4.1 (CPU FW ≥ V4.1)
TSEND_C (Page 4564)	V2.1	V3.0
TRCV_C (Page 4565)	V2.1	V3.0
TMAIL_C *	V2.1	V3.0
TCON (Page 4567)	V3.0	V4.0
TDISCON	V2.1	V2.1 (identical to library <= V3.x)
TSEND (Page 4569)	V3.0	V4.0
TRCV (Page 4571)	V3.0	V4.0
TUSEND (Page 4569)	V3.0	V4.0
TURCV (Page 4571)	V3.0	V4.0
T_RESET *	V1.1	V1.2
T_DIAG *	V1.1	V1.2
T_CONFIG	V1.0	V1.0 (identical to library <= V3.x)
* There are no differences in the versions that affect the user program.		

Note

Changing the Open User Communication library from a version <= V3.x to a version >= V4.1

When replacing the Open User Communication library from a version <= V3.x to a version >= V4.1 you also need to replace the Modbus TCP library. Afterwards check all the instructions relevant for your program.

See also

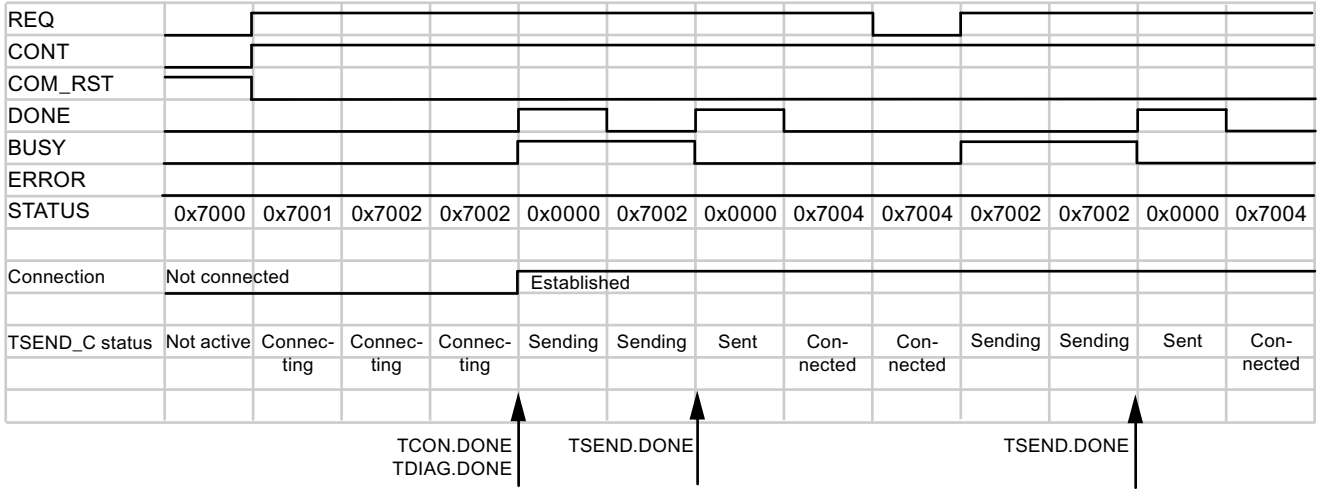
Differences between the versions <= V3.x and >= V4.0 of the Modbus TCP library (Page 4675)

Changes with the TSEND_C instruction (S7-1200, S7-1500)

Call behavior of TSEND_C (V<3.0)

Up to version 2.1 of the TSEND_C instruction, the DONE output parameter is set twice: Once when a connection is established by the internally used TCON instruction and then after a send operation by the internally used TSEND instruction.

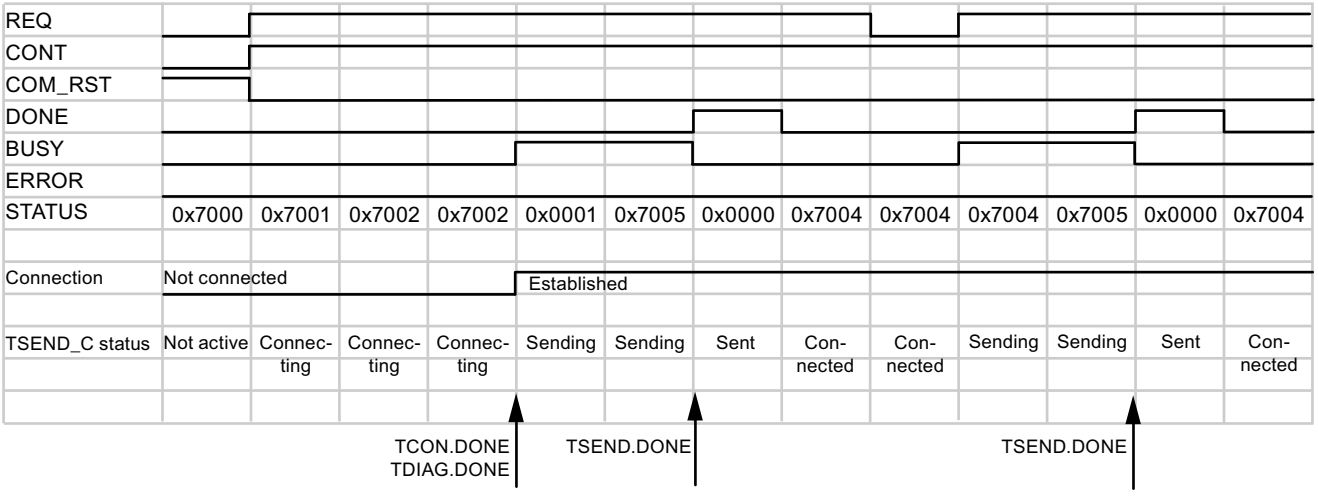
The following diagram shows the connection establishment and the sending of data with TSEND_C V2.1:



When setting the COM_RST parameter to "1", the current connection establishment or a current data transmission can be reset at any time. This terminates the existing communication connection and a new connection is established.

Call behavior of TSEND_C (V≥3.0)

As of the TSEND_C version, the DONE parameter is only set when data transfer has been completed by the internally used TSEND instruction (STATUS = 0000).



The existing connection is temporarily interrupted and reset when the COM_RST parameter is set to "1". However, unlike TSEND_C V2.1, the connection endpoint is retained.

Note

Additional protocols with TSEND_C as of version 3.0

Version 3.0 of the TSEND_C instruction also supports UDP and UDP Broadcast via the interface of the CPU and via CM/CP.

See also

TSEND_C: Send data via Ethernet (Page 4419)

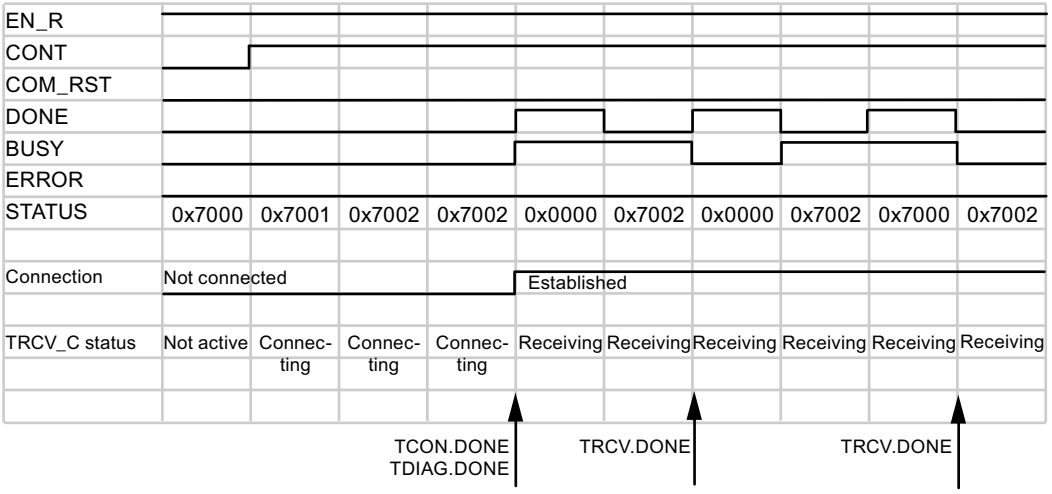
TSEND_C: Establishing a connection and sending data (Page 4423)

Changes with the TRCV_C instruction (S7-1200, S7-1500)

Call behavior of TRCV_C (V<3.0)

Up to version 2.1 of the TRCV_C instruction, the DONE output parameter is set after the connection is established. The STATUS output parameter does not distinguish whether the connection or the data transfer has been completed.

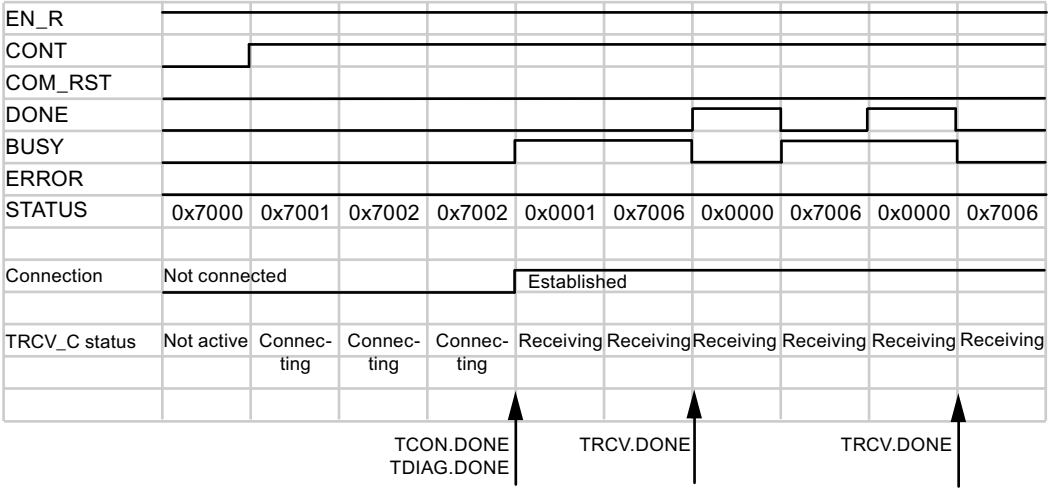
The following diagram shows the connection establishment and the sending of data with TRCV_C V2.1:



The current connection establishment or a current data transmission can be reset at any time by setting the COM_RST parameter to "1". This terminates the existing communication connection and a new connection is established.

Call behavior of TRCV_C (V≥3.0)

As of the TRCV_C version, the DONE parameter is only set when data transfer has been completed by the internally used TRCV instruction (STATUS = 0000). The completion of the connection establishment by the internally used TCON instruction is displayed by the output value 0x0001 at the STATUS parameter.



The existing connection is temporarily interrupted and reset when the COM_RST parameter is set to "1". However, unlike TSEND_C V2.1, the connection endpoint is retained.

Note

Additional protocols with TRCV_C as of version 3.0

Version 3.0 of the TRCV_C instruction also supports UDP and UDP Broadcast via the interface of the CPU and via CM/CP.

Note

ADHOC mode for the TCP protocol variant

In accordance with the TRCV instruction, with TRCV_C up to version 2.1, ADHOC mode is activated by assigning the value "0" to the LEN parameter. As of version 3.0 of the instruction, you use the ADHOC parameter for this. You can find detailed information in the descriptions of the instructions.

See also

Changes with the TRCV/TURCV instructions (Page 4571)

TRCV_C: Receive data via Ethernet (Page 4432)

TRCV_C: Establishing a connection and receiving data (Page 4437)

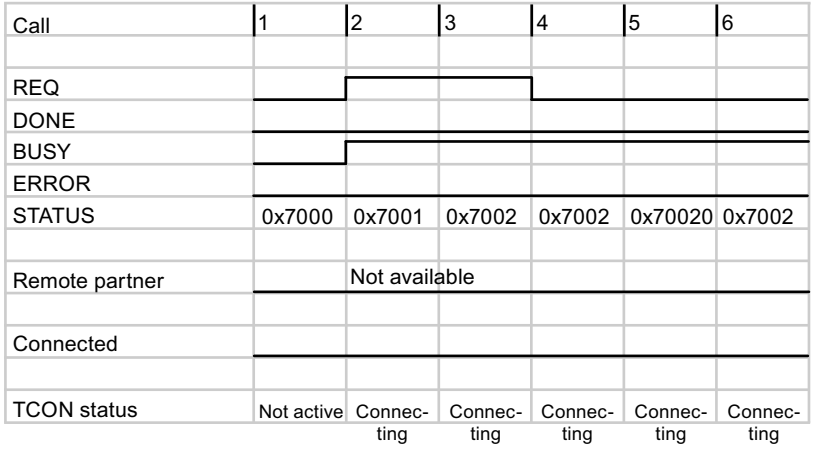
Changes with the TCON instruction (S7-1200, S7-1500)

Changed call behavior with connection errors

Previous call behavior of TCON (V<4.0)

- The previous TCON instruction (V<4.0) is activated with a rising edge at the REQ input parameter.
- If the remote communication partner cannot be reached, the instruction sets the BUSY output parameter.
- An error message is not output.

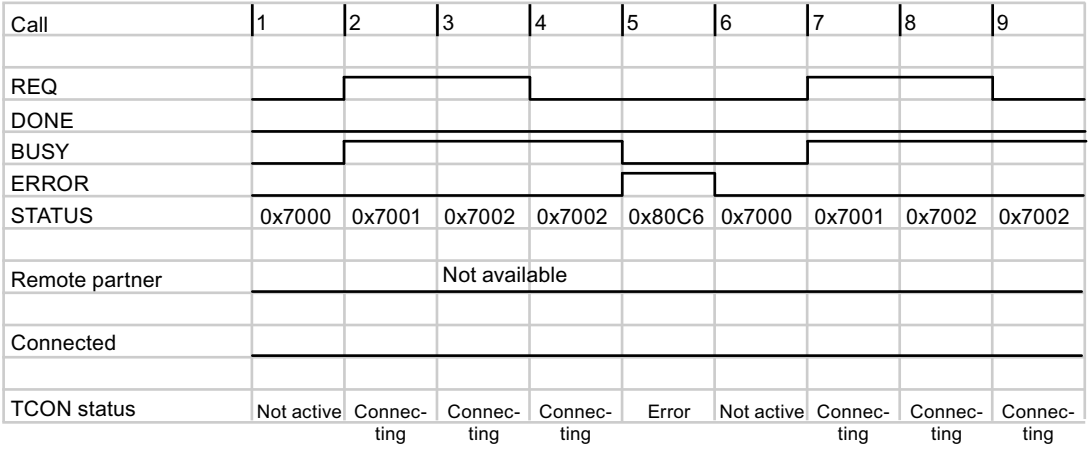
The following diagram shows the behavior of TCON on the part of the active connection partner. If no connection is made, the instruction is not called again.



New call behavior of TCON (V≥4.0)

- The new TCON instruction is also activated with a rising edge at the REQ input parameter.
- Unlike before, an error message is output when the remote communication partner cannot be reached. The error can be queried and the call started again by a new edge at the REQ parameter.
- The following changes result for the user program:
  - Additional error messages of TCON that can be evaluated (see description of TCON (Page 4478)).
  - The connection establishment can be re-initialized via a new rising edge. To do this, query the DONE and ERROR parameters to ensure that an error is present.

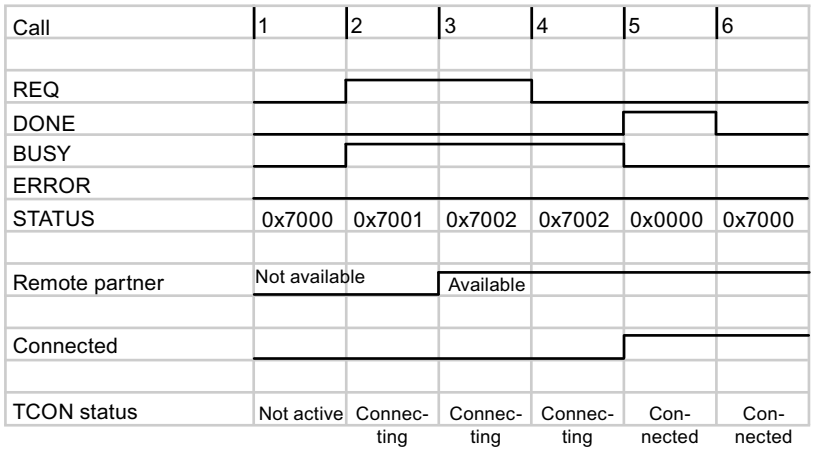
The following diagram shows the behavior of TCON on the part of the active connection partner. The instruction is called again after a network error (error code 80C6).



Call behavior without communication errors

If there is no connection error, the TCON versions have the same behavior.

The following diagram shows the behavior of TCON on the part of the active connection partner.



See also

TCON: Establish communication connection (Page 4474)

Changes with the TSEND/TUSEND instructions (S7-1200, S7-1500)

Send operation with an established connection

If no delay or interruption occurs during sending, versions 3.0 and 4.0 of the TSEND/TUSEND instructions behave the same way:

- The instruction is started with a positive edge at the "REQ" parameter. The instruction is executed asynchronously, i.e. it must be called until the full execution is indicated by the DONE parameter.
- A maximum of 8 KB of data can be sent with TSEND. A maximum of 1472 bytes can be sent with TUSEND. The amounts are based on the full execution of the instruction, i.e. all required calls until the DONE parameter is set.
- Data transmission takes place in three steps:
  - The data is written from the operand area to an internal buffer (indicated by STATUS=7001).
  - The transmission to the remote communication partner follows.
  - If the transmission is successful, DONE is set to "1" and STATUS is reset to "0" (see call 5 in the diagram below).

Call	1	2	3	4	5	6
REQ						
DONE						
BUSY						
ERROR						
STATUS	0x7000	0x7001	0x7002	0x7002	0x0000	0x7000
LEN (max.)	8 KB	8 KB	Internal			
Connected						
TSEND status	Not active	Sending	Sending	Sending	Sent	Not active

Send operation with time-delayed call

If the send operation is not performed via a communication module (CM) or a communication processor (CP), versions 3.0 and 4.0 of the TSEND/TUSEND instructions behave the same way:

- In the following example, TSEND or TUSEND are only called once with a rising edge at the REQ parameter (see call 2 in the diagram below).
- If the time delay is sufficiently long and the data could be transferred, the DONE parameter is set directly to "1" in the next call (call 3 here).

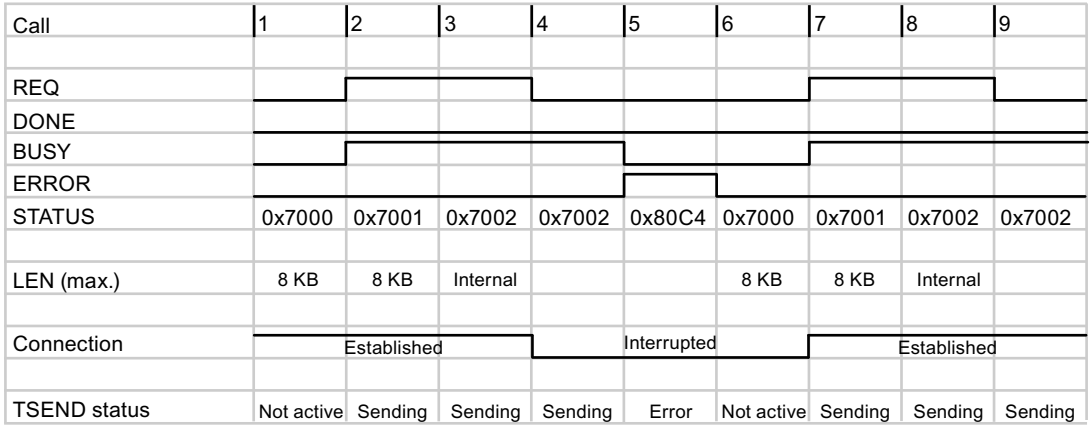
Call	1	2 → Time delay / No call		3	4
REQ					
DONE					
BUSY					
ERROR					
STATUS	0x7000	0x7001		0x0000	0x7000
LEN (max.)	8 KB	8 KB	Internal		
Connected					
TSEND status	Not active	Sending	Sending	Sent	Not active

If the send operation is performed via a CM/CP, version 4.0 of the TSEND/TUSEND instructions behaves differently. In this case, the instruction must be called multiple times until the receipt of data is confirmed by the NDR parameter of the TRCV/TURCV instructions.

Send operation with interrupted connection

If the connection is interrupted during the send operation, the ERROR and STATUS parameters indicate the error and its cause. In the following example, the connection was interrupted during call number 5 and this error is indicated accordingly with the STATUS parameter.

As of version 4.0 of the TSEND/TUSEND instructions, more STATUS alarms are available that can be correspondingly evaluated (see description of TSEND (Page 4490)/TUSEND (Page 4516)) .



See also

TSEND: Send data via communication connection (Page 4486)

Changes with the TRCV/TURCV instructions (S7-1200, S7-1500)

Receive operation with an established connection

If no delay or interruption occurs during sending, version 3.0 and 4.0 of the TRCV/TURCV instructions behave the same way:

- A maximum of 8 KB of data can be sent with TRCV. A maximum of 1472 bytes can be received with TURCV. The amounts are based on the full execution of the instruction, i.e. all required calls until the DONE parameter is set.
- The instruction receives data if the EN_R parameter is set to "1".
- The data reception is not complete until the length of data specified at the the LEN parameter has been completely received. The length of the received data is output at the RCVD_LEN output parameter. Only then does the data in the area defined at the DATA parameter become available.

Call	1	2	3	4	5	6
EN_R						
LEN (max.)	8 KB	8 KB	Internal			
NDR						
BUSY						
ERROR						
STATUS	0x7000	0x7001	0x7002	0x7002	0x0000	0x7000
RCVD_LEN	0	0	0	0	max. 8 KB	0
Connection	Established					
TRCV status	Not active	Receiving	Receiving	Receiving	Receiving	Not active

Receive operation with time-delayed call

If the send operation is not performed via a communication module (CM) or a communication processor (CP), versions 3.0 and 4.0 of the TRCV/TURCV instructions behave the same way:

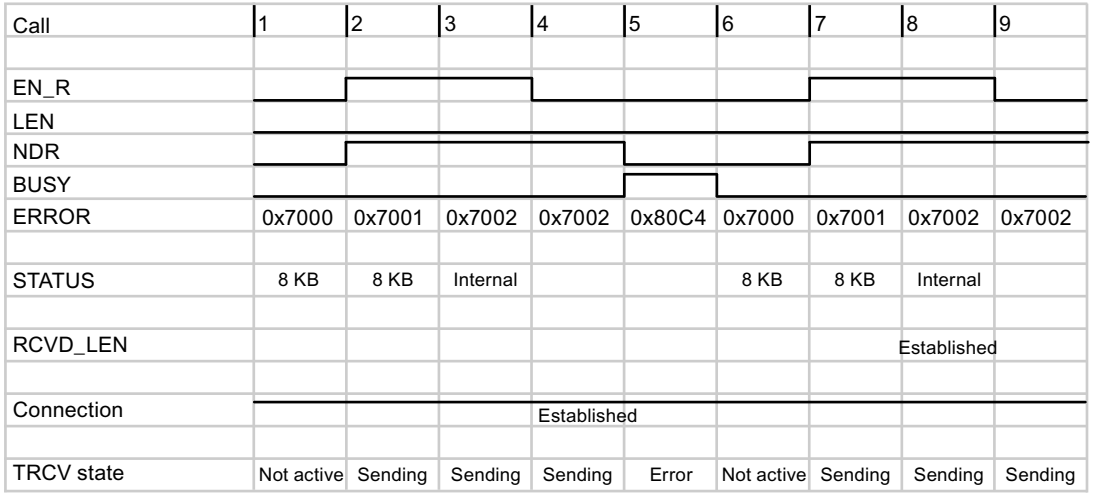
- In the following example, TRCV or TURCV are only called once with a rising edge at the EN_R parameter (see call 2 in the diagram below).
- If the time delay is sufficiently long and the data could be transferred, the NDR parameter is set directly to "1" in the next call (call 3 here).

Call	1	2	Delay No call		3	4
EN_R						
LEN (max.)	8 KB	8 KB	Internal			
NDR						
BUSY						
ERROR						
STATUS	0x7000	0x7001			0x0000	0x7000
RCVD_LEN	0	0	0	0	max. 8 kB	0
Connection	Established					
TRCV status	Not active	Receiving	Receiving	Receiving	Receiving	Not active

If the send operation is performed via a CM/CP, version 4.0 of the TSEND/TUSEND instructions behaves differently. In this case, the instruction must be called multiple times until the receipt of data is confirmed by the NDR parameter.

Receive operation with interrupted connection

If the connection is interrupted during receiving, the ERROR and STATUS parameters indicate the error and its cause. In the following example, call number 5 resulted in a temporary communication error, which is accordingly indicated by the STATUS parameter.



As of version 4.0 of the TRCV/TURCV instructions, more STATUS alarms are available that can be correspondingly evaluated (see description of TRCV (Page 4498)/TURCV (Page 4520)).

Receive operation using ADHOC mode

ADHOC mode is only available with the TCP protocol variant. You can use ADHOC mode to receive data of variable length with the TRCV/TURCV instruction. If ADHOC mode is active, the receipt of data is confirmed at the NDR parameter when at least one byte has been transferred.

Data reception in ADHOC mode with TRCV < 3.0 (S7-1200 < V4.0)

In the older version of TRCV (Page 4493), ADHOC mode was activated by setting the LEN parameter to "0". In the following example, 10 bytes of data are transferred with call 5.

Call	1	2	3	4	5	6
EN_R						
LEN (max.)	0 = Activation of the ADHOC mode					
NDR						
BUSY						
ERROR						
STATUS	0x7000	0x7001	0x7002	0x7002	0x0000	0x7000
RCVD_LEN	0	0	0	0	10 bytes	0
Connection	Established					
TRCV status	Not active	Receiving	Receiving	Receiving	Receiving	Not active

Data receipt in ADHOC mode with TRCV ≥ 3.0 (S7-1200 ≥ V4.0 or S7-1500)

As of version 3.0 of the TRCV instruction, ADHOC mode is activated with its own parameter (ADHOC).

Call	1	2	3	4	5	6
EN_R						
LEN	Max. 8 KB	Max. 8 KB	Internal			
ADHOC						
NDR						
BUSY						
ERROR						
STATUS	0x7000	0x7001	0x7002	0x7002	0x0000	0x7000
RCVD_LEN	0	0	0	0	10 bytes	0
Connection	Established					
TRCV status	Not active	Receiving	Receiving	Receiving	Receiving	Not active

If an error occurs during execution, as of version 4.0 of the TRCV instructions, there are more STATUS alarms available which can be evaluated accordingly.

If the send operation is performed via a CM/CP, version 4.0 of the TRCV/TURCV instructions behaves differently. In this case, the instruction must be called multiple times until the receipt of data is confirmed by the NDR parameter.

11.4.6.3 Web server (S7-1200, S7-1500)

WWW: Synchronizing user pages (S7-1200, S7-1500)

WWW: Synchronizing user pages (S7-1200, S7-1500)

Description

The instruction WWW initializes the web server of the CPU or synchronizes user-defined web pages, in short: "user pages", with the user program in the CPU.

Together with the Web server, user web pages make it possible for the CPU to access freely designed web pages of the CPU with a web browser.

Use script instructions (such as Javascript) and HTML code in user web pages to transfer data via a web browser for further processing to the CPU and to display data from the operand area of the CPU in the web browser. To synchronize the user program and the Web server, but also to initialize, call the instruction "WWW" in the user program.

Initialization

User web pages are "packaged" in data blocks for processing by the CPU. You will have to generate appropriate data blocks from the source files (HTML files, screens, Javascript files, ...) during configuration. The Web Control DB takes on a special role (default: DB 333). It contains status and control information as well as links to additional data blocks with coded web pages. The data blocks with coded web pages are called fragment DBs.

When the data blocks are downloaded into the CPU, the CPU does not "know" that user web pages are coded inside it. The instruction "WWW" in the startup OB, for example, will inform the CPU which DB is the Web Control DB. After this initialization, the user pages are accessible via a web browser.

Synchronization

If you want the user program to interact with the user pages, then the instruction "WWW" must be used in the cyclical program part.

Examples of interaction between user program and web page:

- Check received data
- Assemble and send back data to the web browser making the request

In this case it must be possible to evaluate the current status information and the Web server must receive control information, such as release of a web page requested by a web browser.

Parameter

The following table shows the parameters of the "WWW" instruction:

Parameter	Declaration	Data type	Memory area	Description
CTRL_DB	Input	DB_WWW	I, Q, M, D, L or constant	Data block that describes the user pages (Web control DB)
RET_VAL	Output	INT	I, Q, M, D, L	Error information

For additional information on valid data types, refer to Overview of the valid data types (Page 2239).

Parameter RET_VAL

Error code (W#16#...)	Explanation
0000	No error occurred. There are no web page requests that have to be released by the user program.
00xy	x: indicates whether an error has occurred during initialization of the Web Control DB (CTRL_DB): x=0: No errors occurred. x=1: Error occurred. The error is coded in the byte "CTRL_DB.last_error" of the Web Control DB, see description of Web Control DB. y: Number of the pending request. Several requests are possible (e. g. requests "0" and "1" are pending: y="3". y="1": Request "0" y="2": Request "1" y="4": Request "2" y="8": Request "3"
803A	The specified Web Control DB does not exist on the CPU.
8081	Incorrect version or incorrect format of the Web Control DB.
80C1	There are no resources to initialize the web application, for example, because only two or four web applications may be running.

Example

You will find the example here: Program example of WWW (Page 4577).

You can find additional information and the program code for the example here: Sample Library for Instructions (Page 2568).

See also

Overview of the valid data types (Page 2239)

Program example of WWW (S7-1200, S7-1500)

Introduction

In the following example, you synchronize a user-defined Website with the program example in a CPU S7-1500 and test the program example via a Web server.

Requirement

Create nineteen tags in a global data block for storing the data.

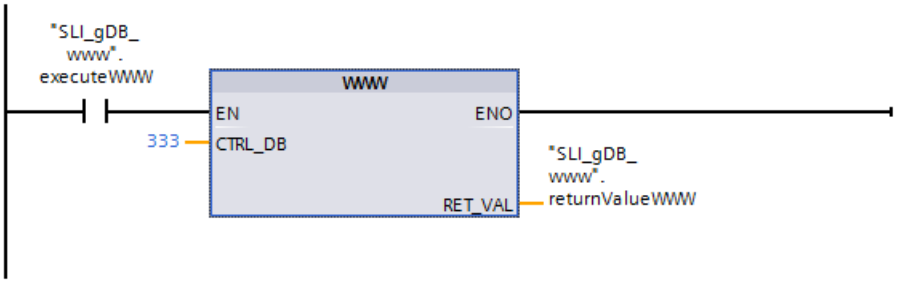
SLI_gDB_www

	Name	Data type	Start value
1	Static		
2	executeWWW	Bool	false
3	returnValueWWW	Word	16#0
4	executeSetOB	Bool	false
5	setDone	Bool	false
6	start	Bool	false
7	stop	Bool	false
8	engineStarted	Bool	false
9	reset	Bool	false
10	tempReset	Bool	false
11	openValve	Bool	false
12	closeValve	Bool	false
13	valveOutput	Bool	false
14	flowrate	Int	1
15	tankLevel	Int	0
16	alarm	Int	0
17	tankLevelOverflow	Int	160
18	tankLevelMax	Int	150
19	tankLevelMidth	Int	75
20	tankLevelMin	Int	5

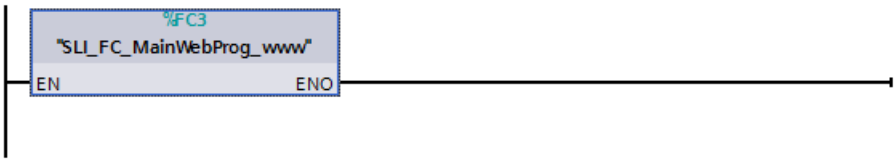
Interconnect parameters - in FB “SLI_FB_www“

You create the following interconnections in an FB "SLI_FB_www": You call the FB in an OB1.

Network 1: Interconnect the parameters of the "WWW" instruction as follows.



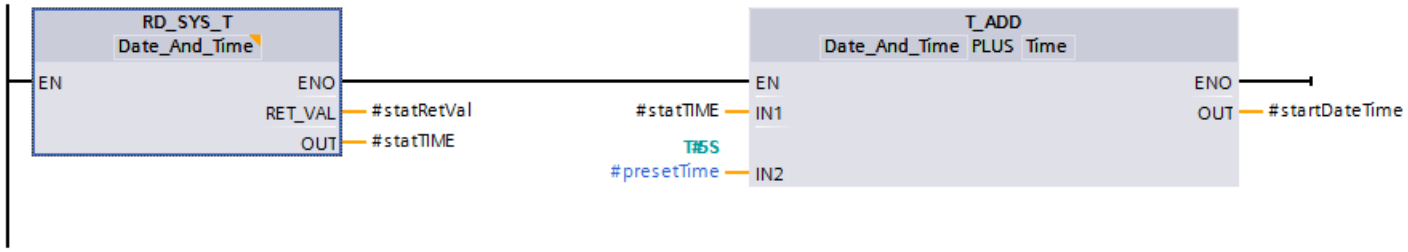
Network 2: You create the program for the Website in the FC “SLI_FC_MainWebProg_www“. You then call the FC “SLI_FC_MainWebProg_www“ in network 2 of the FB.



Interconnecting parameters - in FC "SLI_FC_SetTOD_www"

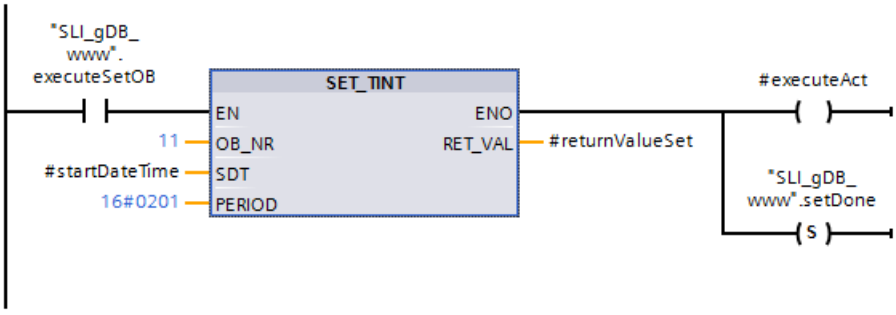
In an FC “SLI_FC_SetTOD_www“ you create a program for using a time-of-day interrupt OB (OB11). You create the following interconnections:

Network 1: In the FC you interconnect the following local tags with the instructions “RD_SYS_T“ and “T_ADD“.

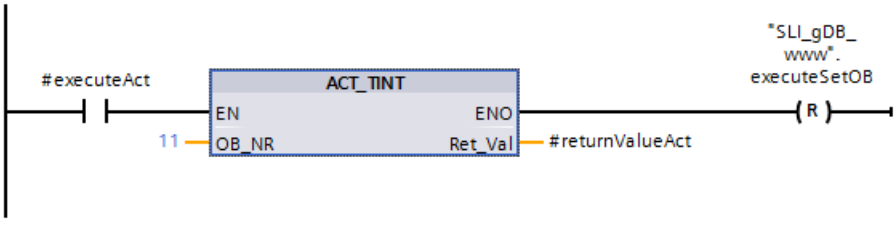


Network 2: This is followed by interconnecting the tags for setting the time-of-day interrupt OB.

The local tag “#startDateTime“ (data type “Date_And_Time”) transfers the start time to the time-of-day interrupt OB.

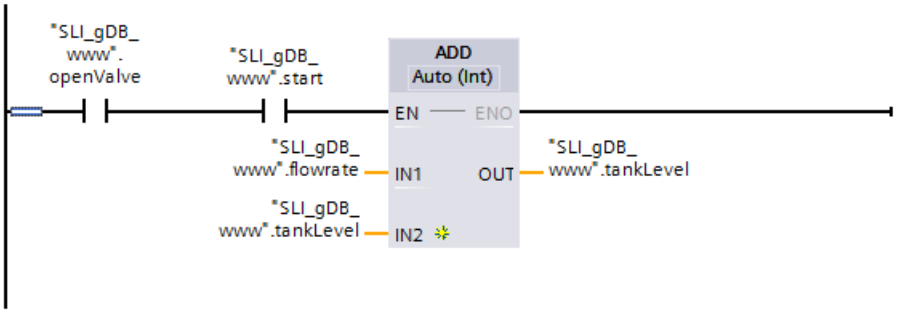


Network 3: This is followed by interconnecting the tags for activating the time-of-day interrupt OB.



Interconnecting parameters - in the OB “SLI_todOB_www“

In the time-of-day interrupt OB “SLI_todOB_www“ (OB11) you create the following interconnections for cyclic increasing of the tag “tankLevel“.

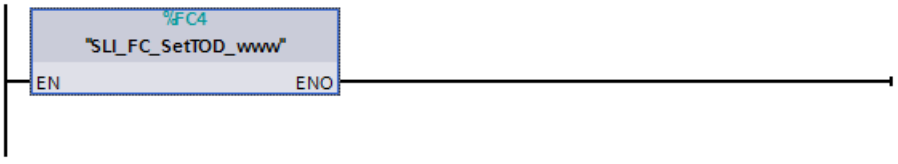


Interconnecting parameters - in FC "SLI_FC_MainWebProg_www"

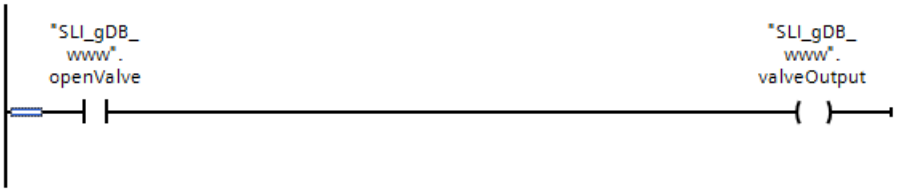
You create the following interconnections in the FC "SLI_FC_MainWebProg_www“.

The program “SLI_FC_MainWebProg_www“ allows control of a motor, a valve and the tank filling. In addition to this the output of various interrupt statuses is made possible. The program is controlled via the Web server.

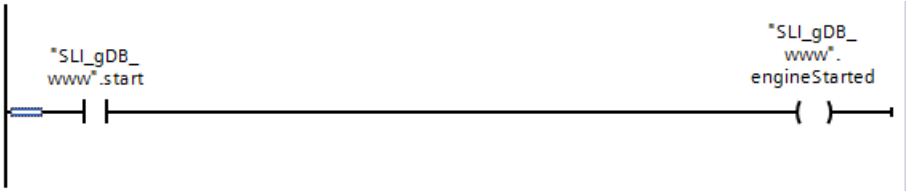
Network 1: You call the FC “SLI_FC_SetTOD_www“ in the FC “SLI_FC_MainWebProg_www“.



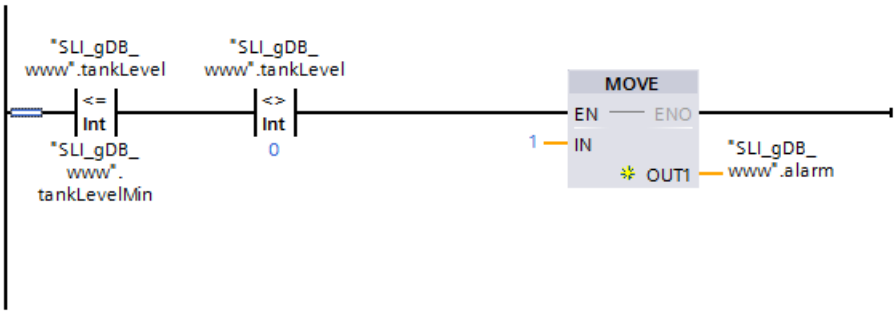
Network 2: This is followed by interconnecting the tags for opening the valve.



Network 3: This is followed by interconnecting the tags for starting the motor.

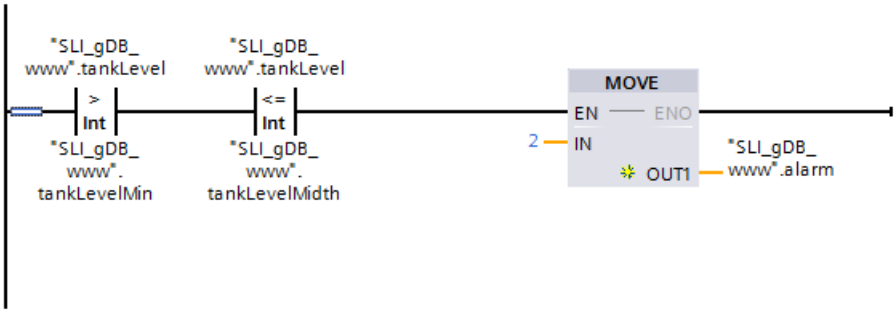


Network 4: This is followed by interconnecting the tags for querying the interrupt status “1“.

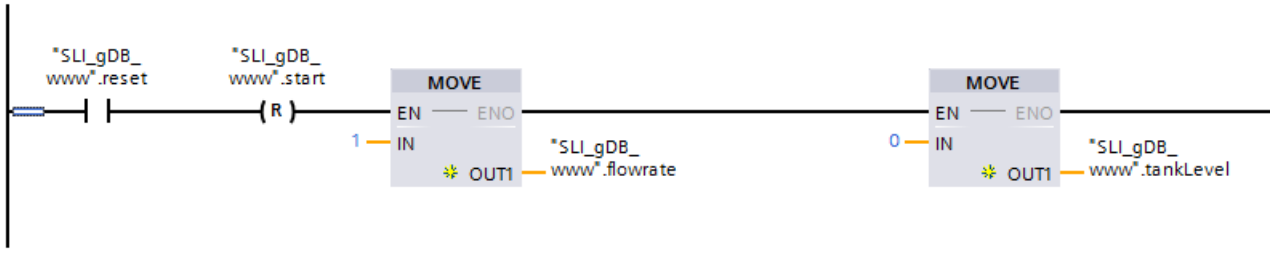


Networks 5 to 8: This is followed by interconnecting the tags for querying the interrupt status "2".

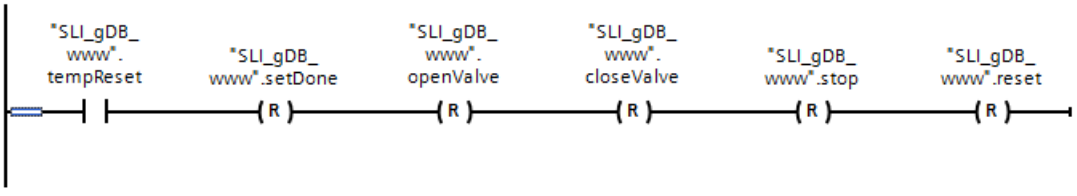
Interconnect the tags for querying the other interrupt statuses in the same way.



Network 9: This is followed by interconnecting the tags for resetting the the motor and the filling of the tank..



Network 10: This is followed by interconnecting the tags for resetting all the other values (BOOL).



Setting CPU properties

Make the following settings in the CPU properties:

- Activate web server access under "... > PROFINET interface > Access to the web server".
- In “CPU properties > Web server enable the Web server for the module,
- In the section “User administration“ create a new user with read and write rights.
- Make the following settings in the section "User pages":
  - You set the “HTML file path”.

    Note

    The storage path should be the same as the path of the Sample Library for Instructions (SLI). The folder name is “SLI_html“. For example: “C:\TIA_library\SLI_html“.

  - Enter the text "index.html" for the Start HTML page.
  - For the application you enter the name “WWW sample“.
  - Click the "Create block" button. The Web DB (333) and the Fragment DB (334) are then created.

Result for WWW

If the normally open contact ("executeWWW") supplies the signal state "TRUE", the "WWW" instruction is executed. The number of the data block that describes user defined Website is stored via the the input parameter CTRL_DB ("333"). The “WWW“ instruction initializes the user defined Website in the Web server of the CPU and immediately synchronizes the program example.

The output parameter RET_VAL ("returnValueWWW") indicates that the processing is running without errors.

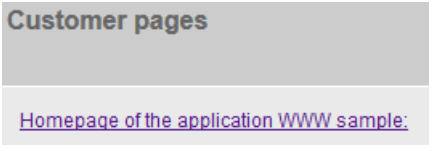
SLI_gDB_www

	Name	Data type	Start value	Monitor value
1	Static			
2	executeWWW	Bool	false	TRUE
3	returnValueWWW	Word	16#0	16#0000
4	executeSetOB	Bool	false	FALSE
5	setDone	Bool	false	TRUE
6	start	Bool	false	TRUE
7	stop	Bool	false	FALSE
8	engineStarted	Bool	false	TRUE
9	reset	Bool	false	FALSE
10	tempReset	Bool	false	FALSE
11	openValve	Bool	false	TRUE
12	closeValve	Bool	false	FALSE
13	valveOutput	Bool	false	TRUE
14	flowrate	Int	1	3
15	tankLevel	Int	0	3
16	alarm	Int	0	1
17	tankLevelOverflow	Int	160	160
18	tankLevelMax	Int	150	150
19	tankLevelMidth	Int	75	75
20	tankLevelMin	Int	5	5

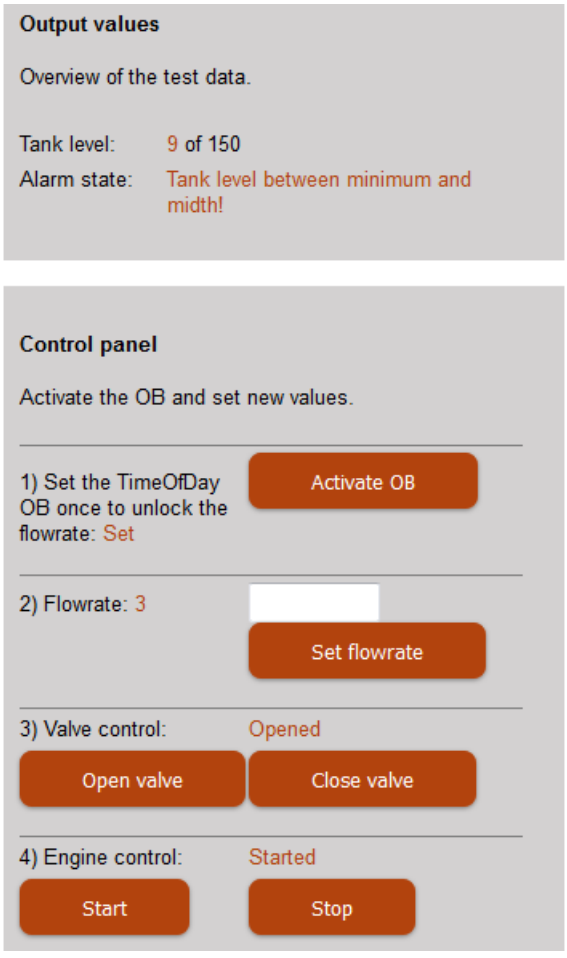
Result for the Web server

You call the Web server via the “Internet browser > IP of the Web server”. You log in with your user name user name and -password.

In the Web server in “User pages” a link to the website defined by the user is displayed.



On the Website of the program example it is possible for you to output tags and transfer new values to the tags.



Functions used in the HTML document

Below you will see excerpts of the functions used in the program example with which the tags are transferred to the source code in the HTML document. For each function a tag is implemented via a call in the HTML document and at the relevant location in the HTML document is used with a further call.

Function	To call in the HTML document	To output in the HTML element
Data output:	`<!-- AWP_In_Variable Name='"SLI_gDB_www".tankLevel' -->`	`:="SLI_gDB_www".tankLevel:`
Enumeration (value replacement):	`<!-- AWP_Enum_Def Name="OpValvValue" Values='0:"Closed", 1:"Opened"' -->`	`<!-- AWP_Enum_Ref Name='"SLI_gDB_www".valveOutput' Enum="OpValvValue" -->:="SLI_gDB_www".valveOutput:`

Function	To call in the HTML document	To output in the HTML element
Value change (from INT):	`<!-- AWP_In_Variable Name='"SLI_gDB_www".flowrate' -->`	`<form method="post" action="" onsubmit="return check();"> <input type="text" name='"SLI_gDB_www".flowrate' size="10px"/> <input class="button1" type="submit" value="Set flowrate"/> </form>`
Value change (from BOOL):	`<!-- AWP_Enum_Def Name="ResetValue" Values='0:"Off", 1:"On"' -->`	`<form method="post" action=""> <input class="button1" type="submit" value="Reset"/> <input type="hidden" name='"SLI_gDB_www".reset' size="34px" value="1"/> </form>`

Program code

You can find additional information and the program code for the above-named example here: Sample Library for Instructions (Page 2568).

11.4.6.4 Communications processor (S7-1200, S7-1500)

SIMATIC NET CM/CP (S7-1200, S7-1500)

Point-to-point (S7-1200)

PORT_CFG: Configure communication parameters dynamically (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The instruction "PORT_CFG" allows dynamic configuration of communications parameters for a point-to-point communications port.

You set up the original static configuration of the port in the hardware configuration. You can change this configuration by executing the "PORT_CFG" instruction. You can also use this

function to save created blocks in libraries and to avoid configuration in the hardware configuration when you reuse it.

With "PORT_CFG" you can influence the following communications parameter settings:

- Parity
- Baud rate
- Number of bits per character
- Number of stop bits
- Type and properties of flow control

The changes made by the "PORT_CFG" instruction are not stored permanently on the target system.

You can transfer serial data via the electrical connections RS-232 (half and full duplex) and RS-485 (half duplex).

Parameters

The following table shows the parameters of the "PORT_CFG" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Activates the configuration change on a rising edge
PORT	Input	PORT	I, Q, M, D, L or constant	Identification of the communication port (HW-ID)
PROTOCOL	Input	UINT	I, Q, M, D, L or constant	Transmission protocol: • 0: Point-to-point communication protocol • 1..n: Future definition for specific transmission protocols
BAUD	Input	UINT	I, Q, M, D, L or constant	Baud rate of the port: • 1: 300 baud • 2: 600 baud • 3: 1200 baud • 4: 2400 baud • 5: 4800 baud • 6: 9600 baud (default) • 7: 19200 baud • 8: 38400 baud • 9: 57600 baud • 10: 76800 baud • 11: 115200 baud

Parameter	Declaration	Data type	Memory area	Description
PARITY	Input	UINT	I, Q, M, D, L or constant	Parity of the port: • 1: No parity (default) • 2: Even parity • 3: Odd parity • 4: Mark parity • 5: Space parity
DATABITS	Input	UINT	I, Q, M, D, L or constant	Bits per character: • 1: 8 bits per character (default) • 2: 7 bits per character
STOPBITS	Input	UINT	I, Q, M, D, L or constant	Number of stop bits: • 1: 1 stop bit (default) • 2: 2 stop bits
FLOWCTRL	Input	UINT	I, Q, M, D, L or constant	Data flow control: • 1: None (default) • 2: XON/XOFF • 3: Hardware flow control (RTS always activated) • 4: Hardware flow control (RTS can be deactivated during transmission)
XONCHAR	Input	CHAR	I, Q, M, D, L or constant	Indicates the character used as XON character. The character DC1 (11H) is set as default.
XOFFCHAR	Input	CHAR	I, Q, M, D, L or constant	Indicates the character used as XOFF character. The character DC3 (13H) is set as default.
WAITTIME	Input	UINT	I, Q, M, D, L or constant	Specifies the wait time for XON or CTS after the start of the transmission. The specified value must be greater than 0. 2000 milliseconds are set as default.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
80A0	The specified protocol is invalid.
80A1	The specified baud rate is invalid.

Error code* (W#16#...)	Description
80A2	The specified parity rate is invalid.
80A3	The specified number of bits per character is invalid.
80A4	The specified number of stop bits is invalid.
80A5	The specified type of flow control is invalid.
80A6	Incorrect value at the WAITTIME parameter When the data flow control is enabled, the value at the WAITTIME parameter must be greater than zero.
80A7	Invalid values at XONCHAR and XOFFCHAR parameters.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

You will find more detailed information on general error codes of the communication instructions in “General status information of the communications blocks (Page 4604)“.

SEND_CFG: Configure serial transmission parameters dynamically (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The instruction "SEND_CFG" allows dynamic configuration of serial transmission parameters for a point-to-point communications port. All the messages waiting for transfer are discarded after execution of SEND_CFG.

You set up the original static configuration of the port in the hardware configuration. You can change this configuration by executing the "SEND_CFG" instruction. You can also use this function to save created blocks in libraries and to avoid configuration in the hardware configuration when you reuse it. With "SEND_CFG" you can influence the following transmission parameter settings:

- Time between the activation of RTS (Request to Send) and the start of the transmission
- Time between the end of transmission and the deactivation of RTS
- Define bit times for breaks

The changes made by the "SEND_CFG" instruction are not stored permanently on the target system.

You can transfer serial data via the electrical connections RS-232 (half and full duplex) and RS-485 (half duplex).

Parameters

The following table shows the parameters of the "SEND_CFG" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Activates the configuration change on a rising edge
PORT	Input	PORT	I, Q, M, D, L or constant	Identification of the communication port (HW-ID)
RTSONDLY	Input	UINT	I, Q, M, D, L or constant	The time that should elapse after activating RTS until the start of transmission. Valid values for this parameter are as follows: • 0 (default) • 0 to 65535 ms in steps of 1 ms This parameter does not apply to RS-485 modules.
RTSOFFDLY	Input	UINT	I, Q, M, D, L or constant	Time that should elapse after the end of transmission until deactivation of RTS. Valid values for this parameter are as follows: • 0 (default) • 0 to 65535 ms in steps of 1 ms This parameter does not apply to RS-485 modules.
BREAK	Input	UINT	I, Q, M, D, L or constant	Specifies the bit times for a break, which are sent at the start of the message. 12 bit times are set as default. A maximum of 25000 bit times can be specified.
IDLELINE	Input	UINT	I, Q, M, D, L or constant	Specifies the bit times for idle line after the break at the start of the message. 12 bit times are set as default. A maximum of 25000 bit times can be specified.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
80B0	The configuration of a transmission interruption is not permitted.
80B1	The specified break time exceeds the permitted maximum of 25000 bit times.

Error code* (W#16#...)	Description
80B2	The specified time for idle line exceeds the permitted maximum of 25000 bit times.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

You will find more detailed information on general error codes of the communication instructions in “General status information of the communications blocks (Page 4604)“.

RCV_CFG: Configure serial receive parameters dynamically (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The instruction "RCV_CFG" allows dynamic configuration of serial receive parameters for a point-to-point communications port. You can use this instruction to configure the conditions that specify the start and end of the message to be transmitted. The receipt of messages that correspond to these conditions can be enabled by the "RCV_PTP (Page 4599)" instruction.

You set up the original static configuration of the port in the properties of the hardware configuration. Execute the "RCV_CFG" instruction in your program to change the configuration. You can also use this function to save created blocks in libraries and to avoid configuration in the hardware configuration when you reuse it. The changes made by the "RCV_CFG" instruction are not stored permanently on the target system.

All the messages waiting for transfer are discarded after execution of the "RCV_CFG" instruction.

Parameters

The following table shows the parameters of the "RCV_CFG" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Activates the configuration change on a rising edge.
PORT	Input	PORT	I, Q, M, D, L or constant	Identification of the communication port (HW-ID)
CONDITIONS	Input	CONDITIONS	D, L	Data structure defining the conditions for start and end of data transmission.
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred.
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Data type CONDITIONS

You can use the CONDITIONS structure to define the start and end conditions for the message transmission. The structure CONDITIONS is included in the instance DB of the "RCV_CFG" instruction. Use the structure CONDITIONS to define the start and end conditions when the transmission of a message is complete and when the next message transfer is to start:

- You define the start conditions for the data transfer in the START structure.
- You define the end conditions for the data transfer in the END structure.

You can define one or more start and end conditions for this. If you specify multiple start or end conditions these are linked by an OR logic instruction.

The following table shows the "CONDITIONS" structure:

Parameters		Data type	Description
START		STRUCT	Start conditions
	STARTCOND	UINT	Specifies the start condition (details, see below). The start condition can be specified as a 16-bit hexadecimal value. Possible values for the start condition are: • 1: Start character • 2: Any character (default) • 4: Line break • 8: Idle line • 16: Character string 1 • 32: Character string 2 • 64: Character string 3 • 128: Character string 4 Multiple start conditions can also be defined at the STARTCOND parameter. The total from the values of the individual conditions is specified for this. If, for example, you want to define "Idle line" OR "Character string 1" OR "Character string 4" as start condition, the value "152" must be specified.
	IDLETIME	UINT	Specifies the maximum idle time of the line before receipt is started. Valid values for this parameter are as follows: • 40 bit times (default) • 0 to 2500 bit times
	STARTCHAR	BYTE	Specifies the start character. This setting is only enabled when the configured start condition is "Start character". Valid values for this parameter are as follows: • 02 (STX): Default setting • B#16#00 to B#16#FF
	SEQ[1].CTL	BYTE	Character string 1: Control sequence for each character You can use the bit position of the character to define which characters of the character string will be considered or ignored. To evaluate the characters, the corresponding bits have to be set. • Bit 0: 1 character • Bit 1: 2 characters • Bit 2: 3 characters • Bit 3: 4 characters • Bit 4: 5 characters A character is ignored when the corresponding bit is reset.
	SEQ[1].STR	CHAR[5]	Character string 1: Start character (5 characters)
	SEQ[2].CTL	BYTE	Character string 2: Ignore/compare control sequence for each character
	SEQ[2].STR	CHAR[5]	Character string 2: Start character (5 characters)
	SEQ[3].CTL	BYTE	Character string 3: Ignore/compare control sequence for each character
	SEQ[3].STR	CHAR[5]	Character string 3: Start character (5 characters)
	SEQ[4].CTL	BYTE	Character string 4: Ignore/compare control sequence for each character
	SEQ[4].STR	CHAR[5]	Character string 4: Start character (5 characters)

Parameters	Data type	Description
END	STRUCT	End conditions

Parameters		Data type	Description
	ENDCOND	UINT	Specifies the end condition (details, see below). The end condition can be specified as a 16-bit hexadecimal value. Possible values for the end condition are: • 1: Reply timeout • 2: Message timeout • 4: Timeout within the character string • 8: Maximum length • 16: N+LEN+M; the information on the message length in integrated in the message and will be evaluated. • 32: Character string 1 Multiple end conditions can also be defined at the ENDCOND parameter. The total from the values of the individual end conditions is specified for this. If, for example, you want to define the end condition "Maximum length" OR "Sequence 1", the value "40" must be specified.
	MAXLEN	UINT	Specifies the maximum number of characters in a message. Valid values* for this parameter are as follows: • 1 character (default) • 0 to 1024 characters This setting is only enabled if the "Maximum length" end condition is set at the ENDCOND parameter.
	N	UINT	Offset of the length field in the message Valid values for this parameter are as follows: • 0 characters (default) • 0 to 1024 characters This setting is only enabled if the "N+LEN+M" end condition is set at the END-COND parameter.
	LENGTHSIZE	UINT	Size of the length field in bytes Valid values* for this parameter are as follows: • 0 bytes (default) • 1 byte • 2 bytes • 4 bytes This setting is only enabled if the "N+LEN+M" end condition is set at the END-COND parameter.
	LENGTHM	UINT	Specifies the number of end characters that follow the length field but are not contained in the length of the message. Valid values for this parameter are as follows: • 0 characters (default) • 0 to 255 characters This setting is only enabled if the "N+LEN+M" end condition is set at the END-COND parameter.
	RCVTIME	UINT	Specifies the maximum duration for the receipt of the first character of a message. Valid values for this parameter are as follows: • 200 ms (default)

Parameters		Data type	Description
			• 0 to 65535 ms in steps of 1 ms This setting is only enabled if the "Reply timeout" end condition is set at the ENDCOND parameter.
	MSGTIME	UINT	Specifies the maximum duration of the receipt of a message. Valid values for this parameter are as follows: • 200 ms (default) • 0 to 65535 ms in steps of 1 ms This setting is only enabled if the "Message timeout" end condition is set at the ENDCOND parameter.
	CHARGAP	UINT	Specifies the time interval between received consecutive characters. Valid values for this parameter are as follows: • 12 bit times (default) • 0 to 2500 bit times This setting is only enabled if the "Timeout within the character string" end condition is set at the ENDCOND parameter.
	SEQ.CTL	BYTE	Character string: Control sequence for each character You can use the bit position of the character to define which characters of the character string will be considered or ignored. To evaluate the characters, the corresponding bits have to be set. • Bit 0: 1 character • Bit 1: 2 characters • Bit 2: 3 characters • Bit 3: 4 characters • Bit 4: 5 characters A character is ignored when the corresponding bit is reset.
	SEQ.STR	CHAR[5]	Character string: Start character (5 characters)
	* These value ranges also apply to the corresponding hardware settings for specifying the end of message.		

Start conditions for the message receipt (STARTCOND parameter)

The start of the message is recognized by the receiver if a configured start condition applies. The following conditions can be defined as start conditions for message receipt:

- Start character: The start of a message is recognized when a certain character occurs. This character is stored as first character of the message. All characters received before the start character are rejected.
- Any character: Any character can define the start of a message. This character is stored as first character of the message.
- Line break: The start of a message is recognized if the received data stream is interrupted for longer than one character length.

- Idle line: The start of a message is recognized when the send transmission line is in the idle state for a certain time (specified in bit times) followed by renewed transmission of characters.
- Character string (sequence): The start of a message is recognized when a specified character sequence occurs in the data stream. You can specify up to four character sequences with up to five characters each.
  Example: A received hexadecimal message includes the following characters: "68 10 aa 68 bb 10 aa 16". The configured start character sequences are listed in the following table. Start character sequences will be evaluated once the first character 68H has been received successfully. After the fourth character has been received successfully (the second 68H), the start condition "1" has been met. Once the start conditions have been met, evaluation of the end conditions will start.
  Processing of the start character sequence can end due to different errors in parity, framing or time intervals between characters. These errors will prevent reception of the message, because the start condition has not been met.

Start condition	First character	First character +1	First character +2	First character +3	First character +4
1	68H	xx	xx	68H	xx
2	10H	aaH	xx	xx	xx
3	dcH	aaH	xx	xx	xx
4	e5H	xx	xx	xx	xx

End conditions for the message receipt (ENDCOND parameter)

The start of a message is recognized by the receiver if a configured end condition applies. The following conditions can be defined as end conditions for message receipt:

- Reply timeout: The receipt of messages will end when the specified maximum duration for the receipt of a character is exceeded. The maximum duration is defined at the RCVTIME parameter. The defined time starts to run down as soon at the last transmission is completed and the RCV_PTP instruction enables the receipt of the message. If no character was received within the defined time (RCVTIME), the RCV_PTP instruction reports an error.
- Message timeout: The receipt of messages will end when the specified maximum duration for the receipt of a message is exceeded. The maximum duration is defined at the MSGTIME parameter. The defined time starts to run down as soon as the first character of the message is received.
- Timeout within the character string: The receipt of messages will end when the time interval between the receipt of two consecutive characters is longer than the value at the CHARGAP parameter.
- Maximum length: The receipt of messages will end when the length of the message defined at the MAXLEN parameter is exceeded.

- Reading message length (N+LEN+M): The receipt of messages will end when a certain message length is reached. This length is calculated by the values of the following parameters:
  - N: Position of the character in the message, as of which the length field begins.
  - LENGTHSIZE: Size of the length field in bytes
  - LENGTHM: Number of end characters that follow the length field. These characters are not taken into account in the evaluation of the message length.
- Character string: The receipt of messages will end when a defined character sequence is received. The character string can contain a maximum of five characters. For each character of the character string, you can use the bit position to define if this will be considered or ignored in the evaluation.

STATUS parameter

Error code* (W#16#...)	Description
80C0	Error in start condition
80C1	• Error in end condition • No end condition defined
80C2	Receive interrupt enabled
80C3	A value that is equal to 0 or greater than 4132 was entered at the MAXLEN parameter while the "Maximum length" end condition was set.
80C4	A value that is greater than 4131 was entered at the N parameter while the "N+LEN+M" end condition was set.
80C5	A value that is equal to 0 or invalid was entered at the LENGTHSIZE parameter while the "N+LEN+M" end condition was set.
80C6	A value that is greater than 255 was entered at the LENGTHM parameter while the "N+LEN+M" end condition was set.
80C7	A message length greater than 4132 was calculated while the "N+LEN+M" end condition was set.
80C8	A value that is equal to 0 was entered at the RCVTIME parameter while the "Reply timeout" end condition was set.
80C9	A value that is equal to 0 or greater than 2500 was entered at the CHARGAP parameter while the "Timeout within a character string" end condition was set.
80CA	A value that is equal to 0 or greater than 2500 was entered at the IDLETIME parameter while the "Idle line" start condition was set.
80CB	All characters of the character string are marked as "Don't care" even though "Character string" is set as the end condition.
80CC	All characters of the character string are marked as "Don't care" even though "Character string" is set as the start condition.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

You will find more detailed information on general error codes of the communication instructions in “General status information of the communications blocks (Page 4604)“.

SEND_PTP: Transmit send buffer data (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

You use the "SEND_PTP" instruction to start the transmission of data. The "SEND_PTP" instruction does not execute the actual transmission of the data. The data of the send buffer is transmitted to the relevant point-to-point communication module (CM). The CM executes the actual transmission.

Parameters

The following table shows the parameters of the "SEND_PTP" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Enables the requested transmission on a rising edge of this enable input. The content of buffer is transmitted to the point-to-point communication module (CM).
PORT	Input	PORT	I, Q, M, D, L or constant	Identification of the communication port (HW-ID)
BUFFER	Input	VARIANT	I, Q, M, D, L or constant	Pointer to the start address of the send buffer. Boolean values or Array of BOOL are not supported.
LENGTH	Input	UINT	I, Q, M, D, L or constant	Length of the send buffer
PTRCL	Input	BOOL	I, Q, M, D, L or constant	This parameter selects the buffer for normal point-to-point communication or for specific Siemens protocols implemented in the connected CM. FALSE = point-to-point operations controlled by the user program (only valid option)
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
7000	The send operation is not active.
7001	The send operation is processing the first call.
7002	The send operation is processing subsequent calls (queries following the first call).
8080	The identifier entered for the communications port number is invalid.
8088	The length of the LENGHT parameter does not correspond to the length of data to be sent. See also: Parameters LENGHT and BUFFER.
80D0	A new send request was received while a transmission was taking place.
80D1	The transmission was interrupted because the CTS signal was not confirmed within the specified wait time.
80D2	The send request was interrupted because the communications partner (DCE) signaled that it was not willing to receive (DSR).
80D3	The send request was interrupted because the maximum size of the waiting loop was exceeded (more than 1024 Byte).
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

You will find more detailed information on general error codes of the communication instructions in “General status information of the communications blocks (Page 4604)“.

LENGTH and BUFFER parameters

The minimum data size that can be sent by the "PTP_SEND" instruction is one byte. The parameter BUFFER defines the size of the data to be sent. You can use neither the BOOL nor the Array of BOOL data type for the BUFFER parameter.

LENGTH parameter	BUFFER parameter	Description
LENGTH = 0	Not used	The complete data is sent as defined by the BUFFER parameter. If LENGTH = 0, you do not need to specify the number of bytes transferred.
LENGTH > 0	Elementary data type	The LENGTH value must contain the byte count of this data type. Otherwise, data is not transferred and error 8088 is output.
	STRUCT	The LENGTH value can contain a byte count that is smaller than the complete byte length of the structure. In this case, only the first LENGTH bytes are transferred.
	ARRAY	The LENGTH value can contain a byte count that is smaller than the complete byte length of the field. In this case, only the field elements that fit completely in the LENGTH bytes are transferred. The LENGTH value must be a multiple of the byte count of the data elements. Otherwise, STATUS = 8088, ERROR = 1, and no data is transferred.
	STRING	The complete memory arrangement of the character sequence format will be transmitted as well as the information about maximum length of the character string and the actual length of the character string. The LENGTH value must contain bytes for maximum length, actual length, and the characters of the character string. With the data type STRING, all lengths and characters have the size of one byte. If a character string is used for the BUFFER parameter, the LENGTH value must also contain two bytes for the two length fields.

RCV_PTP: Enable receive messages (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

With the RCV_PTP instruction you enable receipt of a sent message. Each message must be enabled individually. The sent data is only available in the receive area when the message has been acknowledged by the relevant communications partner.

Parameters

The following table shows the parameters of the "RCV_PTP" instruction:

Parameter	Declaration	Data type	Memory area	Description
EN_R	Input	BOOL	I, Q, M, D, L or constant	Enables receipt on a rising edge
PORT	Input	PORT	I, Q, M, D, L or constant	Identification of the communication port (HW-ID)
BUFFER	Input	VARIANT	I, Q, M, D, L or constant	Points to the start address of the receive buffer. Do not use a tag of the type STRING in the receive buffer.
NDR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction
LENGTH	Output	UINT	I, Q, M, D, L	Length of the message in the receive buffer

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#....)	Description
80E0	Receipt of messages was terminated because the receive buffer is full.
80E1	Receipt of messages was terminated as a result of a parity error.
80E2	Receipt of messages was terminated as a result of a framing error.
80E3	Receipt of messages was terminated as a result of an overflow error.
80E4	Receipt of messages was terminated because the calculated message length (N+LEN+M) exceeds the size of the receive buffer.
8080	The identifier entered for the communications port number is invalid.
8088	A data type STRING is referenced via the BUFFER parameter.
0094	Receipt of messages was terminated because the maximum character length was received.
0095	Receipt of messages was terminated as a result of a timeout.
0096	Receipt of messages was terminated because of a timeout within the character string.
0097	Receipt of messages was terminated as a result of a reply timeout.
0098	Receipt of messages was terminated because the "N+LEN+M" length condition has been satisfied.
0099	Receipt of messages was terminated because the character string defined as the end condition was received.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

You will find more detailed information on general error codes of the communication instructions in “General status information of the communications blocks (Page 4604)“.

RCV_RST: Delete receive buffer (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

With the "RCV_RST" instruction, you delete the receive buffer of a communications partner.

Parameters

The following table shows the parameters of the "RCV_RST" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Enables deleting of the receive buffer on a rising edge
PORT	Input	PORT	I, Q, M, D, L or constant	Identification of the communication port (HW-ID)
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction You will find more detailed information on general error codes of the communication instructions in “General status information of the communications blocks (Page 4604)“.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

SGN_GET: Query RS-232 signals (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

With the "SGN_GET" instruction, you query the current state of several signals of an RS-232 communications module.

Parameters

The following table shows the parameters of the "SGN_GET" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Enables the query on a rising edge
PORT	Input	PORT	I, Q, M, D, L or constant	Identification of the communication port (HW-ID)
NDR	Output	BOOL	I, Q, M, D, L	Is set for one cycle if new data are ready for sending and the instruction was executed error-free.
DTR	Output	BOOL	I, Q, M, D, L	Data terminal ready, module ready
DSR	Output	BOOL	I, Q, M, D, L	Data set ready, communications partner ready
RTS	Output	BOOL	I, Q, M, D, L	Send request, module ready to send
CTS	Output	BOOL	I, Q, M, D, L	Clear to send, communications partner can receive data (reaction to RTS = ON of the module).
DCD	Output	BOOL	I, Q, M, D, L	Data carrier detect, received signal level
RING	Output	BOOL	I, Q, M, D, L	Ring display, display of an incoming call
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#....)	Description
80F0	The communication module is an RS-485 module and no signals are available.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

You will find more detailed information on general error codes of the communication instructions in “General status information of the communications blocks (Page 4604)“.

SGN_SET: Set RS-232 signals (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

With the "SGN_SET" instruction, you set the status of the output signals of an RS-232 communications module.

Parameters

The following table shows the parameters of the "SGN_SET" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Activates the action on a rising edge Initial value: FALSE
PORT	Input	PORT	I, Q, M, D, L or constant	Identification of the communication port (HW-ID) Initial value: FALSE
SIGNAL	Input	BYTE	I, Q, M, D, L or constant	Specifies the signals to be set: • Set 01H = RTS • Set 02H = DTR • Set 04H = DSR Initial value: FALSE
RTS	Input	BOOL	I, Q, M, D, L or constant	Send request, module ready to send Initial value: FALSE
DTR	Input	BOOL	I, Q, M, D, L or constant	Data terminal ready, module ready Initial value: FALSE
DSR	Input	BOOL	I, Q, M, D, L or constant	Data set ready (applies only to interfaces of the DCE type) Initial value: FALSE
DONE	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: Job not yet started or is still executing • 1: Job executed without errors Initial value: FALSE
ERROR	Output	BOOL	I, Q, M, D, L	Status parameter with the following values: • 0: No error • 1: Error occurred Initial value: FALSE
STATUS	Output	WORD	I, Q, M, D, L	Status of the instruction Initial value: 0

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
80F0	The communication module is an RS-485 module and no signals are available.
80F1	No signals are settable because H/W flow control is enabled.
80F2	The DSR signal cannot be set because the module is a DTE device.
80F3	The DTR signal cannot be set because the module is a DCE device.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

You will find more detailed information on general error codes of the communication instructions in “General status information of the communications blocks (Page 4604)“.

General status information of the communications blocks (S7-1200)

General information on execution status of the communications blocks

The following table shows which general information can be output at the STATUS parameter of the communications blocks:

Error code* (W#16#....)	Description
0000	No error
7000	No job processing active.
7001	Start of job processing. Parameter BUSY = 1, DONE = 0.
7002	Intermediate call (REQ irrelevant): Instruction already active; BUSY has the value "1".
8x3A	Invalid pointer at parameter x.
8070	All internal instance memories are in use.
8080	The identifier entered for the communications port is invalid
8081	Timeout, module error, internal error
8082	Parameter assignment failed because parameter assignment is currently being performed in the background.
8083	Buffer overflow: The CM or CB has returned a receipt message with a length that is greater than permitted by the length parameter.
8085	Error specifying the length at the LENGHT parameter. The specified length is "0" or greater than the maximum permitted value.
8090	Message length invalid, module invalid, message invalid
8091	Incorrect version in parameterization message
8092	Invalid record length in parameterization message
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

USS (S7-1200)

Overview of USS instructions (S7-1200)

Introduction

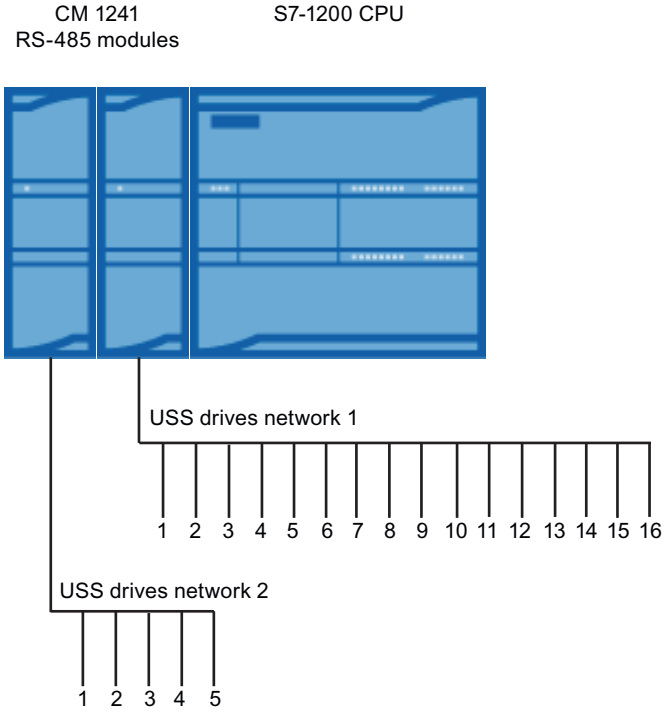
The USS instructions control the operation of drives that support the universal serial interface (USS). With the USS instructions, you can communicate with more than one drive via an RS-485 connection.

To do this, you require a CM 1241 RS-485 communications module or a CB 1241 RS-485 communications board. Up to three CM 1241 RS-485 modules and one CB 1241 RS-485 board can be installed in an S7-1200 CPU.

Each RS-485 port can operate up to sixteen drives.

The USS protocol uses a master/slave network for communication via a serial bus. The master uses an address parameter to send a message to a selected slave. A slave itself can never send without previously receiving a request. Direct exchange of messages between slaves is not possible. USS communication works in half duplex mode.

The following figure shows an example of a USS network diagram:



Requirements for using the USS protocol (S7-1200)

General requirements for setting up a drive

- The use of 4 PKW words must be set up for the drives.
- The drives can be configured for 2, 4, 6 or 8 PZD words.
- The number of PZD words in the drive must correspond to the PZD_LEN input of the "USS_DRIVE (Page 4610)" instruction of the drive.
- The baud rate of all drives must match the baud rate of the BAUD input parameter of the "USS_PORT (Page 4609)" instruction.
- The drive must be set up for remote control.
- USS must be specified for the desired frequency value on the COM connection of the drive.
- 1 to 16 must be set as the drive address. This address must correspond to the address of the DRIVE input parameter of the "USS_DRIVE (Page 4610)" instruction.
- To control the direction of the drive, the polarity of the drive desired value must be set up.
- The RS-485 network must be connected correctly.

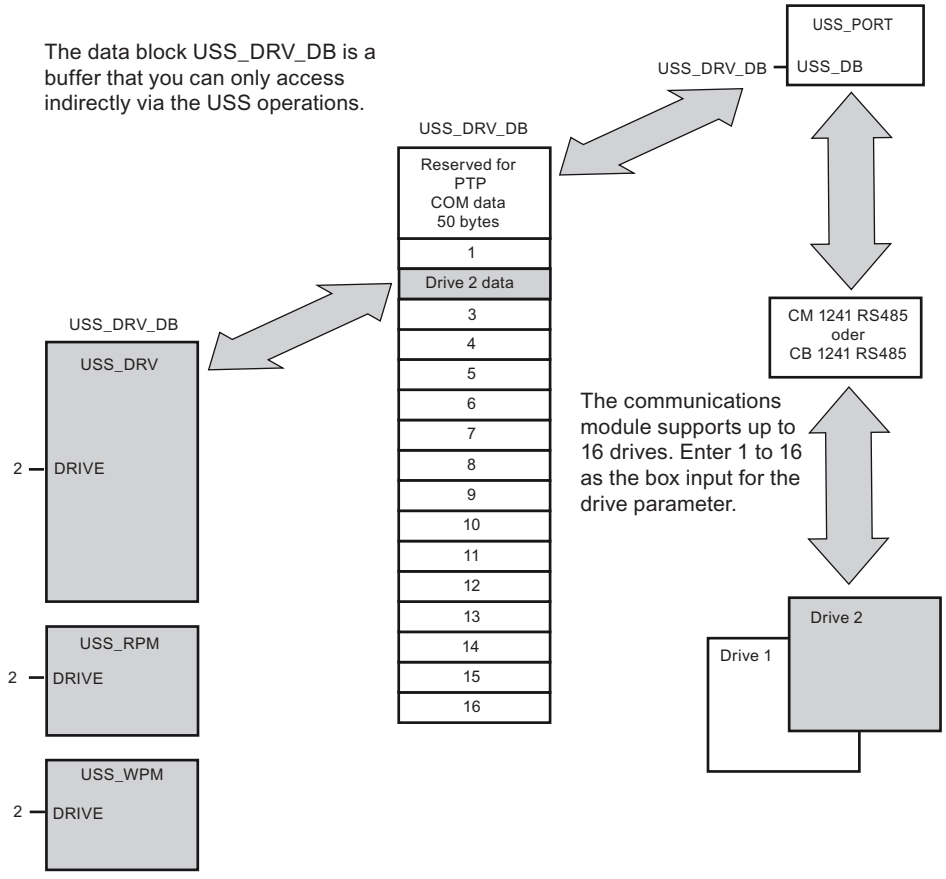
Definition: PKW / PZD area

- The PKW area relates to the handling of the parameter-identifier-value interface.
  The PKW interface is not a physical interface but describes a mechanism that controls the exchange of parameters between two communications partners, in other words, reading and writing parameter values, parameter descriptions and corresponding texts and the handling of parameter changes due to spontaneous messages. All the tasks handled via the PKW interface are essentially tasks for operator control and monitoring, service and diagnostics.
- The PZD area includes the signals required for automation:
  - Control word(s), and setpoint(s) from the master to the slave
  - Status word(s), and actual value(s) from the slave to the master.

Both areas together result in the user data field. This is transferred by the master to the slave as a job frame or by the slave to the master as a reply frame.

Description

Each communication module CM 1241 RS485 supports a maximum of 16 drives. A single instance data block contains temporary memory and buffer functions for all drives in the USS network that are connected to a PtP communications module you installed. The USS instructions for these drives have shared access to the information in this data block.



- All drives (max. 16) that are connected to an RS485 port are part of the same USS network. All drives that are connected to a different RS485 port are part of a different USS network. As the S7-1200 supports up to three CM 1241 RS485 modules, you can set up up to three USS networks, each having a maximum of 16 drives, thus a total of 48 USS drives are supported.
- Each USS network is managed via a unique data block (three data blocks are required for three USS networks with three CM 1241 RS485 modules). All instructions that belong to a USS network must share this data block. This includes all "USS_DRIVE (Page 4610)", "USS_PORT (Page 4609)", "USS_RPM (Page 4613)", and "USS_WPM (Page 4614)" instructions for controlling all drives in a USS network.

- The instruction "USS_DRIVE (Page 4610)" is a function block (FB). When you insert the instruction "USS_DRIVE" in the editor, the "Call options" dialog will ask you to assign a DB to the instruction.
  - If this is the first "USS_DRIVE" instruction in this program for this USS network, you can accept the default DB assignment (or, if necessary, change the name) and the new DB will be created.
  - If, however, this is not the first "USS_DRIVE" instruction for this network, you must select the DB that was previously assigned to this USS network in the drop-down list in the "Call options" dialog.
- All "USS_PORT (Page 4609), USS_RPM (Page 4613)", and "USS_WPM (Page 4614)" instructions are functions (FCs). When you insert these FCs in the editor, no DB is assigned. Instead, you have to assign the DB in question to the USS_DB input of these instructions (double-click on the parameter field and then on the icon to display the available DBs).
- The instruction "USS_PORT (Page 4609)" controls communication between the CPU and the drives via the PtP communications module. Whenever this instruction is called, communication with a drive is processed. Your program must call this function quickly enough so that the drives do not report a timeout. The instruction can be called from the main program or any interrupt OB.
- The function block "USS_DRIVE (Page 4610)" gives your program access to a specified drive in the USS network. Its inputs and outputs correspond to the states and operating functions of the drive. If there are 16 drives in the network, "USS_DRIVE" must be called at least 16 times in your program, i.e., once for each drive. How quickly these blocks are called depends on the required speed for controlling drive functions.
  You can only call the instruction "USS_DRIVE" from the main program OB.

  CAUTION

  Call "USS_DRIVE", "USS_RPM", "USS_WPM" only from the main program OB. The instruction "USS_PORT" can be called from any OB, it is usually called from a time-delay interrupt OB. If the instruction "USS_PORT" is interrupted during execution, this may result in unexpected errors.

The "USS_RPM" and "USS_WPM" instructions are used to read and write the operating parameters of the drive. These parameters control the internal mode of operation of the drive. A definition of these parameters can be found in the drive manual.

Your program may also contain any number of these instructions; however only one read or write request can be active for a drive. You may only call the instructions "USS_RPM" and "USS_WPM" from the main program OB.

Calculating the time for communication with the drive

Communication with the drive is runs asynchronous to the cycle of the S7-1200. The S7-1200 runs through several cycles before communication with a drive is completed.

The interval of "USS_PORT" is the time required for a drive transaction. The following table shows the minimum intervals for "USS_PORT" for each baud rate. Calling the "USS_PORT" more frequently than the "USS_PORT" interval will not increase the number of transactions. The timeout interval of the drive is the period of time available to a transaction if 3 attempts

are required to complete the transaction due to communications errors. By default, up to 2 further attempts are made for each transaction with the USS protocol.

Baud rate	Calculated minimum interval for calling USS_PORT (ms)	Drive message interval timeout per drive (ms)
1200	790	2370
2400	405	1215
4800	212.5	638
9600	116.3	349
19200	68.2	205
38400	44.1	133
57600	36.1	109
115200	28.1	85

USS_PORT: Edit communication via USS network (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The "USS_PORT" instruction handles communication over the USS network. In the program, use one "USS_PORT" instruction per PtP communications port to control the transmission to or from one drive.

All USS instructions that are assigned to one USS network and one PtP communications port must use the same instance data block.

Call

Your program must execute the "USS_PORT" instruction often enough to prevent timeouts in the drive. You should therefore call the "USS_PORT" instruction from a cyclic interrupt OB to prevent drive timeouts and keep the most recent USS data updates available for "USS_DRIVE (Page 4610)" calls.

Parameters

The following table shows the parameters of the "USS_PORT" instruction:

Parameter	Declaration	Data type	Memory area	Description
PORT	Input	PORT	D, L or constant	PtP communications port identifier Constant that can be referenced within the "Constants" tab of the default tag table.
BAUD	Input	DINT	I, Q, M, D, L or constant	Baud rate for USS communication.
USS_DB	InOut	USS_BASE	D	Reference to the instance DB of the "USS_DRIVE (Page 4610)" instruction.
ERROR	Output	BOOL	I, Q, M, D, L	ERROR is set to TRUE if an error occurs. A corresponding error code will be output at the STATUS output.
STATUS (Page 4616)	Output	WORD	I, Q, M, D, L	Status value of the request. It indicates the result of the cycle or initialization. Additional information is available in the "USS_Extended_Error (Page 4616)" tag for some status codes.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

USS_DRIVE: Swap data with drive (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The "USS_DRIVE" instruction exchanges data with the drive by creating request messages and interpreting the drive response messages. A separate instruction must be used for each drive, but all USS instructions assigned to one USS network and one PtP communications module must use the same instance data block. You must create the DB name when you place the first "USS_DRIVE" instruction. Then reuse this DB that was created when the initial instruction was inserted.

When the "USS_DRIVE" instruction is executed the first time, the drive indicated by the USS address (parameter DRIVE) is initialized in the instance DB. After this initialization, subsequent "USS_PORT (Page 4609)" instructions can start communication with the drive at this drive number.

Changing the drive number requires a PLC STOP to RUN mode transition that initializes the instance DB. Input parameters are configured in the USS send buffer, and outputs are read from a "previous" valid response buffer if any exists. There is no data transmission during execution of the "USS_DRIVE" instruction. After executing “USS_PORT (Page 4609) there is communication with the drives, “USS_DRIVE" only configures the messages to be sent and evaluates data received in a previous request.

You can control the drive direction of rotation using either the DIR (BOOL) input or using the sign (positive or negative) at the SPEED_SP (REAL) input. The following table explains how these inputs work together to determine the drive direction, assuming the motor is wired for forward rotation.

SPEED_SP	DIR	Direction of rotation of drive
Value > 0	0	Reverse
Value > 0	1	Forward
Value < 0	0	Forward
Value < 0	1	Reverse

Parameters

Expand the box to display all the parameters by clicking the bottom of the box. The parameter connections that are grayed are optional and do not need to be assigned.

The following table shows the parameters of the "USS_DRIVE" instruction:

Parameter	Declaration	Data type	Memory area	Description
RUN	Input	BOOL	I, Q, M, D, L or constant	Drive start bit: If this parameter has the value TRUE, this input enables the drive to run at the preset speed.
OFF2	Input	BOOL	I, Q, M, D, L or constant	"Electrical stop" bit: If this parameter has the value FALSE, this bit cause the drive to coast to a stop without braking.
OFF3	Input	BOOL	I, Q, M, D, L or constant	Fast stop bit - If this parameter has the value FALSE, this bit causes a fast stop by braking the drive.
F_ACK	Input	BOOL	I, Q, M, D, L or constant	Fault acknowledge bit - This bit resets the fault bit on a drive. This bit is set after the fault is cleared to indicate to the drive that it no longer needs to indicate the previous fault.
DIR	Input	BOOL	I, Q, M, D, L or constant	Drive direction control - This bit is set to indicate that the direction is forward (when SPEED_SP is positive).
DRIVE	Input	USINT	I, Q, M, D, L or constant	Drive address: This input is the address of the USS drive. The valid range is drive 1 to drive 16.
PZD_LEN	Input	USINT	I, Q, M, D, L or constant	Word length - This is the number of words of PZD data. Valid values are 2, 4, 6, or 8 words. The default is 2.
SPEED_SP	Input	REAL	I, Q, M, D, L or constant	Speed setpoint - This is the speed of the drive as a percentage of configured frequency. A positive value specifies forward direction (when DIR has the value TRUE).
CTRL3	Input	WORD	I, Q, M, D, L or constant	Control word 3 - A value written to a user-configurable parameter on the drive. The user must configure this on the drive. Optional parameter
CTRL4	Input	WORD	I, Q, M, D, L or constant	Control word 4 - A value written to a user-configurable parameter on the drive. The user must configure this on the drive. Optional parameter
CTRL5	Input	WORD	I, Q, M, D, L or constant	Control word 5 - A value written to a user-configurable parameter on the drive. The user must configure this on the drive. Optional parameter

Parameter	Declaration	Data type	Memory area	Description
CTRL6	Input	WORD	I, Q, M, D, L or constant	Control word 6 - A value written to a user-configurable parameter on the drive. The user must configure this on the drive.
CTRL7	Input	WORD	I, Q, M, D, L or constant	Control word 7 - A value written to a user-configurable parameter on the drive. The user must configure this on the drive. Optional parameter
CTRL8	Input	WORD	I, Q, M, D, L or constant	Control word 8 - A value written to a user-configurable parameter on the drive. The user must configure this on the drive. Optional parameter
NDR	Output	BOOL	I, Q, M, D, L	New data ready - If this parameter has the value TRUE, the bit indicates that the output contains data from a new communication request.
ERROR	Output	BOOL	I, Q, M, D, L	Error occurred - If this parameter has the value TRUE, this indicates that an error has occurred and the STATUS output is valid. All other outputs are set to zero on an error. Communication errors are only reported at the ERROR and STATUS outputs of the "USS_PORT" instruction.
STATUS (Page 4616)	Output	WORD	I, Q, M, D, L	Status value of the request. It indicates the result of the cycle. This is not a status word returned from the drive.
RUN_EN	Output	BOOL	I, Q, M, D, L	Run enabled - This bit indicates whether the drive is running.
D_DIR	Output	BOOL	I, Q, M, D, L	Drive direction - This bit indicates whether the drive is running forward.
INHIBIT	Output	BOOL	I, Q, M, D, L	Drive inhibited - This bit indicates the state of the inhibit bit on the drive.
FAULT	Output	BOOL	I, Q, M, D, L	Drive fault - This bit indicates that the drive has registered a fault. The user must eliminate the fault and then set the F_ACK bit to clear this bit.
SPEED	Output	REAL	I, Q, M, D, L	Drive current speed (scaled value of drive status word 2) - The value of the speed of the drive as a percentage of configured speed.
STATUS1	Output	WORD	I, Q, M, D, L	Drive status word 1 - This value contains fixed status bits of a drive.
STATUS3	Output	WORD	I, Q, M, D, L	Drive status word 3 - This value contains a user-configurable status word on the drive.
STATUS4	Output	WORD	I, Q, M, D, L	Drive status word 4 - This value contains a user-configurable status word on the drive.
STATUS5	Output	WORD	I, Q, M, D, L	Drive status word 5 - This value contains a user-configurable status word on the drive.
STATUS6	Output	WORD	I, Q, M, D, L	Drive status word 6 - This value contains a user-configurable status word on the drive.
STATUS7	Output	WORD	I, Q, M, D, L	Drive status word 7 - This value contains a user-configurable status word on the drive.
STATUS8	Output	WORD	I, Q, M, D, L	Drive status word 8 - This value contains a user-configurable status word on the drive.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

USS_RPM: Readout parameters from the drive (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The "USS_RPM" instruction reads a parameter from the drive. All USS functions that are assigned to one USS network and one PtP communications module must use the same data block. "USS_RPM" must be called from the main program OB.

Parameter

The following table shows the parameters of the "USS_RPM" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L	Send request: If this parameter has the value TRUE, it indicates that a new read request is desired. This is ignored if the request for this parameter is already pending.
DRIVE	Input	USINT	I, Q, M, D, L or constant	Drive address: This input is the address of the USS drive. The valid range is drive 1 to drive 16.
PARAM	Input	UINT	I, Q, M, D, L or constant	Parameter number: This input specifies which drive parameter is written. The range of this parameter is 0 to 2047. See your drive manual for details on how to access any parameters above this range.
INDEX	Input	UINT	I, Q, M, D, L or constant	Parameter index: This input specifies which drive parameter index is to be written. This is a 16-bit value where the least significant byte is the actual index value with a range of (0 to 255). The most significant byte may also be used by the drive and is drive-specific. See your drive manual for additional information.
USS_DB	InOut	USS_BASE	D	Reference to the instance DB that is created and initialized when a "USS_DRIVE" instruction is inserted in your program.
DONE	Output	BOOL	I, Q, M, D, L	If this parameter has the value TRUE, it indicates that the VALUE output holds the previously requested read parameter value. This bit is set when the "USS_DRIVE" instruction recognizes the read response from the drive. This bit is reset when either: • you request the response data via another "USS_RPM" poll or • the second of the next two calls of "USS_DRIVE (Page 4610)" is executed

Parameter	Declaration	Data type	Memory area	Description
ERROR	Output	BOOL	I, Q, M, D, L	Error occurred - If this parameter has the value TRUE, this indicates that an error has occurred and the STATUS output is valid. All other outputs are set to zero on an error. Communication errors are only reported at the ERROR and STATUS outputs of the "USS_PORT (Page 4609)" instruction.
STATUS (Page 4616)	Output	WORD	I, Q, M, D, L	This is the status value of the request. It indicates the result of the read request. Additional information is available in the "USS_Extended_Error (Page 4616)" tag for some status codes.
VALUE	Output	VARIANT	I, Q, M, D, L	This is the value of the parameter that was read and is valid only when the DONE bit has the value TRUE.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

USS_WPM: Change parameters in the drive (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The "USS_WPM" instruction modifies a parameter in the drive. All USS functions that are assigned to one USS network and one PtP communications module must use the same data block. "USS_WPM" must be called from the main program OB.

Note

EEPROM write operations

Beware of overusing the EEPROM write operation. Minimize the number of EEPROM write operations to extend the EEPROM life.

Parameter

The following table shows the parameters of the "USS_WPM" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L	Send request: If this parameter has the value TRUE, it indicates that a new write request is desired. This is ignored if the request for this parameter is already pending.
DRIVE	Input	USINT	I, Q, M, D, L or constant	Drive address: This input is the address of the USS drive. The valid range is drive 1 to drive 16.

Parameter	Declaration	Data type	Memory area	Description
PARAM	Input	UINT	I, Q, M, D, L or constant	Parameter number: This input specifies which drive parameter is written. The range of this parameter is 0 to 2047. See your drive manual for details on how to access any parameters above this range.
INDEX	Input	UINT	I, Q, M, D, L or constant	Parameter index: This input specifies which drive parameter index is to be written. This is a 16-bit value where the least significant byte is the actual index value with a range of (0 to 255). The most significant byte may also be used by the drive and is drive-specific. See your drive manual for additional information.
EEPROM	Input	BOOL	I, Q, M, D, L or constant	Store to drive EEPROM: If this parameter has the value TRUE, values written to the drive parameter will be stored in the drive EEPROM. If this parameter has the value FALSE, the value written is only temporarily saved and will be lost the next time the drive is switched on.
VALUE	Input	VARIANT	I, Q, M, D, L or constant	The value of the parameter that is to be written. It must be valid on the transition of REQ.
USS_DB	InOut	USS_BASE	D	This is a reference to the instance DB that is created and initialized when a "USS_DRIVE (Page 4610)" instruction is inserted in your program.
DONE	Output	BOOL	I, Q, M, D, L	If this parameter has the value TRUE, the VALUE input was written to the drive. This bit is set when the "USS_DRIVE (Page 4610)" instruction recognizes the write response from the drive. This bit is reset when either: You request the drive's confirmation that the write operation has been completed via another "USS_WPM" query or when the second of the next two calls of "USS_DRIVE (Page 4610)" is executed.
ERROR	Output	BOOL	I, Q, M, D, L	Error occurred: If this parameter has the value TRUE, this indicates that an error has occurred and the STATUS output is valid. All other outputs are set to zero on an error. Communication errors are only reported at the ERROR and STATUS outputs of the "USS_PORT (Page 4609)" instruction.
STATUS (Page 4616)	Output	WORD	I, Q, M, D, L	This is the status value of the request. It indicates the result of the write request. Additional information is available in the "USS_Extended_Error (Page 4616)" tag for some status codes.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Parameter STATUS of USS instructions (S7-1200)

STATUS parameter

The following table contains the status codes of the USS operation that are output at the STATUS output of the USS instructions:

STATUS* (W#16#....)	Description
0000	No error
8180	The length of the drive response did not match the characters received from the drive. The drive number where the error occurred is returned in the "USS_Extended_Error" tag. See the extended error description below this table.
8181	The parameter VALUE is not of the data type WORD, REAL, or DWORD
8182	User supplied a parameter value of the type word and received a DWORD or REAL from the drive in the response
8183	User supplied a parameter value of the type DWORD or REAL and received a word from the drive in the response
8184	Response telegram from drive had a bad checksum. The drive number where the error occurred is returned in the "USS_Extended_Error" tag. See the extended error description below this table.
8185	Illegal drive address (valid drive address range: 1-16)
8186	Speed set point out of valid range (valid speed SP range: -200% to 200%)
8187	Wrong drive number responded to the request sent. The drive number where the error occurred is returned in the "USS_Extended_Error" tag. See the extended error description below this table.
8188	Illegal PZD word length specified (valid range = 2, 4, 6 or 8 words)
8189	Illegal baud rate was specified
818A	Parameter request channel is in use by another request for this drive
818B	Drive has not responded to requests and retries. The drive number where the error occurred is returned in the "USS_Extended_Error" tag. See the extended error description below this table.
818C	Drive returned an extended error on a parameter request operation. See the extended error description below this table.
818D	Drive returned an invalid access error on a parameter request operation. See your drive manual for information of why parameter access may be limited
818E	Drive has not been initialized: This error code is output at "USS_RPM (Page 4613)" or "USS_WPM (Page 4614)" if the "USS_DRIVE (Page 4610)" instruction has not been called at least once for this drive. This prevents the initialization of the first cycle of "USS_DRIVE (Page 4610)" from overwriting a pending parameter read or write request since it initializes the drive as a new entry. To eliminate this error, call the "USS_DRIVE (Page 4610)" instruction for this drive.
80Ax-80Fx	Specific errors returned from PtP (Point-to-Point) communication instructions called by the USS library: These error code values are not modified by the USS library and are defined in the PtP instruction descriptions.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For additional information on toggling display formats, refer to "See also".	

USS_Extended_Error - USS drive extended error codes

USS drives support read and write access to a drive's internal parameters. This feature allows distributed control and configuration of the drive. Drive parameter access operations can fail due to errors such as values out of range or invalid requests in a drive's current mode. The drive generates an error code that is output in the "USS_Extended_Error" variable in the instance DB of the "USS_DRIVE (Page 4610)" instruction. This error code value is only valid for the last execution of the "USS_RPM (Page 4613)" or "USS_WPM (Page 4614)" instruction. The drive error code is put into the "USS_Extended_Error" tag when the value of STATUS is hexadecimal 818C. The error code of "USS_Extended_Error" depends on the drive variant. See the drive's manual for a description of the extended error codes for read and write parameter operations.

MODBUS (RTU) (S7-1200)

MB_COMM_LOAD: Configure port on the PtP module for Modbus RTU (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The "MB_COMM_LOAD" instruction configures a port for communication using the Modbus RTU protocol. The following hardware can be used for this:

- Up to three point-to-point modules (PtP) CM 1241 RS485 or CM 1241 RS232
- A communications board CB 1241 RS485 in addition to this

After configuration of the port, you communicate over Modbus by executing the "MB_SLAVE" or "MB_MASTER" instruction.

Call

"MB_COMM_LOAD" must be called once to configure the port for the Modbus RTU protocol. On completion of the configuration, the port can be used by the "MB_MASTER (Page 4620)" and "MB_SLAVE (Page 4628)" instructions.

"MB_COMM_LOAD" only needs to be called again if one of the communication parameters has to be modified. Each "MB_COMM_LOAD" call deletes the communications buffer. To avoid data loss during communication, you should not call the instruction unnecessarily.

One instance of "MB_COMM_LOAD" must be used to configure the port of each communication module that is used for Modbus communication. You assign a unique "MB_COMM_LOAD" instance data block for each port that you use. The S7-1200 CPU is limited to three communication modules.

An instance data block is assigned when you insert the "MB_MASTER (Page 4620)" or "MB_SLAVE (Page 4628)" instruction. This instance data block is referenced when you specify the MB_DB parameter on the "MB_COMM_LOAD" instruction.

Parameters

The following table shows the parameters of the instruction "MB_COMM_LOAD":

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L	Execution of the instruction upon a rising edge.
PORT	Input	PORT	I, Q, M, D, L or constant	ID of the communications port: After you have inserted the communications module in the device configuration, the port ID appears in the drop-down list at the PORT box connection. This constant can also be referenced within the "Constants" tab of the tag table.
BAUD	Input	UDINT	I, Q, M, D, L or constant	Baud rate selection: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200 All other values are invalid.
PARITY	Input	UINT	I, Q, M, D, L or constant	Parity selection: • 0 – None • 1 – Odd • 2 – Even
FLOW_CTRL	Input	UINT	I, Q, M, D, L or constant	Flow control selection: • 0 – (default) No flow control • 1 – Hardware flow control with RTS always ON (does not apply to RS485 ports) • 2 - Hardware flow control with RTS switched
RTS_ON_DLY	Input	UINT	I, Q, M, D, L or constant	RTS on-delay selection: • 0 – (default) No delay of RTS active until the first character of the message is transmitted. • 1 to 65535 – Delay in milliseconds of "RTS active" until the first character of the message is transmitted (does not apply to RS-485 ports). RTS delays must be used depending on the selection of FLOW_CTRL.
RTS_OFF_DLY	Input	UINT	I, Q, M, D, L or constant	RTS off-delay selection: • 0 – (default) No delay after the last character transmitted until "RTS inactive" • 1 to 65535 – Delay in milliseconds after the last character transmitted until "RTS inactive" (does not apply to RS-485 ports). RTS delays must be used regardless of the selection of FLOW_CTRL.

Parameter	Declaration	Data type	Memory area	Description
RESP_TO	Input	UINT	I, Q, M, D, L or constant	Response timeout: Time in milliseconds allowed by "MB_MASTER (Page 4620)" for the slave to respond. If the slave does not respond in this time, "MB_MASTER (Page 4620)" repeats the request or terminates the request with an error if the specified number of retries has been sent. 5 ms to 65535 ms (default = 1000 ms).
MB_DB	Input	MB_BASE	D	A reference to the instance data block of the "MB_MASTER (Page 4620)" or "MB_SLAVE (Page 4628)" instructions. After you insert "MB_SLAVE (Page 4628)" or "MB_MASTER (Page 4620)" in your program, the DB identifier appears in the drop-down list at the MB_DB box connection.
DONE	Output	BOOL	I, Q, M, D, L	Execution of instruction completed without error.
ERROR	Output	BOOL	I, Q, M, D, L	Error: • 0 – No error detected • 1 – Indicates that an error was detected. An error code is output in the STATUS parameter.
STATUS	Output	WORD	I, Q, M, D, L	Port configuration error code

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Error code* (W#16#...)	Description
0000	No error
8180	Invalid value for the port ID (wrong address for the communications module).
8181	Invalid baud rate value.
8182	Invalid parity value.
8183	Invalid flow control value.
8184	Invalid value for the timeout of the response (the time before which a timeout is reported must be at least 25 ms).
8185	Incorrect pointer in the MB_DB parameter to the instance DB of the "MB_MASTER (Page 4620)" or "MB_SLAVE (Page 4628)" instruction.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

MB_COMM_LOAD data block tags

The table below shows the public static tags in the instance DB of MB_COMM_LOAD that you can use in your program.

Table 11-82 Static tags in the instance DB

Tag	Data type	Default	Description
ICHAR_GAP	WORD	0	Delay for the character spacing between characters. This parameter is specified in milliseconds and is used to increase the anticipated period between the received characters. The corresponding number of bit times for this parameter is added to the Modbus standard value of 35 bit times (3.5 character times).
RETRIES	WORD	2	The number of repeated attempts by the master before the error code 0x80C8 is returned for "No response".
STOP_BITS	USINT	1	Number of stop bits per character Valid values 1 and 2

MB_MASTER: Communicate via the PtP port as Modbus master (S7-1200)

Description of MB_MASTER (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The "MB_MASTER" instruction allows your program to communicate as a Modbus master using a port on a point-to-point module (CM) or a communications board (CB). You can access data in one or more Modbus slave devices.

Before the "MB_MASTER" instruction can communicate with a port, "MB_COMM_LOAD (Page 4617)" must first execute.

An instance DB is created when you insert the "MB_MASTER" instruction in your program. You specify this instance DB in the MB_DB input parameter of the "MB_COMM_LOAD (Page 4617)" instruction.

Rules for Modbus master communication

- A port used for Modbus master requests cannot be used for "MB_SLAVE".
- A port can be used for one or more "MB_MASTER" calls if the same instance DB is used.
- The Modbus instructions do not use communication interrupt events to control the communication process. Your program must poll the "MB_MASTER" instruction for completed send and receive operations.

- Calling the instruction:
  - Call the "MB_MASTER" instruction if possible in a cyclic program OB. The instruction can only be called in a time delay or cyclic interrupt OB.
  - Do not call more than one "MB_MASTER" instruction in organization blocks with different priority classes. If a "MB_MASTER" instruction executes "preemptively" from a higher priority class, the instruction may execute incorrectly.
  - Do not call the "MB_MASTER" instruction in a startup, diagnostics or time error OB.
- After a transfer has started, the EN parameter (LAD/FBD) must remain set to the value "1" until the DONE or ERROR output parameter is set to "1" by the instruction. A renewed call by the REQ parameter while the instruction is executing causes an error. After the instruction executes, the bit in the REQ parameter remains set for the time specified by the BLOCKED_PROC_TIMEOUT parameter in the instance DB.
- If "MB_MASTER" sends a request to a slave, make sure that "MB_MASTER" continues to execute until the response from the slave arrives.

Parameters

The following table shows the parameters of the "MB_MASTER" instruction:

Parameters	Declaration	Data type	Memory area	Description
REQ (Page 4623)	Input	BOOL	I, Q, M, D, L	Request input: • 0 – No request • 1 – Request to transmit data to Modbus slave(s)
MB_ADDR	Input	UINT	I, Q, M, D, L or constant	Modbus RTU station address: • Default address range: 0 to 247 • Extended address range: 0 to 65535 The value "0" is reserved for broadcasting a message to all Modbus slaves. Modbus function codes 05, 06, 15, and 16 are the only function codes supported for broadcast.
MODE (Page 4623)	Input	USINT	I, Q, M, D, L or constant	Mode selection: Specifies the type of request: Read, write, or diagnostics: Refer to the Modbus functions table for details.
DATA_ADDR (Page 4623)	Input	UDINT	I, Q, M, D, L or constant	Starting address in the slave: Specifies the starting address of the data to be accessed in the Modbus slave. You will find the valid addresses in the Modbus functions table.
DATA_LEN	Input	UINT	I, Q, M, D, L or constant	Data length: Specifies the number of bits or words to be accessed in this request. You will find the valid lengths in the Modbus functions table.
DATA_PTR (Page 4625)	Input	VARIANT	M, D	Points to the DB or bit memory address of the CPU for the data to be written or read. For a DB, this must be created with the "Standard - compatible with S7-300/400" access type.
DONE	Output	BOOL	I, Q, M, D, L	• 0: Transaction not completed • 1: Transaction completed without error

Parameters	Declaration	Data type	Memory area	Description
BUSY	Output	BOOL	I, Q, M, D, L	• 0: No "MB_MASTER" transaction in progress • 1: "MB_MASTER" transaction in progress
ERROR	Output	BOOL	I, Q, M, D, L	• 0: No error • 1: Error, the error code is indicated by the STATUS parameter
STATUS	Output	WORD	I, Q, M, D, L	Execution condition code

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

Table 11-83 Communications and configuration error messages of the instruction

Error code* (W#16#....)	Description
0000	No error
80C8	Slave timeout. Check the baud rate, parity and the connectors on the slave.
80D1	The receiver issued a flow control request to suspend an active transmission and never re-enabled the transmission within the wait time. This error is also generated during hardware flow control if the recipient does not detect CTS within the wait time.
80D2	The send request was aborted because no DSR signal is received from the DCE.
80E0	The message was terminated because the receive buffer is full.
80E1	The message was terminated as a result of a parity error.
80E2	The message was terminated as a result of a framing error.
80E3	The message was terminated as a result of an overrun error.
80E4	The message was terminated as a result of the specified length exceeding the total buffer size.
8180	Invalid value for the port ID.
8186	Invalid Modbus station address
8188	The MODE parameter has an invalid value for a broadcast call.
8189	Invalid data address value.
818A	Invalid data length value.
818B	Invalid pointer to the local data source/destination: Size not correct
818C	The DATA_PTR parameter has an invalid pointer. Use a pointer to a bit memory area or a DB with the "Standard - compatible with S7-300/400" access type.
8200	Port is busy processing a send request
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Table 11-84 Error messages of the Modbus protocol

Error code* (W#16#....)	Response code of slave	Description
8380	-	CRC error
8381	01	Function code not supported
8382	03	Data length error
8383	02	Error in the data address or address outside the valid range of DATA_PTR
8384	> 03	Data value error
8385	03	Data diagnostic code value not supported (function code 08)
8386	-	Function code of the response does not match the function code of the query.
8387	-	Response from wrong slave
8388	-	The response of the slave to a write call is not correct. The data sent by the slave does not match the query from the master.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Parameter REQ (S7-1200)

Description

- REQ = FALSE: No request
- REQ = TRUE: Request to transmit data to Modbus slave(s)

You can control this input by means of a level-controlled or edge-controlled contact.

A state machine is started every time this input is activated to ensure that another instruction "MB_MASTER" that uses the same instance DB can only issue a request when the current request has been processed. All other input states are recorded and saved internally for the current request until the reply has been received or an error has been detected.

If the same instance of "MB_MASTER" is executed again with REQ input = 1 before the current request is completely processed, no subsequent transfers are made. However, if the request has been processed, a new request is issued at the time when "MB_MASTER" is executed again with REQ input = 1.

DATA_ADDR and MODE parameters (S7-1200)

Description

You specify the start address for data access to the Modbus slave using the DATA_ADDR parameter.

With the MODE parameter and the Modbus address, you specify the function code to be transferred to the Modbus slave. The following table shows the relationship between the MODE parameter, the function code and Modbus address range.

MODE	Modbus function	Data length	Operation and data	Modbus address
0	01	1 to 2000 1 to 1992 (1)	Read output bits: 1 to (1992 or 2000) bits per query	1 to 9999
0	02	1 to 2000 1 to 1992 (1)	Read input bits: 1 to (1992 or 2000) bits per query	10001 to 19999
0	03	1 to 125 1 to 124 (1)	Read holding register: 1 to (124 or 125) WORD per query	40001 to 49999 or 400001 to 465535
0	04	1 to 125 1 to 124 (1)	Read input WORD: 1 to (124 or 125) WORD per query	30001 to 39999
1	05	1	Writing an output bit: One bit per query	1 to 9999
1	06	1	Writing a holding register: 1 WORD per query	40001 to 49999 or 400001 to 465535
1	15	2 to 1968 2 to 1960 (1)	Writing multiple output bits: 2 to (1960 or 1968) bits per query	1 to 9999
1	16	2 to 123 2 to 122 (1)	Writing multiple holding registers: 2 to (122 or 123) WORD per query	40001 to 49999 or 400001 to 465535
2	15	1 to 1968 2 to 1960 (1)	Writing one or more output bits: 1 to (1960 or 1968) bits per query	1 to 9999
2	16	1 to 123 2 to 122 (1)	Writing one or more holding registers: 1 to (122 or 123) WORD per query	40001 to 49999 or 400001 to 465535
11	11	0	Reading out the communications status word of the slaves and the event counter: The status word indicates execution of the instruction (0: is not executing; 0xFFFF: is executing). The event counter is incremented each time a message is transferred successfully. The DATA_ADDR and DATA_LEN parameters of the "MB_MASTER" instruction are ignored with this function.	-
80	08	1	Checking the slave status by reading the error code (0x0000): 1 WORD per query	-
81	08	1	Resetting the event counter of the slave with the diagnostics code 0x000A: 1 WORD per query	-
3 to 10, 12 to 79, 82 to 2555			Reserved	-
(1) For the "Extended address range", the maximum data length is reduced by one byte or one WORD depending on which data type is used for the function.				

Parameter DATA_PTR (S7-1200)

Description

The DATA_PTR parameter is a pointer to a data block or bit memory from which the data should be written or read. If you use a data block, create a global data block with the "Standard - compatible with S7-300/400" access type.

Data block structures for the DATA_PTR parameter

- These data types are valid for **reading of words** of Modbus addresses 30001 to 39999, 40001 to 49999, and 400001 to 465536 and also for **writing of words** to Modbus addresses 40001 to 49999 and 400001 to 465536.
  - Standard array of WORD, UINT, or INT data types (see below).
  - Named WORD, UINT, or INT structure where each element has a unique name and 16 bit data type.
  - Named complex structure where each element has a unique name and 16 bit or 32 bit data type.
- For reading and writing of bits of Modbus addresses 00001 to 09999 and 10001 to 19999.
  - Standard array of Boolean data types.
  - Named Boolean structure of uniquely named Boolean variables.
- Although not required, it is recommended that each "MB_MASTER" instruction has its own separate memory area in a global data block. The reason for this recommendation is that there is a greater possibility of data corruption if multiple "MB_MASTER" instructions are reading and writing the same area of a global data block.
- The memory areas for DATA_PTR do not need to be in the same global data block. You can create one data block with multiple areas for Modbus read operations, one data block for Modbus write operations, or one data block for each slave station.

Instance DB of the "MB_MASTER" instruction (S7-1200)

Static variables of the instance DB

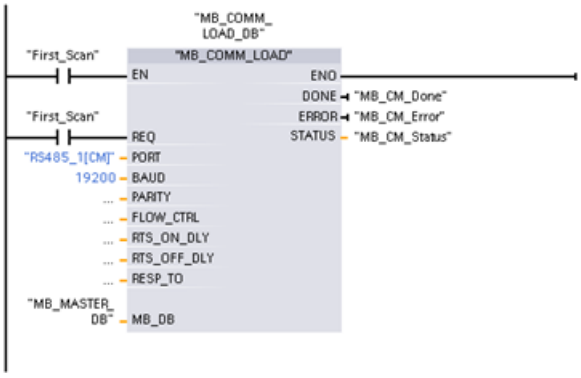
The following table describes the static variables of the instance DB of the instruction that you can use in the user program.

Variable	Data type	Description
MB_STATE	UINT	Internal status of the Modbus instruction
BLOCKED_ PROC_TIMEOUT	REAL	Time between completion of the instruction call and resetting the ACTIVE bit in the instance DB. The time buffer is used to avoid execution of the instruction being terminated before a job has been sent completely. The default time is 500 ms.
EXTENDED_ AD-DRESSING	BOOL	Configuring addressing: • 0: Default address area (1 byte) • 1: Extended address area (2 bytes) For additional information, refer to the section EXTENDED_AD-DRESSING: Instance DB of the "MB_SLAVE" instruction (Page 4631)

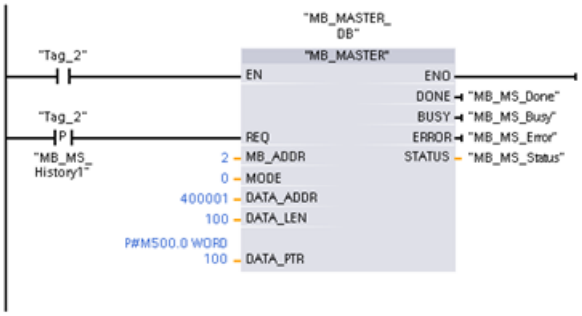
Sample program for a Modbus master (S7-1200)

Networks (LAD)

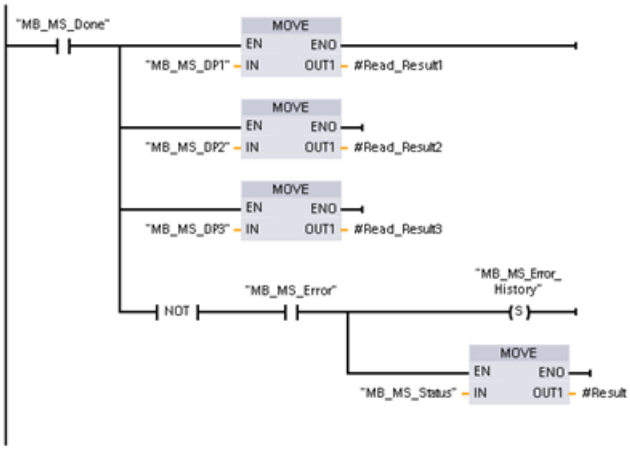
Network 1: Initialize parameters of the RS-485 module only once during the first cycle.



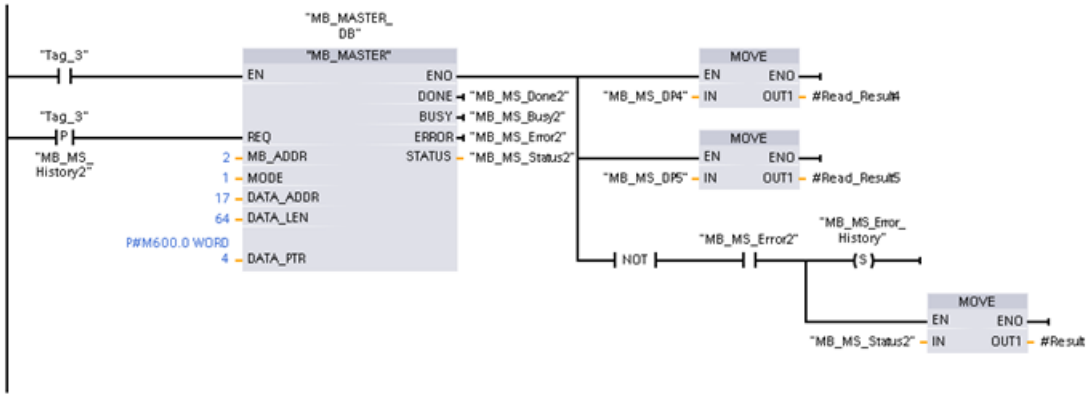
Network 2: Read 100 words from the holding register of the slave.



Network 3: This is an optional network that displays the values of the first 3 words as soon as the read operation has executed.



Network 4: Write 64 bits to the process image output, starting at slave address Q2.0.



MB_SLAVE: Communicate via the PtP port as Modbus slave (S7-1200)

Description of MB_SLAVE (S7-1200)

Note

Use with CM1241

The use of this instruction with a CM1241 is possible only as of firmware version V2.1 of the module.

Description

The "MB_SLAVE" instruction allows your program to communicate as a Modbus slave using a port on a point-to-point module (PtP) or a communications board (CB). A Modbus RTU master can issue a request and then your program responds via "MB_SLAVE" execution.

You must assign a unique instance data block when you insert the "MB_SLAVE" instruction in your program. This instance data block is used when you specify it at the MB_DB parameter of the "MB_COMM_LOAD (Page 4617)" instruction.

Modbus communication function codes (1, 2, 4, 5, and 15) can read and write bits and words directly in the process image input and process image output in the target system. The following table shows the mapping of Modbus addresses to the process image in the CPU.

Modbus functions of "MB_SLAVE"						S7-1200	
Codes	Function	Data area	Address range			Data area	CPU address
01	Read bits	Output	1	to	8192	Process image output	Q0.0 to Q1023.7
02	Read bits	Input	10001	to	18192	Process image input	I0.0 to I1023.7
04	Read words	Input	30001	to	30512	Process image input	IW0 to IW1022
05	Write bit	Output	1	to	8192	Process image output	Q0.0 to Q1023.7
15	Write bits	Output	1	to	8192	Process image output	Q0.0 to Q1023.7

Modbus communication function codes (function codes 3, 6, 16) use a separate holding register. To do this, you can use bit memory or a data block with the "Standard - compatible with S7-300/400" access type.

You specify the type of the holding register using the MB_HOLD_REG parameter of the MB_SLAVE instruction. The following table shows the mapping of the Modbus holding register to the DB address of MB_HOLD_REG in the target system.

Modbus functions of "MB_SLAVE"				S7-1200	
Codes	Function	Data area	Address range (WORD number)	Address in the DB (BYTE number)	Bit memory address (BYTE number)
03	Read words	Holding register	40001 to 49999 or	DW0 to DW19998 or	MW0 to CPU limit
			400001 to 465535	DW0 to DW131068	
06	Write word	Holding register	40001 to 49999 or	DW0 to DW19998 or	
			400001 to 465535	DW0 to DW131068	
16	Write words	Holding register	40001 to 49999 or	DW0 to DW19998 or	
			400001 to 465535	DW0 to DW131068	

The table below shows the supported Modbus diagnostic functions.

S7-1200 "MB_SLAVE" Modbus diagnostic functions		
Codes	Subfunction	Description
08	0000H	Return query data echo test: The "MB_SLAVE" instruction returns the echo of a received data word to a Modbus master.
08	000AH	Clear communication event counter: The "MB_SLAVE" instruction clears the communication event counter that is used for Modbus function 11.
11	-	Get communication event counter: The "MB_SLAVE" instruction uses an internal communication event counter for recording the number of successful Modbus read and write requests that are sent to the Modbus slave. The counter is not incremented on any Function 8, Function 11, or broadcast requests. It is also not incremented on any requests that result in a communication error (for example, parity or CRC errors).

The "MB_SLAVE" instruction supports broadcast write requests from Modbus masters as long as the requests include access to valid addresses.

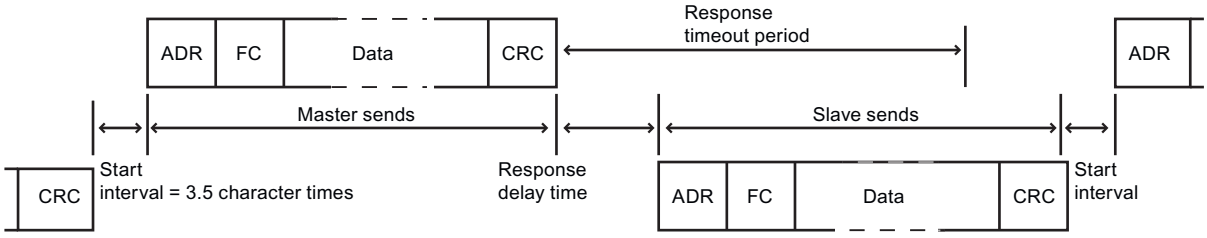
Regardless of the validity of a request, "MB_SLAVE" gives no response to a Modbus master as the result of a broadcast request.

Rules of Modbus slave communication

- "MB_COMM_LOAD" must be executed to configure a port, before the "MB_SLAVE" instruction can communicate with this port.
- If a port is to respond as a slave to a Modbus master, then that port cannot be used by "MB_MASTER (Page 4620)". Only one instance of "MB_SLAVE" can be used with a given port.
- The Modbus instructions do not use communication interrupt events to control the communication process. Your program must control the communication process by polling the "MB_SLAVE" instruction for completed send and receive operations.
- The "MB_SLAVE" instruction must be executed periodically at a rate that allows it to make a timely response to incoming requests from a Modbus master. It is therefore advisable to call the instruction in a cyclic program OB. Calling the "MB_SLAVE" instruction in an interrupt OB is possible but not advisable since it can lead to long delays in execution.

Frequency of execution of "MB_SLAVE"

The "MB_SLAVE" instruction must be executed periodically to receive each request from the Modbus master and to respond as required. The frequency of execution of "MB_SLAVE" is dependent upon the specified response timeout period of the Modbus master. This is illustrated in the following diagram.



The response timeout period is the amount of time a Modbus master waits for the start of a response from a Modbus slave. This time period is not defined by the Modbus protocol, but rather by a parameter of each Modbus master. The frequency of execution (time between one execution and the next execution) of "MB_SLAVE" must be based on the particular parameters of your Modbus master. As a minimum, you should execute "MB_SLAVE" twice within the response timeout period of the Modbus master.

Parameter

The following table shows the parameters of the "MB_SLAVE" instruction:

Parameter	Declaration	Data type	Memory area	Description
MB_ADDR	Input	V1.0: USINT V2.0: UINT	I, Q, M, D, L or constant	Station address of the Modbus slave (address range: 0 to 255)
MB_HOLD_REG	Input	VARIANT	D	Pointer to the Modbus holding register DB. The DB must be created with the "Standard - compatible with S7-300/400" access type.
NDR	Output	BOOL	I, Q, M, D, L	New data ready: • 0: No new data • 1: Indicates that new data has been written by the Modbus master
DR	Output	BOOL	I, Q, M, D, L	Read data: • 0: No data read • 1: Indicates that data has been read by the Modbus master
ERROR	Output	BOOL	I, Q, M, D, L	• 0: No error detected • 1: Error, a corresponding error code is output in the STATUS
STATUS	Output	WORD	I, Q, M, D, L	Error code

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

STATUS parameter

STATUS* (W#16#....)	Description	
80C8	The specified response timeout (refer to RCVTIME or MSGTIME) is "0".	
80D1	The receiver issued a flow control request to suspend an active transmission and never re-enabled the transmission within the wait time. This error is also generated during hardware flow control if the recipient does not detect CTS within the wait time.	
80D2	The send request was aborted because no DSR signal is received from the DCE.	
80E0	The message was terminated because the receive buffer is full	
80E1	The message was terminated as a result of a parity error	
80E2	The message was terminated as a result of a message frame error	
80E3	The message was terminated as a result of a overrun error	
80E4	The message was terminated as a result of the specified length exceeding the total buffer size	
8180	Invalid value for the port ID.	
8186	Invalid Modbus station address	
8187	Invalid pointer to MB_HOLD_REG-DB	
818C	Pointer to a type safe DB type MB_HOLD_REG (must be a Classic DB type)	
Response code sent to Modbus master (B#16#...)		
8380	No response	CRC error
8381	01	Function code not supported or not supported in a broadcast
8382	03	Data length error
8383	02	Error in the data address or address outside the valid range of MB_HOLD_REG
8384	03	Data value error
8385	03	Data diagnostic code value not supported (function code 08)
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".		

Instance DB of the "MB_SLAVE" instruction (S7-1200)

Static variables of the instance DB

The following table describes the static variables of the instance DB of the instruction that you can use in the user program. Your program can write values to the HR_Start_Offset and Extended_Addressing variables and control the Modbus slave operations.

The other variables can be read to monitor the Modbus status.

Variable	Data type	Description
HR_Start_Offset	WORD	Start address of the Modbus holding register (default = " 0")
Extended_ Addressing	BOOL	Configuring addressing: • 0: Default address area (1 byte) • 1: Extended address area (2 bytes)
Request_Count	WORD	Total number of queries received by the slave

Variable	Data type	Description
Slave_Message_Count	WORD	Number of queries sent to this specific slave
Bad_CRC_Count	WORD	Number of received queries with CRC errors
Broadcast_Count	WORD	Number of received broadcast queries
Exception_Count	WORD	Number of Modbus-specific errors that require the return of an exception
Success_Count	WORD	The number of requests received for this specific slave without protocol errors

HR_Start_Offset

The addresses of the Modbus holding register start at 40001 or 400001. These addresses correspond to the start address of the holding register in the target system memory. Using the HR_Start_Offset variable, you can set the offset to a different start address.

Example: A holding register starts at MW 100 and has a length of 100 WORD. With an offset of 20 in the HR_Start_Offset parameter, the holding register begins at address 40021 instead of 40001. Each address below 40021 and above 400119 causes an addressing error.

	HR_Start_Offset = 0			HR_Start_Offset = 20	
	Modbus word address	S7-1200 byte address		Modbus word address	S7-1200 byte address
Minimum	40001	MW100		40021	MW100
Maximum	40099	MW198		40119	MW198

Extended_Addressing

To address the Modbus slave, a single byte (default address range) or a double byte (extended address range) can be configured. Extended addressing is used to address more than 247 devices in a single network. If you decide on extended addressing, you can address a maximum of 64,000 addresses. Below, you will see a frame of Modbus function 1 as an example.

Table 11-85 Slave address with one byte (byte 0)

Function 1	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	
Request	Slave address	F code	Start address		Length of the coils		
Valid response	Slave address	F code	Length	Coil data			
Error response	Slave address	0x81	E code				

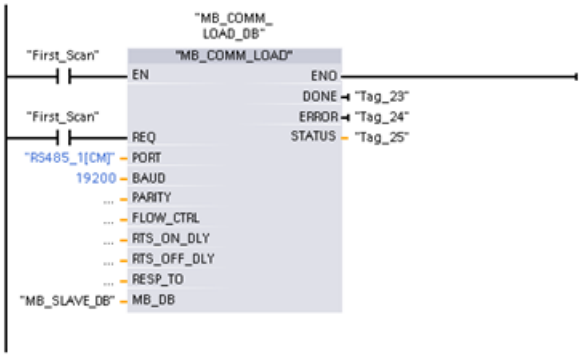
Table 11-86 Slave address with two bytes (byte 0 and byte 1)

Function 1	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
Request	Slave address		F code	Start address		Length of the coils	
Valid response	Slave address		F code	Length	Coil data		
Error response	Slave address		0x81	F code			

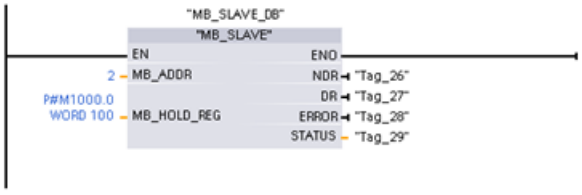
Sample program for a Modbus slave (S7-1200)

Networks (LAD)

Network 1: Initialize parameters of the RS-485 module only once during the first cycle.



Network 2: Check the requests of the Modbus master in every cycle. 100 words starting at MW1000 are configured for the Modbus holding register.



MODBUS (TCP) (S7-1200, S7-1500)

MODBUS (TCP) for the library versions V2.1 and V3.x of the S7-1200 CPUs (S7-1200)

MB_CLIENT: Communicating via PROFINET as a Modbus TCP client (S7-1200)

Description of MB_CLIENT (S7-1200)

Description

The "MB_CLIENT" instruction communicates as a Modbus TCP client via the PROFINET connection of the S7-1200 CPU. To use the instruction, you do not require any additional hardware module. With the "MB_CLIENT" instruction, you establish a connection between the client and the server, send requests and receive responses and control connection termination of the Modbus TCP server.

Parameters

The following table shows the parameters of the "MB_CLIENT" instruction:

Parameter	Declaration	Data type	Description
REQ (Page 4637)	Input	BOOL	Communication request with the Modbus TCP server The REQ parameter is level-controlled. This means that as long as the input is set (REQ=true), the instruction sends communication requests. • The instance DB for other clients is blocked with the communications request. • Changes to the input parameters will not become effective until the server has responded or an error message has been output. • If the parameter REQ is set again during an ongoing Modbus request, no additional transmission takes place afterwards.
DISCONNECT (Page 4637)	Input	BOOL	With this parameter, you control the establishment and termination of the connection to the Modbus server: • 0: Establish a communications connection to the specified IP address and port number. • 1: Disconnect the communication connection. No other function is executed during connection termination. The value 7003 is output at the STATUS parameter after successful connection termination. The request is sent immediately if the REQ parameter is set during connection establishment.
CONNECT_ID	Input	UINT	Unique ID to identify the connection. Each instance of the instructions "MB_CLIENT" and "MB_SERVER (Page 4643)" must be assigned a unique connection ID.
IP_OCTET_1	Input	USINT	1. Octet of the IP address* of the Modbus TCP server.
IP_OCTET_2	Input	USINT	2. Octet of the IP address* of the Modbus TCP server.

Parameter	Declaration	Data type	Description
IP_OCTET_3	Input	USINT	3. Octet of the IP address* of the Modbus TCP server.
IP_OCTET_4	Input	USINT	4. Octet of the IP address* of the Modbus TCP server.
IP_PORT	Input	UINT	IP port number of the server to which the client establishes the connection and communicates using the TCP/IP protocol (default value: 502).
MB_MODE (Page 4637)	Input	USINT	Selects the mode of the request (read, write or diagnostics).
MB_DATA_ADDR (Page 4637)	Input	UDINT	Start address of the data accessed by the "MB_CLIENT" instruction.
DATA_LEN	Input	UINT	Data length: Number of bits or words for the data access (see "MB_MODE and MB_DATA_ADDR parameters" - data length).
MB_DATA_PTR (Page 4638)	InOut	VARIANT	Pointer to the Modbus data register: The register is a buffer for the data received from the Modbus server or to be sent to the Modbus server. The pointer must reference a global data block with standard access. The number of bits addressed must be divisible by 8.
DONE	Out	BOOL	The bit at output parameter DONE is set to "1" as soon as the last job is completed without errors.
BUSY	Out	BOOL	• 0: No "MB_CLIENT " job in progress • 1: "MB_ CLIENT " job in progress
ERROR	Out	BOOL	• 0: No error • 1: Error occurred. The cause of error is indicated by the STATUS parameter.
STATUS (Page 4639)	Out	WORD	Error code of the instruction.
* 8-bit long component of the 32-bit IPv4 IP address of the Modbus TCP server.			

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

Consistent input data during an "MB_CLIENT" call

When a Modbus client calls a Modbus instruction, the status of the input parameters is stored internally and then compared at the next call. The comparison is used to determine whether this particular call initialized the current request. Several "MB_CLIENT" calls can be executed if you use a common instance DB. The values of the input parameters must not be changed while an "MB_CLIENT" instance is being executed. If the input parameters are changed during execution, "MB_CLIENT" cannot be used to check whether or not the instance is currently being executed.

Multiple client connections

A Modbus TCP client can support several TCP connections and the maximum number of connections depends on the CPU being used. The total number of connections of one CPU, including those of the Modbus TCP clients and server must not exceed the maximum number of supported connections. Modbus TCP connections can also be shared by client and/or server connections.

With individual client connections, remember the following rules:

- Each "MB_CLIENT" connection must use a unique instance DB.
- For each "MB_CLIENT" connection, a unique server IP address must be specified.
- Each "MB_CLIENT" connection requires a unique connection ID.
  The relevant individual connection ID must be used for each individual instance DB of the instruction. The connection ID and instance DB belong together in pairs and must be unique for each connection.
- Unique numbers for the IP port may or may not be required depending on the server configuration.

Static tags of the instruction

The following table describes the editable static tags of the instance data block of the "MB_CLIENT" instruction.

Tag	Data type	Start value	Description
Blocked_Proc_Time-out	REAL	3.0	Wait time in seconds before the static tag ACTIVE is reset if there is a blocked Modbus instance. This can, for example, occur if a client request is output and the execution of the client function aborts before the request was fully executed. The maximum wait time is 55 seconds.
MB_Transaction_ID	WORD	1	Transaction ID of the Modbus TCP protocol. The start value of "1" should only be changed if the Modbus TCP server requires a different value.
MB_Unit_ID	BYTE	255	Modbus device detection: A Modbus TCP server is addressed using its IP address. For this reason, the MB_UNIT_ID parameter is not used in the case of Modbus TCP addressing. The MB_UNIT_ID parameter corresponds to the field of the slave address in the Modbus RTU protocol. If a Modbus TCP server is used as a gateway for a Modbus RTU protocol, the slave device in the serial network can be identified using MB_UNIT_ID. The MB_UNIT_ID parameter would in this case forward the request to the correct Modbus RTU slave address. Please note that some Modbus TCP devices may require the MB_UNIT_ID parameter for the initialization within a limited value range.
RCV_TIMEOUT	REAL	2.0	Time interval in seconds in which the "MB_CLIENT" instruction waits for a response from the server.
Connected	BOOL	0	Indicates whether the connection to the assigned client has been established or not: 1 = connected, 0 = not connected.

See also

MB_CLIENT example 1: Send several requests via a TCP connection (Page 4649)

MB_CLIENT example 2: Send multiple requests via several TCP connections (Page 4650)

MB_CLIENT example 3: Coordinate several requests (Page 4651)

Difference between synchronous and asynchronous instructions (Page 2571)

REQ and DISCONNECT parameters (S7-1200)

Description

If no instance of the "MB_CLIENT" instruction is executing and if the value of the DISCONNECT parameter is "0", a new job executes if REQ=1. If there is not yet a connection, this is established during execution.

If the same instance of the "MB_CLIENT" instruction executes again (DISCONNECT = 0 and REQ = 1), before the active job was executed, this is not executed on completion of the active job. A new job can only be started on completion of the active job (REQ = 1).

You can monitor the status of execution with the DONE parameter. You can use this to monitor the status of execution if the "MB_CLIENT" instruction is executed sequentially.

See also

Description of MB_CLIENT (Page 4634)

MB_MODE, MB_DATA_ADDR and DATA_LEN parameters (S7-1200)

Description

The combination of the MB_MODE, MB_DATA_ADDR and DATA_LEN parameters defines the Modbus function code used in the current Modbus message for the values 0, 1 and 2 of MB_MODE:

- MB_MODE contains the information on whether to read or to write.
  MB_MODE=0: Read, MB_MODE=1 and 2: Writing
- MB_DATA_ADDR contains the information on what is to be read or written, as well as address information from which the "MB_CLIENT" instruction calculates the remote address.
- DATA_LEN contains the number of values to be read/written.

Examples:

- The combination MB_MODE=1, MB_DATA_ADDR=1, DATA_LEN=1 specifies the function code 05. 1 output bit is written starting from the remote address 0.
- The combination MB_MODE=1, MB_DATA_ADDR=1, DATA_LEN=2 specifies the function code 15. 2 output bits are written starting from the remote address 0.

The following table shows the relationship between the input parameters MB_MODE, MB_DATA_ADDR, DATA_LEN of the "MB_CLIENT" instruction and the associated Modbus function.

MB_MODE	MB_DATA_ADDR	DATA_LEN	Modbus function	Function and data type
0	1 to 9,999	1 to 2,000	01	Read 1 to 2,000 output bits on the remote address 0 to 9,998
0	10,001 to 19,999	1 to 2,000	02	Read 1 to 2,000 input bits on the remote address 0 to 9,998
0	40,001 to 49,999	1 to 125	03	Read 1 to 125 holding registers on the remote address 0 to 9,998
0	30,001 to 39,999	1 to 125	04	Read 1 to 125 input words on the remote address 0 to 9,998
1	1 to 9,999	1	05	Write 1 output bit on the remote address 0 to 9,998
1	40,001 to 49,999	1	06	Write 1 holding register on the remote address 0 to 9,998
1	1 to 9,999	2 to 1,968	15	Write 2 to 1,968 output bits on the remote address 0 to 9,998
1	40,001 to 49,999	2 to 123	16	Write 2 to 123 holding registers on the remote address 0 to 9,998
2	1 to 9,999	1 to 1,968	15	Write 1 to 1,968 output bits on the remote address 0 to 9,998
2	40,001 to 49,999	1 to 123	16	Write 1 to 123 holding registers on the remote address 0 to 9,998
11	The MB_DATA_ADDR and DATA_LEN parameters are not evaluated when this function is executed.		11	Read status word and event counter of the server: • The status word reflects the the processing status (0 - not processing, 0xFFFF - processing). • The event counter is incremented when the Modbus request was executed successfully. If an error occurred during execution of a Modbus function, a message is sent by the server but the event counter is not incremented.
80	-	1	08	Check the server status with the diagnostic code 0x0000 (return loop test - the server sends the request back): • 1 WORD per call
81	-	1	08	Reset the event counter of the server with the diagnostic code 0x000A: • 1 WORD per call
3 to 10, 12 to 79, 82 to 255				Reserved

See also

Description of MB_CLIENT (Page 4634)

MB_DATA_PTR parameter (S7-1200)

Description

The MB_DATA_PTR parameter is a pointer to a data buffer for storing the data read from or written to the Modbus server. As the data buffer, you can use a global data block or a memory area (M).

For a buffer in the memory area (M), use a pointer in the ANY format as follows: "P#bit address" "data type" "length" (example: P#M1000.0 WORD 500).

The MB_DATA_PTR parameter uses a communications buffer:

- For the communication functions of the "MB_CLIENT" instruction:
  - Reading and writing of 1 bit of data of the Modbus server addresses 00001 to 09999 and 10001 to 19999.
  - Reading of 16-bit WORD data of the Modbus server addresses 30001 to 39999 and 40001 to 49999.
  - Writing 16-bit WORD data of the Modbus server addresses 40001 to 49999.
- During data transmission (length: bit or WORD) from or to the global DB or the memory area (M) that you assigned with the MB_DATA_PTR parameter.

If you use a data block for the buffer in the MB_DATA_PTR parameter, you will need to assign data types to the DB elements.

- Use the 1-bit data type BOOL for a Modbus bit address
- Use a 16-bit data type such as WORD, UINT, INT or REAL for a Modbus WORD address.
- Use a 32-bit data type (double word) such as DWORD, DINT or REAL for two Modbus WORD addresses.
- With MB_DATA_PTR, you can also access complex DB elements such as:
  - Standard arrays
  - Structures with unique element names
  - Complex structures with unique naming of the elements and data type lengths of 16 or 32 bits.
- The data areas for the MB_DATA_PTR parameter can also be in different global data blocks (or in different memory areas). You can, for example, use a data block for the the read jobs and another one for the write jobs or a separate data block for each "MB_CLIENT" station.

See also

Description of MB_CLIENT (Page 4634)

Parameter STATUS (S7-1200)

Parameter STATUS (general status information) (library version V2.1)

STATUS* (W#16#)	Description
0000	Instruction executed without errors.
7000	No call active (REQ=0).
7001	First call with REQ=1: Processing started; BUSY has the value 1.

STATUS* (W#16#)	Description
7002	Intermediate call (REQ irrelevant). Processing already active; BUSY has the value 1.
7003	Connection is being terminated.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Parameter STATUS (general status information) (library version V3.x)

STATUS* (W#16#)	Description
0000	Instruction executed without errors.
0001	Connection established.
0003	Connection terminated.
7000	No call active (REQ=0).
7001	First call with REQ=1: Processing started; BUSY has the value 1.
7002	Intermediate call (REQ irrelevant). Processing already active; BUSY has the value 1.
7003	Terminating connection.
7004	Connection established and monitored. No job processing active.
7005	Data was sent.
7006	Data was received.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Parameter STATUS (protocol error)

STATUS* (W#16#)	Local and/or remote errors	Error code in the answer from MB_SERVER (B#16#)	Description
80C8	Local	-	No response of the server in the defined period. Check the connection to the Modbus server. This error is only reported on completion of the configured repeated attempts. If the "MB_CLIENT" instruction does not receive an answer with the originally transferred transaction ID (MB_TRANSACTION_ID tag) within the defined period, this error code is output.
8380	Local	-	Received Modbus frame has incorrect format or too few bytes were received.
8381	Remote	01	Function code is not supported.

STATUS* (W#16#)	Local and/or remote errors	Error code in the answer from MB_SERVER (B#16#)	Description
8382	Local	-	• The length of the Modbus frame in the frame header does not match the number of received bytes. • The number of bytes does not match the number of actually transmitted bytes (only functions 1-4). For example, this is the case when "MB_CLIENT" requests an odd number of words, but "MB_SERVER" always sends an even number of words. • The start address in the received frame does not match the saved start address (functions 5, 6, 15, 16). • The number of words does not match the number of actually transmitted words (functions 15 and 16).
	Remote	03	Invalid length specification in received Modbus frame. Check the server side.
8383	Local	-	Error reading or writing data or access outside the address area of MB_DATA_PTR (Page 4638).
	Remote	02	Error reading or writing data or access outside the address area of the server
8384	Local	-	• Invalid exception code received. • A different data value was received than was originally sent by the client (functions 5, 6 and 8). • Invalid status value received (function 11)
	Remote	03	Error in data value for function 5
8385	Local	-	• Diagnostics code not supported. • A different subfunction code was received than was originally sent by the client (function 8).
	Remote	03	Diagnostics code not supported
8386	Local	-	Received function code does not match the one sent originally.
8387	Local	-	• The assigned connection ID is different from that used for previous requests. Only one connection ID can be used for each instance DB of the "MB_CLIENT" instruction. • The error code is also output when the ID of the Modbus TCP protocol received by the server is not "0".
8388	Local	-	The Modbus server sent a different data length than was requested. This error occurs only when using the Modbus functions 15 or 16.
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".			

Parameter STATUS (parameter error)

In addition to the errors listed in the following table, errors are also possible with the "MB_CLIENT" instruction caused by the communications instructions ("TCON", "TDISCON", "TSEND" and "TRCV").

STATUS* (W#16#)	Description
80BB	Invalid value at ACTIVE_EST parameter (identifier for the type of connection establishment, see T_CON_PARAM): • Only passive connection establishment permitted for server (ACTIVE_EST = FALSE). • Only active connection establishment permitted for client (ACTIVE_EST = TRUE).
8188	The MB_MODE parameter has an invalid value.
8189	Invalid addressing of data at the MB_DATA_ADDR parameter.
818A	Invalid data length at the MB_DATA_LEN parameter.
818B	The MB_DATA_PTR parameter has an invalid pointer. You should also check the values of the MB_DATA_ADDR (Page 4637) and MB_DATA_LEN parameters.
818C	• The MB_DATA_PTR (Page 4634) pointer references an an optimized data block. Either use a data block with standard access or a memory area. • Timeout at parameter BLOCKED_PROC_TIMEOUT (see static tags of instruction). The limit of 55 seconds has been exceeded.
8200	• A further Modbus request is currently being processed via the port. • Another instance of MB_CLIENT with the same connection parameters is processing an existing Modbus request.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Note

Error codes of internally used communications instructions.

With the "MB_CLIENT" instruction, in addition to the errors listed in the tables, errors caused by the communication instructions ("TCON", "TDISCON", "TSEND" and "TRCV") used by the instruction can occur.

The error codes are assigned via the instance data block of the "MB_CLIENT" instruction. The error codes are displayed for the respective instruction under STATUS in the Static section.

The meaning of the error codes is available in the documentation of the corresponding communications instruction.

See also

MB_HOLD_REG parameter (Page 4646)

MB_SERVER: Communicating via PROFINET as a Modbus TCP server (S7-1200)

Description of MB_SERVER (S7-1200)

Description

The "MB_SERVER" instruction communicates as a Modbus TCP server via the PROFINET connection of the S7-1200 CPU. To use the instruction, you do not require any additional hardware module. The "MB_SERVER" instruction processes connection requests of a Modbus TCP client, receives requests from Modbus functions and sends responses.

NOTICE
Security information Note that each client of the network is given read and write access to the process image inputs and outputs and to the data block or bit memory area defined by the Modbus holding register. The option is available to restrict access to an IP address to prevent unauthorized read and write operations. Note, however, that the shared address can also be used for unauthorized access.

Parameters

The following table shows the parameters of the "MB_SERVER" instruction:

Parameter	Declaration	Data type	Description
DISCONNECT	Input	BOOL	The instruction "MB_SERVER" enters into a passive connection with a partner module, which means the server responds to a TCP connection request from each requesting IP address. You can use this parameter to control when a connection request is accepted: • 0: A passive connection is established when there is no communications connection. • 1: Initialization of the connection termination. If the input is set, no other operations are executed. The value 7003 is output at the STATUS parameter after successful connection termination.
CONNECT_ID	Input	UINT	The parameter uniquely identifies a connection within the CPU. Each individual instance of the instructions "MB_CLIENT (Page 4634)" and "MB_SERVER" must have a unique CONNECT_ID parameter.
IP_PORT	Input	UINT	Start value=502. The number of the IP port defines which IP port is monitored for connection requests of the Modbus client. The following TCP port numbers must not be used for the passive connection of the "MB_SERVER" instruction: 20, 21, 25, 80, 102, 123, 5001, 34962, 34963 and 34964.
MB_HOLD_REG (Page 4646)	InOut	VARIANT	Pointer to the Modbus holding register of the "MB_SERVER" instruction: Use a global data block with standard access as holding register. The holding register contains the values that may be accessed by a Modbus client using the Modbus functions 3 (read), 6 (write) and 16 (read).

Parameter	Declaration	Data type	Description
NDR	Output	BOOL	"New Data Ready": • 0: No new data • 1: New data written by the Modbus client written
DR	Output	BOOL	"Data Read": • 0: No data read • 1: Data read by the Modbus client
ERROR	Output	BOOL	If an error occurs during the call of the "MB_SERVER" instruction, the output of the ERROR parameter is set to TRUE. Detailed information about the cause of the problem is indicated by the STATUS parameter.
STATUS (Page 4647)	Output	WORD	Error code of the instruction.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Mapping of Modbus addresses to the process image

The "MB_SERVER" instruction allows incoming Modbus functions (1, 2, 4, 5 and 15) direct read and write access to the process image input and output of the S7-1200 CPU (use of the data types BOOL and WORD).

For the data transfer of the function codes 3, 6 and 16, the holding register (MB_HOLD_REG parameter) must be defined longer than one byte. The following table shows the mapping of the Modbus addresses to the process image of the CPU.

Modbus function						S7-1200	
Code	Function	Data area	Address space			Data area	CPU address
01	Read: Bits	Output	0	to	8191	Process image output	Q0.0 to Q1023.7
02	Read: Bits	Input	0	to	8191	Process image input	I0.0 to I1023.7
04	Read: WORD	Input	0	to	1021	Process image input	IW0 to IW1022
05	Write: Bits	Output	0	to	8191	Process image output	Q0.0 to Q1023.7
15	Write: Bits	Output	0	to	8191	Process image output	Q0.0 to Q1023.7

Incoming Modbus alarms with the function codes 3, 6 and 16 write to or read from the Modbus holding registers (you specify the holding registers with the MB_HOLD_REG parameter).

Multiple server connections

You can create multiple Server connections. This allows a single CPU to establish connections to more than one Modbus TCP client at the same time.

A Modbus TCP server can support several TCP connections and the maximum number of connections depends on the CPU being used.

The total number of connections of one CPU, including those of the Modbus TCP clients and server must not exceed the maximum number of supported connections.

Modbus TCP connections can also be shared by client and/or server connections.

In the case of Server connections, remember the following rules:

- Each "MB_SERVER" connection must use a unique instance DB.
- Each "MB_SERVER" connection must be created with a unique IP port number. Only one connection is supported for each port.
- Each "MB_SERVER" connection must use a unique connection ID.
  The relevant individual connection ID must be used for each individual instance DB of the instruction. The connection ID and instance DB belong together in pairs and must be unique for each connection.
- For each connection, the "MB_SERVER" instruction must be called individually.

Modbus diagnostics functions

The table below contains a description of the Modbus diagnostics functions.

Code	Subfunction	Description
08	0x0000	Echo test: The "MB_SERVER" instruction receives a data word and returns this unchanged to the Modbus master.
08	0x000A	Reset event counter: The "MB_SERVER" instruction resets the event counter for communication that is used for Modbus function 11.
11	-	Fetch event counter of the communication: The "MB_SERVER" instruction uses an internal event counter for communication to record the number of successfully executed read and write requests sent to the Modbus server. The event counter is not incremented by the functions 8, 11 or broadcast requests. The same applies to requests that result in a communications error (for example parity errors or CRC errors). The broadcast function is not available for Modbus TCP because only one client/server connection can exist at any one time.

Static tags of the instruction

The following table describes the static tags of the instance data block of the "MB_SERVER" instruction used in the program. You can write the HR_Start_Offset tag. You can read the other tags to monitor the Modbus status.

Tag	Data type	Start value	Description
HR_Start_Offset	WORD	0	Assign the start address of the Modbus holding register.
Request_Count	WORD	0	Total number of requests received by the server.
Server_Message_Count	WORD	0	Total number of received alarms for the relevant server.
Xmt_Rcv_Count	WORD	0	Counter for detecting the number of transfers during which an error occurred. The counter is also incremented if an invalid Modbus alarm is received.
Exception_Count	WORD	0	Counter for detecting the number of errors specifically for Modbus cause an exception.
Success_Count	WORD	0	Counter for detecting the number of requests that contain no error in the transferred protocol.
Connected	BOOL	0	Indicates whether the connection to the assigned client has been established or not: 1 = connected, 0 = not connected.

Example: Addressing via static tag HR_Start_Offset

The addresses of the Modbus holding register start at 0 (from the perspective of the MB_CLIENT at 40.001). These addresses correspond to the address area of the CPU memory area for the holding register. You can also define the HR_Start_Offset tag so that the Modbus holding register has a start address other than 0.

Example: The holding register begins at MW100, and has a length of 100 WORD. An offset value in the HR_Start_Offset parameter means that the start address of the holding register is moved from 0 to 20. This causes an error whenever the holding register is addressed below the address 20 and above the address 119.

HR_Start_Offset	Address	Minimum	Maximum
0	Modbus address (WORD)	0	99
	CPU address	MW100	MW298
20	Modbus address (WORD)	20	119
	CPU address	MW100	MW298

See also

MB_SERVER example: Multiple TCP connections (Page 4652)

Difference between synchronous and asynchronous instructions (Page 2571)

MB_HOLD_REG parameter (S7-1200)

Description

The MB_HOLD_REG parameter is a pointer to a data buffer for storing the data read from or written to the Modbus server. As the data buffer, you can use a global data block or a memory area (M).

- The high limit for the number of addresses in the data block (D) is determined by the maximum work memory of the CPU.
- The high limit for the number of bit memories (M) is determined by the size of the CPU memory area.

The following table shows examples of mapping Modbus addresses to the holding register for the Modbus functions 3 (read WORD), 6 (write WORD), 16 (write several WORD) and 23 (write and read several words).

Modbus addresses	MB_HOLD_REG parameter - examples		
0	MW100	DB10.DBW0	"Recipe".ingredient[1]
1	MW102	DB10.DBW2	"Recipe".ingredient[2]
2	MW104	DB10.DBW4	"Recipe".ingredient[3]
3	MW106	DB10.DBW6	"Recipe".ingredient[4]
4	MW108	DB10.DBW8	"Recipe".ingredient[5]

See also

Description of MB_SERVER (Page 4643)

Parameter STATUS (S7-1200)

Parameter STATUS (general status information) (library version V2.1)

STATUS* (W#16#)	Description
0000	Instruction executed without errors.
7000	No call active (REQ=0).
7001	First call with REQ=1: Processing started; BUSY has the value 1.
7002	Intermediate call (REQ irrelevant). Processing already active; BUSY has the value 1.
7003	Connection is being terminated.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Parameter STATUS (general status information) (library version V3.x)

STATUS* (W#16#)	Description
0000	Instruction executed without errors.
0001	Connection established.
0003	Connection terminated.
7000	No call active (REQ=0).
7001	First call. Connection establishment triggered.
7002	Intermediate call. Establishing connection.
7003	Terminating connection.
7005	Data is being sent.
7006	Data is being received.
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Parameter STATUS (protocol error)

STATUS* (W#16#)	Local and/ or remote errors	Error code in the answer from MB_SERVER (B#16#)	Description
8380	Local	-	Received Modbus frame has incorrect format or too few bytes were received.
8381	Remote	01	Function code is not supported.
8382	Remote	03	Error in data length

STATUS* (W#16#)	Local and/or remote errors	Error code in the answer from MB_SERVER (B#16#)	Description
8383	Local	-	Error reading or writing data or access outside the address area of MB_HOLD_REG (Page 4646).
	Remote	02	Error reading or writing data or access outside the address area of the server
8384	Local	-	• Invalid exception code received. • A different data value was received than was originally sent by the server (functions 5, 6 and 8). • Invalid status value received (function 11)
	Remote	03	Error in data value for function 5
8385	Local	-	• Diagnostics code not supported. • A different subfunction code was received than was originally sent by the server (function 8).
	Remote	03	Diagnostics code not supported
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".			

Parameter STATUS (parameter error)

STATUS* (W#16#)	Description
80BB	Invalid value at ACTIVE_EST parameter (identifier for the type of connection establishment, see T_CON_PARAM): • Only passive connection establishment permitted for server (ACTIVE_EST = FALSE). • Only active connection establishment permitted for client (ACTIVE_EST = TRUE).
8187	The MB_HOLD_REG parameter has an invalid pointer. Data area is too small.
818C	• The pointer at parameter MB_HOLD_REG references an optimized data block. Either use a data block with standard access or a memory area. • Error due to timeout of execution (more than 55 seconds).
* The error codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Note

Error codes of internally used communications instructions.

With the "MB_SERVER" instruction, in addition to the errors listed in the tables, errors caused by the communication instructions ("TCON", "TDISCON", "TSEND" and "TRCV") used by the instruction can occur.

The error codes are assigned via the instance data block of the "MB_SERVER" instruction. The error codes are displayed for the respective instruction under STATUS in the Static section.

The meaning of the error codes is available in the documentation of the corresponding communications instruction.

See also

Description of MB_SERVER (Page 4643)

Examples (S7-1200)

MB_CLIENT example 1: Send several requests via a TCP connection (S7-1200)

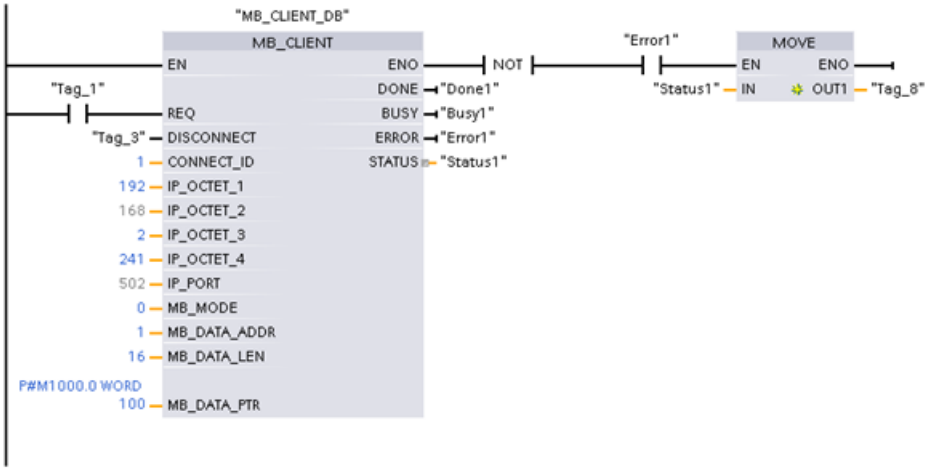
Description

Several requests of the Modbus Client can be sent via a TCP connection. To do this, use the same instance DB, the same connection ID and the same port number.

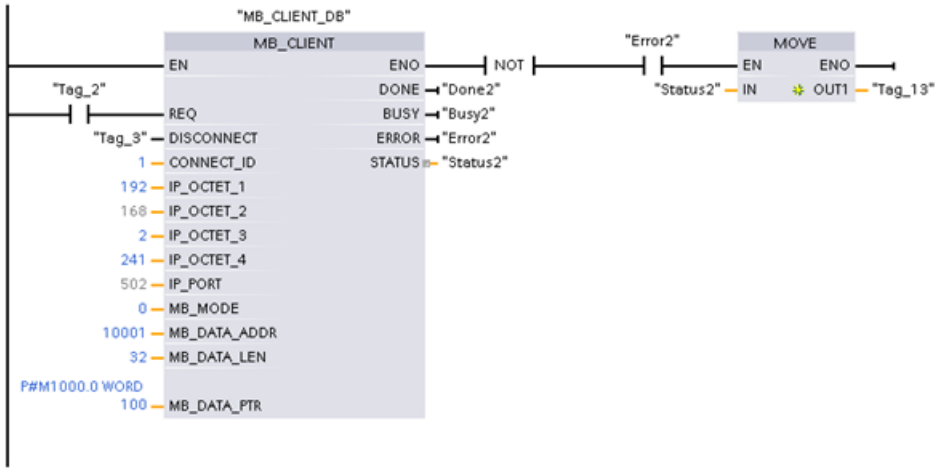
Only one client can be active at any one time. After processing of a client has been completed, the next client is processed. The order of execution must be defined in the program.

In the following sample program, the value of the STATUS output parameter is also copied.

Network 1: Modbus function 1 - 16 read output bits



Network 2: Modbus function 2 - 32 read input bits



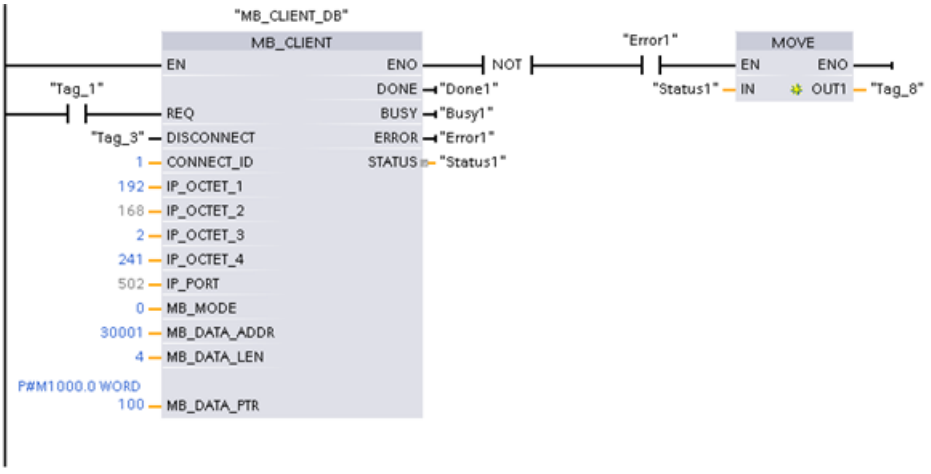
MB_CLIENT example 2: Send multiple requests via several TCP connections (S7-1200)

Description

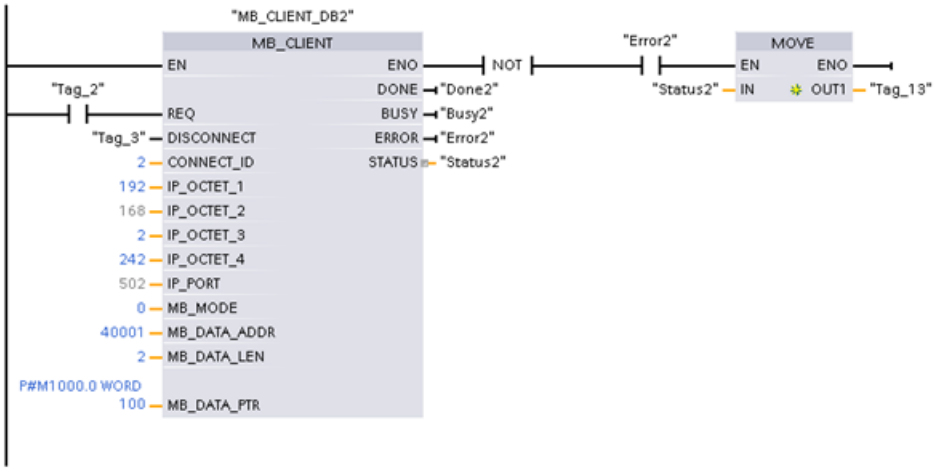
Requests from the Modbus client can be sent via different TCP connections. If you require this, use a different instance DB and a different connection ID.

Use a different port number, if the connections are to the same Modbus server. If the connections are to different Modbus servers, you can freely assign the port number.

Network 1: Modbus function 4 - read input (WORD)



Network 2: Modbus function 3 - read holding register (WORD)

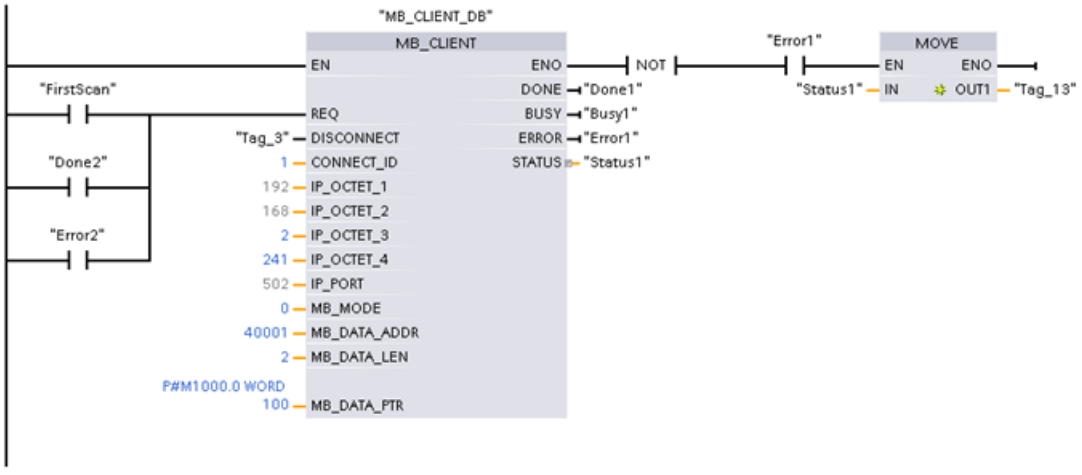


MB_CLIENT example 3: Coordinate several requests (S7-1200)

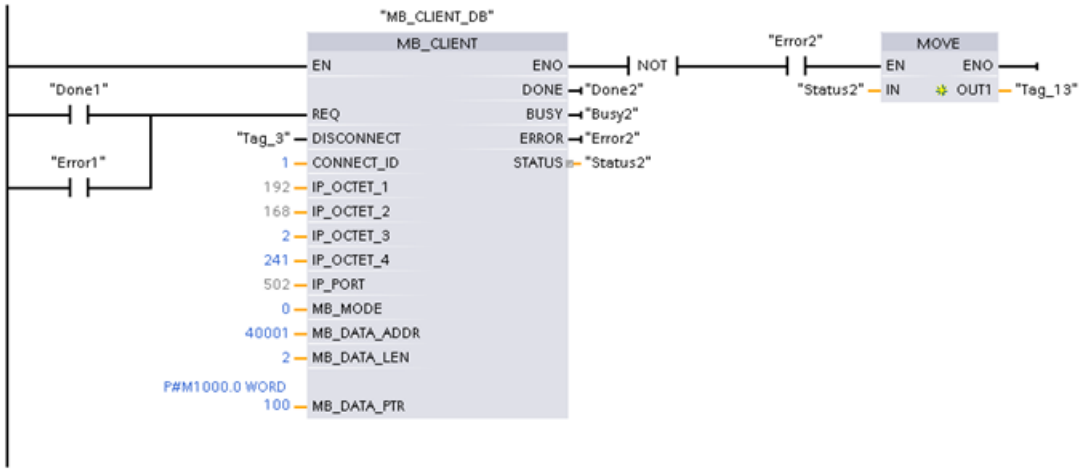
Description

Make sure that the individual Modbus requests are executed. You control coordination of requests with the program. The following example demonstrates how the output parameters of the first and second client request can be used to coordinate execution of the instructions.

Network 1: Modbus function 3 - read holding register (WORD)



Network 2: Modbus function 3 - read holding register (WORD)



MB_SERVER example: Multiple TCP connections (S7-1200)

Description

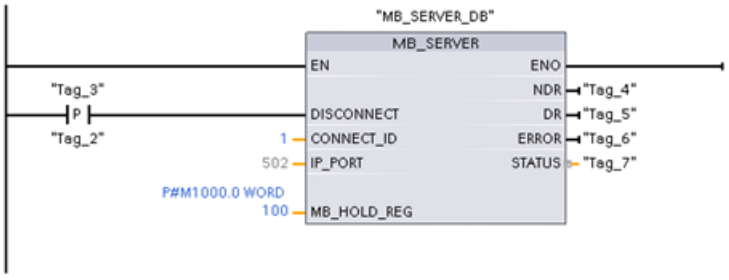
You can use several Modbus TCP server connections. To do this, the "MB_SERVER" instruction must be called independently for each connection.

Every connection requires the following:

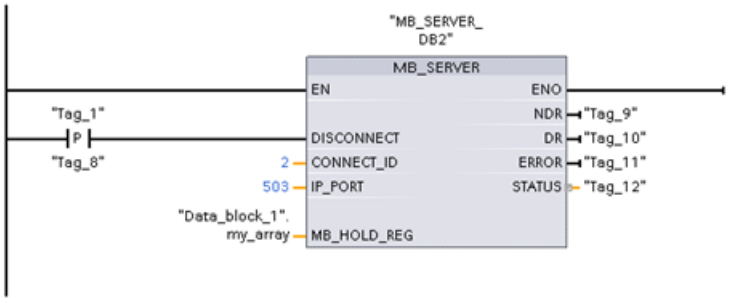
- An independent instance data block of the instruction
- A unique connection ID
- A separate IP port (on the S7-1200, only one connection is permitted per IP port)

To optimize the performance "MB_SERVER" should be executed once per program cycle for each connection.

Network 1: Connection #1 with associated IP port connection ID and instance DB



Network 2: Connection #1 with associated IP port connection ID and instance DB



MODBUS (TCP) for the library versions V4.0 and later of the S7-1200 CPUs and as of V3.x of the S7-1500 CPUs (S7-1200, S7-1500)

MB_CLIENT: Communicating via PROFINET as a Modbus TCP client (S7-1200, S7-1500)

Description of MB_CLIENT (S7-1200, S7-1500)

Description

The "MB_CLIENT" instruction communicates as a Modbus TCP client via the PROFINET connection. With the "MB_CLIENT" instruction, you establish a connection between the client and the server, send Modbus requests and receive responses and control connection termination of the Modbus TCP client.

For the S7-1200 with firmware version V4.0 you can use the "MB_CLIENT" instruction up to and including library version V3.1. With the S7-1200 as of firmware version V4.1 and the S7-1500, you can use the "MB_CLIENT" instruction of all library versions.

The connection can take place via the local interface of the CPU or CM/CP.

To use the instruction, you do not require an additional hardware module.

Multiple client connections

A Modbus TCP client can support several TCP connections and the maximum number of connections depends on the CPU being used. The total number of connections of one CPU, including those of the Modbus TCP clients and server must not exceed the maximum number of supported connections. Modbus TCP connections can also be shared by "MB_CLIENT" and/or "MB_SERVER" instances.

With individual client connections, remember the following rules:

- Each "MB_CLIENT" connection must use a unique instance DB.
- For each "MB_CLIENT" connection, a unique server IP address must be specified.

- Each "MB_CLIENT" connection requires a unique connection ID.
  The relevant individual connection ID must be used for each individual instance DB of the instruction. The connection ID and instance DB belong together in pairs and must be unique for each connection.
- Unique numbers for the IP port may or may not be required depending on the server configuration.

Parameters

The following table shows the parameters of the "MB_CLIENT" instruction:

Parameter	Declaration	Data type	Description
REQ (Page 4657)	Input	BOOL	Modbus query to the Modbus TCP server The REQ parameter is level-controlled. This means that as long as the input is set (REQ=true), the instruction sends communication requests. • After the Modbus query has started, the instance DB is locked for other clients. • Changes to the input parameters will not become effective until the server has responded or an error message has been output. • If the parameter REQ is set again during an ongoing Modbus request, no additional transmission takes place afterwards.
DISCONNECT (Page 4657)	Input	BOOL	With this parameter, you control the establishment and termination of the connection to the Modbus server: • 0: Establish communications connection to the connection partner configured at the CONNECT parameter (see CONNECT parameter). • 1: Disconnect the communication connection. No other function is executed during connection termination. The value 0003 is output at the STATUS parameter after successful connection termination. The Modbus request is sent immediately if the REQ parameter is set during connection establishment.
MB_MODE (Page 4658)	Input	USINT	Selects the mode of the Modbus request (read, write or diagnostics) or direct selection of a Modbus function
MB_DATA_ADDR (Page 4658)	Input	UDINT	depends on MB_MODE
MB_DATA_LEN	Input	UINT	Data length: Number of bits or words for the data access (see MB_MODE, MB_DATA_ADDR and MB_DATA_LEN parameters (Page 4658)).
MB_DATA_PTR (Page 4660)	InOut	VARIANT	Pointer to a data buffer for the data to be received from the Modbus server or to be sent to the Modbus server.

Parameter	Declaration	Data type	Description
CONNECT (Page 4661)	InOut	VARIANT	Pointer to the structure of the connection description The following structures (system data types) can be used: • TCON_IP_v4: Includes all address parameters that are required for establishing a programmed connection. When using TCON_IP_v4, the connection is established when calling the "MB_CLIENT" instruction. • TCON_Configured: Includes the address parameters of a configured connection. When using TCON_Configured, an existing connection is used that was created by the CPU after download of the hardware configuration.
DONE	Out	BOOL	The bit at output parameter DONE is set to "1" as soon as the last Modbus job is completed without errors.
BUSY	Out	BOOL	• 0: No Modbus request in progress • 1: Modbus request being processed The output parameter BUSY is not set during connection establishment and termination.
ERROR	Out	BOOL	• 0: No error • 1: Error occurred. The cause of error is indicated by the STATUS parameter.
STATUS (Page 4663)	Out	WORD	Detailed status information of the instruction.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

Consistent input data during an "MB_CLIENT" call

When a Modbus client instruction is called, the values of the input parameters are stored internally and then compared at the next call. The comparison is used to determine whether this particular call initialized the current request. Several "MB_CLIENT" calls can be executed if you use a common instance DB. The values of the input parameters must not be changed while an "MB_CLIENT" instance is being executed. If the input parameters are changed during execution, "MB_CLIENT" cannot be used to check whether or not the instance is currently being executed.

Static tags of the instruction

The following table describes the editable static tags of the instance data block of the "MB_CLIENT" instruction.

Tag	Data type	Start value	Description
Blocked_Proc_Time-out	REAL	3.0	Wait time in seconds before the static tag ACTIVE is reset if there is a blocked Modbus instance. This can, for example, occur if a client request is output and the execution of the client function aborts before the request was fully executed. The wait time must be between 0.5 s and 55 s.
MB_Transaction_ID	WORD	1	Transaction ID of the Modbus TCP protocol. The start value of "1" should only be changed if the Modbus TCP server requires a different value.
MB_Unit_ID	BYTE	255	Modbus device detection: A Modbus TCP server is addressed using its IP address. For this reason, the MB_UNIT_ID parameter is not used in the case of Modbus TCP addressing. The MB_UNIT_ID parameter corresponds to the field of the slave address in the Modbus RTU protocol. If a Modbus TCP server is used as a gateway for a Modbus RTU protocol, the slave device in the serial network can be identified using MB_UNIT_ID. The MB_UNIT_ID parameter would in this case forward the request to the correct Modbus RTU slave address. Please note that some Modbus TCP devices may require the MB_UNIT_ID parameter for the initialization within a limited value range. Use the following links to get more information on the MB_Unit_ID parameter: https://support.industry.siemens.com/cs/ww/en/view/102420337 (https://support.industry.siemens.com/cs/document/102420337/what-is-the-function-of-the-mb_unit_id-in-the-instance-data-block-of-the-modbus-block-mb_client?dti=0&dl=en&lc=de-WW) http://support.automation.siemens.com/WW/view/en/109736516 (https://support.industry.siemens.com/cs/document/109736516/how-do-you-clear-the-status-168382-in-the-case-of-a-modbus-tcp-connection-between-simatic-s7-1500-s7-1200-and-sentron-pac-devices?dti=0&dl=en&lc=de-WW)
RCV_TIMEOUT	REAL	2.0	Time interval in seconds in which the "MB_CLIENT" instruction waits for a response from the server. It must be between 0.5 s and 55 s.
Connected	BOOL	0	Indicates whether the connection to the assigned server has been established or not: 1 = connected, 0 = not connected.
RETRIES	WORD	0	Number of send attempts made by the "MB_CLIENT" instruction before it returns the error W#16#80C8.

Note

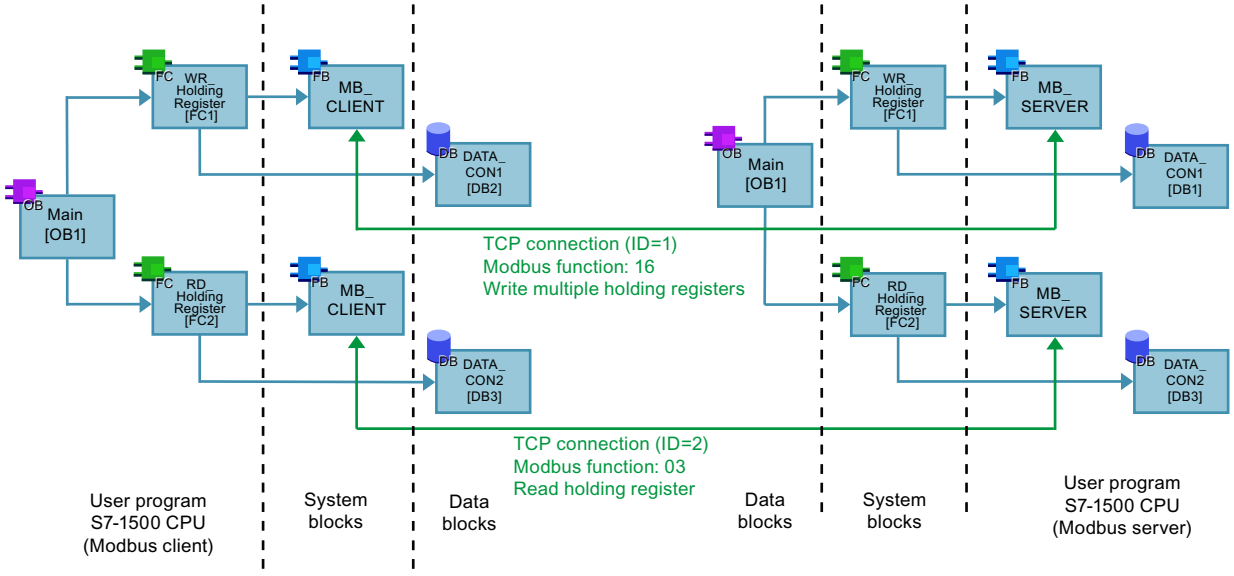
Tag MB_Transaction_ID

If the transaction ID in the answer of the Modbus TCP server does not match the transaction ID of the job from "MB_CLIENT", the "MB_CLIENT" instruction waits for the time period RCV_TIMEOUT * RETRIES for the answer of the Modbus TCP server with the correct transaction ID; once this time has expired, it returns the error W#16#80C8.

Example

An example project for the Modbus TCP communication between two S7-1500 CPUs can be found in the Service and Support Portal under Entry ID 94766380 (http://support.automation.siemens.com/WW/view/en/94766380).

Two Modbus functions are used in this example. For each Modbus function, a Modbus TCP connection is established using a Modbus block pair (MB_CLIENT and MB_SERVER).



See also

Difference between synchronous and asynchronous instructions (Page 2571)

REQ and DISCONNECT parameters (S7-1200, S7-1500)

Description

If no job is active and if the value of the DISCONNECT parameter is "0", a new job executes if REQ=1. If there is not yet a connection, this is established during execution.

If the same instance of the "MB_CLIENT" instruction executes again (DISCONNECT = 0 and REQ = 1), before the active job was executed, this is not executed on completion of the active job. A new job can only be started on completion of the active job (REQ = 1).

The status of the execution is output by the output parameters. You can use it to monitor the execution status when the "MB_CLIENT" instruction is executed sequentially.

MB_MODE, MB_DATA_ADDR and MB_DATA_LEN parameters (S7-1200, S7-1500)

Description

The combination of the MB_MODE, MB_DATA_ADDR and MB_DATA_LEN parameters defines the Modbus function code used in the current Modbus message for the values 0, 1 and 2 of MB_MODE:

- MB_MODE contains the information on whether to read or to write.
  MB_MODE=0: Read, MB_MODE=1 and 2: Write (Note: With MB_MODE=2, there is not distinction between Modbus functions 15 and 05 or between Modbus functions 16 and 06.)
- MB_DATA_ADDR contains the information on what is to be read or written, as well as address information from which the "MB_CLIENT" instruction calculates the remote address.
- MB_DATA_LEN contains the number of values to be read/written.

Examples:

- The combination MB_MODE=1, MB_DATA_ADDR=1, MB_DATA_LEN=1 specifies the function code 05. 1 output bit is written starting from the remote address 0.
- The combination MB_MODE=1, MB_DATA_ADDR=1, MB_DATA_LEN=2 specifies the function code 15. 2 output bits are written starting from the remote address 0.

The following applies to the values 101 to 106 and 115 to 116 from MB_MODE:

- MB_MODE defines the Modbus function code.
- MB_DATA_ADDR contains the remote address.
- MB_DATA_LEN contains the number of values to be read/written.

Example:

- MB_MODE=104, MB_DATA_ADDR=17,834, MB_DATA_LEN=125
  - MB_MODE=104 defines the function code 04 (read input words).
  - MB_DATA_ADDR=17,834 defines the remote address 17,834.
  - MB_DATA_LEN=125 defines that 125 values are read.

The following table shows the relationship between the input parameters MB_MODE, MB_DATA_ADDR, MB_DATA_LEN of the "MB_CLIENT" instruction and the associated Modbus function.

MB_MODE	MB_DATA_ADDR	MB_DATA_LEN	Modbus function	Function and data type
0	1 to 9,999	1 to 2,000	01	Read 1 to 2,000 output bits on the remote address 0 to 9,998
0	10,001 to 19,999	1 to 2,000	02	Read 1 to 2,000 input bits on the remote address 0 to 9,998

MB_MODE	MB_DATA_ADDR	MB_DATA_LEN	Modbus function	Function and data type
0	• 40,001 to 49,999 • 400,001 to 465,535	1 to 125	03	• Read 1 to 125 holding registers on the remote address 0 to 9,998 • Read 1 to 125 holding registers on the remote address 0 to 65,534
0	30,001 to 39,999	1 to 125	04	Read 1 to 125 input words on the remote address 0 to 9,998
1	1 to 9,999	1	05	Write 1 output bit on the remote address 0 to 9,998
1	• 40,001 to 49,999 • 400,001 to 465,535	1	06	• Write 1 holding register on the remote address 0 to 9,998 • Write 1 holding register on the remote address 0 to 65,534
1	1 to 9,999	2 to 1,968	15	Write 2 to 1,968 output bits on the remote address 0 to 9,998
1	• 40,001 to 49,999 • 400,001 to 465,535	2 to 123	16	• Write 2 to 123 holding registers on the remote address 0 to 9,998 • Write 2 to 123 holding registers on the remote address 0 to 65,534
2	1 to 9,999	1 to 1,968	15	Write 1 to 1,968 output bits on the remote address 0 to 9,998
2	• 40,001 to 49,999 • 400,001 to 465,535	1 to 123	16	• Write 1 to 123 holding registers on the remote address 0 to 9,998 • Write 1 to 123 holding registers on the remote address 0 to 65,534
11	The MB_DATA_ADDR and MB_DATA_LEN parameters are not evaluated when this function is executed.		11	Read status word and event counter of the server: • The status word reflects the the processing status (0 - not processing, 0xFFFF - processing). • The event counter is incremented when the Modbus request was executed successfully. If an error occurred during execution of a Modbus function, a message is sent by the server but the event counter is not incremented.
80	-	1	08	Check the server status with the diagnostic code 0x0000 (return loop test - the server sends the request back): • 1 WORD per call
81	-	1	08	Reset the event counter of the server with the diagnostic code 0x000A: • 1 WORD per call
101	0 to 65,535	1 to 2,000	01	Read 1 to 2,000 output bits on the remote address 0 to 65,535
102	0 to 65,535	1 to 2,000	02	Read 1 to 2,000 input bits on the remote address 0 to 65,535
103	0 to 65,535	1 to 125	03	Read 1 to 125 holding registers on the remote address 0 to 65,535
104	0 to 65,535	1 to 125	04	Read 1 to 125 input words on the remote address 0 to 65,535
105	0 to 65,535	1	05	Write 1 output bit on the remote address 0 to 65,535
106	0 to 65,535	1	06	Write 1 holding register on the remote address 0 to 65,535
115	0 to 65,535	1 to 1,968	15	Write 1 to 1,968 output bits on the remote address 0 to 65,535

MB_MODE	MB_DATA_ADDR	MB_DATA_LEN	Modbus function	Function and data type
116	0 to 65,535	1 to 123	16	Write 1 to 123 holding registers on the remote address 0 to 65,535
3 to 10, 12 to 79, 82 to 100, 107 to 114, 117 to 255				Reserved

MB_DATA_PTR parameter (S7-1200, S7-1500)

Description

The MB_DATA_PTR parameter is a pointer to a data buffer for the data to be received from the Modbus server or to be sent to the Modbus server. As the data buffer, you can use a global data block or a memory area (M).

For a buffer in the memory area (M), use a pointer in the ANY format as follows: "P#bit address" "data type" "length" (example: P#M1000.0 WORD 500).

Depending on the memory area in which the data buffer is located, MB_DATA_PTR can reference different data structures:

- When you use a global DB with optimized access, MB_DATA_PTR can reference a tag with elementary data type or an array of elementary data types. The following data types are supported:

Data type	Length in bits
Bool	1
Byte, SInt, USInt, Char	8
Word, Int, WChar, UInt	16
DWord, DInt, UDInt, Real	32

  In such cases, all supported data types can be used for all Modbus functions. For example, MB_CLIENT can also write a received bit in a tag of the byte type to a specified address without changing other bits in this byte. Therefore, it is not necessary to have an array of bits in order to execute bit-oriented functions.

- If you use a bit memory address area or a global DB with standard access as memory area, there is no longer any restriction to the elementary data types for MB_DATA_PTR; MB_DATA_PTR can then also reference complex data structures such as PLC data types (UDTs) and system data types (SDTs).

Note

Using a bit memory address area as data buffer

If you use a bit memory address area as data buffer for MB_DATA_PTR, you need to observe this variable. With S7-1500-CPUs it is 16 KB, with S7-1200-CPUs it is 8 KB.

CONNECT parameter (S7-1200, S7-1500)

Connection descriptions at CONNECT parameter

Two different connection descriptions can be used for the "MB_CLIENT" instruction:

- Programmed connections with the structure TCON_IP_v4
  The connection parameters are stored in the TCON_IP_v4 structure and the connection is set up with the call of the instruction "MB_CLIENT".
- Configured connections with the structure TCON_Configured
  The configured connection has already been established by the CPU. Use the structure TCON_Configured to specify which existing connection is to be used for the instruction.

Each instance of the instruction "MB_CLIENT" requires a unique connection. Create a separate structure TCON_IP_v4 or TCON_Configured for each instance of the instruction to describe the connection.

Connection description for programmed connections

Use the following structure for connection description to TCON_IP_v4 for programmed connections at the CONNECT parameter:

- Make sure that only connections of the type TCP are specified in the TCON_IP_v4 structure.
- The connection may not use the following TCP port numbers: 20, 21, 25, 80, 102, 123, 5001, 34962, 34963 and 34964.

Byte	Parameter	Data type	Start value	Description
0 … 1	InterfaceID	HW_ANY	-	Hardware identifier of the local interface (value range: 0 to 65535).
2 … 3	ID	CONN_OUC	-	Reference to this connection (value range: 1 to 4095). The parameter uniquely identifies a connection within the CPU. Each individual instance of the instruction "MB_CLIENT" must use a unique ID.
4	ConnectionType	BYTE	11	Connection type Select 11 (decimal) for TCP. Other connection types are not permitted. If another connection type (e.g. UDP) is used, a corresponding error message is output at the STATUS parameter of the instruction.
5	ActiveEstablished	BOOL	TRUE	ID for the manner in which the connection is established Select TRUE for active connection establishment.
6 … 9	RemoteAddress	ARRAY [1..4] of BYTE	-	IP address of the connection partner (Modbus server), for example, for 192.168.0.1: • addr[1] = 192 • addr[2] = 168 • addr[3] = 0 • addr[4] = 1

Byte	Parameter	Data type	Start value	Description
10 … 11	RemotePort	UINT	502	Port number of the remote connection partner (value range: 1 to 49151). Use the IP port number of the server to which the client establishes the connection and communicates using the TCP/IP protocol (default value: 502).
12 … 13	LocalPort	UINT	0	Port number of the local connection partner: • Port numbers: 1 to 49151 • Any port: "0"

Note

Migration of the instruction "MB_CLIENT" version 2.1

The parameters CONNECT_ID, IP_PORT and IP_OCTET_x are mapped in version 3.0 of the "MB_CLIENT" instruction in the TCON_IP_v4 structure:

- The parameter CONNECT_ID of the "MB_CLIENT" V2.1 instruction corresponds to the ID parameter of TCON_IP_v4.
- The parameter IP_PORT of the "MB_CLIENT" V2.1 instruction corresponds to the RemotePort parameter of TCON_IP_v4.
- The four parameters IP_OCTET_x of the "MB_CLIENT" V2.1 instruction correspond to the array of the RemoteAddress parameter of TCON_IP_v4.

Connection description for configured connections

For programmed connections at the CONNECT parameter, use the following structure for connection description to TCON_Configured.

- Make sure that only connections of the type TCP are specified in the TCON_Configured structure.
- The connection may not use the following TCP port numbers: 20, 21, 25, 80, 102, 123, 5001, 34962, 34963 and 34964.

Byte	Parameter	Data type	Start value	Description
0 … 1	InterfaceId	HW_ANY	-	Hardware identifier of the local interface (value range: 0 to 65535).
2 … 3	ID	CONN_OUC	-	Reference to this connection (value range: 1 to 4095). Enter the connection ID of the existing connection.
4	ConnectionType	BYTE	0	Connection type Select 254 (decimal) for a configured connection.

Parameter STATUS (S7-1200, S7-1500)

Parameter STATUS (general status information)

STATUS* (W#16#)	Description
0000	Instruction executed without errors.
0001	Connection established.
0003	Connection terminated.
7000	No job active and no connection established (REQ=0, DISCONNECT=1).
7001	Connection establishment triggered.
7002	Intermediate call. Connection is being established.
7003	Connection is being terminated.
7004	Connection established and monitored. No job processing active.
7005	Data is being sent.
7006	Data is being received.
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Parameter STATUS (protocol error)

STATUS* (W#16#)	Local and/ or remote errors	Error code in the answer from MB_SERVER (B#16#)	Description
80C8	Local	-	No response of the server in the defined period. Check the connection to the Modbus server. This error is only reported on completion of the configured repeated attempts. If the "MB_CLIENT" instruction does not receive an answer with the originally transferred transaction ID (see static tag MB_TRANSACTION_ID) within the defined period, this error code is output.
8380	Local	-	Received Modbus frame has incorrect format or too few bytes were received.
8381	Remote	01	Function code is not supported.
8382	Local	-	• The length of the Modbus frame in the frame header does not match the number of received bytes. • The number of bytes does not match the number of actually transmitted bytes (only functions 1-4). For example, this is the case when "MB_CLIENT" requests an odd number of words, but "MB_SERVER" always sends an even number of words. • The start address in the received frame does not match the saved start address (functions 5, 6, 15, 16). • The number of words does not match the number of actually transmitted words (functions 15 and 16).
	Remote	03	Invalid length specification in received Modbus frame. Check the server side.

STATUS* (W#16#)	Local and/or remote errors	Error code in the answer from MB_SERVER (B#16#)	Description
8383	Local	-	Error reading or writing data or access outside the address area of MB_DATA_PTR (Page 4660).
	Remote	02	Error reading or writing data or access outside the address area of the server
8384	Local	-	• Invalid exception code received. • A different data value was received than was originally sent by the client (functions 5, 6 and 8). • Invalid status value received (function 11)
	Remote	03	Error in data value for function 5
8385	Local	-	• Diagnostics code not supported. • A different subfunction code was received than was originally sent by the client (function 8).
	Remote	03	Diagnostics code not supported
8386	Local	-	Received function code does not match the one sent originally.
8387	Local	-	The protocol ID of the Modbus TCP frame received by the server is not "0".
8388	Local	-	The Modbus server sent a different data length than was requested. This error occurs only when using the Modbus functions 5, 6, 15 or 16.
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".			

Parameter STATUS (parameter error)

STATUS* (W#16#)	Description
80B6	Invalid connection type, only TCP connections are supported.
80BB	Invalid value at ActiveEstablished parameter (identifier for the type of connection establishment, see CONNECT parameter (Page 4661)): • Only passive connection establishment permitted for server (ActiveEstablished = FALSE). • Only active connection establishment permitted for client (ActiveEstablished = TRUE).
8188	The MB_MODE parameter has an invalid value.
8189	Invalid addressing of data at the MB_DATA_ADDR parameter.
818A	Invalid data length at the MB_DATA_LEN parameter.
818B	The MB_DATA_PTR parameter has an invalid pointer. You should also check the values of the MB_DATA_ADDR (Page 4660) and MB_DATA_LEN parameters.
818C	Timeout at parameter BLOCKED_PROC_TIMEOUT or RCV_TIMEOUT (see static tags of instruction). BLOCKED_PROC_TIMEOUT and RCV_TIMEOUT must be between 0.5 s and 55 s.
8200	• A different Modbus request is currently being processed via the port. • Another instance of MB_CLIENT with the same connection parameters is processing an existing Modbus request.
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Note

Error codes of internally used communications instructions

With the "MB_CLIENT" instruction, in addition to the errors listed in the tables, errors caused by the communication instructions ("TCON", "TDISCON", "TSEND", "TRCV", "T_DIAG" and "TRESET") used by the instruction can occur.

The error codes are assigned via the instance data block of the "MB_CLIENT" instruction. The error codes are displayed for the respective instruction under STATUS in the "Static" section.

The meaning of the error codes is available in the documentation of the corresponding communications instruction.

Note

Communication error when sending or receiving data

If a communication error occurs when sending or receiving data (80C4 (Temporary communications error. The specified connection is temporarily down.), 80C5 (Remote partner closed connection actively.), 80A1 (The specified connection is disconnected or is not yet established.)), the existing connection is terminated.

This also means that you can see all STATUS values that are returned when the connection is terminated and that the STATUS code that caused the connection to be terminated is only output when the connection is terminated.

Example: If a temporary communication error occurs when data is received, the STATUS 7003 (ERROR=false) is output initially and then 80C4 (ERROR=true).

See also

GET_ERR_ID: Get error ID locally (Page 2840)

MB_SERVER: Communicating via PROFINET as a Modbus TCP server (S7-1200, S7-1500)

Description of MB_SERVER (S7-1200, S7-1500)

Description

The "MB_SERVER" instruction communicates as Modbus TCP server via a PROFINET connection. The "MB_SERVER" instruction processes connection requests of a Modbus TCP client, receives and processes Modbus requests and sends responses.

For the S7-1200 with firmware version V4.0 you can use the "MB_SERVER" instruction up to and including library version V3.1. With the S7-1200 as of firmware version V4.1 and the S7-1500, you can use the "MB_SERVER" instruction of all library versions.

The connection can take place via the local interface of the CPU or CM/CP.

To use the instruction, you do not require an additional hardware module.

NOTICE
Security information Note that each client of the network is given read and write access to the process image inputs and outputs and to the data block or bit memory area defined by the Modbus holding register. The option is available to restrict access to an IP address to prevent unauthorized read and write operations. Note, however, that the shared address can also be used for unauthorized access.

Multiple server connections

You can create multiple Server connections. This allows a single CPU to accept connections from multiple Modbus TCP clients at the same time.

A Modbus TCP server can support several TCP connections and the maximum number of connections depends on the CPU being used.

The total number of connections of one CPU, including those of the Modbus TCP clients and server must not exceed the maximum number of supported connections.

Modbus TCP connections can also be shared by "MB_CLIENT" and/or "MB_SERVER" instances.

In the case of Server connections, remember the following rules:

- Each "MB_SERVER" connection must use a unique instance DB.
- Each "MB_SERVER" connection must use a unique connection ID.
  The relevant individual connection ID must be used for each individual instance DB of the instruction. The connection ID and instance DB belong together in pairs and must be unique for each connection.
- For each connection, the "MB_SERVER" instruction must be called individually.

Parameters

The following table shows the parameters of the "MB_SERVER" instruction:

Parameter	Declaration	Data type	Description
DISCONNECT	Input	BOOL	The "MB_SERVER" instruction is used to enter into a passive connection with a partner module. The server responds to a connection request from the IP address which is entered in the SDT "TCON_IP_v4" in the CONNECT parameter. You can use this parameter to control when a connection request is accepted: • 0: A passive connection is established when there is no communications connection. • 1: Initialization of the connection termination. If the input is set, no other operations are executed. The value 0003 is output at the STATUS parameter after successful connection termination.
MB_HOLD_REG (Page 4670)	InOut	VARIANT	Pointer to the Modbus holding register of the "MB_SERVER" instruction MB_HOLD_REG must always reference a memory area that is larger than two bytes. The holding register contains the values that may be accessed by a Modbus client using the Modbus functions 3 (read), 6 (write), 16 (multiple write) and 23 (reading and writing in one job). As the holding register, use either a global data block with optimized access or the memory area of the bit memories.
CONNECT (Page 4671)	InOut	VARIANT	Pointer to the structure of the connection description The following structures (SDTs) can be used: • TCON_IP_v4: Includes all address parameters that are required for establishing a programmed connection. The default address is 0.0.0.0 (any IP address), but you can enter a specific IP address so that the server only responds to requests from this address. When using TCON_IP_v4, the connection is established when calling the "MB_SERVER" instruction. • TCON_Configured: Includes the address parameters of a configured connection. When using TCON_Configured, the connection is established by the CPU after download of the hardware configuration.
NDR	Output	BOOL	"New Data Ready": • 0: No new data • 1: New data written by the Modbus client written
DR	Output	BOOL	"Data Read": • 0: No data read • 1: Data read by the Modbus client
ERROR	Output	BOOL	If an error occurs during the call of the "MB_SERVER" instruction, the output of the ERROR parameter is set to "1". Detailed information about the cause of the problem is indicated by the STATUS parameter.
STATUS (Page 4673)	Output	WORD	Detailed status information of the instruction.

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Static tags of the instruction

The following table describes the static tags of the instance data block of the "MB_SERVER" instruction used in the program. You can write the HR_Start_Offset tag. You can read the other tags to monitor the Modbus status.

Tag	Data type	Start value	Description
HR_Start_Offset	WORD	0	Assign the start address of the Modbus holding register.
QB_Start	WORD	0	Start address of the permitted addressing range of the outputs that can be written (bytes 0 to 65535)
QB_Count	WORD	0xFFFF	Number of output bytes that can be written by the Modbus master. Example: QB_Start=0 and QB_Count=10: Output bytes 0 to 9 can be written. QB_Count=0: No output byte can be written.
Request_Count	WORD	0	Total number of requests received by the server.
Server_Message_Count	WORD	0	Total number of received alarms for the relevant server.
Xmt_Rcv_Count	WORD	0	Counter for detecting the number of transfers during which an error occurred. The counter is only incremented when an invalid Modbus request is received.
Exception_Count	WORD	0	Counters for detecting the number of errors specifically for Modbus which cause an error message to "MB_CLIENT".
Success_Count	WORD	0	Event counter for detecting the number of requests that were successfully executed by the server.
Connected	BOOL	0	Indicates whether the connection to the assigned client has been established or not: 1 = connected, 0 = not connected.

Mapping of Modbus addresses to the process image

The "MB_SERVER" instruction allows incoming Modbus functions (1, 2, 4, 5 and 15) direct read and write access to the process image inputs and outputs of the CPU (use of the data types BOOL and WORD).

For S7-1200-CPUs, the address space for the process image of the inputs and the process image of the outputs is 1 KB; it is 32 KB each for S7-1500-CPUs.

The following table shows the address space of the Modbus functions listed above.

Modbus function					
Function code	Function	Data area	Address space		
01	Read: Bits	Output	0	to	65.535
02	Read: Bits	Input	0	to	65.535
04	Read: WORD	Input	0	to	65.535
05	Write: Bit	Output	0	to	65.535
15	Write: Bits	Output	0	to	65.535

Incoming Modbus requests with the function codes 3, 6, 16 and 23 write to or read from the Modbus holding register (you specify the holding register with the MB_HOLD_REG parameter).

Example: Addressing via static tag HR_Start_Offset

The addresses of the Modbus holding register start at 0 (from the perspective of the MB_CLIENT at 40.001). These addresses correspond to the address area of the CPU memory area for the holding register. You can also define the HR_Start_Offset tag so that the Modbus holding register has a start address other than 0.

Example: The holding register begins at MW100, and has a length of 100 WORD. An offset value in the HR_Start_Offset parameter means that the start address of the holding register is moved from 0 to 20. This causes an error whenever the holding register is addressed below the address 20 and above the address 119.

HR_Start_Offset	Address	Minimum	Maximum
0	Modbus address (WORD)	0	99
	CPU address	MW100	MW298
20	Modbus address (WORD)	20	119
	CPU address	MW100	MW298

Modbus functions

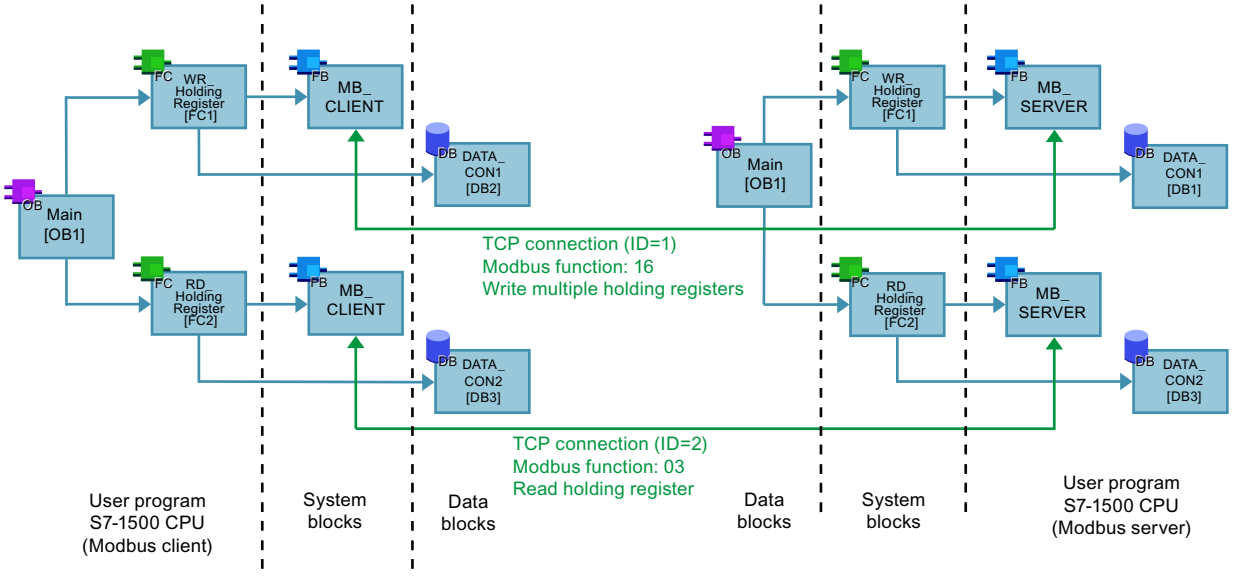
The following table lists all the Modbus functions that are supported by the "MB_SERVER" instruction.

Function code	Description
01	Read output bits
02	Read input bits
03	Read a holding register
04	Read input words
05	Write an output bit
06	Write a holding register
08	Diagnostics function: • Echo test (subfunction 0x0000): The "MB_SERVER" instruction receives a data word and returns this unchanged to the Modbus client. • Reset event counter (subfunction 0x000A): The "MB_SERVER" instruction resets the following event counters: "Success_Count", "Xmt_Rcv_Count", "Exception_Count", "Server_Message_Count" and "Request_Count".
11	Diagnostics function: Fetch event counter of the communication The "MB_SERVER" instruction uses an internal event counter for communication to record the number of successfully executed read and write requests sent to the Modbus server. The event counter is not incremented with the functions 8 or 11. The same holds true for requests that cause a communications error, for example, if a protocol error has occurred (e.g., the function code in the received Modbus request is not supported).
15	Write output bits
16	Write a holding register
23	Write a holding register and read holding register with a request

Example

An example project for the Modbus TCP communication between two S7-1500 CPUs can be found in the Service and Support Portal under Entry ID 94766380 (http://support.automation.siemens.com/WW/view/en/94766380).

Two Modbus functions are used in this example. For each Modbus function, a Modbus TCP connection is established using a Modbus block pair (MB_CLIENT and MB_SERVER).



See also

Difference between synchronous and asynchronous instructions (Page 2571)

MB_HOLD_REG parameter (S7-1200, S7-1500)

Description

The MB_HOLD_REG parameter is a pointer to a data buffer for storing the data read from or written to the Modbus server. You can use a global data block or a bit memory (M) as memory area.

- The high limit for the number of addresses in the data block (D) is determined by the maximum work memory of the CPU.
- The high limit for the number of bit memories (M) is determined by the size of the CPU memory area.

The following table shows examples of mapping Modbus addresses to the holding register for the Modbus functions 3 (read WORD), 6 (write WORD), 16 (write several WORD) and 23 (write and read several words).

Modbus addresses	MB_HOLD_REG parameter - examples		
0	MW100	DB10.DBW0	"Recipe".ingredient[1]
1	MW102	DB10.DBW2	"Recipe".ingredient[2]
2	MW104	DB10.DBW4	"Recipe".ingredient[3]
3	MW106	DB10.DBW6	"Recipe".ingredient[4]
4	MW108	DB10.DBW8	"Recipe".ingredient[5]

CONNECT parameter (S7-1200, S7-1500)

Connection descriptions at CONNECT parameter

Two different connection descriptions can be used for the "MB_SERVER" instruction:

- Programmed connections with the structure TCON_IP_v4
  The connection parameters are stored in the TCON_IP_v4 structure and the connection is set up with the call of the instruction "MB_SERVER".
- Configured connections with the structure TCON_Configured
  The configured connection has already been established by the CPU. Use the structure TCON_Configured to specify which existing connection is to be used for the instruction.

Each instance of the instruction "MB_SERVER" requires a unique connection. Create a separate structure TCON_IP_v4 or TCON_Configured for each instance of the instruction to describe the connection.

Connection description for programmed connections

For programmed connections at the CONNECT parameter, use the following structure for connection description to TCON_IP_v4.

- Make sure that only connections of the type TCP are specified in the TCON_IP_v4 structure.
- The connection may not use the following TCP port numbers: 20, 21, 25, 80, 102, 123, 5001, 34962, 34963 and 34964.

Byte	Parameter	Data type	Start value	Description
0 … 1	InterfaceID	HW_ANY	-	Hardware identifier of the local interface (value range: 0 to 65535).
2 … 3	ID	CONN_OUC	-	Reference to this connection (value range: 1 to 4095). The parameter uniquely identifies a connection within the CPU. Each individual instance of the instruction "MB_SERVER" must use a unique ID.

Byte	Parameter	Data type	Start value	Description
4	ConnectionType	BYTE	11	Connection type Select 11 (decimal) for TCP. Other connection types are not permitted. If another connection type (e.g. UDP) is used, a corresponding error message is output at the STATUS parameter of the instruction.
5	ActiveEstablished	BOOL	FALSE	ID for the manner in which the connection is established Select FALSE for passive connection establishment.
6 … 9	RemoteAddress	ARRAY [1..4] of BYTE	0.0.0.0	IP address of the connection partner, for example, for 192.168.0.1: • addr[1] = 192 • addr[2] = 168 • addr[3] = 0 • addr[4] = 1 If the instruction "MB_SERVER" is to accept connection requests from any connection partner, use "0.0.0.0" as IP address.
10 … 11	RemotePort	UINT	0	Port number of the remote connection partner (value range: 1 to 49151). If the instruction "MB_SERVER" is to accept connection requests from any remote partner, use "0" as port number.
12 … 13	LocalPort	UINT	502	Port number of the local connection partner (value range: 1 to 49151). The number of the IP port defines which IP port is monitored for connection requests of the Modbus client. The following TCP port numbers must not be used for the passive connection of the "MB_SERVER" instruction: 20, 21, 25, 80, 102, 123, 5001, 34962, 34963 and 34964.

Note

Migration of the instruction "MB_SERVER" version 2.1

The parameters CONNECT_ID and IP_PORT are mapped in version 3.0 of the "MB_SERVER" instruction in the TCON_IP_v4 structure:

- The parameter CONNECT_ID of the "MB_SERVER" V2.1 instruction corresponds to the ID parameter of TCON_IP_v4.
- The parameter IP_PORT of the "MB_SERVER" V2.1 instruction corresponds to the LocalPort parameter of TCON_IP_v4.

Connection description for configured connections

For configured connections at the CONNECT parameter, use the following structure for connection description to TCON_Configured.

- Make sure that only connections of the type TCP are specified in the TCON_Configured structure.
- The connection may not use the following TCP port numbers: 20, 21, 25, 80, 102, 123, 5001, 34962, 34963 and 34964.

Byte	Parameter	Data type	Start value	Description
0 … 1	InterfaceID	HW_ANY	-	Hardware identifier of the local interface (value range: 0 to 65535).
2 … 3	ID	CONN_OUC	-	Reference to this connection (value range: 1 to 4095). Enter the connection ID of the existing connection.
4	ConnectionType	BYTE	-	Connection type Select 254 (decimal) for a configured connection.

Parameter STATUS (S7-1200, S7-1500)

Parameter STATUS (general status information)

STATUS* (W#16#)	Description
0000	Instruction executed without errors.
0001	Connection established.
0003	Connection terminated.
7000	No call active and no connection established (REQ=0, DISCONNECT=1).
7001	First call. Connection establishment triggered.
7002	Intermediate call. Connection is being established.
7003	Connection is being terminated.
7005	Data is being sent.
7006	Data is being received.
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Parameter STATUS (protocol error)

STATUS* (W#16#)	Local and/or remote errors	Error code in the answer from MB_SERVER (B#16#)	Description
8380	Local	-	Received Modbus frame has incorrect format or too few bytes were received.
8381	Remote	01	Function code is not supported.
8382	Remote	03	Error in data length • Invalid length specification in received Modbus frame • The frame length entered in the header of the Modbus frame does not match the number of actually received bytes. • The number of bytes entered in the header of the Modbus frame does not match the number of actually received bytes (functions 15 and 16).

STATUS* (W#16#)	Local and/or remote errors	Error code in the answer from MB_SERVER (B#16#)	Description
8383	Local	-	Error reading or writing data or access outside the address area of MB_HOLD_REG.
	Remote	02	Error reading or writing data or access outside the address area of the server
8384	Local	-	• Invalid exception code received. • A different data value was received than was originally sent by the server (functions 5, 6 and 8). • Invalid status value received (function 11)
	Remote	03	Error in data value for function 5
8385	Local	-	• Diagnostics code not supported. • A different subfunction code was received than was originally sent by the server (function 8).
	Remote	03	Diagnostics code not supported
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".			

Parameter STATUS (parameter error)

STATUS* (W#16#)	Description
80BB	Invalid value at ActiveEstablished parameter (identifier for the type of connection establishment, see CONNECT parameter (Page 4671)): • Only passive connection establishment permitted for server (active_established = FALSE). • Only active connection establishment permitted for client (active_established = TRUE).
8187	The MB_HOLD_REG parameter has an invalid pointer. Data area is too small.
* The status codes can be displayed as integer or hexadecimal values in the program editor. For information on switching the display formats, refer to "See also".	

Note

Error codes of internally used communications instructions

With the "MB_SERVER" instruction, in addition to the errors listed in the tables, errors caused by the communication instructions ("TCON", "TDISCON", "TSEND", "TRCV", "T_DIAG" and "T_RESET") used by the instruction can occur.

The error codes are assigned via the instance data block of the "MB_SERVER" instruction. The error codes are displayed for the respective instruction under STATUS in the "Static" section.

The meaning of the error codes is available in the documentation of the corresponding communications instruction.

Note

Communication error when sending or receiving data

If a communication error occurs when sending or receiving data (80C4 (Temporary communications error. The specified connection is temporarily down.), 80C5 (Remote partner closed connection actively.), 80A1 (The specified connection is disconnected or is not yet established.)), the existing connection is terminated.

This also means that you can see all STATUS values that are returned when the connection is terminated and that the STATUS code that caused the connection to be terminated is only output when the connection is terminated.

Example: If a temporary communication error occurs when data is received, the STATUS 7003 (ERROR=false) is output initially and then 80C4 (ERROR=true).

See also

GET_ERR_ID: Get error ID locally (Page 2840)

MB_HOLD_REG parameter (Page 4670)

Differences between the versions <= V3.x and >= V4.0 of the Modbus TCP library (S7-1200, S7-1500)

Differences between the versions <= V3.x and >= V4.0 of the Modbus instructions

The following differences between versions exist with the MB_SERVER/MB_CLIENT MODBUS instructions:

- Address parameter
  - In versions <= 3.x, the address data for the Modbus TCP server is specified via the input parameters "IP_x".
  - Versions >= 4.0 use TCON_IP_V4 and TCON_Configured system data types at the CONNECT input parameter for this purpose.
- If an error occurs during execution, as of version 4.0 of the Modbus instructions, there are more STATUS alarms available which can be evaluated accordingly.

You can find detailed information in the instruction descriptions for MB_CLIENT (Page 4653) and MB_SERVER (Page 4665).

Note

Changing the Modbus TCP library from a version <= V3.x to a version >= V4.0

When you replace the Modbus TCP library from a version <= V3.x to a version >= V4.0 you also need to replace the Open User Communication library. Afterwards check all the instructions relevant for your program.

11.4.6.5 TeleService (S7-1200)

TM_MAIL: Transfer email (S7-1200)

Description of TM_MAIL (S7-1200)

Description

The "TM_MAIL" instruction works asynchronously, in other words, its execution extends over multiple calls. You must specify an instance when you call the instruction "TM_MAIL". The attribute "retentive" may not be set in the instance. This attribute ensures that the instance is initialized when the CPU switches from STOP to RUN and that a new e-mail send job can be triggered afterwards.

You start the sending of an e-mail with an edge change from "0" to "1" for the REQ parameter. The job status is indicated by the output parameters BUSY, "DONE", "ERROR", "STATUS" and "SFC_STATUS". "SFC_STATUS" corresponds in this case to the "STATUS" output parameter of the called communication blocks.

The output parameters DONE, ERROR, STATUS, and SFC_STATUS are each displayed for only one cycle if the status of the BUSY output parameter changes from "1" to "0". The following table shows the relationship between BUSY, DONE, and ERROR. Using this table, you can determine the current status of the instruction "TM_MAILand when the sending of the e-mail is complete.

DONE	BUSY	ERROR	Description
0	1	0	The job is being processed.
1	0	0	Job successfully completed.
0	0	1	The job ended with an error. The cause of the error can be found in the STATUS and SFC_STATUS parameters.
0	0	0	The "TM_MAIL" instruction was not assigned a (new) job.

If the CPU changes to STOP mode while "TM_MAIL" is active, the communication connection to the mail server aborts. The communication connection to the mail server will also be lost if

communication problems occur on the Industrial Ethernet bus. In this case, the transfer of the e-mail will be interrupted and it will not reach its recipient.

NOTICE
Changing user programs You can change the parts of your user program that directly affect calls of "TM_MAIL" only: • when the CPU is in "STOP" mode or • when no mail is being sent (REQ = 0 and BUSY = 0). This relates, in particular, to deleting and replacing program blocks that contain "TM_MAIL" calls or calls for the instance of "TM_MAIL". Ignoring this restriction can tie up connection resources. The automation system can change to an undefined status for the TPC/IP communication functions via Industrial Ethernet. A warm or cold restart of the CPU is required after the changes are transferred.

Data consistency

The ADDR_MAIL_SERVER input parameter of the instruction is taken from the "TM_MAIL" instruction again each time the sending of an e-mail is triggered. If a change is made during operation, the "new" value only becomes effective once an e-mail is triggered again.

In contrast, the WATCH_DOG_TIME, TO_S, CC, FROM, SUBJECT, TEXT, ATTACHMENT, and, if applicable, USERNAME and PASSWORD parameters are taken from the "TM_MAIL" instruction while it is running, which means that they may only be changed after the job is complete (BUSY = 0)

Setting the parameters of the TS Adapter IE

On the TS Adapter IE, you need to specify parameters for outgoing calls in such a way as to enable the TS Adapter IE to establish a connection to the dial-up server of your service provider.

If you set "When required" for connection establishment, the connection will only be established when an e-mail needs to be sent.

Connection establishment can take longer (approx. one minute) for an analog modem connection. When you specify the WATCH_DOG_TIME parameter, remember to allow for the time required to establish the connection.

Parameter

The following table shows the parameters of the "TM_MAIL" instruction:

Parameter	Declaration	Data type	Memory area	Description
REQ	Input	BOOL	I, Q, M, D, L or constant	Control parameter REQUEST: Activates the sending of an e-mail on a rising edge.
ID	Input	CONN_OUC (Word)	D, L or constant	Reference to the connection to be established. See ID parameter of the TCON (Page 4474), TDISCON (Page 4484), TSEND (Page 4490), and TRCV (Page 4493) instructions. A number that is not used for any additional instances of this instruction in the user program must be entered here.
TO_S (Page 4680)	Input	STRING	D	Input parameter for receiver addresses: STRING with a maximum length of 240 characters (see example call).
CC (Page 4680)	Input	STRING	D	Input parameter for CC recipient addresses (optional): STRING with a maximum length of 240 characters (see example call). If an empty string is assigned here, the e-mail is not sent to a CC recipient.
SUBJECT	Input	STRING	D	Input parameter for subject of the e-mail: STRING with a maximum length of 240 characters.
TEXT	Input	STRING	D	Input parameter for text of the e-mail (optional): Reference to a data string with a maximum length of 240 characters. If an empty string is assigned at this parameter, the e-mail is sent without text.
ATTACHMENT	Input	VARIANT	I, Q, M, D, L	Input parameter for e-mail attachment (optional): Reference to a byte/word/double word field with a maximum length of 65534 bytes. The specified attachment must be in the file format BIN. If no value is assigned, the e-mail is sent without an attachment.
DONE	Output	BOOL	I, Q, M, D, L	• DONE = 0: Job not yet started or still executing. • DONE = 1: Job executed without errors.
BUSY	Output	BOOL	I, Q, M, D, L	• BUSY = 1: Sending of an e-mail is not yet complete. • BUSY = 0: Processing of "TM_MAIL" was terminated
ERROR	Output	BOOL	I, Q, M, D, L	ERROR = 1: An error occurred during processing. STATUS and SFC_STATUS supply detailed information about the type of error.
STATUS (Page 4681)	Output	WORD	I, Q, M, D, L	Output/status parameter STATUS: Return value or error information of the "TM_MAIL" instruction.

Parameter	Declaration	Data type	Memory area	Description
ADDR_MAIL_SERVER	Static*	DWORD	I, Q, M, D, L	IP address input parameter of the mail server: Specify as a data word in HEX format, for example: IP address = 192.168.0.200. ADDR_MAIL_SERVER = DW#16#C0A800C8, where: • 192 = 16#C0, • 168 =16#A8 • 0 = 16#00 and • 200 = 16#C8.
WATCH_DOG_TIME	Static*	TIME	I, Q, M, D, L	Input parameter for max. period of time: The "TM_MAIL" instruction should establish a connection within the period specified by WATCH_DOG_TIME. The block is exited with an error if this period is exceeded. The time before the block is exited and the error is output can exceed the WATCH_DOG_TIME because connection termination also takes a certain amount of time. To begin with, you should set a period of 2 minutes. If you have an ISDN telephone connection, a significantly shorter period can be selected.
USERNAME	Static*	STRING	D	Input parameter for user name: STRING with a maximum length of 180 characters. A user name is an absolute requirement for authentication.
PASSWORD	Static*	STRING	D	Input parameter for password: STRING with a maximum length of 180 characters. A password is an absolute requirement for authentication.
FROM (Page 4680)	Static*	STRING	D	Input parameter for sender address: STRING with a maximum length of 240 characters (see example call).
SFC_STATUS (Page 4681)	Static*	WORD	I, Q, M, D, L	Output/status parameter "SFC_STATUS": Error information of the called communication blocks.
*The values of the parameter are not modified at every call of the instruction "TM_MAIL". The values are in the static parameters of the instance and are only written once at the first call of the instruction.				

You can find additional information on valid data types under "Overview of the valid data types (Page 2239)".

Note

Optional parameters

The optional parameters CC, TEXT, and ATTACHMENT are only sent with the e-mail if the corresponding parameters contain a string of length > 0.

Note

Parameter ATTACHMENT

The specified attachment must be in the file format BIN. If text is to be transferred, you have the following options:

- In Windows, you link the file format BIN with a text editor.
- You use the Windows functions "Send to" or "Open with".

SMTP authentication

Authentication refers here to a procedure for ensuring an identity, for example, by a password query.

The "TM_MAIL" instruction supports the SMTP authentication procedure AUTH-LOGIN that is requested by most mail servers. For information about the authentication procedure of your mail server, please refer to your mail server manual or the website of your Internet service provider.

To use the AUTH-LOGIN authentication procedure, the "TM_MAIL" instruction requires the user name with which it can log onto the mail server. This user name corresponds to the user name with which you set up a mail account on your mail server. It is made available to the "TM_MAIL" instruction in the USERNAME parameter.

To log on, the "TM_MAIL" instruction also requires the associated password. This password corresponds to the password you specified when you set up your mail account. It is made available to the "TM_MAIL" instruction in the PASSWORD parameter.

The user name and the password are each transferred to the mail server unencrypted (BASE64-coded).

If no user name is specified in the DB, the AUTH-LOGIN authentication procedure is not used. The e-mail is then sent without authentication.

Example

You can find a detailed application example in the Siemens Industry Online Support under the following FAQ ID: 67262019 (https://support.industry.siemens.com/cs/ww/en/view/67262019).

Parameters TO_S, CC, and FROM (S7-1200)

Description

The TO_S, CC, and FROM parameters are strings with, for example, the following content:

- TO: <wenna@mydomain.com>, <ruby@mydomain.com>,
- CC: <admin@mydomain.com>, <judy@mydomain.com>,
- FROM: <admin@mydomain.com>

Note the following rules when entering the parameters:

- The "TO:", "CC:", and "FROM:" characters must be entered.
- A space and an opening pointed bracket "<" must be entered before each address.
- A closing pointed bracket ">" must be entered after each address.
- A comma must be entered after each address in TO and CC.
- Only one e-mail may be entered in FROM, and there must not be a comma at the end of this address

Because of runtime and memory space, the "TM_MAIL" instruction does not perform any syntax check of the parameters TO_S, CC and FROM.

STATUS and SFB_STATUS parameters (S7-1200)

Description

The return values of the "TM_MAIL" instruction can be classified as follows:

- W#16#0000: "TM_MAIL" was completed successfully
- W#16#7xxx: Status of "TM_MAIL"
- W#16#8xxx: An error was reported during the internal call of a communication block or from the mail server.

The following table shows the return values of "TM_MAIL" except for error codes of the internally called communication blocks.

Return value STATUS* (W#16#...):	Return value SFB_STATUS (W#16#...):	Explanation	Notes
0000	-	The processing of "TM_MAIL" was completed without error.	A without error completion of "TM_MAIL" does not mean that the sent e-mail will necessarily arrive (see below - note on point 1)
7001		"TM_MAIL" is active (BUSY = 1).	Initial call; job has started
7002	7002	"TM_MAIL" is active (BUSY = 1).	Intermediate call; job already active
8xxx	xxxx	The processing of "TM_MAIL" was completed with an error code of the internally called communication instructions.	For detailed information on the evaluation of the SFB_STATUS parameter, refer to the descriptions of the STATUS parameter of the communication instructions.
8010	xxxx	Error during connection establishment.	For further information on the evaluation of the SFB_STATUS parameter, refer to the descriptions of the STATUS parameter of the "TCON (Page 4474)" instruction.
8011	xxxx	Error sending the data.	For further information on the evaluation of SFB_STATUS, refer to the descriptions of the STATUS parameter of the "TSEND (Page 4490)" instruction.

Return value STATUS* (W#16#...):	Return value SFB_STATUS (W#16#...):	Explanation	Notes
8012	xxxx	Error receiving the data.	For further information on the evaluation of SFB_STATUS, refer to the descriptions of the STATUS parameter of the "TRCV (Page 4493)" instruction.
8013	xxxx	Error during connection establishment.	For further information on the evaluation of SFB_STATUS, refer to the descriptions of the STATUS parameter of the "TCON (Page 4474)" and "TDISCON (Page 4484)" instructions.
8014	-	Establishment of a connection is not possible.	You have possibly entered an incorrect mail server IP address (ADDR_MAIL_SERVER) or a time span that is too short (WATCH_DOG_TIME) to establish the connection. It is also possible that the CPU has no connection to the network or that the CPU configuration is incorrect.
8016	-	Error copying the attachment	-
82xx, 84xx, or 85xx	-	The error message originates from the mail server and the last three digits correspond to the error number of the SMTP protocol. The following columns list several error codes that can occur:	See point 2 in the note.
8450	-	Action not executed: Mailbox not available/ not reachable.	Try again later.
8451	-	Action aborted: Local processing error	Try again later.
8500	-	Syntax error: Error not recognized. This also includes the error when a command string is too long. This can also be occur when the e-mail server does not support the LOGIN authentication procedure.	Check the parameters of "TM_MAIL". Try to send an e-mail without authentication. To do this, replace the USERNAME parameter with an empty string.
8501	-	Syntax error: Parameter or argument incorrect	You have possibly entered an incorrect address in TO_S or CC.
8502	-	Command unknown or not implemented.	Check your entries, in particular the FROM parameter. This is possibly incomplete and you have forgotten "@" or ".".
8535	-	SMTP authentication incomplete.	You have possibly entered an incorrect user name or incorrect password.
8550	-	Mail server cannot be reached, you have no access rights.	You have possibly entered an incorrect user name or password, or the mail server does not support your LOGIN. Another cause of error could be an incorrect entry of the domain name after the "@" in TO_S or CC.
8552	-	Action aborted: Assigned memory size has been exceeded	Try again later.

Return value STATUS* (W#16#...):	Return value SFB_STATUS (W#16#...):	Explanation	Notes
8554	-	Transmission failed.	Try again later.
* The error codes can be displayed as integer or hexadecimal values in the program editor. For additional information on toggling display formats, refer to "See also".			

Note

Status error

1. An incorrect entry of the recipient addresses does not generate a status error of the "TM_MAIL" instruction. In this case, there is no guarantee that the e-mail will be sent to other recipients, even if these were entered correctly.
2. You can find more detailed information on the SMTP error code and other error codes in SMTP protocol on the Internet or in the error documentation of the mail server. You can also view the most recent message from the mail server in your instance DB in the BUFFER1 parameter. If you look under "Data", you will find the data most recently sent by the "TM_MAIL" instruction.

11.5 Program editor

11.5.1 Overview of the program editor

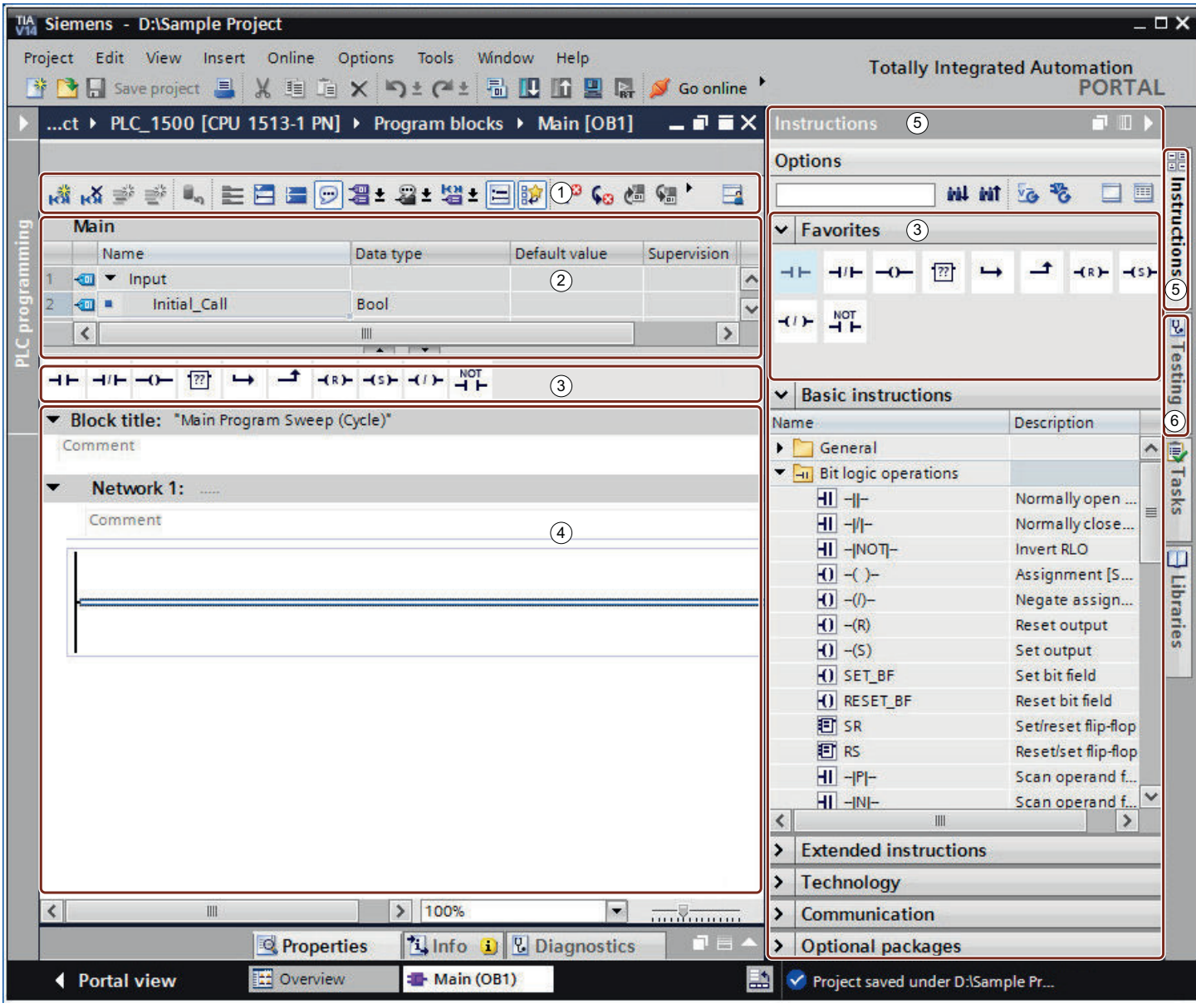
Function of the program editor

The program editor is the integrated development environment for programming functions, function blocks, and organization blocks. If offers comprehensive support for programming and troubleshooting.

The appearance and functionality of the program editor can vary depending on the CPU, programming language and block type used.

Structure of the program editor

Using LAD as an example, the following figure shows the components of the program editor:



① Toolbar
② Block interface
③ "Favorites" pane in the "Instructions" task card and favorites in the programming window
④ Programming window
⑤ "Instructions" task card
⑥ "Testing" task card

Toolbar

The toolbar allows you to access the principal functions of the program editor, such as:

- Show and hide absolute operands
- Showing and hiding favorites
- Skip to syntax errors
- Update block calls
- Show and hide program status

The functions available in the toolbar can vary depending on the programming language used.

Block interface

The block interface contains the declarations for local tags used solely within the block. The sections available depend on the block type.

Favorites

You can save frequently used instructions as favorites. These favorites are then displayed in the "Instructions" task card and the "Favorites" pane. You can also display favorites in the programming window using the program editor toolbar. This allows you to access your favorites even when the "Instructions" task card is not visible.

Programming window

The programming window is the work area of the program editor. You can enter the program code in this window. The appearance and functionality of the program window can vary depending on the programming language used.

"Instructions" task card

The "Instructions" task card offers you easy access to all instructions available for creating your program. The instructions are broken down by area into a number of different palettes. If an instruction profile is active, the offered instructions may vary. You receive a brief description for each instruction. If several versions of an instruction are available, the versions are listed in a drop-down list and the currently selected version is used in the program. If required, you can select a different instruction version. Read the information in the section "Basic information on instruction versions (Page 4697)".

With the task card toolbar, you can access the following functions:

- Search up or down for instructions
  You have the option of searching the panes for specific instructions.
- Upgrade project and instructions
  If you are still working with a project which was created with TIA Portal V14, only the instruction versions from V14 will be available. Using the "Upgrade project and instructions" button you can easily upgrade your project to TIA Portal V14 SP1. The "Instructions" task card then receives the new instruction versions of TIA Portal V14 SP1.

  Note

  Please note the following:
  - Your program is not changed by the upgrade, i.e. it still contains the old instruction versions from V14. To use the new instruction versions you must update your program.
  - The "project und operation upgrade" button is only active if you are still working with a TIA Portal V14 project. The button becomes inactive as soon as you upgrade your project to TIA Portal V14 SP1.
- Update program in the current PLC
  Updates the CPU program to the new instruction versions.
- Changing the pane mode
  You can choose between single and multi-pane mode.
  See also: Change pane mode (Page 259)
- Showing/hiding column headers
  You can show or hide column headers in the panes. You can modify the arrangement of columns by clicking a column header and moving the column to the new position by means of drag-and-drop.

"Testing" task card

You can set settings in the "Testing" task card for troubleshooting using the program status. The functions of the "Testing" task card are only available in online mode. It contains the following panes, which are displayed depending on the selected CPU and the configured programming language of the block:

- CPU operator panel
  The CPU operator panel allows you to switch the CPU operating mode.
- Breakpoints
  You can test blocks you created in the textual programming languages STL and SCL in single step mode. To do this, set breakpoints in the program code.
  In the "Breakpoints" pane, you can find all breakpoints that you have set and you can activate, delete, navigate to, or set the call environment for the individual breakpoints.
- PLC register
  This pane allows you to read off the values for the PLC registers and the accumulators.
- Sequence control
  In this pane you can set the operating mode for testing sequencers for GRAPH blocks.
- Test settings
  This pane allows you to specify the test settings for the GRAPH block.

- Call environment
  This pane allows you to specify the call environment for the block.
- Call hierarchy
  In this pane, you can trace the call hierarchy of the blocks. You only see the call hierarchy during block monitoring.

See also

Overview of the block interface (Page 4764)

Enlarging the programming window (Page 4693)

Find instructions (Page 4697)

Using instruction profiles (Page 4701)

11.5.2 Using the keyboard in the program editor

Navigating in the editor

Function	Shortcut key
Open "Instructions" task card	<Ctrl+Shift+C>
Open "Testing" task card	<Ctrl+Shift+O>
Add new block	<Ctrl+N>
Open block/PLC data type	<F7>
Expand all networks	<Alt+F11>
Collapse all networks	<Alt+F12>
Navigate to the next point of use of the selected block or operand	<Ctrl+Shift+G>
Navigate to the previous point of use of the selected block or operand	<Ctrl+Shift+F>
Navigate to the next read/write access	<Alt+F8>
Navigate to the previous read/write access	<Alt+F9>
Navigate to the definition of the selected block or operand	<Ctrl+Shift+D>

Navigating in the program code (LAD/FBD)

Function	Selected object	Shortcut key
Navigating between objects in the network	Object in the network	Arrow keys
Navigate to the first element of the network	Object in the network	<Home>
Navigate to the last element of the network	Object in the network	<End>
Navigate to the next element of the network	Object in the network	<Tab>
Navigate to the previous element of the network	Object in the network	<Shift+Tab>
Insert network	Any	<Ctrl+R>

Navigating in the program code (STL/SCL)

Function	Position of the cursor	Shortcut key
Navigating in the program code	Line	Arrow keys
One word to the right/left	Line	<Ctrl+arrow keys>
Cursor to start of line	Line	<Home>
Cursor to end of line	Line	<End>
Cursor to start of code section	Line	<Ctrl+Home>
Cursor to end of code section	Line	<Ctrl+End>
To the next network (STL only)	Network title	<Down arrow>
To the next network (STL only)	Line	<Tab> Repeat the shortcut keys until the insertion point is in the next network.
To the previous network (STL only)	Network title	<Up arrow>
To the previous network (STL only)	Line	<Shift+Tab> Repeat the shortcut keys until the insertion point is in the previous network.
Insert network	Any	<Ctrl+R>

Inserting instructions (LAD)

Function	Selected object	Shortcut key
Insert normally open contact	Rung	<Shift+F2>
Insert normally closed contact	Rung	<Shift+F3>
Insert empty box	Rung	<Shift+F5>
Insert assignment	Rung	<Shift+F7>
Insert "Open branch"	Rung	<Shift+F8>
Insert "Close branch"	Rung	<Shift+F9>

Inserting instructions (FBD)

Function	Selected object	Shortcut key
Insert AND logic operation	Network, input or output	<Shift+F2>
Insert OR logic operation	Network, input or output	<Shift+F3>
Insert empty box	Network	<Shift+F5>
Insert assignment	Network, input or output	<Shift+F7>
Insert "Open branch"	Connection line between two boxes	<Shift+F8>

Function	Selected object	Shortcut key
Invert RLO	Network, input or output	<Ctrl+Shift+4>
Insert input	Network, input or output	<Ctrl+Shift+3>

Entering operands (LAD/FBD/GRAPH)

Function	Selected object	Shortcut key
Activate the input field for the first operand of the instruction	Instruction	<Return> Or <Any letters/numbers> An empty input field opens on <Return>, any letters or numbers will be entered in the input field.
Activate input field for the operand	Operand	<F2>
Delete operand	Operand	<Del>
Define tag	Operand	<Ctrl+Shift+I>
Rewire tag	Operand	<Ctrl+Shift+P>
Rename tag	Operand	<Ctrl+Shift+T>
Entering operands	Input field for operands	<Any letters/numbers>
Confirm entry of the operand	Input field for operands	<Return>
Open autocompletion	Input field for operands	<Ctrl+I>
Discard current change	Input field for operands	<Esc> The input field is deactivated and the previous contents restored.

Process instructions (STL/SCL)

Function	Selected object	Shortcut key
Indent line (SCL only)	Line	<Tab> or <Ctrl+R>
Outdent line (SCL only)	Line	<Shift+Tab> or <Ctrl+Shift+R>
Automatically formatting selected text (SCL only)		<Ctrl+Shift+W>
Open "Call options" dialog	Cursor after block call	<Return>
Define tag	Operand	<Ctrl+Shift+I>
Rewire tag	Operand	<CtrlShift+P>
Rename tag	Operand	<Ctrl+Shift+T>
Expand/collapse parameter list (SCL only)	Operand	<Ctrl+Shift+Space>

Function	Selected object	Shortcut key
Expand/collapse code section	Insertion mark within the code section	<Ctrl+Shift+Num+> <Ctrl+Shift+Num->
Expand/collapse all code sections	Any	<Ctrl+Shift+Num*> <Ctrl+Shift+Num/>
Open autocompletion	Any	<Ctrl+I> or <Ctrl+Spacebar>
Set/delete bookmarks		<Ctrl+Shift+M>
To the next bookmark		<Ctrl+Shift+6>
To the previous bookmark		<Ctrl+Shift+5>
Comment out selection	Line	<Ctrl+Shift+Y>
Remove comment	Line	<Ctrl+Shift+U>

Programming window of GRAPH

Function	Block	Shortcut key
Page Up/Down	Navigation view, single step view, sequence view, permanent instructions	<Page Up>/ <Page Down>
Navigate in the navigation view	Navigation view	<Up arrow> <Down arrow>
Expand/collapse object	Navigation view	<+> or <Right arrow> <-> or <Left arrow>
Switch between single step view and sequence view when a step or a transition is selected	Navigation view	<Return>
Switch between navigation view and work area	Navigation view, single step view, sequence view, permanent instructions	<ALT+F6>
Go to first element in a network	Single step view	<Home>
Go to last element in a network	Single step view	<End>
Switch to interlock	Single step view	<Ctrl+Home>
Switch to transition	Single step view	<Ctrl+End>
Navigate in the structure	Sequence view	Arrow keys
Go to first step	Sequence view	<Home> or <Ctrl+Home>
Go to last step	Sequence view	<End> or <Ctrl+End>
Open branch	Sequence view	<Shift+F8>
Close branch	Sequence view	<Shift+F9>
Insert sequence end	Sequence view	<Shift+F7>
Insert jump	Sequence view	<Shift+F12>
Insert step and transition	Sequence view	<Shift+F5>
Delete element	Sequence view	<Del>

Function	Block	Shortcut key
Go to first editable element	Permanent instructions	<Home>
Go to next editable element	Permanent instructions	<Tab>
Go to last editable element	Permanent instructions	<End>
Go to previous editable element	Permanent instructions	<Shift+Tab>
Jump to the start of the "Action" cell	Actions	<Home>
Jump to the end of the "Action" cell	Actions	<End>
Insert new action	Actions	<Return>

Programming window of SCL

Function	Block	Shortcut key
Expand block	Block overview, block in the programming window	<Ctrl+Shift+Num+>
Collapse block	Block overview, block in the programming window	<Ctrl+Shift+Num->
Expand all blocks	Block overview, programming window	<Ctrl+Shift+Num*>
Collapse all blocks	Block overview, programming window	<Ctrl+Shift+Num/>

Monitor program

Function	Shortcut key
Set/remove breakpoint (STL, SCL)	<Ctrl+Shift+F9>
Step over breakpoint (STL, SCL)	<Ctrl+Shift+F10>
Jump to lower-level block (STL, SCL)	<Ctrl+Shift+F11>
Return to calling block (STL, SCL)	<Ctrl+Shift+F12>
Run program up to marker (cursor position) (STL, SCL)	<Ctrl+F3>
Display program status (SCL, STL)	<Ctrl+T>
Enable all breakpoints (STL, SCL)	<Ctrl+Shift+F2>
Disable all breakpoints (STL, SCL)	<Ctrl+Shift+F3>
Modify to 0 (LAD, FBD)	<Ctrl+Shift+9>
Modify to 1 (LAD, FBD)	<Ctrl+Shift+1>
Modify operand (LAD, FBD)	<Ctrl+Shift+2>

See also

Keyboard operation in the TIA Portal (Page 270)

Using project-related functions (Page 272)

Arranging windows (Page 272)

Editing tables (Page 278)

Text editing (Page 277)

11.5.3 Enlarging the programming window

Introduction

The programming window is relatively small when all components of the application are shown. If the program code is large, you may find you have to rearrange the work area constantly. To avoid this problem, you can hide or minimize the display of the following components of the application and of the program editor:

- Project tree
- Task cards
- Block interface
- Favorites
- Comments
- Networks

Note

You can also use the "Reduce automatically" option for the task cards, project tree, and Inspector window. These windows will then be minimized automatically when you do not need them.

See also: Maximizing and minimizing the work area

Hiding and showing the project tree

The project tree allows you to access all areas of the project. You can hide the project tree while you are creating a program so you have more space for the programming window.

To show and hide the project tree, follow these steps:

1. To hide the project tree, deselect the "Project tree" check box in the "View" menu, or click on "Collapse" on the project tree title bar.
2. To show the project tree, select the "Project tree" check box in the "View" menu or click on "Extend" on the project tree title bar.

Opening and closing task cards

The task cards are located at the right-hand edge of the programming window.

To open or close the task cards, follow these steps:

1. To close the task cards, deselect the "Task card" check box in the "View" menu or click "Collapse" on the task cards title bar.
2. To open the task cards, select the "Task card" check box in the "View" menu or click "Expand" on the task cards title bar.

Hiding and showing the block interface

The block interface is shown in the upper section of the program editor. During programming you can show and hide it as required.

To show and hide the block interface, follow these steps:

1. In the lower part of the interface within the window splitter, click on the Up arrow or Down arrow.

Showing and hiding favorites

To hide or show the favorites in the program editor, follow these steps:

1. Click the "Display favorites in the editor" button in the program editor toolbar.

Showing and hiding comments

Within a block you can enter a comment for the block or for each network. These two types of comments are shown and hidden differently.

To show or hide a block comment, follow these steps:

1. Click the the triangle at the start of the line with the block title.

To show or hide network comments, follow these steps:

1. Click "Network comments on/off" on the program editor toolbar.

Note

The comments available can vary depending on the programming language used.

Opening and closing networks

Some programming languages use networks. You can open or close these networks as required.

To open or close networks, follow these steps:

1. If you want to open a network, click the right arrow in front of the network title. If you want to close a network, click the down arrow in front of the network title.

To open and close all networks, follow these steps:

1. Click "Open all networks" or "Close all networks" in the program editor toolbar.

Note

Networks are not used in every programming language.

See also

Overview of the program editor (Page 4684)

Maximizing and minimizing the work area (Page 249)

11.5.4 Setting the mnemonics

You can program blocks using German or international mnemonics. When you open the TIA Portal for the first time, international mnemonics is set as default. You can change the mnemonics at any time.

Procedure

To set the mnemonics, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General" group in the area navigation.
3. In the "General settings" group, select the mnemonics that you want to use.
   The mnemonics is changed in all blocks.

11.5.5 Displaying symbolic and absolute addresses

You have the following options for displaying operands in the program editor:

- Symbolic representation
  The symbolic operands are displayed in the program. The corresponding absolute addresses are shown in tooltips if you hold the mouse pointer over the operand.
- Absolute representation
  The absolute addresses are displayed in the program. The corresponding symbolic operands are displayed in tooltips.
- Symbolic and absolute representation
  Symbolic operands and absolute addresses are displayed in program. This setting only has an effect on blocks that were programmed with LAD, FBD, STL, and GRAPH.

Note

Note that absolute operands are not displayed for GRAPH blocks if you have disabled the option "Display absolute operands" in the "Action" column.

You can set the display of operands centrally for all new blocks of the project or for individual blocks.

Setting operand representation for new blocks

To set the operand representation for all new blocks in the project, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "PLC programming > General" group in the area navigation.
3. Select the desired representation in the "Operand representation" drop-down list.
   The selected representation is enabled for all new blocks in the program.

Setting operand representation for an individual block

To change the representation of the operands, follow these steps:

1. Open the block in the program editor.
2. Click the "Absolute/symbolic operands" button in the program editor toolbar.
   Each time you click the button, the representation and the symbol on the button change.

Or:

1. Open the block in the program editor.
2. Click the small arrow next to the "Absolute/symbolic operands" button in the program editor toolbar.
   A drop-down list is displayed.
3. Select the required representation from the drop-down list.
   The symbol on the button changes.

11.5.6 Find instructions

You have the option of finding specific instructions in the "Instructions" task card and then inserting them in your program. Note the following rules when entering search terms:

- No distinction is made between upper and lower case text.
- The search function considers parts of a search term.
- You cannot use placeholders, such as "*" and "?".
- If an instruction name includes underscores, the instruction is found even if you do not enter the underscore.
- The text in the columns "Name" and "Description" are taken into consideration in the search.

Requirement

- A block is open.
- The "Instructions" task card is open.

Procedure

To search for a specific instruction in the "Instructions" task card, follow these steps:

1. Select a starting point for the search if you want to start the search at a specific point. If you do not select a starting point, the search starts either at the top or bottom in the task card, depending on the type of search you have selected.
2. Enter a search term in the text box of the task card toolbar.
3. Click "Search down" if you want to search the task card from top to bottom.
4. Click "Search up" if you want to search the task card from bottom to top.
   The first match with the search term found is displayed as the result. If you want to continue the search, click the "Search down" or "Search up" button again. If no matches are found, you will receive a corresponding message.

See also

Overview of the program editor (Page 4684)

11.5.7 Use instruction versions

11.5.7.1 Basic information on instruction versions

Introduction

The instructions available to you for programming the user program are managed in system libraries. If a new version of a system library is installed by an update, the newer versions of the instructions of this system library may also be installed.

If there are several versions for an instruction, these are listed in a drop down list in the "Instructions" task card after the respective instruction. If the instruction versions are not shown, you can show them again using the shortcut menu of the column header. In the drop-down list, select the version of an instruction which is to be used in the program. If you do not select any versions, the most recent versions are used.

Note

Please note the following:

- You can only ever use the same version of an instruction within a device.
- If you use an instruction in a know-how-protected block with a version other than the version set in the project, you can compile the know-how-protected block without a password if the interfaces of the instruction versions are identical. The next time you open the know-how-protected block, you will need to update the call for the instruction. The corresponding call is marked in red. The call is updated automatically by the system by compiling the entire program.
- If you change the version of an instruction that other instructions depend on, the versions of the dependent instructions are also changed.
- If you select a version for an instruction that can not be run on the CPU used, the instruction is shaded out. This means that you cannot use this instruction in this version with your CPU.
- The system block in the project tree displays the block version number in its properties; it does not have to match the instruction version number of the associated instruction in the task card.
- If you change the version of an instruction in the task card, you must compile the associated system block in the project tree before you update its block version number.

Changes in the versions

New versions can be main versions or secondary versions. New versions, such as 2.0 or 3.0, have more substantial changes to them. New main versions may therefore result in changes to the block interface. New secondary versions, such as 1.3 or 1.4, contain lesser changes or remedies to errors.

Using instruction versions

You can decide within a device which version of an instruction you want to use. If you select another version for an instruction, the new version is specified for all locations of use of this instruction within your program. These instructions are identified in the program by a red frame. You must then download your program to the device to use the new instruction version.

If you open your project with a newer TIA Portal version, you can upgrade your project and the instructions to the new TIA Portal version within the "Instructions" task card. Newer versions for some instructions are then available. These do not, however, automatically replace the old versions in your program. To use the new instruction versions in your program, you must explicitly update your program.

See also

Upgrade project and instructions (Page 4699)

Update program in the current PLC (Page 4699)

11.5.7.2 Upgrade project and instructions

You can upgrade a project and the instruction versions to a newer TIA Portal version in one step. This will not change your program.

Note

No backward compatibility

Projects that are saved with the current version of the TIA Portal are not backward compatible with older versions due to their enhanced functionality. This means that after upgrade you will no longer be able to open your project with an older version of the TIA Portal.

Procedure

To upgrade a project and the instructions, follow these steps:

1. Open a code block.
2. Open the "Instructions" task card.
3. Click "Upgrade project and instructions" in the toolbar.
   The "Project compatibility" dialog opens.
4. Click "OK" to continue with the upgrade.

Result

The project and instructions are upgraded. The "Upgrade project and instructions" button becomes inactive and the function can no longer be executed.

As the next step, you can update your program so that the new instruction versions can be used.

See also

Basic information on instruction versions (Page 4697)

Update program in the current PLC (Page 4699)

11.5.7.3 Update program in the current PLC

If there are new instruction versions in your project, you can update the program in the PLC so that the new instruction versions are used. This may be the case, for example after upgrading your project to a new TIA Portal version. Please note that you have to compile your program after update.

Procedure

Proceed as follows to update the program in the current PLC:

1. Right-click on the PLC for which you would like to update the program in the project tree.
2. Select the "Update program" command in the shortcut menu.
   The "Update program in the current CPU" dialog opens.
3. Click "OK" to perform the update.
4. Compile your program to complete the upgrade.

Or:

1. Open a code block.
2. Open the "Instructions" task card.
3. Click on "Update program in the current PLC" in the toolbar.
   The "Update program in the current CPU" dialog opens.
4. Click "OK" to perform the update.
5. Compile your program to complete the upgrade.

See also

Basic information on instruction versions (Page 4697)

Upgrade project and instructions (Page 4699)

11.5.8 Using instruction profiles

11.5.8.1 Basics of instruction profiles

Introduction

The TIA Portal provides you with numerous instructions that you can use to program the user program. However, you may filter out specific instructions that you do not want to use. To this end you can create instruction profiles in which you can explicitly specify the instructions to be listed in the "Instructions" task card. However, although you may create several instruction profiles in a project, only one of these profiles may be active at any given time. You can exchange instruction profiles with other users by means of shared libraries.

Note

Please note the following:

- The use of instructions that are not allowed in the active profile in a block will trigger the output of a block compilation error. Such a situation may be triggered if you drag-and-drop a block from the library to your program.
- Instructions of a profile that are not supported by the currently installed products are deleted from the profile the next time it is edited. If you transfer this profile to an engineering system in which these instructions are supported by the installed products, the instructions are again present in the profile but they are disabled. You can enable these instructions as required at any time.
- If you want to make changes to the active profile, you must recompile the blocks in the project. This is also necessary when you disable or delete the active profile or when you enable a profile.

See also

Creating new instruction profiles (Page 4701)

Opening and editing instruction profiles (Page 4703)

Activating and deactivating instruction profiles (Page 4704)

Deleting instruction profiles (Page 4705)

11.5.8.2 Creating new instruction profiles

Requirement

The "Common data > Instruction profiles" folder is open in the project navigation.

Procedure

Proceed as follows to create a new instruction profile:

1. Double-click the "Add new profile" command.
   The Instruction Profile Editor opens and displays the new instruction profile. All instructions are activated for the new instruction profile.
2. Edit the new instruction profile to suit your requirements.

If necessary, you can rename the new instruction profile. To do this, follow these steps:

1. Right-click on the new instruction profile.
2. Select the "Rename" command in the shortcut menu.
3. Enter a name for the new instruction profile.

Note

The first instruction profile that you create will be used as active profile. In this case, compile all blocks in the project. If other instruction profiles are already available you must explicitly activate the new one in order to use it as active profile. You can identify the active profile by its icon in the project navigation.

See also

Basics of instruction profiles (Page 4701)

Opening and editing instruction profiles (Page 4703)

Activating and deactivating instruction profiles (Page 4704)

Deleting instruction profiles (Page 4705)

11.5.8.3 Opening and editing instruction profiles

Once you have opened an instruction profile, you can edit it as follows:

- Activating and deactivating instructions
  You can explicitly specify the instructions to be allowed in the instruction profile.

  Note

  Note that dependencies exist between some instructions. As a result, it is possible that several instructions may be activated or deactivated by an action. The check box icon indicates the folders in which instructions are deactivated.

- Activating and deactivating instruction versions
  Certain instructions are available with different versions. You can explicitly specify the instruction versions to be allowed in the instruction profile.
- Renumbering blocks
  An instruction representing an internal function block (FB) or function (FC) in the system is assigned a specific block number by the system. You can replace this block number with your own block number. Within a version, there are several implementations for certain instructions. The block numbers in such instructions can only be changed for the specific implementation.

  Note

  If an instruction from the instruction profile is used in the program and the specified block number is already in use by a different block, the specified block number of the instruction will be replaced by a free block number.

Requirement

The "Common data > Instruction profiles" folder is open in the project navigation.

Opening instruction profiles

Proceed as follows to open an instruction profile:

1. Double-click the instruction profile that you want to edit.
   The instruction profile opens in the Instruction Profile Editor.

Editing instruction profiles

Proceed as follows to edit an instruction profile in the Instruction Profile Editor:

1. Select the device that you want to edit from the "Device family" drop-down list box.
2. Select the programming language for which you want to edit the instruction profile from the "Language" drop-down list box.
3. Deactivate the instructions or instruction versions that you want to exclude from the instruction profile. You can deactivate a folder to deactivate all subordinate instructions.

4. Activate the instructions or instruction versions that you want to allow in the instruction profile.
5. You may assign your own block numbers.

Note

You can assign the number up to 65535 for CPUs of the S7-1200/1500 series. For CPUs of the S7300/400 series you find the restrictions of the number ranges in the respective CPU manual.

Note

A new compilation process is required for all blocks in the project when you change the active profile.

See also

Basics of instruction profiles (Page 4701)

Creating new instruction profiles (Page 4701)

Activating and deactivating instruction profiles (Page 4704)

Deleting instruction profiles (Page 4705)

Basic information on instruction versions (Page 4697)

11.5.8.4 Activating and deactivating instruction profiles

You first need to activate an instruction profile in order to include filtering of its instructions. You can always deactivate the instruction profile to reset the instructions task card to the default scope of instructions.

Note

A new compilation process is required for all blocks in the project.

Requirement

The "Common data > Instruction profiles" folder is open in the project navigation.

Activating instruction profiles

Proceed as follows to activate an instruction profile:

1. Right-click on the instruction profile that you want to activate.
2. Select the "Activate profile" command from the shortcut menu.
   The selected instruction profile is now active. Instructions can now only be used in accordance with the settings of this profile.

Deactivating instruction profiles

Proceed as follows to deactivate the instruction profile:

1. Right-click on the instruction profile that you want to deactivate.
2. Select the "Deactivate profile" command from the shortcut menu.
   No instruction profile is active and the "Instructions" task card once again shows all instructions that are available for use.

See also

Basics of instruction profiles (Page 4701)

Creating new instruction profiles (Page 4701)

Opening and editing instruction profiles (Page 4703)

Deleting instruction profiles (Page 4705)

11.5.8.5 Deleting instruction profiles

Requirement

The "Common data > Instruction profiles" folder is open in the project tree.

Procedure

Proceed as follows to delete an instruction profile:

1. Right-click on the instruction profile that you want to delete.
2. Select the "Delete" command in the shortcut menu.

Note

A new compilation process is required for all blocks in the project when you delete the active profile.

Result

The selected instruction profile is deleted. If you deleted the active instruction profile, no more active profiles are available and the "Instructions" task card once again shows all instructions that are available for use.

See also

Basics of instruction profiles (Page 4701)

Creating new instruction profiles (Page 4701)

Opening and editing instruction profiles (Page 4703)

Activating and deactivating instruction profiles (Page 4704)

11.5.9 Using autocompletion

11.5.9.1 Basics of autocompletion

Function

You can use autocompletion in the program window of the program editor as an easy way to access available tags or instructions during programming. Autocompletion means a context-specific list appears in a dialog from which you can select the tags or instructions you need.

See also

Using autocompletion in graphic programming languages (Page 4706)

Using autocompletion in textual programming languages (Page 4707)

11.5.9.2 Using autocompletion in graphic programming languages

Inserting tags using autocompletion

To insert tags in graphic programming languages using autocompletion, follow these steps:

1. Select an operand of the instruction to which you wish to assign a tag.
   The input field for the operand opens. The autocompletion button will appear beside the input field.
2. Either click the autocompletion button or use the shortcut <Ctrl+l>.
   Autocompletion opens. It contains only the local and global tags, data blocks and multiple instances which are admissible for the operand in the given context. You can exit autocompletion at any time by pressing <Esc>.

3. Select the required tag from the list. If necessary, you can also filter the list:
   - For example, enter the first few letters of the name of the tag or instruction you wish to insert. Autocompletion will be filtered further with each letter entered. If there is no tag or instruction starting with the letters entered, autocompletion will remain at the last match.
   - Enter # to access the local tags from the block interface.
   - Enter " to access the global tags.
   - Enter % to access absolute addresses.

   If the tag is a structured tag, a data block or a multiple instance, then an arrow is displayed at the end of the row. Click on the arrow to display the lower-level elements. You can navigate to the very last level in this way. If a structure is allowed as a data type for the operand, you can choose "None" from the list. This assigns the entire structure to the operand as a tag. Use the <Backspace> key to return to the previous level.
4. Press the <Return> key to apply the tag.

See also

Basics of autocompletion (Page 4706)

Using autocompletion in textual programming languages (Page 4707)

11.5.9.3 Using autocompletion in textual programming languages

Inserting tags and instructions using autocompletion

To insert tags and instructions in textual programming languages using autocompletion, follow these steps:

1. Enter the first few letters of the name of the tag or instruction you wish to insert. If necessary, you can directly filter the kind of tags:
   - Enter # to access the local tags from the block interface.
   - Enter " to access the global tags.
   - Enter % to access absolute addresses.

   Autocompletion opens. It contains only the local and global tags, data blocks, multiple instances and instructions which are admissible at the current position. You can exit autocompletion at any time by pressing <Esc>.
2. Enter more letters of the name of the tag or instruction you wish to insert. You can use <Enter> or <Tab> to apply the tag or instruction and close autocompletion. Autocompletion will be filtered further with each letter entered. If there is no tag or instruction starting with the letters entered, autocompletion only contains the previous matches.

3. Select the tag or instruction required from the list.
   If a tag is a structured tag, a data block or a multiple instance, first select the tag, the data block or multiple instance from the autocompletion and apply the selection with <Enter>. To select the additional components of the structure, data block, or multiple instance, enter a period. Autocompletion then reopens and you can select the next component.
4. Press the <Return> key to apply the tag.

See also

Basics of autocompletion (Page 4706)

Using autocompletion in graphic programming languages (Page 4706)

11.5.10 General settings for the PLC programming

11.5.10.1 Overview of the general settings

Overview

The following table shows the general settings that you can make:

Group	Setting	Description
View	Operand representation	Representation of the operand in the program editor. You can select between the following options: • Symbolic and absolute • Symbolic • Absolute (only for LAD, FBD, STL and GRAPH)
	Tag information	Additional information for the tags used is displayed in the program editor. When you select the option "Tag information with hierarchy", the comments of the higher structure levels are also displayed for structured tags.
	Location of tag information (LAD/FBD)	Specifies the position at which the tag information is to be displayed. You can display the tag information either collectively below a LAD/FBD network or directly at the respective operand.
	with network comments	Network comments are shown.
Compilation	Delete actual parameters on interface update	Actual parameters are deleted if the associated formal parameters were deleted in the called block, and you run the "Update block call" function or compile the block.

Group	Setting	Description
Default settings for new blocks	IEC check	The compatibility of operands in comparison operations and arithmetic operations are tested according to IEC rules. You have to explicitly convert non-compatible operands.
Additional settings	Set network title automatically	Sets the title of a network based on the comment of the output parameter of the first writing instruction in the network. See also: Inserting network title (Page 4887)
	Show autocompletion list	The autocompletion list is displayed.
	Mnemonics	German or international designation of operations and operands
Download without reinitialization	Memory reserve	Defines the size of reserve in volatile memory that can be used for interface extensions.

Group	Setting	Description
Block interface / data blocks	Set "Accessible from HMI/OPC UA" for new elements and ARRAY data blocks.	Enables the option "Accessible from HMI" for new tags in the block interface and in data blocks. The use of this option makes sense especially when working with large amounts of data in ARRAY data blocks. See also: Basic principles for programming of data blocks (Page 4795)
	Set data block elements with access type "Standard" to retentive.	When this check box is selected, the "Retain" attribute is set for newly created tags. The setting influences the following types of data blocks: • S7-1200/S7-1500: Instance data blocks without optimized access – the setting influences all data block tags. • S7-1200/S7-1500: Instance data blocks with optimized access – the setting influences all tags with the attribute "Retain: set in IDB". **Note** With other types of data blocks you cannot specify any default for the retentivity. • S7-300/S7-400: Tags in instance data blocks and global data blocks are retentive by default. If you change the retentivity setting, the change affects all tags in the block. • S7-1200/S7-1500: Tags in instance data blocks with optimized access are non-retentive by default. You can set the retentivity for each tab individually.

See also

Overview of the block interface (Page 4764)

Changing the settings (Page 4711)

Permissible addresses and data types of PLC tags (Page 4840)

Overview of the print settings (Page 233)

Setting and canceling the IEC check (Page 2467)

Using autocompletion (Page 4706)

11.5.10.2 Changing the settings

Procedure

To change the settings, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.
3. Change the settings.

Result

The change will be loaded directly, there is no need to save it explicitly.

See also

Overview of the general settings (Page 4708)

11.5.11 Eliminating syntax errors in the program

11.5.11.1 Basic information on syntax errors

Syntax errors

Below are some examples of syntax errors:

- Missing separators or the use of too many separators
- Incorrect keyword spelling
- Incorrect jump label spelling/notation
- Notation which does not match the set mnemonics (for example, "I2.3" instead of "E2.3")
- The use of key words as operands

Identification of syntax errors

Syntax errors are underlined in red or appear in red type.

This identification allows you to recognise incorrect inputs at a glance and jump from error to error to eliminate them. Syntax errors are also listed in the "Info" tab of the inspector window with an error message.

See also

Finding syntax errors in the program (Page 4712)

11.5.11.2 Finding syntax errors in the program

Procedure

To find syntax errors in the program, follow these steps:

1. Select the position in the program in which you wish to look for errors.
2. Click "Go to next error" in the toolbar.
   The first error after the position you have selected will be marked.

You can use "Go to next error" and "Go to previous error" in the toolbar to find and correct all errors in the block.

Or:

1. Open the error list in the inspector window with "Info > Syntax".
   All syntax errors are listed in the table with a short description of the error.
2. If there are any errors, click on the blue question mark next to the error text to obtain information on eliminating the problem.
3. Double-click the error you want to correct.
   The corresponding error is highlighted.

See also

Basic information on syntax errors (Page 4711)

11.5.12 Changing the programming language

11.5.12.1 Rules for changing the programming language

Rules

Observe the following rules if you want to change the programming language for a block:

- All CPU series:
  - You can only change the programming language of entire blocks. The programming language cannot be changed for individual networks.
  - You cannot switch blocks programmed in the programming languages SCL or GRAPH. In GRAPH blocks, however, you can change between LAD and FBD as network languages.
- S7-300/400:
  - You can only change between the programming languages LAD, FBD and STL.
  - You can create networks within a block using another programming language and then copy them into the desired block.
  - If the language of individual networks of the block cannot be changed, these networks is displayed in their original language.
- S7-1200/1500:
  - You can change between the programming languages LAD and FBD.
- S7-1500:
  - You can create STL networks within the LAD and FBD blocks. However, you cannot copy between STL and LAD/FBD.

11.5.12.2 Change the programming language

Procedure

To change the programming language, follow these steps:

1. Right-click the block in the project tree.
2. Select the "Properties" command in the shortcut menu.
   The dialog with the properties of the block opens.
3. Select the "General" entry in the area navigation.
4. Select the new programming language in the "Language" drop-down list.
5. Confirm your selection with "OK".

See also

Rules for changing the programming language (Page 4713)

11.6 Creating and managing blocks

11.6.1 Creating blocks

11.6.1.1 Block folder

Function

You can find a "Program blocks" folder in the project tree, in which you can create and manage the following blocks:

- Organization blocks (OB)
- Function blocks (FB)
- Functions (FCs)
- Data blocks (DB)

A "System blocks" subfolder containing another subfolder, "Program resources", is also created in the "Program blocks" folder the first time you drag an instruction to your program which is an internal system function block. The instance data block of the internal system function block is also pasted to the "Program resources" folder. You can copy, rename or delete such instance data blocks in the "Program resources" folder. The "Program resources" folder is deleted together with the "System blocks" folder if contains no more blocks.

A program cycle OB is automatically generated for each device and inserted in the "Program blocks" folder.

Note

Moving blocks from the "Program resources" folder or to the "Program resources" folder

You can move blocks between the "Program resources" folder and other folders. When moving blocks from the "Program resources" folder or to the "Program resources" folder, note the following:

- Moving blocks from the "Program resources" folder into another folder: When the next time the user program is compiled, the shifted instance data blocks are moved back into the "Program resources" folder.
- Moving by any blocks to the "Program blocks" folder: Blocks in the "Program resources" folder that are not required to run the user program are removed during the next compilation. This may cause your moved block to be deleted.

See also

Creating functions and function blocks (Page 4716)

Creating data blocks (Page 4716)

Creating organization blocks (Page 4715)

Using blocks from libraries (Page 4718)

11.6.1.2 Creating organization blocks

Requirement

The "Program blocks" folder in the project tree is open.

Procedure

To create an organization block, follow these steps:

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Organization block (OB)" button.
3. Select the type of new organization block.
4. Enter a name for the new organization block.
5. Enter the properties of the new organization block.
6. To enter additional properties for the new organization block, click "Additional information".
   An area with further input fields is displayed.
7. Enter all the properties you require.
8. Activate the "Add new and open" check box if the organization block does not open as soon as it is created.
9. Confirm your entries with "OK".

Result

The new organization block is created. You can find the organization block in the project tree in the "Program blocks" folder. You can assign additional parameters to some organization blocks in the inspector window or device view after they have been created. The organization block description will state whether the newly created organization block has additional parameters.

See also

Block folder (Page 4714)

Creating functions and function blocks (Page 4716)

Creating data blocks (Page 4716)

Using blocks from libraries (Page 4718)

Entering a block title (Page 4723)

Entering a block comment (Page 4724)

11.6.1.3 Creating functions and function blocks

Requirement

The "Program blocks" folder in the project tree is open.

Procedure

To create a function (FC) or a function block (FB), follow these steps:

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Function block (FB)" or "Function (FC)" button.
3. Enter a name for the new block.
4. Enter the properties of the new block.
5. To enter additional properties for the new block, click "Additional information".
   An area with further input fields is displayed.
6. Enter all the properties you require.
7. Activate the "Add new and open" check box if the block does not open as soon as it is created.
8. Confirm your entries with "OK".

Result

The new block is created. You can find the block in the project tree in the "Program blocks" folder.

See also

Block folder (Page 4714)

Creating organization blocks (Page 4715)

Creating data blocks (Page 4716)

Using blocks from libraries (Page 4718)

Entering a block title (Page 4723)

Entering a block comment (Page 4724)

11.6.1.4 Creating data blocks

Requirement

The "Program blocks" folder in the project tree is open.

Procedure

To create a data block, follow these steps:

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Select the type of the data block. You have the following options available to you:
   - To create a global data block, select the list entry "Global DB".
   - To create an ARRAY data block, select the "ARRAY DB" entry in the list.
   - To create an instance data block, select the function block to which you want to assign the instance data block from the list. The list contains only the function blocks that were previously created for the CPU.
   - To create a data block based on a PLC data type, select the PLC data type from the list. The list contains only the PLC data types that were previously created for the CPU.
   - To create a data block based on a system data type, select the system data type from the list. The list contains only those system data types that have already been inserted to program blocks in the CPU.
4. Enter a name for the data block.
5. Enter the properties of the new data block.
6. If you have selected an ARRAY DB as the data block type, enter the ARRAY data type and the high limit for the ARRAY.
   You can change the high limit for the ARRAY at any time in the property window of the created block. The ARRAY data type cannot be changed subsequently.
7. If you have selected a block that contains monitoring as the "Type", assign a ProDiag function block to the monitoring functions.
8. To enter additional properties of the new data block, click "Additional information".
   An area with further input fields is displayed.
9. Enter all the properties you require.
10. Activate the "Add new and open" check box if the block does not open as soon as it is created.
11. Confirm your entry with "OK".

Result

The new data block is created. You can find the data block in the project tree in the "Program blocks" folder.

Note

Retentivity of ARRAY data blocks

ARRAY data blocks and their components cannot be set retentively.

See also

Basic principles for programming of data blocks (Page 4795)

System data types (Page 2326)

Using blocks from libraries (Page 4718)

11.6.1.5 Using blocks from libraries

You can save blocks in the project library or in a global library, so that you can use them more than once within a user program. You can insert the blocks as master copies or as types.

See also: Library basics (Page 439)

Requirement

- The "Libraries" task card is displayed.
- No write protection is set for global libraries.

Adding blocks as master copies to the project library or to a global library

To add new blocks as master copies to the project library or to a global library, follow these steps:

1. Maximize the project library or the global library.
2. Use drag-and-drop to move the block you wish to add to the library to the "Master copies" folder or any one of the "Master copies" subfolders in the project library or a global library. Do not release the left mouse button until a small plus sign appears underneath the mouse pointer.

Or:

1. Copy the block you want to add as master copy.
2. Maximize the project library or the global library.
3. Right-click the "Master copies" folder or any of its subfolders.
4. Select "Paste" in the shortcut menu.

Adding blocks as types to the project library or to a global library

To add new blocks as types to the project library or to a global library, follow these steps:

1. Maximize the project library.
2. Drag-and-drop the block you want to add as a type into the "Types" folder or any subfolder of "Types" in the project library. Do not release the left mouse button until a small plus sign appears underneath the mouse pointer.
3. If necessary, drag the block from the "Types" folder of the project library into the "Types" folder of a global library. Or you can copy the block in the project library, and add it to a global library.

Or:

1. Copy the block you want to add as type.
2. Maximize the project library.
3. Right-click the "Types" folder or any of its subfolders.
4. Select "Paste" in the shortcut menu.
5. If necessary, drag the block from the "Types" folder of the project library into the "Types" folder of a global library. Or you can copy the block in the project library, and add it to a global library.

Using blocks of the project library or a global library

To use a block from the project library or a global library in your project, follow these steps:

1. Maximize the project library or the global library so that you can see the block you wish to use.
2. Use a drag-and-drop operation to move the block to the CPU block folder. If the selected insertion points is not allowed, the mouse pointer will appear as a circle with a slash.

Note

If you derive an instance from a type in a global library, the type is also inserted into the project library. The instance is then only linked to the type in the project library.

11.6.1.6 Copying and pasting blocks

Basics of copying and pasting blocks

Function

You can also create new blocks by copying existing blocks and pasting the copy. Note the following principles when copying to CPUs of the same device family:

- You can copy organization blocks (OBs), functions (FCs), function blocks (FBs), and global data blocks (DBs) without restriction.
- You can copy instance data blocks only for the same function block, since the assignment to the function block cannot be changed afterwards. However, the assignment is canceled if you copy the instance data block to a different CPU. If a function block with the same name exists there, the instance data block will be assigned to this function block. If you copy the instance data block together with the function block into the other CPU, the instance data block is assigned to the copy of the function block.

The various device families sometimes support different blocks, especially in the case of organization blocks. However, function blocks and functions can also be programmed on the

various devices with different access types. Therefore, not all blocks are supported on all devices. Note the following principles when copying to a different device family:

- Copying to an S7-1200 CPU:
  - Organization blocks with "Optimized" access type can be copied to an S7-1200. If the copied OB type is supported by the S7-1200 CPU, the copied OB retains the properties of its event. You must, however, compile it again.
  - Although organization blocks with the "Standard" access type can be copied to an S7-1200, they are not supported by the CPU.
  - Function blocks (FBs), functions (FCs) and global data blocks (DBs) with "Optimized" access type can be copied to an S7-1200. However, they must be recompiled after this.
  - Instance data blocks: If there is a function block in the target CPU with the name that was assigned to the instance data block in the source CPU, the instance data block is assigned to the function block in the target CPU. If you copy the instance data block together with the function block to which it was assigned in the source CPU into the target CPU, the instance data block is assigned to the copy of the function block.
- Copying to an S7-1500 CPU:
  - Organization blocks with "Optimized" access type can be copied to an S7-1500. If the copied OB type is supported by the S7-1500 CPU, the copied OB retains the properties of its event. You must, however, compile it again. OB types that are not supported receive a no parking symbol.
  - Organization blocks with "Standard" access type can be copied to an S7-1500. If the OB derives from an S7-300/400 CPU, it receives the standard event of the corresponding OB type. If the OB derives from an S7-1200/1500 CPU, it receives the properties of its event. However, it must be compiled again.
  - Function blocks (FBs), functions (FCs) and global data blocks (DBs) with "Optimized" access type can be copied to an S7-1500. However, they must be recompiled after this.
  - Instance data blocks: If there is a function block in the target CPU with the name that was assigned to the instance data block in the source CPU, the instance data block is assigned to the function block in the target CPU. If you copy the instance data block together with the function block to which it was assigned in the source CPU into the target CPU, the instance data block is assigned to the copy of the function block.
- Copying to S7-300/400 CPUs:
  - Organization blocks can be copied as required between S7-300 and S7-400.
  - Although organization blocks from S7-1200/1500 CPUs can be copied to S7-300/400 CPUs, they are not supported by the target CPU.
  - Function blocks (FBs), functions (FCs) and global data blocks (DBs) can be copied as required between S7-300 and S7-400.
  - Although function blocks (FBs), functions (FCs) and global data blocks (DBs) can be copied from S7-1200/1500 CPUs to S7-300/400 CPUs, they are not supported by the target CPU.

- Instance data blocks: If there is a function block in the target CPU with the name that was assigned to the instance data block in the source CPU, the instance data block is assigned to the function block in the target CPU. If you copy the instance data block together with the function block to which it was assigned in the source CPU into the target CPU, the instance data block is assigned to the copy of the function block.
- You may be able to cop function blocks (FBs) and functions (FCs) that are programmed with LAD or FBD and contain SCL networks to S7-300/400 CPUs, but they are not supported by the target CPU.

In the project tree, blocks that are not supported are indicated by the no parking symbol. Blocks with a no parking symbol cannot be edited, but only used again as copy source.

Note

When blocks are copied between different device families, it is possible that the copied block needs to be recompiled. This applies also to the coping of blocks between CPUs and software controllers. If the block has know-how protection, re-compilation is only possible with the correct password.

Copying data

With paste, all the block data is copied and forwarded to the copy. This data includes:

- Block interface tags
- All networks
- Comments in all existing compilations
- Messages defined in the block
- The entire program code of the copied block including the call instructions contained in the block.
  However, the following objects are not copied:
  - Called blocks and associated instance data blocks
  - Global tags that are referenced in the copied block
  - The PLC data types used in the copied block for the declaration of the block parameters and local tags
  - The technology objects used in the copied block

Avoiding name conflicts during pasting

When pasting copied blocks with identical names to already existing blocks, the following mechanisms are used to avoid name conflicts:

- Pasting the copied block into the same CPU:
  The copy of the block gets a name that is extended by a number. For example, if block "A" is copied, a possible name for the copy is "A_1". Consecutive numbering is not used, but rather the smallest free number. The copy of block "A" can also get the name "A_25", if no lower number is available.
- Pasting the copied block into another CPU:
  A dialog box opens in which you can select whether the block with the same name will be replaced or the copied block will be pasted with a duplicate designation (Name_Number).

Note

Number conflicts

Number conflicts may occur, if the pasted block has the same block number as an existing block. The block number is not automatically changed during pasting. This double number may have an effect on block calls. When you copy blocks you should therefore check the block number carefully and correct duplicate block numbers manually or using the block properties. Duplicate block numbers lead to a compilation error.

See also

Copying blocks (Page 4722)

Pasting blocks (Page 4723)

Copying blocks

Requirement

The "Program blocks" folder is opened in the project tree.

Procedure

To copy a block, follow these steps:

1. Right-click the block that you want to copy.
2. Select "Copy" in the shortcut menu.

Result

A copy of the block is now on the clipboard and can be pasted either into the same CPU or into another one.

See also

Basics of copying and pasting blocks (Page 4719)

Pasting blocks (Page 4723)

Pasting blocks

Requirement

You have copied a block.

Procedure

To paste a copied block and its data into a CPU, follow these steps:

1. In the project tree, open the folder structure for the CPU into which you want to paste the copied block.

   Note

   Please note that you can only paste the copied block into a CPU which supports the block programming language and type.

2. Right-click on the "Program blocks" folder.
3. Select "Paste" in the shortcut menu.
   - If you are pasting the block into the same CPU as the original block, "_<consecutive number>" will be appended to the name of the copy.
   - If you are pasting the block into a different CPU where a block of the same name already exists, the "Paste" dialog box opens. Select the required option and confirm with "OK".

See also

Basics of copying and pasting blocks (Page 4719)

Copying blocks (Page 4722)

11.6.1.7 Entering a block title

The block title is the title of the block. It is not the same thing as the block name, which was assigned when the block was created. The length of the block title is restricted to one row. You can enter the block title for open and closed blocks.

Requirement

A code block is available.

Enter block title for open block

To insert the block title in an open block, follow these steps:

1. Click on the title bar of the block in the program editor.
2. Enter the block title.

Enter block title for closed blocks

To insert the block title in a closed block, follow these steps:

1. Right-click the block in the project tree.
2. Select the "Properties" command in the shortcut menu.
   The dialog with the properties of the block opens.
3. Select the entry "Information" in the area navigation.
4. Enter the block title in the "Title" input field.
5. Confirm your entry with "OK".

See also

Creating organization blocks (Page 4715)

Creating functions and function blocks (Page 4716)

Entering a block comment (Page 4724)

11.6.1.8 Entering a block comment

You can use the block comment to document the entire code block. For example, you can indicate the purpose of the block or draw attention to special characteristics. You can enter the block comment for open and closed blocks.

Note

The block comment should not exceed 32,767 Unicode characters.

Requirement

A code block is available.

Enter block comment for open blocks

To insert a block comment in an open block, proceed as follows:

1. Click the small arrow in front of the block title.
   The right arrow becomes a down arrow, and the comment area is displayed.
2. Click "Comment" in the comment area.
   The "Comment" text passage is selected.
3. Enter the block comment.

Enter block comments for closed blocks

To insert the block comment in a closed block, follow these steps:

1. Right-click the block in the project tree.
2. Select the "Properties" command in the shortcut menu.
   The dialog with the properties of the block opens.
3. Select the entry "Information" in the area navigation.
4. Enter the block comment in the "Comment" input field.
5. Confirm your entry with "OK".

See also

Creating organization blocks (Page 4715)

Creating functions and function blocks (Page 4716)

Entering a block title (Page 4723)

11.6.2 Specifying block properties

11.6.2.1 Basics of block properties

Block properties

Each block has certain properties, which you can display and edit. These properties are used amongst other things to:

- Identify the block
- Display the memory requirements and the compilation status of the block
- Display the time stamp
- Display the reference information
- Specify the access protection

See also

Overview of block properties (Page 4726)

Block time stamps (Page 4730)

Displaying and editing block properties (Page 4732)

Setting the mnemonics (Page 4695)

11.6.2.2 Overview of block properties

Overview

The properties of the blocks are block and CPU-specific. Not all properties are therefore available for all blocks or in all CPU families. The following table gives an overview of block properties:

Group	Property	Description
General	Name	Unique block name within the station
	Constant name	Name of the constant pasted for the OB in the PLC tag table
	Type	Block type (cannot be changed)
	Number	Block number
	Event class	Event class of an OB (cannot be changed)
	Language	Programming language of the block
	Language in networks	Language used to program conditions in GRAPH blocks.
	Version	Version of the GRAPH block
	Process image partition number	Display of the process image partitions that are assigned to the organization block (cannot be changed)
	ARRAY data type	Data type of an ARRAY data block (cannot be changed)
	ARRAY limit	High limit of an ARRAY data block The "Move operations" section of the "Instructions" task card offers options for addressing of ARRAY data blocks.
Information	Title	Block title
	Comment	Block comment
	Version	Version number of the block
	Family	Block family name
	Author	Name of the author, company name, department name, or other names
	User-defined ID	ID created by the user

Group	Property	Description
Time stamp	Block	Times of creation and time of change of the block (cannot be changed)
	Interface	Time of creation of the block interface (cannot be changed)
	Code	Code modification time (non-editable)
	Data	Data modification time (non-editable)
	Load-relevant	Time of the last load-relevant modification (cannot be changed)
Compilation	Status	Details of the last compilation run (cannot be changed)
	Lengths	Details of the block lengths (cannot be changed)
	Simulation	Indicates whether the block can be simulated with SIMATIC S7-PLCSIM Advanced for compiling. This requires the "Support simulation during block compilation" option to be selected in the project properties. See also: Displaying properties of the project
	Library compliance	Indicates whether the block in the library can be used with know-how protection.
Protection	Protection	Setting up know-how protection and copy protection for the block See also: Protecting blocks (Page 4746)

Group	Property	Description
Attributes	IEC check	The compatibility of the operands in comparison operations and arithmetic operations are tested according to IEC 61131. You have to explicitly convert non-compatible operands. See also: Overview of data type conversion (Page 2465)
	Local error handling within block	Troubleshooting inside the block with the GET_ERROR or GET_ERR_ID instruction (cannot be changed). See also: AUTOHOTSPOT
	Optimized block access	The tag declaration for blocks with optimized access contains only the symbolic names of the data elements. The system automatically optimizes and manages the addresses. The CPU performance increases and access errors, for example, from SIMATIC HMI, are prevented. See also: AUTOHOTSPOT
	Create extended status information	Allows you to monitor all tags in an SCL block. This option does, however, increase the program memory space required and the execution times.
	Check ARRAY limits	Checks whether array indexes are within the range declared for an ARRAY during the runtime of an SCL block. The block enable output ENO is set to "0" if an array index is outside of the permitted range.
	Set ENO automatically	Checks whether errors occur in the processing of certain instructions during the runtime of an SCL block. The block enable output ENO is set to "0" if a runtime error occurs.
	Create minimized DB	Creates instance data blocks for GRAPH blocks of the S7-300 and S7-400 in minimized format. This option reduces the GRAPH FB memory space required, however you will only receive limited program status information.
	Skip steps	If the transitions before and after a step become valid simultaneously in a GRAPH block, the step will not be activated and will be skipped.
	Acknowledge supervision alarms	Any supervision error which occurs during the operation of a GRAPH block must be acknowledged before the program can continue.
	Permanent processing of all interlocks in manual operation	Permanently monitors all interlock conditions in a GRAPH block in manual mode.
	Initial value acquisition (Can only be activated as of block version V4.0)	The signal state of operands and the results of comparisons from a step or a transition are stored in the static parameters "CRIT_LOC", "CRIT_LOC_ERR", "CRIT" and "CRIT_FLT" of the GRAPH block interface.
	Lock operating mode selection	Prevents a GRAPH block operating mode being selected.
	Data block write-protected in the device	Indicates whether the data block is read-only in the target system, and cannot be overwritten while the

Group	Property	Description
		program is running. The option is available only for data blocks.
	Only store in load memory	On activation the data block is stored only in the load memory, occupies no space in the work memory, and is not linked into the program. The Task Card "Instructions" in the "Extended Instructions > Data block functions" pane provide options for handling the data blocks. The option is available only for data blocks.
	Start information	Here you specify the structure of the start information of the OB for S7-1500 CPUs: either like for S7-300 and S7-400 CPUs or optimized start information.
	Priority	Shows the priority set for organization blocks. The CPU family used and the type of the organization block determine whether you can change the priority.
	Parameter input via registers	Enables parameter input by means of registers in an STL block of S7-1500. This feature allows you to use the "Conditional call of blocks" (CC) and "Unconditional call of blocks" (UC) instructions in the block.
	Enable readback	Enables you to mark individual parameters of the block for readback. The "Tag readback" function is relevant if the block is used in a CFC chart.
	Block representation	Specifies how the block is represented in a CFC chart.
Time-of-day interrupt	Time-of-day interrupt	Parameters of the time-of-day interrupt OB: active (yes or no), execution, start date and time, local time or system time
Cyclic interrupt	Cyclic interrupt	Cycle time and phase shift of the cyclic interrupt OB
Triggers	Triggers	Display of start events of the hardware interrupt OB
Isochronous mode	Isochronous mode	Parameters of the isochronous mode interrupt OB: application cycles, automatic setting (yes or no), delay time. Also indicated is the PROFINET IO system or DP master system whose IO devices or DP slaves are assigned to the isochronous mode interrupt OB.
Download without reinitialization	Reserve in volatile memory	Defines the reserve in volatile memory that can be used for interface extensions. The number of currently available bytes is displayed in parentheses. This information is updated with each compilation.
	Activate download without reinitialization for retentive tags	Enables definition of a reserve in retentive memory.
	Reserve in retentive memory	Defines the reserve in retentive memory that can be used for interface extensions. The number of currently available bytes is displayed in parentheses. This information is updated with each compilation.

See also

Basics of block properties (Page 4725)

Block time stamps (Page 4730)

Displaying and editing block properties (Page 4732)

General settings for the PLC programming (Page 4708)

11.6.2.3 Block time stamps

Introduction

Blocks receive a number of different time stamps which show you when the block was created and when it was last changed. These time stamps are also used for automatic consistency checks before compilation.

Code block time stamps

The following time stamps are generated for code blocks (OBs, FBs, FCs):

- Block: Created on, Modified on
- Interface: Modified on
- Code/data: Modified on
- Load-relevant: Modified on

A time stamp conflict is displayed during compilation if the time stamp for the block calling is older than that of the interface for the block called.

Time stamps for code blocks are updated as follows:

- Block: The time stamp for the last block modification is always the same as the time stamp either of the interface or of the code depending on which area was modified last.
- Interface: The interface time stamp is updated each time the interface is modified. Even if you manually undo a change to the interface, for example change the name back, that is also a change which updates the time stamp. However, if you undo the change using the "Undo" function, the time stamp will be reset to the value it had before the undone change.
- Code/data: The code time stamp is updated each time the block code is changed. Even if you manually undo a change to the code, for example remove an instruction, that is also a change which will update the time stamp. However, if you undo the change using the "Undo" function, the time stamp will be reset to the value it had before the undone change.
- Load-relevant: The time stamp for "Load-relevant" is updated each time a code block is changed. These changes can affect the code, the data or the interface.

Global data block time stamps

The following time stamps are generated for global data blocks:

- Block: Created on, Modified on
- Interface: Modified on

- Data: Modified on
- Load-relevant: Modified on

Time stamp conflict is reported during compilation of a global data block based on a PLC data type if the time stamp of the global data block is older than the time stamp of the PLC data type used.

The time stamps for global data blocks are updated as follows:

- Block: The time stamp for the last change to a global data block is always the same as the time stamp for the interface and the data.
- Interface and data: The interface time stamps and the data are updated each time the global data block is changed. Even if you manually undo a change, for example remove a tag, that is also a change which will update the time stamp. However, if you undo the change using the "Undo" function, the time stamps will be reset to the value they had before the undone change.
- Load-relevant: The time stamp for "Load-relevant" is updated each time a global data block is changed. These changes can affect the data or the interface.

Instance data block time stamps

The following time stamps are generated for instance data blocks:

- Block: Created on, Modified on
- Interface: Modified on
- Data: Modified on
- Load-relevant: Modified on

A time stamp conflict will be reported during compilation of an instance data block if the interface time stamps of the instance data block are not identical to those of the function block.

The time stamps for instance data blocks are updated as follows:

- Block: The time stamp for the last change to an instance data block is always the same as the time stamp for the interface and the data.
- Interface and data: The interface time stamps and the data are updated each time the instance data block is changed. Even if you manually undo a change, for example cancel the tag retain setting, that is also a change which will update the time stamps. However, if you undo the change using the "Undo" function, the time stamps will be reset to the value they had before the undone change.
- Load-relevant: The time stamp for "Load-relevant" is updated each time an instance data block is changed. These changes can affect the data or the interface.

PLC data type time stamps

The following time stamps are generated for PLC data types:

- Block: Created on, Modified on
- Interface: Modified on
- Load-relevant: Modified on

The time stamps for PLC data types are updated as follows:

- Block: The time stamp for the last change to a PLC data type is always the same as the interface time stamp.
- Interface: The interface time stamp is updated each time the PLC data type is changed. Even if you manually undo a change, for example delete the content of a PLC data type, that is also a change which will update the time stamp. However, if you undo the change using the "Undo" function, the time stamp will be reset to the value it had before the undone change.
- Load-relevant: The time stamp for "Load-relevant" is updated each time a PLC data type is changed.

See also

Basics of block properties (Page 4725)

Overview of block properties (Page 4726)

Displaying and editing block properties (Page 4732)

11.6.2.4 Displaying and editing block properties

The properties of the blocks are block and CPU-specific. Not all properties are therefore available for all blocks or in all CPU families. Properties that can only be displayed are write-protected.

Displaying and editing properties of a closed block

To display and edit the properties of a closed block, follow these steps:

1. Open the "Program blocks" folder in the project tree.
2. Right-click the block whose properties you want to display or edit.
3. Select the "Properties" command in the shortcut menu.
   The properties dialog box of the block opens.
4. In the area navigation, click a group whose properties you want to display or edit.
5. Change the relevant property.
6. Confirm your entries with "OK".

Displaying and editing properties of an open block

To display or edit the properties of an open block, follow these steps:

1. Select the "Inspector window" check box in the "View" menu.
   The Inspector window opens.
2. Click the "Properties" tab.
   The properties of the block are shown in the "Properties" tab of the Inspector window.

3. In the area navigation, click a group whose properties you want to display or edit.
4. Change the relevant property.

Result

The properties of the block will be changed. The changes are not saved until the project is saved.

See also

Basics of block properties (Page 4725)

Overview of block properties (Page 4726)

Block time stamps (Page 4730)

11.6.3 Managing blocks

11.6.3.1 Opening blocks

You can open both the blocks in the project (offline blocks) and also the blocks in the device (online blocks).

To open an offline block, you have the following options:

- Open offline block directly
  You can open a block directly if the corresponding block folder is open in the project tree or in the overview window.
- Find and open offline block
  You can search for blocks within a project, a device, the "Program blocks" folder and in the Program editor, and then open these blocks.

Note the following special features when opening online blocks:

- You cannot edit online blocks.
- It is possible that another user can carry out a loading process on the selected CPU through joint parallel working on a CPU. Consequently, it can happen that the online block that you have open is deleted by the loading process if the block only exists in the device. In this case, the online block is closed and a message is displayed in the Inspector window.

Open offline block directly

To open an offline block directly, you have the following options:

1. Open the folder with the block you wish to open in the project tree or in the overview window.
2. Double-click on the block you wish to open.
   The block will open in the program editor.

Find and open offline block

To find an offline block and then open it, follow these steps:

1. You can start the search process for the following three types:
   - In the shortcut menu of the project, a device or "Program block" folder, select the "Open block/PLC data type" command or press the <F7> key.
   - Press the <F7> key in the open program editor.

   The "Open block/PLC data type" dialog box opens.
2. Enter the name, the address or the type of block you are looking for.
   The block list is filtered further with each letter entered. The block list is closed if there is no block that matches the input. You can show the complete block list at any time by clicking the button to the right of the text box. Keep in mind that there is no filtering in this case. If you want to filter for your inputs again, click the button once again.
3. In the block list, click the block you wish to open.
   The block is opened in the program editor and displayed as selection in the project tree.

Open online block

To open an online block directly, you have the following options:

1. Open the "Online access" folder in the project tree.
2. Click on the arrow symbol to the left of the interface to show all the objects that are arranged below the interface.
3. Double-click on the "Update accessible devices" command below the interface.
   All devices that are accessible via this interface are displayed.
4. Open the folder of the device that contains the block you want to open.
5. Open the "Program blocks" folder.
   All the device's blocks are displayed.
6. Double-click on the block you wish to open.
   The block will open in the program editor.

See also

Saving blocks (Page 4734)

Closing blocks (Page 4735)

Renaming blocks (Page 4735)

Deleting blocks offline (Page 4736)

Deleting blocks online (Page 4737)

Opening know-how protected blocks (Page 4759)

11.6.3.2 Saving blocks

Blocks are always saved together with the project. Faulty blocks can also be saved. This allows the error to be resolved at a convenient time.

Procedure

To save a block, follow these steps:

1. Select the "Save" or "Save as" command in the "Project" menu.
   See also: Saving projects

See also

Opening blocks (Page 4733)

Closing blocks (Page 4735)

Renaming blocks (Page 4735)

Deleting blocks offline (Page 4736)

Deleting blocks online (Page 4737)

11.6.3.3 Closing blocks

Procedure

To close a block, follow these steps:

1. Click the "Close" button in the title bar of the program editor.

Note

Note that the block will not be saved on closing.

See also

Opening blocks (Page 4733)

Saving blocks (Page 4734)

Renaming blocks (Page 4735)

Deleting blocks offline (Page 4736)

Deleting blocks online (Page 4737)

11.6.3.4 Renaming blocks

Requirement

The "Program blocks" folder is opened in the project tree.

Procedure

To change the name of a block, follow these steps:

1. Right-click the block that you want to rename.
2. Select the "Rename" command in the shortcut menu.
   The block name in the project tree changes to an input field.
3. Input the new name for the block.
4. Confirm your entry with the Enter key.

Result

The name of the block is now changed at all points of use in the program.

See also

Opening blocks (Page 4733)
Saving blocks (Page 4734)
Closing blocks (Page 4735)
Deleting blocks offline (Page 4736)
Deleting blocks online (Page 4737)

11.6.3.5 Deleting blocks offline

Requirement

The "Program blocks" folder opens in the project tree.

Procedure

To delete a block that exists offline, proceed as follows:

1. In the project tree in the "Program blocks" folder, right-click on the block that you want to delete.
2. Select the "Delete" command in the shortcut menu.
3. Confirm the safety prompt with "Yes".
   The block is deleted offline from the project.

Note

If you are deleting organization blocks, note that events may be assigned to these blocks. If you delete such organization block the program cannot respond to parameterized events.

See also

Opening blocks (Page 4733)

Saving blocks (Page 4734)

Closing blocks (Page 4735)

Renaming blocks (Page 4735)

Deleting blocks online (Page 4737)

11.6.3.6 Deleting blocks online (S7-300, S7-400)

Requirement

The "Program blocks" folder in the project tree is open.

Procedure

To delete a block that exists online, follow these steps:

1. In the "Program blocks" folder in the project tree, right-click on the block that you wish to delete from the device.
2. Select the "Delete" command from the shortcut menu.
   The "Delete" dialog opens.
3. Select the "Delete from device" check box.
4. Click "Yes".
   The block will be deleted from the device online.

See also

Opening blocks (Page 4733)

Saving blocks (Page 4734)

Closing blocks (Page 4735)

Renaming blocks (Page 4735)

Deleting blocks offline (Page 4736)

11.6.3.7 Deleting CPU data blocks

You may delete CPU data blocks both in offline and online mode.

Deleting CPU data blocks in offline mode

Proceed as follows to delete a CPU data block from the offline project:

1. Right-click the CPU data blocks that you want to delete in the project navigation, "Program blocks" folder.
2. Select the "Delete" command from the shortcut menu.
3. Confirm the safety prompt with "Yes".
   The CPU data block is deleted from the offline project.

Deleting CPU data blocks in online mode

Proceed as follows to delete a CPU data block from the online project:

1. Establish an online connection to the device containing the CPU data block that you want to delete.
2. Right-click the CPU data block that you want to delete from the device in the project navigation, "Program blocks" folder.
3. Select the "Delete" command from the shortcut menu.
   The "Delete" dialog opens.
4. Select the "Delete from device" check box.
5. Click "Yes".
   The CPU data block is deleted from the online device.

11.6.4 Using external source files

11.6.4.1 Basics of using external source files

Function

The textual programming languages STL and SCL allow you to enter the program code in any ASCII editor and save it as an external source file. This enables you to perform a range of tasks, for example:

- Declaring tags
- Specify block properties
- Programming blocks

Note

For blocks to be generated from the external source file in the TIA Portal, the source file must be saved with one of the following character encodings:

- ANSI
- UTF-8 with BOM (Byte Order Mark)
- UTF-16 with BOM, little and big endian format
- UTF-32 with BOM, little and big endian format
- UCS2 in little and big endian format

You can import these source files to your project and use them to generate blocks. You can generate a number of different blocks from one source file. Observe the following special features when generating blocks from a source file:

- A block that exists under the same name in the project will be overwritten.
- If a block was programmed with its absolute block number instead of a symbolic name in the source file and this number is already assigned by a block in the project, the new generated block is initially assigned the next free symbolic name.
- If you have not explicitly defined the access mode for a block in the external source file, the block access mode is set depending on the CPU series used:
  - Blocks generated for a CPU of the S7-1200/1500 series are assigned "optimized" access mode by default.
  - Blocks generated for a CPU of the S7-300/400 product range are assigned "standard" access mode by default.

  Organization blocks are the exception in this case, as they are always assigned the "standard" access mode by default, regardless of the CPU series. You have the option of changing the block access mode manually.
- It is possible that not all comments from the source file will be applied in the block.
- If you use absolute addressing in the external source file, a symbolic tag is created for each absolute address during the generation of the block. The names of these tags are made up of "Tag_" and a time stamp. This may result in relatively long tag names, which you can change manually if required.
- If you are using instructions in a different version in the external source file than in the target device, this may result in compilation errors. Correct the respective instructions in this case and start the compilation process once again. You can also select a different version for the target device.
- CPUs of the S7-300/400 series accept organization blocks with any name during import. If you import an OB with a standard name (for example, CYCL_EXC or CYC_INT5), the correct OB class is automatically assigned. If you import an OB with an unknown name, an OB 0 is created. You can change this number after the import and hereby assign the desired OB class. To change the OB number, select "Properties" in the shortcut menu.

You also have the option of saving existing blocks as external source files. You can either write only the program code of the selected block to a source file or additionally the program code of the blocks and PLC data types dependent on the selected block. When saving several blocks

to a source file, all blocks must use the same programming language and none of the blocks may have know-how protection.

See also

Rules for programming external source files (Page 4740)

Saving blocks as external source files (Page 4741)

Inserting external source files (Page 4743)

Opening and editing external source files (Page 4743)

Generating blocks from external source files (Page 4744)

11.6.4.2 Rules for programming external source files

An external source file basically consists of continuous text. To compile the source into blocks, certain structures and syntax rules must however be adhered to.

Syntax rules

The syntax of the instructions in external source files is very similar to that in the creation of user programs in the program editor with STL or SCL. Note, however, the following additional syntax rules:

- Block call
  When calling a block, transfer the parameters in the defined order in the ASCII editor. If you do not, the comment assignments for these lines may not match.
  Enter the parameters in brackets. The individual parameters are separated by a comma.
- Upper or lower case
  The program editor generally disregards upper or lower case. Jump labels are an exception to this. Character string entries are also case-sensitive ("STRING" data type). Keywords are displayed in upper case. For compilation purposes, however, case is disregarded; you can therefore specify keywords in upper or lower case or a mixture of the two.
- Semicolon
  Mark the end of every instruction and every tag declaration with a semicolon. You can enter several instructions per line.
- Forward slashes
  Begin every comment with two forward slashes (//) and end the comment with the <Enter> key.
- Use of String constants
  To avoid compilation errors when using String constants, enter the text in the language of the target project. You can explicitly use the data type WSTRING for CPUs of the S7-1200/1500 series by using the prefix "WString#":
  `Operand := WString#'<String constant>';`

See also

Basics of using external source files (Page 4738)

Saving blocks as external source files (Page 4741)

Inserting external source files (Page 4743)

Opening and editing external source files (Page 4743)

Generating blocks from external source files (Page 4744)

11.6.4.3 Saving blocks as external source files

You have the following options for saving STL and SCL blocks as external source files:

- Copying a block as text
  Note that with protected blocks only the information that you can also see without a password is copied to the source file.
- Generating external source file from blocks
  You can either write only the program code of the selected block to a source file or additionally the program code of all the blocks and PLC data types dependent on the selected block. All blocks must use the same programming language and none of the blocks may have know-how protection.

Copying a block as text

To copy a block as text and save it to an external source file, follow these steps:

1. In the project tree, right-click on the block that you want to save in an external source file.
2. Select the "Copy as text" command in the shortcut menu.
3. Open an external text editor.
4. Paste the copied text from the clipboard.
5. Save the file with one of the following file name extensions:
   – ".scl", if you want to generate a source file for an SCL block
   – ".stl", if you want to generate a source file for an STL block
   – ".DB" if you wish to generate a source file for a data block
   – ".UDT", if you want to generate a source file for a PLC data type

Generating external source file from blocks

To generate external source files from STL or SCL blocks, follow these steps:

1. In the project tree or in the overview window, select the STL or SCL blocks from which you want to generate an external source file.

   Note

   Select only blocks with the same programming language and without know-how protection.

2. Select the "Generate source from blocks > Selected blocks only" command from the shortcut menu to save only the selected block to an external source file. Or select the command "Generate source from blocks > Including dependent blocks" if you also want to save the program code of the dependent blocks and referenced PLC data types to the external source file.
   The "Save as" dialog opens.
3. Specify a path and a name for the external source file.
4. Click "OK".

You can also generate an opened STL or SCL block as external source file. To do this, follow these steps:

1. In the program editor, select the "Generate source from this block" command in the toolbar in the drop-down list "Generate source from block" to save only the open block in an external source file. Or select the command "Generate source from this block and all dependent blocks" if you also want to save the program code of the dependent blocks and referenced PLC data types to the external source file.
   The "Save as" dialog opens.
2. Specify a path and a name for the external source file.
3. Click "OK".

Result

Depending on the selected option, either only the block or the block together with the dependent blocks and PLC data types are saved as an external source file. You can include this source file in a project in the TIA Portal and use it to generate other blocks. However, please note that you can use STL source files only in S7-300/400/1500 CPUs.

See also

Basics of using external source files (Page 4738)

Rules for programming external source files (Page 4740)

Inserting external source files (Page 4743)

Opening and editing external source files (Page 4743)

Generating blocks from external source files (Page 4744)

11.6.4.4 Inserting external source files

Requirement

- An external source file is available and complies with the syntax and structure rules.
- The "External source files" folder is open in the project tree.

Procedure

Follow these steps to insert an external source file:

1. Double-click on the "Add new external file" command.
   The "Open" dialog box opens.
2. Navigate to and select existing external source files.
3. Confirm your selection with "Open".
   If there is already an external source file with this name in the project, the "Add external file" dialog opens. In this case, continue with step 4.
4. Select whether the new external source is to be renamed and inserted or whether the existing external source to be replaced.
5. Click "OK".

Result

The new source file will be added to the "External source files" folder.

See also

Basics of using external source files (Page 4738)

Rules for programming external source files (Page 4740)

Saving blocks as external source files (Page 4741)

Opening and editing external source files (Page 4743)

Generating blocks from external source files (Page 4744)

11.6.4.5 Opening and editing external source files

By linking the files with the file name extensions "stl" and "scl" to an editor you will be able to open and edit external source files with these formats directly. Use Notepad as editor because other text editors may not let you open several sources at the same time.

This means you do not need to insert the external source files again after editing.

Linking files with the file name extensions "stl" and "scl" file types to an editor

Proceed as follows to link files with the file name extensions "stl" and "scl" to an editor:

1. Open Windows Explorer.
2. Right-click on an STL file.
3. Select "Properties" in the shortcut menu.
   The "Properties" dialog box opens.
4. Click "Change" in the "File type" area on the "General" tab.
   The "Open with" dialog box opens.
5. Select the text editor you want to link to the "stl" file type.
6. Confirm your selection with "OK".
7. Close the "Properties" dialog with "OK".
8. Repeat steps 2 to 7 with an SCL file.

Opening and editing an external source file

To open an external source file, follow these steps:

1. Open the "External source files" folder in the project tree.
2. Double-click on the external source file you want to open.
   The external source file will open in the linked editor and can be edited.

See also

Basics of using external source files (Page 4738)

Rules for programming external source files (Page 4740)

Saving blocks as external source files (Page 4741)

Inserting external source files (Page 4743)

Generating blocks from external source files (Page 4744)

11.6.4.6 Generating blocks from external source files

Requirement

- The "External source files" folder is open in the project tree.
- An external source file is available.
- The external source file was saved with a permitted character encoding.

Procedure

To generate blocks from an external source file, follow these steps:

1. Open the external source file from which you wish to generate blocks.
2. Select the "Generate blocks from source" command in the "Edit" menu.
3. A prompt will appear telling you any existing blocks will be overwritten.
4. Confirm the safety prompt with "Yes".

Result

The external source file blocks will be generated and inserted in the "Program blocks" folder in the project tree. In the event of errors, information about the errors which have occurred will be displayed in the inspector window. This information, however, relates to the external source file and not to the block generated.

See also

Basics of using external source files (Page 4738)

Rules for programming external source files (Page 4740)

Saving blocks as external source files (Page 4741)

Inserting external source files (Page 4743)

Opening and editing external source files (Page 4743)

Specifying block properties (Page 4725)

11.7 Protecting blocks

11.7.1 Protecting blocks

Introduction

You have the following options for protecting your blocks

- Know-how protection
  With the help of know-how protection you can use a password to protect blocks of the OB, FB, FC type and global data blocks from unauthorized access.
- Copy protection:
  For S7-1200/1500 CPUs, you can also set up copy protection for code blocks which binds execution of the block to the CPU or the memory card with the defined serial number. When using copy protection for a block it makes sense to also know-how-protect the block to ensure that copy protection can only be revoked with a password.
- Write protection:
  You can assign write protection for code blocks, which prevent unintentional changes to the block.

Note the following points for know-how protection:

- You can not manually protect instance data blocks; they depend on the know-how protection of the assigned FB. This means that when you create an instance data block for a know-how protected FB, the instance data block also receives this know-how protection. This is independent of whether you explicitly create the instance data block or if it is created by a block call.
- The know-how protection acts as write protection in global blocks and in instance data blocks.
- You cannot provide ARRAY data blocks with know-how protection.
- The load memory requirement can be higher for know-how-protected blocks.

If a block is know-how protected, only the following data is readable without the correct password:

- Interface parameters Input, Output, InOut, Return, Static
- Block title
- Block comment
- Block properties
- Tags of global data blocks without specification of the location of use

The following actions can be performed with a know-how protected block:

- Copying and deleting
- Calling in a program
- Online/offline comparison
- Downloading

The code of the block, on the other hand, is protected from unauthorized reading and modification.

Note

Please note the following:

- S7-1200 version 1.0 and S7-300/400 (only GRAPH and SCL blocks): If you download a know-how-protected block to a device, no restore information is loaded along with it. This means that you cannot open a know-how-protected block again even with the correct password if you upload it from the device.
- Only the non-protected data is compared in offline-online comparison of know-how protected blocks.
- You will no longer be able to access the block if you do not have the password.
- If you add a know-how-protected block to a library, the master copy created will also be know-how protected.
- You will find information on whether or not the block can be used as protected library item in the block properties under "Compilation > Library conformance". For this purpose, the block cannot use any tags from the operand areas Output (Q), Input (I), Bit memory (M), Timer function (T) or Count function (C), and cannot access data blocks.
- Cross references to used tags, bit memories, inputs and outputs in know-how protected blocks are not displayed even after the correct password is entered.
- Automatic renumbering and manual renumbering without password of know-how-protected blocks is only possible for CPUs of the series S7-1500 and S7-1200 (V4). The know-how protection must also be set with a TIA Portal version V13 SP1 or higher. If these requirements for a know-how-protected block are not met, the loadable binary component of the block is no longer up-to-date when you change the number. In this case, the block must be recompiled before downloading it to a device. For know-how-protected blocks, this is only possible with the correct password. Keep this in mind particularly if you want to copy a know-how-protected block to another device in which there is already a block with the same number.
- Always pass on a project that includes know-how-protected blocks as a project archive or library archive. In this way, you ensure that the know-how protection cannot be bypassed.
- You cannot change the know-how protection settings for an open, know-how protected block.

Using a password provider

You have the option of connecting the TIA Portal to an external password management system via a password provider. For this, a project product version V14 SP1 and higher is required.

If you are using a password provider, you can select the password from a list of passwords when setting know-how protection, copy or write protection. Only the names of the passwords are displayed in the list, the passwords themselves are not visible. When a block is opened, the TIA Portal connects to the password provider and retrieves the corresponding password. If the password provider is not available at this time, you can only open the block fully if manual input is defined as the fallback strategy and the corresponding password is known. The name of the password is not sufficient here. Depending on the selected security setting in the password provider, you can change the password for a know-how protected block or revoke protection again.

The use of a password provider increases security for your program code, as it is not necessary to pass on passwords for know-how protected blocks to other editors. Instead, all editors can connect the same password provider in order to obtain access to the blocks. If an editor leaves the project, there is no need to change the password.

In the event of problems with connected password providers, you can record the activities of password providers and forward the recording to Customer Support for analysis.

See also

Connecting the password provider (Page 4748)

Show available password provider (Page 4750)

Recording activities of the password providers (Page 4751)

Define the type of password assignment (Page 4753)

Setting up and removing block copy protection (Page 4754)

Setting up and removing block write protection (Page 4755)

Setting up block know-how protection (Page 4758)

Opening know-how protected blocks (Page 4759)

Printing know-how protected blocks (Page 4760)

Changing a password (Page 4761)

Removing block know-how protection (Page 4762)

Archiving global libraries (Page 466)

11.7.2 Connecting the password provider

As an alternative to manual password input, a password provider can be connected to the TIA Portal. When using a password provider, you can select a password from a list of available passwords. When a know-how protected block is opened, the TIA Portal connects to the password provider and retrieves the corresponding password.

To connect a password provider you have to install and activate it. A settings file in which the use of a password provider is defined is also required.

Requirement

The current project is a V14 SP1 project.

Installing and activating password providers

To install and activate a password provider, proceed as follows:

1. Install the desired password provider in accordance with the installation instructions of the password provider.
2. Open the TIA Portal.

3. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
4. In the area navigation, select the "Password provider > Available password providers" group.
   The newly installed password provider is displayed in the table of available password providers.
5. Select the check box in the "Status" column to use the password provider.
   The installed password provider is now active and can be used.
6. Set the preferred password input such that the password provider is used.
   See also: Define the preferred password assignment (Page 4753)

Create a settings file for using password providers

To create a settings file for the use of password providers, proceed as follows:

1. Create an XML file with the content specified below. Use the coding "UTF-8".
2. In the setting tag with the name "EnablePasswordPreference", define whether a user can switch between manual password assignment and password assignment via a password provider.
3. In the setting tag with the name "PasswordPreference", define which type of password assignment is to be used.
4. Save the XML file.
5. Save the file in the following directory on your computer:
   C:\ProgramData\Siemens\Automation\Portal V14\CorporateSettings\

Content of settings file

The settings file must have the following content:

XML

```
<?xml version="1.0" encoding="utf-8"?>
<--! Copyright (R) Siemens AG 2016. All rights reserved. -->
<Settingshttp://www.w3.org/2001/XML-Schema-instance>
 <SettingsNode name="General">
   <SettingNode name="Application">
     <Setting name="EnablePasswordPreference">
       <Bool>true</Bool>
     </Setting>
     <Setting name="PasswordPreference">
       <String>AdvancedWithManualFallback</String>
     </Setting>
   </SettingNode>
 </SettingNode>
</Settings>
```

The following values are possible as the setting for "EnablePasswordPreference":

- "true": In the TIA Portal the user can switch between manual password assignment and password assignment via password providers.
- "false": Switching between manual password assignment and password assignment via password providers is not possible in the TIA Portal.

The default setting is "false".

The following values are possible as the setting for "PasswordPreference" :

- "Manual": Assignment of passwords in the TIA Portal is done manually.
- "Advanced": Passwords are assigned in the TIA Portal via a password provider.
- "AdvancedWithManualFallback": Passwords are assigned in the TIA Portal via a password provider. If the password provider is not available, the protected block can be accessed via manual password input.

The default setting is "Manual".

See also

Protecting blocks (Page 4746)

Show available password provider (Page 4750)

Recording activities of the password providers (Page 4751)

Setting up and removing block copy protection (Page 4754)

Setting up and removing block write protection (Page 4755)

Setting up block know-how protection (Page 4758)

Opening know-how protected blocks (Page 4759)

Printing know-how protected blocks (Page 4760)

Changing a password (Page 4761)

Removing block know-how protection (Page 4762)

11.7.3 Show available password provider

You can display the password providers which are available in the TIA Portal.

Requirement

- The current project is a V14 SP1 project.
- Password assignment via password provider was defined using a settings file.

Procedure

Proceed as follows to display the available password providers:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "Password provider > Available password providers" group.
   The available password providers are displayed in the overview table.

See also

Protecting blocks (Page 4746)

Connecting the password provider (Page 4748)

Recording activities of the password providers (Page 4751)

Define the type of password assignment (Page 4753)

Setting up and removing block copy protection (Page 4754)

Setting up and removing block write protection (Page 4755)

Setting up block know-how protection (Page 4758)

Opening know-how protected blocks (Page 4759)

Printing know-how protected blocks (Page 4760)

Changing a password (Page 4761)

Removing block know-how protection (Page 4762)

11.7.4 Recording activities of the password providers

In the event of problems in connection with connected password providers, you can record the activities of password providers. No passwords are recorded. You can then forward the recording to Customer Support for analysis.

Requirement

- The current project is a V14 SP1 project.
- A password provider is connected and activated.

Activate recording of the activities of the password providers

Proceed as follows to activate the recording:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "Password provider > Password provider analysis" group.

3. Select the "Record password provider activities" check box.
   The text field for the storage location of the recording file and the "Browse" button become active.
4. Enter a storage location for the recording file or select it using the "Browse" button.

Deactivate recording of the activities of password providers

Proceed as follows to deactivate recording:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "Password provider > Password provider analysis" group.
3. Deselect the "Record password provider activities" check box.
   The text field for the storage location of the recording file and the "Browse" button become inactive.

See also

Protecting blocks (Page 4746)

Connecting the password provider (Page 4748)

Show available password provider (Page 4750)

Define the type of password assignment (Page 4753)

Setting up and removing block copy protection (Page 4754)

Setting up and removing block write protection (Page 4755)

Setting up block know-how protection (Page 4758)

Opening know-how protected blocks (Page 4759)

Printing know-how protected blocks (Page 4760)

Changing a password (Page 4761)

Removing block know-how protection (Page 4762)

11.7.5 Define the type of password assignment

The TIA Portal offers the following options for the type of password assignment for blocks for know-how protection, copy protection and write protection:

- "Manual password input"
  With manual password input, you enter the password directly in the dialog for the password request.
- "Selection via password provider"
  With password input via an external password provider, you can select the password using a name from the list of available passwords. If the password provider is not available, it is not possible to set protection. The block can only be accessed later if there is a connection to the password provider.
- "Selection via password provider with manual input as fallback strategy for access"
  Like password input via an external password provider. If the password provider is not available and you know the password, you can, however, open the block for editing by entering the password manually.

Manual password input is the default setting.

Requirement

The following requirements must be met in order that password assignment via an external password provider can be used

- The current project is a V14 SP1 project.
- Password assignment via password provider was defined using a settings file.
- The change of the type of password assignment was allowed in the settings file.
- A password provider is connected.

Procedure

Proceed as follows to define the type of password assignment:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "Password provider > Password provision" group.
3. Activate the desired type of password assignment.

See also

Protecting blocks (Page 4746)

Connecting the password provider (Page 4748)

Show available password provider (Page 4750)

Recording activities of the password providers (Page 4751)

Setting up and removing block copy protection (Page 4754)

Setting up and removing block write protection (Page 4755)
Setting up block know-how protection (Page 4758)
Opening know-how protected blocks (Page 4759)
Printing know-how protected blocks (Page 4760)
Changing a password (Page 4761)
Removing block know-how protection (Page 4762)

11.7.6 Setting up and removing block copy protection (S7-1200, S7-1500)

For S7-1200 /1500 CPUs, you can set up copy protection which binds execution of the block to a specific CPU or a specific memory card. The block can then only be executed if it is in the device with the set serial number. If you want to add the serial number automatically through a download process, you must specify a copy protection password. This password is queried during a download process of the block. You can change the password without having to remove the know-how protection of the block. When you change the copy protection settings or remove the copy protection, the password is retained until the next compilation process.

It is important that you also know-how-protect any block for which you have set up copy protection. If you do not, anyone can reset the copy protection.

Note

S7-1500 and S7-1200 V2.2 and higher: If you download a copy protected block to a device that does not match the specified serial number, the entire download operation will be rejected. This means that blocks without copy protection, too, will not be downloaded.

Setting up copy protection

To set up copy protection for a block, follow these steps:

1. Open the properties of the block for which you wish to set up copy protection.
2. Select the "Protection" entry in the area navigation in the block properties.
3. Select either "Bind to serial number of the CPU" or "Bind to serial number of the memory card" from the drop-down list in the "Copy protection" area.
4. Enter either the serial number directly or enable the option "Serial number inserted when downloading to a device or memory card" if the serial number is to be inserted automatically when you load.
   Continue with step 7 if you have entered the serial number directly. If the serial number is to be added automatically during the download process, you must also execute steps 5 and 6.
5. Click "Define password".
   The "Define protection" dialog opens.
6. If you are using manual password input, enter the password in the fields "New password" and "Confirm password".

7. If you are using a password provider, first select the password provider and then the password.
8. Confirm your entries with "OK".
   The status box provides the information that a copy protection password is defined.
9. You can now set up the know-how protection for the block in the "Know-how protection" area, if the block does not already have know-how protection.

Removing copy protection

To remove copy protection, follow these steps:

1. Open the properties of the block for which you wish to remove copy protection.

   Note

   If the block has know-how protection, you must open the block with the correct password in order to be able to change the copy protection settings.

2. Select the "Protection" entry in the area navigation in the block properties.
3. Select "No binding" in the drop-down list in the "Copy protection" area.

See also

Protecting blocks (Page 4746)

Connecting the password provider (Page 4748)

Show available password provider (Page 4750)

Recording activities of the password providers (Page 4751)

Define the type of password assignment (Page 4753)

Setting up and removing block write protection (Page 4755)

Setting up block know-how protection (Page 4758)

Opening know-how protected blocks (Page 4759)

Printing know-how protected blocks (Page 4760)

Changing a password (Page 4761)

Removing block know-how protection (Page 4762)

11.7.7 Setting up and removing block write protection

You can set up write protection for blocks of the type OB, FB, FC to prevent them from being changed unintentionally. Blocks with write protection can only be opened as "read only", the block properties can however still be edited.

Please note that write protection is not know-how protection, as you can remove write protection again at any time. If a block is write-protected, you cannot additionally set up know-how protection. Remove the write protection for the block if you want to set up know-how protection for it.

Setting up write protection

To set write protection for a block, follow these steps:

1. Open the properties of the block for which you wish to set up write protection.
2. Select the "Protection" entry in the area navigation in the block properties.
3. In the area "Write protection" select "Define password".
   The "Define protection" dialog opens.
4. If you are using manual password input, enter the password in the fields "New password" and "Confirm password".
5. If you are using a password provider, first select the password provider and then the password.
6. Confirm your entries with "OK".
   The status box provides the information that a write protection password is defined. The text of the button changes to "Change password".
7. Select the "Write protection" check box.
   The "Access protection" dialog will open.
8. Enter the correct password or select the correct password if you are using a password provider.
   Write protection is enabled and will be active the next time the block is opened.

Disabling write protection

Proceed as follows to disable write protection:

1. Open the properties of the block for which you wish to disable write protection.
2. Select the "Protection" entry in the area navigation in the block properties.
3. In the area "Write protection" select the "Write protection" check box.
   The "Access protection" dialog will open.
4. Enter the correct password or select the correct password if you are using a password provider.
   The write protection for the block is disabled the block can be edited again the next time the block is opened. The password is retained, so that you can enable write protection again at any time.

Enable write protection.

Proceed as follows to enable write protection:

1. Open the properties of the block for which you wish to enable write protection.
2. Select "Protection" in the area navigation in the inspector window.
3. In the area "Write protection" select the "Write protection" check box.
   The "Access protection" dialog will open.
4. Enter the correct password or select the correct password if you are using a password provider.
   The write protection for the block is reset and will be active the next time the block is opened.

Changing the write protection password

To change the write protection password, follow these steps:

1. Open the properties of the block for which you wish to change the write protection password.
2. Select the "Protection" entry in the area navigation in the block properties.
3. Click the "Change password" button.
   The "Change protection" dialog opens.
4. If you are using manual password input, enter the old password in the field "Old password" and the new password in the field "New password". Also enter the new password in the "Confirm password" box
5. If you are using a password provider, first select the password provider and then the new password.
6. Confirm your entries with "OK".
   The write protection password for the block is changed.

Note

It is always possible to change the write protection password for the block. This depends on whether write protection is active or not.

Removing write protection

To set write protection for a block, follow these steps:

1. Open the properties of the block for which you wish to remove write protection.
2. Select the "Protection" entry in the area navigation in the block properties.
3. Click the "Change password" button.
   The "Change protection" dialog opens.
4. If you are using manual password input, enter the current password in the field "Old password".
5. If you are using a password provider, first select the password provider and then the current password.
6. Click "Remove"
   The write protection for the block is removed. Unlike deactivation of write protection, the password is also deleted.

See also

Protecting blocks (Page 4746)

Connecting the password provider (Page 4748)

Show available password provider (Page 4750)

Recording activities of the password providers (Page 4751)

Define the type of password assignment (Page 4753)
Setting up and removing block copy protection (Page 4754)
Setting up block know-how protection (Page 4758)
Opening know-how protected blocks (Page 4759)
Printing know-how protected blocks (Page 4760)
Changing a password (Page 4761)
Removing block know-how protection (Page 4762)

11.7.8 Setting up block know-how protection

You can set up know-how protection for blocks in the devices in your project.

Procedure

To set up block know-how protection, follow these steps:

1. Select the blocks with no know-how protection which you want to protect.
2. Select the command "Know-how protection" in the "Edit" menu.
   The "Define protection" dialog opens.
3. If you are using manual password input, enter the password in the fields "New password" and "Confirm password".
4. If you are using a password provider, first select the password provider and then the password.
5. Confirm your entries with "OK".
6. Close the "Define protection" dialog with "OK".

Result

The blocks selected will be know-how-protected. Know-how protected blocks are marked with a lock in the project tree. The password entered is valid for all blocks selected.

See also

Protecting blocks (Page 4746)
Connecting the password provider (Page 4748)
Show available password provider (Page 4750)
Recording activities of the password providers (Page 4751)
Define the type of password assignment (Page 4753)
Setting up and removing block copy protection (Page 4754)
Setting up and removing block write protection (Page 4755)

Opening know-how protected blocks (Page 4759)

Printing know-how protected blocks (Page 4760)

Changing a password (Page 4761)

Removing block know-how protection (Page 4762)

11.7.9 Opening know-how protected blocks

If the use of a password provider is set as the type of password assignment, the TIA Portal connects to the password provider when a protected block is opened. If the connection is successful, there is no need to enter a password. If the password provider is not available, you can only open the block fully if you had defined manual input as the fallback strategy and know the corresponding password. The name of the password is not sufficient here.

You can only open multiple know-how protected blocks at once if they are protected with the same password.

Requirement

- "Manual password input" is set as the type of password assignment.
- "Selection via password provider with manual input as fallback strategy for access" is set and there is no connection to the password provider.

Procedure

To open a know-how protected block, follow these steps:

1. Double-click on the block you wish to open.
   The "Access protection" dialog will open.
2. Enter the password for the know-how protected block.
3. Confirm your entry with "OK".

Result

The know-how protected block will open provided you have entered the correct password. However, the block will remain know-how protected. If you copy the block or add it to a library, for example, the copies will also be know-how protected.

Once you have opened the block, you can edit the program code and the block interface of the block for as long as the block or TIA Portal is open. The password must be entered again the next time the block is opened. If you close the "Access protection" dialog with "Cancel", the block will open but the block code will not be displayed and you will not be able to edit the block.

See also

Protecting blocks (Page 4746)
Connecting the password provider (Page 4748)
Show available password provider (Page 4750)
Recording activities of the password providers (Page 4751)
Define the type of password assignment (Page 4753)
Setting up and removing block copy protection (Page 4754)
Setting up and removing block write protection (Page 4755)
Setting up block know-how protection (Page 4758)
Printing know-how protected blocks (Page 4760)
Changing a password (Page 4761)
Removing block know-how protection (Page 4762)

11.7.10 Printing know-how protected blocks

You can only print complete know-how protected blocks if they have been opened with the correct password. If you print a closed block or if the block was not opened with the correct password, only the non-protected block data will be printed.

Procedure

To print a know-how protected block in full, follow these steps:

1. Open the know-how protected block you wish to print.
   See also: Opening know-how protected blocks (Page 4759)
2. Select the "Print" command in the "Project" menu.
   The "Print" dialog will open.
3. Select the printer in the "Name" field.
4. Click "Advanced" to modify the Windows printer settings.
5. Select the documentation information set in the "Document information" drop-down list that you want to use for the frame layout.
6. Under "Print objects/area" select whether you want to print all objects or the complete area, or only a selection.
7. Under "Properties" select the print scope.
   - Select "All" to print the complete block.
   - Choose "Visible" to print all the information within the block that is visible on the screen.
   - Select "Compact" to print a shortened form of the block.

8. Click "Preview" to generate a print preview in advance.
   A print preview is created in the work area.
9. Click "Print" to start the printout.

See also

Printing project contents (Page 380)

Protecting blocks (Page 4746)

Connecting the password provider (Page 4748)

Show available password provider (Page 4750)

Recording activities of the password providers (Page 4751)

Define the type of password assignment (Page 4753)

Setting up and removing block copy protection (Page 4754)

Setting up and removing block write protection (Page 4755)

Setting up block know-how protection (Page 4758)

Changing a password (Page 4761)

Removing block know-how protection (Page 4762)

11.7.11 Changing a password

If you are using a password provider, whether you are able to change the password for know-how protected blocks depends on the specifications of the password provider.

Procedure

To change the password, follow these steps:

1. Select the know-how protected blocks for which you want to change the password.

   Note

   You can only change the password for several blocks at once if all blocks selected have the same password.

2. Select the command "Know-how protection" in the "Edit" menu.
   The "Change protection" dialog opens.
3. If you are using manual password input, enter the old password in the field "Old password" and the new password in the field "New password". Also enter the new password in the "Confirm password" box
4. If you are using a password provider, first select the password provider and then the new password.
5. Close the "Change protection" dialog with "OK".

See also

Protecting blocks (Page 4746)
Connecting the password provider (Page 4748)
Show available password provider (Page 4750)
Recording activities of the password providers (Page 4751)
Define the type of password assignment (Page 4753)
Setting up and removing block copy protection (Page 4754)
Setting up and removing block write protection (Page 4755)
Setting up block know-how protection (Page 4758)
Opening know-how protected blocks (Page 4759)
Printing know-how protected blocks (Page 4760)
Removing block know-how protection (Page 4762)

11.7.12 Removing block know-how protection

If you are using a password provider, whether you are able to remove the know-how protection of a block depends on the specifications of the password provider.

Procedure

To remove block know-how protection, follow these steps:

1. Select the blocks for which you want to remove know-how protection.

   Note

   You can only remove know-how protection for several blocks at once if all blocks selected have the same password.

2. Select the command "Know-how protection" in the "Edit" menu.
   The "Change protection" dialog opens.
3. If you are using manual password input, enter the password for know-how protection of the block in the field "Old password". This is not required if you are using a password provider.
4. Click the "Remove" button.

Result

Know-how protection will be disabled for the blocks selected.

See also

Protecting blocks (Page 4746)
Connecting the password provider (Page 4748)

Show available password provider (Page 4750)
Recording activities of the password providers (Page 4751)
Define the type of password assignment (Page 4753)
Setting up and removing block copy protection (Page 4754)
Setting up and removing block write protection (Page 4755)
Setting up block know-how protection (Page 4758)
Opening know-how protected blocks (Page 4759)
Printing know-how protected blocks (Page 4760)
Changing a password (Page 4761)

11.8 Declaring the block interface

11.8.1 Overview of the block interface

Introduction

The interface contains the declarations of local variables and constants that are used within the block. The variables are subdivided into two groups:

- Block parameters that form the block interface when it is called in the program.
- Local data that are used for storage of intermediate results.

You use the variable declaration to define the call interface of a block in the program and the names and data types of variables and constants that you want to use in the block.

The interface of function blocks also defines the structure of the instances that are assigned to the function block.

Block parameters

The following table shows the types of block parameters:

Type	Section	Function	Available in
Input parameters	Input	Parameters whose values are read by the block.	Functions, function blocks and some types of organization blocks
Output parameters	Output	Parameters whose values are written by the block.	Functions and function blocks
In/out parameters	InOut	Parameters whose values are read by the block when it is called, and whose values are written again by the block after execution.	Functions and function blocks
Return value	Return	Value that is returned to the calling block.	Functions

Depending on the type of block opened, additional sections may be displayed.

Local data

The following table shows the types of local data:

Type	Section	Function	Available in
Temporary local data	Temp	Tags that are used to store temporary intermediate results. Temporary local data are retained for only one cycle. If you use temporary local data, you have to make sure that the values are written within the cycle in which you want to read them. Otherwise the values will be random.	Functions, function blocks and organization blocks **Note**: Temporary local data is not shown in instance data blocks.
Static local data	Static	Tags that are used for storage of static intermediate results in the instance data block. Static data is retained until overwritten, which may be after several cycles. The names of the blocks, which are called in this code block as multiple instance, will also be stored in the static local data.	Function blocks
Constant	Constant	Constants with declared symbolic names that are used within the block.	Functions, function blocks and organization blocks **Note**: Local constants are not shown in instance data blocks.

See also

Valid data types in the block interface (Page 4768)

Setting the retentivity of local tags (Page 4769)

Using setpoints during commissioning (Page 4831)

11.8.2 Rules for declaring the block interface

11.8.2.1 General rules for declaring the block interface

Using block parameters

The following rules apply to the use of block parameters within the block:

- Input parameters may only be read.
- Output parameters may only be written.
- In/out parameters may be read and written.

Note

Maximum number of parameters per block

The maximum permitted number of formal parameters is sufficient for all usual application scenarios. The exact number depends on the selected data types, declaration subsections and other factors.

If the maximum number is exceeded, you receive a message during the compilation process. In this case, you can combine several parameters in a PLC data type (UDT) or in a global data block (DB) and transfer this as a block parameter.

Assigning default values to block parameters

You can assign default values to specific parameters in the function block interface. The possibility of the assignment depends on the declaration subsection and data type of the parameter.

The following table shows which parameters can be assigned a default value:

Parameter type	Section	Assignment of a default value is possible		
		Elementary data types	Structured data types	Parameter types
Input parameters	Input	X	X	-
Output parameters	Output	X	X	-
In/out parameters	InOut	X	- (1)	-
Static local data	Static	X	X	-
Temporary local data	Temp	-	-	-
Constants	Constant	X	-	-

(1) Exception: In blocks with optimized access, you have the option to use PLC data types as default values under certain conditions.

Note

You can find more information on permissible characters in tag names in the Siemens Industry Online Support in the following entries:

When should identifiers and operands be used in "quotation marks" in STEP 7 (TIA Portal)?

https://support.industry.siemens.com/cs/ww/en/view/109477857 (https://support.industry.siemens.com/cs/ww/en/view/109477857)

See also

Declaring predefined actual parameters (Page 4783)

11.8.2.2 Valid data types in the block interface (S7-300, S7-400)

Valid data types in the block interface in S7-300/400

The following table shows which data types you can assign to the parameters in the individual sections of the interface.

Section	Standard Data types	ARRAY STRUCT STRING DT	Parameter types	VOID	POINTER	ANY
Organization block						
Temp	X	X	-	-	-	X
Constant	X	X (3)	-	-	-	-
Function block						
Input	X	X	X	-	X	X
Output	X	X	-	-	-	-
InOut	X	X (1)	-	-	X	X
Static	X	X	-	-	-	-
Temp	X	X	-	-	-	X
Constant	X	X (3)	-	-	-	-
Function						
Input	X	X (1)	X	-	X	X
Output	X	X (1)	-	-	X	X
InOut	X	X (1)	-	-	X	X
Temp	X	X	-	-	-	X
Return	X	X	-	X	X	X (2)
Constant	X	X (3)	-	-	-	-

(1) STRING can only be defined in the standard length of 254 characters.

(2) In SCL, ANY is not permitted as a function value.

(3) Constants with the ARRAY or STRUCT data type are not permitted.

11.8.2.3 Valid data types in the block interface (S7-1200, S7-1500)

Valid data types in the block interface in S7-1200

The following table shows which data types you can assign in the parameters in the individual sections of the interface.

Section	Standard Data types	ARRAY STRUCT STRING / WSTRING DT	ARRAY [*]	VOID	VARIANT
Organization block					
Temp	X	X	-	-	X
Constant	X	X (1) (2)	-	-	-
Function block					
Input	X	X	-	-	X
Output	X	X	-	-	-
InOut	X	X (1)	X (4)	-	X
Static	X	X	-	-	-
Temp	X	X	-	-	X
Constant	X	X (1) (2)	-	-	-
Function					
Input	X	X (1)	X (4)	-	X
Output	X	X (1)	X (4)	-	X
InOut	X	X (1)	X (4)	-	X
Temp	X	X	-	-	X
Return	X	X (3)	-	X	-
Constant	X	X (1) (2)	-	-	-

(1) You cannot make length declarations for STRING and WSTRING in these sections. STRINGs always have the standard length 254; WSTRINGs the standard length 16832. A declaration in the format MyString[3] would not be permitted. WSTRING is only permitted in blocks with optimized access in these sections.

(2) Constants with the ARRAY or STRUCT data type are not permitted.

(3) Function values of the WSTRING data type must not be longer than 1022 characters.

(4) ARRAY[*] is available in blocks with optimized access as of firmware V4.2.

Valid data types in the block interface in S7-1500

The following table shows which data types you can assign in the parameters in the individual sections of the interface.

Section	Standard Data types	ARRAY STRUCT STRING / WSTRING DT	AR-RAY[*]	Parame-ter types	VOID	DB_ANY	POINT-ER	ANY	VAR-IANT
Organization block									
Temp	X	X	-	- (4)	-	X	-	X (3)	X
Constant	X	X (1) (5)	-	-	-	-	-	-	-
Function block									
Input	X	X	-	X	-	X	X	X	X
Output	X	X	-	-	-	X	-	-	-
InOut	X	X (1)	X(7)(8)	- (4)	-	X	X	X	X
Static	X	X	-	-	-	X	-	-	-
Temp	X	X	-	- (4)	-	X	-	X (3)	X
Constant	X	X (1) (5)	-	-	-	-	-	-	-
Function									
Input	X	X (1)	X (7)	X	-	X	X	X	X
Output	X	X (1)	X (7)	-	-	X	X	X	X
InOut	X	X (1)	X (7)	- (4)	-	X	X	X	X
Temp	X	X	-	- (4)	-	X	-	X (3)	X
Return	X	X (6)	-	-	X	X	X	x (2)	-
Constant	X	X (1) (5)	-	-	-	-	-	-	-

(1) You cannot make length declarations for STRING and WSTRING in these sections. STRINGs always have the standard length 254; WSTRINGs the standard length 16832. A declaration in the format MyString[3] would not be permitted. WSTRING is only permitted in blocks with optimized access in these sections.

(2) In SCL, ANY is not permitted as a function value.

(3) ANY can only be used in blocks with standard access in the "Temp" section.

(4) The "INSTANCE" parameter type is the only exception permissible in the "TEMP" and "InOut" sections.

(5) Constants with the ARRAY or STRUCT data type are not permitted.

(6) Function values of the WSTRING data type must not be longer than 1022 characters.

(7) ARRAY[*] is available in blocks with optimized access as of firmware V2.0.

(8) ARRAY[*] is only available when the "Set in IDB" attribute is not set.

11.8.2.4 Setting the retentivity of local tags

Introduction

Function blocks store their data in an instance. To prevent data loss in the event of power failure, you can mark the data as retentive. This data is stored in a retentive memory area. The option of setting the retentivity depends on the set access type of the function block.

Retentive behavior in blocks with standard access

In blocks with standard access you cannot set the retentive behavior of individual tags. You can only define them as retentive in the assigned instance. All tags contained in the block are then considered as retentive.

Retentivity for optimized block access

In data blocks with optimized access you can define the retentive behavior of individual tags.

For structured data type tags, the retentivity setting always applies to the entire structure. You can make no individual retentivity setting for individual elements within the structure.

You cannot create retentive tags of the structured data type in the "InOut" section. In/out parameters with structured data type, for example ARRAY, STRUCT, or STRING, are always non-retentive.

The following settings are available:

- Retentive
  The values of the tags or the structure are available even after a power failure.
- Non-retentive
  The values of the tags or the structure are lost in the event of a power failure.
- Set in IDB
  The retentivity can be set in the instance data block. The setting that is made in the instance data block than applies, however, centrally to all tags that are selected with "Set in IDB".

See also

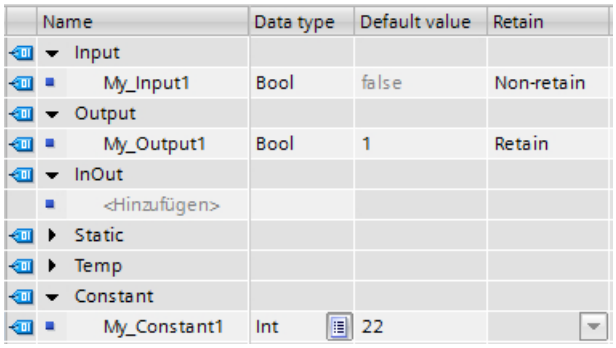
Properties of local tags and constants (Page 4784)

11.8.3 Declaring the block interface

11.8.3.1 Layout of the block interface

Layout of the block interface

The following figure shows the structure of the block interface. The number of columns and sections varies depending on the type of block.



Meaning of the columns

The following table shows the meaning of the individual columns. You can show or hide the columns as required. The number of columns displayed varies depending on the CPU series and the type of the open object.

Column	Explanation
	Symbol you can click on to drag-and-drop an element to a program for use as an operand.
Name	Name of the element.
Data type	Data type of the element.
Offset	Relative address of a tag within the block. The column is only visible in blocks with standard access. Note: Many instructions from the SIMATIC system libraries have the "optimized block access" property and therefore do not occupy any fixed memory addresses. No offset is displayed for these instructions, even if you use them as multi-instance in a block with standard access.

Column	Explanation
Default value	Value with which you can preassign specific tags in the interface of the code block, or the value of a local constant. Specification of the default value is optional for tags. If you do not specify any value the predefined value for the indicated data type is used. For example, the value "false" is predefined for BOOL. The default value of a tag is applied as the start value in the corresponding instance data block. You can replace these values with instance-specific start values in the instance data block. Constants always have the default value declared in the block interface. They are not shown in instance data blocks and cannot be assigned instance-specific values there.
Retain	Marks a tag as retentive. The values of retentive tags are retained even after the power supply is switched off. This column is only visible in the interface of the function block with optimized access.
Visible in HMI engineering	Indicates whether a tag is visible by default in the HMI selection list.
Accessible from HMI/OPC UA	Indicates whether HMI/OPC UA can access this tag during runtime.
Writable from HMI/OPC UA	Indicates whether the tag from HMI/OPC UA can be written during runtime.
Setpoint value	Marks a tag as a setpoint value Setpoint values are the values that will probably have to be fine tuned during commissioning. The column is only available in the interface of function blocks.
Supervision	Indicates whether Supervision has been created for process diagnostics for the tag.
Comment	Comments on documentation of the element.

See also

Valid data types in the block interface (Page 4768)

Setting the retentivity of local tags (Page 4769)

Using setpoints during commissioning (Page 4831)

11.8.3.2 Declaring local tags and constants in the block interface

Requirement

The block interface is open.

Procedure

To declare a tag or constant of the elementary data type, follow these steps:

1. Select the desired declaration section in the block interface.
   You must declare a constant in the "Constant" section.
2. Enter a name for the element in the "Name" column.

3. Enter the required data type in the "Data type" column. You will be supported by Autocomplete during input.
4. For constants, enter a value in the "Default value" column.
5. Optional: Change the properties that are displayed in the other columns of the block interface.

Result

The element is created.

Syntax check

A syntax check is performed after each entry, and any errors found are displayed in red. You do not have to correct these errors immediately - you can continue editing and make any corrections later. However, you will not be able to compile the program if the tag declaration contains syntax errors.

Note

If you change the interface of a block, the calls of the block in the program will possibly become inconsistent. The call locations are automatically updated, if possible.

If an automatic updating is not possible, the inconsistent blocks have to be updated manually.

See also:

Updating block calls in LAD (Page 4900)

Updating block calls in FBD (Page 4957)

See also

Rules for declaring the block interface (Page 4765)

Basic information on start values (Page 4807)

Setting the retentivity of local tags (Page 4769)

Hiding parameters during the block call (Page 4782)

Declaring predefined actual parameters (Page 4783)

Properties of local tags and constants (Page 4784)

11.8.3.3 Declaring a local tag in the program editor

Requirement

The program editor is open.

Procedure

To declare a local tag, follow these steps:

1. Insert an instruction in your program.
   The "<???>", "<??.?>" or "..." strings represent operand placeholders.
2. Replace an operand placeholder with the name of the tag to be created.
3. Select the name of the element.
   If you want to declare multiple elements, select the names of all the elements to be declared.
4. Select the "Define tag" command in the shortcut menu.
   The "Define tag" dialog box opens. It displays a declaration table in which the name of the element is already entered.
5. To declare a local tag, select one of the following sections:
   - Local In
   - Local Out
   - Local InOut
   - Local Static
   - Local Temp
6. In the other columns, enter data type and comments.
7. Click the "Define" button to complete your entry.

Result

The declaration is written directly into the block interface and is valid within the entire block.

Note

If you change the interface of a block, the calls of the block in the program will possibly become inconsistent. The call locations are automatically updated, if possible.

If an automatic updating is not possible, the inconsistent blocks have to be updated manually.

See also:

Updating block calls in LAD (Page 4900)

Updating block calls in FBD (Page 4957)

See also

Editing tables (Page 278)

Basic information on start values (Page 4807)

Properties of local tags and constants (Page 4784)

Setting the retentivity of local tags (Page 4769)

Edit table of the block interface (Page 4788)

Rules for declaring the block interface (Page 4765)

11.8.3.4 Declaring tags of the ARRAY data type

Requirement

The block interface is open.

Procedure

To declare a tag of the ARRAY data type, follow these steps:

1. Select the appropriate declaration section in the interface.
2. Enter a tag name in the "Name" column.
3. In the "Data type" column, click the button for the data type selection.
   A list of the permissible data types is opened.
4. Select the "Array" data type.
   The "Array" dialog opens.
5. In the "Data type" text box, specify the data type of the array elements.
6. In the "ARRAY limits" input field, specify the high and low limit for each dimension.
   Example of a one-dimensional ARRAY:
   `Array[0..3] of Bool`
   Example of a three-dimensional ARRAY:
   `Array[0..3, 0..15, 0..33]of Bool`
   Example of a one-dimensional ARRAY with variable limits:
   `Array[*]of Bool`
   Example of a three-dimensional ARRAY with variable limits:
   `Array[*, *, *]of Bool`
   Example of a one-dimensional ARRAY with local constants as ARRAY limits
   `Array[#My_local_const1..#My_local_const2]of Bool`
   Example of a one-dimensional ARRAY with global constants as ARRAY limits
   `Array["My_global_const1".."My_global_const1"]of Bool`
7. Confirm your entry.
8. Optional: Change the properties of the tags that are displayed in the other columns of the block interface.

The tag of ARRAY data type is created.

Note

Availability of ARRAY[*]

ARRAY[*] is available in optimized blocks for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0. In functions (FC) you can use ARRAY[*] in all declaration subsections. In function blocks (FB) you can declare ARRAY[*] only as an in-out parameter in the "InOut" section.

Entering default values of ARRAY elements

To pre-assign default values to the individual elements of an ARRAY, follow these steps:

1. Click the triangle in front of the ARRAY data type tags.
   The ARRAY is expanded and the individual ARRAY elements are shown in separate lines.
2. Enter the desired values in the "Default value" column.

Adding comments for ARRAYs

To add a comment for the entire ARRAY, enter the comment in the top line of the ARRAY declaration. The comment is used as a preset comment for all lower-level ARRAY elements.

To provide specific comments for the individual elements of an ARRAY, follow these steps:

1. Click the triangle in front of the ARRAY data type tags.
   The ARRAY is expanded and the individual ARRAY elements are shown in separate lines.
2. Enter the desired values in the "Comment" column.

Note

Restriction when displaying ARRAY elements

Depending on the amount of work memory available on your computer, the following maximum limits apply when displaying ARRAYs:

- RAM < 8 GB: A maximum of 10,000 elements are displayed.
- RAM >= 8 GB: A maximum of 40,000 elements are displayed.

See also

ARRAY (Page 2286)

Rules for declaring the block interface (Page 4765)

Properties of local tags and constants (Page 4784)

Setting the retentivity of local tags (Page 4769)

11.8.3.5 Declaring tags of STRUCT data type

Requirement

The block interface is open.

Procedure

To declare a tag of the STRUCT data type, follow these steps:

1. Select the appropriate declaration section in the interface.
2. Enter a tag name in the "Name" column.
3. Enter "Struct" in the "Data type" column. You will be supported by autocompletion during input.
   An empty, indented row is inserted after the new tag.
4. Insert the first structural element in the first empty row.
   An additional empty row is inserted after the element.
5. Select a data type for the structure element.
6. Optional: Change the properties of the structural element that is displayed in the other columns of the block interface.
7. Repeat the step 4 to 7 for all additional structure elements.
   It is not necessary to end the structure explicitly. The structure ends with the last element that is entered.
8. To insert a new tag after the structure, leave a blank row after the end of the structure and then start the new tag in the second empty row.

Result

The tag of STRUCT data type is created.

Note

S7-1500: A maximum of 252 structures in one data block

A maximum of 252 structures is permitted in one data block. If you need additional structures, you must restructure your program. You can, for example, create the structures in several global data blocks.

See also

Using autocompletion (Page 4706)

Rules for declaring the block interface (Page 4765)

Properties of local tags and constants (Page 4784)

Setting the retentivity of local tags (Page 4769)

Editing tables (Page 278)

11.8.3.6 Declare variables of the STRING and WSTRING data types

Requirement

The block interface is open.

Procedure

To declare a variable the STRING or WSTRING data type, proceed as follows:

1. Select the appropriate declaration section in the interface.
2. Enter a variable name in the "Name" column.
3. Enter "STRING" or "WSTRING" in the "Data type" column. You will be supported by autocompletion during input.
4. Optional: Specify the maximum length of the string using square brackets after the keyword STRING or WSTRING. If you do not specify a maximum length, the string has a default length of 254 characters.
   Example of a WSTRING with the maximum length of 4:
   `WSTRING[4]`
   Example of a string whose maximum length is defined by a local constant:
   `STRING[#My_local_const1]`
   Example of a string whose maximum length is defined by a global constant:
   `STRING["My_global_const1"]`

Result

The variable of the STRING or WSTRING data type is applied.

11.8.3.7 Declaring tags based on a PLC data type

Requirement

A PLC data type is declared in the current CPU.

Procedure

To declare a tag based on a PLC data type, follow these steps:

1. Select the appropriate declaration section in the interface.
2. Enter the PLC data type in the "Data type" column. You will be supported by Autocomplete during input.
   The tag is created.
3. If you have already defined default values or comments for the tags within a PLC data type during declaration of the PLC data type, these are shown in gray. You can change these values in the block interface.
   The changed values are displayed in black and apply only to the specific point of use.

Note

If you change or delete PLC data types that are used in the block interface, the interface becomes inconsistent. To eliminate this inconsistency, you must recompile the program.

See also: Updating the block interface (Page 4791)

See also

Editing tables (Page 278)

11.8.3.8 Declaring higher-level tags

Introduction

To access data areas within a declared tag, you can overlay the declared tags with an additional declaration. This provides you with the option of addressing an already declared tag with a different data type. You can, for example, address the individual bits of a tag of WORD data type with an ARRAY of BOOL.

Overlaying tags

To overlay a tag with a new data type, follow these steps:

1. Open the block interface.
2. In the interface, select the tag that you want to overlay with a new data type.
3. Click "Add row" in the toolbar.
   A row is inserted after the tag to be overlaid. The overlaying tag must be declared in the row directly after the tag that is to be overlaid.
4. Enter a tag name in the "Name" column.

5. Enter the "AT" entry in the "Data type" column. You will be supported by Autocomplete in this step.
   The following is added to the entry in the "Name" column.
   "AT<Name of the higher-level tag>"
6. Click the data type selection button again and select the data type for the new tag.
   The tag is created. It points to the same data as the higher-level tag, however interprets this data with the new data type.

Removing overlay

To remove the overlay of a tag, follow these steps:

1. Select the overlaid tag that you want to remove.
2. Select the "Delete" command in the shortcut menu.
3. The overlay is removed.

See also

Editing tables (Page 278)

11.8.3.9 Declaring multi-instances

Requirement

- The function block to be called exists in project tree and is multi-instance capable.
- The block interface of the calling function block is open.

Declaring a multi-instance

To declare a function block to be called as a multi-instance, follow these steps:

1. In the "Name" column of the "Static" section, enter a designation for the block call.
2. In the "Data type" column, enter the symbolic name for the function block to be called.

Note

The program editor will declare the multi-instance automatically if you program a block call in a network and then specify in the "Call options" dialog that you want to call the block as a multiple instance.

Declaring multi-instances as ARRAY

To declare an ARRAY of multi-instances, follow these steps:

1. In the "Name" column of the "Static" section, enter a designation for the ARRAY.
2. Select the "Array" data type in the "Data type" column.
   The "Array" dialog opens.
3. In the "Data type" input field, enter the name of the function block for which an instance should be defined. The block name must be in quotation marks.
4. In the "ARRAY limits" input field, specify the high and low limit for each dimension.
   Example of a one-dimensional ARRAY:[0..3]
   Example of a three-dimensional ARRAY:[0..3, 0..15, 0..33]
5. Confirm your entry.

See also

Updating the block interface (Page 4791)

11.8.3.10 Declaring parameter instances

Introduction

The following table shows the options for declaring and using parameter instances in the block interface:

Declaration	Block type	Application
InOut	FB FC	You can then call the instance that you transfer as actual parameter within the function block.
Input	FC	You have read access to the data of the instance that you transfer as actual parameter. However, you cannot call the instance from within the block.
Output	FC	You have write access to the data of the instance that you transfer as actual parameter. However, you cannot call the instance from within the block.

Requirement

- The block interface is open.

Procedure

To declare a parameter instance, follow these steps:

1. Open the required section of the block interface (see table).
2. In the "Name" column, enter a designation for the in/out parameter to which the instance is transferred.
3. In the "Data type" column, enter the name of the function block for which an instance should be defined. The block name must be in quotation marks.

Note

Declaring parameter instances automatically

The program editor will declare the parameter instance automatically if you program a block call in a network and then specify in the "Call options" dialog that you want to call the block as a parameter instance.

11.8.3.11 Hiding parameters during the block call

Introduction

You can hide the block parameters when calling the block in LAD or FBD. Hidden parameters are not visible initially, but can be displayed via a small arrow at the bottom edge of the call box.

Two options are available for hiding a parameter:

- Hide
  The parameter is always hidden.
- Hide, if no parameter is assigned
  The parameter is hidden as long as it has not yet been interconnected. If you assign an actual parameter, the parameter is displayed at the call box.

Procedure

Proceed as follows to define whether a block parameter is to be displayed or hidden during the call:

1. Open the block interface.
2. Select a parameter in the block interface.
3. Open the "Properties" tab in the Inspector window.
4. Select the "Attributes" group in the area navigation.
5. Select one of the following options under "Visibility at block calls in LAD / FBD":
   - "Show"
   - "Hide"
   - "Hide, if no parameter is assigned"

Result

The parameter is displayed or hidden at the block call in LAD or FBD depending on the setting.

If a syntax error is found during the block call, the parameter cannot be hidden.

See also

Wiring hidden parameters (Page 4914)

Declaring predefined actual parameters (Page 4783)

11.8.3.12 Declaring predefined actual parameters

Introduction

You can specify during the declaration of a block parameter which actual parameter is to be used during the block call.

This can be useful if you are using program blocks as library elements and want to store information about the actual parameters to be used along with the library element.

In addition you have the option in LAD and FBD to hide parameters that have a valid predefined actual parameter during the block call. Hidden parameters are not visible initially, but can be displayed via a small arrow at the bottom edge of the call box.

Two options are available for hiding a predefined parameter:

- The parameter is always hidden.
- The parameter is hidden as long as the predefined actual parameter is assigned. However, when a different parameter is assigned, the parameter becomes visible.

Requirement

- The parameter is located in the section "Input", "Output" or "InOut".

Procedure

Proceed as follows to predefine an actual parameter:

1. Open the block interface.
2. Select a parameter in the block interface.
3. Open the "Properties" tab in the Inspector window.
4. Select the "Attributes" group in the area navigation.
5. Enter the required actual parameter in the "Predefined actual parameter" input box.
   Optional:
6. Select the "Hide" option under "Visibility at block calls in LAD / FBD".
7. As an additional option, select "Show if parameter assigned on block call is not identical to the predefined actual parameter". This option is only available if the "Hide" option was activated beforehand.

Result

- An actual parameter is predefined. If you save the program block as a library element, it thus also contains information about the actual parameter to be used.
- If the library element is used in the program, a check is made to determine whether the actual parameter you have predefined can be addressed. If so, it is automatically used as the actual parameter.
- If the actual parameter If the actual parameter is not found in the program, a syntax error is signaled and you have to carry out the parameter assignment manually. Parameters with an invalid assignment are not hidden.

See also

Hiding parameters during the block call (Page 4782)

11.8.3.13 Editing the properties of local tags and constants

Properties of local tags and constants

Properties

The table below provides an overview of the properties of local tags and constants:

Group	Property	Description
General	Name	Name of the element.
	Data type	Data type of the element.
	Default value	Value with which you can pre-assign specific tags in the interface of the code block, or the value of a local constant. Specification of the default value is optional for tags. If you do not specify any value the predefined value for the indicated data type is used. For example, the value "false" is predefined for BOOL. The default value of a tag is applied as the start value in the corresponding instance. You can then replace these adopted values with instance-specific start values.
	Comment	Comment for the element.

Group	Property	Description
Attributes	Retain	Marks the tag as retentive. The values of retentive tags are retained even after the power supply is switched off. This attribute is only available in the interface of the function block with optimized access.
	Visible in HMI engineering	Shows whether the tag is visible by default in the HMI selection list.
	Writable from HMI/ OPC UA	Indicates whether the tag from HMI/OPC UA can be written during runtime.
	Accessible from HMI/OPC UA	Indicates whether HMI/OPC UA can access this tag during runtime. Please note, however, that you cannot implement general access protection for the tag with the "Accessible from HMI/OPC UA" attribute. Read or write access from other applications is possible even if the attribute is not enabled.
	Visibility at block calls in LAD/FBD	Show: The parameter is always visible at block calls in LAD or FBD.
		Hide: The parameter is always hidden at block calls in LAD or FBD. The instruction box then has a small arrow on the lower edge. You can have the parameters displayed and interconnect them by clicking this arrow.
		Hide if no parameter is assigned: The parameter is always hidden at block calls in LAD or FBD as long as it has not yet been interconnected. If you assign an actual parameter, the parameter is displayed at the call box.
	Predefined actual parameter	Defines a parameter that is to be used as actual parameter during the block call.
	Show if the assigned actual parameter is not identical with the predefined actual parameter	The parameter is hidden at block calls in LAD or FBD as long as the predefined actual parameter has not yet been assigned. However, when a different parameter is assigned, the parameter becomes visible. This option is only active if you have selected the "Hide" parameter and have defined a predefined actual parameter.

Group	Property	Description
User-defined attributes	CFC_Configurable	Configurable Indicates whether a parameter is configurable in CFC.
	CFC_ForTest	For test Indicates whether a parameter is registered for the CFC test mode.
	CFC_Visible	Visible Indicates whether a parameter is visible in CFC.
	CFC_Interconnectable	Interconnectable Indicates whether a parameter is interconnectable in CFC.
	CFC_EnableTagReadback	Enable tag readback Indicates whether a parameter is relevant for the "Read back chart" function in CFC.
	CFC_EnumerationTexts	Enumeration texts Assigns a parameter to an enumeration in CFC.
	CFC_EngineeringUnit	Engineering unit Assigns a parameter to a unit in CFC.
	CFC_LowLimit	Low limit Defines the low limit for the parameter in CFC.
	CFC_HighLimit	High limit Defines the high limit for the parameter in CFC.

See also

Setting the retentivity of local tags (Page 4769)

Changing properties of local tags and constants (Page 4786)

Hiding parameters during the block call (Page 4782)

Declaring predefined actual parameters (Page 4783)

Changing properties of local tags and constants

Editing properties of an element in the block interface

To edit the properties of an element in the block interface, follow these steps:

1. Open the block interface.
2. Select the required element in the table.
3. Change the entries in the columns.

Editing properties of several elements in the block interface

You can also simultaneously set or reset some properties for multiple selected elements.

To change one of these properties for several elements, follow these steps:

1. Open the block interface.
2. Hold down the CTRL key.
3. In the required column, select each of the table cells whose value you want to change.
4. Select the "Set <property>" or "Reset <property>" command in the shortcut menu.

Editing properties in the properties window

To edit the properties of an individual tag or constant, follow these steps:

1. Select an element in the table.
   The properties of the element are shown in the Inspector window.
2. Change the entries in the inspector window.

Renaming tags directly in the program editor

To rename one or more elements, follow these steps:

1. Select one or more elements in the program.
2. Select the "Rename tag" command in the shortcut menu.
   The "Rename tag" dialog opens. It displays a declaration table with the selected elements.
3. Change the entries in the "Name" column.
4. Confirm the input by clicking the "Change" button.

Editing the data type or comment in the program editor

Proceed as follows to edit the data type or tag comment in the program editor:

1. Select the name of the tag.
2. Select the "Rewire tag" command in the shortcut menu.
   The "Rewire tag" dialog will open. The dialog shows a declaration table.
3. Change the entry in the "Data type" or "Comment" columns.
4. Click the "Change" button to confirm the input.

Effect in the program

In case of a change of the name, data type or address of a tag or constant, each location of use of the tag is automatically updated in the program.

Note

If you change the interface of a block, the program may become inconsistent. The inconsistencies are automatically updated, if possible.

If an automatic updating is not possible, the inconsistent calls are marked in red. You than have to manually updated the inconsistencies.

See also:

Updating block calls in LAD (Page 4900)

Updating block calls in FBD (Page 4957)

See also

Overview of the block interface (Page 4764)

Editing tables (Page 278)

Properties of local tags and constants (Page 4784)

Setting the retentivity of local tags (Page 4769)

Basic information on start values (Page 4807)

Updating the block interface (Page 4791)

Edit table of the block interface (Page 4788)

Rules for declaring the block interface (Page 4765)

11.8.3.14 Edit table of the block interface

Inserting table rows

Procedure

Proceed as follows to insert a row above the selected row:

1. Select the row in front of which you want to insert a new row.
2. Click the "Insert row" button on the toolbar of the table.

Result

A new row is inserted above the selected row.

See also

Editing tables (Page 278)

Inserting table rows

Procedure

Proceed as follows to insert a row below the selected row:

1. Select the row below which you want to insert a new row.
2. Click the "Add row" button on the table toolbar.

Result

A new empty row will be inserted below the selected row.

See also

Editing tables (Page 278)

Deleting tags

Procedure

Follow the steps below to delete elements:

1. Select the row with the element to be deleted. You can also select several rows by clicking on them one after the other while holding down the <Ctrl> key or by pressing and holding down <Shift> and clicking on the first and last row.
2. Select the "Delete" command in the shortcut menu.

See also

Editing tables (Page 278)

Automatically filling in successive cells

You can load the contents of one or several table cells into the cells below, automatically filling in the successive cells.

If you automatically fill in cells in the "Name" column, a consecutive number will be appended to each name. For example, "Motor" will become "Motor_1".

You can define individual or more cells as well as entire rows as source area.

If less rows exist in the open table than you want to fill, then you will first have to insert additional empty rows.

Requirement

- The table is open.
- Sufficient declaration rows are available.

Procedure

To automatically fill in successive cells, follow these steps:

1. Select the cells to be loaded.
2. Click the "Fill" symbol in the bottom right corner of the cell.
   The mouse pointer is transformed into a crosshair.
3. Keep the mouse button pressed and drag the mouse pointer downwards over the cells that you want to fill in automatically.
4. Release the mouse button.
   The cells are filled in automatically.
5. If entries are already present in the cells that are to be automatically filled in, a dialog appears. In this dialog you can indicate whether you want to overwrite the existing entries or insert new rows for the new tags.

See also

Editing tables (Page 278)

Show and hide table columns

You can show or hide the columns in a table as needed.

Procedure

To show or hide table columns, follow these steps:

1. Click a column header.
2. Select the "Show/Hide" command in the shortcut menu.
   The selection of available columns is displayed.
3. To show a column, select the column's check box.
4. To hide a column, clear the column's check box.
5. To hide or show several columns, click "More" and activate or deactivate the check box of the corresponding columns in the "Show/Hide" dialog.

Editing tags with external editors

To edit individual tags in external table editors, such as Excel, you can export or import these tags using copy and paste.

Note

Restriction when displaying and copying ARRAY elements

Depending on the amount of work memory available on your computer, the following maximum limits apply when displaying and copying ARRAYs from the block interface in an Excel table:

- RAM < 8 GB: A maximum of 10,000 elements are displayed and copied.
- RAM >= 8 GB: A maximum of 40,000 elements are displayed and copied.

Requirements

The block interface and an external editor are opened.

Procedure

To export individual tags to an external editor and import them again, follow these steps:

1. Click the "Expanded mode" button to show all elements of structured data types.
2. Select one or more tags.
3. Select "Copy" in the shortcut menu.
4. Switch to the external editor and paste the copied tags.
5. Edit the tags as required.
6. Copy the tags in the external editor.
7. Select the tags in the external editor.
8. Switch back to the block interface.
9. Select "Paste" in the shortcut menu.

11.8.4 Updating the block interface

Introduction

If you change or delete PLC data types or multiple instances that are used in the block interface, the interface will become inconsistent. To remedy this inconsistency, the interface has to be updated.

You have two options for updating the block interface:

- Explicit updating of the block interface.
  The used PLC data types and multiple instances will be updated. The instance data blocks that belong to the block are not implicitly updated during this process.
- Implicit updating during compilation.
  All used PLC data types and multiple instances as well as the related instance data blocks will be updated.

Explicit updating of the block interface

To explicitly update the block interface, follow these steps:

1. Open the block interface.
2. Select the "Update" command in the shortcut menu.

Implicit Updating during Compilation

Proceed as follows to implicitly update all uses of PLC data types and multiple instances as well as the instance data blocks during compilation:

1. Open the project tree.
2. Select the "Program blocks" folder.
3. Select the command "Compile > Software (rebuild all blocks)" in the shortcut menu.

See also

Declaring tags based on a PLC data type (Page 4778)

Editing tables (Page 278)

Basic information on start values (Page 4807)

Properties of local tags and constants (Page 4784)

Setting the retentivity of local tags (Page 4769)

Updating block calls in LAD (Page 4900)

Declaring multi-instances (Page 4780)

Edit table of the block interface (Page 4788)

Rules for declaring the block interface (Page 4765)

11.8.5 Extending the block interface (S7-1200, S7-1500)

Description

In order to enable the editing of PLC programs that have already been commissioned and that are running without error on a system, CPUs of the S7-1500 series and most CPUs of the S7-1200 V4 and higher series support the option of extending the interfaces of function blocks during runtime.

You can download the modified blocks without setting the CPU to STOP and without affecting the values of already loaded tags.

This is a simple means of implementing program changes. This load process (download without reinitialization) will not have a negative impact on the controlled process.

Principle of operation

Each function block is always assigned a default memory reserve. The memory reserve is not used initially. Activate the memory reserve if you decide on loading interface changes after having compiled and downloaded the block. All tags that you subsequently declare will be saved to the memory reserve. The subsequent download does not influence any tags that are already loaded or have a negative impact on runtime.

If you decide to review your program at a later time while the plant is not in operation, you are also provided an option of reworking the memory layout of individual or several blocks in a single pass. With this action, you move all tags from the reserve area to the regular area. The memory reserve is now cleared and made available for further interface extensions.

Requirements

This "Download without reinitialization" function is available if the following requirements are met:

- The project is in the "TIA Portal V12" format or a higher version.
- You are working with a CPU that supports "Download without reinitialization".
- The blocks were created in LAD, FBD, STL, or SCL.
- The blocks were created by the user, i.e. they are not included with the blocks delivered in your package.
- These blocks are assigned the optimized access attribute.

Basic steps

Perform the following steps if you want to extend the interface of a function block and then load the block without re-initialization.

1. All blocks have a default memory reserve of 100 bytes. You can adapt this memory reserve to suit your requirements.
2. Activate the memory reserve.
3. Extend the block interface.

4. Compile the block.
5. Download the block to the CPU as usual.

For more information on the various steps, refer to chapter "Loading blocks (S7-1200/1500) ".

Note

The full scope of the "Download without reinitialization" function is only available on CPUs of the S7-1500 and S7-1200 V4 series.

However, all CPU families support the option of extending the interface of function blocks and downloading newly declared tags without repercussion:

- You may add new tags in the "Temp" section and download these without influencing the process.
- You may create new tags of a structured data type in the "InOut" section and download these without influencing the process.

11.9 Programming data blocks

11.9.1 Basic principles for programming of data blocks

A data block (DB) is used to save the values that are written during execution of the program.

In contrast to the code block, the data block contains only tag declarations. It contains no networks or instructions. The tag declarations define the structure of the data block.

Types of data blocks

There are two types of data blocks:

- Global data blocks
  The global data block is not assigned to a code block. You can access the values of a global data block from any code block. A global data block contains only static tags.
  The structure of the global data block can be freely defined. In the declaration table for data blocks, you declare the data elements that are to contained in the global data block.
- Instance data blocks
  The instance data block is assigned directly to a function block (FB). The structure of an instance data block cannot be freely defined, but is instead determined by the interface declaration of the function block. The instance data block contains exactly those block parameters and tags that are declared there.
  However, you can define instance-specific values in the instance data block, for example, start values for the declared tags.

ARRAY data blocks (S7-1500)

ARRAY data blocks are global data blocks that consist of an ARRAY. This ARRAY can be based on any data type. For example, an ARRAY of a PLC data type (UDT) is possible. The DB contains no other elements besides the ARRAY. Because of their flat structure, ARRAY data blocks facilitate access to the ARRAY elements and their transfer to called blocks.

You use ARRAY data blocks to address ARRAYs indirectly: You can determine during runtime which ARRAY DB is read or written and how large it is.

The "Move operations" section of the "Instructions" task card offers options for addressing of ARRAY DBs.

Note

Retentivity of ARRAY data blocks

ARRAY data blocks and their components cannot be set retentively.

See also:

AUTOHOTSPOT

PLC data types as a template for global data blocks

PLC data types can be used as templates for the creation of global data blocks with identical data structures. You create the structure as PLC data type only once and then generate the required data blocks by assigning the PLC data type.

System data types as a template for global data blocks

System data types can also be used as templates for creating global data blocks with identical data structure. System data types already have a pre-defined structure. You insert the system data type in the program only once and then generate additional data blocks with an identical structure by assigning the system data type.

Access modes

There are two different modes of accessing data values in data blocks:

- Data blocks with optimized access (only S7-1200)
  Data blocks with optimized access have no fixed defined structure. In the declaration, the data elements are assigned only a symbolic name and no fixed address within the block. You access the data values in these block via symbolic names.
  The "Optimized block access" attribute is always enabled for ARRAY data blocks.
- Data blocks with standard access (all CPU families)
  Data blocks with standard access have a fixed structure. In the declaration, the data elements are assigned both a symbolic name and a fixed address within the block. You can access the data values in this block via the symbolic names or addresses.
  Data blocks with standard access are tailored to WORD limits. This means that they always occupy one WORD (16 bits) or a multiple thereof. If necessary, tags are automatically added to the data block during compilation so that the block fills the space to the next WORD.
  ARRAY data blocks with standard access are not possible.

Retentivity of data values

To prevent data loss in the event of power failure, you can store the data values in a retentive memory area.

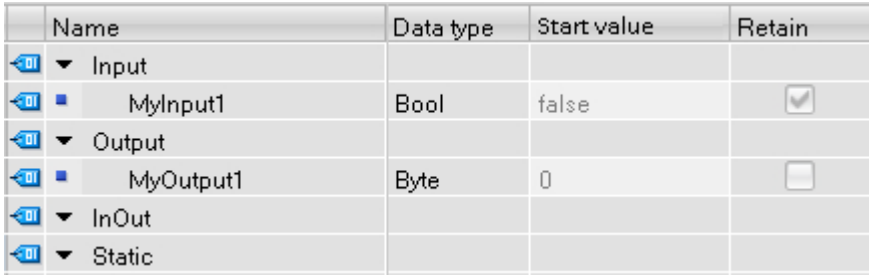
See also

Creating data blocks (Page 4716)

11.9.2 Structure of the declaration table for data blocks

Structure of the declaration table for data blocks

The figure below shows the structure of the declaration table for data blocks. The display will vary depending on type of block and type of access.



Display of instance-specific values

In instance data blocks, you can apply the already defined values from the interface of the assigned function block or define instance-specific start values. Values that are applied from the function block cannot be edited. You can replace the grayed-out values with instance-specific values. Values that were already changed instance specific are not grayed out.

Meaning of the columns

The following table shows the meaning of the individual columns. You can show or hide the columns as required. The number of columns displayed varies depending on the CPU series.

Column	Explanation
	Symbol you can click to move or copy the tag. You can, for example, drag-and-drop the tag into a program and use it there as operand.
Name	Name of the tags.
Data type	Data type of the tags.
Offset	Relative address of the tags. The column is only visible in data blocks with standard access. Note: Many instructions from the SIMATIC system libraries have the "optimized block access" property and therefore do not occupy any fixed memory addresses. No offset is displayed for these instructions, even if you use them as multi-instance in a block with standard access.
Default value	Default value of the tag in the interface of a higher-level code block or in a PLC data type. The values contained in the "Default value" column can only be changed in the higher-level code block or PLC data type. The values are only displayed in the data block.

Column	Explanation
Start value	Value that the tag should assume at startup. The default values defined in a code block are used as start values during the creation of the data block. You can then replace these adopted values with instance-specific start values. Specification of an start value is optional. If you do not specify any value, the tag assumes the default value at startup. If a default is not defined either, the default value valid for the data type is used. For example, the value "FALSE" is specified as standard for BOOL.
Monitor value	Current data value in the CPU. This column only appears if an online connection is available and you click "Monitor".
Snapshot	Shows values that were loaded from the device.
Retain	Marks the tag as retentive. The values of retentive tags are retained even after the power supply is switched off.
Visible in HMI engineering	Shows whether the tag is visible by default in the HMI selection list.
Accessible from HMI/ OPC UA	Indicates whether HMI/OPC UA can access this tag during runtime.
Writable from HMI/ OPC UA	Indicates whether the tag from HMI/OPC UA can be written during runtime.
Setpoint value	Setpoint values are the values that will probably have to be fine tuned during commissioning. After commissioning, the values of these tags can be transferred to the offline program as start values and stored there.
Supervision	Indicates whether Supervision has been created for process diagnostics for the tag.
Comment	Comment to document the tags.

See also

Editing the properties of tags in data blocks (Page 4811)
Using setpoints during commissioning (Page 4831)
Creating data blocks (Page 4716)
Creating a data structure for global data blocks (Page 4802)
Basic information on start values (Page 4807)
Defining start values (Page 4807)
Setting retentivity (Page 4809)

11.9.3 Creating data blocks

Requirement

The "Program blocks" folder in the project tree is open.

Procedure

To create a data block, follow these steps:

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Select the type of the data block. You have the following options available to you:
   - To create a global data block, select the list entry "Global DB".
   - To create an ARRAY data block, select the "ARRAY DB" entry in the list.
   - To create an instance data block, select the function block to which you want to assign the instance data block from the list. The list contains only the function blocks that were previously created for the CPU.
   - To create a data block based on a PLC data type, select the PLC data type from the list. The list contains only the PLC data types that were previously created for the CPU.
   - To create a data block based on a system data type, select the system data type from the list. The list contains only those system data types that have already been inserted to program blocks in the CPU.
4. Enter a name for the data block.
5. Enter the properties of the new data block.
6. If you have selected an ARRAY DB as the data block type, enter the ARRAY data type and the high limit for the ARRAY.
   You can change the high limit for the ARRAY at any time in the property window of the created block. The ARRAY data type cannot be changed subsequently.
7. If you have selected a block that contains monitoring as the "Type", assign a ProDiag function block to the monitoring functions.
8. To enter additional properties of the new data block, click "Additional information".
   An area with further input fields is displayed.
9. Enter all the properties you require.
10. Activate the "Add new and open" check box if the block does not open as soon as it is created.
11. Confirm your entry with "OK".

Result

The new data block is created. You can find the data block in the project tree in the "Program blocks" folder.

Note

Retentivity of ARRAY data blocks

ARRAY data blocks and their components cannot be set retentively.

See also

Basic principles for programming of data blocks (Page 4795)

System data types (Page 2326)

Using blocks from libraries (Page 4718)

11.9.4 Updating data blocks

Introduction

Changes in the interface of a function block or a PLC data type can lead to the corresponding data blocks becoming inconsistent. These inconsistencies are marked in red in the declaration table and at the call point of the block. To remedy these inconsistencies, the data blocks must be updated.

You have three options to update block calls:

- Explicit updating in the declaration table for data blocks.
  The data block is updated. Changes from the interface of the assigned function block and changes to the used PLC data types are applied.
- Explicit updating in the program editor.
  The block calls in the open block will be updated. The associated instance data block is also adjusted in the process.
- Implicit updating during compilation.
  All block calls in the program as well as the used PLC data types and the corresponding instance data blocks are updated.

Explicit Updating in the Declaration Table for Data Blocks

To explicitly update an individual data block, follow these steps:

1. Open the data block.
2. Select "Update interface" in the shortcut menu.

Explicit Updating in the Program Editor

To update all block calls or a specific call within a block, follow these steps:

1. Open the block in the program editor.
2. Right-click on the instruction with the block call.
3. Select the "Update" command in the shortcut menu.
4. The "Interface update" dialog opens. This dialog shows the differences between the block interface in use and the changed interface of the called block.
5. If you want to update the block call, click "OK". To cancel the update, click "Cancel".

Implicit Updating during Compilation

To implicitly update all block calls and uses of PLC data types as well as the instance data blocks during the compiling, follow these steps:

1. Open the project tree.
2. Select the "Program blocks" folder.
3. Select the command "Compile > Software (rebuild all blocks)" in the shortcut menu.

See also

Changing the properties of tags in instance data blocks (Page 4814)

11.9.5 Extending data blocks (S7-1200, S7-1500)

Description

In order to enable the editing of PLC programs that have already been commissioned and that are running without error on a system, CPUs of the S7-1500 series and most CPUs of the S7-1200 V4 and higher series support the option of extending global data blocks during runtime.

You can download the modified blocks without setting the CPU to STOP and without affecting the values of already loaded tags.

This is a simple means of implementing program changes. This load process (download without reinitialization) will not have a negative impact on the controlled process.

Principle of operation

Each data block is always assigned a default memory reserve. The memory reserve is not used initially. Activate the memory reserve if you decide on loading interface changes after having compiled and downloaded the block. All tags that you subsequently declare will be saved to the memory reserve. A subsequent download has no impact on the values of tags that have already been loaded.

If you decide to review your program at a later time while the plant is not in operation, you are also provided an option of reworking the memory layout of individual or several blocks in a single pass. With this action, you move all tags from the reserve area to the regular area. The memory reserve is now cleared and made available for further interface extensions.

Requirements

This "Download without reinitialization" function is available if the following requirements are met:

- The project is in the "TIA Portal V12" format or a higher version.
- You are working with a CPU that supports "Download without reinitialization".
- The blocks were created in LAD, FBD, STL, or SCL.

- The blocks were created by the user, i.e. they are not included with the blocks delivered in your package.
- These blocks are assigned the optimized access attribute.

Basic steps

Perform the following steps if you want to extend the data block and then load the block without re-initialization.

1. All blocks have a default memory reserve of 100 bytes. You can adapt this memory reserve to suit your requirements.
2. Activate the memory reserve.
3. Extend the block interface.
4. Compile the block.
5. Download the block to the CPU as usual.

Reference

For more information on the various steps, refer to chapter "Loading blocks (S7-1200/1500) ".

11.9.6 Creating a data structure for global data blocks

11.9.6.1 Declaring tags of elementary data type

Requirement

A global data block is open.

Note

You cannot change the structure of instance data blocks and of data blocks based on a PLC data type directly, since the structures of these blocks are defined by the respective function block or the PLC data type.

The type of the data block is entered in the block properties.

Procedure

To declare a tag of the elementary data type, follow these steps:

1. Enter a tag name in the "Name" column.
2. In the "Data type" column, click the button for the data type selection.
   A list of the permissible data types is opened.

3. Select the desired data type.
4. Optional: Change the properties of the tags that are displayed in the other columns.
5. Repeat steps 1 to 4 for all tags that are to be declared.

See also

Displaying and editing block properties (Page 4732)

Declaring tags of the ARRAY data type (Page 4803)

Declaring tags of STRUCT data type (Page 4804)

Editing tables (Page 278)

Editing the declaration table for data blocks (Page 4817)

Defining start values (Page 4807)

Setting retentivity (Page 4809)

Online and diagnostic functions for data blocks (Page 4820)

11.9.6.2 Declaring tags of the ARRAY data type

Requirement

A global data block is open.

Procedure

To declare a tag of the ARRAY data type, follow these steps:

1. Enter a tag name in the "Name" column.
2. Enter the "Array" data type in the "Data type" column. You will be supported by Autocomplete in this step.
   The "Array" dialog opens.
3. In the "Data type" text box, specify the data type of the array elements.
4. In the "ARRAY limits" input field, specify the high and low limit for each dimension.
   Example of a one-dimensional ARRAY:
   `ARRAY [0..3] of Bool`
   Example of a three-dimensional ARRAY:
   `ARRAY[0..3, 0..15, 0..33] of Bool`
   Example of a one-dimensional ARRAY with local constants as ARRAY limits
   `ARRAY[#My_local_const1..#My_local_const2] of Bool`
   Example of a one-dimensional ARRAY with global constants as ARRAY limits
   `ARRAY["My_global_const1".."My_global_const1"] of Bool`
5. Confirm your entry.
6. Optional: Change the properties of the tags that are displayed in the other columns.

The tag is created but remains collapsed. To expand the ARRAY, click the triangle in front of the tag. Not that you cannot expand very large ARRAYs for reasons of clarity.

Note

Restriction when displaying ARRAY elements

Depending on the amount of work memory available on your computer, the following maximum limits apply when displaying ARRAYs:

- RAM < 8 GB: A maximum of 10,000 elements are displayed.
- RAM >= 8 GB: A maximum of 40,000 elements are displayed.

Entering start values of ARRAY elements

To set default start values for the individual elements of an ARRAY, follow these steps:

1. Click the triangle in front of the ARRAY data type tags.
   The ARRAY is expanded and the individual ARRAY elements are shown in separate lines.
2. Enter the required values in the "Start value" column.

Adding comments for ARRAYs

To add a comment for the entire ARRAY, enter the comment in the top line of the ARRAY declaration. The comment is used as a preset comment for all lower-level ARRAY elements.

To provide specific comments for the individual elements of an ARRAY, follow these steps:

1. Click the triangle in front of the ARRAY data type tags.
   The ARRAY is expanded and the individual ARRAY elements are shown in separate lines.
2. Enter the desired values in the "Comment" column.

See also

Defining start values (Page 4807)

Setting retentivity (Page 4809)

Editing the declaration table for data blocks (Page 4817)

ARRAY (Page 2286)

Using autocompletion (Page 4706)

11.9.6.3 Declaring tags of STRUCT data type

Requirement

A global data block is open.

Procedure

To declare a tag of the STRUCT data type, follow these steps:

1. Enter a tag name in the "Name" column.
2. Enter "Struct" in the "Data type" column. You will be supported by autocompletion during input.
   An empty, indented row is inserted after the new tag.
3. Insert the first structural element in the first empty row.
   An additional empty row is inserted after the element.
4. Select a data type for the structure element.
5. Optional: Change the properties of the structural element that is displayed in the other columns of the block interface.
6. Repeat the step 4 to 7 for all additional structure elements.
   It is not necessary to end the structure explicitly. The structure ends with the last element that is entered.
7. To insert a new tag after the structure, leave a blank row after the end of the structure and then start the new tag in the second empty row.

Result

The tag of STRUCT data type is created.

Enter start values of structure elements

To set default start values for the individual elements of a structure, follow these steps:

1. Click the triangle in front of the STRUCT data type tags.
   The structure opens and the individual structure elements are shown in separate rows.
   Note that, for reasons of clarity, you cannot expand very large structures.
2. Enter the required values in the "Start value" column.

Note

S7-1500: A maximum of 252 structures in one data block

A maximum of 252 structures is permitted in one data block. If you need additional structures, you must restructure your program. You can, for example, create the structures in several global data blocks.

See also

Basic information on STRUCT (Page 2282)

Defining start values (Page 4807)

Setting retentivity (Page 4809)

Editing the declaration table for data blocks (Page 4817)

Using autocompletion (Page 4706)

11.9.6.4 Declaring tags of the STRING data type

Requirement

A global data block is open.

Procedure

To declare a tag of STRING data type, proceed as follows:

1. Select the appropriate declaration section in the interface.
2. Enter a tag name in the "Name" column.
3. Enter "STRING" in the "Data type" column. You will be supported by autocompletion during input.
4. Optional: Specify the maximum length of the string using square brackets after the keyword STRING. If you do not specify a maximum length, the string has a default length of 254 characters.
   Example of a STRING with the maximum length of 4:
   `STRING[4]`
   Example of a string whose maximum length is defined by a local constant:
   `STRING[#My_local_const1]`
   Example of a STRING whose maximum length is defined by a global constant:
   `STRING["My_global_const1"]`

Result

The tag of STRING data type is created.

11.9.6.5 Declaring tags based on a PLC data type

Requirement

- A global data block is open.
- A PLC data type is declared in the current CPU.

Procedure

To declare a tag based on a PLC data type, follow these steps:

1. Enter a tag name in the "Name" column.
2. Enter the PLC data type in the "Data type" column. You will be supported by Autocomplete during input.

3. The tag is created.
4. Optional: Change the properties of the tags that are displayed in the other columns of the table.
   If you have already defined default values or comments for the tags within a PLC data type during declaration of the PLC data type, these are shown in gray. You can change these values in the block interface.
   The changed values are displayed in black and apply only to the specific point of use.

Note

If you change or delete PLC data types that are being used in the data block, the data block becomes inconsistent. To eliminate this inconsistency, you must recompile the program.

See also

Overview of the block interface (Page 4764)

Editing the declaration table for data blocks (Page 4817)

Using autocompletion (Page 4706)

11.9.7 Defining start values

11.9.7.1 Basic information on start values

Definition of "Start value"

The start value of a tag is a value defined by you which the tag assumes after a CPU startup.

The retentive tags have a special status. Their values take the defined start value only after a "cold restart". After a "warm restart", they retain their values and are not reset to the start value.

Definition of "Default value"

The structure of the data blocks can be derived from higher-level elements.

- An instance data block is based, for example, on the interface of a higher-level code block.
- A global data block can be based on a predefined PLC data type.

In this case you can define a default value for each tag in the higher-level element. These default values are used as start values during the creation of the data block. You can then replace these values with instance-specific start values in the data block.

Specification of an start value is optional. If you do not specify any value, the tag assumes the default value at startup. If a default is not defined either, the default value valid for the data type is used. For example, the value "FALSE" is specified as standard for BOOL.

See also

Defining start values (Page 4808)

Structure of the declaration table for data blocks (Page 4797)

Declaring local tags and constants in the block interface (Page 4772)

Applying values from the online program as start values (Page 4834)

11.9.7.2 Defining start values

Defining start values

To define the start values for the tags of a data block, follow these steps:

1. Open the data block.
   The "Default value" column shows the default values that were defined for the tags in the interface of a higher-level code block or in a PLC data type.
2. Click the "Expanded mode" button to show all elements of structured data types.
3. Enter the desired start values in the "Start value" column. The value must match the data type of the tag and should not exceed the range of the data type.
   The start values are defined. The tag takes the defined value at startup, provided it was not declared as retentive.

Resetting a tag to the default value

To reset a tag for which you have defined a start value to the default value, follow these steps:

1. Select a modified value in the table.
2. Delete the value.
   The default value is entered. The default value is displayed.

Resetting all tags to the default value

To reset to the default value all tags for which you have defined an start value, follow these steps:

1. Select the "Reset start values" icon in the toolbar.
   The default values are transferred to the "Start value" column. Write-protected start values are not overwritten.

See also

Basic information on start values (Page 4807)

Applying values from the online program as start values (Page 4834)

Editing the declaration table for data blocks (Page 4817)

11.9.8 Setting retentivity

11.9.8.1 Retentivity of tags in data blocks

Retentive behavior

To prevent data loss in the event of power failure, you can mark the data as retentive. This data is stored in a retentive memory area. The options for setting the retentivity depend on the type of data block and the type of block access that is set.

See also

Setting retentivity in an instance data block (Page 4809)

Setting retentivity in a global data block (Page 4810)

11.9.8.2 Setting retentivity in an instance data block

Introduction

In an instance data block, the editability of the retentive behavior depends on the type of access of the higher-level function block:

- Function block with standard access
  You can define the instance data both as retentive or non-retentive. Individual retentivity settings are not possible for individual tags.
- Function block with optimized access
  In the instance data block, you can define the retentivity settings of the tags that are selected in the block interface with "Set in IDB". With these tags also, you cannot individually set the retentive behavior for each tag. The retentivity setting has an impact on all tags that are selected in the block interface with "Set in IDB".

Setting Retentivity for Standard Access

To centrally set the retentivity of all tags in the data block with standard access, follow these steps:

1. Open the instance data block.
2. Select the check box in the "Retain" column of a tag.
   All tags are defined as retentive.
3. To reset the retentivity setting for all tags, clear the check box in the "Retain" column of a tag.
   All tags will be defined as non-retentive.

Setting Retentivity for Optimized Access

To set the retentive behavior of the tags that are selected with "Set in IDB" in data blocks with optimized access, follow these steps:

1. Open the instance data block.
2. Select the check box in the "Retain" column of a tag.
   All tags selected with "Set in IDB" in the data block interface are defined as retentive.
3. To reset the retentivity setting for the tags, clear the check box in the "Retain" column of a tag.
   All tags selected with "Set in IDB" in the data block interface will be defined as non-retentive.

See also

Retentivity of tags in data blocks (Page 4809)

Editing the declaration table for data blocks (Page 4817)

11.9.8.3 Setting retentivity in a global data block

Introduction

In a global data block, the editability of the retentive behavior depends on the type of access:

- Global data block with standard access
  You can define the data both as retentive or non-retentive. Individual retentivity settings are not possible for individual tags.
- Global data block with optimized access
  You can individually define the retentivity settings of the tags. For tags with structured data types, retentivity settings are transferred for all tag elements.

Note

Retentivity of ARRAY data blocks

ARRAY data blocks and their components cannot be set retentively.

Setting Retentivity for Standard Access

To centrally set the retentivity of all tags in the data block with standard access, follow these steps:

1. Open the global data block.
2. Select the check box in the "Retain" column of a tag.
   All tags are defined as retentive.
3. To reset the retentivity setting for all tags, clear the check box in the "Retain" column of a tag.
   All tags are defined as non-retentive.

Setting Retentivity for Optimized Access

To individually set the retentivity of all tags in data blocks with optimized access, follow these steps:

1. Open the global data block.
2. In the "Retain" column, select the check box for the tags for which you want to set a retentive behavior.
   The selected tag is defined as retentive.
3. To reset the retentivity setting for the tags, clear the check box in the "Retain" column of a tag.
   All selected tags are defined as non-retentive.

See also

Retentivity of tags in data blocks (Page 4809)

Editing the declaration table for data blocks (Page 4817)

11.9.9 Editing the properties of tags in data blocks

11.9.9.1 Properties of the tags in data blocks

Properties

The properties of a tag are displayed in the Inspector window when you select the tag in the declaration table or at a point of use in the program editor.

Note

Properties of ARRAY elements

When you use ARRAY accesses with variable index in the form `#Program_Alarm_1[#myIndex]` in your program, the specific ARRAY element to be used during runtime is not yet known at the time of programming. For this reason, when you select this type of ARRAY access in the program editor, the properties of the first ARRAY element are always displayed.

The following table provides an overview of the properties of tags in data blocks:

Group	Property	Description
General	Name	Name of the tags.
	Data type	Data type of the tags.
	Default value	Default value of the tag in the interface of a higher-level code block or in a PLC data type. The values contained in the "Default value" column can only be changed in the higher-level code block or PLC data type. The values are only displayed in the data block.
	Start value	Value that the tag should assume at CPU startup. The default values defined in a code block are used as start values during the creation of the data block. You can then replace these adopted values with instance-specific start values. Specification of an start value is optional. If you do not specify any value, the tag assumes the default value at startup. If a default is not defined either, the default value valid for the data type is used. For example, the value "FALSE" is specified as default value for BOOL.
	Comment	Comment on the tag.

Group	Property	Description
Attributes	Retain	Marks the tag as retentive. The values of retentive tags are retained even after the power supply is switched off. This attribute is only available in the interface of the function block with optimized access.
	Visible in HMI engineering	Shows whether the tag is visible by default in the HMI selection list.
	Writable from HMI/OPC UA	Indicates whether or not the tag can be written from HMI/OPC UA during runtime.
	Accessible from HMI/OPC UA	Indicates whether HMI/OPC UA can access this tag during runtime. Please note, however, that you cannot implement general access protection for the tag with the "Accessible from HMI/OPC UA" attribute. Read or write access from other applications is possible even if the attribute is not enabled.
	Visibility in block calls in LAD/FBD	Show: The parameter is always visible at block calls in LAD or FBD.
		Hide: The parameter is always hidden at block calls in LAD or FBD. The instruction box then has a small arrow on the lower edge. You can have the parameters displayed and interconnect them by clicking this arrow.
		Hide if no parameter is assigned: The parameter is always hidden at block calls in LAD or FBD as long as it has not yet been interconnected. If you assign an actual parameter, the parameter is displayed at the call box.
	Predefined actual parameter	Defines a parameter that is to be used as actual parameter during the block call.
	Show, if parameter assigned on block call is not identical to the predefined actual parameter	The parameter is hidden at block calls in LAD or FBD as long as the predefined actual parameter has not yet been assigned. However, when a different parameter is assigned, the parameter becomes visible. This option is only active if you have selected the "Hide" parameter and have defined a predefined actual parameter.

Group	Property	Description
User-defined attributes	CFC_Visible	Visible Indicates whether a parameter is visible in CFC.
	CFC_Configurable	Configurable Indicates whether a parameter is configurable in CFC.
	CFC_ForTest	For test Indicates whether a parameter is registered for the CFC test mode.
	CFC_Interconnectable	Interconnectable Indicates whether a parameter is interconnectable in CFC.
	CFC_EnableTagReadback	Enable tag readback Indicates whether a parameter is relevant for the "Read back chart" function in CFC.
	CFC_Enumeration-Texts	Enumeration texts Assigns a parameter to an enumeration in CFC.
	CFC_EngineeringUnit	Engineering unit Assigns a parameter to a unit in CFC.
	CFC_LowLimit	Low limit Defines the low limit for the parameter in CFC.
	CFC_HighLimit	High limit. Defines the high limit for the parameter in CFC.

See also

Changing the properties of tags in instance data blocks (Page 4814)

Changing the properties of tags in global data blocks (Page 4816)

Setting retentivity (Page 4809)

Defining start values (Page 4807)

11.9.9.2 Changing the properties of tags in instance data blocks

Instance-specific tag properties

Two options are available for defining the tag properties:

- The tag properties are applied from the interface of the assigned function block. Properties that are applied from the function block are displayed grayed out out in the columns of the declaration table. The "Name" and "Data type" properties are always applied.
- You define instance-specific properties.
  You can change some properties instance specific. Changeable values are, for example, "Comment" or "Visible in HMI". Properties that were changed instance specific are not grayed out in the columns of the declaration table. The instance-specific changes are retained, even if the interface of the higher-level function block is changed and the instance data blocks are subsequently updated.

Editing properties of an element in the declaration table

To edit the properties of an element, follow these steps:

1. Open the instance data block.
2. Select the required element in the table.
3. Change the entries in the columns.

Editing properties of several elements in the declaration table

You can also simultaneously set or reset some properties for multiple selected elements.

To change one of these properties for several elements, follow these steps:

1. Open the data block.
2. Hold down the CTRL key.
3. In the required column, select each of the table cells whose value you want to change.
4. Select the "Set <property>" or "Reset <property>" command in the shortcut menu.

Editing properties in the properties window

To edit the properties of an individual tag, follow these steps:

1. Select a tag in the table.
2. Select the "Properties" command in the shortcut menu.
   The properties window opens. It shows the properties of the tag in the "General" and "Attributes" areas.
3. Select the required area in the area navigation.
4. Change the entries in the text boxes.

Reset individual properties to the default value.

To reset individual tag properties to the value that was defined as default in the function block, follow these steps:

1. Select an instance-specific, modified value in the table.
2. Delete the value.
   The instance-specific value will be deleted and the default value from the interface of the function block entered. The default value is displayed grayed out.

See also

Updating data blocks (Page 4800)

Properties of the tags in data blocks (Page 4811)

Editing the declaration table for data blocks (Page 4817)

11.9.9.3 Changing the properties of tags in global data blocks

Introduction

Two options are available for defining the tag properties:

- The tag properties are applied from the PLC data type.
  Properties that are applied from the PLC data type are shown grayed out in the columns of the declaration table. The "Name" and "Data type" properties are always applied.
- You define specific properties.
  You can change some properties in the global data block. Changeable values are, for example, "Comment" or "Visible in HMI". Properties that were changed are not grayed out in the columns of the declaration table. The changes are retained, even if the PLC data type changes and the global data block is subsequently updated.

Editing properties of an element in the declaration table

To edit the properties of an element, follow these steps:

1. Open the global data block.
2. Select the required element in the table.
3. Change the entries in the columns.

Editing properties of several elements in the declaration table

You can also simultaneously set or reset some properties for multiple selected elements.

To change one of these properties for several elements, follow these steps:

1. Open the data block.
2. Hold down the CTRL key.
3. In the required column, select each of the table cells whose value you want to change.
4. Select the "Set <property>" or "Reset <property>" command in the shortcut menu.

Editing properties in the properties window

To edit the properties of an individual tag, follow these steps:

1. Select a tag in the table.
2. Select the "Properties" command in the shortcut menu.
   The properties window opens. It shows the properties of the tag in the "General" and "Attributes" areas.
3. Select the required area in the area navigation.
4. Change the entries in the text boxes.

Reset individual properties to the default value.

To reset individual tag properties to the value that was defined as default in the PLC data type, follow these steps:

1. Select a modified value in the table.
2. Delete the value.
   The default value from the PLC data type is entered. The default value is displayed grayed out.

See also

Properties of the tags in data blocks (Page 4811)

Editing the declaration table for data blocks (Page 4817)

11.9.10 Editing the declaration table for data blocks

11.9.10.1 Inserting table rows

Procedure

Proceed as follows to insert a row above the selected row:

1. Select the row in front of which you want to insert a new row.
2. Click the "Insert row" button on the toolbar of the table.

Result

A new row is inserted above the selected row.

See also

Editing tables (Page 278)

11.9.10.2 Inserting table rows

Procedure

Proceed as follows to insert a row below the selected row:

1. Select the row below which you want to insert a new row.
2. Click the "Add row" button on the table toolbar.

Result

A new empty row will be inserted below the selected row.

See also

Editing tables (Page 278)

11.9.10.3 Deleting tags

Requirements

A global data block is open.

Procedure

To delete a tag, follow these steps:

1. Select the row with the tag to be deleted. You can also select several rows by clicking on them one after the other while holding down the <Ctrl> key or by pressing and holding down <Shift> and clicking on the first and last row.
2. Select the "Delete" command in the shortcut menu.

Note

You cannot directly change the structure of instance data blocks and of global data blocks based on a PLC data type, since the structures of these blocks are defined in the higher-level object.

The type of the data block is entered in the block properties.

See also: Displaying and editing block properties (Page 4732)

See also

Editing tables (Page 278)

11.9.10.4 Automatically filling in successive cells

You can load the contents of one or several table cells into the cells below, automatically filling in the successive cells.

If you automatically fill in cells in the "Name" column, a consecutive number will be appended to each name. For example, "Motor" will become "Motor_1".

You can define individual or more cells as well as entire rows as source area.

If less rows exist in the open table than you want to fill, then you will first have to insert additional empty rows.

Requirement

- The table is open.
- Sufficient declaration rows are available.

Procedure

To automatically fill in successive cells, follow these steps:

1. Select the cells to be loaded.
2. Click the "Fill" symbol in the bottom right corner of the cell.
   The mouse pointer is transformed into a crosshair.
3. Keep the mouse button pressed and drag the mouse pointer downwards over the cells that you want to fill in automatically.
4. Release the mouse button.
   The cells are filled in automatically.
5. If entries are already present in the cells that are to be automatically filled in, a dialog appears. In this dialog you can indicate whether you want to overwrite the existing entries or insert new rows for the new tags.

11.9.10.5 Show and hide table columns

You can show or hide the columns in a table as needed.

Procedure

To show or hide table columns, follow these steps:

1. Click a column header.
2. Select the "Show/Hide" command in the shortcut menu.
   The selection of available columns is displayed.
3. To show a column, select the column's check box.
4. To hide a column, clear the column's check box.
5. To hide or show several columns, click "More" and activate or deactivate the check box of the corresponding columns in the "Show/Hide" dialog.

11.9.10.6 Editing tags with external editors

To edit individual tags in external table editors, such as Excel, you can export or import these tags using copy and paste.

Note

Restriction when displaying and copying ARRAY elements

Depending on the amount of work memory available on your computer, the following maximum limits apply when displaying and copying ARRAYs from the data block editor in an Excel table:

- RAM < 8 GB: A maximum of 10,000 elements are displayed and copied.
- RAM >= 8 GB: A maximum of 40,000 elements are displayed and copied.

Requirement

The data block and an external editor are opened.

Procedure

To export and re-import individual tags by drag-and-drop operation, follow these steps:

1. Click the "Expanded mode" button to show all elements of structured data types.
2. Select one or more tags.
3. Select "Copy" in the shortcut menu.
4. Switch to the external editor and paste the copied tags.
5. Edit the tags as required.
6. Copy the tags in the external editor.
7. Switch back to the declaration table.
8. Select "Paste" in the shortcut menu.

11.9.11 Online and diagnostic functions for data blocks

11.9.11.1 Editing data blocks in online mode

Introduction

When you load a data block to the CPU, there are two versions of the block:

- An offline version in the project on your programming device
- An online version on the CPU
  Data blocks can either be located only in the load memory, only in the work memory or in both memory areas.

Start values and actual values of tags

The values of the tags in the offline and online version of a data block are different because the tags in the online block can be changed by the user program during runtime.

When you declare a data block, you assign a start value to the tags. The start value is the value of the tag when the data block is called for the first time. This means the tag is "initialized" with the start value.

The start values of the tag are declared in the offline version of the block. When the data block is loaded to the CPU, it is transferred to the load memory with the start values. From there, the data block is loaded to the work memory of the CPU. When the block is loaded for the first time, the tags in the work memory are initialized with the declared start values. They can now be changed with the user program. The values of the tags in the work memory are referred to as "actual values".

Loading changed data blocks

When you make changes to a data block after initial loading, you are editing the offline version of the block. Depending on the existing conditions, subsequent loading of the data block has different effects on the actual values.

- Loading changes in "RUN":
  - If you have not made any changes to the structure of the data block, the actual values of all tags are retained. The program is processed further "bumplessly" Any changes to the start values in the offline data block are written to the load memory. They are not transferred to the CPU until the next transition from "STOP" to "RUN".
  - If you have changed the structure of the data block, all tags are reinitialized with their start values. Initializing the tags may have unwanted effects on the process because the process and the program may not be synchronous any longer.
    You can prevent this by using the memory reserve of the block. The memory reserve enables you to load data blocks in "RUN" without reinitialization. This means you can reload "bumplessly" even if you have changed the structure of the data block and have added new tags, for example.
    See also: AUTOHOTSPOT
- Loading changes in "STOP":
  - The start values of non-retentive tags are placed in the load memory of the CPU. The program runs with the new start values at the next transition from "STOP" to "RUN".
  - Retentive tags are not reinitialized. Your actual values are retained after loading in "STOP".
    To also reinitialize the retentive tags when loading in "STOP", select the blocks you want to load in the project tree and select the command "Compile > Software (rebuild all blocks)" in the shortcut menu. In the "Online" menu, select the command "Download and reset PLC program". The online blocks are deleted and overwritten by the new blocks. This reinitializes all tags, including the retentive tags.

Editing actual values with online and diagnostic functions

To manipulate actual values directly in online mode without having to reload data blocks, you can use the online and diagnostic functions of the TIA Portal.

See also: Overview of the online and diagnostic functions in data blocks (Page 4822)

Loading data blocks from a device

You can also load data blocks from a device again. This makes sense, for example, if the original offline project is not available during servicing. Please note that the two CPU families handle the values of tags differently when loading blocks from a device:

- S7-1200/1500
  CPUs of the S7-1200 and S7-1500 series store the start values with which a data block is loaded to the CPU in the load memory. If you load a data block from an S7-1200/1500 CPU, these start values are applied from the load memory to the offline data block again. Even if you have made value changes in the load memory during runtime by means of the "WRIT_DBL" instruction, the values originally loaded to the device are applied to the offline data block.
- S7-300/400
  CPUs of the S7-300/400 series also store the start values with which a data block is loaded to the CPU in the load memory. However, they cannot read the values back from the load memory. If you load a data block from a device, the current monitored values are loaded from the CPU and entered in the "Start values" column in the offline data block.

11.9.11.2 Overview of the online and diagnostic functions in data blocks

Overview of functions

The data block editor offers different options for editing data blocks in online mode. These functions directly access the actual values of the tags in the online program. Actual values are the values which the tags have at the current time during program execution in the CPU work memory.

The following table provides an overview of the online and diagnostic functions. Detailed descriptions of the individual functions can be found in the following chapters.

Button	Function	Description	S7-300/400	S7-1200/1500
	Monitor tags online (Page 4823)	Displays the actual values which the tags currently have in the CPU.	X	X
-	Modify individual actual values (Page 4824)	Modifies individual tags immediately and once only to specific values in the declaration table. The CPU then uses these values as actual values in the online program.	X	X
	Create a snapshot of the actual values (Page 4826)	Saves the actual values present at the current time as snapshot in the offline project. The snapshot always captures the actual values of all tags of the data block.	X	X
	Reinitialize actual values with a snapshot (Page 4828)	Loads the snapshot to the CPU as actual values. The CPU then uses these values as actual values in the online program.	-	S7-1200 V4.1 and higher S7-1500 V1.7 and higher

Button	Function	Description	S7-300/ 400	S7-1200/1500
	Apply snapshot values as start values (Page 4830)	Copies the snapshot to the start values in the offline program. The program runs with the new start values at the next transition from STOP to RUN. You can copy all start values or only the start values of the tags identified as "setpoint".	X	X
	Reinitialize actual values with start values (Page 4825)	Reinitializes the actual values of tags. The start values are written directly to the CPU work memory. You can copy all actual values or only the actual values of the tags identified as "setpoint".	X	X

See also

Initializing setpoint values in the online program (Page 4833)

Applying values from the online program as start values (Page 4834)

Editing data blocks in online mode (Page 4820)

Using setpoints during commissioning (Page 4831)

Defining start values (Page 4807)

11.9.11.3 Monitoring tags

You can monitor the actual values of the tags in the CPU directly in the declaration table.

Requirement

- An online connection is available.
- The data block has been loaded to the CPU.
- The program execution is active (CPU in "RUN").
- The data block is open.

Procedure

To monitor the tags, follow these steps:

1. Start monitoring by clicking the "Monitor all" button.
   The additional "Monitor value" column is displayed in the table. This shows the current data values.
   See also: Structure of the declaration table for data blocks (Page 4797)
2. End monitoring by clicking the "Monitor all" button again.

See also

Overview of the online and diagnostic functions in data blocks (Page 4822)

11.9.11.4 Modifying tags

You can modify an individual tag in the data block to a specific value. The CPU then uses this value as the actual value in the online program.

DANGER

Danger when changing tag values

Changing the tag values while the plant is operating can cause serious damage to property or injury to persons if there are functional disturbances or program errors.

Make sure that no dangerous situations can arise before you modify the tags.

Requirement

- An online connection to the CPU is available.
- The data block whose tags you wish to modify is identical offline and online.
- The data block is open.

Procedure

To modify an individual tag in the data block, follow these steps:

1. Start monitoring by clicking the "Monitor all" button.
   The additional "Monitor value" column is displayed in the table. This shows the current data values.
2. Double-click the tag to be modified.
   The "Modify value" dialog opens.
   If there is a snapshot, the value is already entered as default.
3. Enter the desired value in the "Modify value" input field and confirm with "OK".

Result

The tag will have the specified value once when you execute the modify job. The job is executed immediately and is not tied to the next cycle control point.

See also

Introduction to modifying tags (Page 5216)

Overview of the online and diagnostic functions in data blocks (Page 4822)

11.9.11.5 Loading start values as actual values

You can load the start values from the offline program to the CPU as actual values. The start values are loaded directly to the CPU work memory. The tags in the online block are reinitialized, and the CPU uses the new values as actual values in the online program. No distinction is made between retentive and non-retentive values. You can copy all actual values or only the actual values of the tags identified as "setpoint".

You can quickly bring the controlled process into a defined state this way.

You cannot reinitialize the following types of data blocks:

- Data blocks that are part of a reference project
- Fail-safe data blocks
- Data blocks generated by the CPU
- Data blocks with the attribute "Only store in load memory"
- Data blocks that only exist online

You can execute this function in "RUN" mode as well as "STOP" mode.

DANGER

Danger when changing tag values

Changing the tag values while the plant is operating can cause serious damage to property or injury to persons if there are functional disturbances or program errors.

If the volume of data to be transmitted is too large, the values may be transmitted in several cycles. If the program accesses tags before all values are completely transmitted, there is a risk that inconsistent value combinations may be created and processed.

Copying the values of elementary data types may also take several cycles. These values are potentially invalid until they have been completely transmitted. Dangerous states may occur if the program accesses these values before they have been completely transmitted.

To prevent dangerous states from occurring, please note the following:

- Make sure that the plant is in a safe state before you overwrite the actual values.
- Make sure that the program does not read or write the affected data during transmission.
- You may want to use the "Modify tag" function in the watch table or in the DB editor as an alternative.

Requirement

- You are using an S7-1200 V4.1 or higher or an S7-1500 V1.7 or higher.
- An online connection to the CPU is available.
- The data blocks whose actual values you want to reinitialize are identical offline and online.

Reinitializing actual values of an open block

To reinitialize all actual values or only the setpoints of a block with start values, follow these steps:

1. Open the data block.
2. Click the "Load start values as actual values > All values" button.

Or:

1. Open the data block.
2. Click the "Load start values as actual values > Only setpoints" button.

Reinitializing actual values of multiple blocks

To reinitialize all actual values of multiple blocks in one step, proceed as follows:

1. Select multiple blocks or the entire block folder in the project tree.
2. Select the "Load start values as actual values" command in the shortcut menu.

Result

The actual values in the online program are reinitialized with the start values.

The maximum number of tags that can be initialized is dependent on the CPU. If there are too many tags, an alarm informs you about this. In this case, you can insert the tags in a watch table and initialize them using the "Modify" function in the watch table.

See also

Overview of the online and diagnostic functions in data blocks (Page 4822)

11.9.11.6 Creating a snapshot of the actual values

You can save the actual values of the tags of one or more data blocks as snapshot. Actual values are the values which the tags have at the current time during program execution in the CPU work memory.

A snapshot cannot be created for data blocks that are stored only in the load memory and are not integrated in the program.

You have the following options for creating a snapshot:

- Create a snapshot of an open data block
- Create a snapshot of several selected data blocks

A snapshot is also created automatically when you download a block or a program from the device.

CAUTION

Creating the snapshot

The values in the snapshot may originate in several cycles.

Requirement

- An online connection to the CPU is available.
- The data blocks for which you want to create a snapshot are identical offline and online.

Procedure

To create a snapshot of an open data block, follow these steps:

1. Open the data block.
2. Click "Snapshot".

To create a snapshot of several selected data blocks, follow these steps:

1. Select the blocks in the project tree.
   You can select the blocks individually or select devices, groups or folders in the project tree which include the data blocks.
2. In the shortcut menu, select "Snapshot of the current values" or select the menu command "Online > Snapshot of the current values".

Result

- The latest monitored values will be applied in the "Snapshot" column.
- An alarm is shown in the Inspector window after the operation is complete.
- The time stamp of the snapshot is displayed above the declaration table.

Note

If you subsequently change the structure of the data block, the display of the current values gets lost. The "Snapshot" column will then be empty.

See also

Overview of the online and diagnostic functions in data blocks (Page 4822)

11.9.11.7 Loading snapshots as actual values

You can load the snapshots from the offline program to the CPU as actual values. The values from the snapshot are loaded directly to the CPU work memory. The tags in the online block are reinitialized, and the CPU uses the new values as actual values in the online program.

You can quickly bring the controlled process into a defined state this way.

You cannot reinitialize the following types of data blocks with snapshots:

- Data blocks that are part of a reference project
- Fail-safe data blocks
- Data blocks generated by the CPU
- Data blocks with the attribute "Only store in load memory"
- Data blocks that only exist online

⚠ DANGER

Danger when changing tag values

Changing the tag values while the plant is operating can cause serious damage to property or injury to persons if there are functional disturbances or program errors.

- Make sure that the plant is in a safe state before you overwrite the actual values.
- Make sure that the program does not read or write the affected data during transmission.
- You may want to use the "Modify tag" function in the watch table or in the DB editor as an alternative.

Dependencies on the CPU mode

You can execute this function in "RUN" mode as well as "STOP" mode. The table below shows the reactions of the CPU in the different modes:

Action	System reaction	Consequences for the online program
Overwrite actual values in "RUN" mode	The values of all DB tags are overwritten in the current program. No distinction is made between retentive and non-retentive values.	Changing the actual values can result in inconsistencies between the program and the actual process. If the volume of data to be transmitted is too large, the values may be transmitted in several cycles. If the program accesses tags before all values are completely transmitted, there is a risk that inconsistent value combinations may be created and processed. Copying the values of elementary data types may also take several cycles. These values are potentially invalid until they have been completely transmitted. Dangerous states may occur if the program accesses these values before they have been completely transmitted.
Overwrite actual values in "STOP" mode	Only the actual values of the retentive tags are overwritten by the snapshot. Non-retentive tags are initialized with their start values during the transition from "STOP" to "RUN". The values from the snapshot are not taken into consideration.	Because only the retentive data from the snapshot is transmitted, there is a risk that inconsistent value combinations may be created and processed.

Requirement

- You are using an S7-1200 V4.1 or higher or an S7-1500 V1.7 or higher.
- An online connection to the CPU is available.
- The data blocks whose actual values you want to reinitialize are identical offline and online.
- A snapshot of the data blocks has been created.

Reinitializing actual values of an open block

To reinitialize all actual values of a block with snapshots, follow these steps:

1. Open the data block.
2. Click "Load snapshots as actual values".

Reinitializing actual values of multiple blocks

To reinitialize the actual values of multiple blocks in one step, follow these steps:

1. Select multiple blocks or the entire block folder in the project tree.
2. Select the "Load snapshots as actual values" command in the shortcut menu.

Result

The actual values in the online program are reinitialized with the values from the snapshot.

See also

Overview of the online and diagnostic functions in data blocks (Page 4822)

11.9.11.8 Copying the snapshot to the start values

You can copy the snapshot to the start values in the offline program. The program runs with the new start values at the next transition from "STOP" to "RUN".

You can copy all start values, the start values of the retentive tags or only the start values of the tags identified as "setpoint".

Note that the values from the snapshot are always copied. There is no check to determine whether all values originate from the same cycle.

Write-protected start values are not overwritten.

You have the following basic options for applying the values:

- Applying the values of an open data block
  You can apply all values or only the values of the tags marked as "setpoint" as start values in an open data block.
- Applying the values of multiple blocks in the project tree
  You can either apply all values, all setpoints or all retentive values as start values in the project tree.

Requirement

A snapshot has been created for the data block.

Procedure

To apply the snapshots as start values in a data block, follow these steps:

1. Open a data block.
2. In the toolbar, click "Copy snapshots to start values > All values".

Or:

1. Open a data block.
2. In the toolbar, click "Copy snapshots to start values > Only setpoints".

To apply the monitored values of multiple data blocks in the project tree, follow these steps:

1. Select the blocks in the project tree.
2. Then select one of the following commands in the shortcut menu:
   - "Copy snapshots to start values > All values"
   - "Copy snapshots to start values > Only setpoints"
   - "Copy snapshots to start values > Only retain values"

Result

The values from the "Snapshot" column are applied to the "Start value" column. The new start values are stored in the offline program.

Note

Applying values of individual tags

You can also apply the values of individual tags from the "Snapshot" column to the "Start values" column. Use the "Copy" and "Paste" commands from the shortcut menu to copy the values and insert them in the "Start value" column. Note that only the values that are currently located in the visible area of the table are copied.

See also

Overview of the online and diagnostic functions in data blocks (Page 4822)

11.9.11.9 Using setpoints during commissioning

Basic information on adjusting setpoints during commissioning

Introduction

During commissioning of a plant, data values have to be frequently adjusted in order to optimally adapt the program to the general operating conditions on site. To this end, you can use the online and diagnostic functions of the declaration table for data blocks.

To use the functions, first define specific tags as "setpoints" in the program. Setpoints are the values that will probably have to be fine-tuned during commissioning.

The following table provides an overview of the functions for tuning values during commissioning. Detailed descriptions of the individual functions can be found in the following chapters.

In addition, there are also the general functions for monitoring and controlling the data block:

Overview of the online and diagnostic functions in data blocks (Page 4822)

Button	Function	Description
	Initialize setpoints in "RUN" mode (Page 4833)	This function enables you to change the values of individual tags online to quickly determine the optimum tag values.
 	Apply values from the online program as start values to the offline program (Page 4834)	When you have determined the optimum tag values, you can apply these as start values in the offline program. This allows you to ensure that the program starts with the optimized values the next time it is loaded.

See also

Marking data as values that can be set (Page 4832)

Marking data as values that can be set

You can mark specific tags in the program as "Setpoints". Setpoints are the values that will probably have to be fine-tuned during commissioning.

Rules

You can mark tags as "setpoints" in the following block types:

- In function blocks (FB), but only in the "Static" section
- in global data blocks (DB)
- in PLC data types (UDT)
  In the case of PLC data types, however, the setting is only effective if the UDT is used in the "Static" section of a function block or data block.

It is not possible to define setpoints in the following block types:

- In data blocks based on a PLC data type and in instance data blocks. These inherit the setting from the higher-level FB or UDT.
- You cannot mark tags as "setpoints" in ARRAY data blocks.
- You also cannot mark tags as "setpoints" at the call point of a multi-instance. You have to make the setting in the interface of the function block that is called as multiple instance.
- You cannot change the "setpoint" marking in know-how-protected blocks. To do so, you must first remove the know-how protection.

Requirement

A function block, a global data block or a PLC data type (UDT) is open.

Procedure

To mark a tag as "setpoint", follow these steps:

1. Select a tag from the "Static" section.
2. Select the check box in the "Setpoint" column.
   - You cannot define the higher-level element of a structure or a PLC data type as "setpoint". You have to make the setting for the lower-level elements individually.
   - In the case of ARRAYs, you can only mark the higher-level element as "setpoint". The lower-level elements inherit the setting.
   - For ARRAYs of STRUCT, you can only mark the elements below the first structure as setpoints. The elements of other structures inherit the setting.

Result

The tags are marked as setpoints. During commissioning, these tags can be initialized online. You do not need to set the CPU to "STOP" mode; it can remain in "RUN". In addition, the current tag values can be transferred as start values to the offline program and saved there.

See also

Overview of the online and diagnostic functions in data blocks (Page 4822)

Online and diagnostic functions for data blocks (Page 4820)

Defining start values (Page 4807)

Initializing setpoint values in the online program

Basics on initializing setpoints

You can initialize all tags marked as "setpoints" with new values in the online program. At the same time, the start values will be loaded from the offline program to the online program. The CPU remains in "RUN" mode. All tags that are marked as setpoints are initialized once at the next cycle control point. This applies both to retentive and non-retentive tags. The program execution is then continued with the new tag values.

The maximum number of initializable tags is CPU-dependent: For S7-300/400 CPUs, you can initialize up to 35 setpoints. For S7-1200/1500 CPUs, the maximum amount is 200 setpoints. The maximum amount may be less, however, depending on the data types used.

If too many setpoints are selected, an alarm informs you of this. In this case, you can insert the tags in a watch table and initialize them using the "Modify" function in the watch table.

DANGER

Danger when changing tag values

Changing the tag values while the plant is operating can cause serious damage to property or injury to persons if there are functional disturbances or program errors.

Make sure that no dangerous situations can arise before you reinitialize the setpoints.

Requirement

- An online connection to the CPU exists.
- The structure of the data block is identical offline and online.
- One or more tags are marked as "setpoints".

Procedure

To initialize all setpoints of the data block, follow these steps:

1. Open a global data block or an instance data block.
2. Enter the required values in the "Start value" column. The start values must correspond to the indicated data type.
3. Click the "Load start values as actual values > Only setpoints" button.

Result

The setpoints in the online program are initialized with the start values from the offline program at the next cycle control point.

See also

Overview of the online and diagnostic functions in data blocks (Page 4822)

Applying values from the online program as start values

In order to apply tag values from the online program to the offline program as start values, first create a snapshot of the tag values from the online program. You can then apply them to the offline program. Note that the values from the snapshot are always copied. There is no check to determine whether all values originate from the same cycle.

Write-protected start values are not overwritten.

You have the following basic options for applying the values:

- Applying the setpoints of an open data block
- Applying the setpoints of multiple blocks in the project tree

Requirement

- An online connection to the CPU is available.
- As least one data block has been loaded to the CPU.

Procedure

To apply all setpoints as start values in a data block, follow these steps:

1. Open the data block.
2. Start monitoring by clicking the "Monitor all" button.
   The "Monitor value" column is displayed in the table. This shows the current data values.
3. In the toolbar, click "Snapshot".
   The actual values are applied in the "Snapshot" column.
4. In the toolbar, click "Copy snapshots to start values > Only setpoints".

The values from the "Snapshot" column are applied to the "Start values" column.

To apply the monitored values of multiple data blocks in the project tree, follow these steps:

1. Select the blocks in the project tree.
2. Select the "Snapshot" command in the shortcut menu.
   The actual values of all selected blocks are applied in the "Snapshot" column.
   An alarm is shown in the Inspector window after the operation is complete.
3. Next, select the command "Copy snapshot values as start values > Only setpoints" in the shortcut menu.

The values from the "Snapshot" column are applied to the "Start value" column.

Result

The new start values are stored in the offline program.

Note

Applying values of individual tags

You can also apply the values of individual tags that were not marked as setpoints beforehand from the "Snapshot" column to the "Start values" column. Use the "Copy" and "Paste" commands from the shortcut menu to copy the values and insert them in the "Start value" column. Note that only the values that are currently located in the visible area of the table are copied.

See also

Basic information on start values (Page 4807)

Defining start values (Page 4808)

Overview of the online and diagnostic functions in data blocks (Page 4822)
Online and diagnostic functions for data blocks (Page 4820)

11.10 Declaring PLC tags

11.10.1 Overview of PLC tag tables

Introduction

PLC tag tables contain the definitions of the PLC tags and symbolic constants that are valid throughout the CPU. A PLC tag table is created automatically for each CPU used in the project. You can create additional tag tables and use these to sort and group tags and constants.

In the project tree there is a "PLC tags" folder for each CPU of the project. The following tables are included:

- "All tags" table
- Standard tag table
- Optional: Other user-defined tag tables

All tags

The "All tags" table gives an overview of all PLC tags, user constants and system constants of the CPU. This table cannot be deleted or moved.

Standard tag table

There is one standard tag table for each CPU of the project. It cannot be deleted, renamed or moved. The default tag table contains PLC tags, user constants and system constants. You can declare all PLC tags in the default tag table, or create additional user-defined tag tables as you want.

User-defined tag tables

You can create multiple user-defined tag tables for each CPU to group tags according to your requirements. You can rename, gather into groups, or delete user-defined tag tables. User-defined tag tables can contain PLC tags and user constants.

See also

Structure of the PLC tag tables (Page 4838)

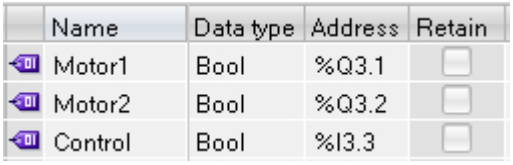
11.10.2 Structure of the PLC tag tables

Introduction

Each PLC tag table contains a tab for tags and a tab for user constants. The default tag table and the "All tags" table also have a "System constants" tab.

Structure of the "PLC tags" tab

In the "Tags" tab you declare the global PLC tags that you require in the program. The following figure shows the tab structure. The number of columns shown may vary.



	Name	Data type	Address	Retain
	Motor1	Bool	%Q3.1	
	Motor2	Bool	%Q3.2	
	Control	Bool	%I3.3	

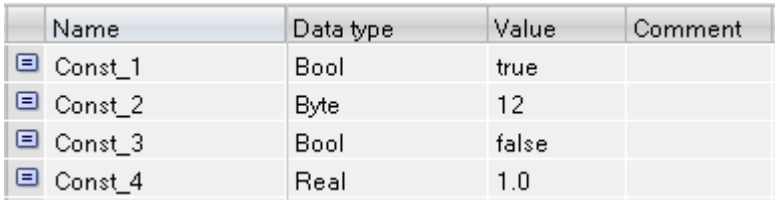
The following table shows the meaning of the individual columns. The number of columns shown may vary. You can show or hide the columns as required.

Column	Explanation
	Symbol you can click on to drag-and-drop a tag to a program for use as an operand.
Name	Unique name for the constants throughout the CPU.
Data type	Data type of the tags.
Address	Tag address.
Retain	Marks the tag as retentive. The values of retentive tags are retained even after the power supply is switched off.
Visible in HMI engineering	Shows whether the tag is visible by default in the operand selection of HMI.
Accessible from HMI/OPC UA	Indicates whether HMI/OPC UA can access this tag during runtime.
Writable from HMI/OPC UA	Indicates whether the tag from HMI/OPC UA can be written during runtime.
Supervision	Indicates whether Supervision has been created for process diagnostics for the tag.
Monitor value	Current data value in the CPU. This column only appears if an online connection is established and you select the "Monitor all" button.
Tag table	Shows which tag table includes the tag declaration. This column is only available in the "All tags" table.
Comment	Comment to document the tags.

Structure of the "User constants" and "System constants" tabs

In the "User constants" you define symbolic constants that are valid throughout the CPU. The constants required by the system are shown in the "Systems constants" tab. System constants can be hardware IDs, for example, which can be used for identification of modules.

The following figure shows the structure of both tabs. The number of columns shown may vary.



	Name	Data type	Value	Comment
	Const_1	Bool	true	
	Const_2	Byte	12	
	Const_3	Bool	false	
	Const_4	Real	1.0	

The following table shows the meaning of the individual columns. You can show or hide the columns as required.

Column	Explanation
	Symbol you can click to move a tag into a network via a drag-and-drop operation for use as an operand.
Name	Unique name for the constants throughout the CPU.
Data type	Data type of the constants
Value	Value of the constants
Tag table	Shows which tag table includes the constant declaration. This column is only available in the "All tags" table.
Comment	Comments to document the tags.

See also

Hardware data types (Page 2328)

Overview of PLC tag tables (Page 4837)

Show and hide table columns (Page 4861)

Editing tables (Page 278)

11.10.3 Rules for PLC tags

11.10.3.1 Valid names of PLC tags

Permissible characters

The following rules apply to the use of names for PLC tags:

- Letters, numbers, special characters are permitted.
- The use of reserved keywords is not recommended.
- Quotation marks are not permitted as a component of the tag name.

Unique tag names

The names of the PLC tags must be unique throughout the CPU, even if the tags are located in different tag tables of a CPU. A name that is already used for a block, another PLC tag or a constant within the CPU, cannot be used for a new PLC tag. The uniqueness check does not differentiate between use of small and capital letters.

If you enter an already assigned name another time, a sequential number is automatically appended to the second name entered. For example, if you enter the name "Motor" a second time, the second entry is changed to "Motor(1)".

Unique table names

The names of the PLC tag tables must also be unique throughout the CPU. A unique name is automatically suggested when user-defined PLC tag tables are being created.

Note

You can find more information on permissible characters in tag names in the Siemens Industry Online Support in the following entries:

When should identifiers and operands be used in "quotation marks" in STEP 7 (TIA Portal)?

https://support.industry.siemens.com/cs/ww/en/view/109477857 (https://support.industry.siemens.com/cs/ww/en/view/109477857)

See also

Permissible addresses and data types of PLC tags (Page 4840)

Entering a PLC tag declaration (Page 4846)

11.10.3.2 Permissible addresses and data types of PLC tags

The addresses of PLC tags are made up of the particulars of the operand area and the address within this area.

The addresses must be unique throughout the CPU. If you enter an address that is already assigned to another tag, the address will be highlighted at both places in yellow and an error message will be issued.

Operand areas

The following table shows the possible operand areas. The available data types depend on the CPU you use:

Operand area		Explanation	Data type	Format	Address area:		
International mnemonics	German mnemonics				S7-1200	S7-300/400	S7-1500
I	E	Input bit	BOOL	I x.y E x.y	0.0..1023.7	0.0..65535.7	0.0..32767.7
I	E	Input (64-bit)	LWORD, LINT, ULINT, LTIME, LTOD, LDT, LREAL, PLC data type	I x.0 E x.0	-	-	0.0..32760.0
IB	EB	Input byte	BYTE, CHAR, SINT, USINT, PLC data type	IB x EB y	0..1023	0..65535	0..32767
IW	EW	Input word	WORD, INT, UINT, DATE, WCHAR, S5TIME, PLC data type	IW x EW y	0..1022	0..65534	0..32766
ID	ED	Input double word	DWORD, DINT, UDINT, REAL, TIME, TOD, PLC data type	ID x ED y	0..1020	0..65532	0..32764
Q	A	Output bit	BOOL	Q x.y A x.y	0.0..1023.7	0.0..65535.7	0.0..32767.7
Q	A	Output (64-bit)	LWORD, LINT, ULINT, LTIME, LTOD, LDT, LREAL, PLC data type	Q x.0 A x.0	-	-	0.0..32760.0
QB	AB	Output byte	BYTE, CHAR, SINT, USINT, PLC data type	QB x AB y	0..1023	0..65535	0..32767

Operand area		Explanation	Data type	Format	Address area:		
Interna-tional mnemon-ics	German mnemon-ics				S7-1200	S7-300/400	S7-1500
QW	AW	Output word	WORD, INT, UINT, DATE, WCHAR, S5TIME, PLC data type	QW x AW y	0..1022	0..65534	0..32766
QD	AD	Output dou-ble word	DWORD, DINT, UDINT, RE-AL, TIME, TOD, PLC da-ta type	QD x AD y	0..1020	0..65532	0..32764
M	M	Memory bit	BOOL	M x.y	0.0..8191.7	0.0..65535.7	0.0..16383.7
M	M	Bit memory (64-bit)	LREAL	M x.0	0.0..8184.0	-	0.0..16376.0
M	M	Bit memory (64-bit)	LWORD, LINT, ULINT, LTIME, LTOD, LDT	M x.0	-	-	0.0..16376.0
MB	MB	Memory byte	BYTE, CHAR, SINT, USINT	MB x	0..8191	0..65535	0..16383
MW	MW	Memory word	WORD, INT, UINT, DATE, WCHAR, S5TIME	MW x	0..8190	0..65534	0..16382
MD	MD	Memory dou-ble word	DWORD, DINT, UDINT, RE-AL, TIME, TOD	MD x	0..8188	0..65532	0..16380
T	T	Time function (for S7-300/400 only)	Timer	T n	-	0..65535	0..2047
C	Z	Counter func-tion (for S7-300/400 only)	Counter	Z n C n	-	0..65535	0..2047

Addresses

The following table shows the possible addresses of tags:

Data type	Address	Example
BOOL	Tags with BOOL data type are addressed with a byte number and a bit number. The numbering of the bytes begins at 0 for each operand range. The numbering of the bits is from 0 to 7.	A 1.0
BYTE, CHAR, SINT, USINT	Tags with BYTE, CHAR, SINT, and USINT data type are addressed with a byte number.	MB 1
WORD, INT, UINT, DATE, WCHAR, S5TIME	Tags with WORD, INT, UINT, DATE, S5TIME data type consist of two bytes. They are addressed with the number of the lowest byte.	IW 1
DWORD, DINT, UDINT, REAL, TIME, TOD	Tags with DWORD, DINT, UDINT, REAL, TIME, TOD data type consist of four bytes. They are addressed with the number of the lowest byte.	AD 1
LWORD, LINT, ULINT, LTIME, LTOD, LDT, LREAL	Tags of data type LWORD, LINT, ULINT, LTIME, LTOD, LDT, and LREAL consist of eight bytes. They are addressed with it number 0 and the number of the lower byte.	I 1.0

Mnemonics used

The addresses that you enter in the PLC tag table are automatically adapted to the set mnemonics.

See also

Setting the mnemonics (Page 4695)

Valid names of PLC tags (Page 4840)

Overview of the valid data types (Page 2239)

Entering a PLC tag declaration (Page 4846)

11.10.4 Creating and managing PLC tag tables

11.10.4.1 Creating a PLC tag table

You can created multiple user-defined PLC tag tables in a CPU. Each tag table must have have a unique name throughout the CPU.

Requirement

The project view is open.

Procedure

To created a new PLC tag table, follow these steps:

1. Open the "PLC tags" folder under the CPU in the project tree.
2. Double-click the "Add new tag table" entry.
   A new PLC tag table with the default name "TagTable_x" is created.
3. Select the PLC tag table in the project tree.
4. Select the "Rename" command in the shortcut menu.
5. Type in a name that is unique throughout the CPU.

Result

A new PLC tag table is created. You can now declare tag and constants in this table.

See also

Overview of PLC tag tables (Page 4837)

Structure of the PLC tag tables (Page 4838)

Declaring PLC tags (Page 4846)

Declaring global constants (Page 4852)

Exporting and importing PLC tags (Page 4862)

11.10.4.2 Grouping PLC tag tables

You can gather the user-defined tag tables of the CPU into groups. You cannot, however, move the standard tag table and the "All tags" table into a group.

Requirement

Multiple user-defined tag tables are contained in the "PLC tags" folder of the CPU.

Procedure

To gather multiple PLC tag tables into a group, follow these steps:

1. Select the "PLC tags" folder under the CPU in the project tree.
2. Select the "Insert > Group" menu command.
   A new group with the standard name "Group_x" is inserted.
3. Select the newly inserted group in the project tree.
4. Select the "Rename" command in the shortcut menu.

5. Assign the new group a unique name throughout the CPU.
6. Drag to the new group the tables you want to group together.

Result

The tag tables are gathered in the new group.

See also

Overview of PLC tag tables (Page 4837)

Structure of the PLC tag tables (Page 4838)

Declaring PLC tags (Page 4846)

Declaring global constants (Page 4852)

11.10.4.3 Opening the PLC tag table

Procedure

To open the PLC tag table in a CPU, proceed as follows:

1. Open the "PLC tags" folder under the CPU in the project tree.
2. Double-click the PLC tag table in the folder.
3. Select the desired tab in the upper corner.

Result

The PLC tag table associated with the CPU opens. You can declare the required tags and constants.

See also

Overview of PLC tag tables (Page 4837)

Structure of the PLC tag tables (Page 4838)

Declaring PLC tags (Page 4846)

Declaring global constants (Page 4852)

11.10.5 Declaring PLC tags

11.10.5.1 Entering a PLC tag declaration

Declaring tags in the PLC tag table

Requirement

The "Tags" tab of the PLC tag table is open.

Procedure

To define PLC tags, follow these steps:

1. Enter a tag name in the "Name" column.
2. Enter the required data type in the "Data type" column. You will be supported by autocompletion during input.
   An address corresponding to the data type is automatically appended.
3. Optional: Click on the arrow key in the "Address" column and enter an operand identifier, an operand type, an address and a bit number in the dialog which then opens.
4. Optional: Enter a comment in the "Comments" column.
5. Repeat steps 1 to 4 for all the tags you require.
   See also: Permissible addresses and data types of PLC tags (Page 4840)

Syntax check

A syntax check is performed automatically after each entry, and any errors found are displayed in red. You do not have to correct these errors immediately - you can continue editing and make any corrections later. As long as the tag declaration contains syntax errors and the tag is used in the program, you will not be able to compile the program.

See also

Exporting and importing PLC tags (Page 4862)

Using autocompletion (Page 4706)

Valid names of PLC tags (Page 4840)

Declaring PLC tags in the program editor (Page 4847)

Editing PLC tag tables (Page 4859)

Structure of the PLC tag tables (Page 4838)

Editing tables (Page 278)

Declaring PLC tags in the program editor

Requirement

- The program editor is open.

Procedure

To declare operands as global PLC tags, follow these steps:

1. Insert an instruction in your program.
   The "<???>", "<??.?>" or "..." strings represent operand placeholders.
2. Replace an operand placeholder with the name of the PLC tag to be created.
3. Select the tag name.
   If you want to declare multiple PLC tags, select the names of all the tags to be declared.
4. Select the "Define tag" command in the shortcut menu.
   The "Define tag" dialog box opens. This dialog displays a declaration table in which the name of the tag is already entered.
5. Click the arrow key in the "Section" column and select one of the following entries:
   - Global Memory
   - Global Input
   - Global Output
   - Global Timer
   - Global Counter
6. In the other columns, enter the address, data type, and comments.
   See also: Permissible addresses and data types of PLC tags (Page 4840)
7. If the CPU contains multiple PLC tag tables, you can use an entry in the "PLC tag table" column to indicate in which table the tag is to be inserted. If you make no entry in the column, the new tag will be inserted in the default tag table.
8. Click the "Define" button to complete your entry.

Result

The tag declaration is written to the PLC tag table and is valid for all blocks in the CPU.

See also

Valid names of PLC tags (Page 4840)

Editing tables (Page 278)

Declaring tags in the PLC tag table (Page 4846)

Using autocompletion (Page 4706)

11.10.5.2 Setting the retentivity of PLC tags

Retentive behavior of PLC tags

Retentive PLC tags

Each CPU has a memory area whose content remains available even after the supply voltage has been switched off. This area is referred to as retentive memory area.

To avoid data loss during power failure, you can save specific PLC tags to this memory area. You specify the retain setting of PLC tags in the PLC tag table.

Depending on the CPU family, the retentive memory area can accommodate various type of PLC tags. The following table provides an overview of the options of the various CPUs:

CPU type	Retentive bit memories	Retentive SIMATIC timers	Retentive SIMATIC counters
S7-300/400 series	✓	-	-
S7-1200 series	✓	-	-
S7-1500 series	✓	✓	✓

See also

Setting the retentive behavior of PLC tags (Page 4848)

Setting the retentive behavior of PLC tags

Introduction

In the PLC tag table you can specify the width of the retentive memory area for PLC tags. All tags with addresses in this memory area are then designated as retentive. You can recognize the retentivity setting of a tag by the check mark set in the "Retain" column of the PLC tag table.

Requirement

The "PLC tags" tab of the PLC tag table is open.

Procedure

To define the width of the retentive memory area for PLC tags, follow these steps:

1. On the toolbar, click the "Retain" button.
   The "Retain memory" dialog will open.
2. Specify the width of the retentive memory area by entering the number of retentive bytes, timers or counters in the input field.
3. Click the "OK" button.

Result

The width if the retentive memory area is defined. In the "Retain" column of the tag table a check mark is automatically set for all tags that are located within the retentive memory area.

See also

Retentive behavior of PLC tags (Page 4848)

Editing tables (Page 278)

11.10.6 Grouping PLC tags for inputs and outputs in structures

11.10.6.1 Other useful information regarding structured PLC tags

Use of structured PLC tags (S7-1200 V4 and higher/S7-1500)

To make your program easier to view, you can group several input or output addresses in a higher-level PLC tag. The higher-level PLC tag represents a structure that contains several logically related inputs or outputs. When the block is called, you transfer the higher-level tag and thus need only an input or output parameter for all associated inputs or outputs.

Principle of operation

To create a structured PLC tag, you initially define a PLC data type (UDT). In it you declare the necessary data elements and specify their names and data types.

Then, you switch to the PLC tag table and specify the higher-level PLC tag there. As a data type for the tag, you select your PLC data types. The system now reserves a certain number of input or output addresses starting from the start address of the higher-level tag. The number of reserved addresses depends on the length of your PLC data type.

If you call a block that requires the reserved inputs or outputs for program execution, you transfer the higher-level tag as a block parameter.

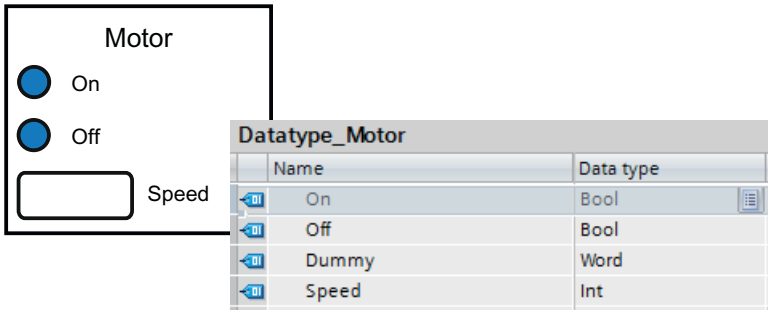
You can address the individual PLC tags like structure elements in the program code.

A detailed description of the individual handling steps can be found in the following chapters.

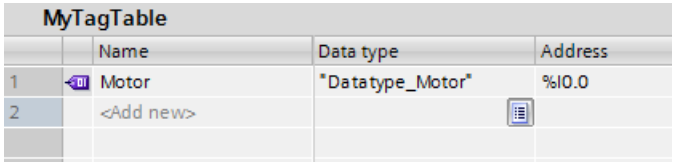
Application example

You can use structured PLC tags in order to group the inputs or outputs of a function module. The following figure shows the schematic representation of a motor: A component was created in the "Datatype_Motor" PLC data type for each of the three inputs.

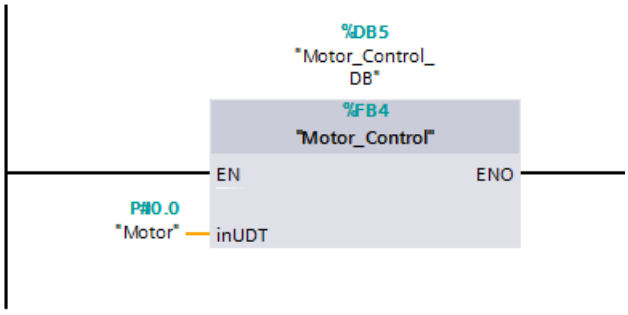
The memory areas of the declared tags must not overlap. In the example, you see that the "Speed" component has the data type "Integer" and therefore must start at a word address. For this reason, the first input word has been filled with the "Dummy" fill tag. This means that "Speed" is located on the second input word.



The following figure shows the higher-level "Motor" PLC tag that is based on the "Data type_Motor" data type. With the declaration of "Motor", the addresses IW0 and IW1 are reserved on the input module.



The following figure shows the transfer of the "Motor" PLC tag as an input parameter of the "Motor_Control" block.



You can address the individual components of the tag in the "Motor_Control" block.

Addressing	Description
"Motor"	Addressing the higher-level PLC tag.
"Motor".On	Addressing a component of a structured PLC tag.
"Motor".On:P	Addressing an I/O input or output (PI or PQ).

Rules for use of structured PLC tags

Note the following rules when creating and using structured PLC tags.

- Structured PLC tags can be used in the "Inputs" and "Outputs" operand areas.
- Structured tags are not permitted in the bit memory address area.
- Structured PLC tags cannot be addressed from HMI.

Observe the following rules when creating the PLC data type that is to serve as the basis for a PLC tag:

- The memory areas of the individual elements must not overlap.
  See also: Permissible addresses and data types of PLC tags (Page 4840)
- Do not group inputs and outputs in one PLC data type but create different PLC data types for inputs and outputs.
- Do not group inputs or outputs from different modules in a PLC data type because it is not guaranteed that the process images of the modules are updated synchronously.
- In the lower-level PLC data types, all data types are permitted except for "STRING" and "WSTRING".

See also

Creating structured PLC tags (Page 4851)

11.10.6.2 Creating structured PLC tags

Rules

Note the following rules when creating structured PLC tags:

- Use separate PLC data types for the "Inputs" and "Outputs" operand areas.
- Structured tags are not permitted in the bit memory address area.
- Do not group inputs or outputs from different modules in a PLC data type because it is not guaranteed that the process images of the modules are updated synchronously.

Procedure

To create a structured PLC tag, follow these steps:

1. Double-click the "Add new data type" command in the "PLC data types" folder in the project tree.
   A new declaration table for creating a PLC data type will be created and opened.
2. Declare all the necessary components in the PLC type. All data types except for "STRING" and "WSTRING" are permitted.

3. Select the PLC data type in the project tree and select the "Compile > Software (only changes)" command from the shortcut menu.
   The PLC data type is compiled and can then be used in the PLC tag table.
   Even when you make changes to existing PLC data types, you must recompile the program. This updates all locations of use of the PLC data type.
4. Open a PLC tag table within the same CPU.
5. Declare a new tag or select an existing tag.
6. In the "Data type" column, select the PLC data type and assign it to the PLC tag.
   The PLC tag receives the structure of the PLC data type. A suitable address is assigned automatically. Structured PLC tags always start at word addresses.
   The highest structure element is displayed without its subelements in the table.

Note

Assignment rules and default values

- For the declaration of the PLC data type, note that the memory areas of the individual tags must not overlap. For example, tags of data type "Integer" must start at a word limit. If necessary, insert "fill tags" to prevent overlaps.
  See also: Permissible addresses and data types of PLC tags (Page 4840)
- It is not possible to assign default values for the individual components. Values that you enter in the "Default value" column are not evaluated. Tags of data types "DT" and "DTL" may therefore contain invalid values.

See also

Other useful information regarding structured PLC tags (Page 4849)

Declaring PLC tags (Page 4846)

11.10.7 Declaring global constants

11.10.7.1 Rules for global user constants

Permitted characters

Names of global constants may consist of the following characters:

- Letters, numbers, special characters are permitted.
- Quotation marks are not permitted.

Unique constant names

The names of global constants must be unique throughout the CPU, even if the constants are located in different tag tables of a CPU. A name that is already used for a block, a PLC tag or another constant within the CPU, cannot be used for new constant. The uniqueness check does not differentiate between use of small and capital letters.

If you enter an already assigned name another time, a sequential number is automatically appended to the second name entered. For example, if you enter the name "Motor" a second time, the second entry is changed to "Motor(1)".

Permitted data types

For constants, all data types supported by the CPU are permitted, with the exception of structured data types.

Permitted values

You can select any value from the value range of the specified data type as constant value. For information on the value ranges, refer to the "Data types" chapter.

See also: AUTOHOTSPOT

See also

Declaring global constants (Page 4854)

11.10.7.2 Rules for global system constants

Definition

System constants are global constants, unique throughout the CPU, that are required and automatically created by the system. System constants can, for example, be used to address and identify hardware objects.

Rules

System constants are assigned automatically when components are inserted in the device or network view, and entered in the default tag table ("System constants" tab). A system constant is created for each module, but also for each submodule. An integrated counter, for example, therefore receives a system constant as well. System constants consist of a symbolic name and a numeric HW identifier and cannot be changed.

Names of system constants

The names of the system constants are structured hierarchically. They consist of a maximum of four hierarchical levels, each of which is separated by a tilde "~". Based on the name, you can therefore recognize the "path" to the relevant hardware module.

Example

A system constant with the name "`Local~PROFINET_interface_1~Port_1`" designates Port 1 of the PROFINET interface 1 of the local CPU.

See also

Instructions for address conversion (Page 4073)

11.10.7.3 Declaring global constants

Introduction

You declare constants in the "User constants" tab of a PLC tag table. During declaration you have to to enter a symbolic name, a data type and a fixed value for each constant. You can select any value from the value range of the specified data type as constant value. For information on the value ranges, refer to the "Data types" chapter.

See also: AUTOHOTSPOT

Procedure

To declare constants, follow these steps:

1. Open a PLC tag table.
2. Open the "User constants" tab.
   The constants table opens.
3. Enter a constant name in the "Name" column.
4. Enter the required data type in the "Data type" column. You will be supported by autocompletion during input.
5. Enter a constant value in the "Value" column; this constant value must be valid for the selected data type.
6. If you want, enter comments on the constants in the "Comments" column. The entry of a comment is optional.
7. If you want to declare additional constants, place the cursor in the next row and repeat steps 3 to 6.

Syntax check

A syntax check is performed automatically after each entry, and any errors found are displayed in red. You do not have to correct these errors immediately - you can continue editing and make any corrections later. As long as the tag declaration contains syntax errors and the constant is used in the program, you will not be able to compile the program.

See also

Opening the PLC tag table (Page 4845)

Inserting a table row in the PLC tag table (Page 4859)

Structure of the PLC tag tables (Page 4838)

Rules for global user constants (Page 4852)

Editing tables (Page 278)

Using autocompletion (Page 4706)

11.10.8 Editing properties

11.10.8.1 Editing the properties of PLC tags

Properties of PLC tags

Overview

The following table provides an overview of the properties of PLC tags. The display of properties may vary depending on the CPU type.

Group	Property	Description
General	Name	A unique name within the CPU.
	Data type	Data type of the tags.
	Address	Tag address.
	Retain	Shows whether the tag is in the retentive memory area.
	Comment	Comment on the tag.
History	Date created	Time when the tag was created (cannot be changed).
	Last modified	Time when the tag was last changed (cannot be changed).
Application	Visible in HMI engineering	Shows whether the tag is visible by default in the HMI selection list.
	Accessible from HMI/ OPC UA	Indicates whether HMI/OPC UA can access this tag during runtime.
	Writable from HMI/OPC UA	Indicates whether the tag from HMI/OPC UA can be written during runtime.

See also

Editing the properties of PLC tags (Page 4856)

Editing the properties of PLC tags

Editing properties in a PLC tag table

To edit the properties of one or more tags, follow these steps:

1. In the project tree, double-click the PLC tag table that contains the tags.
   The PLC tag table opens.
2. Change the entries in the columns.

Editing addresses in the program editor

To edit the address of a tag in the program editor, follow these steps:

1. Select the tag name.
2. Select the "Rewire tag" command in the shortcut menu.
   The "Rewire tag" dialog will open. The dialog shows a declaration table.
3. Enter the new address in the "Address" column.
4. Click the "Change" button to confirm the input.

Editing names in the program editor

To edit the name of a tag in the program editor, follow these steps:

1. Select the tag name.
2. Select the "Rename tag" command in the shortcut menu.
   The "Rename tag" dialog opens. The dialog shows a declaration table.
3. Enter the new name in the "Name" column.
4. Click the "Change" button to confirm the input.

Effect in the program

In the case of a change of the tag's name, data type or address, each location of use of the tag is automatically updated in the program.

See also

Properties of PLC tags (Page 4855)

11.10.8.2 Editing properties of global constants

Properties of global constants

Overview

The following table gives an overview of the properties of constants:

Group	Property	Description
General	Name	A unique name within the table
	Data type	Data type of the constants
	Value	Value that you defined for the constants. This value must be compatible with the declared data type. See also: AUTOHOTSPOT
	Comment	Comment on the constants
History	Date created	Time when the constant was created (cannot be changed)
	Last modified	Time when the constant was last changed (cannot be changed)

Editing properties of global constants

Editing properties in a PLC tag table

To edit the properties of one or more constants, follow these steps:

1. In the project tree, double-click the PLC tag table that contains the constants.
   The PLC tag table opens.
2. Open the "User constants" tab.
3. Change the entries in the "Name", "Data type", "Value", or "Comments" column.

Effect in the program

In the case of a change of a constant's name, data type or value, each location of use of the constant is automatically updated in the program.

See also

Editing tables (Page 278)

11.10.9 Monitoring of PLC tags

11.10.9.1 Monitoring of PLC tags

You can monitor the current data values of the tags on the CPU directly in the PLC tag table.

Requirement

An online connection to the CPU exists or is possible.

Procedure

To monitor the data values, proceed as follows:

1. Open a PLC tag table.
2. Start monitoring by clicking the "Monitor all" button.
   - If no online connection to the CPU exists, this is established.
   - Monitoring is started with the trigger setting "Permanent".
   - The additional "Monitor value" column is displayed in the table. This shows the current data values.
   - The symbol for the forcing of tags is displayed if a tag is currently being forced.
3. End the monitoring by clicking the "Monitor all" button again.

Note

Editing PLC tags during the monitoring of tags

If the editing of tags is started and the PLC tag table is edited, for example by adding new tags, the monitoring is re-started after editing is complete.

Note

You also have the option of copying PLC tags to a monitor or force table so that you can monitor, control or force them in the table.

See also

Structure of the PLC tag tables (Page 4838)

Introduction to testing with the watch table (Page 5196)

Introduction for testing with the force table (Page 5224)

Copying entries in the PLC tag table (Page 4859)

11.10.10 Editing PLC tag tables

11.10.10.1 Inserting a table row in the PLC tag table

Procedure

Proceed as follows to insert a row above the selected row:

1. Select the row in front of which you want to insert a new row.
2. Click the "Insert row" button on the toolbar of the table.

Result

A new row is inserted above the selected row.

See also

Editing tables (Page 278)

11.10.10.2 Copying entries in the PLC tag table

You can copy PLC tags within a table or to other tables.

Procedure

To copy a tag, follow these steps:

1. Select the tags you want to copy.
   You can also select several tags by clicking on them one after the other while holding down the <Ctrl> key or by pressing and holding down <Shift> and clicking on the first and last tag.
2. Select "Copy" in the shortcut menu.
3. Position the insertion pointer at the location where you want to insert the tags.
4. Select "Paste" in the shortcut menu.

Or

1. Select the tag.
2. Hold down the left mouse button.
3. At the same time, press <Ctrl>.
4. Drag the tag to the destination.

Result

- The tag is copied to the destination.
- If there is a name conflict, a number is automatically appended to the tag name. For example, "Tag" becomes "Tag(1)".
- All other properties of the tag remain unchanged.

See also

Editing tables (Page 278)

11.10.10.3 Deleting entries in the PLC tag table

Procedure

Follow the steps below to delete elements:

1. Select the row with the element to be deleted. You can also select several rows by clicking on them one after the other while holding down the <Ctrl> key or by pressing and holding down <Shift> and clicking on the first and last row.
2. Select the "Delete" command in the shortcut menu.

See also

Editing tables (Page 278)

11.10.10.4 Sorting rows in the PLC tag tables

You can sort the rows in the tables alphanumerically by name, data type, or address.

Procedure

To sort the table rows, follow these steps:

1. Select the column by which you want to sort.
2. Click the column header.
   The column will be sorted in order of increasing values.
   An up arrow shows the sort sequence.
3. In order to change the sort sequence, click the arrow.
   The column will be sorted in order of decreasing values. A down arrow shows the sort sequence.
4. To restore the original sequence, click a third time on the column header.

See also

Editing tables (Page 278)

11.10.10.5 Automatically filling in cells in the PLC tag table

You can load the contents of one or several table cells into the cells below, automatically filling in the successive cells.

If you automatically fill in cells in the "Name" column, a consecutive number will be appended to each name. For example, "Motor" will become "Motor_1".

If you fill the cells in the column "Address" automatically, the addresses will be increased depending on the indicated data type.

Procedure

To automatically fill in successive cells, follow these steps:

1. Select the cells to be loaded.
2. Click the "Fill" symbol in the bottom right corner of the cell.
   The mouse pointer is transformed into a crosshair.
3. Keep the mouse button pressed and drag the mouse pointer downwards over the cells that you want to fill in automatically.
4. Release the mouse button.
   The cells are filled in automatically. If entries are already present in the cells that are to be automatically updated, a dialog appears in which you can indicate whether you want to overwrite the existing entries or whether you want to insert new rows for the new tags.

See also

Editing tables (Page 278)

11.10.10.6 Show and hide table columns

You can show or hide the columns in a table as needed.

Procedure

To show or hide table columns, follow these steps:

1. Click a column header.
2. Select the "Show/Hide" command in the shortcut menu.
   The selection of available columns is displayed.
3. To show a column, select the column's check box.
4. To hide a column, clear the column's check box.
5. To hide or show several columns, click "More" and activate or deactivate the check box of the corresponding columns in the "Show/Hide" dialog.

See also

Editing tables (Page 278)

11.10.10.7 Exporting and importing PLC tags

Basics for exporting and importing PLC tags

Introduction

You can export PLC tag tables to edit them further with external applications. Similarly, you can import PLC tag tables edited externally to the TIA Portal.

The following formats are supported for import and export:

- XLSX (Excel)
  For the export to Excel you can specify which elements you want to export:
  - Tags and/or constants
  - All elements or just the elements that are actually being used in the program on the current CPU.
- XML
- SDF

Overwriting existing PLC tags and constants during import

Existing entries of the same name will be overwritten during import if they have the same name as the entries that will be imported.

Link to existing objects

References to PLC tags or constants that already exist in the project are updated automatically during import. The update is executed based on the name of the PLC tags and constants.

See also

Format of the export file (*.xlsx) (Page 4862)

Exporting PLC tags (Page 4864)

Importing PLC tags (Page 4865)

Format of the export file (*.xlsx)

Introduction

During the export of PLC tag tables, a standardized XSLX format will be generated that you can edit with external table editors.

This format is also expected during the import of tables.

Format of the export file

The sheet is always named "PLCTags". This sheet can contain the displayed columns. The sorting order of columns can vary. The sheet does not necessarily have to include all columns. During import, the following values will be identified by a <no value> entry.

The names of the column headers are also clearly defined and are always expected in English.

The following table specifies the contents expected for the individual columns:

Element	Description
Name	Name of the tags
Path	Group and name of the PLC tag table
Data Type	The notation of the data type corresponds to the notation used in the PLC tag table.
Logical Address	The address can be specified with German or international mnemonics.
Comment	Free-form comments
Hmi Visible	The value TRUE or FALSE is expected.
Hmi Accessible	The value TRUE or FALSE is expected.
Hmi Writeable	The value TRUE or FALSE is expected.

See also

Basics for exporting and importing PLC tags (Page 4862)

Exporting PLC tags (Page 4864)

Importing PLC tags (Page 4865)

Format of the export file (*.xml)

Introduction

During the export of PLC tag tables, a standardized xml format will be generated. This format is also expected during the import of xml files.

Example of an xml export file

The following example shows a PLC tag table which was exported to the *.xml format:

```
<?xml version='1.0' encoding='utf-8'?>
<Tagtable name='Default tag table'>
  <!-- exported by 'Totally Integrated Automation Portal' version 14.0.1.0   © 2004-2016 Siemens A
  <Tag type='Bool' hmiVisible='True' hmiWriteable='True' hmiAccessible='True' retain='False' remark
  <Constant type='Bool' remark='comment' value='TRUE'>xxxx</Constant>
</Tagtable>
```

Format of the export file (*.sdf)

Introduction

During the export of PLC tag tables, a standardized sdf format will be generated. This format is also expected during the import of sdf files.

Example of an sdf export file

The following example shows a PLC tag table which was exported to the *.sdf format:

```
"Tag_1","%I0.0","Bool","True","True","False","Comment","","True"
"Tag_2","%I0.1","Bool","True","True","False","","","True"
"Const_1","CONSTANT","Bool","","","","Comment1","TRUE",""
"Const_2","CONSTANT","Bool","","","","","TRUE",""
```

Exporting PLC tags

Requirement

A PLC tag table is open.

Procedure

To export PLC tags and constants, follow these steps:

1. In the PLC tag table, click the "Export" button.
   The "Export" dialog box opens.
2. Click the selection button under "Path of export file".
   The "Save as" dialog opens.
3. Enter the name of the export file and select the file in which it is to be stored.
4. Select *.xlsx, *.sdf or *.xml as the file type.
5. Click the "Save" button.
6. If you have selected the "*.xlsx" format, select the export options:
   - Select whether to export tags and/or constants.
   - Select whether you want to export all elements of the table or just the elements that are actually being used in the program on the current CPU.
7. Click the "OK" button.

Result

The export file will be generated. Errors and warnings generated during export are indicated in the "Info" tab of the Inspector window.

See also

Basics for exporting and importing PLC tags (Page 4862)

Format of the export file (*.xlsx) (Page 4862)

Importing PLC tags (Page 4865)

Importing PLC tags

Requirement

A PLC tag table is open.

Procedure

To import PLC tags, follow these steps:

1. Click the "Import" button.
   The "Import" dialog box opens.
2. Select the file you want to import.
3. If you have selected a file in the format "XLSX", select whether you want to import tags and/or constants.
4. Click the "OK" button.

Result

Errors and warnings generated during import are indicated in the import log.

If the PLC tag table "All tags" is opened in the TIA Portal, a new user-defined PLC tag table for the imported tags is created. If the standard tag table or a user-defined tag table is opened, the imported tags are integrated in the open PLC tag table. Existing entries of the same name will be overwritten if they have the same name as the entries that will be imported.

References to PLC tags or constants that already exist in the project are updated automatically during import. The update is executed based on the name of the PLC tags and constants.

See also

Basics for exporting and importing PLC tags (Page 4862)

Format of the export file (*.xlsx) (Page 4862)

Exporting PLC tags (Page 4864)

Editing individual PLC tags with external editors

To edit individual PLC tags in external editors outside the TIA portal, you can export or import these tags using copy and paste. However, you cannot copy structured tags to an editor.

Requirement

A PLC tag table and an external editor are opened.

Procedure

To export and import individual PLC tags, follow these steps:

1. Select one or more PLC tags.
2. Select "Copy" in the shortcut menu.
3. Switch to the external editor and paste the copied tags.
4. Edit the tags as required.
5. Copy the tags in the external editor.
6. Switch back to the PLC tag table.
7. Select "Paste" in the shortcut menu.

Note

You also have the option of export or importing PLC tags as mass data.

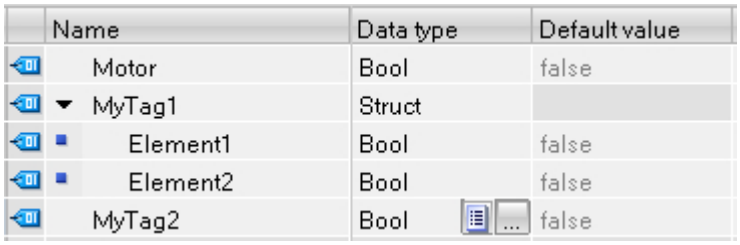
See also: Exporting and importing PLC tags (Page 4862)

11.11 Declaring PLC data types (UDT)

11.11.1 Structure of the declaration table for PLC data types

Structure of the declaration table for PLC data types

The figure below shows the structure of the declaration table for PLC data types.



	Name	Data type	Default value
	Motor	Bool	false
	MyTag1	Struct	
	Element1	Bool	false
	Element2	Bool	false
	MyTag2	Bool	false

Meaning of the columns

The following table shows the meaning of the individual columns. You can show or hide the columns as required. The number of columns displayed varies depending on the CPU series.

Column	Explanation
	Symbol you can click to move or copy the tag.
Name	Name of the tags.
Data type	Data type of the tags.
Default value	Value with which you predefine the tag in the declaration of the PLC data type. Specification of the default value is optional. If you do not specify any value the predefined value for the indicated data type is used. For example, the value "false" is predefined for BOOL.
Visible in HMI engineering	Shows whether the tag is visible by default in the HMI selection list.
Accessible from HMI/OPC UA	Indicates whether HMI/OPC UA can access this tag during runtime.
Writable from HMI/OPC UA	Indicates whether the tag from HMI/OPC UA can be written during runtime.
Setpoint value	Setpoint values are the values that will probably have to be fine tuned during commissioning. After commissioning, the values of these tags can be transferred to the offline program as start values and stored there.
Comment	Comment to document the tags.

See also

Creating PLC data types (Page 4868)

Show and hide table columns (Page 4877)

Basics of PLC data types (UDT) (Page 2277)

Using setpoints during commissioning (Page 4831)

11.11.2 Creating PLC data types

Requirement

The "PLC data types" folder opens in the project tree.

Procedure

To create a PLC data type, proceed as follows:

1. In the "PLC data types" folder, click the "Add new data type" command.
   A new declaration table for creating a PLC data type will be created and opened.
2. Select the PLC data type and select the "Rename" command in the shortcut menu.
3. Enter the name of the PLC data type.

The new PLC data type is created. You can find the PLC data type in the project tree in the "PLC data types" folder.

Adding comments for a PLC data type

To enter a comment for a PLC data type, follow these steps:

1. Right-click the PLC data type in the project tree.
2. Select the "Properties" command in the shortcut menu.
   A dialog containing the properties of the PLC data type opens.
3. Select the "Information" entry in the area navigation.
4. Enter the comment in the "Comment" input field.
5. Confirm your entry with "OK".

See also

Structure of the declaration table for PLC data types (Page 4867)

Basics of PLC data types (UDT) (Page 2277)

11.11.3 Delete PLC data types

Requirement

The PLC data type you want to delete is not open.

Procedure

To delete a PLC data type, follow these steps:

1. In the project tree, open the "PLC data types" folder.
2. Select the PLC data type to be deleted. You can also select several PLC data types by clicking on them one after the other while holding down the <Ctrl> key or by pressing and holding down <Shift> and clicking on the first and last data type.
3. Select the "Delete" command in the shortcut menu.

Note

If you delete a PLC data type, the blocks that use the data type will become inconsistent. These inconsistencies are marked in red in the block used. To remedy these inconsistencies, the data blocks have to be updated.

See also

Updating the block interface (Page 4791)

Updating data blocks (Page 4800)

Basics of PLC data types (UDT) (Page 2277)

11.11.4 Renumbering PLC data types

For performance reasons, PLC data types are processed internally with numbers. If there are number conflicts, these are resolved automatically. But this is not possible for a PLC data type which is used by a know-how protected block. The block must be recompiled when you change the number of the PLC data type which results in a password prompt for the know-how protected block. You can bypass this step by setting up a separate numbering scheme for your PLC data types. Use numbers greater than 5000.

Procedure

To change the default number of a PLC data type, follow these steps:

1. Open the project library in the "Libraries" task card.
2. Drag the compilable PLC data type to the "Types" folder.
   The "Add type" dialog opens.
3. Enter the properties of the new type.

4. Click "OK" to confirm.
5. Right-click the PLC data type in the project library and select the "Edit type" command from the shortcut menu.
6. Click "OK" to confirm your selection of the instance.
   The library view opens.
7. Close the library view.
   The extension "in test" is now added to the name of the PLC data type.
8. Right-click the PLC data type and select the "Properties" command from the shortcut menu.
9. Select the "General" group in the area navigation.
   You can now edit the number of the PLC data type.
10. Change the number of the PLC data type.
11. Click "OK" to confirm.
12. Right-click the PLC data type in the project library and select the "Release version" command from the shortcut menu.
    The PLC data type has a new number. The assigned number is retained even if the type of the PLC data type is revoked.

11.11.5 Programming the structure of PLC data types

11.11.5.1 Declaring tags of elementary data type

Requirement

A PLC data type is open.

Procedure

To declare a tag, follow these steps:

1. Enter a tag name in the "Name" column.
2. Enter the required data type in the "Data type" column. You will be supported by autocompletion during input.
3. Optional: Change the properties of the tags that are displayed in the other columns.
4. Repeat steps 1 to 3 for all tags that are to be declared.

See also

Editing tables (Page 278)

Editing the declaration table for PLC data types (Page 4875)

11.11.5.2 Declaring tags of the ARRAY data type

Requirement

A PLC data type is open.

Procedure

To declare a tag of the ARRAY data type, follow these steps:

1. Enter a tag name in the "Name" column.
2. Enter the "Array" data type in the "Data type" column. You will be supported by Autocomplete in this step.
   The "Array" dialog opens.
3. In the "Data type" text box, specify the data type of the array elements.
4. In the "ARRAY limits" input field, specify the high and low limit for each dimension.
   Example of a one-dimensional ARRAY:
   `[0..3]`
   Example of a three-dimensional ARRAY:
   `[0..3, 0..15, 0..33]`
5. Confirm your entry.
6. Optional: Change the properties of the tags that are displayed in the other columns.

Entering default values of ARRAY elements

To pre-assign default values to the individual elements of an ARRAY, follow these steps:

1. Click the triangle in front of the ARRAY data type tags.
   The ARRAY is expanded and the individual ARRAY elements are shown in separate lines.
2. Enter the desired values in the "Default value" column.

Adding comments for ARRAYs

To add a comment for the entire ARRAY, enter the comment in the top line of the ARRAY declaration. The comment is used as a preset comment for all lower-level ARRAY elements.

To provide specific comments for the individual elements of an ARRAY, follow these steps:

1. Click the triangle in front of the ARRAY data type tags.
   The ARRAY is expanded and the individual ARRAY elements are shown in separate lines.
2. Enter the desired values in the "Comment" column.

Note

Restriction when displaying ARRAY elements

Depending on the amount of work memory available on your computer, the following maximum limits apply when displaying ARRAYs:

- RAM < 8 GB: A maximum of 10,000 elements are displayed.
- RAM >= 8 GB: A maximum of 40,000 elements are displayed.

See also

Structure of the declaration table for PLC data types (Page 4867)

ARRAY (Page 2286)

Using autocompletion (Page 4706)

11.11.5.3 Declaring tags of STRUCT data type

Requirements

A PLC data type is open.

Procedure

To declare a tag of the STRUCT data type, follow these steps:

1. Enter a tag name in the "Name" column.
2. Enter "Struct" in the "Data type" column. You will be supported by autocompletion during input.
   An empty, indented row is inserted after the new tag.
3. Insert the first structural element in the first empty row.
   An additional empty row is inserted after the element.
4. Select a data type for the structure element.
5. Optional: Change the properties of the structural element that is displayed in the other columns.
6. Repeat steps 3 to 5 for all additional structure elements.
   It is not necessary to end the structure explicitly. The structure ends with the last element that is entered.
7. To insert a new tag after the structure, leave a blank row after the end of the structure and then start the new tag in the second empty row.

Result

The tag of STRUCT data type is created.

See also

Basic information on STRUCT (Page 2282)

Structure of the declaration table for PLC data types (Page 4867)

Using autocompletion (Page 4706)

11.11.5.4 Declaring tags based on a different PLC data type

Requirement

- A global data block is open.
- A PLC data type is declared in the current CPU.

Procedure

To declare a tag based on a different PLC data type, follow these steps:

1. Enter a tag name in the "Name" column.
2. Enter the PLC data type in the "Data type" column. You will be supported by Autocomplete during input.
   The tag is created.
3. Optional: change the properties of the tags that are displayed in the other columns of the table.
   If you have already defined default values or comments for the tags within a PLC data type during declaration of the PLC data type, these are shown in gray. You can change these values here.
   The changed values are displayed in black and apply only to the specific point of use.

Note

If you change or delete PLC data types that are being used in the data block, the data block becomes inconsistent. To eliminate this inconsistency, you must recompile the program.

See also

Structure of the declaration table for PLC data types (Page 4867)

Using autocompletion (Page 4706)

11.11.6 Editing tag properties in PLC data types

11.11.6.1 Properties of tags in PLC data types

Properties

The following table gives an overview of tag properties in PLC data types:

Group	Property	Description
General	Name	Name of the tags.
	Data type	Data type of the tags.
	Default value	Default value of the tag in the interface of a higher-level code block or in a PLC data type. The values contained in the "Default value" column can only be changed in the higher-level code block or PLC data type. The values are only displayed in the data block.
	Start value	Not relevant in PLC data types
	Comment	Comment on the tag.
Attributes	Retain	Not relevant in PLC data types
User-defined attributes	CFC_Visible	Visible Indicates whether a parameter is visible in CFC.
	CFC_Configurable	Configurable Indicates whether a parameter is configurable in CFC.
	CFC_ForTest	For test Indicates whether a parameter is registered for the CFC test mode.
	CFC_Interconnectable	Interconnectable Indicates whether a parameter is interconnectable in CFC.
	CFC_EnableTagReadback	Enable tag readback Indicates whether a parameter is relevant for the "Read back chart" function in CFC.
	CFC_EnumerationTexts	Enumeration texts Assigns a parameter to an enumeration in CFC.
	CFC_EngineeringUnit	Engineering unit Assigns a parameter to a unit in CFC.
	CFC_LowLimit	Low limit Defines the low limit for the parameter in CFC.
	CFC_HighLimit	High limit Defines the high limit for the parameter in CFC.

See also

Changing the properties of tags in PLC data types (Page 4875)

Structure of the declaration table for PLC data types (Page 4867)

11.11.6.2 Changing the properties of tags in PLC data types

Editing general properties in the declaration table

To edit the general properties of one or more tags, follow these steps:

1. Open the PLC data type.
2. Change the entries in the columns.

Editing detailed properties in the properties window

To edit the detailed properties of an individual tag, follow these steps:

1. Select a tag in the table.
2. Select the "Properties" command in the shortcut menu.
   The inspector window shows the properties of the tag in the "General" and "Attributes" areas.
3. Select the required area in the area navigation.
4. Change the entries in the text boxes.

See also

Updating the block interface (Page 4791)

Updating data blocks (Page 4800)

11.11.7 Editing the declaration table for PLC data types

11.11.7.1 Inserting table rows

Procedure

Proceed as follows to insert a row above the selected row:

1. Select the row in front of which you want to insert a new row.
2. Click the "Insert row" button on the toolbar of the table.

Result

A new row is inserted above the selected row.

11.11.7.2 Inserting table rows

Procedure

Proceed as follows to insert a row below the selected row:

1. Select the row below which you want to insert a new row.
2. Click the "Add row" button on the table toolbar.

Result

A new empty row will be inserted below the selected row.

11.11.7.3 Deleting tags

Procedure

Follow the steps below to delete elements:

1. Select the row with the element to be deleted. You can also select several rows by clicking on them one after the other while holding down the <Ctrl> key or by pressing and holding down <Shift> and clicking on the first and last row.
2. Select the "Delete" command in the shortcut menu.

See also

Updating the block interface (Page 4791)

Updating data blocks (Page 4800)

11.11.7.4 Automatically filling in successive cells

You can load the contents of one or several table cells into the cells below, automatically filling in the successive cells.

If you automatically fill in cells in the "Name" column, a consecutive number will be appended to each name. For example, "Motor" will become "Motor_1".

You can define individual or more cells as well as entire rows as source area.

If less rows exist in the open table than you want to fill, then you will first have to insert additional empty rows.

Requirement

- The table is open.
- Sufficient declaration rows are available.

Procedure

To automatically fill in successive cells, follow these steps:

1. Select the cells to be loaded.
2. Click the "Fill" symbol in the bottom right corner of the cell.
   The mouse pointer is transformed into a crosshair.
3. Keep the mouse button pressed and drag the mouse pointer downwards over the cells that you want to fill in automatically.
4. Release the mouse button.
   The cells are filled in automatically.
5. If entries are already present in the cells that are to be automatically filled in, a dialog appears. In this dialog you can indicate whether you want to overwrite the existing entries or insert new rows for the new tags.

11.11.7.5 Show and hide table columns

You can show or hide the columns in a table as needed.

Procedure

To show or hide table columns, follow these steps:

1. Click a column header.
2. Select the "Show/Hide" command in the shortcut menu.
   The selection of available columns is displayed.
3. To show a column, select the column's check box.
4. To hide a column, clear the column's check box.
5. To hide or show several columns, click "More" and activate or deactivate the check box of the corresponding columns in the "Show/Hide" dialog.

11.12 Creating LAD programs

11.12.1 Basic information on LAD

11.12.1.1 LAD programming language

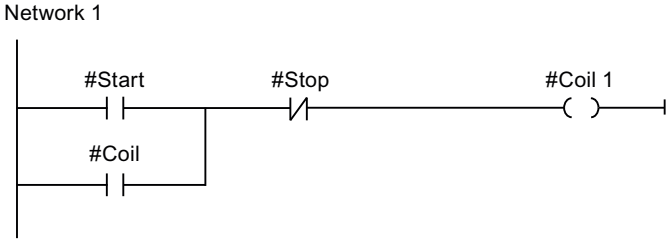
Overview of the Ladder Logic (LAD) programming language

LAD is a graphical programming language. The representation is based on circuit diagrams.

The program is mapped in one or more networks. A network contains a power rail on the left where the rungs originate. The binary signal scans are arranged in the form of contacts on the rungs. The serial arrangement of the elements on a rung creates a series connection; arrangement on simultaneous branches creates a parallel connection. Complex functions are represented by boxes.

Example of networks in LAD

The following figure shows a LAD network with two normally open contacts, one normally closed contact and one coil:



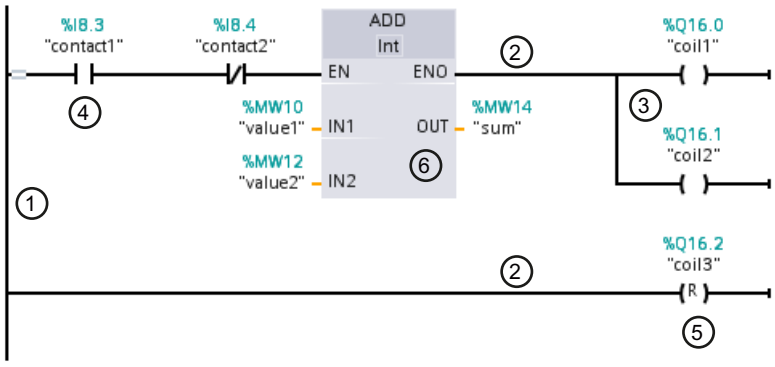
11.12.1.2 Overview of the LAD elements

LAD elements

A LAD program consists of separate elements that you can arrange in series or parallel on the power rail of a network. Most program elements must be supplied with tags.

There is at least one rung from the power rail. Network programming starts at the left edge of the rung. You can expand the power rail by several rungs and branches.

For example, the following figure shows elements of a LAD network:



1) Power rail

2) Rung

3) Branch

4) Contact

5) Coil

6) Box

Power rail

Each LAD network consists of a power rail that contains at least one rung. A network can be extended by adding additional rungs. You can use branches to program parallel connections in the specific rungs.

Contacts

You can use contacts to create or interrupt a current-carrying connection between two elements. The current is relayed from left to right. You can use contacts to query the signal state or the value of an operand and control it depending on the result of the current flow.

The following types of contact are available to you in a LAD program:

- Normally open contact:
  Normally open contacts forward the current if the signal state of a specified binary operand is "1".
- Normally closed contacts:
  Normally closed contacts forward the current if the signal state of a specified binary operand is "0".
- Contact with additional function:
  Contacts with additional function forward the current if a specific condition is met. With these contacts you can also execute an additional function, such as an RLO edge detection and a comparison.

Coils

You can use coils to control binary operands. Coils can set or reset a binary operand depending on the signal state of the result of logic operation.

The following types of coils are available to you in a LAD program:

- Standard coils:
  Standard coils set a binary operand if current flows in the coil. The "Assignment" instruction is an example of a standard coil.
- Coils with additional function:
  These coils have additional functions in addition to the evaluation of the logic operation result. Coils for RLO edge detection and program control are examples of coils with additional function.

Boxes

Boxes are LAD elements with complex functions. The empty box is an exception. You can use the empty box as a placeholder in which you can select the required instruction.

The following types of boxes are available to you in a LAD program:

- Boxes without EN/ENO mechanism:
  A box is executed depending on the signal state at the box inputs. The error status of the processing cannot be queried.
- Boxes with EN/ENO mechanism:
  A box is only executed if the enable input "EN" carries the signal state "1". If the box is processed correctly, the "ENO" enable output has signal state "1". If an error occurs during the processing, the "ENO" enable output is reset.

Calls of code block are also shown in the network as boxes with EN/ENO mechanism.

See also

Rules for the use of LAD elements (Page 4890)

11.12.2 Settings for LAD

11.12.2.1 Overview of the settings for LAD

Overview

The following table shows the settings that you can make:

Group	Setting	Description
Font	Font size	Font size in program editor
View	Layout	Compact or wide Changes the vertical spacing between operands and other objects (such as operand and contact). The change becomes visible once the block is reopened.
	With absolute information	Additional display of the absolute addresses
Operand field	Maximum width	Maximum number of characters that can be entered horizontally in the operand field. This setting recalculates the layout of the networks.
	Maximum height	Maximum number of characters that can be entered vertically in the operand field. This setting recalculates the layout of the networks.

See also

Changing the settings (Page 4881)

11.12.2.2 Changing the settings

Procedure

To change the settings, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.
3. Change the settings.

Result

The change will be loaded directly, there is no need to save it explicitly.

See also

Overview of the settings for LAD (Page 4881)

11.12.3 Working with networks

11.12.3.1 Using networks

Function

The user program is created in the block within networks. For a code block to be programmed, it must contain at least one network. To achieve a better overview of the user program, you can also subdivide your program into several networks.

You have the option to insert networks for the textual programming languages SCL and STL in LAD and FBD blocks, and then use instructions in these programming languages. Depending on the CPU you are using, you can insert networks for the following programming languages:

- S7-300/400: STL networks
- S7-1200: SCL networks
- S7-1500: STL and SCL networks

The programming editor always adapts to the network that currently has the focus. This means that you have access to the SCL instructions as well as the SCL functions when programming an SCL network.

Note

You cannot access the blocks that contain SCL networks via the Openness interface.

EN/ENO mechanism in blocks with different network languages

You can also use the EN/ENO mechanism in blocks with different network languages. Each programming language displays the error status differently:

- SCL has an "ENO" tag that stores the error status and can be queried. Direct access to this tag is only possible with SCL.
- LAD/FBD/STL have no special tag for "ENO". However, you can read the error status for STL from the BR bit, and query it via the RET coil for LAD/FBD.

The following rules apply to reading out the error status for the entire block:

- The last network in the block is a LAD/FBD network:
  If you do not use RET coils, the error status is "TRUE" by default.
- The last network in the block is an STL network:
  The BR bit determines the error status. The BR bit can be edited in STL networks using the BR tab.
- The last network in the block is an SCL network:
  The "ENO" tag determines the error status of the block.

See also

Inserting networks (Page 4883)

Inserting network title (Page 4887)

Entering a network comment (Page 4888)

Navigating networks (Page 4889)

11.12.3.2 Inserting networks

You can either insert a network of the same programming language or a network for a textual programming language in an LAD or FBD block. Keep in mind, however, that not all CPU types support all textual languages.

Requirement

An LAD or FBD block is open.

Insert LAD or FBD network

To insert a new network, follow these steps:

1. Select the network after which you want to insert a new network.
2. Select the "Insert network" command in the shortcut menu.

Insert STL or SCL network

To insert a new network, follow these steps:

1. Select the network after which you want to insert a new network.
2. Select the "Insert STL network" or "Insert SCL network" command in the shortcut menu.

Result

A new, empty network is inserted in the block in the corresponding programming language.

See also

Using networks (Page 4882)

Selecting networks (Page 4884)

Copying and pasting networks (Page 4885)

Deleting networks (Page 4885)

Expanding and collapsing networks (Page 4886)

Inserting network title (Page 4887)

Entering a network comment (Page 4888)

Navigating networks (Page 4889)

11.12.3.3 Selecting networks

Requirements

A network is available.

Selecting a network

To select a network, follow these steps:

1. Click the title bar of the network that you want to select.

Selecting several networks

Proceed as follows to select several individual networks:

1. Press and hold down the <Ctrl> key.
2. Click all the networks that you want to select.

To select several successive networks, follow these steps:

1. Press and hold down the <Shift> key.
2. Click the first network that you want to select.
3. Click the last network that you want to select.
   The first and last networks and all those in between are selected.

See also

Using networks (Page 4882)

Inserting networks (Page 4883)

Copying and pasting networks (Page 4885)

Deleting networks (Page 4885)

Expanding and collapsing networks (Page 4886)

Inserting network title (Page 4887)
Entering a network comment (Page 4888)
Navigating networks (Page 4889)

11.12.3.4 Copying and pasting networks

Copied networks can be pasted within the block or in another block. Networks that were created in LAD or FBD can also be inserted in blocks of the respective other programming language.

Requirement

A network is available.

Procedure

To copy and paste a network, follow these steps:

1. Select the network or networks to be copied.
2. Select "Copy" in the shortcut menu.
3. Select the network after which you want to paste in the copied network.
4. Select "Paste" in the shortcut menu.

See also

Using networks (Page 4882)
Inserting networks (Page 4883)
Selecting networks (Page 4884)
Deleting networks (Page 4885)
Expanding and collapsing networks (Page 4886)
Inserting network title (Page 4887)
Entering a network comment (Page 4888)
Navigating networks (Page 4889)

11.12.3.5 Deleting networks

Requirement

A network is available.

Procedure

To delete a network, follow these steps:

1. Select the network that you want to delete.
2. Select the "Delete" command in the shortcut menu.

See also

Using networks (Page 4882)
Inserting networks (Page 4883)
Selecting networks (Page 4884)
Copying and pasting networks (Page 4885)
Expanding and collapsing networks (Page 4886)
Inserting network title (Page 4887)
Entering a network comment (Page 4888)
Navigating networks (Page 4889)

11.12.3.6 Expanding and collapsing networks

Requirements

A network is available.

Opening and closing a network

To open a network, follow these steps:

1. Click on the right arrow in the network title bar.

To close a network, follow these steps:

1. Click on the down arrow in the network title bar.

Opening and closing all networks

To open and close all networks, follow these steps:

1. In the toolbar, click "Open all networks" or "Close all networks".

See also

Using networks (Page 4882)
Inserting networks (Page 4883)
Selecting networks (Page 4884)
Copying and pasting networks (Page 4885)

Deleting networks (Page 4885)

Inserting network title (Page 4887)

Entering a network comment (Page 4888)

Navigating networks (Page 4889)

11.12.3.7 Inserting network title

The network title is the header of a network. The length of the network title is limited to one line. You can enter the title manually or set it automatically. When you set it automatically, you can do this for individual networks or use the settings to specify that the network title is always set automatically.

For automatic insertion of the network title, the comment of the operand in one of the following instructions in the network is evaluated:

- Assignment
- Set output
- Reset output

The instruction that is listed first in the network is used.

The network title is only inserted automatically when the following conditions are fulfilled:

- The network does not have a title yet.
- The operand of the instruction used for the comment has a comment.

Note

Note the following restrictions for automatic insertion of the network title:

- The network title is not adapted if you change the comment of the operand at a later time.
- The network title is not adapted if you change the operand of the instruction.
- The network title is only set by the writing instructions listed above.
- If the operand is of the data type array, the comment of the array is used and not the comments of the array elements.
- Comments of invalid operands are not taken into consideration.

Entering the network title manually

To enter a network title, follow these steps:

1. Click on the title bar of the network.
2. Enter the network title.

Setting the network title automatically

To specify that the network titles are always set automatically, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.
3. Select the "Set network title automatically" check box in the "Additional settings" group.
   The network titles are set automatically as of this time if the conditions listed above are fulfilled.

To set an individual network title automatically, follow these steps:

1. Right-click "Network <Number of the network>" in the title bar of a network.
2. Select the "Set network title automatically" command in the shortcut menu.
   The title of the selected network is set based on the comment of the operand if the conditions listed above are fulfilled.

See also

Using networks (Page 4882)

Inserting networks (Page 4883)

Selecting networks (Page 4884)

Copying and pasting networks (Page 4885)

Deleting networks (Page 4885)

Expanding and collapsing networks (Page 4886)

Entering a network comment (Page 4888)

Navigating networks (Page 4889)

11.12.3.8 Entering a network comment

You can use network comments to provide comments on the program contents of individual networks. For example, you can indicate the function of the network or draw attention to special characteristics.

Requirement

A network is available.

Procedure

To enter a network comment, follow these steps:

1. Click on the right arrow before the network title.
2. If the comment area is not visible, click "Network comments on/off" in the toolbar.
   The comment area is displayed.

3. Click "Comment" in the comment area.
   The "Comment" text passage is selected.
4. Enter the network comment.

See also

Using networks (Page 4882)

Inserting networks (Page 4883)

Selecting networks (Page 4884)

Copying and pasting networks (Page 4885)

Deleting networks (Page 4885)

Expanding and collapsing networks (Page 4886)

Inserting network title (Page 4887)

Navigating networks (Page 4889)

11.12.3.9 Navigating networks

You can navigate straight to a specific position within a block.

Procedure

To navigate to a specific position within a block, follow these steps:

1. Right-click in the code area of the programming window.
2. Select the "Go to > Network/line" command in the shortcut menu.
   The "Go to" dialog will open.
3. Enter the network to which you want to navigate.
4. Enter the line number of the network to which you want to navigate.
5. Confirm your entry with "OK".

Result

The relevant line will be displayed if this is possible. If the network or line requested does not exist, the last existing network or the last existing line in the network requested will be displayed.

See also

Using networks (Page 4882)

Inserting networks (Page 4883)

Selecting networks (Page 4884)

Copying and pasting networks (Page 4885)

Deleting networks (Page 4885)
Expanding and collapsing networks (Page 4886)
Inserting network title (Page 4887)
Entering a network comment (Page 4888)

11.12.4 Inserting LAD elements

11.12.4.1 Rules for the use of LAD elements

Rules

Note the following rules when inserting LAD elements:

- Every LAD network must terminate with a coil or a box. However, the following LAD elements must not be used to terminate a network:
  - Comparator boxes
  - Instructions for positive and negative RLO edge detection
- The starting point of the branch for a box connection must always be the power rail. Logic operations or other boxes can be present in the branch before the box.
- Only contacts can be inserted into simultaneous branches with preceding logic operations. The contact for negating the result of logic operation (-|NOT|-) is an exception here. The contact for negating the result of logic operation, as well as coils and boxes, can be used in simultaneous branches if they originate directly from the power rail.
- Constants (e.g. TRUE or FALSE) cannot be assigned to normally closed or normally open contacts. Instead, use operands of the BOOL data type.
- Only one jump instruction can be inserted in each network.
- Only one jump label can be inserted in each network.
- Instructions with positive or negative edge detection may not be arranged directly at the left margin of the rung as they requires a prior logic operation.

Placement rules for S7-1200/1500 CPUs

The following table sets out the instructions that can only be positioned at the end of the network:

Instruction		Preceding logic operation required
Mnemonics	Name	
SET_BF	Set bit field	No
RESET_BF	Reset bit field	No
JMP	Jump if RLO = 1	No
JMPN	Jump if RLO = 0	Yes

Instruction		Preceding logic operation required
Mnemonics	Name	
JMP_LIST	Define jump list	No
SWITCH	Jump distributor	No
RET	Return	No

Placement rules for S7-300/400 CPUs

The following table sets out the instructions that can only be positioned at the end of the network:

Instruction		Preceding logic operation required
Mnemonics	Name	
S	Set output	Yes
R	Reset output	Yes
SP	Start pulse timer	Yes
SE	Start extended pulse timer	Yes
SD	Start on-delay timer	Yes
SS	Start retentive on-delay timer	Yes
SF	Start off-delay timer	Yes
SC	Set counter value	Yes
CU	Count up	Yes
CD	Count down	Yes
JMP	Jump if RLO = 1	No
JMPN	Jump if RLO = 0	Yes
RET	Return	No
OPN	Open global data block	No
OPNI	Open instance data block	No
CALL	Call block	No
SAVE	Save RLO in BR bit	No
MCRA	Enable MCR range	No
MCRD	Disable MCR range	No
MCR<	Open MCR ranges	No
MCR>	Close MCR ranges	No

Note

Calling blocks with binary inputs in S7-300/400 CPUs

The following FAQ describes the special features when calling blocks with binary inputs:

FAQ 13988019: What should you watch out for when programming nested FB/FC calls with binary inputs in FBD? (Page 4892)

See also

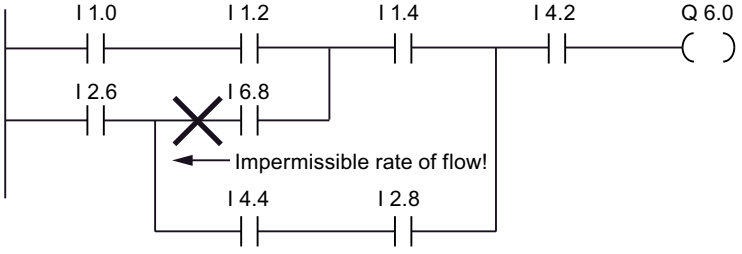
Overview of the LAD elements (Page 4878)

FAQ 13988019 (https://support.industry.siemens.com/cs/de/de/view/13988019)

11.12.4.2 Prohibited interconnections in LAD

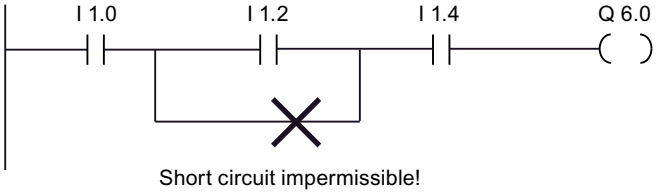
Power flow from right to left

No branches can be programmed that could result in a power flow in the reverse direction.



Short-circuit

No branches may be programmed that would cause a short-circuit.



Logic operations

The following rules apply to logic operations:

- Only Boolean inputs can be combined with preceding logic operations.
- Only the first Boolean output can be combined with a further logic operation.
- Only one complete logical path can exist per network. Paths that are not connected can be linked.

See also

Rules for the use of LAD elements (Page 4890)

11.12.4.3 Inserting LAD elements using the "Instructions" task card

Requirement

A network is available.

Procedure

To insert a LAD element into a network using the "Instructions" task card, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to the LAD element that you want to insert.
3. Use drag-and-drop to move the element to the desired place in the network.
   If the element is an internal system function block (FB), the "Call options" dialog opens. In this dialog, you can determine if the data of the inserted element is to be stored in a single, multi or parameter instance. If you select "Single instance", you can find the new instance data block once it is created in the "Program resources" folder of the project tree under "Program blocks > System blocks". You can find multi-instances in the block interface in the "Static" section and parameter instances in the "InOut" section.

Or:

1. Select the point in the network at which you want to insert the element.
2. Open the "Instructions" task card.
3. Double-click on the element you want to insert.
   If the element is an internal system function block (FB), the "Call options" dialog opens. In this dialog, you can determine if the data of the inserted element is to be stored in a single, multi or parameter instance. If you select "Single instance", you can find the new instance data block once it is created in the "Program resources" folder of the project tree under "Program blocks > System blocks". You can find multi-instances in the block interface in the "Static" section and parameter instances in the "InOut" section.

Result

The selected LAD element is inserted with placeholders for the parameters.

11.12.4.4 Inserting LAD elements using an empty box

Requirement

A network is available.

Procedure

To insert an LAD element into a network using an empty box, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "General > Empty box" in the "Basic instructions" palette.

3. Use a drag-and-drop operation to move the "Empty box" element to the desired place in the network.
4. Position the cursor over the triangle in the top right-hand corner of the empty box.
   A drop-down list is displayed.
5. Select the required LAD element from the drop-down list.
   If the element is an internal system function block (FB), the "Call option" dialog opens. In this dialog, you can determine if the data of the inserted element is to be stored in a single, multi or parameter instance. If you select "Single instance", you can find the new instance data block once it is created in the "Program resources" folder of the project tree under "Program blocks > System blocks". You can find multi-instances in the block interface in the "Static" section and parameter instances in the "InOut" section.

Result

The empty box is changed to the respective LAD element. Placeholders are inserted for the parameters.

11.12.4.5 Selecting the data type of a LAD element

Selecting a data type

Introduction

Some instructions can be executed with several different data types. If you use one of these instructions in the program, you have to specify a valid data type for the instruction at the specific point in the program. For some instructions, you have to select the data types for the inputs and outputs separately.

Note

The valid data type (BOOL) for the tags on the enable input EN and the enable output ENO is predefined by the system and cannot be changed.

The valid data types for an instruction are listed in the instruction drop-down list. You specify the data type of the instruction by selecting an entry from the drop-down list. If the data type of an operand differs from the data type of the instruction and cannot be converted implicitly, the operand is displayed in red and a rollout with the corresponding error message appears.

Data type selection of mathematical instructions

Some mathematical instructions provide you with the option of having the data type automatically set corresponding to the data types of the operand. In the drop-down list for data type selection, these instructions have the entry "Auto" in addition to the actual data types. If you select this entry and then allocate the first operand, the data type of the operand is selected as data type for the instruction. The entry in the drop-down list changes to "Auto (<Data type>)", e.g. "Auto (Real)". If you allocate additional operands, the automatically set data type of the instruction is adjusted according to the following criteria:

- You supply all other operands with tags of the same data type:
  The data type of the instruction is not changed.
- You supply all other operands with tags whose data type is smaller than the data type of the instruction:
  The data type of the instruction is not changed. For the operand with the smaller data type, an implicit conversion is conducted if necessary.
- You supply an additional operand with a tag whose data type is greater than the data type of the instruction:
  The data type of the instruction is changed to the larger data type. An implicit conversion is performed, if necessary, for operands that deviate from the newly set data type of the instruction.

Each change in the data type of an operand can result in a change of the data type of the instruction. Other operands may possibly be implicitly converted as a result. Operands for which an implicit conversion is performed are marked with a gray square.

Note

Please also observe the information on data type conversion for your device and, in particular, the notes on the IEC check.

See also: Data type conversion (Page 2465)

See also

Defining the data type of an instruction (Page 4895)

Defining the data type of an instruction

Introduction

Some instructions can be executed with several different data types. When you insert such instructions into your program, you must specify the data type for these instructions at the actual point in the program.

Specifying the data type by means of the drop-down list

To define the data type of an instruction using the drop-down list, follow these steps:

1. Insert the instruction at the required point in the program using drag-and-drop.
   The entry "???" (undefined) is displayed in the drop-down list of the inserted instruction.
2. Click the triangle in the upper corner of the drop-down list.
   The drop-down list will open to display the data types valid for the instruction.
3. Select a data type from the drop-down list.
   The selected data type is displayed.
4. If the instruction has two drop-down lists, select the data type for the instruction inputs in the left-hand drop-down list and the data type for the instruction outputs in the right-hand drop-down list.

Specifying data type by assigning tags

To define the data type of an instruction by assigning tags, follow these steps:

1. Insert the instruction at the required point in the program using drag-and-drop.
   The entry "???" (undefined) is displayed in the drop-down list of the inserted instruction.
2. At an input or output, specify a valid tag, the data type of which is to be applied as the instruction data type.
   The data type of the tag is displayed in the drop-down list.
3. Enter a valid tag at an input and a valid tag at an output if data types need to be defined for both the inputs and outputs of the instruction. The tag specified at the input determines the data type of the inputs; the tag specified at the output determines the data type of the outputs of the instruction.

Automatically specifying the data type of mathematical instructions

To automatically specify the data type for mathematical instructions, follow these steps:

1. Insert the mathematical instruction at the required point in the program using drag-and-drop.
   The entry "???" (undefined) is displayed in the drop-down list of the inserted instruction.
2. Select the "Auto" entry from the drop-down list.
3. Enter a valid tag at an input or output.
   The data type of the tag is applied as data type of the instruction. The entry in the drop-down list changes to "Auto (<Data type>)".

See also: Selecting a data type (Page 4894)

See also

Selecting a data type (Page 4894)

11.12.4.6 Using favorites in LAD

Adding LAD elements to Favorites

Requirement

- A block is open.
- The multipane mode is set for the "Instructions" task card or the Favorites are also displayed in the editor.

Procedure

To add SCL instructions to the Favorites, follow these steps:

1. Open the "Instructions" task card.
2. Maximize the "Basic instructions" pane.
3. Navigate in the "Basic instructions" pane to the instruction that you want to add to the Favorites.
4. Drag-and-drop the instruction into the "Favorites" pane or into the Favorites area in the program editor.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Removing LAD elements from Favorites (Page 4898)

Overview of the program editor (Page 4684)

Inserting LAD elements using favorites

Requirement

- A block is open.
- Favorites are available.

Procedure

To insert an instruction into a program using Favorites, follow these steps:

1. Drag-and-drop the desired instruction from Favorites to the desired position.

Or:

1. Select the position in the program where you want to insert the instruction.
2. In the Favorites, click on the instruction you want to insert.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Removing LAD elements from Favorites (Page 4898)

Overview of the program editor (Page 4684)

Removing LAD elements from Favorites

Requirement

A code block is open.

Procedure

To remove instructions from Favorites, follow these steps:

1. Right-click on the instruction you want to remove.
2. Select the "Remove instruction" command in the shortcut menu.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Adding LAD elements to Favorites (Page 4897)

Inserting LAD elements using favorites (Page 4897)

Overview of the program editor (Page 4684)

11.12.4.7 Insert block calls in LAD

Inserting block calls using a drag-and-drop operation

You can insert calls for existing functions (FC) and function blocks (FB) using a drag-and-drop operation from the project tree. If you call function blocks from other function blocks, you can either call them as a single instance, multi-instance or parameter instance.

See also: AUTOHOTSPOT

Requirement

- A network is available.
- The block that is to be called is available.

Inserting a call of a function (FC)

To insert a call of a function (FC) into a network using a drag-and-drop operation, follow these steps:

1. Drag the function from the project tree to the required network.

Inserting a call for a function block (FB)

To insert a call for a function block (FB), follow these steps:

1. Drag the function block from the project tree to the required network.
   The "Call options" dialog opens.
2. In the dialog, enter whether you want to call the block as a single, multi or parameter instance.
   - If you click on the "Single instance" button, you will have to enter a name in the "Name" text box for the data block that you want to assign to the function block.
     If you call a block that contains monitoring, assign a ProDiag function block to the monitoring functions in the "ProDiag FB" text box.
   - If you click on the "Multi-instance" button, you will have to enter the name of the tag in the "Name in the interface" text box; this is the name that you use to enter the called function block as a static tag in the interface of the calling block.
   - If you click on the "Parameter instance" button, enter the name of the in/out (InOut) parameter to which the instance should be passed during runtime in the "Name in the interface" text box.
3. If you call a block that contains monitoring, assign a ProDiag function block to the monitoring functions.
4. Confirm your entries with "OK".

Result

The function or the function block is inserted with its parameters. You can then assign the parameters.

See also: AUTOHOTSPOT

Note

If when calling a function block you specify an instance data block that does not exist, it will be created.

See also

Updating block calls in LAD (Page 4900)

Changing the instance type (Page 4902)

Updating block calls in LAD

If interface parameters of a called block are changed, the block call can no longer be executed correctly. You can avoid such inconsistent block calls by updating the block calls.

You have the following options for updating the block calls:

- Explicit updating of all inconsistent block calls in the program editor.
  The inconsistent block calls within the open block are updated. The following actions are carried out in the process:
  - New parameters are added.
  - Deleted parameters are removed if they are not connected.
  - Renamed parameters get the new parameter names.
- Explicit updating of a block call in the program editor.
  The "Interface update" dialog is displayed. In this dialog, you have the option of changing the connection of the operands of the new interface. The inconsistent call of this block is then updated. The following actions are carried out in the process:
  - New parameters are added.
  - Deleted parameters are removed if they are not connected.
  - Renamed parameters get the new parameter names.
- Implicit updating during compilation.
  All block calls in the program as well as the used PLC data types will be updated. Note that when you call functions (FCs) before the next compilation process, all new formal parameters must be supplied with actual parameters.

Updating all inconsistent block calls in the program editor

To open all block calls in a block, follow these steps:

1. Open the calling block in the program editor.
2. Click "Update inconsistent block calls" in the toolbar.

Updating a specific block call in the program editor

To update a specific block call in the program editor, follow these steps:

1. Open the calling block in the program editor.
2. Right-click on the block call that you want to update.
3. Select the "Update" command in the shortcut menu.
   The "Interface update" dialog opens. This dialog shows the differences between the block interface in use and the changed interface of the called block.
4. If necessary, change the connection of the operands. To do this, you have the following options:
   - You can use either a drag-and-drop operation or a cut/copy-and-paste operation to move the operand from the old interface to the new interface.
   - You can delete an operand.
   - You can rename an operand.
   - You can specify a new operand via autocompletion.
5. Click "OK" to update the block call. If you want to cancel the update, click "Cancel".

Note

Note that the "Update block call" command is only available provided you did not previously update all block calls in the editor with the "Update inconsistent block calls" command.

Update block calls during compilation

Follow these steps to update all block calls and uses of PLC data types during compilation implicitly:

1. Open the project tree.
2. Select the "Program blocks" folder.
3. Select the command "Compile > Software (rebuild all blocks)" in the shortcut menu.

See also

Inserting block calls using a drag-and-drop operation (Page 4899)

Changing the instance type (Page 4902)

Changing the block call

You have the option of changing the called block for a block call. But keep in mind that no new instance data blocks are created, for example, when changing from a function (FC) to a function block (FB).

Procedure

To change the called block of a block call, follow these steps:

1. Click on the name of the called block within the block call and press the <F2> key. Or double-click the name of the called block.
   A text box opens, and the name of the currently called block is selected.
2. Enter the name of the block you want to call or select a block in the autocompletion.
3. If you want to call an FB, create a new instance data block, if necessary, and specify it as operand.

Changing the instance type

Instance type

There are two ways of calling function blocks:

- As a single instance
- As a multiple instance
- As parameter instance

See also: AUTOHOTSPOT

You can modify a defined instance type at any time.

Requirement

The user program contains a block call.

Procedure

To change the instance type of a function block, follow these steps:

1. Open the code block and select the block call.
2. Select the "Change instance" command in the shortcut menu.
   The "Call options" dialog opens.
3. Click on the button "Single instance", "Multi-instance" or "Parameter instance".
   - If you select the "Single instance" instance type, enter a name for the data block that is to be assigned to the function block.
   - If you select "Multiple instance" as the instance type, enter in the "Name in the interface" text field the name of the tag with which the called function block is to be entered as a static tag in the interface of the calling block.
   - If you select "Parameter instance" as the instance type, enter the name of the in/out (InOut) parameter to which the instance should be passed during runtime in the "Name in the interface" text box.
4. Confirm your entries with "OK".

Note

The previous instances are not deleted automatically.

See also

Inserting block calls using a drag-and-drop operation (Page 4899)

Updating block calls in LAD (Page 4900)

11.12.4.8 Inserting complex LAD instructions

Using the "Calculate" instruction (S7-1200, S7-1500)

Requirement

A network is available.

Procedure

Proceed as follows to use the "Calculate" instruction:

1. Open the "Instructions" task card.
2. Navigate to "Math functions > CALCULATE" in the "Basic instructions" pane.
3. Use drag-and-drop to move the element to the desired place in the network.
   The instruction "Calculate" will be inserted for the data type with a placeholder expression and question mark.
4. Enter the data type for the calculation.
5. Enter the operands for the calculation.

   Note

   The calculation is run with the inputs of the "Calculate" instruction. If you want to use constants you must also insert appropriate inputs for them.

6. Click on the "Edit 'Calculate' instruction" button to replace the placeholder expression with the correct expression.
   The "Edit 'Calculate' instruction" dialog will open.

7. Enter the required expression in the "OUT:= " text box.

   Note

   In the "Example" area you can find an example of a valid expression and possible instructions that you can use.

   To determine a value with the help of Pythagoras' theorem, for example, enter "OUT := SQRT (SQR (IN1) + SQR (IN2))".

8. Confirm your entry with "OK".

See also

CALCULATE: Calculate (Page 2695)

11.12.4.9 Using free-form comments

Basic information on using free-form comments in LAD

Introduction

Free-form comments allow you to add comments to the source code for graphic programming languages similar to line comments for textual languages.

Free-form comments can be used for the following elements:

- Boxes
- Coils

Commenting out program code

To deactivate individual instructions or entire program sections in the LAD and FBD programming languages, you must set a bit parallel to the instruction or in series to an instruction block. Commenting out, as in STL and SCL, is not possible in LAD and FBD.

Examples of commenting out program code in all programming languages are available in the Siemens Industry Online under the following FAQ ID: 109482004 (https://support.industry.siemens.com/cs/hu/en/view/109482004)

See also

Inserting free-form comments (Page 4905)

Editing free-form comments (Page 4905)

Deleting free-form comments (Page 4906)

Inserting free-form comments

Requirement

A network with instructions is available.

Procedure

To insert a free comment on an instruction, proceed as follows:

1. If necessary, activate the "Free-form comments on/off" button in the toolbar.
2. Right-click on the instruction for which you want to insert a free-form comment.
3. Select the "Insert comment" command in the shortcut menu.
   A comment box with a standard comment opens. The comment box is connected by an arrow to the corresponding instruction.
4. Enter the required comment in the comment box.

See also

Basic information on using free-form comments in LAD (Page 4904)

Editing free-form comments (Page 4905)

Deleting free-form comments (Page 4906)

Editing free-form comments

Introduction

Free-form comments can be edited as follows:

- Changing the comment text
- Changing the position and size of the comment box
- Attaching a comment to another element
- Showing and hiding free comments

Changing the comment text

To change the text of free-form comments, follow these steps:

1. Click on the comment box.
2. Enter the desired text.

Changing the position of the comment box

To change the positioning of the comment box, follow the steps below:

1. Left-click the comment box and keep the mouse button pressed.
2. Drag the comment box to the desired location.

Changing the size of the comment box

To change the size of the comment box, follow the steps below:

1. Click on the comment box.
2. Drag the comment box on the move handle in the lower right corner to the desired size.

Attaching a comment to another element

To attach a free-form comment to another element, follow these steps:

1. Left-click the point of the arrow that links the comment box with the instruction and keep the mouse button pressed.
2. Drag the arrow to the element to which you want to attach the comment. Possible insertion points are marked with a green square.
3. Release the mouse button.

Showing and hiding free comments

To show or hide a free-form comments, follow these steps:

1. Click the "Free-form comment on/off" button in the toolbar.

See also

Basic information on using free-form comments in LAD (Page 4904)

Inserting free-form comments (Page 4905)

Deleting free-form comments (Page 4906)

Deleting free-form comments

Procedure

To delete a free-form comment, proceed as follows:

1. Right-click on the free-form comment that you want to delete.
2. Select the "Delete" command in the shortcut menu.

See also

Basic information on using free-form comments in LAD (Page 4904)

Inserting free-form comments (Page 4905)

Editing free-form comments (Page 4905)

11.12.5 Editing LAD elements

11.12.5.1 Selecting LAD elements

You can select several individual elements or all elements in a network.

Requirement

LAD elements are available

Selecting several individual LAD elements

To select several individual LAD elements, follow these steps:

1. Press and hold down the <Ctrl> key.
2. Click on all the LAD elements you wish to select.
3. Now release the <Ctrl> key.

Selecting all LAD elements in a network

To select all LAD elements in a network, follow these steps:

1. Go to the network whose elements you wish to select.
2. Select the "Select all" command in the "Edit" menu or press <Ctrl A>.

See also

Copying LAD elements (Page 4908)

Cutting LAD elements (Page 4908)

Pasting an LAD element from the clipboard (Page 4909)

Replacing LAD elements (Page 4909)

Inserting additional inputs and outputs in LAD elements (Page 4910)

Removing inputs and outputs (Page 4911)

Deleting LAD elements (Page 4912)

11.12.5.2 Copying LAD elements

Requirement

An LAD element is available.

Procedure

To copy a LAD element, follow these steps:

1. Right-click the LAD element that you want to copy.
2. Select "Copy" in the shortcut menu.

Result

The LAD element will be copied and saved to the clipboard.

See also

Selecting LAD elements (Page 4907)
Cutting LAD elements (Page 4908)
Pasting an LAD element from the clipboard (Page 4909)
Replacing LAD elements (Page 4909)
Inserting additional inputs and outputs in LAD elements (Page 4910)
Removing inputs and outputs (Page 4911)
Deleting LAD elements (Page 4912)

11.12.5.3 Cutting LAD elements

Requirement

An LAD element is available.

Cutting

To cut a LAD element, follow these steps:

1. Right-click the LAD element that you want to cut.
2. Select "Cut" in the shortcut menu.

Result

The LAD element will be cut and saved to the clipboard.

See also

Selecting LAD elements (Page 4907)
Copying LAD elements (Page 4908)
Pasting an LAD element from the clipboard (Page 4909)
Replacing LAD elements (Page 4909)
Inserting additional inputs and outputs in LAD elements (Page 4910)
Removing inputs and outputs (Page 4911)
Deleting LAD elements (Page 4912)

11.12.5.4 Pasting an LAD element from the clipboard

Requirement

An LAD element is available.

Procedure

To paste an LAD element from the clipboard, follow these steps:

1. Copy a LAD element or cut a LAD element.
2. Right-click the point in the network where you want to paste the element.
3. Select "Paste" in the shortcut menu.

See also

Selecting LAD elements (Page 4907)
Copying LAD elements (Page 4908)
Cutting LAD elements (Page 4908)
Replacing LAD elements (Page 4909)
Inserting additional inputs and outputs in LAD elements (Page 4910)
Removing inputs and outputs (Page 4911)
Deleting LAD elements (Page 4912)

11.12.5.5 Replacing LAD elements

You can easily exchange LAD elements with other LAD elements of the same type. This has the advantage that the parameters are retained and need not be entered again. For example, you can exchange normally open contacts and normally closed contacts or RS FlipFlop and SR FlipFlop.

Requirements

A network with at least one LAD element is present.

Procedure

To replace an LAD element with another LAD element, follow these steps:

1. Select the LAD element that you want to replace.
2. Position the cursor over the triangle in the top right-hand corner of the LAD element.
   A drop-down list is displayed.
3. From the drop-down list, select the LAD element that you want to use to replace the existing LAD element.

See also

Selecting LAD elements (Page 4907)

Copying LAD elements (Page 4908)

Cutting LAD elements (Page 4908)

Pasting an LAD element from the clipboard (Page 4909)

Inserting additional inputs and outputs in LAD elements (Page 4910)

Removing inputs and outputs (Page 4911)

Deleting LAD elements (Page 4912)

11.12.5.6 Inserting additional inputs and outputs in LAD elements (S7-1200, S7-1500)

Introduction

You can expand LAD elements which execute commutative arithmetic instructions by adding additional inputs. Such elements are, for example, the instructions "Add" (ADD) and "Multiply" (MUL). You can expand the MOVE and DEMUX instruction boxes by adding additional outputs.

Requirement

An LAD element is available that permits the insertion of additional inputs and outputs.

Inserting an additional input

To add an additional input to the box of a LAD element, follow these steps:

1. Right-click on an existing input of the LAD element.
2. Select "Insert input" in the shortcut menu.
   An additional input is added to the box of the LAD element.

Or:

1. Click on the yellow star symbol beside the last input in the instruction box.
   An additional input is added to the box of the LAD element.

Inserting an additional output

To add an additional output to the box of a LAD element, follow these steps:

1. Right-click on an existing output of the LAD element.
2. Select "Insert output" from the shortcut menu.
   An additional output is added to the box of the LAD element.

Or:

1. Click on the yellow star symbol beside the last input in the instruction box.
   An additional output is added to the box of the LAD element.

See also

Inserting LAD elements (Page 4890)
Selecting LAD elements (Page 4907)
Copying LAD elements (Page 4908)
Cutting LAD elements (Page 4908)
Pasting an LAD element from the clipboard (Page 4909)
Replacing LAD elements (Page 4909)
Removing inputs and outputs (Page 4911)
Deleting LAD elements (Page 4912)

11.12.5.7 Removing inputs and outputs (S7-1200, S7-1500)

Introduction

Inputs and outputs which you have added to an instruction can be removed.

Requirement

An LAD element is available to which you have added additional inputs and outputs.

Remove input

To remove an input, follow these steps:

1. Select the input that you want to remove.
2. Select the "Delete" command in the shortcut menu.
   The input of the LAD element is removed.

Remove output

To remove an output, follow these steps:

1. Select the output that you want to remove.
2. Select the "Delete" command in the shortcut menu.
   The output of the LAD element will be removed.

See also

Inserting LAD elements (Page 4890)

Selecting LAD elements (Page 4907)

Copying LAD elements (Page 4908)

Cutting LAD elements (Page 4908)

Pasting an LAD element from the clipboard (Page 4909)

Replacing LAD elements (Page 4909)

Inserting additional inputs and outputs in LAD elements (Page 4910)

Deleting LAD elements (Page 4912)

11.12.5.8 Deleting LAD elements

Requirement

An LAD element is available.

Procedure

To delete a LAD element, follow these steps:

1. Right-click the LAD element that you want to delete.
2. Select the "Delete" command in the shortcut menu.

See also

Inserting LAD elements (Page 4890)

Selecting LAD elements (Page 4907)

Copying LAD elements (Page 4908)

Cutting LAD elements (Page 4908)

Pasting an LAD element from the clipboard (Page 4909)

Replacing LAD elements (Page 4909)

Inserting additional inputs and outputs in LAD elements (Page 4910)

Removing inputs and outputs (Page 4911)

11.12.6 Inserting operands into LAD instructions

11.12.6.1 Inserting operands

The character strings "<???>", "<??.?>" and "..." are inserted as placeholders for the parameters when an LAD element is inserted. The "<???>" and "<??.?>" strings displayed in red indicate parameters that need to be connected. The "..." string displayed in black indicates parameters that may be connected. "<??.?>" stands for Boolean placeholders. In addition, LAD elements can have the following different I/Os for the parameters:

- Orange I/Os: Only tags or constants can be interconnected at this I/O.
- Black I/Os: Boolean input or output to which other elements can be connected upstream or downstream.

Note

If you position the cursor over the placeholder, the expected data type will be displayed.

Requirement

An LAD element is available.

Procedure

To connect the parameters of a LAD element, follow these steps:

1. Double-click the placeholder of the parameter.
   An entry field opens, and the placeholder is selected.
2. Enter the appropriate parameter.

   Note

   If you enter the absolute address of a parameter that has already been defined, this absolute address will be changed to the symbolic name of the parameter as soon as the input is confirmed. If you have not yet defined the parameter, a new tag with this absolute address and the default name "Tag_<n>" will be entered in the PLC tag table. When you confirm your input, the absolute address will be replaced with the symbolic name "Tag_<n>".

3. Confirm the parameter with the Enter key.
4. If you have not yet defined the parameter, you can define it directly in the program editor using the shortcut menu.
   See also:
   Declaring PLC tags in the program editor (Page 4847)
   Declaring local tags in the program editor (Page 4773)

Or drag from it the PLC tag table:

1. In the project tree, select the "PLC tags" folder or open the PLC tag table.
2. If you have opened the PLC tag table, drag the symbol from the first column of the desired tag to the appropriate place in your program. If you have not opened the PLC tag table yet, open the detail view now. Drag the desired tag from the detail view to the appropriate place in your program.

Or drag from it the block interface:

1. Open the block interface.
2. Drag the required operand from the block interface to the instruction window.

Result

- If the syntax is error-free, the displayed parameter is black. The editor then jumps to the next placeholder.
- If there is an error in the syntax, the cursor stays in the entry field and a corresponding error message is displayed in the status line. If you press the Enter key again, the entry field is closed and the faulty entry is displayed in red italics.

See also

Using autocompletion (Page 4706)

11.12.6.2 Wiring hidden parameters

Introduction

If an instruction contains hidden parameters, the instruction box has a small arrow on the lower edge. You can recognize hidden parameters by their white font.

You can show and wire the hidden parameters at any time.

Showing or hiding hidden parameters

To show or hide hidden parameters, follow these steps:

1. Click on the down arrow at the bottom edge of the instruction box to show hidden parameters.
2. Click on the up arrow at the bottom edge of the instruction box to hide hidden parameters.

Wiring hidden parameters

To wire parameters, follow these steps:

1. Wire the hidden parameters like normally visible parameters.
   Its visibility may change due to the interconnection of the parameter:
   - The parameter is visible if you have selected the "Hide, if no parameter is assigned" option in the properties.
   - The parameter remains hidden if you have selected the "Hide" option in the properties.

See also

Hiding parameters during the block call (Page 4782)

11.12.6.3 Displaying or hiding tag information

Introduction

You can display the following information about the tags to be used in the Program editor:

- Name of the tag
- Tag addresses
- Simple or hierarchical comments to document the tags

The information is taken from the block interface for local tags and DB tags and from the PLC tag table for tags that are valid CPU-wide.

You can display the tag information either for all the blocks or for individually opened blocks. If you display the tag information for all the blocks, the tag information for all the blocks currently opened and opened in future is shown. You can hide the tag information at any time again. If you have hidden the tag information for all blocks, you can display it again for individual blocks that are open.

When you select the display of tag information with hierarchical comments, the comments of the higher structure levels are also displayed for structured tags. The comments are shown in brackets after the tag comment; the comments of the individual levels are separated by a dot. If there is no comment for a tag on a structure level, it is not displayed. This is indicated by two successive dots.

You can display the tag information either collectively below a LAD/FBD network or directly at the respective operand. As with showing and hiding tag information, you can set this via the settings for all block or change specifically for an opened block.

Displaying or hiding tag information for all the blocks

Proceed as follows to display or hide the tag information for all the blocks:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.

3. If you want to display the tag information, select the "Show" option or the "Tag information with hierarchical comments" option in the drop-down list, depending on whether you want to display simple or hierarchical comments.
4. Use the "Position of the tag information (LAD/FBD)" drop-down list to specify whether the tag information is to be displayed below the LAD/FBD network or on the operand.
   For all open blocks, the tag information is shown and displayed at the selected position. If you open additional blocks the tag information for these blocks is also shown.
5. If you want to hide the tag information, select the option "Hide" in the "Tag information" drop-down list.
   The tag information is hidden for all open blocks. If you open additional blocks the tag information for these blocks is also hidden.

Displaying or hiding tag information for an opened block

Proceed as follows to display or hide the tag information for an opened block:

1. If you want to display the tag information, select the "Show tag information" option or the "Tag information with hierarchical comments" option in the "Shows the tag information" drop-down list, depending on whether you want to display simple or hierarchical comments.
2. To define the position of the tag information, click the "Position of the tag information (LAD/FBD)" button in the program editor toolbar. The position changes with each click on the button. Alternatively, you can click the small arrow beside the "Position of the tag information (LAD/FBD)" button to open the drop-down list. Than select the position from the drop-down list.
   The tag information is shown and displayed at the selected position.
3. If you want to hide the tag information, select the option "Hide tag information" in the "Shows the tag information" drop-down list.
   The tag information is hidden.

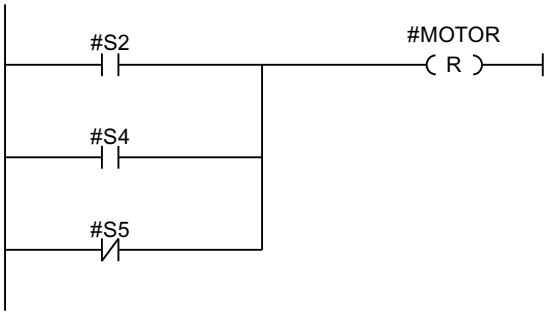
11.12.7 Branches in LAD

11.12.7.1 Basic information on branches in LAD

Definition

You use branches to program parallel circuits with the Ladder Logic (LAD) programming language. Branches are inserted in the main rung. You can insert several contacts into the branch and thus achieve a parallel circuit of series connections. This allows you to program complex ladder logic.

The figure below shows an example of the use of branches:



MOTOR carries signal 1, if one of the following conditions is fulfilled:

- Signal 1 is pending on S2 or S4
- Signal 0 is pending on S5.

See also

Rules for branches in LAD (Page 4917)

Inserting branches into the LAD network (Page 4918)

Closing branches in the LAD network (Page 4918)

Deleting branches in LAD networks (Page 4919)

11.12.7.2 Rules for branches in LAD

Rules

The following rules apply to simultaneous branches:

- A simultaneous branch can only be inserted if the main branch already contains an LAD element.
- Simultaneous branches are opened downwards or are connected directly to the power rail. They are terminated upwards.
- Simultaneous branches are opened after the selected LAD element.
- Simultaneous branches are terminated after the selected LAD element.
- To delete a simultaneous branch, you must delete all LAD elements of this branch. When the last LAD element is removed from the branch, the rest of the branch is also removed.
- You can only insert a coil in a parallel connection when the parallel connection starts directly at the power rail.

See also

Basic information on branches in LAD (Page 4916)

Inserting branches into the LAD network (Page 4918)

Deleting branches in LAD networks (Page 4919)

Closing branches in the LAD network (Page 4918)

11.12.7.3 Inserting branches into the LAD network

You can create several branches in a network.

Requirement

- A network is available.
- The network contains elements.

Procedure

To insert a new branch in a network, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "General > Open branches" in the "Simple instructions" palette.
3. Use drag-and-drop to move the element to the desired place in the network.
   If you want to connect the new branch directly to the power rail, drag the element to the power rail.

See also

Basic information on branches in LAD (Page 4916)

Rules for branches in LAD (Page 4917)

Deleting branches in LAD networks (Page 4919)

11.12.7.4 Closing branches in the LAD network

Branches must be closed again at suitable places. If necessary, branches will be arranged so that they do not cross each other.

Requirement

A branch is available.

Procedure

To close an open branch, follow these steps:

1. Select the open branch.
2. Press and hold down the left mouse button.
   A dashed line will appear as soon as the cursor is moved.

3. Drag the dashed line to a suitable place on the network. Permissible connections are indicated by green lines.
4. Release the left mouse button.

See also

Basic information on branches in LAD (Page 4916)

Rules for branches in LAD (Page 4917)

11.12.7.5 Deleting branches in LAD networks

Requirement

A branch is available.

Procedure

To delete a branch, follow these steps:

1. Select the connection line that links the branch to the main branch.
2. Select the "Delete" command in the shortcut menu.

See also

Basic information on branches in LAD (Page 4916)

Rules for branches in LAD (Page 4917)

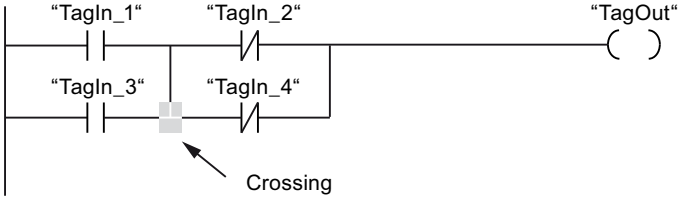
Inserting branches into the LAD network (Page 4918)

11.12.8 Crossings in LAD

11.12.8.1 Basic information on crossings in LAD

Definition

A crossing is a place in a LAD network where one branch is closed and at the same time another branch is opened.



"TagOut" receives signal 1, if the following two conditions are met:

- "TagIn_1" or "TagIn_3" has signal 1
- "TagIn_2" or "TagIn_4" has signal 0

11.12.8.2 Inserting crossings

You can insert crossings in a LAD network by creating connections between the main branch and an additional branch or between different branches.

Requirements

A branch is available.

Procedure

To insert a new crossing in an LAD network, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "General > Open branches" in the "Simple instructions" palette.
3. Drag the element behind the existing branch.
4. Insert any element into the open branch.
5. Click the arrow of the open branch after the inserted element.
6. Hold down the left mouse button and drag the dashed connecting line to the main branch.
7. Release the left mouse button.

See also

Rearranging crossings (Page 4920)

Deleting crossings (Page 4921)

Inserting branches into the LAD network (Page 4918)

11.12.8.3 Rearranging crossings

Requirement

A crossing is available.

Procedure

To rearrange a connection, follow these steps:

1. Select the connection line that defines the crossings in the respective branches.
2. Select the "Delete" command in the shortcut menu.
3. Open the "Instructions" task card.
4. Navigate to "General > Open branches" in the "Simple instructions" palette.
5. Use a drag-and-drop operation to move the element to the place in the network where you want to insert the new crossing.
6. Click on the arrow for the open branch.
7. Hold down the left mouse button and drag the dashed connecting line to the subsidiary branch in which you wish to insert the new crossing.
8. Release the left mouse button.

See also

Inserting crossings (Page 4920)

Deleting crossings (Page 4921)

11.12.8.4 Deleting crossings

Requirement

A crossing is available.

Procedure

To delete a crossing, follow these steps:

1. Select the connection line that defines the crossings in the respective branches.
2. Select the "Delete" command in the shortcut menu.

See also

Inserting crossings (Page 4920)

Rearranging crossings (Page 4920)

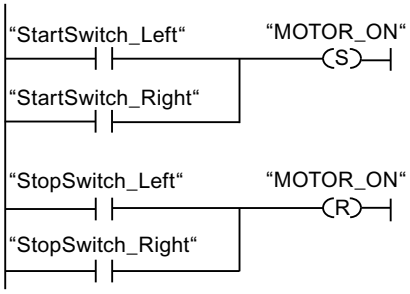
11.12.9 Rungs in LAD

11.12.9.1 Basic information on rungs in LAD

Using rungs

The program is mapped in one or more networks. A network contains a power rail on the left where one or more rungs originate. The binary signal scans are arranged in the form of contacts on the rungs. The serial arrangement of the elements on a rung creates a series connection; arrangement on simultaneous branches creates a parallel connection. A rung is closed by a coil or a box in which the result of logic operation will be written.

The figure below shows an example of the use of several rungs within a network:



Rules

Remember the following rules when using several rungs:

- Connections are not permitted between rungs.
- Only one jump instruction is permissible per network. The positioning rules for jump instructions remain valid.

Running rungs

Rungs and networks are executed from top to bottom and from left to right. This means that the first instruction in the first rung of the first network is processed first. All instructions of this rung are then processed. After this come all other rungs of the first network. The next network is processed only after all rungs have first been run.

Differences between branches and rungs

The difference between branches and rungs is that the rungs are independent branches that can also stand in a different network. Branches, on the other hand, permit the programming of a parallel connection.

See also

Insert rung (Page 4923)

Deleting a rung (Page 4923)

11.12.9.2 Insert rung

Requirement

- A block is open.
- A network is available.

Procedure

To insert a new rung in a network, proceed as follows:

1. Insert any coil on the power rail.
   A new rung will be inserted and the coil positioned at the end of the rung.
2. Insert additional instructions in the new rung.

See also

Basic information on rungs in LAD (Page 4922)

Deleting a rung (Page 4923)

11.12.9.3 Deleting a rung

Requirement

A rung is available.

Procedure

To delete a rung, proceed as follows:

1. Hold down the left mouse button and draw a frame around the rung. At the same time, make sure that you select all instructions. Alternatively, you can hold down the <Shift> key and select the first the last instruction of the rung.
2. Right-click on one of the instructions in the rung.
3. Select the "Delete" command in the shortcut menu.

See also

Basic information on rungs in LAD (Page 4922)

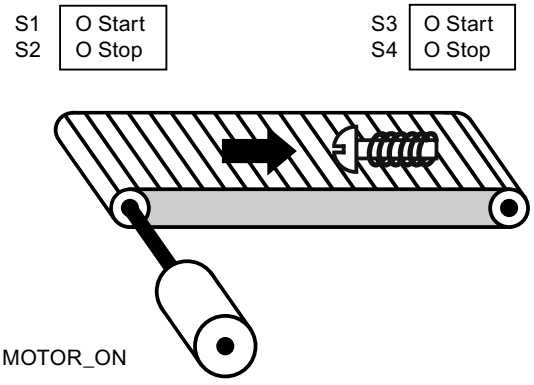
Insert rung (Page 4923)

11.12.10 LAD programming examples

11.12.10.1 Example of controlling a conveyor belt

Controlling a conveyor belt

The following figure shows a conveyor belt that can be activated electrically. There are two pushbuttons at the beginning of the conveyor belt: S1 for START and S2 for STOP. There are also two pushbuttons at the end of the conveyor belt: S3 for START and S4 for STOP. It is possible to start and stop the conveyor belt from either end.



Implementation

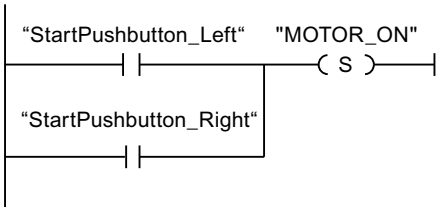
The following table shows the definition of the tags used:

Name	Declaration	Data type	Description
StartPushbutton_Left (S1)	Input	BOOL	Start pushbutton on the left side of the conveyor belt
StopPushbutton_Left (S2)	Input	BOOL	Stop pushbutton on the left side of the conveyor belt
StartPushbutton_Right (S3)	Input	BOOL	Start pushbutton on the right side of the convey-or belt
StopPushbutton_Right (S4)	Input	BOOL	Stop pushbutton on the right side of the convey-or belt
MOTOR_ON	Output	BOOL	Turn on the conveyor belt motor

The following networks show the LAD programming for solving this task:

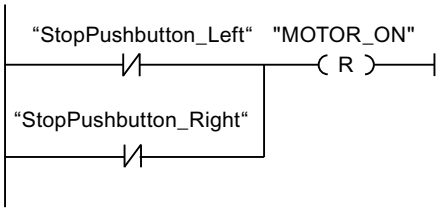
Network 1:

The conveyor belt motor is switched on when start pushbutton "S1" or "S3" is pressed.



Network 2:

The conveyor belt motor is switched off when stop pushbutton "S2" or "S4" is pressed.



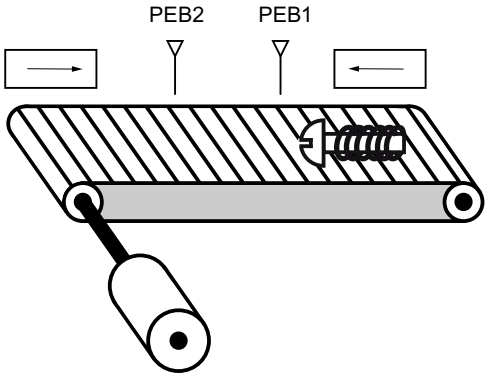
See also

Basic information on LAD (Page 4878)
Settings for LAD (Page 4881)
Working with networks (Page 4882)
Inserting LAD elements (Page 4890)
Editing LAD elements (Page 4907)
Inserting operands into LAD instructions (Page 4913)
Branches in LAD (Page 4916)
Crossings in LAD (Page 4919)
Rungs in LAD (Page 4922)
Example of detecting the direction of a conveyor belt (Page 4926)
Example of detecting the fill level of a storage area (Page 4928)
Example of calculating an equation (Page 4931)
Example of controlling room temperature (Page 4933)

11.12.10.2 Example of detecting the direction of a conveyor belt

Detecting the direction of a conveyor belt

The detected running direction of the belt is indicated by a RIGHT arrow or a LEFT arrow. If additional conveyed material is approaching PEB1 from the right or PEB2 from the left, the displayed arrow is initially switched off until, after both photoelectric barriers are passed, the running direction can be detected again and the corresponding arrow displayed. For the solution of the task, 2 edge memory bits are needed that detect the signal change from "0" to "1" at the two photoelectric barriers.



Implementation

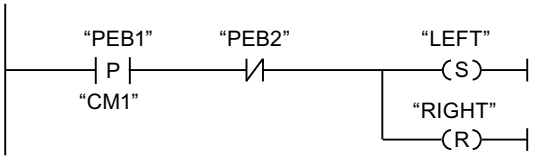
The following table shows the definition of the tags used:

Name	Declaration	Data type	Description
PEB1	Input	BOOL	Photoelectric barrier 1
PEB2	Input	BOOL	Photoelectric barrier 2
RIGHT	Output	BOOL	Display during movement to right
LEFT	Output	BOOL	Display during movement to left
CM1	Input	BOOL	Edge bit memory 1
CM2	Input	BOOL	Edge bit memory 2

The following networks show the LAD programming for solving this task:

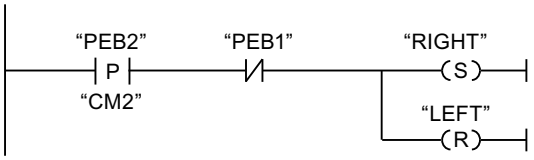
Network 1:

If the signal state changes from "0" to "1" (positive edge) at photoelectric barrier "PEB1" and, at the same time, the signal state at "PEB2" is "0", the object on the belt is moving to the left.



Network 2:

If the signal changes from "0" to "1" (positive edge) at photoelectric barrier "PEB2" and, at the same time, the signal state at "PEB1" is "0", the object on the belt is moving to the right.



See also

Basic information on LAD (Page 4878)

Settings for LAD (Page 4881)

Working with networks (Page 4882)

Inserting LAD elements (Page 4890)

Editing LAD elements (Page 4907)

Inserting operands into LAD instructions (Page 4913)

Branches in LAD (Page 4916)

Crossings in LAD (Page 4919)

Rungs in LAD (Page 4922)

Example of controlling a conveyor belt (Page 4924)

Example of detecting the fill level of a storage area (Page 4928)

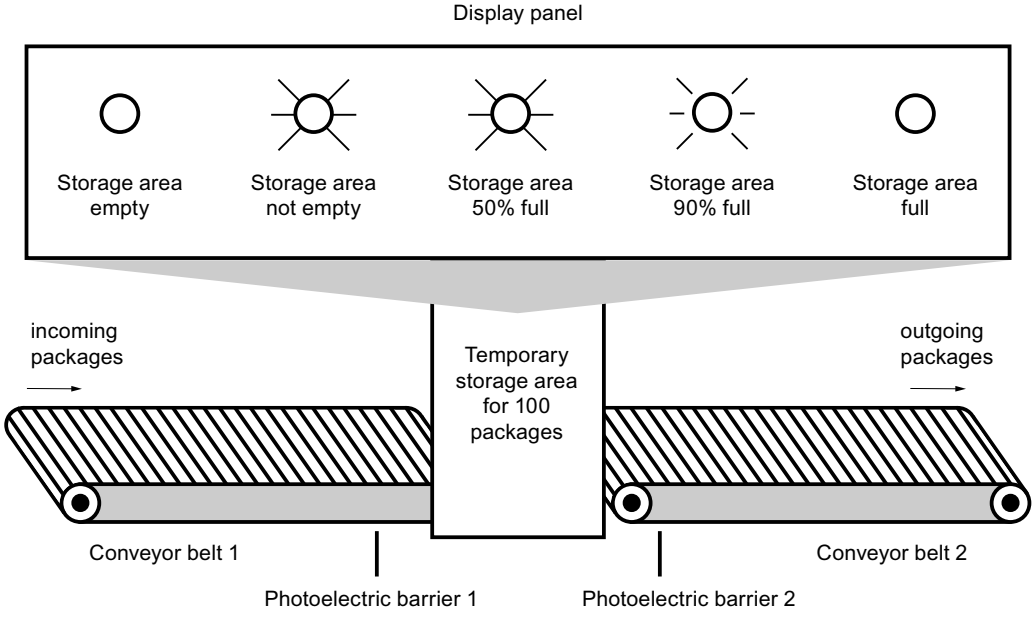
Example of calculating an equation (Page 4931)

Example of controlling room temperature (Page 4933)

11.12.10.3 Example of detecting the fill level of a storage area

Detecting the fill level of a storage area

The following figure shows a system with two conveyor belts and a temporary storage area between them. Conveyor belt 1 delivers packages to the storage area. A photoelectric barrier at the end of conveyor belt 1 near the storage area detects how many packages are delivered to the storage area. Conveyor belt 2 transports packages from the temporary storage area to a loading dock where they are loaded onto trucks. A photoelectric barrier at the storage area exit detects how many packages leave the storage area to be transported to the loading dock. Five display lamps indicate the capacity of the temporary storage area.



Implementation

The following table shows the definition of the tags used:

Name	Declaration	Data type	Description
PEB1	Input	BOOL	Photoelectric barrier 1
PEB2	Input	BOOL	Photoelectric barrier 2
RESET	Input	BOOL	Reset counter
LOAD	Input	BOOL	Adjust the current counter value to the value of the PV parameter.
MAX STORAGE AREA FILL AMOUNT	Input	INT	Maximum possible number of packages in the storage area
PACKAGECOUNT	Output	INT	Number of packages in the storage area (current count value)

Name	Declaration	Data type	Description
STOCK_PACKAGES	Output	BOOL	Is set when the current count value is greater than or equal to the value of the "MAX STORAGE AREA FILL AMOUNT" tag.
STOR_EMPTY	Output	BOOL	Display lamp: Storage area empty
STOR_NOT_EMPTY	Output	BOOL	Display lamp: Storage area not empty
STOR_50%_FULL	Output	BOOL	Display lamp: Storage area 50 % full
STOR_90%_FULL	Output	BOOL	Display lamp: Storage area 90 % full
STOR_FULL	Output	BOOL	Display lamp: Storage area full
VOLUME_50	Input	INT	Comparison value: 50 packages
VOLUME_90	Input	INT	Comparison value: 90 packages
VOLUME_100	Input	INT	Comparison value: 100 packages

The following networks show the LAD programming for activating the lamps:

Network 1:

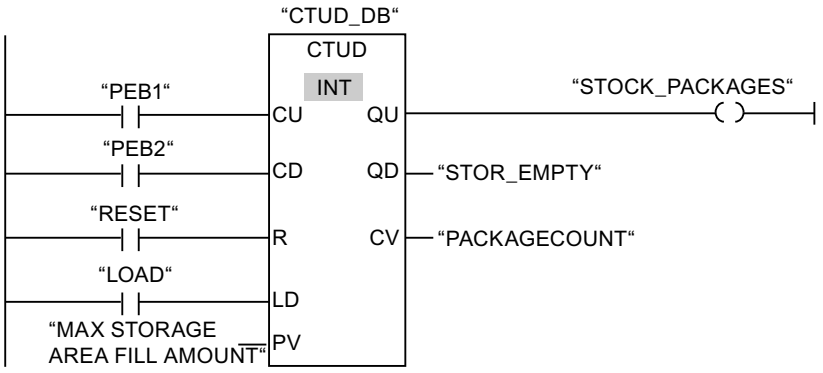
When a package is delivered to the storage area, the signal state at "PEB1" switches from "0" to "1" (positive signal edge). On a positive signal edge at "PEB1", the "Up" counter is enabled, and the current count value of "PACKAGECOUNT" is increased by one.

When a package is delivered from the storage area to the loading dock, the signal state at "PEB2" switches from "0" to "1" (positive signal edge). On a positive signal edge at "PEB2", the "Down" counter is enabled, and the current count value of "PACKAGECOUNT" is decreased by one.

If there are no packages in the storage area ("PACKAGECOUNT" = "0"), the "STOR_EMPTY" tag is set to signal state "1", and the "Storage area empty" lamp is switched on.

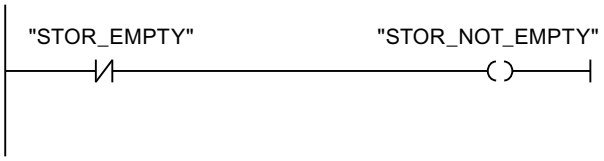
The current count value can be reset to "0" if the "RESET" tag is set to signal state "1".

If the "LOAD" tag is set to signal state "1", the current count value is set to the value of the "MAX STORAGE AREA FILL AMOUNT" tag. As long as the current count value is greater than or equal to the value of the "MAX STORAGE AREA FILL AMOUNT" tag, the "STOCK_PACKAGES" tag supplies the signal state "1".



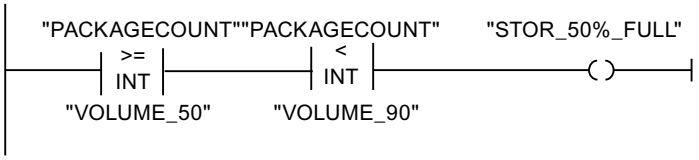
Network 2:

As long as there are packages in the storage area, the "STOR_NOT_EMPTY" tag is set to signal state "1", and the "Storage area not empty" lamp is switched on.



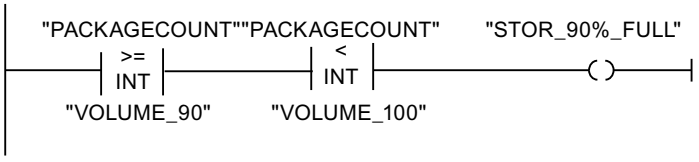
Network 3:

If the number of packages in the storage area is greater than or equal to 50, the "Storage area 50 % full" lamp switches on.



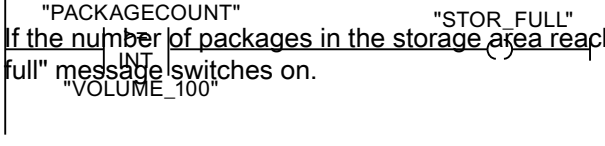
Network 4:

If the number of packages in the storage area is greater than or equal to 90, the "Storage area 90% full" lamp switches on.



Network 5:

If the number of packages in the storage area reaches 100, the lamp for the "Storage area full" message switches on.



See also

Basic information on LAD (Page 4878)
Settings for LAD (Page 4881)
Working with networks (Page 4882)
Inserting LAD elements (Page 4890)
Editing LAD elements (Page 4907)
Inserting operands into LAD instructions (Page 4913)
Branches in LAD (Page 4916)
Crossings in LAD (Page 4919)
Rungs in LAD (Page 4922)
Example of controlling a conveyor belt (Page 4924)
Example of detecting the direction of a conveyor belt (Page 4926)
Example of calculating an equation (Page 4931)
Example of controlling room temperature (Page 4933)

11.12.10.4 Example of calculating an equation

Calculating an equation

The following program example shows you how to solve equations of the following type with the LAD programming language:

X= ((A + B) x C) / D

You can always solve this equation with mathematical operations. To do so, you can use the instructions "ADD", "MUL" and "DIV". If you are using a CPU of the S7-1200 or S7-1500 series, you can also use the instruction "CALCULATE". Depending on the data type you choose, different mathematical operations are available for this instruction which can be combined with one another.

Implementation with the instructions "ADD", "MUL" and "DIV"

You want to calculate the following equation:

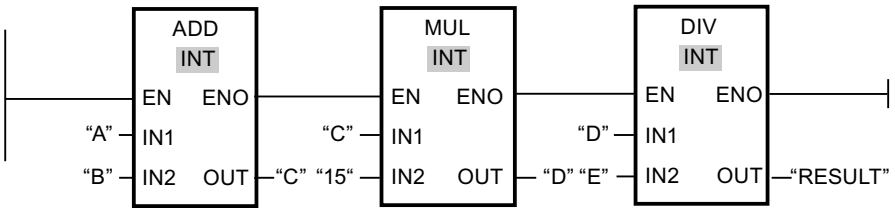
RESULT = ((A + B) x 15) / E

The following table shows the definition of the tags used:

Name	Data type	Comment
A	INT	First value for addition
B	INT	Second value for addition
C	INT	First intermediate result
15	INT	Multiplier
D	INT	Second intermediate result

Name	Data type	Comment
E	INT	Divisor
RESULT	INT	End result

The following network shows the LAD programming for calculating the equation:



The value of operand "A" is added to the value of operand "B". The sum is stored in operand "C". The value of operand "C" is multiplied by "15". The multiplication result is stored in operand "D". The value stored in operand "D" is then divided by the value of operand "E". The end result is saved in the "RESULT" operand.

Implementation with the instruction "CALCULATE" (S7-1200/1500 only)

You want to calculate the following equation:

RESULT = ((5 + 10) x 4) / 6

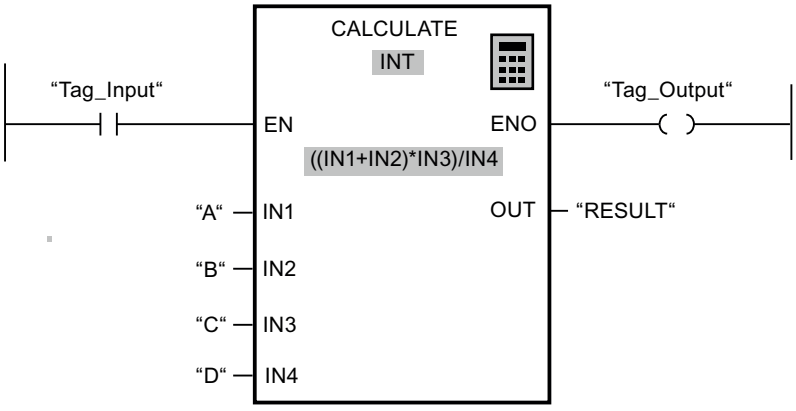
The following table shows the definition of the tags used:

Name	Declaration	Data type	Comment	Value
A	Input	INT	First value for addition	5
B	Input	INT	Second value for addition	10
C	Input	INT	Multiplier	4
D	Input	INT	Divisor	6
RESULT	Output	INT	End result	10

To program the equation with the instruction "CALCULATE", follow these steps:

1. Drag the instruction "CALCULATE" from the "Instructions" task card to an LAD network.
2. You can select the data type INT for the instruction from the "<???>" drop-down list.
3. Interconnect the tags declared in the block interface with the inputs and outputs of the instruction box.
4. Click the "Calculator" icon on the top right corner of the instruction box to enter the equation to be calculated.
   The dialog "Edit 'Calculate' instruction" opens.
5. Enter the following expression in the "OUT:=" field:
   ((IN1 + IN2) * IN4) / IN3
   The equation is displayed in the instruction box.

The following network shows the result in the LAD programming language:



If the "Tag_Input" has signal state "1", the instruction is executed. The value of operand "A" is added to the value of operand "B". The subtotal is multiplied by "C" and then divided by the value of the "D" operand. The end result is saved in the "RESULT" operand.

See also

Basic information on LAD (Page 4878)

Settings for LAD (Page 4881)

Working with networks (Page 4882)

Inserting LAD elements (Page 4890)

Editing LAD elements (Page 4907)

Inserting operands into LAD instructions (Page 4913)

Branches in LAD (Page 4916)

Crossings in LAD (Page 4919)

Rungs in LAD (Page 4922)

Example of controlling a conveyor belt (Page 4924)

Example of detecting the direction of a conveyor belt (Page 4926)

Example of detecting the fill level of a storage area (Page 4928)

Example of controlling room temperature (Page 4933)

11.12.10.5 Example of controlling room temperature

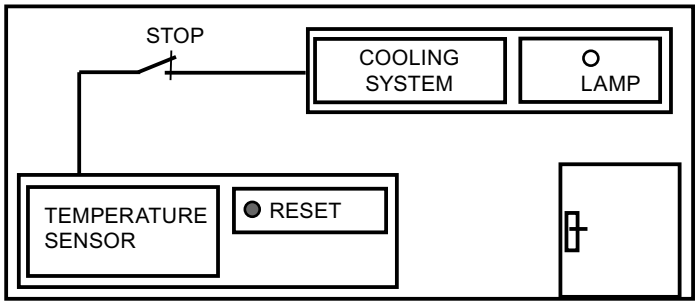
Controlling room temperature

In a cold room, the temperature must be maintained below zero degrees Celsius. A sensor is used to check for any temperature fluctuations. If the temperature rises above zero degrees Celsius, the cooling system switches on for a preset time. The "Cooling system on" lamp is lit during this time.

The cooling system and the lamp are turned off if one of the following conditions is met:

- The sensor reports a temperature fall below zero degrees Celsius.
- The preset cooling time has elapsed.
- The "STOP" pushbutton is pressed.

If the preset cooling time has expired and the temperature in the cold room is still too high, the cooling system can be restarted with the "RESET" pushbutton.



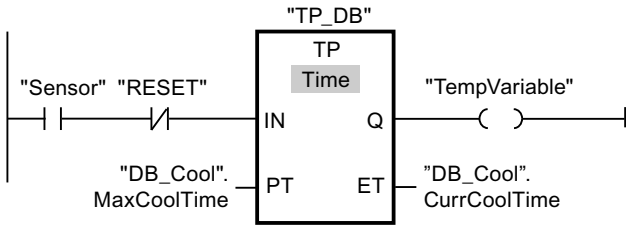
Implementation

The following table shows the definition of the tags used:

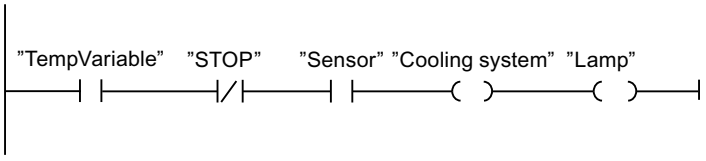
Name	Declaration	Data type	Comment
Sensor	Input	BOOL	Temperature sensor signal
RESET	Input	BOOL	Restart
STOP	Input	BOOL	The cooling system is switched off.
MaxCoolTime	-	TIME	Preset cooling time This tag is defined in the "DB_Cool" data block.
CurrCoolTime	-	TIME	Currently elapsed cooling time This tag is defined in the "DB_Cool" data block.
Cooling system	Output	BOOL	The cooling system is switched on.
Lamp	Output	BOOL	The lamp for the "Cooling system on" message is switched on.
TempVariable	Temp	BOOL	Temporary tag This tag stores the signal state of the IEC timer TP.

The following network shows the LAD programming for controlling room temperature:

Network 1:



Network 2:



When the temperature in the cold room rises above zero degrees Celsius, the signal state at the "Sensor" operand switches from "0" to "1" (positive signal edge). In the case of a positive signal edge at the IN input of the timer function, the preset cooling time is started and the "TempVariable" receives the signal state "1". The signal state "1" of the "TempVariable" causes the cooling system as well as the display lamp to be turned on in network 2. The "Sensor", "Cooling system" and "Lamp" outputs must be programmed in network 2, because program only one coil can be programmed at the Q output of the timer function.

If the temperature in the cold room falls below zero degrees Celsius, the signal state of the sensor switches back to "0". This switches the cooling system and lamp off.

If the sensor does not signal a temperature drop, the cooling system and lamp are switched off after the preset cooling time has elapsed, at the latest. In this case, the cooling process can be restarted with the "RESET" pushbutton. Pressing and releasing the pushbutton generates a new positive signal edge at the IN input that restarts the cooling system.

The cooling system and the display lamp can be turned off with the "STOP" pushbutton at any time.

See also

Basic information on LAD (Page 4878)

Settings for LAD (Page 4881)

Working with networks (Page 4882)

Inserting LAD elements (Page 4890)

Editing LAD elements (Page 4907)

Inserting operands into LAD instructions (Page 4913)

Branches in LAD (Page 4916)

Crossings in LAD (Page 4919)

Rungs in LAD (Page 4922)

Example of controlling a conveyor belt (Page 4924)

Example of detecting the direction of a conveyor belt (Page 4926)

Example of detecting the fill level of a storage area (Page 4928)
Example of calculating an equation (Page 4931)

11.13 Creating FBD programs

11.13.1 Basic information on FBD

11.13.1.1 FBD programming language

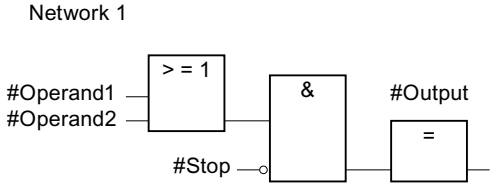
Overview of the Function Block Diagram (FBD) programming language

FBD is a graphical programming language. The representation is based on electronic circuit systems.

The program is mapped in one or more networks. A network contains one or more logic operation paths. The binary signal scans are linked by boxes. The representation of the logic is based on the graphical logic symbols used in Boolean algebra.

Example of networks in FBD

The following figure shows an FBD network with AND and OR boxes and an assignment:



See also

Working with networks (Page 4940)

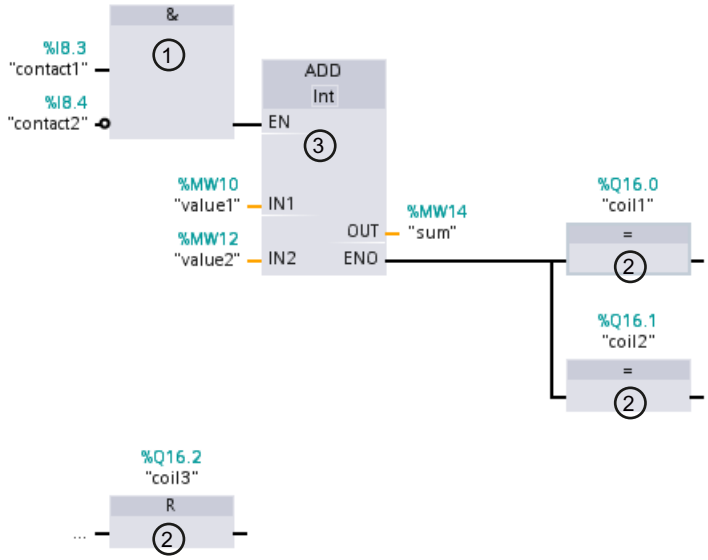
11.13.1.2 Overview of the FBD elements

FBD elements

An FBD program consists of separate elements that are linked by means of a binary signal flow. Most program elements must be supplied with tags.

A FBD network is programmed from left to right.

For example, the following figure shows elements of an FBD network:



1) Binary function

2) Standard box

3) Complex box

Binary functions

You can use binary functions to query binary operands and to combine their signal states. The following operations are examples of binary functions: "AND operation", "OR operation" and "EXCLUSIVE OR operation".

Standard boxes:

You can use standard boxes to control binary operands, perform RLO edge detection or execute jump functions in the program. Standard boxes generally have only one single input.

Complex boxes

Complex boxes represent program elements with complex functions. The empty box is an exception. You can use the empty box as a placeholder in which you can select the required instruction.

The following types of boxes are available to you in an FBD program:

- Complex boxes without EN/ENO mechanism:
  A box is executed independently of the signal state at the box inputs. The error status of the processing cannot be queried.
- Complex boxes with EN/ENO mechanism:
  A box is only executed if the enable input "EN" has the signal state "1". If the box is processed correctly, the "ENO" enable output has signal state "1". If an error occurs during processing, the "ENO" output is reset.
  If the EN enable input is not interconnected, the box is always executed.

Calls of code block are also shown in the network as complex boxes with EN/ENO mechanism.

11.13.2 Settings for FBD

11.13.2.1 Overview of the settings for FBD

Overview

The following table shows the settings that you can make:

Group	Setting	Description
Font	Font size	Font size in program editor
View	Layout	Compact or wide Changes the vertical spacing between operands and other objects (such as operand and contact). The change becomes visible once the block is reopened.
	With absolute information	Additional display of the absolute addresses
Operand field	Maximum width	Maximum number of characters that can be entered horizontally in the operand field. This setting recalculates the layout of the networks.
	Maximum height	Maximum number of characters that can be entered vertically in the operand field. This setting recalculates the layout of the networks.

See also

Changing the settings (Page 4940)

11.13.2.2 Changing the settings

Procedure

To change the settings, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.
3. Change the settings.

Result

The change will be loaded directly, there is no need to save it explicitly.

See also

Overview of the settings for FBD (Page 4939)

11.13.3 Working with networks

11.13.3.1 Using networks

Function

The user program is created in the block within networks. For a code block to be programmed, it must contain at least one network. To achieve a better overview of the user program, you can also subdivide your program into several networks.

You have the option to insert networks for the textual programming languages SCL and STL in LAD and FBD blocks, and then use instructions in these programming languages. Depending on the CPU you are using, you can insert networks for the following programming languages:

- S7-300/400: STL networks
- S7-1200: SCL networks
- S7-1500: STL and SCL networks

The programming editor always adapts to the network that currently has the focus. This means that you have access to the SCL instructions as well as the SCL functions when programming an SCL network.

Note

You cannot access the blocks that contain SCL networks via the Openness interface.

EN/ENO mechanism in blocks with different network languages

You can also use the EN/ENO mechanism in blocks with different network languages. Each programming language displays the error status differently:

- SCL has an "ENO" tag that stores the error status and can be queried. Direct access to this tag is only possible with SCL.
- LAD/FBD/STL have no special tag for "ENO". However, you can read the error status for STL from the BR bit, and query it via the RET coil for LAD/FBD.

The following rules apply to reading out the error status for the entire block:

- The last network in the block is a LAD/FBD network:
  If you do not use RET coils, the error status is "TRUE" by default.
- The last network in the block is an STL network:
  The BR bit determines the error status. The BR bit can be edited in STL networks using the BR tab.
- The last network in the block is an SCL network:
  The "ENO" tag determines the error status of the block.

See also

Inserting network title (Page 4944)

Entering a network comment (Page 4946)

Navigating networks (Page 4947)

11.13.3.2 Inserting networks

You can either insert a network of the same programming language or a network for a textual programming language in an LAD or FBD block. Keep in mind, however, that not all CPU types support all textual languages.

Requirement

An LAD or FBD block is open.

Insert LAD or FBD network

To insert a new network, follow these steps:

1. Select the network after which you want to insert a new network.
2. Select the "Insert network" command in the shortcut menu.

Insert STL or SCL network

To insert a new network, follow these steps:

1. Select the network after which you want to insert a new network.
2. Select the "Insert STL network" or "Insert SCL network" command in the shortcut menu.

Result

A new, empty network is inserted in the block in the corresponding programming language.

See also

Inserting network title (Page 4944)

Entering a network comment (Page 4946)

Navigating networks (Page 4947)

11.13.3.3 Selecting networks

Requirements

A network is available.

Selecting a network

To select a network, follow these steps:

1. Click the title bar of the network that you want to select.

Selecting several networks

Proceed as follows to select several individual networks:

1. Press and hold down the <Ctrl> key.
2. Click all the networks that you want to select.

To select several successive networks, follow these steps:

1. Press and hold down the <Shift> key.
2. Click the first network that you want to select.
3. Click the last network that you want to select.
   The first and last networks and all those in between are selected.

See also

Using networks (Page 4940)

Inserting networks (Page 4941)

Inserting network title (Page 4944)

Entering a network comment (Page 4946)

Navigating networks (Page 4947)

11.13.3.4 Copying and pasting networks

Copied networks can be pasted within the block or in another block. Networks that were created in LAD or FBD can also be inserted in blocks of the respective other programming language.

Requirement

A network is available.

Procedure

To copy and paste a network, follow these steps:

1. Select the network or networks to be copied.
2. Select "Copy" in the shortcut menu.
3. Select the network after which you want to paste in the copied network.
4. Select "Paste" in the shortcut menu.

See also

Using networks (Page 4940)
Inserting networks (Page 4941)
Selecting networks (Page 4942)
Inserting network title (Page 4944)
Entering a network comment (Page 4946)
Navigating networks (Page 4947)

11.13.3.5 Deleting networks

Requirement

A network is available.

Procedure

To delete a network, follow these steps:

1. Select the network that you want to delete.
2. Select the "Delete" command in the shortcut menu.

See also

Using networks (Page 4940)
Inserting networks (Page 4941)

Selecting networks (Page 4942)
Copying and pasting networks (Page 4943)
Inserting network title (Page 4944)
Entering a network comment (Page 4946)
Navigating networks (Page 4947)

11.13.3.6 Expanding and collapsing networks

Requirements

A network is available.

Opening and closing a network

To open a network, follow these steps:

1. Click on the right arrow in the network title bar.

To close a network, follow these steps:

1. Click on the down arrow in the network title bar.

Opening and closing all networks

To open and close all networks, follow these steps:

1. In the toolbar, click "Open all networks" or "Close all networks".

See also

Using networks (Page 4940)
Inserting networks (Page 4941)
Selecting networks (Page 4942)
Copying and pasting networks (Page 4943)
Deleting networks (Page 4943)
Inserting network title (Page 4944)
Entering a network comment (Page 4946)
Navigating networks (Page 4947)

11.13.3.7 Inserting network title

The network title is the header of a network. The length of the network title is limited to one line. You can enter the title manually or set it automatically. When you set it automatically, you can do this for individual networks or use the settings to specify that the network title is always set automatically.

For automatic insertion of the network title, the comment of the operand in one of the following instructions in the network is evaluated:

- Assignment
- Set output
- Reset output

The instruction that is listed first in the network is used.

The network title is only inserted automatically when the following conditions are fulfilled:

- The network does not have a title yet.
- The operand of the instruction used for the comment has a comment.

Note

Note the following restrictions for automatic insertion of the network title:

- The network title is not adapted if you change the comment of the operand at a later time.
- The network title is not adapted if you change the operand of the instruction.
- The network title is only set by the writing instructions listed above.
- If the operand is of the data type array, the comment of the array is used and not the comments of the array elements.
- Comments of invalid operands are not taken into consideration.

Entering the network title manually

To enter a network title, follow these steps:

1. Click on the title bar of the network.
2. Enter the network title.

Setting the network title automatically

To specify that the network titles are always set automatically, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.
3. Select the "Set network title automatically" check box in the "Additional settings" group.
   The network titles are set automatically as of this time if the conditions listed above are fulfilled.

To set an individual network title automatically, follow these steps:

1. Right-click "Network <Number of the network>" in the title bar of a network.
2. Select the "Set network title automatically" command in the shortcut menu.
   The title of the selected network is set based on the comment of the operand if the conditions listed above are fulfilled.

See also

Using networks (Page 4940)
Inserting networks (Page 4941)
Selecting networks (Page 4942)
Copying and pasting networks (Page 4943)
Deleting networks (Page 4943)
Expanding and collapsing networks (Page 4944)
Entering a network comment (Page 4946)
Navigating networks (Page 4947)

11.13.3.8 Entering a network comment

You can use network comments to provide comments on the program contents of individual networks. For example, you can indicate the function of the network or draw attention to special characteristics.

Requirement

A network is available.

Procedure

To enter a network comment, follow these steps:

1. Click on the right arrow before the network title.
2. If the comment area is not visible, click "Network comments on/off" in the toolbar.
   The comment area is displayed.
3. Click "Comment" in the comment area.
   The "Comment" text passage is selected.
4. Enter the network comment.

See also

Using networks (Page 4940)
Inserting networks (Page 4941)
Selecting networks (Page 4942)
Copying and pasting networks (Page 4943)
Deleting networks (Page 4943)
Expanding and collapsing networks (Page 4944)

Inserting network title (Page 4944)

Navigating networks (Page 4947)

11.13.3.9 Navigating networks

You can navigate straight to a specific position within a block.

Procedure

To navigate to a specific position within a block, follow these steps:

1. Right-click in the code area of the programming window.
2. Select the "Go to > Network/line" command in the shortcut menu.
   The "Go to" dialog will open.
3. Enter the network to which you want to navigate.
4. Enter the line number of the network to which you want to navigate.
5. Confirm your entry with "OK".

Result

The relevant line will be displayed if this is possible. If the network or line requested does not exist, the last existing network or the last existing line in the network requested will be displayed.

See also

Using networks (Page 4940)

Inserting networks (Page 4941)

Selecting networks (Page 4942)

Copying and pasting networks (Page 4943)

Deleting networks (Page 4943)

Expanding and collapsing networks (Page 4944)

Inserting network title (Page 4944)

Entering a network comment (Page 4946)

11.13.4 Inserting FBD elements

11.13.4.1 Rules for the use of FBD elements

Rules

Note the following rules when inserting FBD elements:

- An FBD network can consist of several elements. All elements of a logic path must be linked to each other according to IEC 61131-3.
- Standard boxes (flip flops, counters, timers, math operations, etc.) can be added as output to boxes with binary logic operations (for example, AND, OR). Comparison boxes are excluded from this rule.
- Only Boolean inputs in an instruction can be combined with preceding logic operations.
- Only the bottom Boolean output in an instruction can be combined with an additional logic operation.
- Enable input EN or enable output ENO can be connected to boxes, but this is not mandatory.
- Constants (for example, TRUE or FALSE) cannot be assigned to binary logic operations. Instead, use tags of the BOOL data type.
- Only one jump instruction can be inserted in each network.
- Only one jump label can be inserted in each network.
- Instructions for positive or negative RLO edge detection may not be arranged right at the left of the network as this requires a prior logic operation.

Placement rules for S7-1200/1500 CPUs

The following table sets out the instructions that can only be positioned at the end of the network:

Instruction		Preceding logic operation required
Mnemonics	Name	
SET_BF	Set bit field	No
RESET_BF	Reset bit field	No
JMP	Jump if RLO = 1	No
JMPN	Jump if RLO = 0	Yes
JMP_LIST	Define jump list	No
SWITCH	Jump distributor	No
RET	Return	No

Placement rules for S7-300/400 CPUs

The following table sets out the instructions that can only be positioned at the end of the network:

Instruction		Preceding logic operation required
Mnemonics	Name	
S	Set output	Yes
R	Reset output	Yes
SP	Start pulse timer	Yes
SE	Start extended pulse timer	Yes
SD	Start on-delay timer	Yes
SS	Start retentive on-delay timer	Yes
SF	Start off-delay timer	Yes
SC	Set counter value	Yes
CU	Count up	Yes
CD	Count down	Yes
JMP	Jump if RLO = 1	No
JMPN	Jump if RLO = 0	Yes
RET	Return	No
OPN	Open global data block	No
OPNI	Open instance data block	No
CALL	Call block	No
SAVE	Save RLO in BR bit	No
MCRA	Enable MCR range	No
MCRD	Disable MCR range	No
MCR<	Open MCR ranges	No
MCR>	Close MCR ranges	No

Note

Calling blocks with binary inputs in S7-300/400 CPUs

The following FAQ describes the special features when calling blocks with binary inputs:

FAQ 13988019: What should you watch out for when programming nested FB/FC calls with binary inputs in FBD? (https://support.industry.siemens.com/cs/de/de/view/13988019)

11.13.4.2 Inserting FBD elements using the "Instructions" task card

Requirement

A network is available.

Procedure

To insert FBD elements into a network using the "Instructions" task card, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to the FBD element that you want to insert.
3. Use drag-and-drop to move the element to the desired place in the network.
   If the element is an internal system function block (FB), the "Call options" dialog opens. In this dialog, you can determine if the data of the inserted element is to be stored in a single, multi or parameter instance. If you select "Single instance", you can find the new instance data block once it is created in the "Program resources" folder of the project tree under "Program blocks > System blocks". You can find multi-instances in the block interface in the "Static" section and parameter instances in the "InOut" section.

Or:

1. Select the point in the network at which you want to insert the element.
2. Open the "Instructions" task card.
3. Double-click on the element you want to insert.
   If the element is an internal system function block (FB), the "Call options" dialog opens. In this dialog, you can determine if the data of the inserted element is to be stored in a single, multi or parameter instance. If you select "Single instance", you can find the new instance data block once it is created in the "Program resources" folder of the project tree under "Program blocks > System blocks". You can find multi-instances in the block interface in the "Static" section and parameter instances in the "InOut" section.

Result

The selected FBD element is inserted with dummy entries for the parameters.

See also

Rules for the use of FBD elements (Page 4948)

11.13.4.3 Inserting FBD elements using an empty box

Requirement

A network is available.

Procedure

To insert FBD elements into a network using an empty box, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "General > Empty box" in the "Basic instructions" palette.
3. Use a drag-and-drop operation to move the "Empty box" element to the desired place in the network.

4. Position the cursor over the triangle in the top right-hand corner of the empty box.
   A drop-down list is displayed.
5. Select the desired FBD element from the drop-down list.
   If the element is an internal system function block (FB), the "Call options" dialog opens. In this dialog, you can determine if the data of the inserted element is to be stored in a single, multi or parameter instance. If you select "Single instance", you can find the new instance data block once it is created in the "Program resources" folder of the project tree under "Program blocks > System blocks". You can find multi-instances in the block interface in the "Static" section and parameter instances in the "InOut" section.

Result

The empty box is changed to the respective FBD element. Placeholders are inserted for the parameters.

11.13.4.4 Selecting the data type of an FBD element

Selecting a data type

Introduction

Some instructions can be executed with several different data types. If you use one of these instructions in the program, you have to specify a valid data type for the instruction at the specific point in the program. For some instructions, you have to select the data types for the inputs and outputs separately.

Note

The valid data type (BOOL) for the tags on the enable input EN and the enable output ENO is predefined by the system and cannot be changed.

The valid data types for an instruction are listed in the instruction drop-down list. You specify the data type of the instruction by selecting an entry from the drop-down list. If the data type of an operand differs from the data type of the instruction and cannot be converted implicitly, the operand is displayed in red and a rollout with the corresponding error message appears.

Data type selection of mathematical instructions

Some mathematical instructions provide you with the option of having the data type automatically set corresponding to the data types of the operand. In the drop-down list for data type selection, these instructions have the entry "Auto" in addition to the actual data types. If you select this entry and then allocate the first operand, the data type of the operand is selected as data type for the instruction. The entry in the drop-down list changes to "Auto (<Data type>)", e.g. "Auto (Real)". If you allocate additional operands, the automatically set data type of the instruction is adjusted according to the following criteria:

- You supply all other operands with tags of the same data type:
  The data type of the instruction is not changed.
- You supply all other operands with tags whose data type is smaller than the data type of the instruction:
  The data type of the instruction is not changed. For the operand with the smaller data type, an implicit conversion is conducted if necessary.
- You supply an additional operand with a tag whose data type is greater than the data type of the instruction:
  The data type of the instruction is changed to the larger data type. An implicit conversion is performed, if necessary, for operands that deviate from the newly set data type of the instruction.

Each change in the data type of an operand can result in a change of the data type of the instruction. Other operands may possibly be implicitly converted as a result. Operands for which an implicit conversion is performed are marked with a gray square.

Note

Please also observe the information on data type conversion for your device and, in particular, the notes on the IEC check.

See also: Data type conversion (Page 2465)

See also

Defining the data type of an instruction (Page 4952)

Defining the data type of an instruction

Introduction

Some instructions can be executed with several different data types. When you insert such instructions into your program, you must specify the data type for these instructions at the actual point in the program.

Specifying the data type by means of the drop-down list

To define the data type of an instruction using the drop-down list, follow these steps:

1. Insert the instruction at the required point in the program using drag-and-drop.
   The entry "???" (undefined) is displayed in the drop-down list of the inserted instruction.
2. Click the triangle in the upper corner of the drop-down list.
   The drop-down list will open to display the data types valid for the instruction.
3. Select a data type from the drop-down list.
   The selected data type is displayed.
4. If the instruction has two drop-down lists, select the data type for the instruction inputs in the left-hand drop-down list and the data type for the instruction outputs in the right-hand drop-down list.

Specifying data type by assigning tags

To define the data type of an instruction by assigning tags, follow these steps:

1. Insert the instruction at the required point in the program using drag-and-drop.
   The entry "???" (undefined) is displayed in the drop-down list of the inserted instruction.
2. At an input or output, specify a valid tag, the data type of which is to be applied as the instruction data type.
   The data type of the tag is displayed in the drop-down list.
3. Enter a valid tag at an input and a valid tag at an output if data types need to be defined for both the inputs and outputs of the instruction. The tag specified at the input determines the data type of the inputs; the tag specified at the output determines the data type of the outputs of the instruction.

Automatically specifying the data type of mathematical instructions

To automatically specify the data type for mathematical instructions, follow these steps:

1. Insert the mathematical instruction at the required point in the program using drag-and-drop.
   The entry "???" (undefined) is displayed in the drop-down list of the inserted instruction.
2. Select the "Auto" entry from the drop-down list.
3. Enter a valid tag at an input or output.
   The data type of the tag is applied as data type of the instruction. The entry in the drop-down list changes to "Auto (<Data type>)".

See also: Selecting a data type (Page 4951)

See also

Selecting a data type (Page 4951)

11.13.4.5 Using favorites in FBD

Adding FBD elements to Favorites

Requirement

- A block is open.
- The multipane mode is set for the "Instructions" task card or the Favorites are also displayed in the editor.

Procedure

To add SCL instructions to the Favorites, follow these steps:

1. Open the "Instructions" task card.
2. Maximize the "Basic instructions" pane.
3. Navigate in the "Basic instructions" pane to the instruction that you want to add to the Favorites.
4. Drag-and-drop the instruction into the "Favorites" pane or into the Favorites area in the program editor.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Removing FBD elements from Favorites (Page 4955)

Overview of the program editor (Page 4684)

Inserting FBD elements using favorites

Requirement

- A block is open.
- Favorites are available.

Procedure

To insert an instruction into a program using Favorites, follow these steps:

1. Drag-and-drop the desired instruction from Favorites to the desired position.

Or:

1. Select the position in the program where you want to insert the instruction.
2. In the Favorites, click on the instruction you want to insert.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Overview of the program editor (Page 4684)

Removing FBD elements from Favorites (Page 4955)

Removing FBD elements from Favorites

Requirement

A code block is open.

Procedure

To remove instructions from Favorites, follow these steps:

1. Right-click on the instruction you want to remove.
2. Select the "Remove instruction" command in the shortcut menu.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Adding FBD elements to Favorites (Page 4954)

Inserting FBD elements using favorites (Page 4954)

Overview of the program editor (Page 4684)

11.13.4.6 Inserting block calls in FBD

Inserting block calls using a drag-and-drop operation

You can insert calls for existing functions (FC) and function blocks (FB) using a drag-and-drop operation from the project tree. If you call function blocks from other function blocks, you can either call them as a single instance, multi-instance or parameter instance.

See also: AUTOHOTSPOT

Requirement

- A network is available.
- The block that is to be called is available.

Inserting a call of a function (FC)

To insert a call of a function (FC) into a network using a drag-and-drop operation, follow these steps:

1. Drag the function from the project tree to the required network.

Inserting a call for a function block (FB)

To insert a call for a function block (FB), follow these steps:

1. Drag the function block from the project tree to the required network.
   The "Call options" dialog opens.
2. In the dialog, enter whether you want to call the block as a single, multi or parameter instance.
   - If you click on the "Single instance" button, you will have to enter a name in the "Name" text box for the data block that you want to assign to the function block.
     If you call a block that contains monitoring, assign a ProDiag function block to the monitoring functions in the "ProDiag FB" text box.
   - If you click on the "Multi-instance" button, you will have to enter the name of the tag in the "Name in the interface" text box; this is the name that you use to enter the called function block as a static tag in the interface of the calling block.
   - If you click on the "Parameter instance" button, enter the name of the in/out (InOut) parameter to which the instance should be passed during runtime in the "Name in the interface" text box.
3. If you call a block that contains monitoring, assign a ProDiag function block to the monitoring functions.
4. Confirm your entries with "OK".

Result

The function or the function block is inserted with its parameters. You can then assign the parameters.

See also: AUTOHOTSPOT

Note

If when calling a function block you specify an instance data block that does not exist, it will be created.

See also

Updating block calls in FBD (Page 4957)

Changing the instance type (Page 4959)

Updating block calls in FBD

If interface parameters of a called block are changed, the block call can no longer be executed correctly. You can avoid such inconsistent block calls by updating the block calls.

You have the following options for updating the block calls:

- Explicit updating of all inconsistent block calls in the program editor.
  The inconsistent block calls within the open block are updated. The following actions are carried out in the process:
  - New parameters are added.
  - Deleted parameters are removed if they are not connected.
  - Renamed parameters get the new parameter names.
- Explicit updating of a block call in the program editor.
  The "Interface update" dialog is displayed. In this dialog, you have the option of changing the connection of the operands of the new interface. The inconsistent call of this block is then updated. The following actions are carried out in the process:
  - New parameters are added.
  - Deleted parameters are removed if they are not connected.
  - Renamed parameters get the new parameter names.
- Implicit updating during compilation.
  All block calls in the program as well as the used PLC data types will be updated. Note that when you call functions (FCs) before the next compilation process, all new formal parameters must be supplied with actual parameters.

Updating all inconsistent block calls in the program editor

To open all block calls in a block, follow these steps:

1. Open the calling block in the program editor.
2. Click "Update inconsistent block calls" in the toolbar.

Updating a specific block call in the program editor

To update a specific block call in the program editor, follow these steps:

1. Open the calling block in the program editor.
2. Right-click on the block call that you want to update.
3. Select the "Update" command in the shortcut menu.
   The "Interface update" dialog opens. This dialog shows the differences between the block interface in use and the changed interface of the called block.
4. If necessary, change the connection of the operands. To do this, you have the following options:
   - You can use either a drag-and-drop operation or a cut/copy-and-paste operation to move the operand from the old interface to the new interface.
   - You can delete an operand.
   - You can rename an operand.
   - You can specify a new operand via autocompletion.
5. Click "OK" to update the block call. If you want to cancel the update, click "Cancel".

Note

Note that the "Update block call" command is only available provided you did not previously update all block calls in the editor with the "Update inconsistent block calls" command.

Update block calls during compilation

Follow these steps to update all block calls and uses of PLC data types during compilation implicitly:

1. Open the project tree.
2. Select the "Program blocks" folder.
3. Select the command "Compile > Software (rebuild all blocks)" in the shortcut menu.

See also

Inserting block calls using a drag-and-drop operation (Page 4956)

Changing the instance type (Page 4959)

Changing the block call

You have the option of changing the called block for a block call. But keep in mind that no new instance data blocks are created, for example, when changing from a function (FC) to a function block (FB).

Procedure

To change the called block of a block call, follow these steps:

1. Click on the name of the called block within the block call and press the <F2> key. Or double-click the name of the called block.
   A text box opens, and the name of the currently called block is selected.
2. Enter the name of the block you want to call or select a block in the autocompletion.
3. If you want to call an FB, create a new instance data block, if necessary, and specify it as operand.

Changing the instance type

Instance type

There are two ways of calling function blocks:

- As a single instance
- As a multiple instance
- As parameter instance

See also: AUTOHOTSPOT

You can modify a defined instance type at any time.

Requirement

The user program contains a block call.

Procedure

To change the instance type of a function block, follow these steps:

1. Open the code block and select the block call.
2. Select the "Change instance" command in the shortcut menu.
   The "Call options" dialog opens.
3. Click on the button "Single instance", "Multi-instance" or "Parameter instance".
   - If you select the "Single instance" instance type, enter a name for the data block that is to be assigned to the function block.
   - If you select "Multiple instance" as the instance type, enter in the "Name in the interface" text field the name of the tag with which the called function block is to be entered as a static tag in the interface of the calling block.
   - If you select "Parameter instance" as the instance type, enter the name of the in/out (InOut) parameter to which the instance should be passed during runtime in the "Name in the interface" text box.
4. Confirm your entries with "OK".

Note

The previous instances are not deleted automatically.

See also

Inserting block calls using a drag-and-drop operation (Page 4956)

Updating block calls in FBD (Page 4957)

11.13.4.7 Inserting complex FBD instructions

Using the "Calculate" instruction (S7-1200, S7-1500)

Requirement

A network is available.

Procedure

Proceed as follows to use the "Calculate" instruction:

1. Open the "Instructions" task card.
2. Navigate to "Math functions > CALCULATE" in the "Basic instructions" pane.
3. Use drag-and-drop to move the element to the desired place in the network.
   The instruction "Calculate" will be inserted for the data type with a placeholder expression and question mark.
4. Enter the data type for the calculation.
5. Enter the operands for the calculation.

   Note

   The calculation is run with the inputs of the "Calculate" instruction. If you want to use constants you must also insert appropriate inputs for them.

6. Click on the "Edit 'Calculate' instruction" button to replace the placeholder expression with the correct expression.
   The "Edit 'Calculate' instruction" dialog will open.

7. Enter the required expression in the "OUT:= " text box.

   Note

   In the "Example" area you can find an example of a valid expression and possible instructions that you can use.

   To determine a value with the help of Pythagoras' theorem, for example, enter "OUT := SQRT (SQR (IN1) + SQR (IN2))".

8. Confirm your entry with "OK".

See also

CALCULATE: Calculate (Page 3020)

11.13.4.8 Using free-form comments

Basic information on using free comments in FBD

Introduction

Free-form comments allow you to add comments to the source code for graphic programming languages similar to line comments for textual languages.

Free-form comments can be used for all non-binary boxes.

Commenting out program code

To deactivate individual instructions or entire program sections in the LAD and FBD programming languages, you must set a bit parallel to the instruction or in series to an instruction block. Commenting out, as in STL and SCL, is not possible in LAD and FBD.

Examples of commenting out program code in all programming languages are available in the Siemens Industry Online under the following FAQ ID: 109482004 (https://support.industry.siemens.com/cs/hu/en/view/109482004)

See also

Inserting free-form comments (Page 4961)

Editing free-form comments (Page 4962)

Deleting free-form comments (Page 4963)

Inserting free-form comments

Requirement

A network with instructions is available.

Procedure

To insert a free comment on an instruction, proceed as follows:

1. If necessary, activate the "Free-form comments on/off" button in the toolbar.
2. Right-click on the instruction for which you want to insert a free-form comment.
3. Select the "Insert comment" command in the shortcut menu.
   A comment box with a standard comment opens. The comment box is connected by an arrow to the corresponding instruction.
4. Enter the required comment in the comment box.

See also

Basic information on using free comments in FBD (Page 4961)

Editing free-form comments (Page 4962)

Deleting free-form comments (Page 4963)

Editing free-form comments

Introduction

Free-form comments can be edited as follows:

- Changing the comment text
- Changing the position and size of the comment box
- Attaching a comment to another element
- Showing and hiding free comments

Changing the comment text

To change the text of free-form comments, follow these steps:

1. Click on the comment box.
2. Enter the desired text.

Changing the position of the comment box

To change the positioning of the comment box, follow the steps below:

1. Left-click the comment box and keep the mouse button pressed.
2. Drag the comment box to the desired location.

Changing the size of the comment box

To change the size of the comment box, follow the steps below:

1. Click on the comment box.
2. Drag the comment box on the move handle in the lower right corner to the desired size.

Attaching a comment to another element

To attach a free-form comment to another element, follow these steps:

1. Left-click the point of the arrow that links the comment box with the instruction and keep the mouse button pressed.
2. Drag the arrow to the element to which you want to attach the comment. Possible insertion points are marked with a green square.
3. Release the mouse button.

Showing and hiding free comments

To show or hide a free-form comments, follow these steps:

1. Click the "Free-form comment on/off" button in the toolbar.

See also

Basic information on using free comments in FBD (Page 4961)

Inserting free-form comments (Page 4961)

Deleting free-form comments (Page 4963)

Deleting free-form comments

Procedure

To delete a free-form comment, proceed as follows:

1. Right-click on the free-form comment that you want to delete.
2. Select the "Delete" command in the shortcut menu.

See also

Basic information on using free comments in FBD (Page 4961)

Inserting free-form comments (Page 4961)

Editing free-form comments (Page 4962)

11.13.5 Editing FBD elements

11.13.5.1 Selecting FBD elements

You can select several individual elements or all elements in a network.

Requirement

FBD elements are available

Selecting several individual FBD elements

To select several individual FBD elements, follow these steps:

1. Press and hold down the <Ctrl> key.
2. Click on all the FBD elements you wish to select.
3. Now release the <Ctrl> key.

Selecting all FBD elements in a network

To select all FBD elements in a network, follow these steps:

1. Go to the network whose elements you wish to select.
2. Select the "Select all" command in the "Edit" menu or press <Ctrl+A>.

See also

Copying FBD elements (Page 4964)
Cutting FBD elements (Page 4965)
Pasting an FBD element from the clipboard (Page 4966)
Replacing FBD elements (Page 4966)
Adding additional inputs and outputs to FBD elements (Page 4967)
Removing instruction inputs and outputs (Page 4968)
Deleting FBD elements (Page 4969)

11.13.5.2 Copying FBD elements

Requirement

An FBD element is available.

Procedure

To copy an FBD element, follow these steps:

1. Right-click the FBD element that you want to copy.
2. Select "Copy" in the shortcut menu.

Result

The FBD element will be copied and saved to the clipboard.

See also

Selecting FBD elements (Page 4964)
Cutting FBD elements (Page 4965)
Pasting an FBD element from the clipboard (Page 4966)
Replacing FBD elements (Page 4966)
Adding additional inputs and outputs to FBD elements (Page 4967)
Removing instruction inputs and outputs (Page 4968)
Deleting FBD elements (Page 4969)

11.13.5.3 Cutting FBD elements

Requirement

An FBD element is available.

Cutting

To cut an FBD element, follow these steps:

1. Right-click the FBD element that you want to cut.
2. Select "Cut" in the shortcut menu.

Result

The FBD element will be cut and saved to the clipboard.

See also

Selecting FBD elements (Page 4964)
Copying FBD elements (Page 4964)
Pasting an FBD element from the clipboard (Page 4966)
Replacing FBD elements (Page 4966)

Adding additional inputs and outputs to FBD elements (Page 4967)

Removing instruction inputs and outputs (Page 4968)

Deleting FBD elements (Page 4969)

11.13.5.4 Pasting an FBD element from the clipboard

Requirement

An FBD element is available.

Procedure

To paste an FBD element from the clipboard, follow these steps:

1. Copy an FBD element or cut an FBD element.
2. Right-click the point in the network where you want to paste the element.
3. Select "Paste" in the shortcut menu.

See also

Selecting FBD elements (Page 4964)

Copying FBD elements (Page 4964)

Cutting FBD elements (Page 4965)

Replacing FBD elements (Page 4966)

Adding additional inputs and outputs to FBD elements (Page 4967)

Removing instruction inputs and outputs (Page 4968)

Deleting FBD elements (Page 4969)

11.13.5.5 Replacing FBD elements

You can easily exchange FBD elements with other FBD elements of the same type. This has the advantage that the parameters are retained and need not be entered again. For example, you can exchange OR and AND, RS-FlipFlop and SR-FlipFlop, comparison functions or jump instructions.

Requirements

A network with at least one FBD element is present.

Procedure

To replace an FBD element with another FBD element, follow these steps:

1. Select the FBD element that you want to replace.
   If elements compatible with the selected FBD element are available, a triangle will appear in the upper right-hand corner of the element.
2. Position the cursor above the triangle of the FBD element.
   A drop-down list is displayed.
3. From the drop-down list, select the FBD element that you want to use to replace the existing FBD element.

See also

Selecting FBD elements (Page 4964)

Copying FBD elements (Page 4964)

Cutting FBD elements (Page 4965)

Pasting an FBD element from the clipboard (Page 4966)

Adding additional inputs and outputs to FBD elements (Page 4967)

Removing instruction inputs and outputs (Page 4968)

Deleting FBD elements (Page 4969)

11.13.5.6 Adding additional inputs and outputs to FBD elements (S7-1200, S7-1500)

Introduction

You can expand several FBD elements with additional inputs that execute arithmetic or binary operations. Such elements are, for example, the instructions "Add" (ADD), "Multiply" (MUL), AND or OR. You can expand the "MOVE value" (MOVE) and "Demultiplex" (DEMUX) instruction boxes by adding additional outputs.

The name of the new inputs and outputs is comprised of the type of inserted element and a consecutive number. The name of a new input is may be "IN2"; the name of a new output may be "OUT2".

Requirements

An FBD element is available that permits the insertion of additional inputs and outputs.

Inserting an additional input

To add an additional input to the box of an FBD element, follow these steps:

1. Right-click on an existing input of the FBD element.
2. Select "Insert input" in the shortcut menu.
   An additional input is added to the box of the FBD element.

Or:

1. Click on the yellow star symbol beside the last input in the instruction box.
   An additional input is added to the box of the FBD element.

Inserting an additional output

To add an additional output to the box of an FBD element, follow these steps:

1. Right-click on an existing output of the FBD element.
2. Select "Insert output" from the shortcut menu.
   An additional output is added to the box of the FBD element.

Or:

1. Click on the yellow star symbol beside the last output of the instruction box.
   An additional output is added to the box of the FBD element.

See also

Inserting FBD elements (Page 4948)

Selecting FBD elements (Page 4964)

Copying FBD elements (Page 4964)

Cutting FBD elements (Page 4965)

Pasting an FBD element from the clipboard (Page 4966)

Replacing FBD elements (Page 4966)

Removing instruction inputs and outputs (Page 4968)

Deleting FBD elements (Page 4969)

11.13.5.7 Removing instruction inputs and outputs (S7-1200, S7-1500)

Introduction

Inputs and outputs which you have added to an instruction can be removed.

Requirement

An FBD element is available, which you have expanded with additional inputs or outputs.

Remove input

To remove an input, follow these steps:

1. Select the input that you want to remove.
2. Select the "Delete" command in the shortcut menu.
   The input of the FBD element is removed.

Remove output

To remove an output, follow these steps:

1. Select the output that you want to remove.
2. Select the "Delete" command in the shortcut menu.
   The output of the FBD element will be removed.

See also

Inserting FBD elements (Page 4948)

Selecting FBD elements (Page 4964)

Copying FBD elements (Page 4964)

Cutting FBD elements (Page 4965)

Pasting an FBD element from the clipboard (Page 4966)

Replacing FBD elements (Page 4966)

Adding additional inputs and outputs to FBD elements (Page 4967)

Deleting FBD elements (Page 4969)

11.13.5.8 Deleting FBD elements

Requirement

An FBD element is available.

Procedure

To delete an FBD element, follow these steps:

1. Right-click the FBD element that you want to delete.
2. Select the "Delete" command in the shortcut menu.

See also

Inserting FBD elements (Page 4948)

Selecting FBD elements (Page 4964)

Copying FBD elements (Page 4964)
Cutting FBD elements (Page 4965)
Pasting an FBD element from the clipboard (Page 4966)
Replacing FBD elements (Page 4966)
Adding additional inputs and outputs to FBD elements (Page 4967)
Removing instruction inputs and outputs (Page 4968)

11.13.6 Inserting operands in FBD instructions

11.13.6.1 Inserting operands

The character strings "<???>", "<??.?>" and "..." are inserted as placeholders for the parameters when a FBD element is inserted. The "<???>" and "<??.?>" strings displayed in red indicate parameters that need to be connected. The "..." string displayed in black indicates parameters that may be connected. "<??.?>" stands for Boolean placeholders. In addition, FBD elements can have the following different I/Os for the parameters:

- Orange I/Os: Only tags or constants can be interconnected at this I/O.
- Black I/Os: Boolean input or output to which other elements can be connected upstream or downstream.

Note

To display the available data types in a tooltip, move the cursor over the placeholder.

Requirement

An FBD element is available.

Procedure

To connect the parameters of an FBD element, follow these steps:

1. Click the placeholder of the parameter.
   An input field is opened.
2. Enter the corresponding parameters, for example a PLC tag, a local tag or a constant.

   Note

   If you enter the absolute address of a parameter that has already been defined, this absolute address will be changed to the symbolic name of the parameter as soon as the input is confirmed. If you have not yet defined the parameter, a new tag with this absolute address and the default name "Tag_1" will be entered in the PLC tag table. When you confirm your input, the absolute address will be replaced with the symbolic name "Tag_1".

3. Confirm the parameter with the Enter key.
4. If you have not yet defined the parameter, you can define it directly in the program editor using the shortcut menu.
   See also: "Declaring PLC tags in the program editor (Page 4847)".

Or drag from it the PLC tag table:

1. In the project tree, select the "PLC tags" folder and open the PLC tag table.
2. If you have opened the PLC tag table, drag the desired tag to the corresponding location in your program. If you have not opened the PLC tag table yet, open the detail view now. Drag the desired tag from the detail view to the appropriate place in your program.

Or drag from it the block interface:

1. Open the block interface.
2. Drag the desired operand from the block interface to the corresponding location in your program.

Result

- If the syntax is error-free, the displayed parameter is black.
- If there is an error in the syntax, the cursor stays in the input field and a corresponding error message is displayed in the inspector window in the "Info > Syntax" register.

See also

Using autocompletion (Page 4706)

11.13.6.2 Wiring hidden parameters

Introduction

If an instruction contains hidden parameters, the instruction box has a small arrow on the lower edge. You can recognize hidden parameters by their white font.

You can show and wire the hidden parameters at any time.

Showing or hiding hidden parameters

To show or hide hidden parameters, follow these steps:

1. Click on the down arrow at the bottom edge of the instruction box to show hidden parameters.
2. Click on the up arrow at the bottom edge of the instruction box to hide hidden parameters.

Wiring hidden parameters

To wire parameters, follow these steps:

1. Wire the hidden parameters like normally visible parameters.
   Its visibility may change due to the interconnection of the parameter:
   - The parameter is visible if you have selected the "Hide, if no parameter is assigned" option in the properties.
   - The parameter remains hidden if you have selected the "Hide" option in the properties.

See also

Hiding parameters during the block call (Page 4782)

11.13.6.3 Displaying or hiding tag information

Introduction

You can display the following information about the tags to be used in the Program editor:

- Name of the tag
- Tag addresses
- Simple or hierarchical comments to document the tags

The information is taken from the block interface for local tags and DB tags and from the PLC tag table for tags that are valid CPU-wide.

You can display the tag information either for all the blocks or for individually opened blocks. If you display the tag information for all the blocks, the tag information for all the blocks currently opened and opened in future is shown. You can hide the tag information at any time again. If you have hidden the tag information for all blocks, you can display it again for individual blocks that are open.

When you select the display of tag information with hierarchical comments, the comments of the higher structure levels are also displayed for structured tags. The comments are shown in brackets after the tag comment; the comments of the individual levels are separated by a dot. If there is no comment for a tag on a structure level, it is not displayed. This is indicated by two successive dots.

You can display the tag information either collectively below a LAD/FBD network or directly at the respective operand. As with showing and hiding tag information, you can set this via the settings for all block or change specifically for an opened block.

Displaying or hiding tag information for all the blocks

Proceed as follows to display or hide the tag information for all the blocks:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.

3. If you want to display the tag information, select the "Show" option or the "Tag information with hierarchical comments" option in the drop-down list, depending on whether you want to display simple or hierarchical comments.
4. Use the "Position of the tag information (LAD/FBD)" drop-down list to specify whether the tag information is to be displayed below the LAD/FBD network or on the operand.
   For all open blocks, the tag information is shown and displayed at the selected position. If you open additional blocks the tag information for these blocks is also shown.
5. If you want to hide the tag information, select the option "Hide" in the "Tag information" drop-down list.
   The tag information is hidden for all open blocks. If you open additional blocks the tag information for these blocks is also hidden.

Displaying or hiding tag information for an opened block

Proceed as follows to display or hide the tag information for an opened block:

1. If you want to display the tag information, select the "Show tag information" option or the "Tag information with hierarchical comments" option in the "Shows the tag information" drop-down list, depending on whether you want to display simple or hierarchical comments.
2. To define the position of the tag information, click the "Position of the tag information (LAD/FBD)" button in the program editor toolbar. The position changes with each click on the button. Alternatively, you can click the small arrow beside the "Position of the tag information (LAD/FBD)" button to open the drop-down list. Than select the position from the drop-down list.
   The tag information is shown and displayed at the selected position.
3. If you want to hide the tag information, select the option "Hide tag information" in the "Shows the tag information" drop-down list.
   The tag information is hidden.

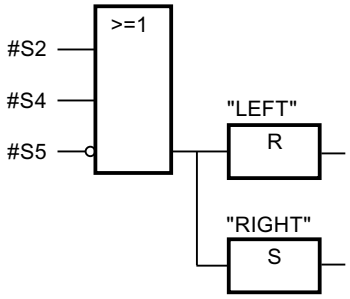
11.13.7 Branches in FBD

11.13.7.1 Basic information on branches in FBD

Definition

You can use the Function Block Diagram (FBD) programming language to program parallel branches. This is done using branches that are inserted between the boxes. You can insert additional boxes within the branch and in this way build up complex function block diagrams.

The figure below shows an example of the use of branches:



See also

Rules for branches in FBD (Page 4974)

Inserting branches in FBD networks (Page 4974)

Deleting branches in FBD networks (Page 4975)

11.13.7.2 Rules for branches in FBD

Rules

The following rules apply to the use of branches in FBD:

- Branches are opened downward.
- Branches can be inserted only between FBD elements.
- To delete a branch, you must delete all FBD elements, including the branch itself.
- If you delete the connection between two branches, the FBD elements of the interrupted branch will be positioned freely in the network.

See also

Basic information on branches in FBD (Page 4973)

Inserting branches in FBD networks (Page 4974)

Deleting branches in FBD networks (Page 4975)

11.13.7.3 Inserting branches in FBD networks

Requirement

A network is available.

Procedure

To insert a new branch in a network, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "General > Branch" in the "Basic instructions" palette.
3. Drag the element from the "Elements" pane to the a required location on a connection line between two boxes.

See also

Rules for branches in FBD (Page 4974)

Basic information on branches in FBD (Page 4973)

Deleting branches in FBD networks (Page 4975)

11.13.7.4 Deleting branches in FBD networks

Requirement

A branch is available.

Procedure

To delete a branch, follow these steps:

1. Select the connection line that links the branch to the main branch.
2. Select the "Delete" command in the shortcut menu.

Result

The branch is now deleted. Boxes connected to the deleted branch are placed freely within the network.

See also

Rules for branches in FBD (Page 4974)

Basic information on branches in FBD (Page 4973)

Inserting branches in FBD networks (Page 4974)

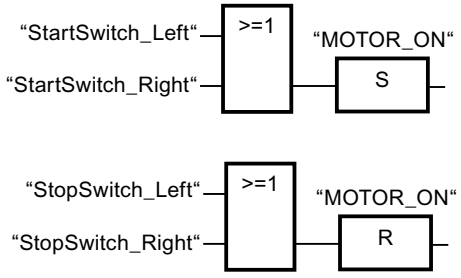
11.13.8 Logic paths in FBD

11.13.8.1 Basic information on logic paths in FBD

Use of logic paths

The user program will be mapped in one or more networks. The networks can contain one or more logic paths on which the binary signals are arranged in the form of boxes.

The following figure shows an example of the use of several logic paths within a network:



Rules

Remember the following rules when using logic paths:

- Connections are not permitted between logic paths.
- Only one jump instruction is permissible per network. The positioning rules for jump instructions remain valid.

Executing logic paths

Logic paths are executed from top to bottom and from left to right. This means that the first instruction in the first logic path of the first network is executed first. All instructions of this logic path are then executed. After this come all other logic paths of the first network. The next network is executed only after all logic paths have first been executed.

When jumps are used the regular execution of the logic paths is circumvented and the instruction is executed at the jump destination.

Differences between branches and logic paths

The difference between branches and logic paths is that the logic paths are independent branches that can also stand in a different network. Branches, on the other hand, permit the programming of a parallel connection and have a common preceding logic operation.

See also

Inserting a logic path (Page 4977)

Deleting a logic operation path (Page 4977)

11.13.8.2 Inserting a logic path

Requirement

- A block is open.
- A network is available.

Procedure

To insert a new logic path in a network, follow these steps:

1. Insert any instruction in a network in such a way that it has no connection to existing instructions.
   A new logic path is inserted.
2. Insert an assignment at the end of the new logic path.
3. Insert additional instructions in the new logic path.

See also

Basic information on logic paths in FBD (Page 4976)

Deleting a logic operation path (Page 4977)

11.13.8.3 Deleting a logic operation path

Requirement

A logic path is available.

Procedure

To delete a logic path, proceed as follows:

1. Hold down the left mouse button and draw a frame around the logic path. At the same time, make sure that you select all instructions of the logic path. Alternatively, you can hold down the <Shift> key and select the first the last instruction of the logic path.
2. Right-click on one of the instructions in the logic path.
3. Select the "Delete" command in the shortcut menu.

See also

Basic information on logic paths in FBD (Page 4976)

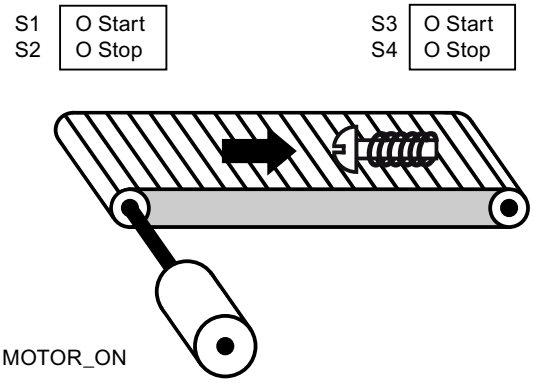
Inserting a logic path (Page 4977)

11.13.9 FBD programming examples

11.13.9.1 Example of controlling a conveyor belt

Controlling a conveyor belt

The following figure shows a conveyor belt that can be activated electrically. There are two pushbuttons at the beginning of the conveyor belt: S1 for START and S2 for STOP. There are also two pushbuttons at the end of the conveyor belt: S3 for START and S4 for STOP. It is possible to start and stop the conveyor belt from either end.



Implementation

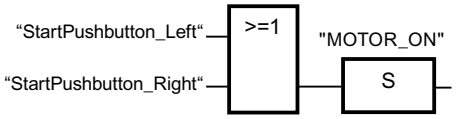
The following table shows the definition of the tags used:

Name	Declaration	Data type	Description
StartPushbutton_Left (S1)	Input	BOOL	Start pushbutton on the left side of the conveyor belt
StopPushbutton_Left (S2)	Input	BOOL	Stop pushbutton on the left side of the conveyor belt
StartPushbutton_Right (S3)	Input	BOOL	Start pushbutton on the right side of the convey-or belt
StopPushbutton_Right (S4)	Input	BOOL	Stop pushbutton on the right side of the convey-or belt
MOTOR_ON	Output	BOOL	Turn on the conveyor belt motor

The following networks show the FBD programming for solving this task:

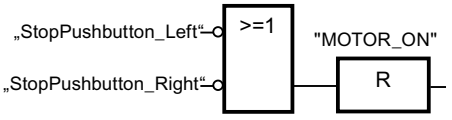
Network 1:

The conveyor belt motor is switched on when start pushbutton "S1" or "S3" is pressed.



Network 2:

The conveyor belt motor is switched off when stop pushbutton "S2" or "S4" is pressed.



See also

Basic information on FBD (Page 4937)

Settings for FBD (Page 4939)

Working with networks (Page 4940)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

Branches in FBD (Page 4973)

Logic paths in FBD (Page 4976)

Example of detecting the direction of a conveyor belt (Page 4980)

Example of detecting the fill level of a storage area (Page 4982)

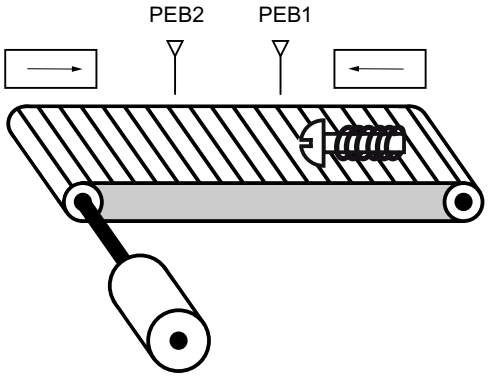
Example of calculating an equation (Page 4986)

Example of controlling room temperature (Page 4988)

11.13.9.2 Example of detecting the direction of a conveyor belt

Detecting the direction of a conveyor belt

The detected running direction of the belt is indicated by a RIGHT arrow or a LEFT arrow. If additional conveyed material is approaching PEB1 from the right or PEB2 from the left, the displayed arrow is initially switched off until, after both photoelectric barriers are passed, the running direction can be detected again and the corresponding arrow displayed. For the solution of the task, 2 edge memory bits are needed that detect the signal change from "0" to "1" at the two photoelectric barriers.



Implementation

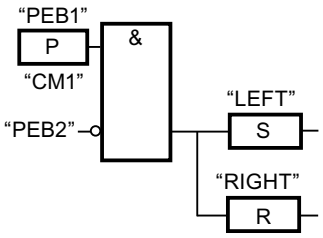
The following table shows the definition of the tags used:

Name	Declaration	Data type	Description
PEB1	Input	BOOL	Photoelectric barrier 1
PEB2	Input	BOOL	Photoelectric barrier 2
RIGHT	Output	BOOL	Display during movement to right
LEFT	Output	BOOL	Display during movement to left
CM1	Input	BOOL	Edge bit memory 1
CM2	Input	BOOL	Edge bit memory 2

The following networks show the FBD programming for solving this task:

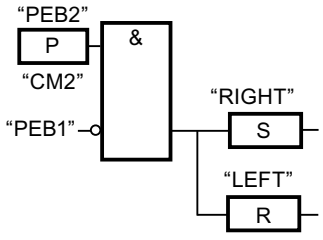
Network 1:

If the signal state changes from "0" to "1" (positive edge) at photoelectric barrier "PEB1" and, at the same time, the signal state at "PEB2" is "0", the object on the belt is moving to the left.



Network 2:

If the signal changes from "0" to "1" (positive edge) at photoelectric barrier "PEB2" and, at the same time, the signal state at "PEB1" is "0", the object on the belt is moving to the right.



See also

Basic information on FBD (Page 4937)

Settings for FBD (Page 4939)

Working with networks (Page 4940)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

Branches in FBD (Page 4973)

Logic paths in FBD (Page 4976)

Example of controlling a conveyor belt (Page 4978)

Example of detecting the fill level of a storage area (Page 4982)

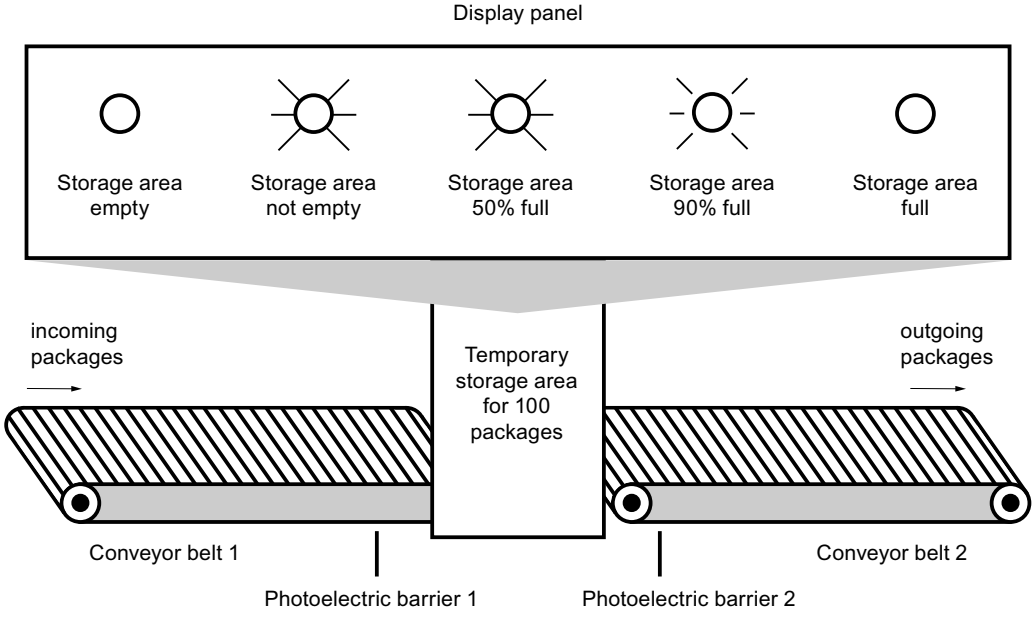
Example of calculating an equation (Page 4986)

Example of controlling room temperature (Page 4988)

11.13.9.3 Example of detecting the fill level of a storage area

Detecting the fill level of a storage area

The following figure shows a system with two conveyor belts and a temporary storage area between them. Conveyor belt 1 delivers packages to the storage area. A photoelectric barrier at the end of conveyor belt 1 near the storage area detects how many packages are delivered to the storage area. Conveyor belt 2 transports packages from the temporary storage area to a loading dock where they are loaded onto trucks. A photoelectric barrier at the storage area exit detects how many packages leave the storage area to be transported to the loading dock. Five display lamps indicate the capacity of the temporary storage area.



Implementation

The following table shows the definition of the tags used:

Name	Declaration	Data type	Description
PEB1	Input	BOOL	Photoelectric barrier 1
PEB2	Input	BOOL	Photoelectric barrier 2
RESET	Input	BOOL	Reset counter
LOAD	Input	BOOL	Adjust the current counter value to the value of the PV parameter.
MAX STORAGE AREA FILL AMOUNT	Input	INT	Maximum possible number of packages in the storage area
PACKAGECOUNT	Output	INT	Number of packages in the storage area (current count value)

Name	Declaration	Data type	Description
STOCK_PACKAGES	Output	BOOL	Is set when the current count value is greater than or equal to the value of the "MAX STORAGE AREA FILL AMOUNT" tag.
STOR_EMPTY	Output	BOOL	Display lamp: Storage area empty
STOR_NOT_EMPTY	Output	BOOL	Display lamp: Storage area not empty
STOR_50%_FULL	Output	BOOL	Display lamp: Storage area 50 % full
STOR_90%_FULL	Output	BOOL	Display lamp: Storage area 90 % full
STOR_FULL	Output	BOOL	Display lamp: Storage area full
VOLUME_50	Input	INT	Comparison value: 50 packages
VOLUME_90	Input	INT	Comparison value: 90 packages
VOLUME_100	Input	INT	Comparison value: 100 packages

The following networks show the FBD programming for activating the lamps:

Network 1:

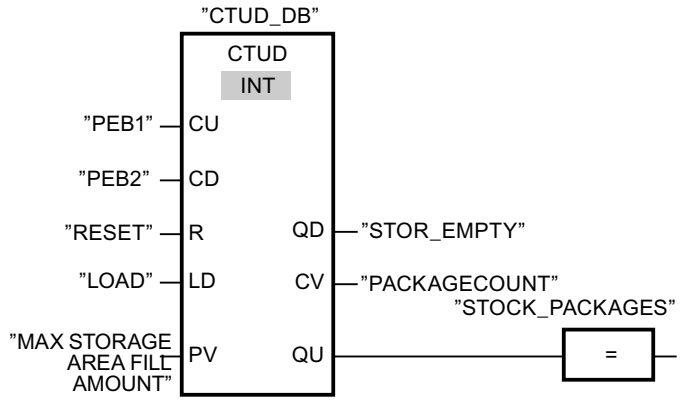
When a package is delivered to the storage area, the signal state at "PEB1" switches from "0" to "1" (positive signal edge). On a positive signal edge at "PEB1", the "Up" counter is enabled, and the current count value of "PACKAGECOUNT" is increased by one.

When a package is delivered from the storage area to the loading dock, the signal state at "PEB2" switches from "0" to "1" (positive signal edge). On a positive signal edge at "PEB2", the "Down" counter is enabled, and the current count value of "PACKAGECOUNT" is decreased by one.

If there are no packages in the storage area ("PACKAGECOUNT" = "0"), the "STOR_EMPTY" tag is set to signal state "1", and the "Storage area empty" lamp is switched on.

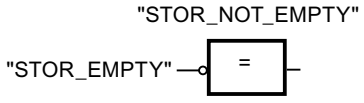
The current count value can be reset to "0" if the "RESET" tag is set to signal state "1".

If the "LOAD" tag is set to signal state "1", the current count value is set to the value of the "MAX STORAGE AREA FILL AMOUNT" tag. As long as the current count value is greater than or equal to the value of the "MAX STORAGE AREA FILL AMOUNT" tag, the "STOCK_PACKAGES" tag supplies the signal state "1".



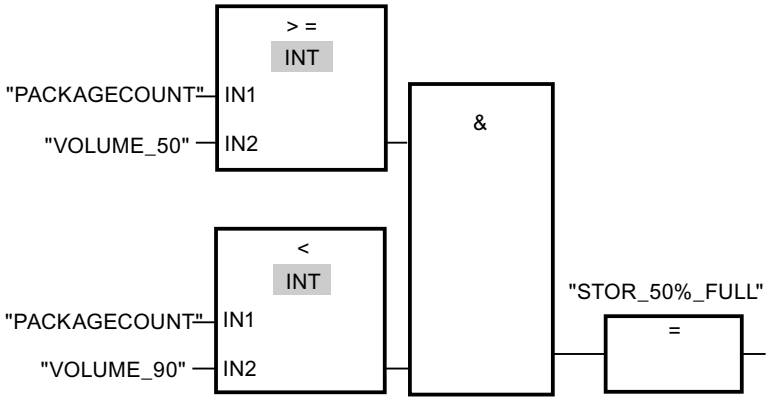
Network 2:

As long as there are packages in the storage area, the "STOR_NOT_EMPTY" tag is set to signal state "1", and the "Storage area not empty" lamp is switched on.



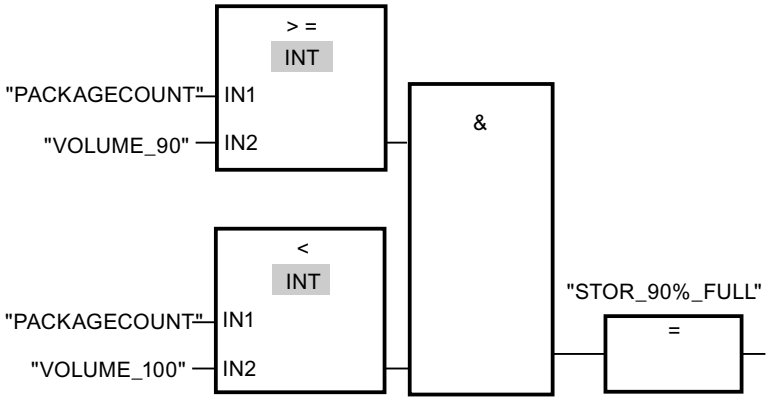
Network 3:

If the number of packages in the storage area is greater than or equal to 50, the "Storage area 50 % full" lamp switches on.



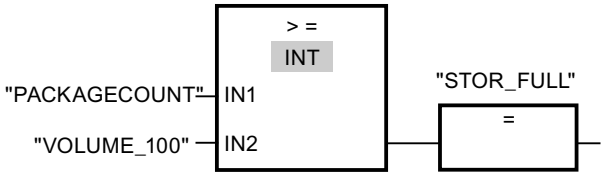
Network 4:

If the number of packages in the storage area is greater than or equal to 90, the "Storage area 90% full" lamp switches on.



Network 5:

If the number of packages in the storage area reaches 100, the lamp for the "Storage area full" message switches on.



See also

Basic information on FBD (Page 4937)

Settings for FBD (Page 4939)

Working with networks (Page 4940)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

Branches in FBD (Page 4973)

Logic paths in FBD (Page 4976)

Example of controlling a conveyor belt (Page 4978)

Example of detecting the direction of a conveyor belt (Page 4980)

Example of calculating an equation (Page 4986)

Example of controlling room temperature (Page 4988)

11.13.9.4 Example of calculating an equation

Calculating a complex equation

The following program example shows you how to solve equations of the following type with the FBD programming language:

X= ((A + B) x C) / D

You can always solve this equation with mathematical operations. To do so, you can use the instructions "ADD", "MUL" and "DIV". If you are using a CPU of the S7-1200 or S7-1500 series, you can also use the instruction "CALCULATE". Depending on the data type you choose, different mathematical operations are available for this instruction which can be combined with one another.

Implementation with the instructions "ADD", "MUL" and "DIV"

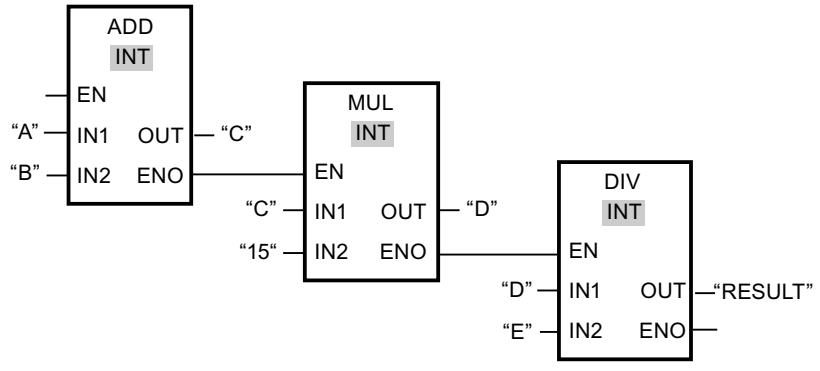
You want to calculate the following equation:

RESULT = ((A + B) x 15) / E

The following table shows the definition of the tags used:

Name	Data type	Comment
A	INT	First value for addition
B	INT	Second value for addition
C	INT	First intermediate result
15	INT	Multiplier
D	INT	Second intermediate result
E	INT	Divisor
RESULT	INT	End result

The following network shows the FBD programming for calculating the equation:



The value of operand "A" is added to the value of operand "B". The sum is stored in operand "C". The value of operand "C" is multiplied by "15". The multiplication result is stored in operand "D". The value stored in operand "D" is then divided by the value of operand "E". The end result is saved in the "RESULT" operand.

Implementation with the instruction "CALCULATE" (S7-1200/1500 only)

You want to calculate the following equation:

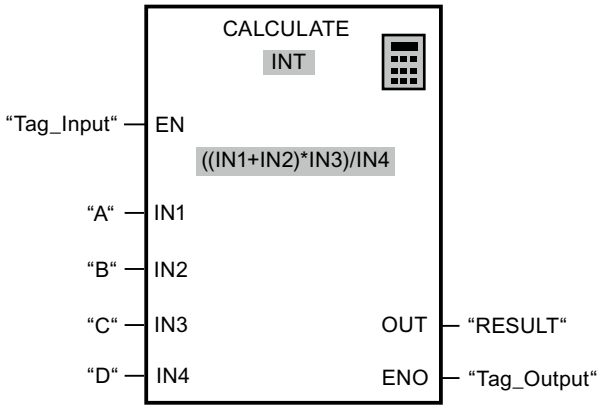
RESULT = ((5 + 10) x 4) / 6

The following table shows the definition of the tags used:

Name	Declaration	Data type	Comment	Value
A	Input	INT	First value for addition	5
B	Input	INT	Second value for addition	10
C	Input	INT	Multiplier	4
D	Input	INT	Divisor	6
RESULT	Output	INT	End result	10

To program the equation with the instruction "CALCULATE", follow these steps:

1. Drag the instruction "CALCULATE" from the "Instructions" task card to an FBD network.
2. You can select the data type INT for the instruction from the "<???>" drop-down list.
3. Interconnect the tags declared in the block interface with the inputs and outputs of the instruction box.
4. Click the "Calculator" icon on the top right corner of the instruction box to enter the equation to be calculated.
   The dialog "Edit 'Calculate' instruction" opens.
5. Enter the following expression in the "OUT:=" field:
   ((IN1 + IN2) * IN4) / IN3
   The equation is displayed in the instruction box.
   The following network shows the result in the FBD programming language:



If the "Tag_Input" has signal state "1", the instruction is executed. The value of operand "A" is added to the value of operand "B". The subtotal is multiplied by "C" and then divided by the value of the "D" operand. The end result is saved in the "RESULT" operand.

See also

Basic information on FBD (Page 4937)
Settings for FBD (Page 4939)
Working with networks (Page 4940)
Inserting FBD elements (Page 4948)
Editing FBD elements (Page 4964)
Inserting operands in FBD instructions (Page 4970)
Branches in FBD (Page 4973)
Logic paths in FBD (Page 4976)
Example of controlling a conveyor belt (Page 4978)
Example of detecting the direction of a conveyor belt (Page 4980)
Example of detecting the fill level of a storage area (Page 4982)
Example of controlling room temperature (Page 4988)

11.13.9.5 Example of controlling room temperature

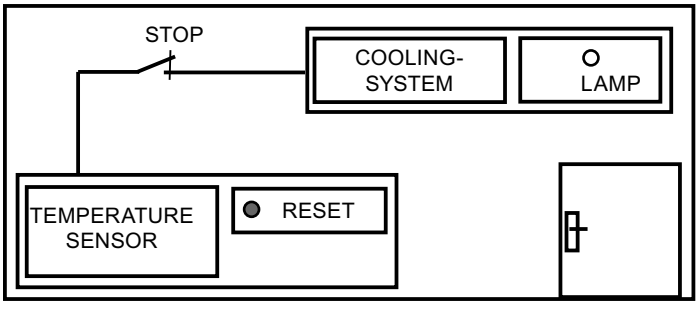
Controlling room temperature

In a cold room, the temperature must be maintained below zero degrees Celsius. A sensor is used to check for any temperature fluctuations. If the temperature rises above zero degrees Celsius, the cooling system switches on for a preset time. The "Cooling system on" lamp is lit during this time.

The cooling system and the lamp are turned off if one of the following conditions is met:

- The sensor reports a temperature fall below zero degrees Celsius.
- The preset cooling time has elapsed.
- The "Stop" pushbutton is pressed.

If the preset cooling time has expired and the temperature in the cold room is still too high, the cooling system can be restarted with the "RESET" pushbutton.



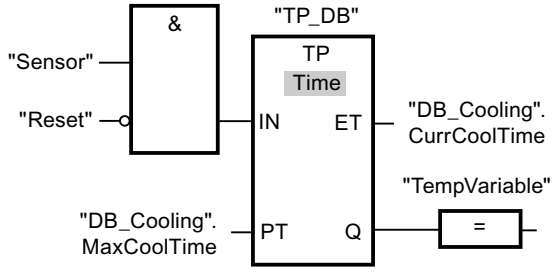
Implementation

The following table shows the definition of the tags used:

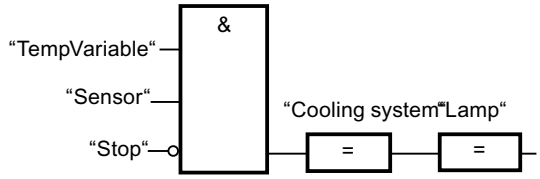
Name	Declaration	Data type	Comment
Sensor	Input	BOOL	Temperature sensor signal
RESET	Input	BOOL	Restart
STOP	Input	BOOL	The cooling system is switched off.
MaxCoolTime	-	TIME	Preset cooling time This tag is defined in the "DB_Cool" data block.
CurrCoolTime	-	TIME	Currently elapsed cooling time This tag is defined in the "DB_Cool" data block.
Cooling system	Output	BOOL	The cooling system is switched on.
Lamp	Output	BOOL	The lamp for the "Cooling system on" message is switched on.
TempVariable	Temp	BOOL	Temporary tag This tag stores the signal state of the IEC timer TP.

The following network shows the FBD programming for controlling room temperature:

Network 1:



Network 2:



When the temperature in the cold room rises above zero degrees Celsius, the signal state at the "Sensor" operand switches from "0" to "1" (positive signal edge). In the case of a positive signal edge at the IN input of the timer function, the preset cooling time is started and the "TempVariable" receives the signal state "1". The signal state "1" of the "TempVariable" causes

the cooling system as well as the display lamp to be turned on in network 2. The "Sensor", "Cooling system" and "Lamp" outputs must be programmed in network 2, because program only one coil can be programmed at the Q output of the timer function.

If the temperature in the cold room falls below zero degrees Celsius, the signal state of the sensor switches back to "0". This switches the cooling system and lamp off.

If the sensor does not signal a temperature drop, the cooling system and lamp are switched off after the preset cooling time has elapsed, at the latest. In this case, the cooling process can be restarted with the "RESET" pushbutton. Pressing and releasing the pushbutton generates a new positive signal edge at the IN input that restarts the cooling system.

The cooling system and the display lamp can be turned off with the "STOP" pushbutton at any time.

See also

Basic information on FBD (Page 4937)

Settings for FBD (Page 4939)

Working with networks (Page 4940)

Inserting FBD elements (Page 4948)

Editing FBD elements (Page 4964)

Inserting operands in FBD instructions (Page 4970)

Branches in FBD (Page 4973)

Logic paths in FBD (Page 4976)

Example of controlling a conveyor belt (Page 4978)

Example of detecting the direction of a conveyor belt (Page 4980)

Example of detecting the fill level of a storage area (Page 4982)

Example of calculating an equation (Page 4986)

11.14 Creating SCL programs

11.14.1 Basics of SCL

11.14.1.1 Programming language SCL

Programming language SCL

SCL (Structured Control Language) is a high-level programming language based on PASCAL. The language is based on DIN EN 61131-3 (international IEC 1131-3).

The standard standardizes programming languages for programmable logic controllers. The SCL programming language fulfills the PLCopen Basis Level of ST language (Structured Text) defined in this standard.

Language elements

SCL also contains higher programming languages in addition to the typical elements of the PLC, such as inputs, outputs, timers or memory bits.

- Expressions
- Value assignments
- Operators

Program control

SCL provides convenient instructions for controlling the program allowing you, for example, to create program branches, loops or jumps.

Application

SCL is therefore particularly suitable for the following areas of application:

- Data management
- Process optimization
- Recipe management
- Mathematical / statistical tasks

11.14.1.2 Expressions

Description

Expressions are calculated during the runtime of the program and return a value. An expression consists of operands (such as constants, tags or function calls) and optionally out of operators (such as *, /, + or -). Expressions can be linked together or nested within each other by operators.

Evaluation order

The evaluation of the expression occurs in a specific order that is defined by the following factors:

- Priority of the operators involved
- Left-to-right order
- Brackets

Types of expressions

The following expression types are available depending on the operator:

- Arithmetic expressions
  Arithmetic expressions consist of either a numerical value or combine two values or expressions with arithmetic operators.
- Relational expressions
  Relational expressions compare the values of two operands and yield a Boolean value. The result is TRUE if the comparison is true, and FALSE if it is not met.
- Logical expressions
  Logical expressions combine two operands with logical operators (AND, OR, XOR) or negating operands (NOT).

How expressions are used

You can use the result of an expression in different ways:

- As a value assignment for a tag
- As as a condition for a control instruction
- As a parameter for a calling a block or instruction

See also

Operators and operator precedence (Page 5001)

Arithmetic expressions (Page 4993)

Relational expressions (Page 4995)

Logical expressions (Page 5000)

11.14.1.3 Arithmetic expressions

Description

Arithmetic expressions consist of either a numerical value or combine two values or expressions with arithmetic operators.

Arithmetic operators can process the data types that are allowed in the CPU in use. If two operands are involved in the operation, the data type of the result is determined based on the following criteria:

- If both operands are integers with sign and have different lengths, the result receives the data type of the longer integer (e. g. INT + DINT = DINT).
- If both operands are integers without sign and have different lengths, the result receives the data type of the longer integer (e. g. USINT + UDINT = UDINT).
- If one operand is an integer with sign and the other integer is an operand without sign, the result receives the next larger data type with sign that covers the integer without sign (e. g. SINT + USINT = INT).
  You can only execute an operation with such operands if the IEC check is not set.
- If one operand is an integer and the other operand is a floating-point number, the result receives the data type of the floating-point number (e. g. INT + REAL = REAL).
- If both operands are floating-point numbers and have different lengths, the result receives the data type of the longer floating-point number (e. g. REAL + LREAL = LREAL).
- The data type of the result of an operation that involves operands of the data type groups "Times" and "Date and time" can be found in the table in section "Data types of arithmetic expressions".
  You cannot use data types of the data type groups "Times" and "Date and time" when the IEC check is set.

Data types of arithmetic expressions

The following table shows the data types you can use in arithmetic expressions:

Operation	Operator	1st operand	2nd operand	Result
Power	**	Integer/floating-point number	Integer/floating-point number	Floating-point number
Unary plus	+	Integer/floating-point number TIME, LTIME	-	Integer/floating-point number TIME, LTIME
Unary minus	-	Integer/floating-point number TIME, LTIME	-	Integer/floating-point number TIME, LTIME
Multiplication	*	Integer/floating-point number	Integer/floating-point number	Integer/floating-point number
		TIME, LTIME	Integer	TIME, LTIME
Division	/	Integer/floating-point number	Integer/floating-point number (not equal 0)	Integer/floating-point number
		TIME, LTIME	Integer	TIME, LTIME

Operation	Operator	1st operand	2nd operand	Result
Modulo function	MOD	Integer	Integer	Integer
Addition	+	Integer/floating-point number	Integer/floating-point number	Integer/floating-point number
		TIME	TIME	TIME
		TIME	DINT	TIME
		LTIME	TIME, LTIME	LTIME
		LTIME	LINT	LTIME
		TOD	TIME	TOD
		TOD	DINT	TOD
		LTOD	TIME, LTIME	LTOD
		LTOD	LINT	LTOD
		DATE	LTOD	DTL
		DATE	TOD	• S7-300/400: DT • S7-1200/1500: DTL
		DT	TIME	DT
		LDT	TIME, LTIME	LDT
		DTL	TIME, LTIME	DTL
Subtraction	-	Integer/floating-point number	Integer/floating-point number	Integer/floating-point number
		TIME	TIME	TIME
		TIME	DINT	TIME
		LTIME [1)]	TIME, LTIME	LTIME
		LTIME	LINT	LTIME
		TOD	TIME	TOD
		TOD	DINT	TOD
		TOD	TOD	TIME
		LTOD	TIME, LTIME	LTOD
		LTOD	LINT	LTOD
		LTOD	LTOD	LTIME
		DATE	DATE	• S7-300/400/1200: TIME • S7-1500: LTIME
		DT	TIME	DT
		DT	DT	TIME
		LDT	TIME, LTIME	LDT
		DTL	TIME, LTIME	DTL
		DTL	DTL	• S7-1200: TIME • S7-1500: LTIME
[1)] Combinations between nanoseconds and milliseconds are not possible within expressions.				

For additional information on valid data types, refer to "See also".

Example

The following example shows an arithmetic expression:

```
SCL
"MyTag1":= "MyTag2" * "MyTag3";
```

See also

Expressions (Page 4992)

Operators and operator precedence (Page 5001)

11.14.1.4 Relational expressions

Description

Relational expressions compare the values or the data types of two operands and yield a Boolean value. The result is TRUE if the comparison is true, and FALSE if it is not met.

Relational operators can process the data types that are allowed in the CPU in use. The data type of the result always is BOOL.

Note the following rules when forming relational expressions:

- All tags are comparable within the following data type groups:
  - Integers/floating-point numbers
  - Binary numbers
  - String
- With the following data types/data groups, only tags of the same type can be compared:
  - TIME, LTIME
  - Date and time
  - PLC data types
  - ARRAY
  - STRUCT
  - Tag to which ANY is pointing
  - Tag to which VARIANT is pointing
- The comparison of STRINGs takes place according to the character encoding in the character set that is set in Windows. The comparison of WSTRINGs takes place according to the character encoding in UTF-16. The length of the tags and the numerical value of each character are used for the comparison.
- S5TIME tags are not permitted as comparison operands. An explicit conversion from S5TIME to TIME or LTIME is necessary.

Comparison of floating-point numbers

When floating-point numbers are compared, the operands to be compared must have the same data type regardless of the setting for the IEC Check.

The special bit patterns of invalid floating-point numbers (NaN) that are the outcome of undefined results (e.g. root of -1) are not comparable. This means if one of the operands has the value NaN, the comparison expression "==: Equal" as well as "<>: Not equal" has the result FALSE.

Comparison of character strings

The individual characters are compared by means of their code (for example, 'a' is greater than 'A') during the comparison of the strings. The comparison is performed from left to right. The first character to be different decides the result of the comparison.

The following table shows examples of "==" comparison of strings:

<Operand1>	<Operand2>	RLO of the instruction
'AA'	'AA'	1
'Hello World'	'HelloWorld'	0
'AA'	'aa'	0
'aa'	'aaa'	0

The following table shows examples of "<>" comparison of strings:

<Operand1>	<Operand2>	RLO of the instruction
'AA'	'aa'	1
'Hello World'	'HelloWorld'	1
'AA'	'AA'	0
'aa'	'aaa'	1

You can also compare individual characters of a string. The number of the character to be compared is specified in square brackets next to the operand name. "MyString[2]", for example, compares the second character of the "MyString" string.

Comparison of timers, date and time

Bit patterns of invalid timers, date and times, e.g. DT#2015-13-33-25:62:99.999_999_999, cannot be compared. This means if one of the operands has an invalid value, the instruction "==: Equal" as well as "<>: Not equal" has the result FALSE.

Not all times can be compared directly with each other, such as S5TIME. In this case they are implicitly converted into another time so that they can be compared, for example to TIME.

If you want to compare dates and times of different data types, the value of the smaller date or time data type is implicitly converted into the larger date or time data type. This means the two date and time data types DATE and DTL, for example, are compared on the basis of DTL.

When the implicit conversions fail, the comparison result is FALSE.

Comparing tags of WORD data type to tags of the S5TIME data type

Both tags are converted to the TIME data type when comparing a tag of WORD data type to a tag of S5TIME data type. The WORD tag is interpreted as an S5TIME value. If one of the two tags cannot be converted, the comparison is not performed and the result is FALSE. If the conversion is successful the comparison is performed based on the selected comparison instruction.

Comparison of structures

Note

Availability of comparison of structures

The option to compare structures is available for a CPU of the S7-1200 series as of firmware version >= 4.2, and for a CPU of the S7-1500 series as of firmware version >= 2.0.

You can compare the values of two structured operands with each other when both tags are of the same structure data type. When structures are compared, the operands to be compared must have the same data type regardless of the setting for the IEC Check. An exception are the comparisons in which one of the operands is a VARIANT or an ANY. If the data type is not yet known at the time when the program is created, you can use the VARIANT data type. In this case you can also compare the operands with a structured tag of any data type. You can also compare two tags of the data type VARIANT or ANY with each other.

The following data types are possible:

- PLC data type
- STRUCT (the structure of the data type STRUCT must be contained in a PLC data type (UDT) or the two structures to be compared are two elements of an ARRAY of STRUCT. Anonymous structures are not permitted.)
- Tag to which ANY is pointing
- Tag to which VARIANT is pointing

The following requirements must be met so that the two tags of the data type ARRAY can be compared with each other:

- The elements must each have the same data type.
- The two ARRAYs must have the same dimension.
- All dimensions must have the same number of elements. The exact ARRAY limits do not have to match.

Note

ARRAY of BOOL

When you compare two operands of the data type ARRAY of BOOL with each other and the number of elements cannot be divided by 8, the fill bits are compared as well. This may have an effect on the comparison result.

The table below shows an example of a comparison of structures for "==: Equal":

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	Tag of data type A <PLC data type>		Tag value	1
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	Tag of data type B <PLC data type>		Tag value	0
	BOOL	FALSE		BOOL	TRUE	
	INT	2		INT	3	

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	VARIANT (supplied with tag of data type A)		Tag value	1
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

The table below shows an example of a comparison of structures for "<>: Not equal":

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	Tag of data type A <PLC data type>		Tag value	0
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

<Operand1>			**<Operand2>**			**RLO of the instruction**
Tag of data type A <PLC data type>		Tag value	Tag of data type B <PLC data type>		Tag value	1
	BOOL	FALSE		BOOL	TRUE	
	INT	2		INT	3	

<Operand1>			<Operand2>			RLO of the instruction
Tag of data type A <PLC data type>		Tag value	VARIANT (supplied with tag of data type A)		Tag value	0
	BOOL	FALSE		BOOL	FALSE	
	INT	2		INT	2	

Data types of relational expressions

The following table shows the data types/data type groups you can use in relational expressions:

Operation	Operator	1st operand	2nd operand	Result
Compare for equal, not equal	=, <>	Integer/floating-point number	Integer/floating-point number	BOOL
		Bit strings	Bit strings	BOOL
		String	String	BOOL
		TIME, LTIME	TIME, LTIME	BOOL
		Date and time	Date and time	BOOL
		VARIANT/ANY	VARIANT/ANY	BOOL
		Any data type (but must correspond to the data type with which the VARIANT is supplied)	VARIANT/ANY	BOOL
		VARIANT/ANY	Any data type	BOOL
		PLC data type	PLC data type	BOOL
		ARRAY of <data type> with fixed and variable ARRAY limits	ARRAY of <data type> with fixed and variable ARRAY limits	BOOL
		STRUCT	STRUCT	BOOL
Compare for less than, less than-equal to, greater than, greater than or equal to	<, <=, >, >=	Integer/floating-point number	Integer/floating-point number	BOOL
		Bit strings (S7-1200/1500 only)	Bit strings (S7-1200/1500 only)	BOOL
		String	String	BOOL
		TIME, LTIME	TIME, LTIME	BOOL
		Date and time	Date and time	BOOL

Example

The following example shows a relational expression:

```
SCL
IF a > b THEN c:= a;
IF A > 20 AND B < 20 THEN C:= TRUE;
IF A<>(B AND C) THEN C:= FALSE;
```

Note

The comparison for STRING and DT are executed internally in the S7-300/400 by extended instructions. The following operands are not permitted for these functions:

- Parameter of a FC
- In-out parameter of an FB of type STRUCT or ARRAY

Note

Comparing the hardware data types HW_IO and HW_DEVICE

If you want to compare the two data types, you must first create a tag of data type HW_ANY in the "Temp" section of the block interface and then copy the LADDR (of data type HW_DEVICE) to the tag. It is then possible to compare HW_ANY and HW_IO.

See also

Expressions (Page 4992)

Operators and operator precedence (Page 5001)

11.14.1.5 Logical expressions

Description

Logical expressions combine two operands with logical operators AND OR XOR or negating operands NOT.

Logical operators can process the data types that are allowed in the CPU in use. The result of a logical expression is of BOOL data type, if both operands are of BOOL data type. If at least one of both operands is a bit string, then the result is also a bit string and is determined by the type of the highest operand. For example, when you link a BYTE type operand to a WORD type operand, the result is type WORD.

To link a BOOL type operand with a bit string, you must first explicitly convert it to a bit string.

Data types of logical expressions

The following table shows the data types you can use in logical expressions:

Operation	Operator	1st operand	2nd operand	Result
Negation (generate one's complement)	NOT	BOOL	-	BOOL
		Bit string	-	Bit string
AND logic operation	AND or &	BOOL	BOOL	BOOL
		Bit string	Bit string	Bit string
OR logic operation	OR	BOOL	BOOL	BOOL
		Bit string	Bit string	Bit string
EXCLUSIVE OR logic operation	XOR	BOOL	BOOL	BOOL
		Bit string	Bit string	Bit string

Example

The following example shows a logical expression:

SCL

```
IF "MyTag1" AND NOT "MyTag2" THEN c := a;
MyTag := ALPHA OR BETA;
```

See also

Expressions (Page 4992)

Operators and operator precedence (Page 5001)

SCL programming examples (Page 5053)

11.14.1.6 Operators and operator precedence

Operators and their order of evaluation

Expressions can be linked together or nested within each other by operators.

The order of evaluation for expressions depends on the precedence of operators and brackets. The following basic rules apply:

- Arithmetic operators are evaluated before relational operators and relational operators are evaluated before logical operators.
- Operators with no precedence are evaluated according to their occurrence from left to right.
- Value assignments are evaluated from right to left.
- Operations in brackets are evaluated first.

The following table provides an overview of the operators and their precedence:

Operator	Operation	Precedence
Arithmetic expressions		
+ (Page 4993)	Unary plus	2
- (Page 4993)	Unary minus	2
** (Page 4993)	Power	3
* (Page 4993)	Multiplication	4
/ (Page 4993)	Division	4
MOD (Page 4993)	Modulo function	4
+ (Page 4993)	Addition	5
- (Page 4993)	Subtraction	5
+= (Page 4993), -= (Page 4993), *= (Page 4993), /= (Page 4993)	Combined value assignments	11
Relational expressions		
< (Page 4995)	Less than	6
> (Page 4995)	Greater than	6
<= (Page 4995)	Less than or equal	6
>= (Page 4995)	Greater than or equal	6
= (Page 4995)	Equal	7
<> (Page 4995)	Not equal	7
Logical expressions		
NOT (Page 5000)	Negation	3
AND (Page 5000) or & (Page 5000)	Boolean AND	8
XOR (Page 5000)	Exclusive OR	9
OR (Page 5000)	Boolean OR	10
Miscellaneous operations		
( ) (Page 4992)	Brackets	1
:= (Page 5002)	Assignment	11

11.14.1.7 Value assignments

Definition

You can use a value assignment to assign the value of an expression to a tag. On the left side of the assignment is the tag that takes the value of the expression on the right.

The name of a function can also be specified as an expression. The function is called by the value assignment and returns its function value to the tag on the left.

The data type of value assignment is defined by the data type of the tag on the left. The data type of the expression on the right must match this type.

You have the following options for programming value assignments:

- Single value assignments
  In case of single value assignments, an expression or tag is assigned to a single tag only:
  Example: `a := b;`
- Multiple value assignments
  In case of multiple value assignments, multiple assignments can be executed with one instruction.
  Example: `a := b := c;`
  This corresponds to the following notation:
  ```
  b := c;
  a := b;
  ```
- Combined value assignments
  In case of combined value assignments, you can combine the operators "+", "-", "*" and "/" with the assignment operator:
  Example: `a += b;`
  This corresponds to the following notation:
  ```
  a := a + b;
  ```
  You can also assign combined value assignments multiple times:
  ```
  a += b += c *= d;
  ```
  This corresponds to the following sequence of assignments:
  ```
  c := c * d;
  b := b + c;
  a := a + b;
  ```

Value assignments for STRUCT data type or PLC data types

An entire structure can be assigned to another if the structures are identically organized and the data types as well as the names of the structural components match.

You can assign a tag, an expression or another structural element to an individual structural element.

Value assignments for the ARRAY data type

An entire ARRAY can be assigned to another ARRAY if both the data types of the ARRAY elements as well as the ARRAY limits match.

You can assign a tag, an expression or another ARRAY element to an individual ARRAY element.

Value assignments for the STRING data type

An entire STRING can be assigned to another STRING.

You can assign another STRING element to an individual STRING element.

Value assignment for data type WSTRING (S7-1200/1500)

An entire WSTRING can be assigned to another WSTRING.

You can assign another WSTRING element to an individual WSTRING element.

Value assignments for the ANY data type

You can assign a tag with the ANY data type only to the following objects:

- Input parameters or temporary local data of FBs that also have the data type ANY.
- Temporary local data of FCs that also have the data type ANY.

Note that you can only point to memory areas with "standard" access mode with the ANY pointer.

Value assignments for the POINTER data type

You cannot use POINTER in value assignments in SCL.

Examples

The following table shows examples of single value assignments:

SCL	
"MyTag1" := "MyTag2";	(* Assignment of a tag*)
"MyTag1" := "MyTag2" * "MyTag3";	(* Assignment of an expression*)
"MyTag" := "MyFC"();	(* Call for a function that assigns its function value to the "MyTag" tag*)
#MyStruct.MyStructElement := "MyTag";	(* Assignment of a tag to a structure element*)
#MyArray[2] := "MyTag";	(* Assignment of a tag to an ARRAY element*)
"MyTag" := #MyArray[1,4];	(* Assignment of an ARRAY element to a tag*)
#MyString[2] := #MyOtherString[5];	(* Assignment of a STRING element to another STRING element*)

The following table shows examples of multiple value assignments:

SCL	
"MyTag1" := "MyTag2" := "MyTag3";	(* Assignment of a tag*)
"MyTag1" := "MyTag2" := "MyTag3" * "MyTag4";	(* Assignment of an expression*)
"MyTag1" := "MyTag2" := "MyTag3 := "MyFC"();	(* Call of a function that assigns its function value to the tags "MyTag1", "MyTag1" and "MyTag1" *)
#MyStruct.MyStructElement1 := #MyStruct.MyStructElement2 := "MyTag";	(* Assignment of a tag to two structure elements*)
#MyArray[2] := #MyArray[32] := "MyTag";	(* Assignment of a tag to two ARRAY elements*)
"MyTag1" := "MyTag2" := #MyArray[1,4];	(* Assignment of an ARRAY element to two tags*)
#MyString[2] := #MyString[3]:= #MyOtherString[5];	(* Assignment of a STRING element to two STRING elements*)

The following table shows examples of combined value assignments:

SCL	
`"MyTag1" += "MyTag2";`	`(* "MyTag1" and "MyTag2" are added and the result of the addition is assigned to "MyTag1".*)`
`"MyTag1" -= "MyTag2" += "MyTag3";`	`(* "MyTag2" and "MyTag3" are added. The result of the addition is assigned to ""MyTag2"" and then subtracted from "MyTag1". The result is assigned to "MyTag1".*)`
`#MyArray[2] += #MyArray[32] += "MyTag";`	`(* The ARRAY element "MyArray[32]" is added to "MyTag". The result is assigned to "MyArray[32]". The two ARRAY elements are then added and the result is assigned to the ARRAY element "MyArray[2]". The data types must be compatible for this operation.*)`
`#MyStruct.MyStructElement1 /= #MyStruct.MyStructElement2 *= "MyTag";`	`(* The structure element "MyStructElement2" is multiplied by "MyTag". The result is assigned to "MyStructElement2". The structure element "MyStructElement1" is then divided by "MyStructElement2" and the result is assigned to "MyStructElement1". The data types must be compatible for this operation.*)`

See also

Operators and operator precedence (Page 5001)

11.14.1.8 Calculating with floating-point numbers (REAL and LREAL) in SCL

Representation of the accuracy of floating-point numbers

The data type REAL, for example, is specified and calculated in the program with an accuracy of 6 decimal places. For the calculation of floating-point numbers (REAL and LREAL), it should be noted that this accuracy applies in general to each individual step of the calculation.

The exponents are adjusted when floating-point numbers are added and subtracted. The base and the exponents are thus equal during the addition and subtraction and only the mantissas are added. For additional information on the structure of floating-point numbers, refer to "See also".

Programming example

In the following programming example, you are to perform a calculation in which two operands of the data type REAL are to be added and one is to be subtracted. In the next step of the calculation, the constant 1 is divided by the previous result. To do this, create a global data block in which you declare your operands and a function in which you program the calculation operations.

Calculation formulas

```
y = a + b - c
Z = 1/y
```

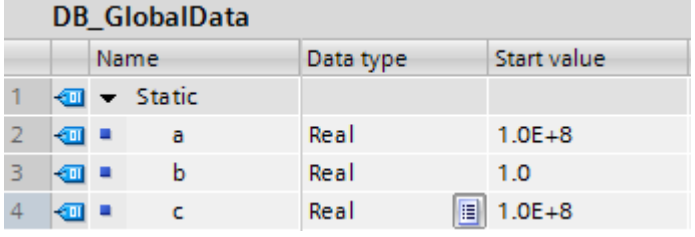
The operands are stored with the following values:

Operand	Value	REAL value
a	100 000 000	$1.000000*10^8$
b	1	$1.000000*10^0$
c	100 000 000	$1.000000*10^8$

Procedure

Create the data block "DB_GlobalData":

1. Double-click the "Add new block" command.
   The "Add new block" dialog box opens.
2. Click the "Data block (DB)" button.
3. Specify the name "DB_GlobalData".
4. Select "Global DB" as the type of the data block.
5. Click "OK".
6. Create the following tags in the data block and enter the corresponding start values:



The start value of both tags is 100000000.0 and is converted into 1.0E+8 according to the data type REAL.

Create an SCL function and name it "FC_Calculate".

1. Declare the block interface as follows:

FC_Calculate			
		Name	Data type
1		Input	
2		Output	
3		InOut	
4		Temp	
5		z	Real
6		y	Real
7		Constant	
8		Return	
9		FC_Calculate	Void

2. Write the following formulas in your program code and establish an online connection to see the result:

SCL

```
#y := "DB_GlobalData".a + "DB_GlobalData".b - "DB_GlobalData".c;
#z := 1/#y;
```

	#y	0.000000E+000
	"Global data".a	1.000000E+008
	"Global data".b	1.000000E+000
	"Global data".c	1.000000E+008

As you can see, the result at the operand is #y = 0, even though the number 1 is actually expected as result.

The incorrect result comes about as follows:

1. In the first step of the calculation, the operands a + b are added. The REAL values of the two operands (a = $1.000000*10^8$ and b = $1.000000*10^0$) look as follows after the exponents have been adjusted:
   a = $1.000000*10^8$ and b = $0.00000001*10^8$. The last two places of the second number (operand b) are truncated, as they can no longer be represented due to the accuracy of 6 decimal places. Therefore, a 0 instead of a 1 is added to the operand.
2. In the second calculation step, the operand C is subtracted from the result of the preceding calculation step (intermediate result = $1.000000*10^8$ - c = $1.000000*10^8$ is $0.000000e^0$).
3. If you now calculate the operand z in the next calculation step, you try to divide by zero.

	#z	16#7F800000
	#y	0.000000E+000

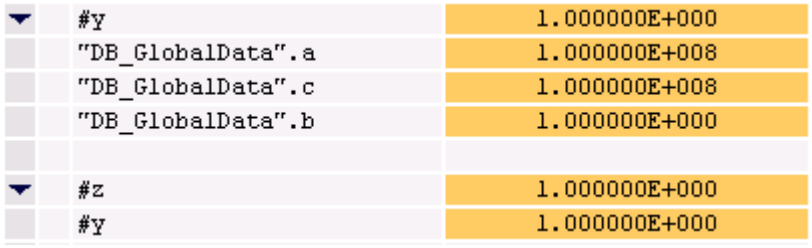
1. Possible solution

To work around such cases, you can simply adjust your calculation formula. Write the formula as follows instead:

Calculation formulas

```
y = a - c + b
Z = 1/y
```

Since the result 0.000000e^{0} is available in this case after the first calculation step (operand a - c), the addition of the REAL number in the second calculation step (intermediate result + b) leads to the correct result (y = 0.000000*10^{0}·+ 1.000000*10^{0} = 1.000000*10^{0}).



#y	1.000000E+000
"DB_GlobalData".a	1.000000E+008
"DB_GlobalData".c	1.000000E+008
"DB_GlobalData".b	1.000000E+000
#z	1.000000E+000
#y	1.000000E+000

We recommend that you check how the calculation can be made most effectively before you program a calculation.

2. Possible solution

To calculate the above named formulas, use the LREAL data type instead of the REAL data type. Since the data type is processed with an accuracy of 15 decimal places, this problem does not even arise.

1. In the global data block "DB_GlobalData", create three new tags with the same values, each with the data type LREAL.

DB_GlobalData

	Name	Data type	Start value
1	Static		
2	a	Real	1.0E+8
3	a_LREAL	LReal	100000000.0
4	b	Real	1.0
5	b_LREAL	LReal	1.0
6	c	Real	1.0E+8
7	c_LREAL	LReal	100000000.0

2. In the block interface of the FC "FC_Calculate", also declare two new tags with the data type LREAL.

FC_Calculate

	Name	Data type
1	Input	
2	Output	
3	InOut	
4	Temp	
5	z_LREAL	LReal
6	z	Real
7	y_LREAL	LReal
8	y	Real
9	Constant	
10	Return	
11	FC_Calculate	Void

3. Use the new LREAL tags for the formulas in your program code and establish an online connection to see the result.

SCL

```
#y_LREAL := "DB_GlobalData".a_LREAL + "DB_GlobalData".b_LREAL -
"DB_GlobalData".c_LREAL;
#z_LREAL := 1/#y_LREAL;
```

▼	#y_LREAL	1.000000000000000E+000
	"DB_GlobalData".a_LREAL	1.000000000000000E+008
	"DB_GlobalData".b_LREAL	1.000000000000000E+000
	"DB_GlobalData".c_LREAL	1.000000000000000E+008
▼	#z_LREAL	1.000000000000000E+000
	#y_LREAL	1.000000000000000E+000

See also

REAL (Page 2257)

LREAL (Page 2258)

Invalid floating-point numbers (Page 2259)

11.14.1.9 Calculating with constants in SCL

Interpretation of typed and non-typed constants

Constants are data with a fixed value that cannot change during program runtime. Constants can be read by various program elements during the execution of the program but cannot be overwritten. There are defined notations for the value of a constant, depending on the data type and data format. A distinction is made between the typed and non-typed notation.

We recommend that you do not mix typed and non-typed constants within mathematical functions as you may otherwise be faced with unwanted implicit conversions and thus end up with incorrect values.

Programming example

In the following programming example, you see a calculation operation with a typed and a non-typed constant.

1. Create an SCL function block and name it "FB_MathsFunctions".
2. Declare the "Variable_DINT" tag in the "Temp" section of the block interface.

FB_MathsFunctions		
	Name	Data type
1	Input	
2	Output	
3	InOut	
4	Static	
5	Temp	
6	Variable_DINT	DInt
7	Constant	

3. Write the following program code:
   Variable_DINT := INT#1 +50000;
   In this math operation, the typed constant INT#1 and the non-typed constant 50000 are to be added. The non-typed constant 50000 is underlined in yellow in the software to indicate that the constant value is outside the permitted range of the data type INT.

```
1
2  #Variable_DINT := INT#1 + 50000;
```

To see the result, go online.

1. Compile the SCL function block "FB_MathsFunctions" by right-clicking the command "Compile > Software (only changes)" to execute it.
2. Download the block with the command "Download to device > Software (only changes)".
3. Go online and monitor your block.

#Variable_DINT	-15535

The data type of the typed constant defines the data type of the addition. This means that the addition is performed in the data type area INT. In the first step, the non-typed constant 50000 is implicitly converted into the data type INT. However, the conversion leads to a negative value (-15536). This value is then added to the typed constant (INT#1). The result is -15535. Since the tag to which the result of the addition is to be written is declared with the data type DINT, the number -15535 is implicitly converted into the data type DINT and written to the tag "Variable_DINT". However, the result remains negative.

1. Possible solution

One option for avoiding this undesired result is to type both constants. If you type both constants, the longer data type determines the calculation operation.

1. Write the following program code in the "FB_MathsFunctions" SCL function block:

```
3
4  #Variable_DINT := INT#1 + DINT#50000;
```

   In this calculation operation, the typed constant INT#1 and the typed constant DINT#50000 are to be added.

To see the result, go online.

1. Compile the SCL function block "FB_MathsFunctions" by right-clicking the command "Compile > Software (only changes)" to execute it.
2. Download the block with the command "Download to device > Software (only changes)".
3. Go online and monitor your block.

	#Variable_DINT	50001

   The constant INT#1 is converted into the DINT data type and the addition of the two constants is executed in the DINT data type area.

2. Possible solution

Another option for avoiding this undesired result is not to type both constants. If you do not type both constants, these are then interpreted as the widest possible data type on the current CPU. This means that on an S7-1500 series CPU, the two constants are interpreted as LINT data type.

1. Write the following program code in the "FB_MathsFunctions" SCL function block:

```
1
2  #Variable_DINT := 1 + 50000;
```

   In this calculation operation, the non-typed constant 1 and the non-typed constant 50000 are to be added.

To see the result, go online.

1. Compile the SCL function block "FB_MathsFunctions" by right-clicking the command "Compile > Software (only changes)" to execute it.
2. Download the block with the command "Download to device > Software (only changes)".
3. Go online and monitor your block.



The constants 1 and 50000 are interpreted as LINT data type and the result of the addition is again converted into the DINT data type.

See also

Overview of the valid data types (Page 2239)

11.14.2 Settings for SCL

11.14.2.1 Overview of the settings for SCL

Overview

The following tables show the settings you can make for SCL:

Editor settings

Group	Setting	Description
View	Keyword highlighting	Notation used to represent the keywords of the programming language. You can choose between uppercase and lowercase letters or a notation corresponding to the conventions of the Pascal programming language.

Default settings for new blocks

If you create new blocks, the following settings are set as default values. You can change these in the block properties at a later point in time.

Group	Setting	Description
Compile	Create extended status information	Allows all tags in a block to be monitored. The memory requirements of the program and execution times increase, however, with this option.
	Check ARRAY limits[1)]	Checks at runtime whether array indices are within the declared range for an ARRAY. If an array index exceeds the permissible range, the enable output ENO of the block is set to "0".
	Set ENO automatically	Checks at runtime whether errors occur in the processing of certain instructions. If a runtime error occurs, the enable output ENO of the block is set to "0".
[1)]For CPUs of the S7-300/400 series: When the ARRAY limits are violated, the enable output ENO is set to FALSE. For CPUs of the S7-1200/1500 series: When the ARRAY limits are violated, the enable output ENO is not set to FALSE. See "Addressing ARRAY components (Page 2292)" for error query options.		

See also

Changing the settings (Page 5014)

11.14.2.2 Changing the settings

Procedure

To change the settings, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.
3. Change the settings.

Result

The change will be loaded directly, there is no need to save it explicitly.

See also

Overview of the settings for SCL (Page 5013)

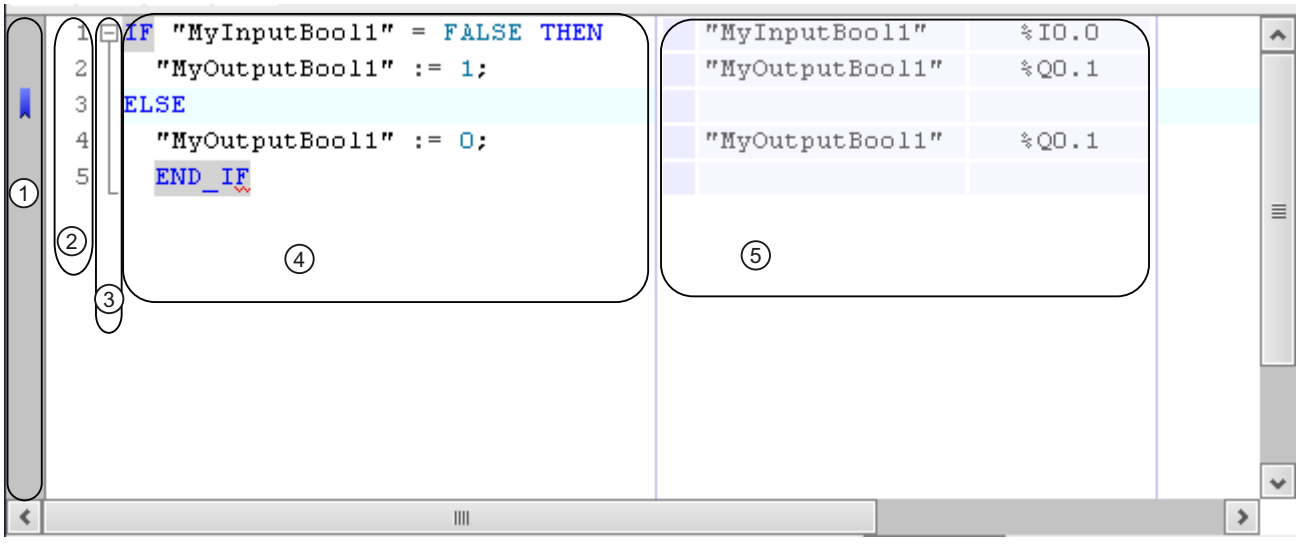
11.14.3 The programming window of SCL

11.14.3.1 Overview of the programming window

Function

The programming window is the work area, where you enter the SCL program.

The following figure shows the programming window of SCL:



The programming window consists of the following sections:

Section	Meaning
① Sidebar	You can set bookmarks and breakpoints in the sidebar.
② Line numbers	The line numbers are displayed to the left of the program code.
③ Outline view	The outline view highlights related code sections.
④ Code area	You edit the SCL program in the code area.
⑤ Display of the absolute operands	This table shows the assignment of symbolic operands to absolute addresses.

See also

Customizing the programming window (Page 5016)

Formatting SCL code (Page 5017)

Expanding and collapsing sections of code (Page 5018)

Navigate to definitions (Page 5019)

Using bookmarks (Page 5020)

11.14.3.2 Customizing the programming window

Introduction

You can customize the appearance of the programming window and the program code in the following way:

- By setting the font, size and color
- By setting the tab spacing
- By displaying the line numbers
- By showing or hiding the absolute operands

Setting the font, size and color

To set the font, size and color, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General > Script/text editors" group.
3. Select the desired font and font size or choose a font color for the individual language elements.

Setting the tab spacing

To provide a better overview of the program, lines are indented according to syntax. Define the depth of indentation with the tab spacing.

To set the tab spacing, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General > Script/text editors" group.
3. Set the tab spacing.

Show line numbers

To display the line numbers, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "General > Script/text editors" group.
3. Select the "Show line numbers" option.

Show or hide the absolute operands

You can show the assignment of symbolic and absolute operands in a table next to the program code, if required.

To hide or show the display of the absolute operands, follow these steps:

1. Click the "Absolute/symbolic operands" icon in the toolbar.
   The display of the absolute operands appears.
2. To move the display, click the table and drag it to the desired position while holding down the mouse button.
3. To change the width of the table, click on the right or left table border and drag it to the right or left while holding down the mouse button.

See also

Overview of the programming window (Page 5015)

Formatting SCL code (Page 5017)

Expanding and collapsing sections of code (Page 5018)

Navigate to definitions (Page 5019)

Using bookmarks (Page 5020)

11.14.3.3 Formatting SCL code

Introduction

To make the program clearer, you can indent or outdent individual lines manually or format code sections. Note the following information about formatting code sections:

- The type of formatting is based on the general settings for indents, but at least the line or the section is always indented. If you selected the setting "Smart", unnecessary spaces within the SCL instruction are also removed.
- Only syntactically correct code sections can be formatted.
- If you place the insertion point in the first or last line of an instruction for program control, for example in an IF instruction in the line with the "IF", the entire instruction is formatted.
- If you select text, only the selected text is formatted.

Indenting or outdenting lines

To indent or outdent individual lines, follow these steps:

1. Click on the line you want to indent or outdent.
2. Press the "Indent text", "Outdent text" button into the toolbar of the programming editor.

Note

You can set the width of the indent in "Options > Settings".

Formatting code sections

To format code sections, follow these steps:

1. Select the text that you want to format or place the insertion point in the appropriate line.
2. Select the "Format selected text automatically" button into the toolbar of the programming editor.

See also

Overview of the programming window (Page 5015)

Customizing the programming window (Page 5016)

Expanding and collapsing sections of code (Page 5018)

Overview of the script and text editor settings (Page 232)

Navigate to definitions (Page 5019)

Using bookmarks (Page 5020)

11.14.3.4 Expanding and collapsing sections of code

Introduction

SCL instructions can span several lines. Examples for this are program control instructions or block calls.

Such instructions that belong together are identified as follows:

- An outline view between the display line number and the program code marks the entire code section.
- When you select the opening keyword, the closing keyword is automatically highlighted.

To improve clarity, you can expand or collapse sections of code that belong together in the outline display. The selected outline display is retained when you close the block or the project so that, the next time you open the block, the sections of code are displayed in exactly the same way as they were when you closed it.

Procedure

To expand or collapse the code section, follow these steps:

1. Click the minus sign in the outline view.
   The code section closes.
2. Click the plus sign in the outline view.
   The code section opens.

See also

Overview of the programming window (Page 5015)

Customizing the programming window (Page 5016)

Formatting SCL code (Page 5017)

Navigate to definitions (Page 5019)

Using bookmarks (Page 5020)

11.14.3.5 Navigate to definitions

You can use elements in your program code that were defined in other editors, for example, tags or data blocks (DBs) of called function blocks (FBs). To view the definition instances in the corresponding editors, you can specifically navigate to these instances.

Procedure

To navigate to the definition of a code element, follow these steps:

1. Right-click on the code element.
2. Select the "Go to > Definition" command in the shortcut menu.
   The editor in which the code element was defined opens and the definition instance is displayed.

Or:

1. Press and hold down the <Ctrl> key.
2. Move the mouse pointer over your program code.
   If the mouse pointer moves across a code element whose definition can be displayed, it is shown as underlined and the name of the element turns into a link.
3. Click on the link.
   The editor in which the code element was defined opens and the definition instance is displayed.

See also

Overview of the programming window (Page 5015)

Customizing the programming window (Page 5016)

Formatting SCL code (Page 5017)

Expanding and collapsing sections of code (Page 5018)

Using bookmarks (Page 5020)

11.14.3.6 Using bookmarks

Basics of bookmarks

Function

You can use bookmarks to mark program locations in extensive programs so that you can find them quickly later if they need revising. Bookmarks are displayed in the sidebar of the programming window. You can navigate between multiple bookmarks within a block using menu commands.

Bookmarks are saved with the project and are therefore available for anyone who wants to edit the block. However, they are not loaded to a device.

Bookmarks are not evaluated when blocks are compared.

See also

Setting bookmarks (Page 5020)

Navigating between bookmarks (Page 5021)

Deleting bookmarks (Page 5021)

Setting bookmarks

Requirement

The SCL block is open.

Procedure

To set a bookmark, follow these steps:

1. Right-click on the desired line in the sidebar.
2. Select the "Bookmarks > Set" command in the shortcut menu.

Or:

1. Click on the line in which you want to place the bookmark.
2. Click the "Set/delete bookmark" button in the toolbar.

Or:

1. Hold down the <Ctrl> key.
2. Click on the line in the sidebar in which you want to place the bookmark.

Result

A bookmark is placed in the program code.

See also

Basics of bookmarks (Page 5020)

Navigating between bookmarks (Page 5021)

Deleting bookmarks (Page 5021)

Navigating between bookmarks

Requirement

Several bookmarks are set in a block.

Procedure

To navigate between bookmarks, follow these steps:

1. Set the insertion cursor in the program code.
2. In the "Edit" menu, select the "Go to > Next bookmark" or "Go to > Previous bookmark" command.

Or:

1. Set the insertion cursor in the program code.
2. In the toolbar of the programming editor, click the "Go to next bookmark", "Go to previous bookmark" button.

Or:

1. Click in the sidebar.
2. Select the "Bookmarks > Next" or "Bookmarks > Previous" command in the shortcut menu.

Result

The line with the bookmark is highlighted.

See also

Basics of bookmarks (Page 5020)

Setting bookmarks (Page 5020)

Deleting bookmarks (Page 5021)

Deleting bookmarks

You can delete individual bookmarks or all bookmarks from the block or the CPU.

Deleting individual bookmarks

To delete an individual bookmark, follow these steps:

1. Right-click in the sidebar on the line in which you want to delete the bookmark.
2. Select the "Bookmarks > Remove" command in the shortcut menu.

Or:

1. Click on the line in which you want to delete the bookmark.
2. In the "Edit" menu, select the "Bookmarks > Remove" command.

Or:

1. Click on the line in which you want to delete the bookmark.
2. Click the "Set/delete bookmark" button in the toolbar.

Deleting all bookmarks from the block

To delete all bookmarks from the block, follow these steps:

1. Right-click in the sidebar.
2. Select the "Bookmarks > Delete all from block" command in the shortcut menu.

Or:

1. In the "Edit" menu, select the "Bookmarks > Delete all from block" command.

See also

Basics of bookmarks (Page 5020)
Setting bookmarks (Page 5020)
Navigating between bookmarks (Page 5021)

11.14.4 Entering SCL instructions

11.14.4.1 Rules for SCL instructions

Instructions in SCL

SCL recognizes the following types of instructions:

- Value assignments
  Value assignments are used to assign a tag a constant value, the result of an expression or the value of another tag.
- Instructions for program control
  Instructions for program control are used to implement program branches, loops or jumps.
- Additional instructions from the "Instructions" task card
  The "Instructions" task card offers a wide selection of standard instructions that you can use in your SCL program.
- Block calls
  Block calls are used to call up subroutines that have been placed in other blocks and to further process their results.

Rules

You need to observe the following rules when entering SCL instructions:

- Instructions can span several lines.
- Each instruction ends with a semicolon (;).
- No distinction is made between upper and lower case.
- Comments serve only for documentation of the program. They do not affect the program execution.

Examples

The following examples shows the various types of instructions:

```
SCL
// Example of a value assignment
"MyTag":= 0;
// Example of a block call
"MyDB"();
// Example of a program control instruction
WHILE "Counter" < 10 DO
    "MyTAG" := "MyTag" + 2;
END_WHILE;
```

See also

Basics of SCL (Page 4991)

11.14.4.2 Entering SCL instructions manually

Requirement

An SCL block is open.

Procedure

To enter SCL instructions, follow these steps:

1. Enter the syntax of the instruction using the keyboard.
   You are supported by the auto-complete function when performing this task. It offers all the instructions and operands that are allowed at the current location.
2. Select the required instruction or the desired operand from the auto-complete function.
   If you select an instruction that requires specification of operands, placeholders for the operands are inserted into the program. The placeholders for the operands are highlighted in yellow. The first placeholder is selected.
3. Replace this placeholder with an operand.
4. Use the <TAB> key to navigate to all other placeholders and replace them with operands.

Note

You can also drag-and-drop a defined operand from the PLC tag table or from the block interface into the program. To replace an operand that has already been inserted, hover the mouse pointer briefly over the operand to be replaced before releasing the mouse button. This selects the operand and when you release the mouse button it is replaced by the new operand.

Result

The instruction is inserted.

The programming editor performs a syntax check. Incorrect entries are displayed in red and italics. In addition, you also receive a detailed error message in the inspector window.

See also

Using autocompletion in textual programming languages (Page 4707)

Expanding and reducing the parameter list (Page 5048)

Surround program code with structure elements (Page 5026)

Data type conversion for S7-1200 (Page 2465)

11.14.4.3 Inserting SCL instructions using the "Instructions" task card

The "Instructions" task card offers a wide selection of instructions that you can use in your SCL program. The SCL-specific instructions for program control are available in the "Instructions" task card.

Requirement

An SCL block is open.

Procedure

To insert SCL instructions into a program using the "Instructions" task card, follow these steps:

1. Open the "Instructions" task card.
2. To insert the instruction, select one of the following steps:
   - Navigate to the SCL instruction you want to insert and drag-and-drop it to the required line in the program code. The insertion location is highlighted by a green rectangle.
   - Select the location in the program code where you want to insert the instruction and then double-click on the instruction you want to insert.

   The instruction is inserted in the program. The placeholders for the operands are highlighted in yellow. A light yellow indicates the optional parameters that you do not necessarily have to interconnect. A darker yellow indicates the mandatory parameters that you must interconnect. The first placeholder is selected.
3. Replace this placeholder with an operand. You can also drag a tag from the interface or the PLC tag table with drag-and-drop to the placeholder.
4. Use the <TAB> key to navigate to all other placeholders and replace them with operands.

Result

The instruction is inserted.

The programming editor performs a syntax check. Incorrect entries are displayed in red and italics. In addition, you also receive a detailed error message in the inspector window.

See also

Using autocompletion in textual programming languages (Page 4707)

Expanding and reducing the parameter list (Page 5048)

Surround program code with structure elements (Page 5026)

Data type conversion for S7-1200 (Page 2465)

11.14.4.4 Surround program code with structure elements

You have the option of surrounding your program code with the following structure elements:

- Ranges
  - REGION
- Control structures
  - IF ... THEN
  - CASE ... OF ...
  - FOR ... TO ... DO ...
  - WHILE ... DO ...
- Comment section
  - /**/

Procedure

To surround your program code with a structure element, proceed as follows:

1. Select the program code that you want to surround with a structure element.
2. Right-click the selected program code.
3. Select the required structure element under "Surround with" in the shortcut menu.
4. Alternatively, you can also select the structure element in the favorites or in the "Instructions" task card in the "Simple instruction > Program control" pane.

See also

Entering SCL instructions manually (Page 5024)

Inserting SCL instructions using the "Instructions" task card (Page 5024)

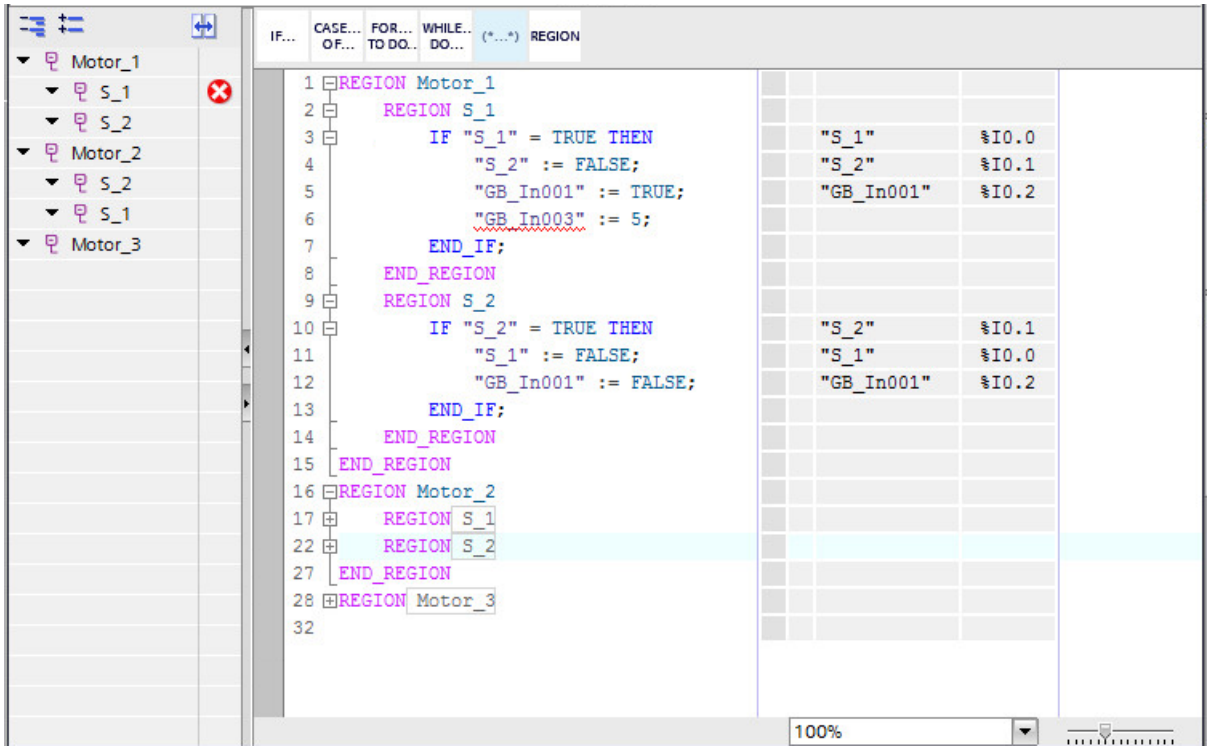
11.14.4.5 Working with regions

Using regions

Function

In SCL, you can structure your program code in regions. This improves the clarity of your program because you combine related parts of the program and can expand and collapse regions as needed. A region overview is available for fast navigation in the regions. The region overview appears next to the program window and can be displayed or hidden as needed.

The following figure shows the region overview and the programming window:



The region overview also shows the regions in which there are syntax errors. This allows you to quickly navigate to faulty code positions and rectify errors.

To summarize a program part in a region, enclose it with the keywords "REGION" and "END_REGION". Depending on your settings in the keyword highlighting, the notation of the keywords can also be "region" and "end_region" or "Region" and "End_Region". The keywords are automatically converted to the notation that corresponds to the current setting. To find "REGION" and "END_REGION" keywords that belong together, click either "REGION" or "END_REGION". Both keywords are highlighted in color as a result.

You can give each region a name. This has the following advantages:

- The name is also visible when a region is collapsed. This allows you to keep an overview of your program code.
- You can identify the regions in the region overview more easily and navigate to specific regions.

Specifying a name is optional.

Syntax

Use the following syntax to combine the program parts into regions:

```
REGION <Name>
     <Instructions>
END_REGION
```

Note the following information when using regions:

- The keywords "REGION" and "END_REGION" must be at the start of a line. Only spaces are allowed before the keywords.
- All characters after the keyword "END_REGION" are considered to be a comments and therefore have no effect on the execution of the program.
- All instructions must be completed within a region.
- Within CASE statements the instructions can be enclosed by regions. The constants which are used for comparison of the CASE statement must be outside the regions, however. The regions may not cover several branches.
  The following example shows the permitted use of regions within a CASE statement:

```
CASE <Expression> OF
  <Constant1> : REGION <Instructions1> END_REGION
  <Constant2> : REGION <Instructions2> END_REGION
ELSE REGION <Instructions0> END_REGION
END_CASE
```

  The following example shows non-permitted uses of regions within CASE statements:

```
CASE <Expression> OF
 REGION <Constant1> : <Instructions1> END_REGION
 <Constant2> : REGION <Instructions2> END_REGION
ELSE <Instructions0>
END_CASE
CASE <Expression> OF
 <Constant1> : <Instructions1> REGION
 <Constant2> : END_REGION <Instructions2>
ELSE <Instructions0>
END_CASE
```

  In these cases, the constants within a region are interpreted as jump labels which can be reached with GOTO, but which are not are taken into consideration by the CASE statements.

You also have the option to nest regions. However, make sure that regions within other regions are successfully completed:

```
REGION <Name>
     <Instructions>
     REGION <Name>
           <Instructions>
     END_REGION
     <Instructions>
     REGION <Name>
          REGION
               <Instructions>
          END_REGION
          REGION <Name>
               <Instructions>
          END_REGION
     END_REGION
END_REGION
```

See also

Inserting regions (Page 5029)

Copying and inserting regions (Page 5030)

Navigating in regions (Page 5031)

Deleting regions (Page 5033)

Inserting regions

You have the following options to insert regions in your program code:

- You can insert regions manually via the keyboard.
- You can insert regions via the "Instructions" task card. You can hereby insert either empty regions or surround existing program code with regions.
- You can insert regions via the favorites You can hereby insert either empty regions or surround existing program code with regions.

Inserting regions manually via the keyboard

To insert a region into your program code manually via the keyboard, follow these steps:

1. Place the cursor at the position where you want to insert a region.
2. Enter the keyword "REGION".

   Note

   Depending on your settings in the keyword highlighting, the notation of the keywords can also be "region" and "end_region" or "Region" and "End_Region". The keywords are automatically converted to the notation that corresponds to the current setting.

3. Enter a name for the region. This step is optional.
4. Place the cursor at the end of the region.
5. Enter the keyword "END_REGION".
   The corresponding program code is combined in a region that you can expand and collapse. The new region is visible in the region overview. If you have not assigned a name, it is indicated in the region overview with the name "Unnamed".

Inserting regions via the "Instructions" task card

To insert a region in your program code via the "Instructions" task card, proceed as follows:

1. Place the cursor at the position where you want to insert an empty region or select the program code which you want to surround with a region.
2. Open the "Instructions" task card.
3. Navigate to "Control Panel > REGION".

4. Drag the instruction REGION to the position in your program code at which you want to insert a region, or insert the instructionREGION by double-clicking.
   Either an empty region is inserted or the selected program code is surrounded with a region. The new region is visible in the region overview and the placeholder "_name_" is used as name.
5. Replace the placeholder "_name_" with the required name or delete the placeholder if you do not want to assign a name. If you have assigned no name, the new region appears in the region overview with the name "Unnamed".

Insert via the favorites

To insert a region in your program code via the favorites, proceed as follows:

1. Place the cursor at the position where you want to insert an empty region or select the program code which you want to surround with a region.
2. Click the "REGION" instruction in the Favorites pane or on the Favorites bar in the program editor.
   Either an empty region is inserted or the selected program code is surrounded with a region. The new region is visible in the region overview and the placeholder "_name_" is used as name.
3. Replace the placeholder "_name_" with the required name or delete the placeholder if you do not want to assign a name. If you have assigned no name, the new region appears in the region overview with the name "Unnamed".

See also

Using regions (Page 5026)

Copying and inserting regions (Page 5030)

Navigating in regions (Page 5031)

Deleting regions (Page 5033)

Copying and inserting regions

You can also copy regions and their contents and insert them back into your program code.

Procedure

To copy regions and their contents, follow these steps:

1. Open the region overview.
2. Right-click the region you want to copy and select the "Copy" command from the shortcut menu.
3. Place the cursor at the position in your program code at which you want to insert the region.

4. Insert the region either via the "Insert" shortcut menu command or via the key combination <Ctrl+V> in your program code.
   The copied region is inserted expanded together with its contents into your program code. In addition, the region is inserted in the region overview.
5. If necessary, assign a new name for the inserted region.

Or:

1. Select the region in the program window. The region can be expanded or collapsed.
2. Copy the region either via the shortcut menu "Copy" or the key combination <Ctrl+C>.
3. Place the cursor at the position in your program code at which you want to insert the region.
4. Insert the region either via the "Insert" shortcut menu command or via the key combination <Ctrl+V> in your program code.
   The copied region is inserted expanded together with its contents into your program code. In addition, the region is inserted in the region overview.
5. If necessary, assign a new name for the inserted region.

See also

Using regions (Page 5026)

Inserting regions (Page 5029)

Navigating in regions (Page 5031)

Deleting regions (Page 5033)

Navigating in regions

You can use the region overview to navigate quickly in the regions. The region overview displays all regions that exist in your program code. In addition, it offers the following functions:

- Synchronizing the display of the regions in the region overview and in the programming window:
  You can specify whether expanding and collapsing regions affects only the region overview or the programming window, or both.
- Expanding and collapsing an individual region:
  You can expand and collapse individual regions in either the region overview or in the programming window. If you have switched on synchronization, the region is expanded or collapsed in both windows.
- Expanding or collapsing all regions:
  You can expand or collapse all regions simultaneously either in the region overview or in the programming window. If you have switched on synchronization, the regions are expanded or collapsed in both windows.

Synchronizing the display of the regions in the region overview and in the programming window

To define the synchronization of the display for the regions in the region overview and in the programming window, proceed as follows:

1. Open the region overview.
2. To switch on the synchronization of the display, click "Synchronization on/off" on the toolbar in the region overview.
   The button for the synchronization is displayed as active.
3. To switch off the synchronization of the display, click "Synchronization on/off" on the toolbar in the region overview.
   The button for the synchronization is displayed as inactive.

Expanding and collapsing an individual region

To expand or collapse an individual region, follow these steps:

1. Open the region overview.
2. In the region overview, right-click on the region that you want to expand or collapse.
3. Select the command "Expand" or "Collapse" in the shortcut menu.

Or:

1. Open the region overview.
2. In the region overview, select the region that you want to expand or collapse.
3. Press the shortcut <Ctrl+Shift+Num+> to expand or <Ctrl+Shift+Num-> to collapse.

Or:

1. In the programming window, place the cursor in the region that you want to expand or collapse.
2. Press the shortcut <Ctrl+Shift+Num+> to expand or <Ctrl+Shift+Num-> to collapse.

Depending on the synchronization settings, the region is either expanded or collapsed in the region overview or in the programming window or in both windows.

Expanding or collapsing all regions:

To expand or collapse all regions simultaneously, follow these steps:

1. Open the region overview.
2. To expand all regions, click "Expand all" on the toolbar in the region overview.
3. To collapse all regions, click "Collapse all" on the toolbar in the region overview.

Or:

1. Open the region overview.
2. Select any region in the region overview.
3. To expand all regions, press the shortcut <Ctrl+Shift+Num*>.
4. To collapse all regions, press the shortcut <Ctrl+Shift+Num/>.

Or:

1. Place the cursor in the programming window.
2. To expand all regions, press the shortcut <Ctrl+Shift+Num*>.
3. To collapse all regions, press the shortcut <Ctrl+Shift+Num/>.

Depending on the synchronization settings, all regions are either expanded or collapsed in the region overview or in the programming window or in both windows.

See also

Using regions (Page 5026)

Inserting regions (Page 5029)

Copying and inserting regions (Page 5030)

Deleting regions (Page 5033)

Deleting regions

You can remove regions from program code at any time.

Procedure

To delete a region, proceed as follows:

1. In your program code, remove the keyword "REGION" and the name of the region if you have assigned a name.
2. Delete the corresponding keyword "END_REGION" in your program code.
   The region is deleted from the program code and the region overview.

Note

As soon as you have removed either the keyword "REGION" or "END_REGION" from the program code, you can in each case recognize the respective associated keyword by the fact that it is marked with a red line for a syntax error.

See also

Using regions (Page 5026)

Inserting regions (Page 5029)

Copying and inserting regions (Page 5030)

Navigating in regions (Page 5031)

11.14.4.6 Defining the data type of an SCL instruction

Basic information on the data types of SCL instructions

Introduction

The SCL instructions that you employ for block programming use specific data types to calculate function values. Certain SCL instructions only support the use of a specific data type. You cannot change the data type for these instructions. However, most of the SCL instructions support the use of different data types. We differentiate between the following two types of such instructions:

- Instructions for which the data type of the function value is determined by the data type of the input parameters. This is the case for most instructions.
- Instructions with default data type. The instructions listed in the following table are of this type.

You will have to change the default data type if this is incompatible with the data type of the input parameter used. You can always change the data type based on the following syntax:
`_<data type>`

SCL instructions with default data type

The following table lists the SCL instructions with default data types:

Instruction	Default data type
CEIL	DINT
DECO	DWORD
FLOOR	DINT
NORM_X	REAL
PEEK	BYTE
SCALE_X	INT
TRUNC	DINT
CONCAT	STRING
T_DIFF	TIME

See also

Changing the data type of an SCL instruction (Page 5035)

Example for changing the data type of an SCL instruction (Page 5035)

Changing the data type of an SCL instruction

Procedure

Proceed as follows to insert an SCL instruction and change its data type:

1. Insert the instruction at the required point in the program using drag-and-drop.
2. Specify the operands for the instruction.
   The data type of the function value is specified based on the input parameters, or the default data type of the instruction is used.
3. Append the "_<data type>" string to the instruction name.
   "<data type>" represents the data type you need for the instruction.

See also

Basic information on the data types of SCL instructions (Page 5034)

Example for changing the data type of an SCL instruction (Page 5035)

Modifying the data types of IEC timers and IEC counters

IEC timers and IEC counters are internal system function blocks and require an instance data block. You can create the instance data blocks either as single or multi-instance. The data type of the instance data block is determined according to the associated instruction. For CPUs of the S7-1200 and S7-1500 series, you can, however execute the instructions with different data types, depending on your requirements.

If the newly set data type of the instance data block does not match the data type of the input parameter, an implicit conversion takes place if possible. If the conversion is not possible, you will receive an error message.

Procedure

To change the data type of an IEC timer or IEC-counter instance data block, proceed as follows:

1. Open the block in which you call the IEC timer or IEC counter.
   Depending on the instance type of the instance data block, there is a green-bordered box before (multi-instance) or after (single instance) the name of the instance data block.
2. Click the green-bordered box.
   A drop-down list box with the valid data types for the instance data block is opened.
3. Select the desired data type.

Example for changing the data type of an SCL instruction

Changing the default data type of the "Decode" instruction (DECO)

Data type DWORD is set as default if you insert the "Decode" instruction in the program.

```
"Tag_Result" := DECO(IN := "Tag_Value");
```

Modify the program code as follows to convert the data type from DWORD to BYTE:

```
"Tag_Result_BYTE" := DECO_BYTE(IN := "Tag_Value");
```

See also

Basic information on the data types of SCL instructions (Page 5034)

Changing the data type of an SCL instruction (Page 5035)

11.14.4.7 Displaying or hiding tag information

Introduction

Regardless of whether the operands are represented in absolute or symbolic form, you can show and hide simple or hierarchical comments used to document global tags. This information is taken from the PLC tag table.

You can display the tag information either for all the blocks or for individually opened blocks. If you display the tag information for all the blocks, the tag information for all blocks currently opened and opened in future is shown.

You can hide the tag information at any time again. If you have hidden the tag information for all blocks, you can display it again for individual ones that you have opened.

If you select the display of tag information with hierarchical comments, the comments of the higher structure levels of structured tags will also be displayed. The display is in brackets after the comment of the tags; the comments of the individual levels are separated by a period. If there is no comment at a structure level for a tag, it is omitted in the display and this is recognizable because there are two periods.

Displaying or hiding tag information for all blocks

Follow the steps below to display or hide the tag information for all blocks:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. In the area navigation, select the "PLC programming" group.
3. If you want to show the tag information, either select the "Expand" option in the "Tag information" drop-down list or the "Tag information with hierarchy" depending on whether you want to display simple or hierarchical comments.
4. If you want to hide the tag information, select the "Collapse" option in the "Tag information" drop-down list.
   The tag information is displayed or hidden for all blocks. When you open further blocks, the tag information is displayed or hidden depending on the selected setting.

Displaying or hiding tag information for an opened block

Follow the steps below to display or hide the tag information for an opened block:

1. If you want to show the tag information, either select the "Show tag information" option in the "Shows the tag information" drop-down list or the "Tag information with hierarchy" depending on whether you want to display simple or hierarchical comments.
2. If you want to hide the tag information, select the "Hide tag information" option in the "Hides tag information" drop-down list.
   The tag information is displayed or hidden.

11.14.4.8 Using Favorites in SCL

Adding SCL instructions to the Favorites

Requirement

- A block is open.
- The multipane mode is set for the "Instructions" task card or the Favorites are also displayed in the editor.

Procedure

To add SCL instructions to the Favorites, follow these steps:

1. Open the "Instructions" task card.
2. Maximize the "Basic instructions" pane.
3. Navigate in the "Basic instructions" pane to the instruction that you want to add to the Favorites.
4. Drag-and-drop the instruction into the "Favorites" pane or into the Favorites area in the program editor.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Overview of the program editor (Page 4684)

Inserting SCL instructions using Favorites (Page 5038)

Removing SCL instructions from the Favorites (Page 5038)

Inserting SCL instructions using Favorites

Requirement

- A block is open.
- Favorites are available.

Procedure

To insert an instruction into a program using Favorites, follow these steps:

1. Drag-and-drop the desired instruction from Favorites to the desired position.

Or:

1. Select the position in the program where you want to insert the instruction.
2. In the Favorites, click on the instruction you want to insert.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Overview of the program editor (Page 4684)

Adding SCL instructions to the Favorites (Page 5037)

Removing SCL instructions from the Favorites (Page 5038)

Removing SCL instructions from the Favorites

Requirement

A code block is open.

Procedure

To remove instructions from Favorites, follow these steps:

1. Right-click on the instruction you want to remove.
2. Select the "Remove instruction" command in the shortcut menu.

Note

To additionally display the Favorites in the program editor, click the "Display favorites in the editor" button in the program editor toolbar.

See also

Overview of the program editor (Page 4684)

Adding SCL instructions to the Favorites (Page 5037)

Inserting SCL instructions using Favorites (Page 5038)

11.14.4.9 Insert block calls in SCL

Basic information on the block call in SCL

Calling function blocks

Syntax of a call

The following syntax is used to call a function block as a single or multi-instance:

- Single instance:
  - If the function block originates from the project:
    `<DBName> (Parameter list)`
  - If the function block originates from the "Instructions" task card:
    `<DB name>.<Instruction name> (Parameter list)`
    or
    `<Instruction name> (Parameter list)`
- Multi-instance
  `<#Instance name> (Parameter list)`

Calling as single instance or multi-instance

Function blocks can be called either as a single instance or a multi-instance.

- Calling as a single instance
  The called function block stores its data in a data block of its own.
- Calling as a multi-instance
  The called function block stores its data in the instance data block of the calling function block.

For additional information on the types of calls, refer to "See also".

Recursive block calls

You have the option of calling a block recursively. This means a block can call itself again. Keep in mind that the call depth is limited to 24 and that you cannot use multi-instances.

Parameter list

If you call another code block from a SCL block, you can supply the formal parameters of the called block with actual parameters.

The specification of the parameters has the form of a value assignment. This value assignment enables you to assign values (actual parameters) to the parameters you have defined in the called block.

The formal parameters of the called code block are listed in brackets directly after the call. Input and in-out parameters have the assignment identifier ":=", output parameters have the assignment identifier "=>". A placeholder placed after the parameter shows the required data type and the type of the parameter.

Rules for supplying parameters

The following rules apply to supplying parameters:

- Constants, tags and expressions can be used as actual parameters.
- The assignment order is not of importance.
- The data types of formal and actual parameters match. You can also select data types for the actual parameter for which implicit data type conversion into the data type of the formal parameter is possible.
- The individual assignments are separated by commas.
- If the called block has only one parameter, it is sufficient to specify the actual parameter in the brackets. The formal parameter need not be specified.

See also

Manually inserting block calls (Page 5044)

Inserting block calls with drag-and-drop (Page 5045)

Examples for calling a function block in SCL (Page 5042)

Calling functions

Syntax of a call

The following syntax is used to call a function:

```
<Function name> (Parameter list);     //Standard call
<Operand>:=<Function name> (Parameter list); // Call in an expression
```

Function value

Functions that provide a return value can be used in any expression in place of an operand in SCL. For this reason, the return value is also known as the "function value" in SCL.

The call options of functions depend on whether the function returns a function value to the calling block.

The function value is defined in the RET_VAL parameter. If the RET_VAL parameter is of the VOID data type, then the function will not return a value to the calling block. If the RET_VAL parameter has another data type, then the function returns a function value of this data type.

In SCL, all data types are permitted for the RET_VAL parameter except ANY, ARRAY, STRUCT and VARIANT, as well as the parameter types TIMER and COUNTER.

Call options

There are two possibilities for calling functions in SCL:

- Standard call for functions with and without a function value
  With a standard call, the results of the function is made available as an output and in-out parameter.
- Call in an expression for functions with a function value
  Functions that return a function value can be used in any expression in place of an operand, for example, a value assignment.
  The function calculates the function value, which has the same name as the function and returns it to the calling block. There the value replaces the function call.
  Following the call, the results of the function is made available as a function value or as an output and in-out parameter.

Recursive block calls

You have the option of calling a block recursively. This means a block can call itself again. Keep in mind that the call depth is limited to 24 and that you cannot use multi-instances.

Parameter list

If you call another code block from a SCL block, you need to supply the formal parameters of the called block with actual parameters.

The specification of the parameters has the form of a value assignment. This value assignment enables you to assign values (actual parameters) to the parameters you have defined in the called block.

The formal parameters of the called code block are listed in brackets directly after the call. Input and in-out parameters have the assignment identifier ":=", output parameters have the assignment identifier "=>". A gray placeholder placed after the parameter shows the required data type and the type of the parameter.

Rules for supplying parameters

The following rules apply to supplying parameters to functions:

- All parameters of the function must be supplied.
- The assignment order is not of importance.
- Constants, tags and expressions can be used as actual parameters.
- The data types of formal and actual parameters match. You can also select data types for the actual parameter for which implicit data type conversion into the data type of the formal parameter is possible.

- The individual assignments are separated by commas.
- If the called block has only one parameter, it is sufficient to specify the actual parameter in the brackets. The formal parameter need not be specified.
- When you call functions in SCL, you cannot use the release mechanism via EN. Use an IF statement instead to call functions conditionally.

See also

Manually inserting block calls (Page 5044)

Inserting block calls with drag-and-drop (Page 5045)

Examples for calling functions in SCL (Page 5043)

Examples for calling a function block in SCL

Calling as a single instance

The following example shows the call of an FB as a single instance:

```
SCL
// Call as a single instance
"MyDB" (MyInput:=10, MyInout:= "Tag1");
```

Result

After the call is executed, the value determined for the "MyInout" in/out parameter is available in "Tag1" in the "MyDB" data block.

Calling as a multi-instance

The following example shows the call of an FB as a multi-instance:

```
SCL
// Call as a multi-instance
#MyFB (MyInput:= 10, MyInout:= "Tag1");
```

Result

After the "#MyFB" block is executed, the value determined for the "MyInout" in/out parameter is made available in "Tag1" in the data block of the calling code block.

See also

Calling function blocks (Page 5039)

Manually inserting block calls (Page 5044)

Inserting block calls with drag-and-drop (Page 5045)

Examples for calling functions in SCL

Standard call

The following example shows a standard function call:

```
SCL
// Standard function call
"MyFC" (MyInput := 10, MyInOut := "Tag1");
```

Result

After the "MyFC" block is executed, the value determined for the "MyInOut" in/out parameter is available in "Tag1" in the calling block and needs to be further processed there.

Call in a value assignment

The following example shows a function call in a value assignment:

```
SCL
(*Call in a value assignment, a function value was defined for "MyFC" *)
#MyOperand := "MyFC" (MyInput1 := 3, MyInput2 := 2, MyInput3 := 8.9,
MyInOut := "Tag1");
```

Result

The function value of "MyFC" is transferred to "#MyOperand".

Call in an arithmetic expression

The following example shows a function call in an arithmetic expression:

```
SCL
(*Call in a mathematical expression, a function value was defined for
"MyFC"  *)
#MyOperand := "Tag2" + "MyFC" (MyInput1 := 3, MyInput2 := 2, MyInput3 :=
8.9);
```

Result

The function value of "MyFC" will be added to "Tag2" and the result will be transferred to "MyOperand".

See also

Calling functions (Page 5040)

Manually inserting block calls (Page 5044)

Inserting block calls with drag-and-drop (Page 5045)

Manually inserting block calls

You can insert calls for functions (FCs) and function blocks (FBs). Function blocks can be called either as a single, multi or parameter instance.

See also: AUTOHOTSPOT

Inserting a call for a function (FC)

Proceed as follows to insert a function call:

1. Enter the function name.
2. Confirm your entry with the Return key.
3. Select the "Show all parameters" command from the context menu or press the key combination <Ctrl+Shift+space bar>. Alternatively, you can click "Expands/collapses the parameter list of block calls" in the function bar.
   The syntax for the function call including the parameter list is added to the SCL program. The placeholders for the actual parameters are highlighted in yellow. The first placeholder is selected.
4. Replace this placeholder with an actual parameter. You can also drag a tag from the interface or the PLC tag table with drag-and-drop to the placeholder.
5. Use the <TAB> key to navigate to all other placeholders and replace them with actual parameters. You can navigate to the previous placeholder with <Shift + TAB>.

Inserting a call for a function block (FB)

To insert a call for a function block (FB), follow these steps:

1. Enter the name of the function block.
2. Enter an opening parenthesis "(".
   The "Call options" dialog opens.
3. Confirm your entries with "OK".
   The syntax for the function block call including the parameter list is added to the SCL program. The placeholders for the actual parameters are highlighted in yellow. The first placeholder is selected.

4. In the dialog, enter whether you want to call the block as a single, multi or parameter instance.
   - If you click the "Single instance" button, enter a name for the data block that will be assigned to the call in the entry field "Name".
   - If you click the "Multi-instance" button, in the "Name in the interface" field, enter the name of the tag with which the called function block will be entered as a static tag in the interface of the calling block.
   - If you click on the "Parameter instance" button, enter the name of the in/out (InOut) parameter to which the instance should be passed during runtime in the "Name in the interface" text box.
5. If you call a block that contains monitoring, assign a ProDiag function block to the monitoring functions.
6. Replace this placeholder with an actual parameter. You can also drag a tag from the interface or the PLC tag table with drag-and-drop to the placeholder.
7. Use the <TAB> key to navigate to all other placeholders and replace them with actual parameters. You can navigate to the previous placeholder with <Shift + TAB>.

Result

The block call is inserted.

If you specify an instance data block that does not exist when calling a function block, it is created.

See also

Updating block calls (Page 5047)

Expanding and reducing the parameter list (Page 5048)

Basic information on the block call in SCL (Page 5039)

Using autocompletion in textual programming languages (Page 4707)

Inserting block calls with drag-and-drop

You can insert calls for existing functions (FC) and function blocks (FB) using a drag-and-drop operation from the project tree.

Function blocks can be called either as a single, multi or parameter instance.

See also: AUTOHOTSPOT

Requirement

The function to be called (FC) or the function block (FB) to be called is present.

Inserting a call for a function (FC)

To insert a function call using drag-and-drop, follow these steps:

1. Drag the function from the project tree into the program.
   The syntax for the function call including the parameter list is added to the SCL program. The placeholders for the actual parameters are highlighted in yellow. The first placeholder is selected.
2. Replace this placeholder with an actual parameter. You can also drag a tag from the interface or the PLC tag table with drag-and-drop to the placeholder.
3. Use the <TAB> key to navigate to all other placeholders and replace them with actual parameters. You can navigate to the previous placeholder with <Shift + TAB>.

Inserting a call for a function block (FB)

To insert a call for a function block (FB) using drag-and-drop, follow these steps:

1. Drag the function block from the project tree and drop it into the program.
   The "Call options" dialog opens.
2. In the dialog, enter whether you want to call the block as a single, multi or parameter instance.
   - If you click the "Single instance" button, enter a name for the data block that will be assigned to the call in the entry field "Name".
   - If you click the "Multi-instance" button, in the "Name in the interface" field, enter the name of the tag with which the called function block will be entered as a static tag in the interface of the calling block.
   - If you click on the "Parameter instance" button, enter the name of the in/out (InOut) parameter to which the instance should be passed during runtime in the "Name in the interface" text box.
3. If you call a block that contains monitoring, assign a ProDiag function block to the monitoring functions.
4. Confirm your entries with "OK".
   The syntax for the function block call including the parameter list is added to the SCL program. The placeholders for the actual parameters are highlighted in yellow. The first placeholder is selected.
5. Replace this placeholder with an actual parameter. You can also drag a tag from the interface or the PLC tag table with drag-and-drop to the placeholder.
6. Use the <TAB> key to navigate to all other placeholders and replace them with actual parameters. You can navigate to the previous placeholder with <Shift + TAB>.

Result

The block call is inserted.

If you specify an instance data block that does not exist when calling a function block, it is created.

See also

Updating block calls (Page 5047)

Expanding and reducing the parameter list (Page 5048)

Basic information on the block call in SCL (Page 5039)

Using autocompletion in textual programming languages (Page 4707)

Updating block calls

If interface parameters of a called block are changed, the block call can no longer be executed correctly. You can avoid such inconsistent block calls by updating the block calls.

You have the following options for updating the block calls:

- Explicit updating of all inconsistent block calls in the programming editor.
  The inconsistent block calls within the open block are updated. The following actions are carried out in the process:
  - New parameters are added. Please note, however, that the parameters are hidden for function blocks (FBs) and the parameters are supplied via the corresponding instance data block (DB). If required, you can show the parameters using the shortcut menu command "Show all parameters".
  - Deleted parameters are not removed. If necessary, expand the parameter list to remove deleted parameters manually.
  - Renamed parameters get the new parameter names.
- Explicit updating of a block call in the programming editor.
  The inconsistent call of this block is updated at all call locations. The following actions are carried out in the process:
  - New parameters are added.
  - Deleted parameters are not removed. If necessary, expand the parameter list to remove deleted parameters manually.
  - Renamed parameters get the new parameter names.
- Implicit updating during compilation.
  All block calls in the program as well as the used PLC data types will be updated. Make sure that you manually remove deleted parameters before the compilation process and supply all new formal parameters with actual parameters when you call functions.

Updating all inconsistent block calls in the programming editor

To update all block calls in a block, follow these steps:

1. Open the calling block in the programming editor.
2. Click "Update inconsistent block calls" in the toolbar.
   All inconsistent calls are updated. If necessary, supply new formal parameters of functions (FCs) with actual parameters.

Updating a specific block call in the programming editor

To update a specific block call in the programming editor, follow these steps:

1. Open the calling block in the programming editor.
2. Right-click on the block call that you want to update.
3. Select the "Update block call" command in the shortcut menu.
4. If parameters were added, enter the values for the new block parameters.

Note

Note that the "Update block call" command is only available as long as you did not previously update all block calls in the editor with the "Update inconsistent block calls" command.

Updating block calls during compilation

To implicitly update all block calls and uses of PLC data types during compilation, follow these steps:

1. Open the project tree.
2. Select the "Program blocks" folder.
3. Select the "Compile > Software (rebuild all blocks)" command in the shortcut menu.

See also

Manually inserting block calls (Page 5044)

Inserting block calls with drag-and-drop (Page 5045)

Expanding and reducing the parameter list (Page 5048)

11.14.4.10 Expanding and reducing the parameter list

In SCL, if you call blocks or insert instructions that are system-internal function blocks, the syntax and the parameter list with the placeholders for the actual parameters are inserted in the SCL program. To make the program code easier to read, the unused optional parameters are removed from the parameter list when you edit other instructions. You can restore these at any time. You can also explicitly reduce the parameter list when you have finished assigning the parameters.

Expanding the parameter list

To expand the parameter list, follow these steps:

1. Right-click in the block call or the instruction.
2. Select the "Expand parameter list" command from the shortcut menu or press the key combination <Ctrl+Shift+Space bar>.
   The parameter list is displayed in full again.

Reducing the parameter list

To reduce the parameter list, follow these steps:

1. Right-click in the block call or the instruction.
2. Select the "Reduce parameter list" command from the shortcut menu or press the key combination <Ctrl+Shift+Space bar>.
   All unused optional parameters are hidden.

See also

Entering SCL instructions manually (Page 5024)

Inserting SCL instructions using the "Instructions" task card (Page 5024)

Manually inserting block calls (Page 5044)

Inserting block calls with drag-and-drop (Page 5045)

11.14.4.11 Insert pragmas (S7-1200, S7-1500)

You have the option of assigning validities to your program code using pragmas. To do so, surround the program code with Pragma keywords and assign at least one parameter. Optionally, you can assign additional parameters to define the content. Note that the character "$" is used as escape sequence in string constants and cannot therefore be used as normal sign for the parameter name.

The general syntax for pragmas is as follows:

```
{PRAGMA_BEGIN 'Param1', 'Param2', ... , 'ParamN'}
      //Programcode
{PRAGMA_END}
```

Pragmas can also be nested. However, make sure that pragmas within other pragmas are successfully completed. Pragmas can be expanded and collapsed as needed.

Note

Pragmas are currently evaluated only in SiVArc. The "SIVARC" parameter and another parameter must be assigned.

For more information, refer to the SiVArc help.

Procedure

To insert a pragma, follow these steps:

1. Place the cursor at the position where you want to insert a pragma.
2. Enter "{ PRAGMA_BEGIN".
3. Enter at least one parameter with single quotation marks.
4. Enter any number of additional parameters in single quotation marks. This step is optional.

5. Enter "}".
6. Place the cursor at the end of the program code that you want to surround with the pragma.
7. Enter "{ PRAGMA_END}".
   The corresponding program code is combined in a pragma.

11.14.4.12 Inserting comments

Commenting program code

You have various options for commenting SCL programs:

- Line comment
  A line comment starts with "//" and extends to the end of the line.
- Comment section
  A comment section is started with "(* and completed by "*)". It can span several lines.

Inserting a line comment

To insert line comments, follow these steps:

1. Type "//" at the position where you want to place the comment. This does not have to be the beginning of the line.
2. Enter the comment text.

Inserting a comment section

To insert a comment section, follow these steps:

1. Type "(*" at the position where you want to place the comment. This does not have to be the beginning of the line.
2. Enter the comment text.
3. Complete the comment with "*)".

Disabling one or more lines with comments

To disable program code with comments, follow these steps:

1. Select the code lines you want to comment out.
2. Click the "Comment selection" button in the editor.
   "//" is inserted at the beginning of the line in the selected lines. The code that follows is interpreted as a comment. If lines already containing a line comment are disabled, "//" is inserted as well. If these lines are enabled again, the original comments are retained.

Enabling comment lines

To enable lines that have been commented out to be enabled as code again, proceed as follows:

1. Select the code lines you want to enable.
2. Click the "Remove comment" button in the editor.
   The "//" mark for line comments at the beginning of the line is removed.

Example

The following code contains comment sections and line comments

```
(*******************************************************************************
  A description of the instructions that follow can be placed here
*******************************************************************************)
IF "MyVal1" > 0 THEN //No division by 0
    "MyReal" := "MyVal2" (* input value *) / "MyVal1" (* measured value *);
END_IF;
//Data type conversion
"MyInt" := REAL_TO_INT("MyReal");
```

See also

(*...*): Insert a comment section (Page 3397)

11.14.5 Editing SCL instructions

11.14.5.1 Selecting instructions

You can select individual instructions or all instructions of a block.

Requirement

An SCL block is open.

Selecting individual instructions

To select individual instructions, follow these steps:

1. Set the insertion mark before the first character that you want to select.
2. Press and hold down the left mouse button.
3. Move the cursor to a position after the last character that you want to select.
4. Release the left mouse button.

Selecting all the instructions of a program

To select all instructions, follow these steps:

1. In the "Edit" menu, select the "Select All" command or use the keyboard shortcut <Ctrl+A>.

Note

When you select the opening keyword of an instructing, the closing keyword is automatically highlighted.

11.14.5.2 Copying, cutting and pasting instructions

Copying an instruction

To copy an instruction, follow these steps:

1. Select the instruction you want to copy.
2. Select "Copy" in the shortcut menu.

Cutting an instruction

To cut an instruction, follow these steps:

1. Select the instruction you want to cut.
2. Select the "Cut" command in the shortcut menu.

Inserting an instruction from the clipboard

To insert an instruction from the clipboard, follow these steps:

1. Copy or cut an instruction.
2. Click on the position at which you want to insert the instruction.
3. Select "Paste" in the shortcut menu.

11.14.5.3 Deleting instructions

Requirement

An SCL block is open.

Procedure

To delete an instruction, follow these steps:

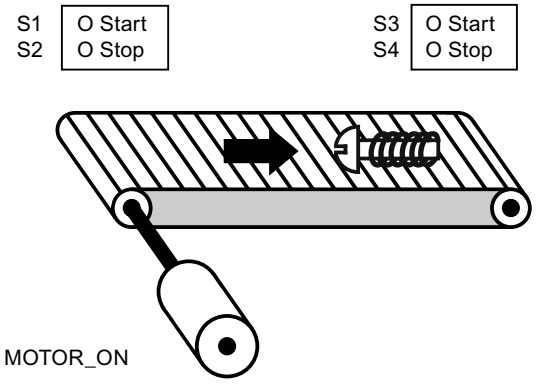
1. Select the instruction you want to delete.
2. Select the "Delete" command in the shortcut menu.

11.14.6 SCL programming examples

11.14.6.1 Example of controlling a conveyor belt

Controlling a conveyor belt

The following figure shows a conveyor belt that can be activated electrically. There are two pushbuttons at the beginning of the conveyor belt: S1 for START and S2 for STOP. There are also two pushbuttons at the end of the conveyor belt: S3 for START and S4 for STOP. It is possible to start and stop the conveyor belt from either end.



Implementation

The following table shows the definition of the tags used:

Operand	Declaration	Data type	Description
StartPushbutton_Left (S1)	Input	BOOL	Start pushbutton on the left side of the conveyor belt
StopPushbutton_Left (S2)	Input	BOOL	Stop pushbutton on the left side of the conveyor belt
StartPushbutton_Right (S3)	Input	BOOL	Start pushbutton on the right side of the convey-or belt
StopPushbutton_Right (S4)	Input	BOOL	Stop pushbutton on the right side of the convey-or belt

Operand	Declaration	Data type	Description
MOTOR_ON	Output	BOOL	Turn on the conveyor belt motor
MOTOR_OFF	Output	BOOL	Turn off the conveyor belt motor

The following SCL program shows how to implement this task:

SCL

```
IF "StartPushbutton_Left_S1" OR "StartPushbutton_Right_S3" THEN
"MOTOR_ON" := 1;
"MOTOR_OFF" := 0;
END_IF;
IF "StopPushbutton_Left_S2" OR "StopPushbutton_Right_S4" THEN
"MOTOR_ON" := 0;
"MOTOR_OFF" := 1;
END_IF;
```

The conveyor belt motor is switched on when start pushbutton "StartPushbutton_Left_S1" or "StartPushbutton_Right_S3" is pressed. The conveyor belt motor is switched off when stop pushbutton "StopPushbutton_Left_S2" or "StopPushbutton_Right_S4" is pressed.

See also

Logical expressions (Page 5000)

Basics of SCL (Page 4991)

Settings for SCL (Page 5013)

The programming window of SCL (Page 5015)

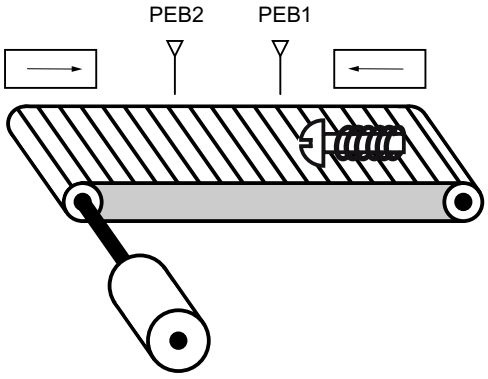
Entering SCL instructions (Page 5023)

Editing SCL instructions (Page 5051)

11.14.6.2 Example of detecting the direction of a conveyor belt

Detecting the direction of a conveyor belt

The detected running direction of the belt is indicated by a RIGHT arrow or a LEFT arrow. If additional conveyed material is approaching PEB1 from the right or PEB2 from the left, the displayed arrow is initially switched off until, after both photoelectric barriers are passed, the running direction can be detected again and the corresponding arrow displayed. For the solution of the task, 2 edge memory bits are needed that detect the signal change from "0" to "1" at the two photoelectric barriers.



Implementation

The following table shows the definition of the tags used:

Name	Declaration	Data type	Description
Photoelectric barrier PEB1	Input	BOOL	Photoelectric barrier 1
Photoelectric barrier PEB2	Input	BOOL	Photoelectric barrier 2
RIGHT	Output	BOOL	Display for movement to the right
LEFT	Output	BOOL	Display for movement to the left
Auxiliary flag PEB1	Input	BOOL	Edge bit memory 1
Auxiliary flag PEB2	Input	BOOL	Edge bit memory 2

The following SCL program shows how to implement this example:

SCL

```
// Program code for left running
IF "Photolelectric barrier PEB1" = 1 AND "Auxiliary flag PEB2" = 0 THEN
          "Auxiliary flag PEB1" := 1; // Set auxiliary flag for PEB1
          "LEFT" := 0; // Switch off arrow display left
          "RIGHT" := 0; // Switch off arrow display right
END_IF;
```

SCL

```
IF "Auxiliary flag PEB1" = 1 AND "Photoelectric barrier PEB2" = 1 THEN // The conveyor belt
is running to the left
          "LEFT" = 1;
          "RIGHT" := 0;
END_IF;

IF "LINKS" = 1 AND "Photoelectric barrier PEB2" = 0 THEN // Reset auxiliary flag for PEB1
          "Auxiliary flag PEB1" = 0
END_IF;
```

SCL

```
// Program code for right running
IF "Photolelectric barrier PEB2" = 1 AND "Auxiliary flag PEB1" = 0 THEN
          "Auxiliary flag PEB2" := 1; // Set auxiliary flag for PEB2
          "LEFT" := 0; // Switch off arrow display left
          "RIGHT" := 0; // Switch off arrow display right
END_IF;

IF "Auxiliary flag PEB2" = 1 AND "Photoelectric barrier PEB1" = 1 THEN // The conveyor belt
is running to the right
          "LEFT" = 0;
          "RIGHT" := 1;
END_IF;

IF "RIGHT" = 1 AND "Photoelectric barrier PEB1" = 0 THEN // Reset auxiliary flag for PEB2
          "Auxiliary flag PEB2" := 0;
END_IF;
```

If the photoelectric barrier "PEB1" has signal state "1" and the photoelectric barrier "PEB2" has signal state "0" at the same time, the object on the belt is moving to the left. If the photoelectric barrier "PEB2" has signal state "1" and the photoelectric barrier "PEB1" has signal state "0" at the same time, the object on the belt is moving to the right. The displays for a movement to the left or right will be turned off when the signal state at both photoelectric barriers is "0".

See also

Logical expressions (Page 5000)

Basics of SCL (Page 4991)

Settings for SCL (Page 5013)

The programming window of SCL (Page 5015)

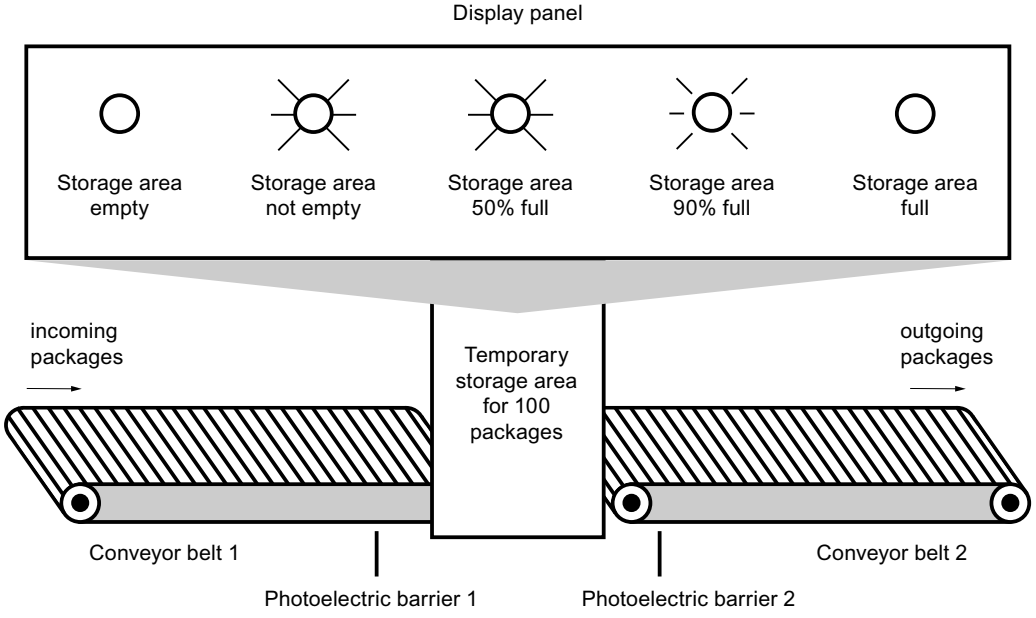
Entering SCL instructions (Page 5023)

Editing SCL instructions (Page 5051)

11.14.6.3 Example of detecting the fill level of a storage area

Detecting the fill level of a storage area

The following figure shows a system with two conveyor belts and a temporary storage area between them. Conveyor belt 1 delivers packages to the storage area. A photoelectric barrier at the end of conveyor belt 1 near the storage area detects how many packages are delivered to the storage area. Conveyor belt 2 transports packages from the temporary storage area to a loading dock where they are loaded onto trucks. A photoelectric barrier at the storage area exit detects how many packages leave the storage area to be transported to the loading dock. Five display lamps indicate the capacity of the temporary storage area.



Implementation

The following table shows the definition of the tags used:

Name	Declaration	Data type	Description
PEB1	Input	BOOL	Photoelectric barrier 1
PEB2	Input	BOOL	Photoelectric barrier 2
RESET	Input	BOOL	Reset counter
LOAD	Input	BOOL	Adjust the current counter value to the value of the PV parameter.
MAX STORAGE AREA FILL AMOUNT	Input	INT	Maximum possible number of packages in the storage area
PACKAGECOUNT	Output	INT	Number of packages in the storage area (current count value)

Name	Declaration	Data type	Description
STOCK_PACKAGES	Output	BOOL	Is set when the current count value is greater than or equal to the value of the "MAX STORAGE AREA FILL AMOUNT" tag.
STOR_EMPTY	Output	BOOL	Display lamp: Storage area empty
STOR_NOT_EMPTY	Output	BOOL	Display lamp: Storage area not empty
STOR_50%_FULL	Output	BOOL	Display lamp: Storage area 50 % full
STOR_90%_FULL	Output	BOOL	Display lamp: Storage area 90 % full
STOR_FULL	Output	BOOL	Display lamp: Storage area full
VOLUME_50	Input	INT	Comparison value: 50 packages
VOLUME_90	Input	INT	Comparison value: 90 packages
VOLUME_100	Input	INT	Comparison value: 100 packages

The following SCL program shows how to implement this example:

When a package is delivered to the storage area, the signal state at "PEB1" switches from "0" to "1" (positive signal edge). On a positive signal edge at "PEB1", the "Up" counter is enabled, and the current count value of "PACKAGECOUNT" is increased by one.

When a package is delivered from the storage area to the loading dock, the signal state at "PEB2" switches from "0" to "1" (positive signal edge). On a positive signal edge at "PEB2", the "Down" counter is enabled, and the current count value of "PACKAGECOUNT" is decreased by one.

If there are no packages in the storage area ("PACKAGECOUNT" = "0"), the "STOR_EMPTY" tag is set to signal state "1", and the "Storage area empty" lamp is switched on.

The current count value can be reset to "0" if the "RESET" tag is set to signal state "1".

If the "LOAD" tag is set to signal state "1", the current count value is set to the value of the "MAX STORAGE AREA FILL AMOUNT" tag. As long as the current count value is greater than or equal to the value of the "MAX STORAGE AREA FILL AMOUNT" tag, the "STOCK_PACKAGES" tag supplies the signal state "1".

SCL

```
"VOLUME_50" := 5; // Preassigning of the comparison value to 50 packages (for the test only
5 packages)
"VOLUME_90" := 9; // Preassigning of the comparison value to 90 packages (for the test only
9 packages)
"VOLUME_100" := 10; // Preassigning of the comparison value to 100 packages (for the test
only 10 packages)
"MAX STORAGE AREA FILL AMOUNT" := 10; // Preassigning of the maximum amount in storage area
to 100 packages (for the test only 10 packages)
```

SCL

```
"IEC_Counter_0_DB".CTUD(CU := "PEB1",
                        CD := "PEB2",
                        R := "RESET",
                        LD := "LOAD",
                        PV := "MAX STORAGE AREA FILL AMOUNT",
                        QU => "STOCK_PACKAGES",
                        QD => "STOR_EMPTY",
                        CV => "PACKAGECOUNT");
```

As long as the storage area contains packages, the "Storage area not empty" lamp is switched on.

SCL

```
"STOR_NOT_EMPTY" := NOT "STOR_EMPTY"
```

If the number of packages in the storage area is lower than 50%, the lamps for the alarms "Storage area 50% full", "Storage area 90% full" and "Storage area full" switch off.

SCL

```
IF "PACKAGECOUNT" < "VOLUME_50" THEN
"STOR_50%_FULL" := 0;
"STOR_90%_FULL" := 0;
"STOR_FULL" := 0;
END_IF;
```

If the number of packages in the storage area is greater than or equal to 50 %, the "Storage area 50 % full" lamp switches on.

SCL

```
IF "PACKAGECOUNT" >= "VOLUME_50" AND "PACKAGECOUNT <= "VOLUME_90" THEN
"STOR_50%_FULL" := 1;
"STOR_90%_FULL" := 0;
"STOR_FULL" := 0;
END_IF;
```

If the number of packages in the storage area is greater than or equal to 90 %, the "Storage area 90 % full" lamp switches on. The display lamp for 50 % full also remains on.

SCL

```
IF "PACKAGECOUNT" >= "VOLUME_90" AND "PACKAGECOUNT < "VOLUME_100" THEN
"STOR_50%_FULL" := 1;
"STOR_90%_FULL" := 1;
"STOR_FULL" := 0;
END_IF;
```

If the number of packages in the storage area reaches 100 %, the lamp for the "Storage area full" message switches on. The display lamps for 50 % and 90 % full also remain on.

SCL

```
IF "PACKAGECOUNT" >= "VOLUME_100" THEN
"STOR_50%_FULL" := 1;
"STOR_90%_FULL" := 1;
"STOR_FULL" := 1;
END_IF;
```

See also

Basics of SCL (Page 4991)

Settings for SCL (Page 5013)

The programming window of SCL (Page 5015)

Entering SCL instructions (Page 5023)

Editing SCL instructions (Page 5051)

11.15 Compiling and loading blocks

11.15.1 Compiling blocks

11.15.1.1 Basic information on compiling blocks

Introduction

A user program must first be compiled before the CPU can execute it. You need to recompile your program each time you make a change.

The following procedures take place during compilation:

- The user program is checked for syntax errors.
- Unneeded instructions are removed from the user program.
- All the block calls within the compiled blocks are checked. In case of changes to the interface of called blocks, errors will be shown in the "Compilation" tab of the information window. You have to correct these errors first.
- The blocks must be numbered uniquely in the user program. If more than one block has the same number, the blocks with number conflicts are renumbered automatically during compilation. A block will not be renumbered in the following cases:
  - The block was selected either individually or as part of a multiselection for the compilation.
  - The number assignment is set to "manual" in the properties of the block.

  Number conflicts that cannot be resolved by automatic renumbering must be corrected manually. Note the messages in the Inspector window for this.

  Note

  Renumbering know-how-protected blocks

  Automatic renumbering and manual renumbering without password of know-how-protected blocks is only possible for CPU of the series S7-1500 and S7-1200 (V4). The know-how protection must also be set with a TIA Portal version V13 SP1 or higher.

- Finally, the user program is compiled into a code that can be read by the CPU.

Compilation methods

You can start compilation in the following windows or editors:

- Compiling blocks in the project tree
  Serves to compile individual blocks or the simultaneous compilation of one or several blocks in the "Program blocks" folder.
- Compiling blocks in the program editor
  This is intended for compilation of a single open block.
- Compiling blocks in the call or dependency structure
  Used to compile individual blocks.
  See also: Call structure (Page 5129), Dependency structure (Page 5135)

Compilation options

If you are compiling blocks in project tree, you have further options:

- Software (only changes)
  All program changes of the selected blocks are compiled. If you have selected a block folder, all program changes to the bocks contained in the folder are compiled.
- Software (compile all blocks)
  All blocks are compiled. This is recommended for the first compilation and after major revisions.
- Software (reset memory reserve)
  All tags declared in the reserve area of the interface of selected blocks are moved to the standard area of the interface. Memory reserve is now available for further interface extensions.

  Note

  This option is only available for CPUs of the S7-1500 and S7-1200 V4 and higher series.

Consistency check

Changing the interfaces of blocks called or PLC data types used can result in inconsistencies between calling blocks and called blocks or between the PLC data types and the global data blocks which use these PLC data types.

To avoid such inconsistencies in the user program, the system performs an automatic consistency check before each compilation process. The time stamps are compared and compilation is then either carried out or canceled depending on the results of the comparison.

- The calling block can only be compiled if the time stamps of the interfaces of the called blocks are older than those of the calling block.
- A global data block based on a PLC data type can only be compiled correctly if the time stamp of the global data block is newer than the time stamp of the PLC data type used.
- The instance data block can only be compiled correctly if the interface time stamps for the interface of the instance data block are identical to those of the assigned function block.

If the compilation process is cancelled, an alarm is displayed in the inspector window. Update the block calls in the relevant blocks and the PLC data types in the global data blocks and then

restart compilation. The consistency check also finds know-how protected blocks which cannot be compiled. The corresponding messages will also be shown in the inspector window.

If you start loading immediately instead of first compiling, the blocks selected will be automatically compiled and the block call and global data blocks implicitly updated. Please note the following differences between the CPU families:

- S7-1200/1500: All blocks affected are loaded to ensure no inconsistencies can arise.
- S7-300/400: Only the block selected is loaded.

See also

Compiling blocks in the project tree (Page 5063)

Compiling blocks in the program editor (Page 5064)

Correcting compilation errors (Page 5065)

Block time stamps (Page 4730)

Updating block calls in LAD (Page 4900)

Updating block calls in FBD (Page 4957)

Compiling project data (Page 347)

11.15.1.2 Compiling blocks in the project tree

You can compile one block, multiple blocks or all of the blocks in the project tree.

For CPUs of the S7-1500 and S7-1200 V4 series, you can also reset the memory layout of blocks with memory reserve by running a compilation. For more information on memory reserve, refer to chapter "Loading blocks (S7-1200/1500) > Loading block changes without reinitialization".

Requirement

The project tree is open.

Compiling one or more blocks in the project tree

To compile multiple blocks in the project tree, follow these steps:

1. Open the "Program blocks" folder in project tree.
2. Select the blocks you want to compile.
3. Select the "Compile > Software (only changes)" command from the shortcut menu.

Compiling all blocks in the project tree

To compile all blocks in the "Program blocks" folder in project tree, follow these steps:

1. Select the "Program blocks" folder in the project tree.
2. You can select one of two different options for the compilation:
   - If you want to compile only the changes since the last compilation, select the "Compile > Software (only changes)" command in the shortcut menu.
   - If you want to compile all blocks completely, select the "Compile > Software (compile all blocks)" command in the shortcut menu.

Resetting memory layout (S7-1500/S7-1200 V4)

Proceed as follows to reset the memory layout of blocks:

1. Select the "Program blocks" folder, or specific blocks in this folder.
2. Select the "Compile > Software (Reset memory reserve)" command from the shortcut menu.

Result

The code for the blocks will be generated if the consistency check has been successful. Instance data blocks generated by the system which are no longer needed will be deleted.

The message under "Info > Compilation" in the inspector window reports whether the compilation was successful.

See also

Basic information on compiling blocks (Page 5061)

Compiling blocks in the program editor (Page 5064)

Correcting compilation errors (Page 5065)

Finding syntax errors in the program (Page 4712)

11.15.1.3 Compiling blocks in the program editor

Note

Note that the block is recompiled even if you have not made any changes and the time stamp of the block is modified.

Requirement

The block to be compiled is open.

Procedure

To compile a block in the program editor, follow these steps:

1. Right-click in the instruction window of the programming editor.
2. Select the "Compile" command in the shortcut menu.

Result

The code for the block is generated. Instance data blocks generated by the system which are no longer needed will be deleted.

The message under "Info > Compilation" in the inspector window reports whether the compilation was successful.

See also

Basic information on compiling blocks (Page 5061)

Compiling blocks in the project tree (Page 5063)

Correcting compilation errors (Page 5065)

11.15.1.4 Correcting compilation errors

In the Inspector window in "Info > Compile", you can see whether any compilation was successful or whether errors were detected in the program. If errors occur, you will need to correct them and then start the compilation again.

Procedure

To correct errors following compilation, follow these steps:

1. Open the error list in the Inspector window with "Info > Compile".
2. If there is one, click on the blue question mark next to the error text for information on remedying errors.
3. Double-click the error you want to correct.
   The corresponding error is highlighted.
4. Correct the error.
5. Restart compilation.

See also

Basic information on compiling blocks (Page 5061)

Compiling blocks in the program editor (Page 5064)

Compiling blocks in the project tree (Page 5063)

11.15.2 Downloading blocks for S7-1200/1500 (S7-1200, S7-1500)

11.15.2.1 Introduction to downloading blocks (S7-1200, S7-1500)

Downloading blocks to device

So that the CPU can execute the user program, the program must first be compiled and then downloaded to the device. Loading into the device can be started through different methods:

- By using the commands in the "Online" menu
- By using the shortcut menu
- By using the "Load" button in the toolbar

Depending on the selected object (such as individual blocks in the programming editor, the complete block folder or the connected CPU in the project navigation) and the command selected, you can load the following components:

- Hardware and software (only changes)
- Hardware configuration
- Software (only changes)
- Software (all)

During the loading operation, all information that is required for the reconstruction of the program, including symbolic information such as the names and comments for code and data blocks, is also loaded in the current project language. If you change the project language, you must therefore re-load the program.

The symbolic information is not loaded to the work memory, but rather to the load memory.

After the data has been loaded from a device, the symbolic information is available again in your program, which increases the readability of your program code. Please note, however, that loading to and from a device is not a substitute for storing data in an offline project, as watch tables or multi-language capability of projects cannot be reproduced by loading to and from a device.

After loading from a device, you can only display all data from know-how-protected blocks by entering the correct password.

Note

No valid hardware configuration available

If no valid hardware configuration was found on the CPU while loading the software, the existing hardware is loaded as well during a "Download to device".

Please note that, in this case, hardware components are also loaded during the download, even though you executed the command for loading software.

Note

To avoid inconsistencies between calling and called blocks, all blocks affected are compiled and loaded after each global change, such as a change in the block interface.

Note

S7-1200 Version 1.0

If you download an element of your project to the CPU, for example a program block, a data block or the hardware configuration, the CPU runs a cold restart the next time it changes to RUN mode. Apart from deleting the inputs, initializing the outputs and deleting the non-retentive memory, cold restart also deletes the retentive memory areas. All subsequent changes from STOP to RUN are warm restarts in which the retentive memory is not deleted.

Note

S7-1500

The load memory of S7-1500 series CPUs is on the SIMATIC memory card. Therefore, a SIMATIC memory card absolutely must be inserted in order to operate the CPU.

Uploading blocks from device

You can load the blocks of a device to your project. This is necessary, for example, if you want to edit blocks that only exist in this device. You have the option of loading either all available blocks (organization blocks, function blocks, functions and data blocks) and global PLC tags or individual blocks to the project.

Note

Loading CPU S7-300/400

When loading from a device, please note that alarm information may be lost and that this could lead to errors during compilation.

Uploading blocks from or downloading blocks to a memory card

Memory cards are plug-in cards used with an S7-1200 series CPU, for example, to replace the load memory of a device. In the case of S7-1500 series CPUs, they contain the load memory. Only the SIMATIC memory card from Siemens can be used for devices of the S7-1200 and S7-1500 series.

To use a memory card as load memory, you must download the user program or individual blocks to a memory card. You can just as well upload blocks from a memory card back into the project.

Note

S7-1200

Note the following when uploading to or downloading from a memory card:

- If the CPU contains no previous program and you insert an empty memory card in the CPU the program will be loaded from the PG/PC to the memory card and not to the CPU.
- If you insert an empty memory card prior to the startup of the CPU, the program that is on the CPU will be transferred automatically to the memory card. The program on the CPU will then be deleted.
- If you insert a memory card with a program in the CPU prior to the startup of the CPU and the CPU already contains a program, the program on the memory card will be executed and not the program on the CPU. The program on the CPU will be deleted.

Loading GRAPH function blocks

If you load a GRAPH function block together with its instance data block, the processing of the sequencer starts over at the initial step. As a result, problems may occur when synchronizing the sequencer with the process. You can avoid these problems by switching off the sequencer before loading.

Downloading changes without reinitialization

It often proves necessary to edit or expand a PLC program that was already commissioned and that is running on the plant without error. Such operations should be performed without causing any major interruptions of current operations.

S7-1500 therefore offers the option of extending the interfaces of function or data blocks during runtime and loading the modified blocks without setting the CPU to STOP or affecting the value of tags that are already loaded. This is a simple means of implementing program changes. This load process (download without reinitialization) will not have a negative impact on the controlled process.

Effects of a load operation on the tag values of a data blocks

When data blocks are downloaded to a device in STOP mode, the next transition of the device to RUN affects the current tag values as follows:

- Tags not marked as being retentive retain their defined start values.
- Retentive tags of the S7-1200 only retain their values if the following conditions are met:
  - You loaded the data block by means of "Download to device > Software (changes only)".
  - You made no changes to the DB structure.

  Otherwise the retentive tags will also retain their defined start values.
- Retentive tags of the S7-1500 only retain their values if the following conditions are met:
  - You loaded the data block by means of "Download to device > Software (changes only)".
  - You made no changes to the structure of the data block or modified it within the memory reserve.

  Otherwise the retentive tags will also retain their defined start values.

Loading blocks with synchronization

In team engineering, it is possible for several users to work on one project with several engineering systems at the same time and access one S7-1500 CPU. To ensure consistency within the shared project, it is necessary to synchronize the changed data prior to loading so that nothing gets overwritten unintentionally.

If differences are determined between the online and offline data management within the shared project during loading that were caused by a different engineering system, automatic synchronization of the data to be loaded is offered during loading.

In this case, the "Synchronization" dialog displays the data to be synchronized with the current status (online-offline comparison) and the possible actions.

The following options are available for synchronization:

Application case	Recommendation	Synchronization
One or more blocks on the CPU (online) are more recent than in the engineering system (offline).	These blocks should be downloaded from the CPU to the engineering system before loading.	Automatic synchronization is possible: The blocks in the engineering system are updated prior to loading.
One or more new blocks have been created and exist only in the CPU (online).	These blocks should be downloaded from the CPU to the engineering system before loading.	Automatic synchronization is possible: The new blocks are added prior to the download in the engineering system.
One or more blocks on the CPU have been deleted.	The blocks should also be deleted prior to the download in the engineering system.	Automatic synchronization is not possible. The blocks deleted on the CPU should be manually deleted in the offline project in the engineering system.

Application case	Recommendation	Synchronization
One or more blocks on the CPU and in the engineering system are different. This is the case when a different user has changed blocks to which you have also made corrections and has already downloaded them to the CPU.	These blocks with competing changes must be adapted manually. You decide in this case which changes you are going to accept. If the blocks on the CPU are to be retained, you should accept these blocks prior to download from the CPU to your engineering system. If the blocks that you have changed are to be applied, you can continue with the download without synchronization.	Automatic synchronization is not possible: The affected blocks on the CPU or in the engineering system must be adapted manually. One of the existing block versions (online or offline) will be overwritten in the process.
There are differences in the hardware configuration on the CPU (online) and in the engineering system (offline).	Differences in the hardware configuration must be adapted manually. You decide in this case which hardware configuration you are going to accept. If the existing hardware configuration on the CPU is to be retained, you should apply these in your engineering system prior to loading. If you want to apply the changed hardware configuration, you can continue with the download without synchronization.	Automatic synchronization is not possible: The hardware configuration must be adapted manually. One of the existing hardware configurations (online or offline) will be overwritten in the process.

You can use the "Force download to device" command to download blocks without synchronization, if desired.

Loading I&M data in PROFINET IO devices to load your modules

When loaded in a device, you can load I&M data in PROFINET IO devices and their modules in addition to the above-mentioned components.

11.15.2.2 Downloading blocks in the "RUN" operating mode to the device (S7-1200, S7-1500)

Basics on downloading blocks in the "RUN" operating mode

When you download modified blocks to the device, it is not always necessary to switch the device to the "STOP" operating mode. Prior to a download operation, the Engineering System checks whether the device must be stopped before downloading. The result of this check is displayed in the "Load preview" dialog.

If it is necessary to change to the "STOP" operating mode, you cannot continue the download process until you have stopped the CPU.

If the requirements are met, you can also download a modified program or program parts to a CPU in "RUN" operating mode.

The quantity structures may be exceeded for very complex programs which prevents a download in "RUN". In this case you must first create the prerequisites for a download in "RUN".

Tips:

- Use a memory card with sufficient capacity.
- Select a CPU with sufficient work memory.
- If necessary, reduce the number of downloaded objects (blocks, constants, PLC tags, data types).
  If you cannot download all objects at once, proceed in several steps and download a smaller number of objects in each step.

Note

Actual parameters are not overwritten by a download process in the "RUN" operating mode. Changes to the actual parameters will not become effective until the next time you change the operating mode from "STOP" to "RUN".

Downloading changes in "RUN"

The table below shows which program and configuration changes can be downloaded in "RUN" operating mode, sorted by CPU family and taking into consideration the firmware versions of the CPUs.

Explanations on the table:

- "RUN": Change can be downloaded in "STOP" operating mode as well as in "RUN" operating mode to the CPU.
- "RUN (< 57)": The CPU can integrate up to 56 new or modified objects/blocks in one program cycle. If you download more objects/blocks, they are integrated in several successive program cycles. If you want to download all objects/blocks consistently, you must set the CPU to "STOP" operating mode. This number depends on the setting for S7-300 CPUs with configuration option "Process mode/test mode".
- "RUN (Init)": Change can be downloaded in "RUN" operating mode; downloaded data blocks are re-initialized.
- "STOP": Change can only be downloaded in "STOP" operating mode.
- "STOP (Reset)": Change can only be downloaded in "STOP" operating mode; all data including retentive data is reset.

	S7-300	**S7-400**	**S7-1200 V4.0 and higher**	**S7-1500**
Action/type of change	**Download possible in mode ...**			
Modified properties of HW components. This includes changes to comments in the HW configuration.	STOP	STOP	STOP	STOP
Added HW components	STOP	STOP	STOP	STOP
New/revised text lists (messages)	**RUN**	**RUN**	STOP	**RUN (V1.1 and higher)**

	S7-300	S7-400	S7-1200 V4.0 and higher	S7-1500
Action/type of change	**Download possible in mode ...**			
Revised comments (new, revised, deleted) with the exception of comments in the HW configuration	-	-	**RUN**	**RUN**
Number of blocks downloaded at the same time	**RUN (<17)**	**RUN (<57)**	**RUN (<21)**	**RUN (all)**
Download PLC program to the device and reset	STOP (Reset)	STOP (Reset)	STOP (Reset)	STOP (Reset)
New OB	**RUN**	**RUN**	STOP	**RUN**
Modified OB: Code changes	**RUN**	**RUN**	**RUN**	**RUN**
OB with modified properties (e.g., cycle time change)	**RUN**	**RUN**	STOP	**RUN**
Deleted OB	**RUN**	**RUN**	STOP	**RUN**
New FB/FC/DB/User Data Type (UDT)	**RUN**	**RUN**	**RUN**	**RUN**
Deleted FB/FC/DB/User Data Type (UDT)	**RUN**	**RUN**	**RUN**	**RUN**
Revised FB/FC: Code change	**RUN**	**RUN**	**RUN**	**RUN**
Revised FB/FC: Interface change*	STOP	STOP	**RUN**	**RUN**
Modified DB: Modified property ("Only store in load memory" attribute changed)	STOP	STOP	**RUN (Init)**	**RUN (Init)**
Modified DB (memory reserve not enabled): Name/type of tags modified, tags added or deleted **	**RUN (Init)**	**RUN (Init)**	**RUN (Init)**	**RUN (Init)**
Modified DB (memory reserve enabled): New tag added**	-	-	**RUN**	**RUN**
Modified User Data Type (UDT)	STOP	STOP	**RUN (Init)**	**RUN (Init)**
Add new PLC tags (timer, counter, bit memory)	**RUN**	**RUN**	**RUN**	**RUN**
Modified retentivity settings (timer, counter, bit memory, DB area)	STOP	STOP	STOP	STOP
Motion Control technology objects: Changes to MC Servo cycle clock, change from free-running to cyclical (and vice versa). Changes to the HW interface of the TO	-	-	-	STOP

* If the interface change results in structural changes at the instance DB, see "Modified DB...".

** For the effect of download of data block changes to the data block content, see section "Downloading data blocks to the CPU"

Download of changes in "RUN" with older CPU firmware versions

The table below shows which changes you can download in "RUN" operating mode, in particular, for which older CPU firmware versions.

	S7-1200 V1.0 - 2.1	**S7-1200 V2.2 - V3.0**
Action/type of change	**Download possible in mode ...**	
Modified properties of HW components. This includes changes to comments in the HW configuration.	STOP	STOP
Added HW components	STOP	STOP
New/revised text lists (messages)	STOP	STOP
Revised comments (new, revised, deleted) with the exception of comments in the HW configuration	STOP	**RUN**
Number of blocks downloaded at the same time	STOP	**RUN (<11)**
Download PLC program to the device and reset	STOP (Reset)	STOP (Reset)
New OB	STOP	STOP
Modified OB: Code changes	STOP	**RUN**
OB with modified properties (e.g., cycle time change)	STOP	STOP
Deleted OB	STOP	STOP
New FB/FC/DB/User Data Type (UDT)	STOP	**RUN**
Deleted FB/FC/DB/User Data Type (UDT)	STOP	**RUN**
Revised FB/FC: Code change	STOP	**RUN**
Revised FB/FC: Interface change	STOP	STOP
Modified DB: Modified property ("Only store in load memory" attribute changed)	STOP	STOP
Modified DB (memory reserve not enabled): Name/type of tags modified, tags added or deleted	STOP	STOP
Modified User Data Type (UDT)	STOP	STOP
Add new PLC tags (timer, counter, bit memory)	STOP	STOP
Modified retentivity settings (timer, counter, bit memory, DB area)	STOP	STOP

Downloading data blocks to the CPU

Depending on the conditions, the download of new or modified data blocks has an effect on the actual values in the data block:

Download new data blocks	Actual values in the new data blocks are set to start values.
Download structurally modified data blocks (memory reserve not enabled)	Actual values of added tags in the structurally modified data blocks are set to start values.

Download structurally modified data blocks (memory reserve enabled)	The following actual values are retained: • Actual values of tags outside the memory reserve • Actual values of tags that were not modified within the memory reserve Actual values of added tags within the memory reserve are set to start values.
Download of simply modified data blocks (no structural modification)	Actual values are retained.

Additional information

Additional information on download of block extensions without re-initialization and download of modified values for data blocks is available at "See also".

See also

Downloading blocks from program editor to device (Page 5074)

Downloading blocks from the project tree to the device (Page 5076)

Downloading project data to a device (Page 350)

11.15.2.3 Downloading blocks from program editor to device (S7-1200, S7-1500)

Requirement

The block to be downloaded is open.

Procedure

To download a block from the program editor to the device, follow these steps:

1. Right-click in the instruction window of the programming editor.
2. Select the "Download to device" command in the shortcut menu.
   - If you have not already established an online connection, the "Extended download to device" dialog opens. In this case, set all parameters required for the connection and click "Load".
   - You can save your preferred connection parameters as default under "Options > Settings > Online & Diagnostics". When you first load with new connection parameters, a query is automatically displayed whether you want to store it as a default setting. Click "Yes", if you want to save the current connection parameters as default.
   - You have the option of showing all compatible devices by selecting the corresponding option and clicking the "Start search" command. You can also open the "Extended download to device" dialog explicitly via the "Online" menu.
     See also: Establishing and terminating an online connection
   - If you have already specified an online connection, then the project data is compiled if necessary and the "Load preview" dialog opens. This dialog displays alarms and proposes actions necessary for loading.
3. Check the alarms and, where necessary, enable the actions in the "Action" column.
   As soon as downloading becomes possible, the "Load" button is enabled.

Note

Actions

Performing the proposed actions during ongoing plant operation can cause serious damage to property or injury to persons if there are functional faults or program errors.

Make sure that no dangerous situations can arise before you start the actions.

Note

To avoid inconsistencies between calling and called blocks, download all affected blocks each time you make global changes, such as changes in the block interface. Select the "Consistent download" action.

4. Click "Load".
   If there is a need for synchronization, the system automatically displays the "Synchronization" dialog. This dialog displays messages and suggests actions that are needed for the synchronization. You have the option of performing these actions or forcing the download without synchronization by clicking "Force download to device". If you have performed the suggested actions, you will be asked whether you want to continue with the download. Click "Continue download" in order to download the block. The "Load results" dialog then opens and shows you the status and the actions after the download operation.
5. If you want to start the modules again directly after downloading, select the "Start all" check box.
6. To close the "Load results" dialog box, click "Finish".

Result

The code for the block will be downloaded to the device. If the changes affect additional blocks, these will be compiled and also downloaded to the device. Blocks that only exist online in the device are deleted. Existing CPU data blocks are retained, however. Inconsistencies between the blocks in the user program are avoided by loading all blocks affected and deleting the unneeded blocks in the device.

The messages under "Info > General" in the Inspector window show whether the downloading process was successful.

See also

Downloading blocks from the project tree to the device (Page 5076)

Downloading project data to a device (Page 350)

Downloading blocks in the "RUN" operating mode to the device (Page 5070)

11.15.2.4 Downloading blocks from the project tree to the device (S7-1200, S7-1500)

In the project tree you can download one block, multiple blocks or all blocks to a device.

Downloading one or more blocks from the project tree to the device

To download one block or multiple blocks to the device from the project tree, follow these steps:

1. Open the "Program blocks" folder in project tree.
2. Select the blocks you want to download.
3. Select the "Download to device > Software (only changes)" command from the shortcut menu.
   - If you have not already established an online connection, the "Extended download to device" dialog opens. In this case, set all parameters required for the connection and click "Load".
   - You can save your preferred connection parameters as default under "Options > Settings > Online & Diagnostics". You will be prompted as such when you download new connection parameters for the first time. Select "Yes", if you want to save the current connection parameters as default.
   - You have the option of showing all compatible devices by selecting the corresponding option and clicking the "Start search" command. You can also open the "Extended download to device" dialog explicitly via the "Online" menu.
     See also: Go online and Go offline
   - If you have already specified an online connection, then the project data is compiled if necessary and the "Load preview" dialog opens. This dialog displays alarms and proposes actions necessary for loading.

4. Check the alarms and, where necessary, enable the actions in the "Action" column.

   Note

   Performing the proposed actions during ongoing plant operation can cause serious damage to property or injury to persons if there are functional faults or program errors.

   Make sure that no dangerous situations can arise before you start the actions.

   As soon as downloading becomes possible, the "Load" button is enabled.

5. Click "Load".
   If there is a need for synchronization, the system automatically displays the "Synchronization" dialog. This dialog displays messages and suggests actions that are needed for the synchronization. You have the option of performing these actions or forcing the download without synchronization by clicking "Force download to device". If you have performed the suggested actions, you will be asked whether you want to continue with the download. Click "Continue download" in order to download the block. The "Load results" dialog then opens and shows you the status and the actions after the download operation.
6. If you want to start the modules again directly after downloading, select the "Start all" check box.
7. To close the "Load results" dialog box, click "Finish".

Downloading blocks from the project tree to the device

To download all blocks in the "Program blocks" folder to the device from the project tree, follow these steps:

1. Select the "Program blocks" folder in the project tree.
2. Select the "Download to device" submenu in the shortcut menu.
3. If you only want to download the changes since the last download, select the "Software (only changes)" option. If all blocks are to be fully loaded and all values are to be reset to their start values, select "Download PLC program to the device and reset".
   - If you have not already established an online connection, the "Extended download to device" dialog opens. In this case, set all parameters required for the connection and click "Load". You have the option of showing all compatible devices by selecting the corresponding option and clicking the "Start search" command. You can also open the "Extended download to device" dialog with the "Online" menu.
     See also: Go online and Go offline
   - If you have already specified an online connection, then the project data is compiled if necessary and the "Load preview" dialog opens. This dialog displays alarms and proposes actions necessary for loading.
4. Check the alarms and, where necessary, enable the actions in the "Action" column.

   Note

   Performing the proposed actions during ongoing plant operation can cause serious damage to property or injury to persons if there are functional faults or program errors.

   Make sure that no dangerous situations can arise before you start the actions.

   As soon as downloading becomes possible, the "Load" button is enabled.

5. Click "Load".
   If there is a need for synchronization, the system automatically displays the "Synchronization" dialog. This dialog displays messages and suggests actions that are needed for the synchronization. You have the option of performing these actions or forcing the download without synchronization by clicking "Force download to device". If you have performed the suggested actions, you will be asked whether you want to continue with the download. Click "Continue download" in order to download the block. The "Load results" dialog then opens and shows you the status and the actions after the download operation.
6. If you want to start the modules again directly after downloading, select the "Start all" check box.
7. To close the "Load results" dialog box, click "Finish".

Result

The code for the blocks is downloaded to the device. If the changes affect additional blocks, these will be compiled and also downloaded to the device. Blocks that only exist online in the device are deleted. Inconsistencies between the blocks in the user program are avoided by loading all blocks affected and deleting the unneeded blocks in the device.

The messages under "Info > General" in the Inspector window show whether the downloading process was successful.

See also

Downloading blocks from program editor to device (Page 5074)

Downloading project data to a device (Page 350)

Downloading blocks in the "RUN" operating mode to the device (Page 5070)

11.15.2.5 Uploading blocks from device (S7-1200, S7-1500)

You can load either all the blocks or only individual blocks from a device to your project.

Note

Please note the following:

- Please note that when you load individual blocks, no tags or other required blocks to which you may refer are loaded together with the individual blocks. During the loading operation, references to tags and blocks are reassigned where possible based on the names. After the loading operation, check whether these assignments are correct.
- When loading from a device to an empty project, any existing folder structures for blocks and objects from libraries might not be loaded.
- S7-1500: When GRAPH function blocks are loaded from a device to your project, the step-specific alarm texts for the interlock and supervision alarms are not loaded.
- S7-1500: The start values that you have changed with the instruction "WRIT_DBL: Write to data block in the load memory" will be lost during execution of the action "Upload from device".

Requirement

The online and offline versions of the blocks to be loaded differ or the blocks only exist online.

Loading all blocks from a device

To load all blocks from a device, follow these steps:

1. Establish an online connection with the device from which you want to load the blocks.
   See also: Establishing and terminating an online connection
2. In the project tree, select the device folder from which you want to load blocks.
3. In the "Online" menu, select the "Upload from device" command.
   The "Upload preview" dialog box opens. This dialog displays alarms and proposes actions necessary for loading.
4. Check the alarms and, where necessary, enable the actions in the "Action" column.
   As soon as loading becomes possible, the "Upload from device" button is enabled.
5. Click the "Upload from device" button.
   The loading operation is performed.

Loading individual blocks from a device

To load individual blocks from a device, follow these steps:

1. Establish an online connection with the device from which you want to load the blocks.
   See also: Establishing and terminating an online connection
2. In the project tree, select the blocks that you want to load from the device.
3. In the "Online" menu, select the "Upload from device" command.
   The "Upload preview" dialog box opens. This dialog displays alarms and proposes actions necessary for loading.
4. Check the alarms and, where necessary, enable the actions in the "Action" column.
   As soon as loading becomes possible, the "Upload from device" button is enabled.
5. Click the "Upload from device" button.
   The loading operation is performed.

Result

The blocks will be loaded from the device to the project. You can edit them as normal, recompile them and load them to the device again.

11.15.2.6 Downloading blocks to a memory card (S7-1200, S7-1500)

Requirement

- The memory card is marked as a program card.
- The "Program blocks" folder of the memory card is open.

Procedure

To download blocks to a memory card, follow these steps:

1. Open the "Program blocks" folder of the device in the project tree.
2. Select the blocks you want to download to the memory card.
3. Drag the blocks in project tree to the "Program blocks" folder of the memory card. You can also copy the blocks and add them to the memory card.
   If necessary, the blocks are compiled. The "Load preview" dialog then opens. This dialog displays alarms and proposes actions necessary for loading.
4. Check the alarms and, where necessary, enable the actions in the "Action" column.
5. As soon as downloading becomes possible, the "Load" button is enabled.
6. Click the "Load" button.
   If there is a need for synchronization, the system automatically displays the "Synchronization" dialog. This dialog displays messages and suggests actions that are needed for the synchronization. You have the option of performing these actions or forcing the download without synchronization by clicking "Force download to device". If you have performed the suggested actions, you will be asked whether you want to continue with the download. Click "Continue download" to download the block. The "Load results" dialog then opens and shows you the status and the actions after the download operation.
7. Click "Finish".

Or:

1. In the project tree, select the blocks that you want to upload.
2. Select the "Card reader/USB memory > Write to memory card" command in the "Project" menu.
   The "Select memory card" dialog opens.
3. Select a memory card which is compatible with the CPU.
   A button with a green check mark is activated at the bottom of the dialog.
4. Select the button with the green check mark.
   If necessary, the project data are compiled. The "Load preview" dialog then opens. This dialog displays alarms and proposes actions necessary for loading.
5. Check the alarms and, where necessary, enable the actions in the "Action" column.
   As soon as downloading becomes possible, the "Load" button is enabled.
6. Click the "Load" button.
   If there is a need for synchronization, the system automatically displays the "Synchronization" dialog. This dialog displays messages and suggests actions that are needed for the synchronization. You have the option of performing these actions or forcing the download without synchronization by clicking "Force download to device". If you have performed the suggested actions, you will be asked whether you want to continue with the download. Click "Continue download" to download the block. The "Load results" dialog then opens and shows you the status and the actions after the download operation.
7. Click "Finish".

Result

The blocks are downloaded to the memory card. If the changes affect additional blocks, these will also be downloaded to the memory card. Blocks that exist only on the memory card are deleted. Inconsistencies between the blocks in the user program are avoided by downloading all affected blocks and the deleting of the non-required blocks on the memory card.

The messages under "Info > General" in the Inspector window show whether the downloading process was successful.

See also

Uploading blocks from a memory card (Page 5081)

Accessing memory cards (Page 422)

11.15.2.7 Uploading blocks from a memory card (S7-1200, S7-1500)

You can only upload all blocks from one memory card back into your project.

Requirement

The memory card is displayed.

See also: Accessing memory cards (Page 422)

Procedure

To upload blocks from a memory card to your project, follow these steps:

1. In the project tree, drag the folder of the memory card to the folder of the device in the project. You can also copy the memory card and insert it in the device.
   The "Upload preview" dialog box opens. This dialog displays alarms and proposes actions necessary for loading.
2. Check the alarms and, where necessary, enable the actions in the "Action" column.
   The "Upload from device" button will be enabled as soon as uploading becomes possible.
3. Click on the "Upload from device" button.

See also

Downloading blocks to a memory card (Page 5079)

11.15.2.8 Switching off the sequencer prior to loading a GRAPH DB (S7-1500)

You can specify the switching off of the sequencer prior to loading an instance data block either globally or during the load operation.

Globally switching off the sequencer

To switch off the sequencer globally for each loading operation of an instance data block, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The "Settings" window is displayed in the work area.
2. Select the "PLC programming > GRAPH" group in the area navigation.
3. Select the "Turn off sequence before downloading DB" check box.
   For future loading operations, the sequencer is switched off prior to loading of the instance data block.

Switching off the sequencer during the loading operation

To switch off the sequencer during the loading operation, follow these steps:

1. Load the GRAPH function block to the device.
   During the loading operation, the "Load preview" dialog opens. This dialog displays alarms and suggests the required actions for loading. If the instance data block must be loaded together with the GRAPH function block, the "Load preview" dialog suggests the action "Turn off sequence before downloading the DB".
2. Select the "Turn off sequence before downloading DB" check box.

11.16 Comparing PLC programs

11.16.1 Basic information on comparing PLC programs

11.16.1.1 Introduction to comparing PLC programs

Function

You can compare the following objects of a PLC program in order to detect any differences:

- Code blocks with other code blocks
- Data blocks with other data blocks
- PLC tags of a PLC tag table with the PLC tags of another PLC tag table
- PLC data types with other PLC data types

Types and levels of comparison

Two different basic types of comparison can be used:

- Online/offline comparison
  The objects in the project are compared with the objects of the corresponding online device. An online connection to the device is necessary for this comparison.
- Offline/offline comparison
  With offline/offline comparison you can compare the objects of two devices in the project that is currently open. A device from a reference project or from a library can also be dragged onto the right drop area. You can only compare devices within a TIA Portal instance.

Please note that you cannot carry out an unlimited number of comparisons at the same time, but only one comparison per comparison type (online/offline or offline/offline).

You can choose between the following levels of comparison depending on how in-depth an object comparison you require:

- Compare editor
  If you start a comparison on a device, the compare editor opens in which the result of the comparison is displayed.
- Detailed comparison
  For some comparison objects, you can then start a detailed comparison in the compare editor in which the objects compared will be opened side-by-side, each in its own program editor instance. Any differences will be highlighted. For blocks, PLC tags and PLC data types, you can also start the detailed comparison directly from the project tree.

The table below gives an overview of the types and levels of comparison you can apply for each object:

Object	Online/offline		Offline/offline	
Compare editor	Detailed comparison	Compare editor	Detailed comparison	
LAD block	X	X	X	X
FBD block	X	X	X	X
STL block[1]	X	X	X	X
SCL block	X	X[3 4]	X	X
GRAPH block[2]	X	X[4]	X	X[5]
Global data block	X	X	X	X
Instance data block	X	X	X	X
PLC tags	-	-	X	X
PLC data type	X[4]	X[4]	X	X

Legend:

X: available
-: not available

[1]: STL is not available for S7-1200

[2]: GRAPH is not available for S7-1200

[3]: not for S7-1200 older than version 2.0

[4]: not for S7-300/400

[5]: only comparable with a GRAPH block from the same CPU family

Note

Please note the following:

- You cannot perform a detailed comparison for know-how protected blocks.
- If the detail comparison detects differences only with respect to the data types of local tags, with offline being an interrupt data type (C_ALARM C_ALARM_S C_ALARM_8 C_ALARM_8P C_ALARM_T C_AR_SEND C_NOTIFY C_NOTIFY_8P) and online a DWORD, this difference is not marked as such.
- You cannot run a detailed comparison for types and master copies from libraries.

Comparing PLC programs based on checksums

In addition to the comparison options listed above, TIA Portal gives provides you the option of a simple checksum comparison: The system automatically determines a checksum for the entire PLC program. Based on this checksum, you can easily determine whether two PLC programs are identical.

See also: Comparing PLC programs based on checksums (Page 5086)

See also

Basics of project data comparison (Page 358)

Comparing blocks and PLC data types (Page 5085)

Comparing PLC tags (Page 5086)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Performing detailed block comparisons (Page 5089)

11.16.1.2 Comparing blocks and PLC data types

Assignment criteria

The blocks and PLC data blocks to be compared are assigned to each other on the basis of the following criteria:

- Blocks:
  - Online/offline comparison: Addresses, e.g. FB100 or DB100
  - Offline/offline comparison: Symbolic names of the blocks
- PLC data types: PLC data types are assigned to each other by name

Comparison

When comparing blocks and PLC data types, checksums are formed for specific properties that serve as the basis for comparison. Some of these properties are grouped into categories in the Comparison Editor. No checksums are available with an online/offline comparison for CPUs of the S7-300/400 series; the time stamp is evaluated in this case. Basically, the time stamp is issued as information for an online/offline comparison for all CPUs as a separate category.

After performing the comparison, you can use actions to define what is to be done about the differences. You can also start a detailed comparison for blocks. The versions of a block compared are opened beside each other and the differences are highlighted. For blocks, PLC tags and PLC data types, you can also start the detailed comparison directly from the project tree.

See also

Introduction to comparing PLC programs (Page 5083)

Comparing PLC tags (Page 5086)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Performing detailed block comparisons (Page 5089)

11.16.1.3 Comparing PLC tags

Introduction

The device PLC tag tables will also be shown in the comparison editor if you carry out an offline/offline comparison. The PLC tag tables are assigned to each by name and you can see at a glance whether the PLC tag tables are available in both devices. If necessary, you can also perform a detailed comparison for a PLC tag table. For blocks, PLC tags and PLC data types, you can also start the detailed comparison directly from the project tree.

You obtain the overall status of the all PLC tags with an online/offline comparison of CPUs of the S7-1200/1500 series:

See also

Introduction to comparing PLC programs (Page 5083)

Comparing blocks and PLC data types (Page 5085)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Performing detailed block comparisons (Page 5089)

11.16.1.4 Comparison of PLC programs based on checksums

Introduction

PLC programs are automatically identified with unique checksums during compilation. Based on this checksum, you can identify your program and determine whether two PLC programs are identical.

Because the checksum is loaded to the CPU together with the PLC program, it can also serve as important information in case of a service call. You can, for example, easily determine whether the program that is currently running on the CPU is the same program you loaded some time ago or if it has been changed in the meantime. The original program does not need to be installed on your programming device (PG) in this case.

Creation of the system diagnostics blocks

PLC programs are automatically given a unique checksum during compilation. If the next compilation run detects that the PLC program has been changed, the program is given a new checksum. If the PLC program has not changed but is still compiled again, the checksum remains the same.

The checksum also remains unchanged when you make changes and undo them.

The check takes place every time the PLC program is compiled, for example, when you select a CPU or a block folder and use the menu command "Compile > Software".

Two checksums are generated:

- Checksum for software: A checksum is generated across all blocks in the block folder.
  - Blocks that are not called in the program but exist in the block folder are also included in the checksum.
  - Safety blocks and safety PLC data types (UDT) are not taken into consideration.
  - Changes to watch tables or force tables do not change the checksum.
  - Data blocks that are created or modified during runtime, for example, by the instructions "WRIT_DBL", "CREAT_DB" and "DELETE_DB", do not influence the formation of the checksum.
- Checksum for text lists: A checksum is generated for all text lists in the project.

Loading the checksum

The checksum is loaded to the CPU together with the PLC program and is available in the online program. During download from the CPU, the checksum is not applied to the offline project because it is generated again automatically during the next compilation.

Evaluating the checksum

The checksum is displayed in the CPU properties (Properties > General > Checksums). From there, you can apply it manually to your documents. The "GetChecksum" extended instruction is available to read out the checksum while the program is running.

See also

GetChecksum: Read out checksum (Page 3963)

11.16.2 Comparing blocks

11.16.2.1 Comparing blocks in the compare editor

You have the following options for comparing blocks in the compare editor:

- Online/offline comparison
  The blocks in the project are compared with the blocks of the selected device.
- Automatic offline/offline comparison
  All blocks of the selected devices are compared offline.
- Manual offline/offline comparison
  The selected blocks of the devices are compared offline.

Carrying out an online/offline comparison of blocks

To perform an online/offline comparison, follow these steps:

1. Select a device in the project tree that allows online/offline comparison.
2. Select the "Compare > Offline/online" command in the shortcut menu.
3. If you have not already established an online connection to this device, the "Go online" dialog opens. In this case, set all the necessary parameters for the connection and click "Connect".
   The online connection is established and the compare editor opens.
4. Open the "Program blocks" folder.
   You can identify the status based on the symbols in the status and action area. You can define certain actions depending on the status of the objects. Note, however, that you can only perform actions in one direction in a synchronization action.

Performing an automatic offline/offline comparison of blocks

To perform an automatic offline/offline comparison of blocks, follow these steps:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Drag-and-drop an additional device to the drop area of the right pane. The device to be compared can originate from the same project, a reference project or the library.
4. Open the "Program blocks" folder.
   You can identify the status of the objects based on the symbols in the status and action area. You can define certain actions depending on the status of the objects. When you select an object, the object's properties and the corresponding object of the assigned device are clearly shown in the properties comparison.

You can drag any other device to the drop area at any time to perform further comparisons.

Carrying out a manual offline/offline comparison of blocks

To perform a manual offline/offline comparison of blocks, follow these steps:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Drag-and-drop an additional device to the drop area of the right pane. The device to be compared can originate from the same project, a reference project or the library.
4. In the status and action area, click on the button for switching between automatic and manual comparison.
5. Select the objects that you want to compare.
   The properties comparison is displayed. You can identify the status of the objects based on the symbols. You can define certain actions depending on the status of the objects.

You can drag any other device to the drop area at any time to perform further comparisons.

See also

Basics of project data comparison (Page 358)

Introduction to comparing PLC programs (Page 5083)

Using the compare editor (Page 362)

Performing detailed block comparisons (Page 5089)

Comparing PLC tags (Page 5115)

Comparing PLC data types (Page 5117)

11.16.2.2 Performing detailed block comparisons

Start a detailed comparison for LAD/FBD/STL/SCL blocks

You can start a detailed comparison for blocks. The versions of a block compared are opened beside each other and the differences highlighted. You have the following options for starting a detailed comparison for blocks:

- Starting detailed comparisons using the compare editor
  The detailed comparison for blocks is possible with the compare editor, both for online/offline and for offline/offline comparisons.
- Starting detailed comparison in the project tree
  The detailed comparison for blocks is possible with the project tree, both for online/offline and for offline/offline comparisons. An existing online connection is required for the online/offline comparison.
- Starting detailed comparisons in the program editor
  In this case the opened block is compared with its online version.

Note

Please note the following:

- For blocks that are created in the programming language SCL, the detail comparison is not available for S7-1200 series CPUs with a version older than 2.0.
- S7-1500: It is possible that another user can carry out a loading process on the CPU through joint parallel working on a CPU. If the currently compared block is changed or deleted by this loading process, the detailed comparison is terminated and a message is displayed. In this case, re-start the detailed comparison, if required.

Starting detailed comparisons using the compare editor

To start a detailed comparison for a block using the compare editor, follow these steps:

1. First, perform an online/offline or an offline/offline comparison.
   The compare editor opens.
2. In the compare editor, select the block for which you want to perform a detailed comparison.
3. Click "Start detailed comparison" in the toolbar or select the "Start detailed comparison" from the shortcut menu.

Starting detailed comparison in the project tree

To start an offline/offline detailed comparison for a block directly in the project tree, follow these steps:

1. Right-click the block that you want to compare. This can also be a block from a reference project.
2. Select the command "Fast comparison > Select as left object" in the shortcut menu.
3. Right-click the block that you want to compare with the block that you previously selected as left object.
4. Select the command "Fast comparison > Compare with <selected object>" in the shortcut menu. "<selected object> stands for the left comparison object.

To start an online/offline comparison for a block directly in the project tree, follow these steps:

1. Establish an online connection to the device where the block is located.
2. Right-click the block that you want to compare with its online object.
3. Select the command "Fast comparison > Compare with the online object" in the shortcut menu.

Starting detailed comparisons in the program editor

For the comparison type online/offline, you can start the detailed comparison directly in the program editor. To do this, follow these steps:

1. Open the block for which you wish to carry out a detailed comparison.
2. Establish an online connection.
   See also: Go online and Go offline

   Note

   Please note that the block must be available online in order for you to be able to start the detailed comparison for the block within the programming editor.

3. Click "Detailed comparison" in the toolbar.
4. Confirm the dialog for closing the block with "Yes".

Result

One instance of the program editor will be opened for each version of the block compared and the two instances are displayed side by side. Any differences will be highlighted.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out offline/offline comparisons (Page 361)

Carrying out an online/offline comparison (Page 360)

Using the compare editor (Page 362)

Visualization of the comparison result for LAD/FBD (Page 5095)

Visualization of the comparison result for STL (Page 5099)

Visualization of the comparison result for SCL (Page 5102)

Navigating in the detailed comparison (Page 5111)

Changing blocks during the detailed comparison (Page 5113)

Updating comparison results (Page 5114)

Start detailed comparison for GRAPH blocks (S7-300, S7-400, S7-1500)

You can start a detailed comparison for GRAPH blocks. The versions of a block compared are opened beside each other and the differences highlighted. You have the following options for starting a detailed comparison for GRAPH blocks:

- Starting detailed comparisons using the compare editor
  The detailed comparison for blocks is possible with the compare editor, both for online/offline and for offline/offline comparisons.
- Starting detailed comparison in the project tree
  The detailed comparison for blocks is possible with the project tree, both for online/offline and for offline/offline comparisons. An existing online connection is required for the online/offline comparison.
- Starting detailed comparisons in the program editor
  In this case the opened block is compared with its online version.

The following comparison methods are available for GRAPH blocks:

- Compare sequence
  Complete sequencers are compared with each other in this mode. A detailed comparison for a GRAPH block always starts in the "Compare sequence" comparison mode. This means the comparison starts at the beginning of the sequencer and the differences between the sequencers are displayed. If there are structural differences between the sequencers, the comparison results are only displayed up to the first structural difference.
- Compare selection
  You can use the "Compare selection" comparison mode to compare any areas with each other. This means you can compare sections of the sequencer which are downstream from a structural difference in the sequence.

You can toggle at random between both comparison modes.

Note

S7-1500: It is possible that another user can carry out a loading process on the CPU through joint parallel working on a CPU. If the currently compared block is changed or deleted by this loading process, the detailed comparison is terminated and a message is displayed. In this case, re-start the detailed comparison, if required.

Starting detailed comparisons using the compare editor

To start a detailed comparison, follow these steps:

1. First, perform an online/offline or an offline/offline comparison between the two devices. The compare editor opens.
2. In the compare editor, select the GRAPH block for which you want to perform a detailed comparison.

   Note

   The manual comparison is also available for an offline/offline comparison. It lets you select any GRAPH block in the comparison editor for the comparison.

3. Click the "Start detailed comparison" button in the toolbar.
   One instance of the program editor is opened for each version of the GRAPH blocks compared and the two instances are displayed side by side. Any differences will be highlighted. The comparison takes place in the "Compare sequence" mode.

Starting detailed comparison in the project tree

To start an offline/offline detailed comparison for a GRAPH block directly in the project tree, follow these steps:

1. Right-click the block that you want to compare. This can also be a block from a reference project.
2. Select the command "Fast comparison > Select as left object" in the shortcut menu.
3. Right-click the block that you want to compare with the block that you previously selected as left object.
4. Select the command "Fast comparison > Compare with <selected object>" in the shortcut menu. "<selected object> stands for the left comparison object.
   One instance of the program editor is opened for each version of the GRAPH blocks compared and the two instances are displayed side by side. Any differences will be highlighted. The comparison takes place in the "Compare sequence" mode.

To start an online/offline detailed comparison for a GRAPH block directly in the project tree, follow these steps:

1. Establish an online connection to the device where the block is located.
2. Right-click the block that you want to compare with its online object.
3. Select the command "Fast comparison > Compare with the online object" in the shortcut menu.
   One instance of the program editor is opened for each version of the GRAPH blocks compared and the two instances are displayed side by side. Any differences will be highlighted. The comparison takes place in the "Compare sequence" mode.

Starting detailed comparisons in the program editor

For the comparison type online/offline, you can start the detailed comparison directly in the program editor. To do this, follow these steps:

1. Open the GRAPH block for which you wish to carry out a detailed comparison.
2. Establish an online connection.
   See also: Go online and Go offline

   Note

   Please note that the block must be available online in order for you to be able to start the detailed comparison for the block within the programming editor.

3. Click "Detailed comparison" in the toolbar.
4. Confirm the dialog for closing the block with "Yes".
   One instance of the program editor is opened for each version of the GRAPH blocks compared and the two instances are displayed side by side. Any differences will be highlighted. The comparison takes place in the "Compare sequence" mode.

Comparing regions

To compare specific areas within the displayed sequencers, follow these steps:

1. In each sequencer, select the step from which you want to start the comparison.
2. Click "Comparison mode" in the toolbar.
   The comparison mode changes to "Compare selection" and the comparison result is updated. The sequencers are compared as of the selected steps. The compared substrings are highlighted.
3. To compare other areas, select the steps which are to serve as starting point for the comparison.
4. Click "Update the comparison result" in the toolbar.

   Note

   When you change the comparison mode, you are also updating the comparison results at the same time. If you stay in "Compare selection" mode, you must manually update the comparison results every time you change the starting point.

5. To compare the complete sequencers once again, click "Comparison mode" in the toolbar.
   Each time you click "Comparison mode" you are changing the comparison mode. You can also open the drop-down list with the arrow and select the required mode from the list.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out offline/offline comparisons (Page 361)

Carrying out an online/offline comparison (Page 360)

Using the compare editor (Page 362)

Visualization of the comparison result for GRAPH (Page 5105)
Navigating in the detailed comparison (Page 5111)
Changing blocks during the detailed comparison (Page 5113)
Updating comparison results (Page 5114)

Visualization of the comparison result

Visualization of the comparison result for LAD/FBD

Introduction

The detailed comparison allows you to identify the exact places where versions of a block differ. The following color coding allows you to find these places as quickly as possible:

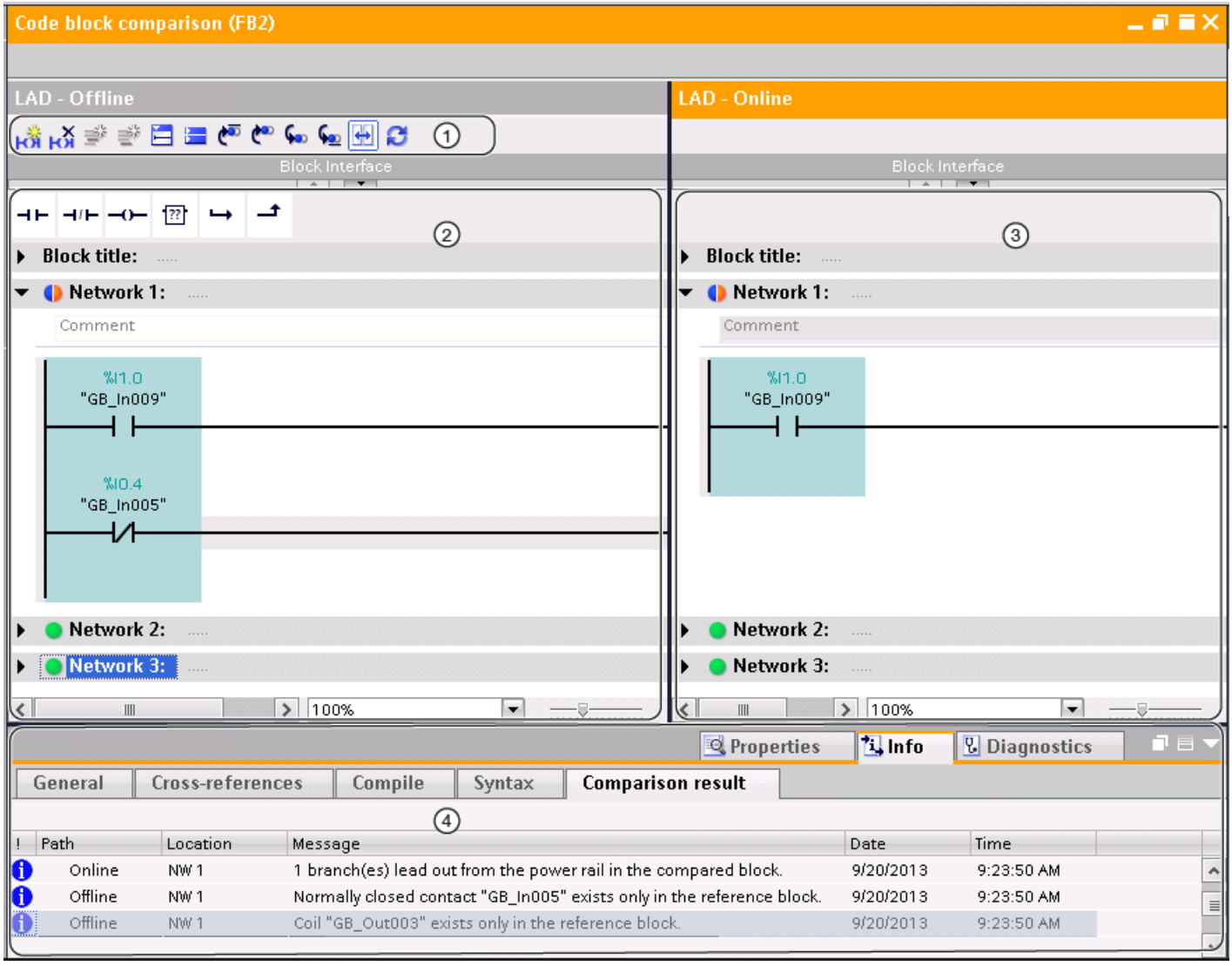
- The lines where there are differences are highlighted in gray.
- Differing operands and instructions are highlighted in green.
- If the number of networks differs, pseudo-networks are added to allow the display of identical networks to be synchronized. These pseudo-networks are highlighted in gray and contain the text "No corresponding network found" in the title bar of the network. Pseudo-networks cannot be edited.

For the sake of clarity, not all the differences are highlighted but rather the first difference of an operation in each case. For example, if all the inputs in an instruction with multiple inputs are different in the offline and online versions of the block, only the first input is highlighted as a difference. You can resolve this difference and update the comparison list. The next input will then be highlighted as a difference.

The number of differences highlighted within a network therefore depends on the number of instructions.

Structure of the detailed comparison

The following figure shows an example of the online/offline detailed comparison for the LAD programming language:



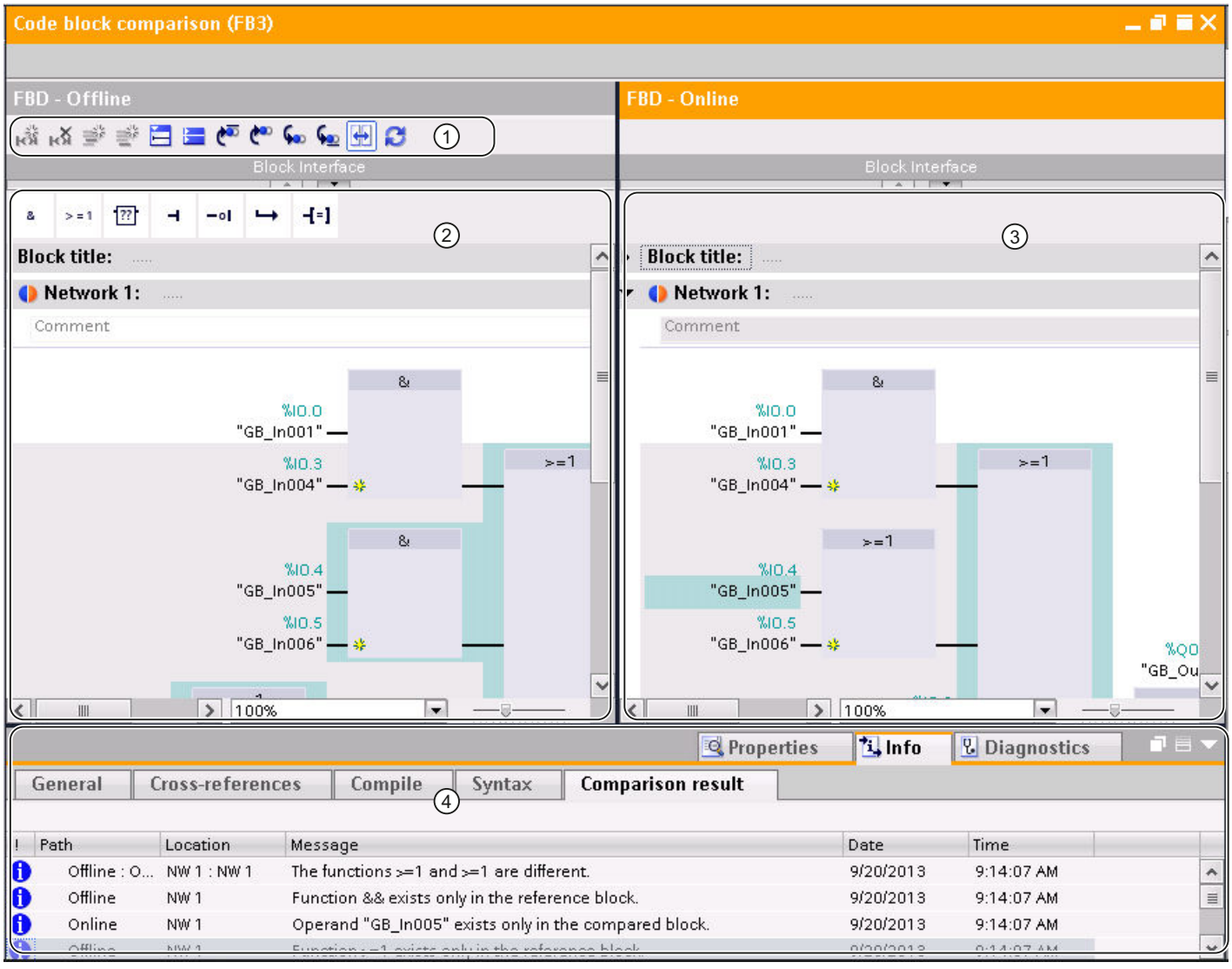
① Toolbar of the detailed comparison for LAD

② Reference block

③ Compared block

④ Comparison result in the Inspector window

The following figure shows an example of the online/offline detailed comparison for the FBD programming language:



① Toolbar of the detailed comparison for FBD

② Reference block

③ Compared block

④ Comparison result in the Inspector window

Note

Display of the symbolic names of the online version of the block is only possible for S7-1200 and S7-1500.

Toolbar of the detailed comparison

With the toolbar, you can access the following functions:

- General functions
  - Insert network
  - Delete network
  - Insert row
  - Add row
  - Open all networks
  - Close all networks
- Comparison-specific functions
  - Position on first difference
  - Position on previous difference
  - Position on next difference
  - Position on last difference
  - Synchronize scrolling between editors
  - Update comparison results

Reference block

The reference block is displayed in the left window. In an online/offline comparison, the reference block is the offline version of the block.

Compared block

The compared block is displayed in the right window. In an online/offline comparison, the compared block is the online version of the block.

Comparison result in the Inspector window

The differences are displayed in the form of a table in the "Info > Comparison result" tab of the Inspector window. Double-click on a row to navigate to the corresponding difference in the block.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Start a detailed comparison for LAD/FBD/STL/SCL blocks (Page 5089)

Navigating in the detailed comparison (Page 5111)

Changing blocks during the detailed comparison (Page 5113)

Updating comparison results (Page 5114)

Visualization of the comparison result for STL (S7-300, S7-400, S7-1500)

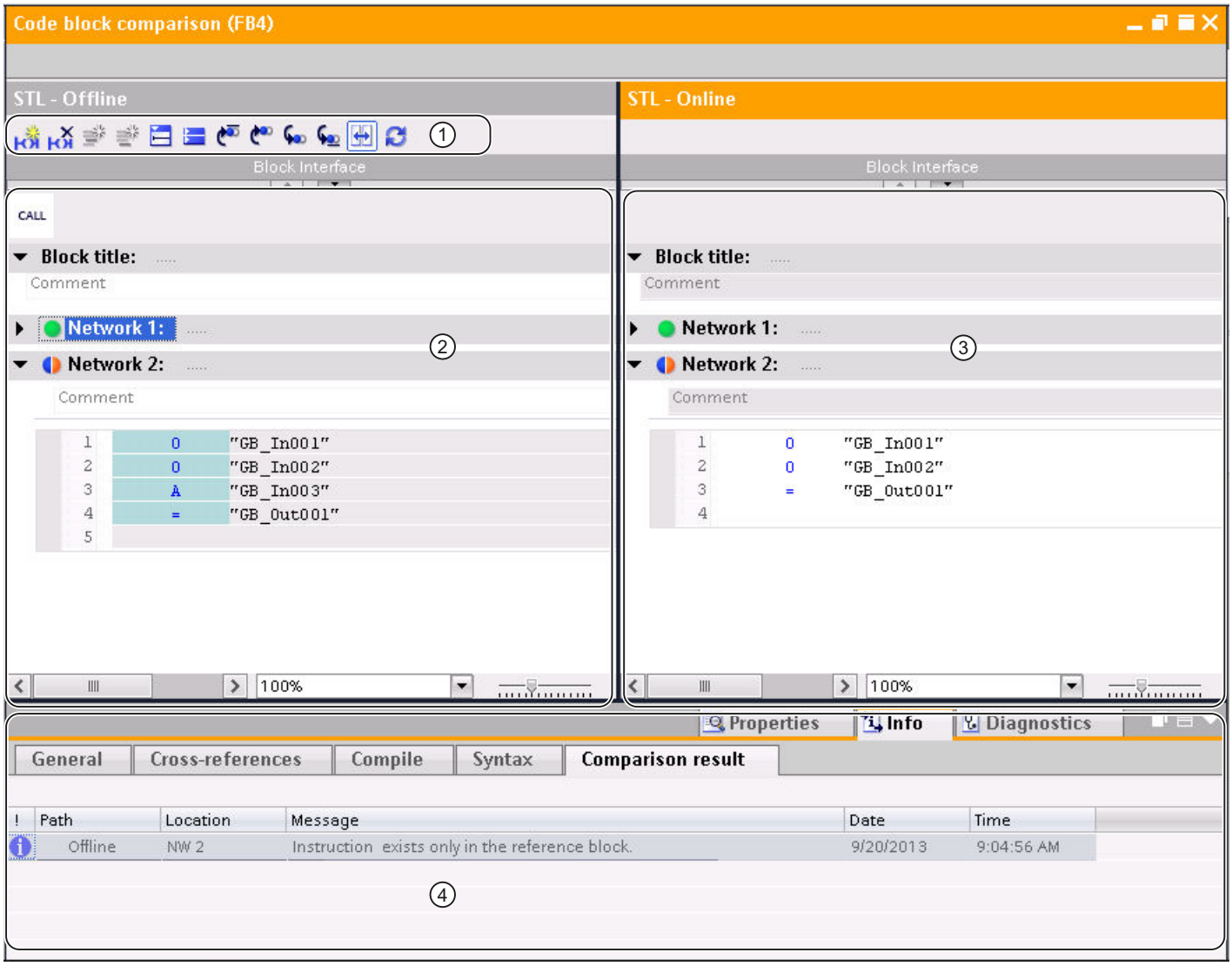
Introduction

The detailed comparison allows you to identify the exact places where versions of a block differ. The following color coding allows you to find these places as quickly as possible:

- The lines where there are differences are highlighted in gray.
- Differing operands and instructions are highlighted in green.
- If the number of networks differs, pseudo-networks are added to allow the display of identical networks to be synchronized. These pseudo-networks are highlighted in gray and contain the text "No corresponding network found" in the title bar of the network. Pseudo-networks cannot be edited.

Structure of the detailed comparison

The following figure shows an example of the online/offline detailed comparison for the STL programming language:



① Toolbar of the detailed comparison for STL

② Reference block

③ Compared block

④ Comparison result in the Inspector window

Note

The display of the symbolic names of the online version of the block is only possible for S7-1500.

Toolbar of the detailed comparison

With the toolbar, you can access the following functions:

- General functions
  - Insert network
  - Delete network
  - Insert row
  - Add row
  - Open all networks
  - Close all networks
- Comparison-specific functions
  - Position on first difference
  - Position on previous difference
  - Position on next difference
  - Position on last difference
  - Synchronize scrolling between editors
  - Update comparison results

Reference block

The reference block is displayed in the left window. In an online/offline comparison, the reference block is the offline version of the block.

Compared block

The compared block is displayed in the right window. In an online/offline comparison, the compared block is the online version of the block.

Comparison result in the Inspector window

The differences are displayed in the form of a table in the "Info > Comparison result" tab of the Inspector window. Double-click on a row to navigate to the corresponding difference in the block.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Start a detailed comparison for LAD/FBD/STL/SCL blocks (Page 5089)

Navigating in the detailed comparison (Page 5111)

Changing blocks during the detailed comparison (Page 5113)

Updating comparison results (Page 5114)

Visualization of the comparison result for SCL

Introduction

The detailed comparison allows you to identify the exact places where versions of a block differ. The following color coding allows you to find these places as quickly as possible:

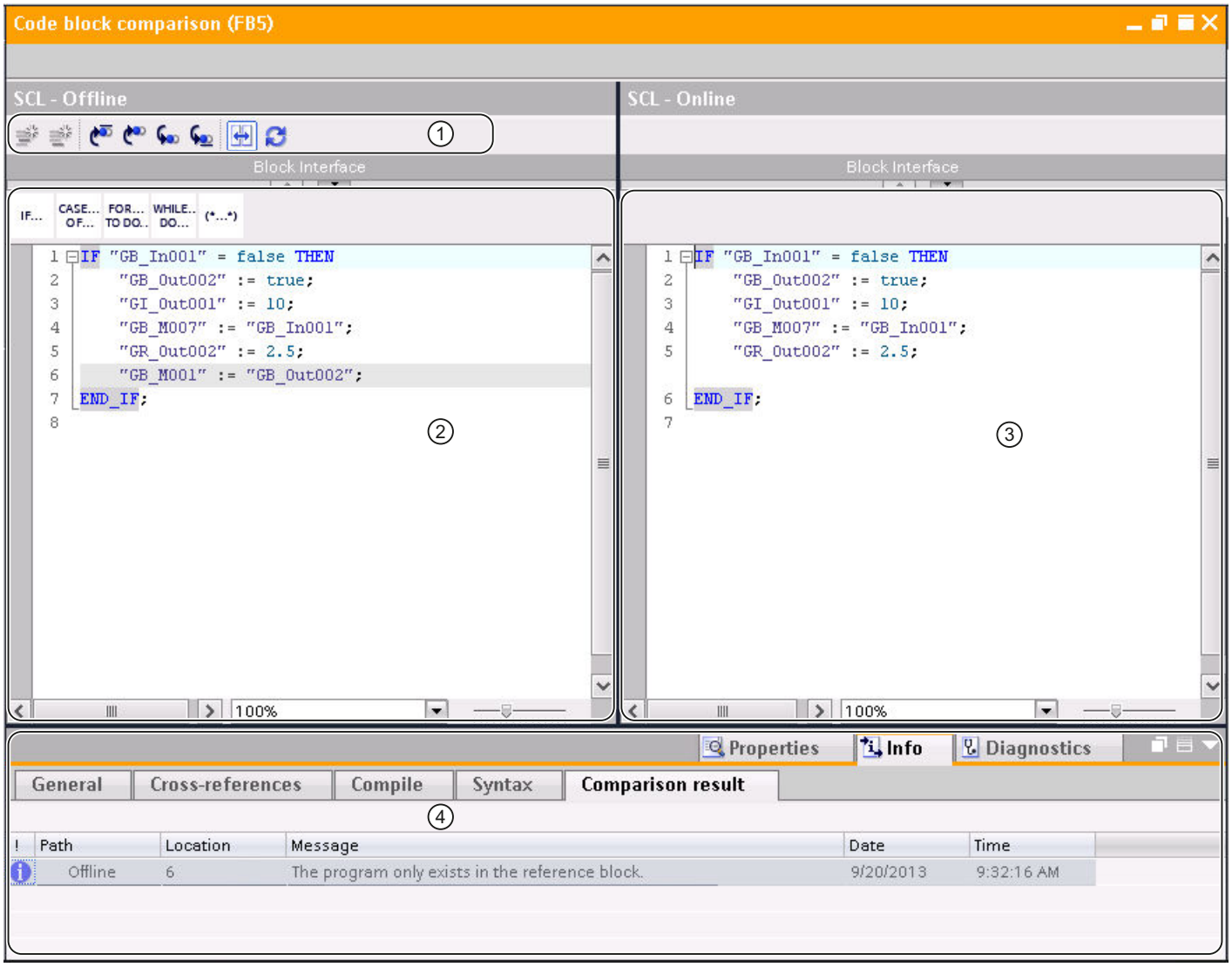
- The lines where there are differences are highlighted in gray.
- Differing operands and instructions are highlighted in green.

Note

The online/offline detailed comparison is not available for the CPU families S7-300/400 and for S7-1200 with a version less than 2.0.

Structure of the detailed comparison

The following figure shows an example of the online/offline detailed comparison for the SCL programming language:



① Toolbar of the detailed comparison for SCL

② Reference block

③ Compared block

④ Comparison result in the Inspector window

Note

The display of the symbolic name of the online version of the block is only possible for S7-1200 and S7-1500.

Toolbar of the detailed comparison

With the toolbar, you can access the following functions:

- General functions
  - Insert row
  - Add row
- Comparison-specific functions
  - Position on first difference
  - Position on previous difference
  - Position on next difference
  - Position on last difference
  - Synchronize scrolling between editors
  - Update comparison results

Reference block

The reference block is displayed in the left window. In an online/offline comparison, the reference block is the offline version of the block.

Compared block

The compared block is displayed in the right window. In an online/offline comparison, the compared block is the online version of the block.

Comparison result in the Inspector window

The differences are displayed in the form of a table in the "Info > Comparison result" tab of the Inspector window. Double-click on a row to navigate to the corresponding difference in the block.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Start a detailed comparison for LAD/FBD/STL/SCL blocks (Page 5089)

Navigating in the detailed comparison (Page 5111)

Changing blocks during the detailed comparison (Page 5113)

Updating comparison results (Page 5114)

Visualization of the comparison result for GRAPH (S7-300, S7-400, S7-1500)

Introduction

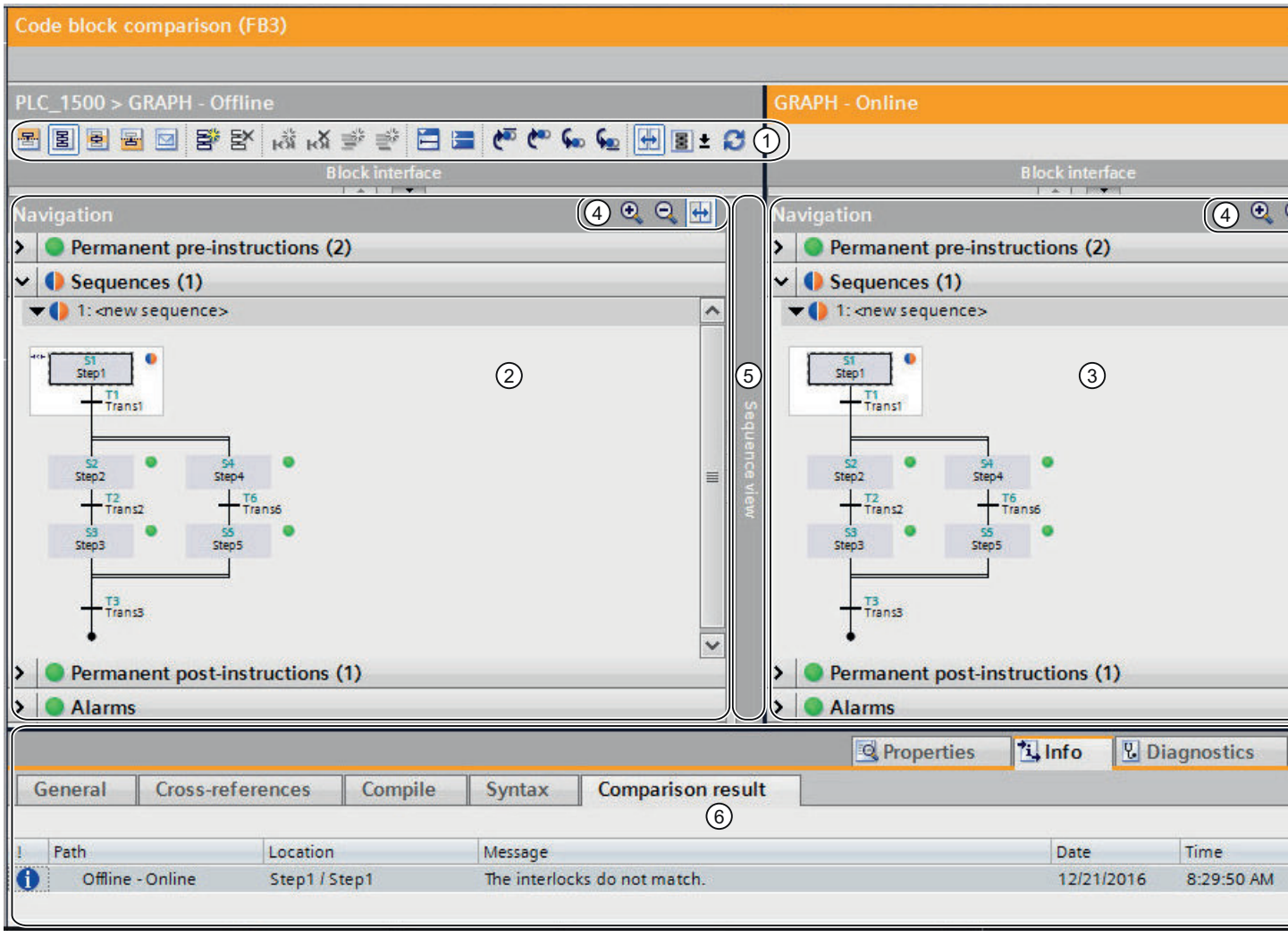
The detailed comparison allows you to identify the exact places where versions of a block differ. When you start a detailed comparison for a GRAPH block, navigation is opened first. Use the dividers to toggle between the navigation and the currently set view. You can select other views with the toolbar of the detailed comparison.

The result of the comparison is indicated by the comparison symbols.

See also: Overview of the comparison editor (Page 362)

Structure of the detailed comparison

The following figure shows an example for the navigation view with an online/offline detailed comparison for the GRAPH programming language:



① Toolbar of the detailed comparison for GRAPH

② Reference block

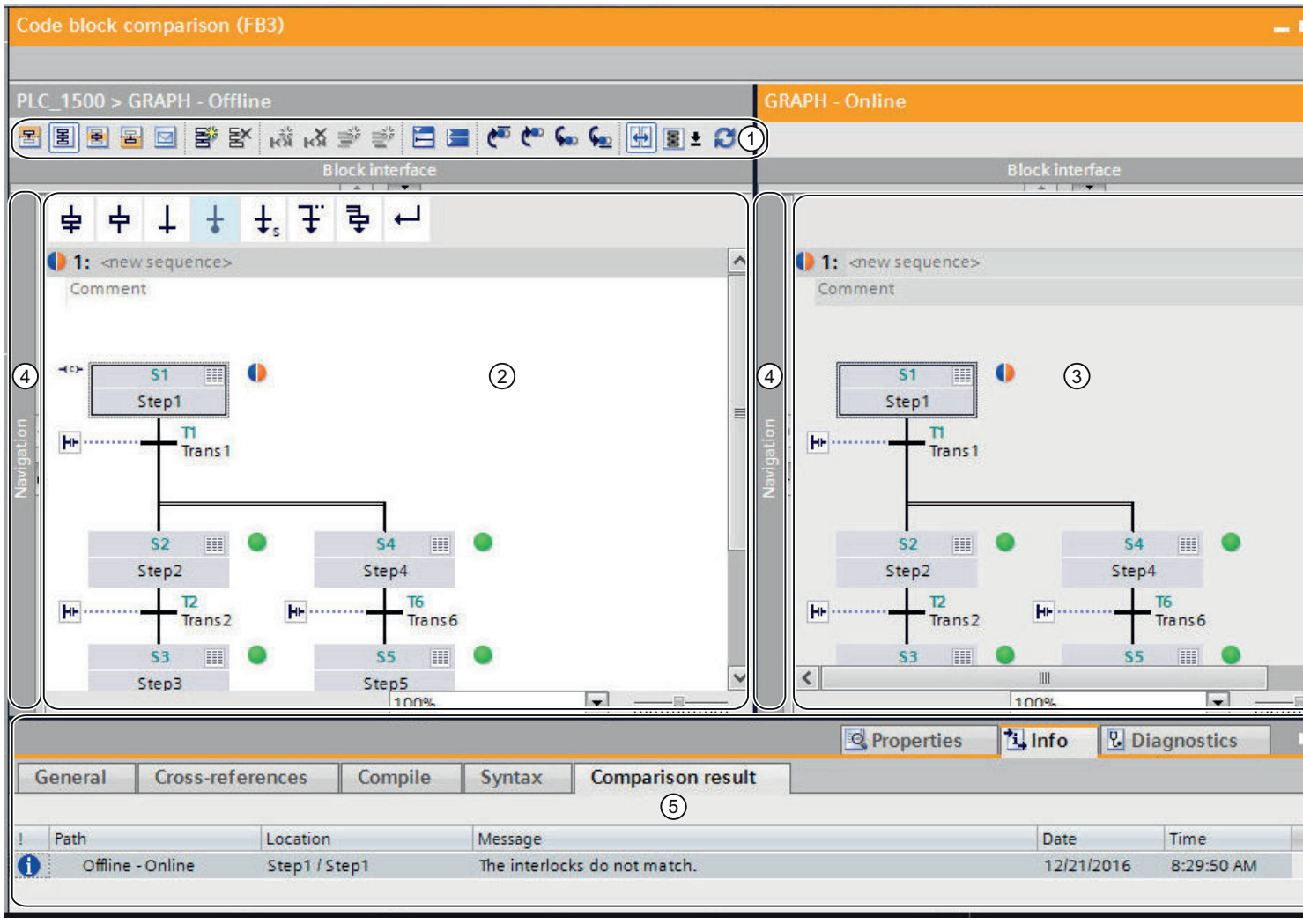
③ Compared block

④ Navigation toolbar

⑤ Dividers

⑥ Comparison result in the Inspector window

The following figure shows an example for the sequence view with an online/offline detailed comparison for the GRAPH programming language:



① Toolbar of the detailed comparison for GRAPH

② Reference block

③ Compared block

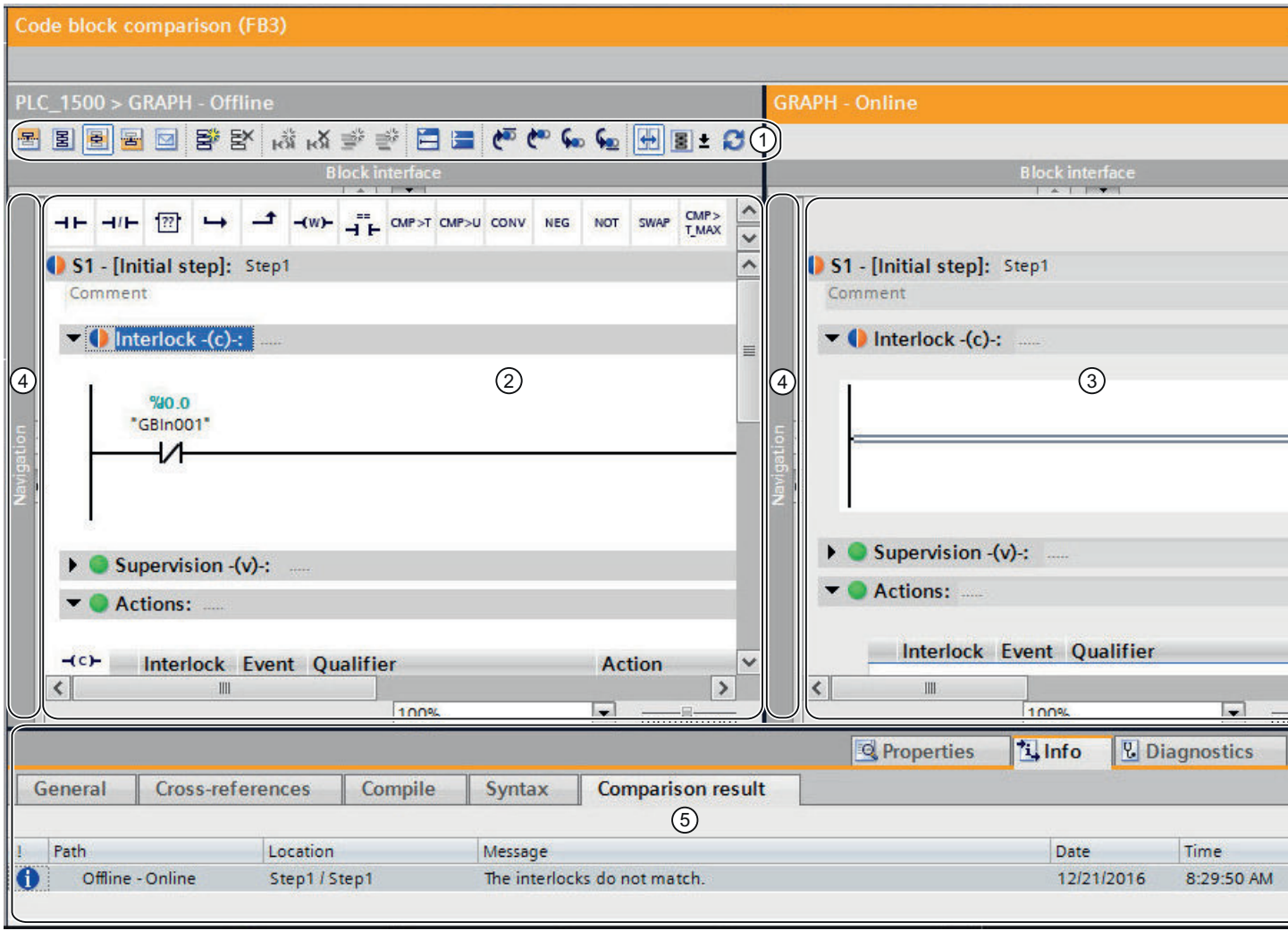
④ Dividers

⑤ Comparison result in the Inspector window

Note

If there are structural differences between the blocks, the comparison results are only displayed up to the first structural difference in the sequence view.

The following figure shows an example for the single step view with an online/offline detailed comparison for the GRAPH programming language:



① Toolbar of the detailed comparison for GRAPH

② Reference block

③ Compared block

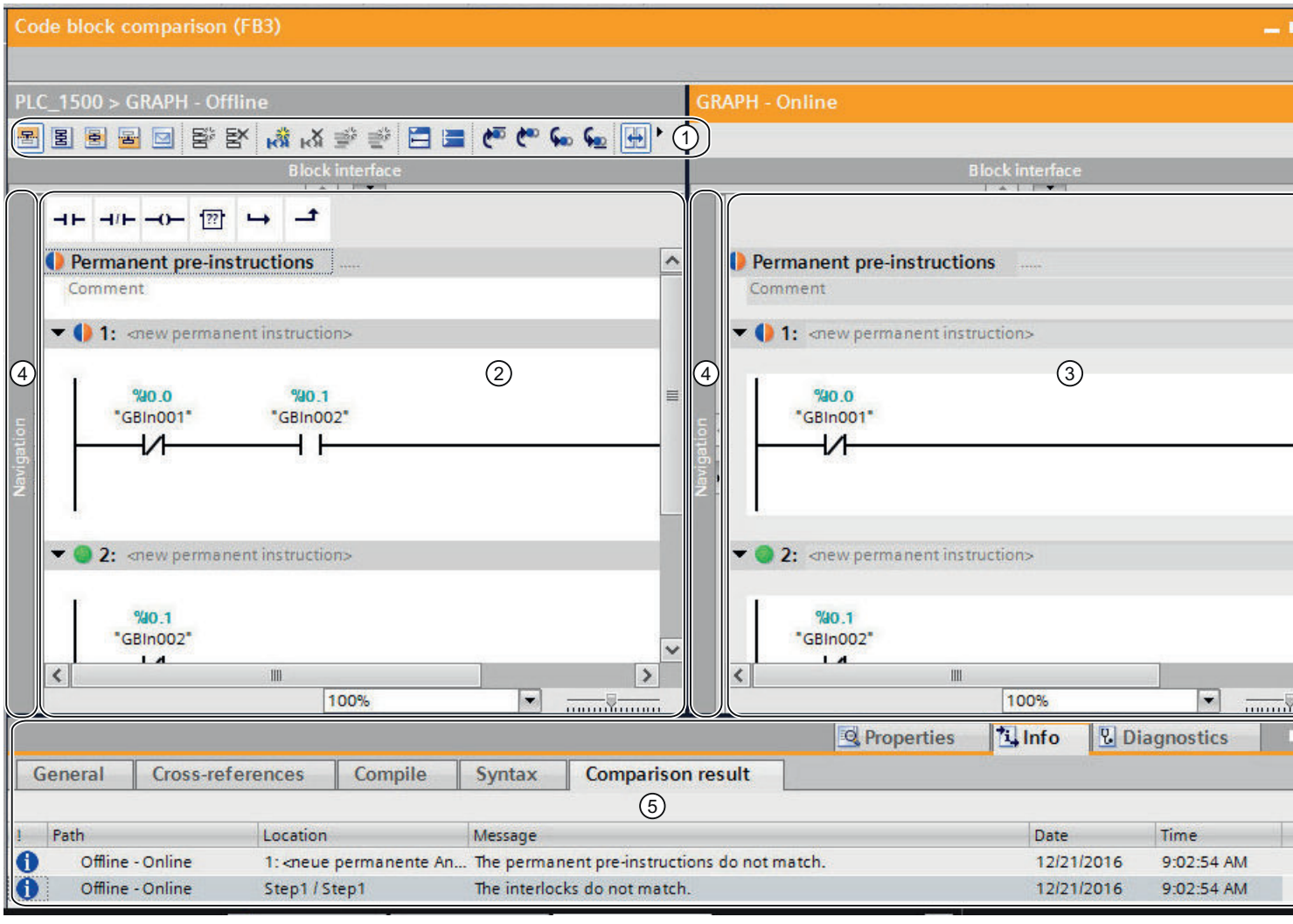
④ Dividers

⑤ Comparison result in the Inspector window

Note

The result of the comparison refers to the complete network Differences within the networks are not identified.

The following figure shows an example for the view of permanent instructions with an online/ offline detailed comparison for the GRAPH programming language:



① Toolbar of the detailed comparison for GRAPH

② Reference block

③ Compared block

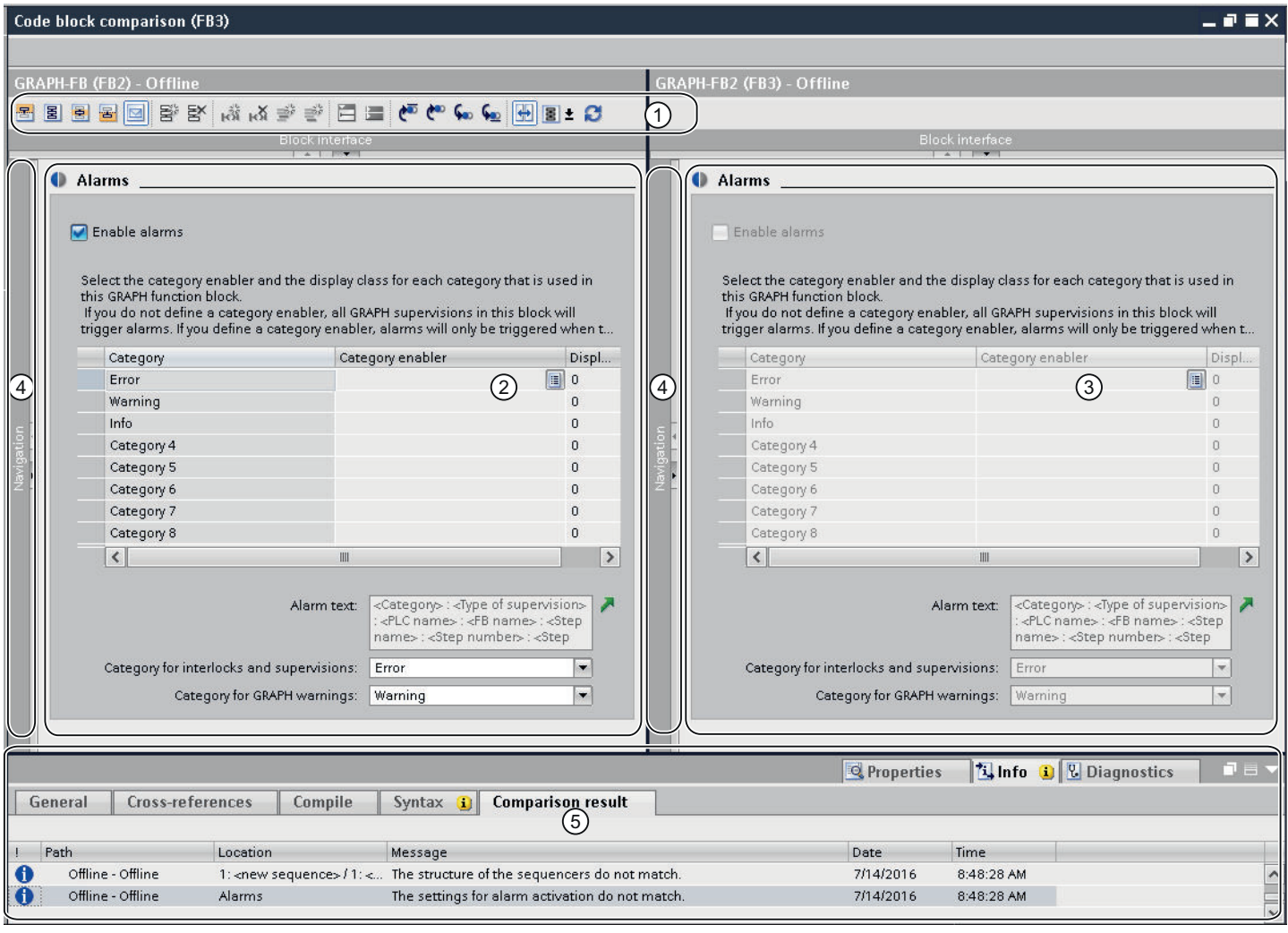
④ Dividers

⑤ Comparison result in the Inspector window

Note

The result of the comparison refers to the complete network Differences within the networks are not identified.

The following figure shows an example for the alarm view with an online/offline detailed comparison for the GRAPH programming language:



① Toolbar of the detailed comparison for GRAPH

② Reference block

③ Compared block

④ Dividers

⑤ Comparison result in the Inspector window

Toolbars

With the toolbar of the detailed comparison, you can access the following functions:

- General functions
  - Change to permanent pre-instructions
  - Change to sequence view
  - Change to single step view
  - Change to permanent post-instructions
  - Change to alarm view
  - Add sequence
  - Delete sequence
  - Insert network
  - Delete network
  - Insert row
  - Add row
  - Open all networks
  - Close all networks
- Comparison-specific functions
  - Position on first difference
  - Position on previous difference
  - Position on next difference
  - Position on last difference
  - Synchronize scrolling between editors
  - Update comparison results
  - Select the comparison mode

The navigation has its own toolbar with the following functions:

- Zoom in or zoom out of elements within the navigation
- Synchronize navigation

Reference block

The reference block is displayed in the left window. In an online/offline comparison, the reference block is the offline version of the block.

Compared block

The compared block is displayed in the right window. In an online/offline comparison, the compared block is the online version of the block.

Dividers

You can click the dividers to toggle quickly between the navigation and the current view.

Comparison result in the Inspector window

The differences are displayed in the form of a table in the "Info > Comparison result" tab of the Inspector window. Double-click on a row to navigate to the corresponding difference in the block.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Start a detailed comparison for LAD/FBD/STL/SCL blocks (Page 5089)

Start detailed comparison for GRAPH blocks (Page 5092)

Navigating in the detailed comparison (Page 5111)

Changing blocks during the detailed comparison (Page 5113)

Updating comparison results (Page 5114)

Navigating in the detailed comparison

Requirement

You have run a detailed comparison.

Navigate to the differences

To navigate to a difference between the two blocks, follow these steps:

1. Open the list of results for the detailed comparison under "Info > Comparison result" in the Inspector window.
2. Double-click a difference.
   The difference is selected in both editors.

Or:

1. Click one of the following navigation buttons on the toolbar:
   - Position on first difference
     Navigates to the first difference in the block, and displays the difference in both editors.
   - Position on previous difference
     Navigates to the previous difference starting from the current position, and displays the difference in both editors.
   - Position on next difference
     Navigates to the next difference starting from the current position, and displays the difference in both editors.
   - Position on last difference
     Navigates to the last difference in the block, and displays the difference in both editors.

Switching off/on the synchronization of the vertical scrolling between the editors

The scrolling for both editors is synchronized to ensure that the corresponding networks are visible parallel to each other during vertical scrolling. You can switch this mode off and on. To do this, follow these steps:

1. To switch off synchronized scrolling, click the "Synchronize scrolling between editors" button in the toolbar.
2. To switch on synchronized scrolling again, click the "Synchronize scrolling between editors" button one more time in the toolbar.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Start a detailed comparison for LAD/FBD/STL/SCL blocks (Page 5089)

Start detailed comparison for GRAPH blocks (Page 5092)

Visualization of the comparison result for LAD/FBD (Page 5095)

Visualization of the comparison result for STL (Page 5099)

Visualization of the comparison result for SCL (Page 5102)

Visualization of the comparison result for GRAPH (Page 5105)

Changing blocks during the detailed comparison (Page 5113)

Updating comparison results (Page 5114)

Changing blocks during the detailed comparison

While you are carrying out the detailed comparison, you can make changes to the blocks that are being compared. Remember the following:

- Online/offline comparison: You can change only the offline block.
- Offline/offline comparison: You can change only the offline block in the left-hand area.

After a block change, it may be necessary to manually update the comparison result in the comparison editor to ensure that the comparison status is displayed correctly. You can then specify actions to synchronize the objects.

Note

You cannot change SCL blocks manually. However, you can apply changes from one block to the other block. In this regard, note the following points:

- You cannot apply any changes in an online block.
- You can only apply changes in an offline block if this is not write-protected. This is the case, for example, if the blocks of the detailed comparison come from different CPUs. It is then also possible to apply the changes to the block in the right-hand area.

Changing LAD, FBD or STL blocks

To change LAD, FBD or STL blocks, follow these steps:

1. Change the block in the left-hand area according to your requirements.
2. If necessary, click on "Update comparison results" in the toolbar.

Changing GRAPH blocks

To change a GRAPH block, follow these steps:

1. Switch to sequence view by clicking "Sequence view" between the two blocks.
2. Change the block in the left-hand area according to your requirements.
3. If necessary, click on "Update comparison results" in the toolbar.

Changing SCL blocks

To apply a change from one block to another one, follow these steps:

1. In the sidebar of the block from which you want to apply a change to another block, click on the arrow in the corresponding line.
   The line is inserted in the other block and the arrow buttons are removed.

   Note

   The colors of the arrows mean the following:
   - Gray: The changes cannot be applied to the other block, as the other block is either an online block or write-protected.
   - Blue: The changes are applied from an offline block to the other block.
   - Orange: The changes are applied from an online block to the other block.

2. If necessary, click on "Update comparison results" in the toolbar.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Start a detailed comparison for LAD/FBD/STL/SCL blocks (Page 5089)

Start detailed comparison for GRAPH blocks (Page 5092)

Visualization of the comparison result for LAD/FBD (Page 5095)

Visualization of the comparison result for STL (Page 5099)

Visualization of the comparison result for SCL (Page 5102)

Visualization of the comparison result for GRAPH (Page 5105)

Navigating in the detailed comparison (Page 5111)

Updating comparison results (Page 5114)

Updating comparison results

As soon as you change an object, the comparison results are no longer valid and must be updated.

Requirement

You have run a detailed comparison.

Procedure

To update the comparison results, follow these steps:

1. Click "Update the comparison result" in the toolbar.

See also

Basic information on comparing PLC programs (Page 5083)

Carrying out an online/offline comparison (Page 360)

Carrying out offline/offline comparisons (Page 361)

Using the compare editor (Page 362)

Start a detailed comparison for LAD/FBD/STL/SCL blocks (Page 5089)

Start detailed comparison for GRAPH blocks (Page 5092)

Visualization of the comparison result for LAD/FBD (Page 5095)

Visualization of the comparison result for STL (Page 5099)

Visualization of the comparison result for SCL (Page 5102)

Visualization of the comparison result for GRAPH (Page 5105)

Navigating in the detailed comparison (Page 5111)

Changing blocks during the detailed comparison (Page 5113)

11.16.3 Comparing PLC tags

You have the following options for comparing PLC tags:

- Automatic offline/offline comparison in the compare editor
  The PLC tag tables of the selected devices are compared offline.
- Manual offline/offline comparison in the compare editor
  The selected PLC tag tables of the devices are compared offline.
- Detailed comparison
  Use the detailed comparison to determine differences within the PLC tag tables. You can start the detailed comparison either via the compare editor or in the project tree.

Performing automatic offline/offline comparison in the compare editor

To perform an automatic offline/offline comparison of PLC tag tables, follow these steps:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.

3. Drag-and-drop an additional device to the drop area of the right pane. The device to be compared can originate from the same project, a reference project or the library.
4. Open the "PLC tags" folder.
   You can identify the status of the PLC tag tables based on the symbols in the status and action area. You can define certain actions depending on the status.

You can drag any other device to the drop area at any time to perform further comparisons.

Performing manual offline/offline comparison in the compare editor

To perform a manual offline/offline comparison of PLC tag tables, follow these steps:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Drag-and-drop an additional device to the drop area of the right pane. The device to be compared can originate from the same project, a reference project or the library.
4. In the status and action area, click on the button for switching between automatic and manual comparison.
5. Select the PLC tag tables that you want to compare.
   The properties comparison is displayed. You can identify the status based on the symbols.

You can drag any other device to the drop area at any time to perform further comparisons.

Starting detailed comparisons using the compare editor

To start a detailed comparison for a PLC tag table using the compare editor, follow these steps:

1. Perform an automatic or manual offline/offline comparison.
2. For an automatic offline/offline comparison in the compare editor, select the PLC tag table for which you want to run a detailed comparison. Note that two PLC tag tables must be selected for comparison for a manual offline/offline comparison.
3. Click the "Start detailed comparison" button in the toolbar.
   A separate compare editor opens. All existing PLC tags of the selected PLC tag tables are displayed depending on the settings of the compare editor. User and system constants are not shown, however. You can identify the status of the PLC tags based on the symbols. You can define certain actions depending on the status of the PLC tags.

Starting detailed comparison in the project tree

To start a detailed comparison for a PLC tag table in the project tree, follow these steps:

1. Right-click the PLC tag table that you want to compare. This can also be a PLC tag table from a reference project.
2. Select the command "Fast comparison > Select as left object" in the shortcut menu.

3. Right-click the PLC tag table that you want to compare with the PLC tag table that you previously selected as left object.
4. Select the command "Fast comparison > Compare with <selected object>" in the shortcut menu. "<selected object> stands for the left comparison object.
   A separate compare editor opens. All existing PLC tags of the selected PLC tag tables are displayed depending on the settings of the compare editor. User and system constants are not shown, however. You can identify the status of the PLC tags based on the symbols. You can define certain actions depending on the status of the PLC tags.

See also

Introduction to comparing PLC programs (Page 5083)

Comparing PLC data types (Page 5117)

Using the compare editor (Page 362)

Comparing blocks (Page 5088)

11.16.4 Comparing PLC data types

You have the following options for comparing PLC data types:

- Online/offline comparison (S7-1200/1500 only)
  The PLC data types in the project are compared with the PLC data types of the selected device.
- Automatic offline/offline comparison in the compare editor
  The PLC data types of the selected devices are compared offline.
- Manual offline/offline comparison in the compare editor
  The selected PLC data types of the devices are compared offline.
- Detailed comparison
  Use the detailed comparison to determine differences between PLC data types. You can start the detailed comparison either via the compare editor or in the project tree.

Performing online/offline comparison of PLC data types in the compare editor

To perform an online/offline comparison, follow these steps:

1. Select a device in the project tree that allows online/offline comparison.
2. Select the "Compare > Offline/online" command in the shortcut menu.
   If you have not already established an online connection to this device, the "Go online" dialog opens. In this case, set all the necessary parameters for the connection and click "Connect".
   The online connection is established and the compare editor opens.
3. Open the "PLC data types" folder.
   You can identify the status of the PLC data types based on the symbols in the status and action area. When you select an object, the properties of the PLC data type and the corresponding PLC data type of the assigned device are displayed in the properties comparison.

Performing automatic offline/offline comparison in the compare editor

To perform an automatic offline/offline comparison of PLC data types, follow these steps:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Drag-and-drop an additional device to the drop area of the right pane. The device to be compared can originate from the same project, a reference project or the library.
4. Open the "PLC data types" folder.
   You can identify the status of the PLC data types based on the symbols in the status and action area. You can define certain actions depending on the status. You can drag any other device to the drop area at any time to perform further comparisons.

Performing manual offline/offline comparison in the compare editor

To perform a manual offline/offline comparison of PLC data types, follow these steps:

1. Select a device in the project tree that allows offline/offline comparison.
2. Select the "Compare > Offline/offline" command in the shortcut menu.
   The compare editor opens and the selected device is displayed in the left area.
3. Drag-and-drop an additional device to the drop area of the right pane. The device to be compared can originate from the same project, a reference project or the library.
4. In the status and action area, click on the button for switching between automatic and manual comparison.
5. Select the PLC data types that you want to compare.
   The properties comparison is displayed. You can identify the status based on the symbols. You can drag any other device to the drop area at any time to perform further comparisons.

Starting detailed comparisons using the compare editor

To start a detailed comparison for a PLC data type using the compare editor, follow these steps:

1. Perform an offline/offline comparison. You can also perform an online/offline comparison for CPUs of the S7-1200/1500 series.
2. For an automatic offline/offline comparison in the compare editor, select the PLC data type for which you want to run a detailed comparison. Note that two PLC data types must be selected for comparison with a manual offline/offline comparison.
3. Click the "Start detailed comparison" button in the toolbar.
   The "PLC data type comparison" window opens and all tags of the compared PLC data types are displayed. You can view the differences under "Info > Result of comparison" in the inspector window.

Starting detailed comparison in the project tree

To start a offline/offline detailed comparison for a PLC data type directly in the project tree, follow these steps:

1. Right-click the PLC data type that you want to compare. This can also be a PLC data type from a reference project.
2. Select the command "Fast comparison > Select as left object" in the shortcut menu.
3. Right-click the PLC data type that you want to compare with the PLC data type that you previously selected as left object.
4. Select the command "Fast comparison > Compare with <selected object>" in the shortcut menu. "<selected object> stands for the left comparison object.
   The "PLC data type comparison" window opens and all tags of the compared PLC data types are displayed. You can view the differences under "Info > Result of comparison" in the inspector window.

To start a offline/online detailed comparison for a PLC data type for a CPU of the S7-1200/1500 series directly in the project tree, follow these steps:

1. Establish an online connection to the device where the block is located.
2. Right-click the PLC data type that you want to compare with its online object.
3. Select the command "Fast comparison > Compare with the online object" in the shortcut menu.
   The "PLC data type comparison" window opens and all tags of the compared PLC data types are displayed. You can view the differences under "Info > Result of comparison" in the inspector window.

See also

Introduction to comparing PLC programs (Page 5083)

Using the compare editor (Page 362)

Comparing blocks (Page 5088)

Comparing PLC tags (Page 5115)

11.17 Displaying program information

11.17.1 Overview of available program information

Program information

The program information of a user program contains the view specified in the following table.

View	Application
Assignment list (Page 5121)	Provides an overview of the address bits for the I, Q, and M memory areas already allocated within the user program. Also indicates if an address has been allocated by access from an S7 program or if the address has been assigned to a SIMATIC S7 module.
Call structure (Page 5129)	Shows the call structure of the blocks within the user program and provides an overview of the blocks used and their relationships.
Dependency structure (Page 5135)	Shows the list of blocks used in the user program. A block is shown at the first level and blocks that call or use this block are indented below it. In contrast to the call structure, instance blocks are listed separately.
Resources (Page 5141)	Shows the hardware resources of the CPU for objects (OB, FC, FB, DB, user-defined data types and PLC tags), for CPU memory areas and for the existing I/O modules.

Displaying several views simultaneously

You can generate and display several views for one or more user programs to facilitate testing and changing your user program.

Displaying multiple views, for example, enables you to:

- Display all program information for a user program next to one another
- Compare different user programs

11.17.2 Displaying an assignment list

11.17.2.1 Introduction to the assignment list

Program information in the assignment list

The assignment list shows if an address has been allocated by access from an S7 program or if the address has been assigned to a SIMATIC S7 module. It is therefore an important basis for locating errors or changes in the user program.

In the assignment list, you have a CPU-specific overview of which bit is used in which byte of the memory areas listed below:

- Input (I)
- Output (O)
- Bit memory (M)
- Timer (T)
- Counter (C)
- I/O (P)

Display of the assignment list

The assignment list of inputs, outputs, and bit memory is displayed in several separate work windows.

Filters

You can filter the display within the assignment list. You can use predefined filters or create your own.

Displaying cross-reference information

You have the option of displaying cross-reference information for selected addresses in the assignment list.

You can display the cross-references for a selected address in the Inspector window using the "Cross-reference information" shortcut menu command. The command "Tools > Cross-references" allows you to also open the cross-reference list for the selected object.

Displaying the PLC tag table

You can open the PLC tag table from the assignment list and edit the properties of the tags used.

To do this select an address of the assignment list and select the "Open editor" command in the shortcut menu.

Enabling the display of retentivity

You can enable and disable the display of the retentive state of bit memory by selecting the "Hide/show retain area" toolbar button.

See also

Symbols in the assignment list (Page 5123)

Layout of the assignment list (Page 5122)

11.17.2.2 Layout of the assignment list

Layout of the assignment list

Depending on the CPU, the assignment list is displayed in several work windows with the following operands.

For S7-300/400 and S7-1500 CPUs:

- Inputs
- Outputs
- Bit memories
- Timers
- Counters

For S7-1200 CPUs:

- Inputs
- Outputs
- Bit memories

Displaying inputs, outputs, bit memory, timers and counters

It shows all operands used and their assignment in the S7 program.

For all displayed operands, each line in the assignment list is dedicated to a byte of the memory area, in which the corresponding eight bits from 7 to 0 are labeled according to their access. Afterwards, a "bar" indicates whether access is made by a byte (B), word (W) or double word (D).

You can find an explanation of the symbols in the assignment list here: (Page 5123)

See also

Introduction to the assignment list (Page 5121)

11.17.2.3 Symbols in the assignment list

Meaning of the symbols in the assignment list

The following table shows the meaning of the symbols in the assignment list:

Symbol	Meaning
	Indicates the address assignment in the selected state.
	Indicates the address assignment in the non-selected state.
	Indicates that a pointer start address and a tag address access the same address range and that they are selected.
	Indicates that a pointer start address and a tag address access the same address range and that they are not selected.
	Indicates the pointer assignment in the selected state.
	Indicates the pointer assignment in the non-selected state.
	Indicates that the byte is in use with byte access and the corresponding tag is selected. The shortcut menu allows you to display cross-reference information for the selected variables as well as the PLC tag table.
	Indicates that the byte is in use with byte access and the corresponding tag is not selected.
	Indicates that the byte is in use with word access and the corresponding tag is selected. The shortcut menu allows you to display cross-reference information for the selected variables as well as the PLC tag table.
	Indicates that the byte is in use with word access and the corresponding tag is not selected.
	Indicates that the byte is in use with double word access and the corresponding tag is selected. The shortcut menu allows you to display cross-reference information for the selected variables as well as the PLC tag table.
	Indicates that the byte is in use with double word access and the corresponding tag is not selected.
Background color: gray	Indicates that a byte is in use with byte, word or double word access and that the address is also in use by the hardware. The gray background color indicates overlapping memory access.
Background color: yellow	Indicates that the address is not in use by the hardware.
	Indicates that the memory area has been defined as system memory.
	Indicates that the memory area has been defined as clock memory.

See also

Layout of the assignment list (Page 5122)

Introduction to the assignment list (Page 5121)

11.17.2.4 Displaying an assignment list

Requirement

A project has been created with programmed blocks.

Procedure

Proceed as follows to display the assignment list:

1. Select the "Program blocks" folder or one or more of the blocks it contains.
2. Select the "Assignment list" command in the "Tools" menu.

Result

The assignment list for the selected program is displayed.

View options in the assignment list

Refer to view respective view options that are set to display the desired information in the assignment list.

See also

Setting the view options for the assignment list (Page 5124)

Layout of the assignment list (Page 5122)

11.17.2.5 Setting the view options for the assignment list

Introduction

The following view options are available for the assignment list:

- Used addresses:
  When this check box is activated, the addresses, I/Os and pointers used in the program are displayed.
- Free hardware addresses:
  When this check box is activated, only the free hardware addresses are displayed.

Requirement

- A project has been created with programmed blocks.
- The assignment list is open.

Procedure

Proceed as follows to set the view options for the assignment list:

1. Click on the arrow of the  symbol ("View options") in the task bar.
   The view options for the assignment list are opened. Check marks are set in front of the activated view options.
2. If you want to activate or deactivate a view option, click on the respective check box and set or remove the check mark.

Result

The view options are set and the desired information is displayed in the assignment list.

11.17.2.6 Filter options in the assignment list

Filter settings

You can define your own filter settings for the assignment list. The following options are available for defining filters:

- Display all addresses of the address areas specified.
- Display of single, defined addresses from the selected address area, for example, "0" and "200".
- Display of complete areas from the selected address area, for example, "0 - 256".

The following table provides an overview of all available options:

Selection in the	Selection	Symbol	Meaning
Address area	All CPU-dependent displayed addresses (I, O, M, T, C) can be activated as they are by default, or individual address areas can be activated.	Check box is activated	Only the activated address areas (I, O, M, T, C) are shown in the assignment list.
Filter area	Show assignment for all addresses	*	Displays the assignment of all addresses of the enabled address areas (I, Q, M).
	Show assignment for selected addresses, for example, for the inputs "IB 0" and "IB 256"	0;256 Separate individual addresses and areas by a semicolon.	Assignments of selected addresses for the activated address areas (I) are shown.
	Show assignment for selected areas, for example, for the inputs "IB 0 to IB 100" and "IB 200 to IB 256".	0-100;200-256 Contiguous areas should be connected by a hyphen.	Assignments of selected areas for the activated address areas (I) are shown.

11.17.2.7 Defining filters for assignment list

Requirement

- A project has been created with programmed blocks.
- The assignment list is open.

Defining filter

Proceed as follows to define a filter for the assignment list:

1. Click on the symbol ("Filter") in the task bar.
   The "Assignment List Filter" dialog opens.
2. Click on the symbol ("Create new filter") in the task bar.
   A new filter is created with the name "Filter_1". The check boxes for all addresses (inputs, outputs, memory bits, timers and counters) are activated by default for the filter.
3. If you want to change the name of the filter, click on the drop-down list in the task bar and enter a new filter name.
4. Deactivate the check boxes of addresses that are not to be affected by the filter.

5. Enter one of the following options in the filter area of the activated address:
   - Show all addresses used = "*"
   - Show single, defined addresses, for example, IB 0" and IB 25 = "0.25". Individual addresses and address areas are separated by commas or semicolons.
   - Show complete address areas, for example, IB 0 to IB 256 = "0-256". Complete address areas should be connected by a hyphen.
6. Confirm your entries with "OK".
   The newly defined filter is shown in the task bar of the assignment list under the specified name.

Delete filter

Proceed as follows to delete a filter:

1. Click on the symbol ("Filter") in the task bar.
   The filter dialog for the assignment list opens.
2. In the drop-down list of the task bar, select the filter you want to delete.
3. Click on the symbol ("Delete selected filter") in the task bar.
   The selected filter is deleted.

See also

Filter options in the assignment list (Page 5125)

Displaying an assignment list (Page 5124)

Introduction to the assignment list (Page 5121)

11.17.2.8 Filtering an assignment list

Requirement

- A project has been created with programmed blocks.
- The assignment list is open.

Procedure

1. Click on the arrow on the drop-down list.
   The available filter are displayed.
2. Select the desired filter.

Result

The assignment list is filtered according to the settings of the selected filter.

Note

The filter settings are saved when the project is closed.

11.17.2.9 Defining retentive memory areas for bit memories

Introduction

In the assignment list you can define the width of the retentive memory area for bit memories. The content of tags which are addressed in retentive memory is retained after power off and at the STOP to RUN transition after power on.

The display of retentive bit memories can be enabled and disabled in the assignment list. If their display is enabled, retentive bit memories are identified by an icon in the "Address" column.

Requirement

The assignment list is open.

Procedure

Proceed as follows to define the width of the retentive memory area for bit memories:

1. Click "Retain" in the toolbar.
   The "Retain memory" dialog will open.
2. Starting at the count of 0, define the width of the retentive memory area by entering the last byte of this area in the input field. Watch out for any addresses of tags already assigned to the retentive area.
3. Load the block to the target system. Select the "Program blocks" folder in the Project tree and select the "Download to device" submenu in the shortcut menu.

Result

The width of the retentive memory area is defined. If enabled in the assignment list, an icon will indicate the retentive state of all tags in the "Address" column.

11.17.2.10 Enabling the display of retentive bit memories

Introduction

In the assignment list you can enable and disable the display of retentive bit memories. The retentive bit memories are identified by means of an icon in the "Address" column if the display of retentivity is enabled.

Requirement

The assignment list is open.

Procedure

Proceed as follows to enable and disable the display of retentive bit memories:

1. Click "Display/hide retentivity" in the toolbar.

Result

The retentive tags are identified by means of an icon in the "Address" column of the bit memory area if the display of retentivity is enabled. The icons in the "Address" column are hidden if the display of retentivity is disabled.

11.17.3 Displaying the call structure

11.17.3.1 Introduction to the call structure

Call structure

The call structure describes the call hierarchy of the blocks within an S7 program.

It provides an overview of:

- The blocks used
- Jumps to the places of use of the blocks
- Dependencies between blocks
- Local data requirements of the blocks
- Status of the blocks

Information in the call structure

Displaying the call structure provides you with a list of the blocks used in the user program. The first level of the call structure is highlighted in color and shows the blocks that are not called by any other block in the program. Organization blocks are always shown on the first level of the call structure. Functions, function blocks and data blocks are only shown on the first level if they are not called by an organization block. When a block calls other blocks or functions, they are listed indented under the calling block. Instructions and blocks are shown in the call structure only if they are called by a block.

Note

Please note that no call structure is displayed for know-how-protected blocks.

View options

The following view options are available for the call structure:

- Show conflicts only:
  When this check box is activated, only conflicts within the call structure are displayed.
- Group multiple calls together:
  When this check box is activated, several block calls are grouped together. The number of block calls is displayed in the "Call frequency" column. The links to the various call locations are offered in a drop-down list in the "Details" column.

Displaying the block calls

You can display the block calls in a block by clicking on the arrow in front of the block title. To display the call information of all blocks, click on the "Expand list" icon in the toolbar.

You can hide the total overview by clicking the "Collapse list" icon.

Displaying cross-reference information

You can display the cross-reference information for a block in the Inspector window by right-clicking on the relevant block and selecting the "Cross-reference information" command from the shortcut menu.

To open the "Cross-references" view, click the "Cross-references" command in the shortcut menu.

Displaying blocks in the program editor

You can open the program editor from the call structure and edit blocks there.

To do this, select the required block in the call structure and select the "Open" command in the shortcut menu.

Displaying deleted blocks

The rows belonging to deleted blocks are identified by an icon.

Note

Please note that any existing local data can only be displayed or updated after a block is compiled.

See also

Symbols in the call structure (Page 5131)

11.17.3.2 Symbols in the call structure

Meaning of the symbols in the call structure

The following table shows the meaning of the symbols in the call structure:

Symbol	Meaning
	Indicates an organization block (OB).
	Indicates a function block (FB).
	Indicates a function (FC).
	Indicates a data block (DB).
	Indicates that the block is declared as a multiinstance.
	The object has an interface dependency to an object connected to the left.
	Indicates that the block needs to be compiled again.
	Indicates that the data block needs to be compiled again.
	Indicates that the object is not available.
	Indicates that the interface causes a time stamp conflict.
	Indicates that the variable causes a time stamp conflict.
	Indicates that the block is not called directly or indirectly from an OB.
	Indicates that an object has know-how protection.

Symbol	Meaning
	Indicates that the block is normally called recursively.
	Indicates that a tag declaration in the interface has a recursive dependency: • Scenario 1: FB1 calls FB2 and this then calls FB1. The instance data blocks of these FBs have a recursion in the interface. • Scenario 2: A multiple instance FB uses the instance DB of its parent FB as a global DB.

11.17.3.3 Layout of the call structure

Layout of the call structure

The view of the call structure consists of the following columns:

Column	Content/meaning
Call structure	Shows an overview of the blocks called If the viewing option "Group multiple calls together" is enabled, several block calls are grouped together and the "Number of calls" column is displayed.
Call type (!)	Shows the type of call, for example recursive block call.
Address	Shows the absolute address of the block. With a function block, the absolute address of the corresponding instance data block is also shown.
Call frequency	Indicates the number of multiple calls of blocks.
Details	Shows the network or interface of the calling block. All information are offered as a link in this column. With this link, you can jump to the location of the block call in the program editor. If the viewing option "Group multiple calls together" option is enabled, the calls are grouped together and are available as links in a drop-down list.
Local data (in path)	Indicates the local data requirement of the full path. Blocks with optimized access have higher local data requirements because the information for the symbolic addressing is stored with them. Please note that any existing local data can only be displayed or updated after compiling a block.
Local data (for blocks)	Show the local data requirements of the block. Blocks with optimized access have higher local data requirements because the information for the symbolic addressing is stored with them. Please note that any existing local data can only be displayed or updated after compiling a block.

See also

Symbols in the call structure (Page 5131)

Introducing the consistency check in the call structure (Page 5134)

11.17.3.4 Displaying the call structure

Requirement

A project has been created with blocks.

Procedure

Proceed as follows to display the call structure:

1. Select the "Program blocks" folder or one or more of the blocks it contains.
2. Select the "Call structure" command in the "Tools" menu.

Result

The call structure for the selected program is displayed.

Note

Please note that any existing local data can only be displayed or updated after compiling a block.

See also

Setting the view options for the call structure (Page 5133)

11.17.3.5 Setting the view options for the call structure

Introduction

The following view options are available for the call structure:

- Show conflicts only:
  Only the blocks causing conflicts within the call structure are displayed if this check box is activated.
  The following blocks cause conflicts:
  - Blocks executing any calls with older or newer code time stamps.
  - Blocks calling a block with modified interface.
  - Blocks using a tag with modified address and/or data type.
  - Block called neither directly, nor indirectly by an OB.
  - Blocks calling a block which no longer exists.
- Group multiple calls together:
  When this viewing option is enabled, several block calls and data block accesses are grouped together. The number of block calls is displayed in the "Call frequency" column. The links to the various call locations are offered in a drop-down list in the "Details" column.

Requirement

- A project has been created with programmed blocks.
- The call structure is open.

Procedure

Proceed as follows to set the view options for the call structure:

1. Click on the arrow of the  symbol ("View options") in the task bar.
   The view options for the call structure opens. Check marks are set in front of the activated view options.
2. If you want to activate or deactivate a view option, click on the respective check box and set or remove the check mark.

Result

The view options are set and the required information is displayed in the call structure.

11.17.3.6 Introducing the consistency check in the call structure

Consistency check

Changing the time stamp of a block during or after the program is generated can lead to time stamp conflicts, which in turn cause inconsistencies among the blocks that are calling and being called.

Using the consistency check

The "Consistency check" function is used to visualize inconsistencies when time stamp conflicts occur. Whe the consistency check is performed, the inconsistent blocks are shown in the call structure and marked with the correspoinding symbols.

- Most time stamp and interface conflicts can be rectified by recompiling the blocks.
- If compilation fails to clear up inconsistencies you can use the link in the "Details" column to go to the source of the problem in the program editor and manually eliminate any inconsistencies.
- The blocks marked in red must be recompiled.

See also

Symbols in the call structure (Page 5131)

11.17.3.7 Checking block consistency in the call structure

Requirement

- A project has been created with programmed blocks.
- The call structure is open.

Procedure

Proceed as follows to check the block consistency:

1. Click on the  symbol ("Consistency check") in the task bar.
   The block consistency is checked. Blocks found to be inconsistent are marked accordingly by a symbol.
2. If a block is inconsistent, click on the arrow in front of the block title in the call structure.
   The inconsistent blocks are displayed. The exact problem locations are listed as links in the "Details" column.
3. Click on the respective link in the "Details" column to jump to the location in the block requiring correction.
4. Check and correct the inconsistencies in the blocks.
5. Recompile the blocks by selecting the required blocks and clicking on the command "Compile" in the shortcut menu.
6. Download the corrected blocks to the target system by clicking the command "Download to device" in the shortcut menu.

Result

The block consistency is checked. The inconsistencies in the blocks are corrected. The corrected blocks are loaded to the target system.

See also

Symbols in the call structure (Page 5131)

11.17.4 Displaying the dependency structure

11.17.4.1 Introduction to the dependency structure

Introduction

The dependency structure shows the dependencies each block in the program has to other blocks.

Information in the dependency structure

Displaying the dependency structure provides you with a list of the blocks used in the user program. A block is shown at the far left, and blocks that call or use this block are indented below it.

The dependency structure also shows the status of the individual blocks using symbols.

Objects that cause a time stamp conflict and could lead to an inconsistency in the program are marked with various symbols.

The dependency structure is an extension of the cross-reference list for objects.

View options

The following view options are available for the dependency structure:

- Show conflicts only:
  When this check box is activated, only the conflicts within the dependency structure are displayed.
- Group multiple calls together:
  When this check box is activated, several block calls are grouped together. The number of block calls is shown numerically in the "Dependency structure" column. The links to the various call locations are offered in a drop-down list in the "Details" column.

Displaying the dependencies

Clicking on the arrow in front of the block title displays the blocks that call or use this block. To display the dependencies of all blocks,

click the "Expand list" icon in the toolbar.

You can hide the total overview by clicking the "Collapse list" icon.

Displaying cross-reference information

You can display the cross-reference information for a block in the Inspector window by right-clicking on the relevant block and selecting the "Cross-reference information" command from the shortcut menu.

To open the "Cross-references" view, click the "Cross-references" command in the shortcut menu.

Displaying blocks in the program editor

You can open the program editor from the dependency structure and edit blocks there. To do this, select the required block in the dependency structure and select the "Open" command in the shortcut menu.

11.17.4.2 Layout of the dependency structure

Layout of the dependency structure

The view of the dependency structure consists of the following columns:

Column	Content/meaning
Dependency	It indicates the dependencies between each block and the other blocks in the program.
Call type (!)	Shows the type of call, for example recursive block call.
Address	Shows the absolute address of the block.
Call frequency	Indicates the number of multiple calls of blocks.
Details	Shows the network or interface of the called block. All information are offered as a link in this column. With this link, you can jump to the location of the block call in the program editor. If the viewing option "Group multiple calls together" option is enabled, the calls are grouped together and are available as links in a drop-down list.

See also

Symbols in the dependency structure (Page 5137)

11.17.4.3 Symbols in the dependency structure

Meaning of the symbols in the dependency structure

The following table shows the meaning of the symbols in the dependency structure:

Symbol	Meaning
	Indicates an organization block (OB).
	Indicates a function block (FB).
	Indicates a function (FC).
	Indicates a data block (DB).
	The object has an interface dependency to an object connected to the left.
	Indicates that the block needs to be compiled again.
	Indicates that the data block needs to be compiled again.
	Indicates that there is an inconsistency with this object.

Symbol	Meaning
	Indicates that an object has know-how protection.
	Indicates that a tag declaration in the interface has a recursive dependency: • Scenario 1: FB1 calls FB2 and this then calls FB1. The instance data blocks of these FBs have a recursion in the interface. • Scenario 2: A multiple instance FB uses the instance DB of its parent FB as a global DB.

11.17.4.4 Displaying the dependency structure

Requirement

A project has been created with programmed blocks.

Procedure

Proceed as follows to display the dependency structure:

1. Select the block folder or one or more of the blocks contained therein.
2. Select the "Dependency structure" command in the "Tools" menu.

Result

The dependency structure for the selected program is displayed.

See also

Setting the view options for the dependency structure (Page 5139)

11.17.4.5 Setting the view options for the dependency structure

Introduction

The following view options are available for the dependency structure:

- Show conflicts only:
  When this check box is activated, only the conflicts within the dependency structure are displayed.
  The following blocks cause conflicts:
  - Blocks executing any calls with older or newer code time stamps.
  - Blocks called by a block with modified interface.
  - Blocks using a tag with modified address and/or data type.
  - Block called neither directly, nor indirectly by an OB.
- Group multiple calls together:
  When this check box is activated, several block calls are grouped together. The number of block calls is shown in the relevant column. The links to the various call locations are offered in a drop-down list in the "Details" column.

Requirement

- A project has been created with programmed blocks.
- The dependency structure is open.

Procedure

Proceed as follows to set the view options for the dependency structure:

1. Click on the arrow of the  symbol ("View options") in the task bar.
   The view options for the dependency structure are opened. Check marks are set in front of the activated view options.
2. If you want to activate or deactivate a view option, click on the respective check box and set or remove the check mark.

Result

The view options are set and the required information is displayed in the dependency structure.

11.17.4.6 Introducing the consistency check in the dependency structure

Consistency check

Changing the time stamp of a block during or after the program is generated can lead to time stamp conflicts, which in turn cause inconsistencies among the blocks that are calling and being called.

Using the consistency check

The "Consistency check" function is used to visualize inconsistencies. Whe the consistency check is performed, the inconsistent blocks are shown in the dependency structure and marked with the correspoinding symbols.

- Most time stamp and interface conflicts can be rectified by recompiling the blocks.
- If compilation fails to clear up inconsistencies you can use the link in the "Details" column to go to the source of the problem in the program editor and manually eliminate any inconsistencies.
- The blocks marked in red must be recompiled.

See also

Layout of the dependency structure (Page 5137)

Symbols in the dependency structure (Page 5137)

11.17.4.7 Checking block consistency in the dependency structure

Requirement

- A project has been created with programmed blocks.
- The dependency structure is open.

Procedure

Proceed as follows to check the block consistency:

1. Click on the  symbol ("Consistency check") in the task bar.
   The block consistency is checked. Blocks found to be inconsistent are marked accordingly by a symbol.
2. If a block is inconsistent, click on the arrow in front of the block title in the dependency structure.
   The inconsistent blocks are displayed. The exact problem locations are listed as links in the "Details" column.
3. Check and correct the inconsistencies in the blocks.
4. Recompile the blocks by selecting the required blocks and clicking on the command "Compile" in the shortcut menu.
5. Download the corrected blocks to the target system by clicking the command "Download to device" in the shortcut menu.

Result

The block consistency is checked. The inconsistencies in the blocks are corrected. The corrected blocks are loaded to the target system.

See also

Symbols in the dependency structure (Page 5137)

11.17.5 Displaying CPU resources

11.17.5.1 Introducing resources

Introduction

The "Resources" tab indicates the hardware resources of the configured CPU for:

- the used programming objects,
- the assignment of the different memory areas within the CPU and
- the assigned inputs and outputs of the existing input and output modules.

Information provided in the "Resources" tab

The resources tab provides an overview of the hardware resources. The display in this tab depends on the CPU which you are using. The following information is displayed:

- the programming objects used in the CPU (e.g. OB, FC, FB, DB, data types and PLC tags)
- the memory areas available on the CPU (load memory, work memory - divided into code work memory and data work memory depending on the CPU -, retentive memory), their maximum size and utilization by the programming objects stated above
- the I/O of modules which can be configured for the CPU (I/O modules, digital input modules, digital output modules, analog input modules, and analog output modules), including the I/O already in use.

Display of the maximum available load memory

The maximum size of available load memory can be selected from a drop-down list box in the "Total" row of the "Load memory" column.

Note

Displaying the allocated load memory in the CPU

Note that the sum of the allocated load memory cannot be precisely determined if not all blocks are compiled.

In this case, a preceding ">" in front of the sum indicates that the amount of allocated space may be greater than that shown because uncompiled blocks are not taken into account in the total.

Display of the maximum available work memory

The maximum size of available work memory is displayed in the "Work memory" column or in the "Code work memory" and "Data work memory" columns in the "Total" row.

Display of the maximum available retentive memory

The maximum size of available retentive memory is displayed in the "Retain memory" column in the "Total" row.

Note

Retentive memory data

All bit memories and data blocks specified as retentive will be integrated in the calculation of the retentive data.

Updating the display in the "Resources" tab

Click the "Update view" toolbar button to update the display of objects.

Benefits of the display in the "Resources" tab

The "Resources" tab of the program information dialog provides a detailed list of all objects and of the corresponding memory area used.

The tab also indicates shortage of resources and helps to avoid such states.

Blocks which are not compiled can be identified as their size is indicated by a question mark.

See also

Layout of the "Resources" tab (Page 5143)

Displaying resources (Page 5144)

Selecting the maximum load memory available (Page 5145)

11.17.5.2 Layout of the "Resources" tab

Layout of the "Resources" tab in the program information

The view of the "Resources" tab consists of the following columns:

Column	Content/meaning
Objects	The "Details" area provides an overview of the programming objects available in the CPU, including their memory assignments.
Load memory	Displays the maximum load memory resources of the CPU as a percentage and as absolute value. The values displayed under "Total" provide information on the maximum memory available in the load memory. The values displayed under "Used" provide information on the memory actually used in the load memory. If a value is displayed in red, the available memory capacity has been exceeded.
Work memory or code and data work memory	Displays the maximum work memory resources of the CPU as a percentage and as absolute value. The work memory depends on the CPU and is divided into "Code work memory" and "Data work memory" for a CPU from the S7-400 or S7-1500 series, for example. The values displayed under "Total" provide information on the maximum memory available in the work memory. The values displayed under "Used" provide information on the memory space actually used in the work memory. If a value is displayed in red, the available memory capacity has been exceeded.
Retentive memory	Displays the maximum resources for retentive memory in the CPU as a percentage and as absolute value. The values displayed under "Total" provide information on the maximum memory available in the retentive memory. The values displayed under "Used" provide information on the memory actually used in the retentive memory. If a value is displayed in red, the available memory capacity has been exceeded.

Column	Content/meaning
I/O	Displays the I/Os which are available on the CPU, including their module-specific availability in the next columns. The values displayed at "Configured" provide information about the maximum number of I/O available. The values displayed under "Used" provide information on the actually used inputs and outputs.
DI / DQ / AI / AQ	Displays the number of configured and used inputs/outputs: DI = Digital inputs DQ = Digital outputs AI = Analog inputs AQ = Analog outputs The values displayed at "Configured" provide information about the maximum number of I/O available. The values displayed under "Used" provide information on the actually used inputs and outputs.

See also

Displaying resources (Page 5144)

Selecting the maximum load memory available (Page 5145)

Introducing resources (Page 5141)

11.17.5.3 Displaying resources

Introduction

The memory use shows you which space is allocated to specific objects in the CPU.

Note

Displaying the allocated load memory in the CPU

Note that the sum of the allocated load memory cannot be precisely determined if not all blocks are compiled.

In this case, a preceding ">" in front of the sum indicates that the amount of allocated space may be greater than that shown because uncompiled blocks are not taken into account in the total.

Requirement

A project with programmed blocks has been created.

Procedure

Proceed as follows to display the resources of the respective CPU memory areas:

1. Select the block folder below the relevant CPU, or one or several of the blocks contained therein.
2. Select the "Resources" command in the "Tools" menu.

Result

The memory resources of the assigned CPU are displayed.

11.17.5.4 Selecting the maximum load memory available

Requirement

A project with programmed blocks has been created.

Procedure

Proceed as follows to display the available maximum of load memory resources:

1. Select the block folder below the relevant CPU, or one or several of the blocks contained therein.
2. Select the "Resources" command in the "Tools" menu.
3. In the dialog that is displayed, open the drop-down list in the "Total" field of the "Load memory" column by clicking the icon.
4. Select a corresponding value for the CPU used by clicking it in the drop-down list box.

Result

The "Total" field displays the selected maximum memory resources.

Note

Display of maximum memory resources

If a value is displayed in red for the maximum memory resources, the available memory capacity has been exceeded.

In this case, adapt the memory capacity as described above.

11.18 Displaying cross-references

11.18.1 General information about cross references

Introduction

The cross-reference list provides an overview of the use of objects and devices within the user program.

You can see which objects are interdependent, what relationships they have with one another and where the individual objects are located.

You can also jump directly to the reference location of an object.

Uses of cross-references

The cross-reference list offers you the following advantages:

- When creating and changing a program, you retain an overview of the devices and objects used such as the blocks, operands and tags etc.
- You can see whether the object uses other objects or is used by itself.
- You can filter the displayed cross-reference information by specific criteria. Predefined filters for source and reference objects are available to you to this purpose.
- In addition, you can also create user-defined filters to find relevant references more easily.
- From the cross-references, you can jump directly to the reference location of the selected object.
- During a program test or when troubleshooting, you are informed of the following, for example:
  - In which block and by which command is an operand processed.
  - Which tag is used how and where.
  - Which block is called by which other block.
  - Cross-reference information for subordinate and higher-level structures.
- As part of the project documentation, the cross-references provide a comprehensive overview of all operands, memory areas, blocks, tags, screens etc. used.

Note

Objects displayed in the cross-reference list

Which objects you can display in the cross-reference list depends on the installed products.

See also

Structure of the cross-reference list (Page 5147)

Displaying the cross-reference list (Page 5152)

Displaying and editing cross-references in the Inspector window (Page 5172)

11.18.2 Structure of the cross-reference list

Introduction

The cross-reference list enables the display of reference objects and lower-level objects for a source object selected in the project tree.

The source object is the object displayed first in the cross-reference list.

The display of the cross-references for an object is project- and device-specific.

You can also have multiple cross-reference lists open at the same time.

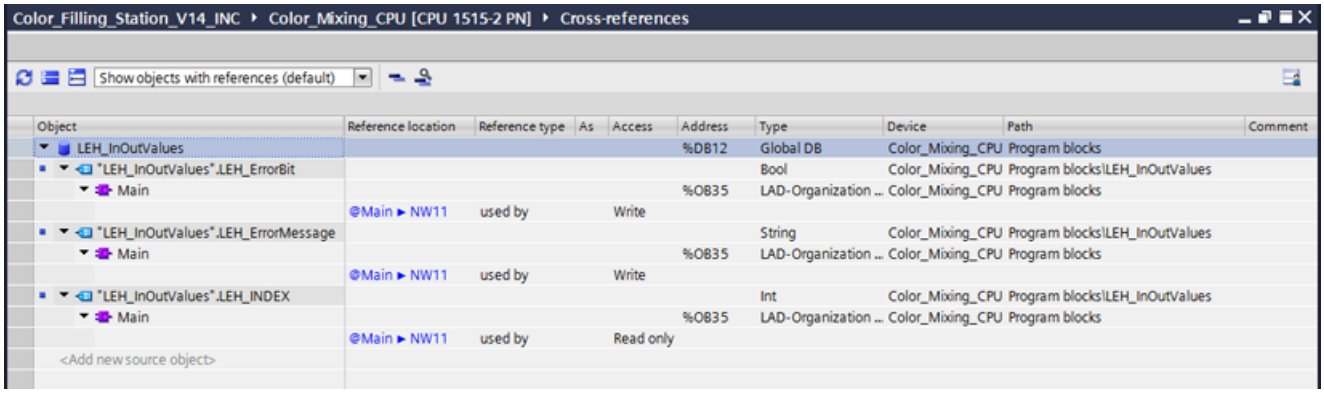
Structure of the cross-reference list

When you have selected a source object and open the cross-reference list, the objects are displayed as follows:

- The source object and all the lower-level objects of this source object and the associated references are structured in a standard view.
- The source object is the object for which you have opened the cross-references and is displayed as the first object in a row with a gray background.
- Lower-level objects are marked with a preceding blue square.
- If lower-level objects also have reference objects, they are displayed with a gray background in the object column and in an hierarchical structure.
- If an object is used several times, it is also displayed several times.
- The "Add new source object" button is displayed at the end of a filled cross-reference list. When you click on this button, you can add a new object to the cross-reference list.
- The source object and the lower-level object are collapsed in the Inspector window. Lower-level objects with reference objects are show with a gray background in the object column.

Example

The graphic below shows an example of the structure of the cross-reference list to a source object:



Display of the newly added source objects in the cross-reference list

Objects which were inserted using the "Add new source object" button are displayed as follows:

- They are always at the end of the cross-reference list and are automatically expanded.
- They are initially displayed independently of the selected filter setting.
- Newly added objects are correctly sorted and the filter settings displayed accordingly after an update or when another filter is used.

Columns in the cross-reference list

The cross-reference list has the following structure:

Column	Content/meaning
Object	Displays the name of the source object which was selected when the cross-reference list was opened, with all lower-level objects and the associated reference objects.
Reference location	Shows the reference location of the objects, e.g. a network.

Column	Content/meaning
Reference type	Shows the relationship between the source object and the referenced objects: • "Uses": The source object uses this object. • "Used by": The source object is used by this object. • "Type -> Instance": The source object is a type of the referenced object. • "Instance -> Type": The source object is an instance of the referenced object. • "Group -> Element": The source object is a group of the referenced object. • "Element -> Group": The source object is an element of the referenced object. • "Defines": The source object defines the referenced object. • "Defined by": The source object is defined by the referenced object.
As	Shows additional information about objects, e.g., that a tag is used by several devices.
Access	Shows the type of access, e.g. whether access to the operand is read access (R) and/or write access (W).
Address	Shows the address of the relevant object.
Type	Shows the type and the language used to create the object.
Device	Shows the associated device name, e.g. "CPU_1".
Path	Displays the path of the object in the project tree with specification of folders and groups.
Comment	Displays comments on individual objects, if available.

See also

General information about cross references (Page 5146)

Displaying the cross-reference list (Page 5152)

11.18.3 Settings in the cross-reference list

Settings in the cross-reference list

Settings can be made in the cross-reference list using the buttons in the toolbar and the selection of pre-defined filter settings.

Settings using the buttons in the cross-reference list

The following settings can be made using the buttons:

- **"Update" button**
  Updates the contents displayed in the open cross-reference list.
  When you press this button, the displayed contents of the cross-reference list are updated with the current contents from the associated project. Automatic update is not carried out within the cross-reference list if data management has changed in the project.
- **"Expand all" button**
  Expands the entries in the current cross-reference list by opening all lower-level objects.
- **"Collapse all" button**
  Reduces the entries in the current cross-reference list by closing the lower-level objects.
- **"Save window settings" button**
  Saves the current window settings in the cross-reference list.
  This includes, for example, the columns displayed, the width of the columns and the arrangement of the columns.
  The relevant filter settings in the cross-reference list are **not** saved.

Filter settings in the cross-reference list

You have the possibility of selecting predefined filters for displaying objects.

You have the following options for filter selection, which relate to source objects:

- **Show objects with references (default):**
  This setting shows all the lower-level objects below the source object with existing references. It is the default setting whenever the cross-reference list is opened.
- **Show objects without references:**
  This setting shows all the lower-level objects below the source object without references. These non-referenced objects are assigned the entry "no references available" in the column "Reference location". Lower-level elements without references are also displayed if their parent objects contain references.
- **Show unused objects:**
  This setting shows all the lower-level but unused objects below the source object. These unused objects are assigned the entry "Object is not used" in the column "Reference location".
- **Show all objects:**
  This setting shows all the lower-level objects below the source object with or without references.

You have the following options for filter selection, which relate to reference objects:

- **Show only "uses":**
  This setting only shows the used reference objects with the reference type "Uses", "Belongs to", "Type instance" and "defined".
- **Show only "used by":**
  This setting shows all the reference objects with the reference type "Used by", "Type instance", "Contains" and "Defined by".
- **Show only local references:**
  This setting only shows local references which belong to a specific device (e.g. to CPU_1).

Sorting in the cross-reference list

The entries are displayed in alphabetical order, depending on the associated object, the associated device and the object type. Associated objects are displayed one after another.

The following sorting criteria are applied:

- For source objects: Sorting by object type, object name and number
- For referenced objects: Sorting by assignment to device, object name and number

Note

No sorting by name if more than 10,000 objects are displayed

Please note that no sorting by name can take place if more than 10,000 objects are displayed in the cross-reference list.

In this case, a banner text notifies you that no sorting is currently available for the displayed objects.

If you click the link in the banner text, you can enable sorting by name. A dialog box is then displayed which informs you about the progress of the sorting.

Display of columns in the cross-reference list

The columns displayed in the cross-reference list can be customized using the shortcut menu.

The shortcut menu offers the following options:

- Showing or hiding selected columns
- Displaying all existing columns
- Optimizing the column width for the selected column or for all columns.

Note

Showing or hiding selected columns

The columns "Object" and "Reference location" cannot be hidden in the cross-reference list.

You can save the settings you made by using the "Save window settings" button.

Adding new objects to the cross-reference list

You have the following options for inserting new source objects in an existing cross-reference list:

- By clicking the "Add new source object" button which is displayed at the end of the cross-reference list.

Note

Multiple insertion of identical objects

If you insert an identical object multiple times in the cross-reference list, the duplicates are automatically removed after the next update.

11.18.4 Displaying the cross-reference list

Introduction

Depending on the object selected in the project tree, all the cross-references belonging to this source object are displayed.

Alternatively, you have the following option to open the cross-reference list:

- Select the "Cross-reference" command from the "Tools" menu.
- Select "Cross-references" in the shortcut menu.
- Click on the icon for cross-references in the toolbar.
- Select an object in the project tree and click "F11".

The cross-references for the relevant selected object and for all the lower-level objects below it are always displayed.

The display of the cross-references for an object is project- and device-specific. You can also have multiple cross-reference lists open at the same time.

Note

Displaying cross-references

- Which objects you can display in the cross-reference list depends on the installed products.
- If more than 50 objects were selected, these are collapsed in the cross-reference list.
- If fewer than 50 objects were selected, these are expanded in the cross-reference list.

Requirements

You have created a project which contains objects with references.

Displaying cross-references

Proceed as follows to display cross-references in the cross-reference list:

1. In the project tree, select the required object and open the cross-references, e.g. using the shortcut menu.
2. The cross-reference list for the selected object is opened.
   - The relevant selected object is the "source object" and is at the top in the "Object" column.
   - In the column "Reference location", you can see the locations at which the objects shown in the cross-reference list can be used.
   - "Used by" and "Used", for example, show the relations between the objects in the "Reference type" column.
3. To jump to the reference location of the relevant object, click on the corresponding link.
4. You can perform the following actions using the buttons in the toolbar:
   - Update cross-reference list
   - Defining filters for the cross-reference list
   - Collapse entries
   - Expand entries
   - Saving settings for the cross-reference list
   - Check if overlapping accesses are available in the currently opened cross-reference list.
   - Display of the overlapping accesses in a separate area below the project window
5. You can sort the entries in the "Object" column, for example, in ascending or descending order by clicking on the corresponding column header.

See also

General information about cross references (Page 5146)

Structure of the cross-reference list (Page 5147)

11.18.5 Sorting the cross-reference list

Introduction

Sorting for the displayed source and reference objects is executed by default for the cross-reference list.

This sorting is also displayed accordingly in the Inspector window.

Note

No sorting by name if more than 10,000 objects are displayed

Please note that no sorting by name is possible if more than 10,000 objects are displayed in the cross-reference list.

In this case, a banner text notifies you that no sorting is currently available for the displayed objects.

If you click the link in the banner text, you can switch on the sorting. A dialog box is then displayed which informs you about the progress of the sorting.

Sorting criteria for displaying cross-references

Sorting takes place according to the following criteria:

For source objects and assigned objects:

- By object type: The source objects are ordered by their object type and displayed together.
- By name: In alphanumeric order

For reference objects:

- By device: The reference objects for a device are displayed together.
- By object type: The reference objects are ordered by their object type and displayed together.
- By name: In alphanumeric order

For the reference location:

- By name: In alphanumeric order

Requirements

You have created a project which contains objects with references.

Sorting the cross-reference list

To sort the cross-reference list when more than 10,000 objects are displayed, follow these steps:

1. To sort the cross-reference list by name, click the link in the displayed banner text.
   Result: A dialog notifies you about the progress in sorting the cross-references.
2. The displayed objects are also shown sorted by name after the completion of the sorting process.

11.18.6 Filter options for cross-references

11.18.6.1 Introduction to filtering of cross-references

Introduction

You have the option of filtering the cross-reference list in order to simplify the search for specific cross-references and to structure it better.

The valid filter setting is displayed in the drop-down list with a yellow background.

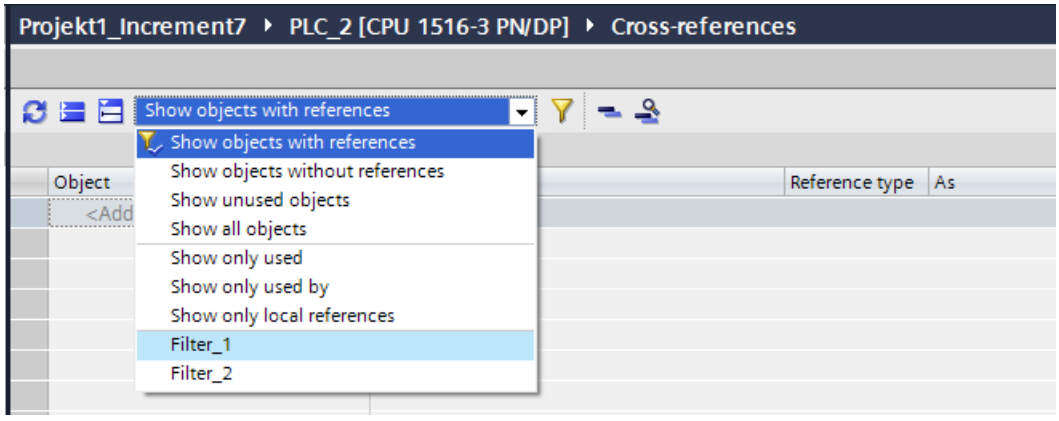
Filter settings predefined by the system (referred to as system filters below) are available via the drop-down list in the toolbar of the cross-references.

In addition to the filter options provided by the system, you can also create user-defined filters in accordance with your own requirements.

The system filters are always available in the drop-down list for filter selection and cannot be deleted.

User-defined filters are displayed after creation in the drop-down list for filter selection below the system filters and can be modified, renamed and deleted as required.

In the figure below you can see the system filters that are always available as well as two user-defined filters "Filter_1" and "Filter_2".



System-defined filter settings in the cross-reference list

The system filters shown in the upper area of the drop-down list are applicable for the displayed source objects; the filter objects in the lower area are used for the reference objects.

The following system filters are available:

- **Show objects with references (default):**
  This setting shows all the lower-level objects below the source object with existing references. It is the default setting whenever the cross-reference list is opened.
- **Show objects without references:**
  This setting shows all the lower-level objects below the source object without references. These non-referenced objects are assigned the entry "no references available" in the column "Reference location". Lower-level elements without references are also displayed if their parent objects contain references.
- **Show unused objects:**
  This setting shows all the lower-level but unused objects below the source object. These unused objects are assigned the entry "Object is not used" in the column "Reference location".
- **Show all objects:**
  This setting shows all the lower-level objects below the source object with or without references.
- **Show only "uses":**
  This setting only shows the used reference objects with the reference type "Uses", "Element -> Group", "Instance -> Type" and "Defined".
- **Show only "used by":**
  This setting shows all reference objects with the reference type "Used by", "Type -> Instance", "Group -> Element" and "Defined by".
- **Show only local references:**
  This setting only shows local references which belong to a specific device (e.g. to CPU_1).

As soon as you have selected one of these system filters via the drop-down list, the associated cross-references are filtered and displayed in accordance with the selected criteria.

See also Filtering the cross-reference list (Page 5166).

User-defined filters in the cross-reference list

You can also create user-defined filters for the cross-references in addition to the system filters that are always available.

This is possible both in the cross-reference editor and in the Inspector window in the "Info" > "Cross-references" tab.

Open the cross-references and click the filter icon in the toolbar of the cross-reference editor.

In the subsequent dialog click "Create new filter" and enter the desired filter criteria. Click "Apply" to display the associated cross-references in accordance with the selected filter criteria.

User-defined filters are displayed in the drop-drop-down list below the system filters with the preset name, for example as "Filter_1".

In contrast to the system filters, user-defined filters can be renamed, modified and deleted freely.

Invalid filters are displayed with a red background after being opened. These filters cannot be applied.

See also Creating user-defined filters for cross-references (Page 5157).

11.18.6.2 Creating user-defined filters for cross-references

Introduction

You can also create user-defined filters to display the cross-references.

After user-defined filters have been created and stored, they can also be selected via the drop-down list in the toolbar.

Newly created user-defined filters are available immediately in all currently open cross-reference editors.

This allows you to reduce the number of objects displayed in the cross-reference list and to search specifically for certain objects.

Note

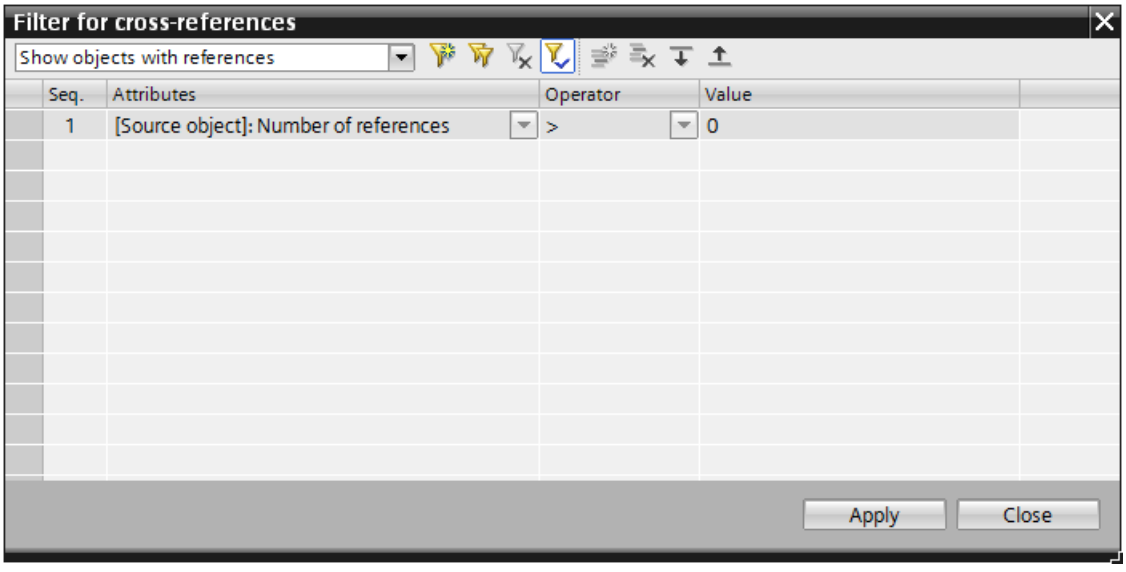
Creating user-defined filters

You can create user-defined filters in all cross-reference editors as well as in the Inspector window under "Info > Cross-references".

The procedure is identical in all cross-reference editors.

Structure of the filter dialog

The filter dialog has the following structure:



The filter dialog contains:

- The drop-down list for selecting existing filters.
  The filter displayed at the top in the drop-down list is the system filter taken from the open cross-reference list.
- The icons for creating, duplicating and deleting filters.

- The icon for specifying a filter as the default.
- The icons for adding, deleting and moving filter criteria.
- The columns "Seq." (sequence), "Attributes", "Operator " and "Value".
  The "Seq." column shows the number of the respective filter criteria.
  In the remaining columns you define the criteria required for new filters.
- "Apply" button: When you exit the filter dialog with "Apply", the changes are saved in the current user profile and are applied in the lower-level editor in the cross-reference list.
- "Close" button: When you exit the filter dialog with "Close", the changes are applied in the current user profile but are not applied in the editor.

Icons in the filter dialog

The filter dialog shows the following icons in the toolbar:

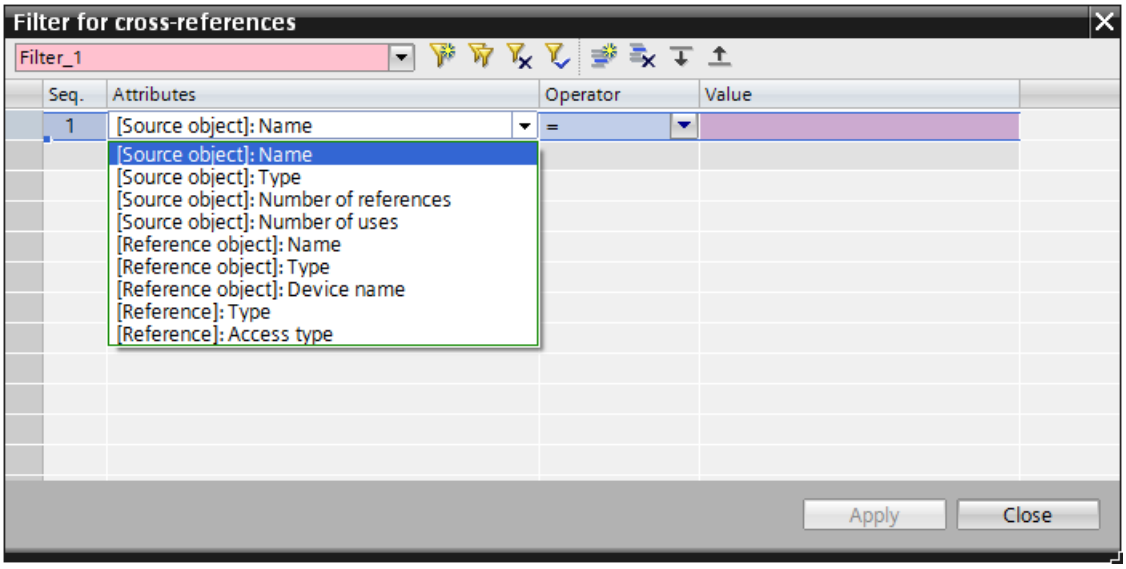
Icon	Meaning	Action
	Create new filter	Creates a new user-defined filter.
	Duplicate selected filter	Duplicates the selected filter so that existing filter criteria can be applied and processed further.
	Delete filter	Deletes the selected user-defined filter. The filter cannot be restored after it has been deleted! Note: System filters cannot be deleted!
	Apply filter as default setting	Applies the selected filter as default setting. This filter is used when the cross-references are opened again.
	Insert new criterion	Inserts a new filter criterion.
	Delete criterion	Deletes the selected filter criterion.
	Move criterion up	Moves the selected filter criterion one line up.
	Move criterion down	Moves the selected filter criterion one line down.

Attributes, operators and values in the filter dialog

To create a valid filter you have to define the associated attributes, operators and values.

When you click in the column "Attribute" or "Operator", a drop-down list is displayed from which you can select the desired entries.

The "Value" column is displayed with a red background when you have selected an attribute and an operator, but have not yet entered a valid value.



Rules for criteria in the filter dialog

The rules specified below have to be observed when creating a new user-defined filter:

- Filter criteria can be defined for source objects, for reference objects and for references.
- Source objects are not affected when a criterion only applies to one reference object or one reference.
- The filter criteria are applied in the specified order.

  Note

  Note that the result of the first criterion is used as a specification for the second criterion, etc. Therefore the sequence of the criteria is also decisive.

  You can move the existing criteria "up" or "down" by using the corresponding symbols in the toolbar. To do so, select the criterion and then click the desired icon.

- Individual criteria can only be combined with "And".
- The permissible operators for the respective attribute can be selected from the drop-down list in the "Operator" column. The drop-down list is displayed as soon as you click the corresponding line in the filter dialog.
- If a filter is invalid because no valid value has been entered as a criterion yet, for example, the "Value" column is displayed with a red background. Such an invalid filter can be stored but cannot be applied. In this case, you can only exit the filter dialog by using "Close".
- When a criterion applies to a structured source object, this source object is displayed with all the parents and child objects and their references.
- The parent objects of the structured source object are, if they are not affected, only displayed because of the hierarchy and thus without references and in gray.

- When a criterion applies to a structured reference object, this reference object is displayed with all its lower-level reference locations.
- When a criterion only applies to elements of a structured reference object, only the respective lower-level reference location is displayed.

Requirement for the creation of a user-defined filter

You have created a project which contains objects with references.

Creating a user-defined filter

To create user-defined filters for the cross-reference list, follow these steps:

1. Select the program blocks folder and open the cross-reference list by using the "Cross-references" shortcut menu command.
2. Click the filter icon in the toolbar of the open cross-reference list.
   The filter dialog opens and shows the criteria that meet the selected system filter.
   Example: You have selected the system filter "Show objects with references" before opening the filter dialog in the drop-down list. You now see the Attributes, Operators and Values defined for this system filter in the filter dialog.
3. Click the "Create new filter" icon.
   The newly defined filter name "Filter_1" is displayed in the drop-down list. The icons for adding new filter criteria are activated.
4. Define the associated attributes, operators and values.
   Permissible attributes and operators can be selected from drop-down lists. The drop-down lists are displayed as soon as you click in the corresponding line in the filter dialog.
5. If you need several criteria, click the "Create new criterion" icon:

   Note

   Note that the result of the first criterion is used as a specification for the second criterion, etc. Therefore the sequence of the criteria is also decisive.

   You can move the existing criteria "up" or "down" by using the corresponding symbols in the toolbar. To do so, select the criterion and then click the desired icon.

6. Enter all criteria required for the new filter.
7. Assign a name for the new filter if you do not want to use the default name "Filter_1". You can enter the desired name directly in the drop-drop-down list.

   Note

   If you enter a name in the drop-down list that already exists as a filter name, this filter is applied in the cross-reference editor.

   The new filter is saved under the name suggested by the system (for example, "Filter_10") and can be selected via the drop-down list in the editor.

8. Click "Close" if you want to store the new filter settings, but do not want them displayed immediately in the lower-level cross-reference editor.
9. Click "Apply" to display the new filter settings immediately in all lower-level cross-reference editors.

Result

When the filter dialog is exited with "Close", the new filter is stored but not applied. The new filter is not used until you select it from the drop-down list in the cross-reference editor.

When the filter dialog is exited with "Apply", the new filter is stored and is applied in the open cross-reference editors.

11.18.6.3 Editing and deleting user-defined filters

Introduction

User-defined filters for the display of cross-references can be selected after they have been created via the drop-down list in the toolbar of the cross-reference editor.

In contrast to system filters, user-defined filters can be edited, renamed, and deleted.

The toolbar contains the icons for creating, editing and deleting filters and filter criteria.

Note

Renaming and deleting filters in the cross-references

Only user-defined filters can be renamed and deleted; system filters cannot be renamed and cannot be deleted either.

Requirement for editing a user-defined filter

You have created a project which contains objects with references.

At least one user-defined filter has been created for the cross-references.

For more details, see: Creating user-defined filters for cross-references (Page 5157)

Editing and renaming user-defined filters

To edit user-defined filters for the cross-reference list, follow these steps:

1. Select the program blocks folder and open the cross-reference list by using the "Cross-references" shortcut menu command.
2. Select the user-defined filter that you want to edit from the drop-down list.
3. Click the filter icon in the toolbar of the open cross-reference list.
   The filter dialog opens and shows the criteria that meet the selected filter.
4. Click the "Insert new criterion" icon in the toolbar.

5. Define the associated attributes, operators and values.
   Permissible attributes and operators can be selected from the drop-down lists. The drop-down lists are displayed as soon as you click in the corresponding line in the filter dialog.
6. Enter all criteria required for the new filter.

   Note

   Note that the result of the first criterion is used as a specification for the second criterion, etc. Therefore the sequence of the criteria is also decisive.

   You can move the existing criteria "up" or "down" by using the corresponding symbols in the toolbar. To do so, select the criterion and then click the desired icon.

7. Click "Delete criterion" if you want to delete a filter criterion.
8. If you wish, assign a name for the new filter. You can enter the desired name directly in the drop-drop-down list.

   Note

   If you enter a name in the drop-down list that already exists as a filter name, this filter is applied in the cross-reference editor.

   The new filter is saved under the name suggested by the system (for example, "Filter_10") and can be selected via the drop-down list in the editor.

9. Click "Close" if you want to store the new filter settings, but do not want them applied immediately in the lower-level cross-reference editor.
10. Click "Apply" to apply the new filter settings immediately in all lower-level cross-reference editors.

Result

When the filter dialog is exited with "Close", the modified filter is stored but not applied. The filter is not used until you click "Update" in the cross-reference editor.

When the filter dialog is exited with "Apply", the new filter is stored and is applied in the open cross-reference editors.

Deleting user-defined filters

To delete user-defined filters for the cross-reference list, follow these steps:

1. Select the program blocks folder and open the cross-reference list by using the "Cross-references" shortcut menu command.
2. Select the user-defined filter that you want to delete from the drop-down list.
3. Click the filter icon in the toolbar of the open cross-reference list.
   The filter dialog opens and shows the criteria that meet the selected filter.

   Note

   Keep in mind that the filter is deleted irrevocably without any further prompts as soon as the "Delete filter" icon is clicked.

4. Click the "Delete filter" icon in the toolbar if you want to delete the selected filter irrevocably.
5. Exit the dialog with "Close" or "Apply".

Result

The deleted user-defined filter is deleted irrevocably.

In the drop-down list either the subsequent filter or, if it does not exist, the preceding filter is displayed.

If you have deleted a filter that is applied actively, the view is not changed in the cross-reference editor until you click the "Update" button.

Invalid user-defined filters

If a user-defined filter becomes invalid through editing, for example, because a value has not been entered yet, the corresponding line in the filter dialog has a red background. This has the following effects:

- The filter cannot be used because the "Apply" button is not active.
- After the filter dialog has been closed, no changes are displayed in the editor.
- If you click "Update" in the cross-reference editor, the filter defined as the default setting is applied for the display of the cross-references.
- If no user-defined filter is defined as the default setting, the system filter "Show objects with references" is applied.

11.18.6.4 Exporting and importing user-defined filters

Introduction

After they have been created, user-defined filters for the display of cross-references can be exported and then imported again.

This has the advantage that you can also pass a filter on to other users when it has been created.

You use the functionality of the TIA Portal for the export and import of user-defined filters.

Requirement for exporting a user-defined filter

You have created a project which contains objects with references.

At least one user-defined filter has been created for the cross-references.

For more details, see: Creating user-defined filters for cross-references (Page 5157)

Exporting user-defined filters

To export user-defined filters, follow the steps below:

1. Select the "Settings" command in the "Options" menu.
   The settings of the TIA Portal are displayed.
2. Select the "General" entry in the area navigation.
3. In the "Import/export settings" area, click on the "Export settings ..." button.
   The "Export settings" dialog opens.
4. Click "Deselect all" to select only the desired filter settings for the export.
5. Select the desired filter for the export under "Cross-references > User-defined filters".
   If you want to accept the settings selected from previous export operations, activate the check box "Use selection from a previous export file". Enter the file in the field.
6. Enter a file name for the initialization file in the "File name" box.
7. Click "Export" to export the selected filters.

Result

Settings are exported to a file with the extension ".tps14".

Importing user-defined filters

To import user-defined filters, follow the steps below:

1. Select the "Settings" command in the "Options" menu.
   The settings of the TIA Portal are displayed.
2. Select the "General" entry in the area navigation.
3. Click the "Import settings ..." button in the "Import/export settings" area.
   The "Export settings" dialog opens.
4. Select the desired file with the extension ".tps14".

   Note

   If a filter with an identical name already exists, it is overwritten by the import and is no longer available.

5. Click "Open" to import the file.

Result

The selected file is imported and, after a restart of the TIA Portal, the user-defined settings are applied to the cross-references.

11.18.6.5 Setting a filter as the default setting

Introduction

You can use various system-defined filter settings (called system filters below), which can be selected via the drop-down list in the toolbar, to display the cross-references. In addition, you can also create user-defined filters in accordance with your own requirements.

You have the option of setting any filter as the default setting for the display of cross-references.

If a filter is set as the default setting, this filter is always applied when the cross-references are opened.

Rules for the default setting of filters

The following rules apply for the default setting of filters:

- Only one filter can be selected as the default setting.
- The default filter is identified with the corresponding icon.
- Invalid filters can also be set as the default setting, but they cannot be applied. Instead of the invalid filter, however, the system filter "Show objects with references" is applied automatically when the cross-reference list is opened.

Requirements

You have created a project which contains objects with references.

Setting a filter as the default setting

To set a filter as the default setting for the display of cross-references, follow these steps:

1. Select the program blocks folder and open the cross-reference list by using the "Cross-references" shortcut menu command.
2. Select the filter that you want to set as the default from the drop-down list.
3. Click the filter icon in the toolbar of the open cross-reference list.
   The filter dialog opens and shows the criteria that meet the selected filter.
4. Click the Apply filter as default setting icon in the toolbar.
5. Click "Close" if you want to store the new default setting, but do not want it applied immediately in the lower-level cross-reference editor.
6. Click "Apply" to apply the new filter settings immediately in all lower-level cross-reference editors.

Result

When the filter dialog is exited with "Close", the new default setting is stored but not applied. The new default setting is not used until you click "Update" in the cross-reference editor.

When the filter dialog is exited with "Apply", the new default setting is stored and is applied in the open cross-reference editors.

See also

Creating user-defined filters for cross-references (Page 5157)

11.18.6.6 Duplicating a filter

Introduction

You can use various system-defined filter settings (called system filters below), which can be selected via the drop-down list in the toolbar, to display the cross-references. In addition, you can also create user-defined filters in accordance with your own requirements.

You can duplicate an existing filter.

This facilitates the adding of further filter criteria.

Requirements

You have created a project which contains objects with references.

Duplicating a filter

Proceed as follows to duplicate a filter:

1. Select the program blocks folder and open the cross-reference list by using the "Cross-references" shortcut menu command.
2. Select the filter that you want to duplicate from the drop-down list.
3. Click the filter icon in the toolbar of the open cross-reference list.
   The filter dialog opens and shows the criteria that meet the selected filter.
4. On the toolbar, click the "Duplicate selected filter" icon.
   The filter is then duplicated and is displayed in the drop-down list under the new name "Filter_x".
5. Enter additional filter criteria as required or change the existing attributes, operators or values.

Result

When the filter dialog is exited, the duplicated filter is stored and is applied in the open cross-reference editors.

11.18.6.7 Filtering the cross-reference list

Introduction

You can use various system-defined filter settings (called system filters below), which can be selected via the drop-down list in the toolbar, to display the cross-references.

In addition to the filter options already offered, you can also create user-defined filters in accordance with your own requirements.

The system filters are always available in the drop-down list for filter selection and cannot be deleted.

User-defined filters are displayed after creation in the drop-down list for filter selection below the system filters and can be deleted.

See also: Creating user-defined filters for cross-references (Page 5157)

Requirements

You have created a project which contains objects with references.

Filtering the cross-reference list

To filter the cross-reference list, follow these steps:

1. Select the required source object and open the associated cross-reference list using the "Cross-references" shortcut menu command.
2. Alternatively, you can open the cross-references by selecting objects in the project tree for which cross-references exist, for example:
   - A device
   - The "Program blocks" folder
   - A block
3. Then click the "Cross-references" icon in the toolbar of the TIA Portal.
4. In the opened cross-reference list, select the required filter setting via the drop-down list.

Result

The selected filter setting is applied for the objects displayed in the lower-level cross-reference editors.

11.18.7 Printing the cross-reference list

Introduction

In the print preview, you can view and print out the project documentation contents shown in the cross-reference list.

The "Project" > "Print preview" and "Project" > "Print" commands are available for this purpose in the TIA Portal.

The following rules apply to printing out:

- The cross-reference list is always expanded and printed with all existing columns (including the hidden columns).
- The set filter criteria are applied at the same time.
- The column width is automatically optimized.
- Any overlapping accesses displayed are not printed.
- Regardless of which object is selected in the cross-reference list, the printout always starts with the displayed source object.

Note

Printing an empty cross-reference list

If the cross-reference list selected for printing contains no entries, an empty cross-reference list is printed out.

A cross-reference list which is printed out without contents contains the note "There are no entries in cross reference list".

Requirements

You have created a project which contains objects with references.

Printing the cross-reference list

To print the cross-reference list, follow these steps:

1. Select an object from the cross-reference list and start the print preview via the "Project" > "Print preview" command.
2. Make sure that you print out the required cross-reference list and select the required options.
3. Start the printout using the "Project" > "Print" command.
4. In the following dialog box, select the required print options and click "OK".

Result

The selected cross-reference list is printed out with the set print options.

11.18.8 Adding a new object to the cross-reference list

Introduction

Depending on the selected object, all cross-references belonging to this source object are displayed when the cross-reference list is opened.

You have the following options for inserting new source objects in an existing cross-reference list:

- By clicking the "Add new source object" button which is displayed at the end of the cross-reference list.

Requirement

You have created a project which contains objects with references.

Adding a new object to the cross-reference list

To add new objects to the cross-reference list, follow these steps:

1. Open an existing cross-reference list.
2. Click the "Add new source object" button and enter the required object. When an existing object is inserted, you are supported by the intelligent pre-selection.

Note

Multiple insertion of identical objects

If you insert an identical object multiple times in the cross-reference list, the duplicates are automatically removed after the next update.

Note

Functionality for newly added source objects

Note that the "Undo and redo operations" functionality is not available for newly added source objects.

Result

The newly added object is displayed as last object in the cross-reference list.

The current filter selection is applied to the newly added object only after the next update.

11.18.9 Show overlapping accesses in the cross-reference list

Introduction

The overlapping access to a selected object can be displayed in a second window in the lower part of the cross-reference list.

The display of the overlapping access cannot be saved and is no longer available after the cross-reference list is opened again.

Definition

Overlapping access for tags and structured tags is determined based on the absolute addresses.

We refer to "overlapping access" if the address for a tag or structured tag is partly or completely in the address space of the associated source object.

Note

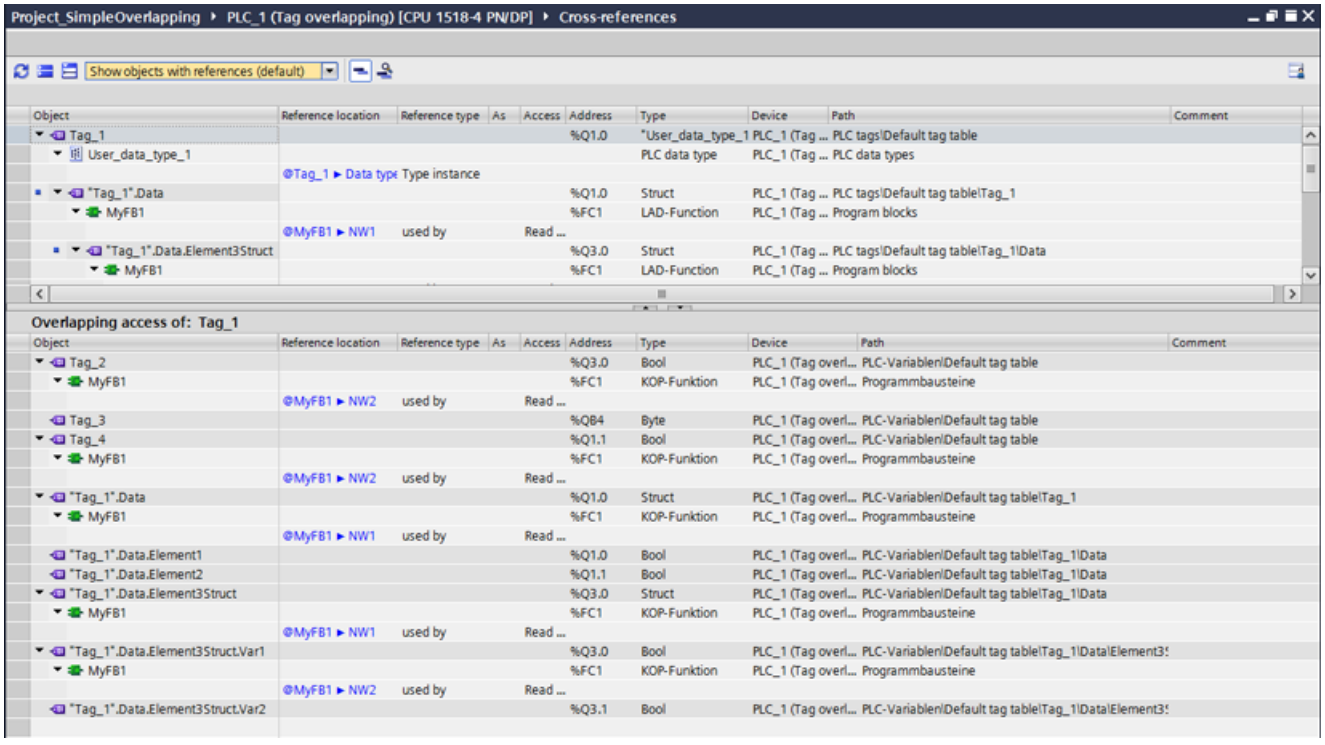
Displaying "overlapping access"

Overlapping access is only displayed for a single selected source object or lower-level source object in the cross-reference list.

If several objects are selected in the cross-references shown, no overlapping access is displayed.

Example

The following graphic shows the cross-reference list with the display of the overlapping access for "Tag_1":



Requirement

A project has been created which contains objects with overlapping access.

Showing overlapping access in the cross-reference list

Proceed as follows to display overlapping access in the cross-reference list:

1. Open the cross-reference list and click the "Show overlapping accesses" button in the toolbar.
   Result: All objects that have overlapping access are marked with the corresponding overlapping access symbol in the "Object" column.
2. In the upper part of the cross-reference list, select a marked object and call the "Show overlapping accesses" command via the shortcut menu.
   Result: The overlapping access to the selected object is displayed in a second window with the header "Overlapping accesses of: <ObjectName>".

Note

Display of the overlapping access

The display of the overlapping access cannot be saved and is no longer shown after the cross-reference list is opened again.

To show the overlapping access again, perform the steps described above.

11.18.10 Restoring cross-references after project upgrade.

Introduction

After a project upgrade to a higher TIA Portal version, you have the option of recreating the associated cross-reference information for the updated project.

This happens automatically at the end of the upgrade process. The current project is opened after this and the cross-references are then available again.

You are notified if the cross-reference information is inconsistent or incomplete. A banner text with a link to additional information is displayed when the cross-reference list is opened.

In this case, you can reorganize the cross-reference information yourself.

Requirements

You have performed a project upgrade and received the message that the cross-reference information is inconsistent or incomplete.

Recreate the cross-reference information

To create new cross-references, follow these steps:

1. To restore the cross-references, click the link in the displayed message.
   Alternatively, you can also call and execute the command via "Options" > "Settings" > "General" > "Cross-references".
   The settings for the TIA Portal then open under "Cross-references".
2. Click the displayed button "Recreate the cross-reference information" to start the process.
3. When the restoration has been completed, check the message in the Inspector window to see if the process was successful.
   If the process was successful, you receive a message which you must confirm with "OK".
   If the process was not successful, you receive an error message.
4. In the cross-reference list, click the "Update" button to close the displayed banner again.

Result

The new cross-reference information for the selected project was created.

11.18.11 Displaying and editing cross-references in the Inspector window

Introduction

The Inspector window displays cross-reference information about an object you have selected in the "Info > Cross-references" tab.

If you select an object in any editor in the TIA Portal, the associated reference to the selected object is shown in this tab.

Structure

The Inspector window displays the cross-reference information in tabular format.

The cross-reference list has the following structure:

Column	Content/meaning
Object	Displays the name of the source object which was selected when the cross-reference list was opened, with all lower-level objects and the associated reference objects.
Reference location	Shows the reference location of the objects, e.g. a network.

Column	Content/meaning
Reference type	Shows the relationship between the source object and the referenced objects: • "Uses": The source object uses this object. • "Used by": The source object is used by this object. • "Type -> Instance": The source object is a type of the referenced object. • "Instance -> Type": The source object is an instance of the referenced object. • "Group -> Element": The source object is a group of the referenced object. • "Element -> Group": The source object is an element of the referenced object. • "Defines": The source object defines the referenced object. • "Defined by": The source object is defined by the referenced object.
As	Shows additional information about objects, for example that a tag is used by several devices.
Access	Shows the type of access, e.g. whether access to the operand is read access (R) and/or write access (W).
Address	Shows the address of the relevant object.
Type	Shows the type and the language used to create the object.
Device	Shows the associated device name, e.g. "CPU_1".
Path	Displays the path of the object in the project tree with specification of folders and groups.
Comment	Displays comments on individual objects, if available.

Displaying cross-references in the Inspector window

To display cross-reference information in the Inspector window, follow these steps:

1. Select the required object (e.g. in the project tree or in the program editor) and click "Show cross reference list" in the shortcut menu.
   Result: The references to the selected object are displayed in the Inspector window in the "Info" > "Cross-references" tab.

Note

Please note that only the cross-reference information for supported objects can be shown in the Inspector window

If, for example, you have selected a folder, you will not receive any cross-reference information for this.

Editing cross-references in the Inspector window

You also have the option of editing and filtering cross-reference information in the Inspector window.

The procedure for filtering cross-reference information is identical to the procedure in the cross-reference editor.

See also:

Creating user-defined filters for cross-references (Page 5157)

Editing and deleting user-defined filters (Page 5161)

Setting a filter as the default setting (Page 5165)

Filtering the cross-reference list (Page 5166)

11.19 Testing the user program

11.19.1 Basics of testing the user program

Functions

You have the option of testing the running of your user program on the device. You can then monitor signal states and values of tags and can assign values to tags to simulate certain situations in the running of the program.

Requirement

There must be an executable program loaded on the device.

Test options

The following test options are available:

- Testing with program status
  The program status allows you to monitor the running of the program. You can display the values of operands and the results of logic operations (RLO) allowing you to recognize and fix logical errors in your program.
- Testing in single step mode (S7-300/400 only)
  You can test blocks you created in STL or SCL in the single step mode. You do this by setting breakpoints in the program code at which program execution stops. You can then continue to run the program one step at a time. Within a CPU, you can test either with program status or in single step mode. You cannot, however, use both test options at the same time within a CPU.
- Testing with the watch table
  With the watch table, you can monitor and modify the current values of individual tags in the user program or on a CPU. You can assign values to individual tags for testing and run the program in a variety of different situations. You can also assign fixed values to the I/O outputs of a CPU in STOP mode, for example to check the wiring.
- Testing with the force table
  With the force table, you can monitor and force the current values of individual tags in the user program or on a CPU. When you force, you overwrite individual tags with specified values. This allows you to test your user program and run through various situations. When forcing, make sure that you keep to the necessary safety measures for forcing (Page 5243)!

See also

Introduction to testing with program status (Page 5176)

Introduction to testing with the watch table (Page 5196)

Introduction for testing with the force table (Page 5224)

11.19.2 Testing with program status

11.19.2.1 Introduction to testing with program status

Program status function

You can monitor the execution of the program by displaying the program status. This shows you the values of the individual operands and their results of logic operations. This allows you to check whether the components of the automation system are being correctly controlled.

The display of the program execution in the program status can differ slightly, depending on the CPU series used.

Testing with program status for S7-300/400

During testing with program status, the CPU cycle time can become extended because the recording of the data affects the duration of the programmed instructions.

During execution of the following test functions, an alarm is displayed once for each online session indicating that there is a risk of a timeout:

- During testing with call conditions
- During testing with breakpoints

You can only perform these test functions after you have acknowledged the alarm.

Note

With older CPUs from the S7-300/400 CPU series, you will need to change the operating response using the hardware configuration and then load the hardware configuration to the device. You have the option to set the "Process operation" or "Test mode" operating response.

Testing with program status for S7-1200/1500

When you run the "Testing with program status" function with a CPU from the S7-1500 series, the cycle time can increase significantly when monitoring loops due to the fast program execution. The CPU may go to STOP, especially when it exceeds the configured cycle clock for interrupt OBs or the configured maximum cycle time.

Note

To avoid a possible STOP in the CPU, ensure that no programmed loops are displayed in the active monitoring window during "Testing with program status". Alternatively, you can also increase the maximum permissible cycle time of the CPU.

Restrictions with the "Program status" function

The monitoring of loops can increase the cycle time significantly, depending in each case on the number of tags to be monitored and on the actual number of loops processed.

To ensure that the cycle time is influenced as little as possible, the "Program status" function is restricted as described below:

For CPUs from the S7-300/400 series:

- The status display of a programmed loop is stopped at the return point.

For CPUs from the S7-1200/1500 series:

- The monitoring of loops is deactivated by default. If required, monitoring can be activated via the shortcut menu.
- If the monitoring of loops is enabled with an S7-1200 CPU, the program status for the area is not displayed after a loop that is processed multiple times. In this case, move the visible editor window to the area after the loop so that the monitor values are displayed again.

⚠ WARNING

Testing with program status

A test with the "Program status" function can cause serious damage to property or injury to persons if malfunctions or program errors occur.

Make sure that no dangerous situations can arise before you conduct a test with the "Program status" function.

Recommended procedure if there is a risk of a timeout

If there is a risk of a timeout during monitoring with program status, you are alerted by the system with a corresponding alarm.

To continue monitoring, you have to acknowledge the alarm with "Yes".

In this case, you also have the following option to avoid exceeding the maximum permissible cycle time during testing with program status:

- Reduce the size of the editor window so that fewer tags are displayed and thus fewer tags are monitored.
- Monitor the required tags within the watch table.

11.19.2.2 Setting the call environment

Basics of the call environment

Function

You can define the call environment for block monitoring and for breakpoints. With this, you specify the conditions for recording the program status of a block or for stopping program execution at a breakpoint.

Specification of the call environment for block monitoring

Alternatively, you can activate one of the following options to specify the call environment:

- No condition defined
  If no other option was selected this option is the default.
- Instance data block
  The program status of a function block will then only be recorded when you call the function block with the selected instance data block.
- Call path
  The program status of a block will then only be recorded when you call the block with a specific block or from a specific path.
- Manually adapted call path
  You can enter the desired call environment manually in this field. The "Transfer to 'adjusted manually'" button is used to transfer the content selected under "Call environment", which you can edit further if required.
  The program status of a block will then only be recorded when you call the block with a specific block or from a specific path.

Specification of the call environment for breakpoints

You can also specify a separate condition for each breakpoint.

If you do not specify the call environment, the program status of a block call is selected at random and recorded within the call structure and program execution is always interrupted at the relevant breakpoint. You should therefore always specify the call environment if you want to display the program status for a specific call.

See also

Setting the call environment for block monitoring (Page 5178)

Setting the call environment for block monitoring

By setting the call environment, you can specify when the program status of a block is recorded.

The section "Setting the call environment for breakpoints" describes how to set the call environment for breakpoints.

Requirement

- The block is open.

Specifying the call environment

To set the call environment, follow these steps:

1. Open the "Testing" task card.
2. Click "Change" in the "Call environment" pane.
   The "Call environment of the block" dialog opens.
3. Select the condition you want to apply.
   See also: Basics of the call environment (Page 5178)
4. Confirm your selection with "OK".

Result

The selected call environment is displayed in the "Testing" task card within the "Call environment" pane. The program status is now carried out in accordance with the set call environment.

Changing the call environment

Proceed as follows to change the call environment:

1. Open the "Testing" task card.
   If a call environment is already set, this is displayed in the "Call environment" pane.
2. Click "Change" in the "Call environment" pane.
   The "Call environment of the block" dialog opens.
3. Select the condition you want to apply.
   See also: Basics of the call environment (Page 5178)
4. Confirm your selection with "OK".

Result

The selected call environment is displayed in the "Testing" task card within the "Call environment" pane. The program status is now carried out in accordance with the set call environment.

See also

Basics of the call environment (Page 5178)

Introduction to testing with program status (Page 5176)

11.19.2.3 Switching test with program status on/off

You can monitor all blocks by switching on the program status of the block. This function is available to you for all code blocks, regardless of the programming language used. For blocks that were programmed with LAD, FBD or SCL you can also enable the program status from a specific position or for a specific selection. You can switch on the program status for an open block directly, or open a block from the calling block and view the program status.

A tooltip shows whether current data about the program status can be received or not. To do so, place the mouse in the upper right-hand corner of the working window above the dashed blue "Indicator".

Note

Note the following:

- The resources for testing with program status are limited. If there are not enough resources for an additional test, the tests that are already running are terminated.
- It is possible that another user can carry out a loading process on the selected CPU through joint parallel working on a CPU. It is therefore possible in the following cases that you can either not start the test with program status or that a running test is terminated:
  - Through the loading process the block for which you want to start or already have carried out the test with program status is loaded.
  - You use an instance data block as the call environment for the test with program status and the block changes structurally through the loading process, e.g. by renumbering.
  - You use a call path as the test condition for the test with program status and a block within the call path changes through the loading process.

  If a running test is terminated, a corresponding message is displayed in the Inspector window.

Requirement

- Code block: The code of the offline block is identical with the code of the online block. In this case the "Code time stamps" of the blocks are identical.
- Data block: The structure of the offline block is identical with the structure of the online block. In this case the "Interface time stamps" of the blocks are identical.

Switching the program status on or off directly in the block

To switch the program status for a block on or off directly in the block, follow these steps:

1. Open the block for which you wish to switch on the program status.
2. Click "Monitoring on/off" in the toolbar.
   If you have not already established an online connection, the "Go online" dialog opens. In this dialog, you can establish an online connection.
   See also: Establishing and terminating an online connection
3. Click "Monitoring on/off" in the toolbar again to switch the program status off.

Switching on or off program status starting at a specific point in a network

To start the program status for LAD and FBD at a specific point, follow these steps:

1. Open the block for which you wish to switch on the program status.
2. Click "Monitoring on/off" in the toolbar.
3. Right-click on the tag you want program status to start from.
4. Select "Modify > "Monitor from here" in the shortcut menu.
5. Click "Monitoring on/off" in the toolbar again to switch the program status off.

To start the program status in SCL at a specific point, follow these steps:

1. Open the block for which you wish to switch on the program status.
2. Click "Monitoring on/off" in the toolbar.
3. Select the tag from which you want to start the program status.
4. After this, click "Display program status from the selected line" in the toolbar.
5. Click "Monitoring on/off" in the toolbar again to switch the program status off.
6. To make matters clearer, you can use the shortcut menu commands "Open" and "Close" or "Open all" and "Close all" to expand or collapse individual or all closed code sections within the block.

Switching on or off program status for selected tags

To start the program status for LAD and FBD for selected tags, follow these steps:

1. Open the block for which you wish to switch on the program status.
2. Click "Monitoring on/off" in the toolbar.
3. Select the tags for which you want to start the program status.
4. Select "Modify > Monitor selection" in the shortcut menu.
5. Click "Monitoring on/off" in the toolbar again to switch the program status off.

Switching on program status from the calling block

To switch on the program status for a block from the calling block (e.g. OB1), follow these steps:

1. Open the calling block.
2. Right-click on the block call.
3. Select the command "Open and monitor" in the shortcut menu.
   The block will open in the program editor. An online connection is established and the program status is displayed. The calling block is set in the call path.

Result

If you enable the display of the program status, an online connection is established and the program status is displayed. When you turn off the display of the program status, you can terminate the online connection at the same time.

The call path of the block is shown under the block interface. If necessary, you can change the call environment in the "Call environment" section at the right-hand edge at "Test" and "Options". In the case of CPUs of the S7-1200/1500 series the "Call hierarchy" is displayed additionally on the right-hand edge. You can open the calling block by clicking the link.

11.19.2.4 Monitoring loops

Introduction to monitoring loops

If you display the program status, you can monitor the execution of the program with a block. This provides you with an overview of the values of the individual operands and the results of the logic operations and you can check whether the components of the automation system are correctly controlled.

If the values are shown in a lighter, less saturated font, these values do not come from the current cycle.

Note

The monitoring of loops can increase the cycle time of the CPU significantly, depending in each case on the number of tags to be monitored and on the actual number of loops processed.

Special rules apply to loop monitoring, depending on the specific CPU and the programming language that are used.

Rules for loop monitoring (S7-300/400)

To ensure that the cycle time is influenced as little as possible, the "Program status" function is restricted as follows during the monitoring of loops with CPU from the S7-300/400 family in all programming languages:

- The status display of a programmed loop is stopped at the return point.

Rules for loop monitoring in STL (S7-1500)

The display of the program status within programmed loops is disabled by default during the monitoring of loops in STL. To avoid any lengthening of the cycle time, no values are displayed for the tags within loops between the "Label" and the associated "Loop" instruction.

If you are monitoring an STL block with programmed loops, a tooltip informs you that the monitoring of loops is currently disabled and can be enabled if required via the shortcut menu.

Note

Please note that only those loops that are programmed within a network are recognized.

Enabling/disabling the monitoring of loops in STL (S7-1500)

To enable the monitoring of loops in STL, follow these steps:

1. Open the STL block that you want to monitor.
2. Click "Monitoring on/off" in the toolbar.
   If you have not already established an online connection, the "Go online" dialog opens. In this dialog, you can establish an online connection.
   See also: Establishing and terminating an online connection
3. Enable the monitoring within the program editor with the shortcut menu command "Monitoring" > "Monitor loops".
4. Confirm the next prompt with "Yes".
   Result: The monitoring of loops is enabled for the entire block and is active until monitoring of loops is once again disabled or the online connection to the CPU is terminated.
5. Click "Monitoring on/off" in the toolbar again to switch the program status off.

Rules for loop monitoring in SCL (S7-1200/1500)

The display of the program status within programmed loops is disabled by default during the monitoring of loops in SCL.

This affects the instructions "FOR", "WHILE" and REPEAT-UNTIL": To avoid any lengthening of the cycle time, no values are displayed for the tags within these instructions.

If you are monitoring an SCL block with programmed loops, a tooltip informs you that the monitoring of loops is currently disabled and can be enabled if required via the shortcut menu.

Enabling/disabling the monitoring of loops in SCL (S7-1200/1500)

To enable the monitoring of loops in SCL, follow these steps:

1. Open the SCL block that you want to monitor.
2. Click "Monitoring on/off" in the toolbar.
   If you have not already established an online connection, the "Go online" dialog opens. In this dialog, you can establish an online connection.
   See also: Establishing and terminating an online connection
3. Right-click within the loop, in the column "value, for example, and enable monitoring with the shortcut menu command "Monitor loops".
4. Confirm the next prompt with "Yes".
   Result: The monitoring of loops is enabled for the entire block and is active until monitoring of loops is once again disabled or the online connection to the CPU is terminated.
5. Click "Monitoring on/off" in the toolbar again to switch the program status off.

11.19.2.5 Monitoring structures

Introduction to monitoring structures

You can monitor the execution of the program with a block. This provides you with an overview of the values of the individual operands and the results of the logic operations and you can check whether the components of the automation system are correctly controlled.

As of TIA Portal V14, it is also possible to monitor structured PLC tags.

Special rules apply to monitoring structures, depending on the specific CPU that is used, the programming language that is used, and the properties of the relevant structures.

Rules for monitoring structures (S7-1200/1500)

During the monitoring of structures, the values of a structured PLC tag are displayed in the program editor and in the Inspector window, with the following exception:

- It is not possible to monitor structures whose elements have adjustable retentivity properties.

To display the monitor values for a structure, you must first enable monitoring via the shortcut menu.

Displaying the monitor values for structures in the Inspector window

If the monitoring of structures is enabled and a structure tag is selected in the program editor, the associated monitor values are displayed in the Inspector window in the tab "Diagnostics" > "Monitor values".

The Inspector window is structured as follows:

- The "Name" column shows the selected structure tag with the lower-level elements on the first step open. If you click the name of the structure tag, the program editor jumps to this tag's point of use.
- The "Data type" column shows the associated data type.
- The "Value" columns shows the associated monitor values. If dashes are shown here instead of values, the monitoring of loops is disabled. In this case, enable the monitoring of loops again by means of the shortcut menu in the program editor.
- Columns "Value - IN" and "Value - OUT": These columns with the corresponding values are only displayed if the selected structure is used as InOut parameter.
- If the monitor values are displayed in a font with only 50 % saturation, this means that these vales are not from the current cycle.
- The monitor values displayed in the Inspector window use the same call path that is set as default for the block.
- The table with monitor values displayed in the Inspector window can be copied to an Excel file.
- If it is not possible to display any monitor values in the Inspector window, you will receive information that these are not available.

Procedure for monitoring structures

To enable the monitoring of structures, follow these steps:

1. Open the block in the program editor.
2. Click "Monitoring on/off" in the toolbar.
   If you have not already established an online connection, the "Go online" dialog opens. In this dialog, you can establish an online connection.
   See also: Establishing and terminating an online connection
3. If the structure to be monitored is located in a loop, first enable the monitoring of loops.
   See also: Monitoring loops (Page 5182)
4. Right-click on a structure and enable the monitoring of structures by means of the shortcut menu command "Monitoring" > "Monitor values".
   Result: The selected structure tag is displayed in the Inspector window in the tab "Diagnostics" > "Monitor values" is open up to the first level with the current values displayed.
5. If required, open the still closed elements of the structure, to display all lower-level elements and their values.
6. Click "Monitoring on/off" in the toolbar again to switch the program status off.

Note

Monitoring a structure within an array

Click directly behind the closing bracket to select the relevant structure for monitoring.

See also

Monitoring loops (Page 5182)

11.19.2.6 Editing blocks during the program test

If you edit blocks while they are being monitored, monitoring will be interrupted and you will be able to edit the block offline. If the block is not available offline in the project, you will first have to load it from the device to the project. After editing the block, you will also have to compile and download it again.

Procedure

To edit blocks while these are being monitored, follow these steps:

1. Edit the block as necessary.
   Monitoring is interrupted and the block is switched offline mode assuming it exists offline.
2. If the block does not exist offline, load it to the project from the device.

3. Compile the block.
   See also: AUTOHOTSPOT
4. Download the block to the device.
   See also: AUTOHOTSPOT

Result

The block now contains your modifications both online and offline. The online connection is re-established and monitoring continues.

11.19.2.7 Modifying tags during monitoring

During monitoring, you have the option of modifying tags immediately and once in the program status.

For Boolean values you can switch between the values "0" and "1".

You can enter a modify value for tags that do not belong to the "BOOL" data type.

Note that you cannot modify peripheral inputs, for example, via TagName:P.

You can control tags during monitoring either by double-clicking the online value (not for data type BOOL) or after the tags have been clicked by using the shortcut menu.

Procedure

To modify tags during monitoring, follow these steps:

1. Open the desired block and start the monitoring of tags by clicking "Monitoring on/off" in the toolbar.
2. Double-click the online value (not for data type BOOL) that you wish to modify.
   Alternatively, you can click the desired tag and start modifying by using the shortcut menu.
3. In the subsequent dialog enter the value you require in the "Modify value" box and confirm with "OK".
   Or confirm the query in the subsequent dialog whether the Boolean online value is to be switched over.
   Example:
   - If the Boolean tag has the value "0", the value is switched over to the value "1" if the query has been confirmed with "Yes".
   - If the Boolean tag has the value "1", the value is switched over to the value "0" if the query has been confirmed with "Yes".
   - If you click "No" or exit the dialog by using " Close", the existing value remains.
4. Alternatively, select one of the following commands in the shortcut menu:
   - "Modify > Modify to 1"
     Modifies tags of the "BOOL" data type to the value "True".
   - "Modify > Modify to 0"
     Modifies tags of the "BOOL" data type to the value "False".
   - "Modify > Modify operand"

5. If you select "Modify operand", the "Modify operand" dialog opens.
6. In the subsequent dialog enter the value you require in the "Modify value" box and confirm with "OK".

See also

Switching test with program status on/off (Page 5180)

11.19.2.8 Switching display formats in the program status

Introduction

The display formats for tags depend on the respective tag type. During monitoring, you have the option of switching the current display format by means of the shortcut menu. The possible display formats for a tag are offered in the shortcut menu.

You can modify the display format for the selected tag, for all tags in the network or for all tags in the block.

This is useful, for example, when you need a hexadecimal display in order to search for a hexadecimal error code.

Procedure

To switch the display format, follow these steps:

1. Open the desired block in the programming editor.
2. Switch on the program status by clicking "Monitoring on/off" in the toolbar.
   If you have not already established an online connection, the "Go online" dialog opens. In this dialog, you can establish an online connection.
3. Select the tag you want to monitor and select the desired display format in the shortcut menu under "Display format".

Result

The display format is modified accordingly.

Note

Switching display format during monitoring

Please note that it is not possible to switch the display format for unconnected outputs, as no monitoring value is output in this case.

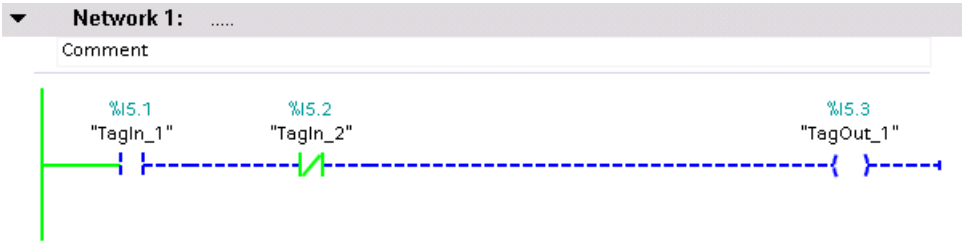
11.19.2.9 Examples of program status display

Program status display for LAD programs

Displays in program status

The display of the program status is updated cyclically.

The following figure shows an example of the program status display for LAD:



Representation of the program status

You can recognize the status of individual instructions and lines of a network quickly based on the color and type of lines and symbols. The following table shows the relationship between representation and status:

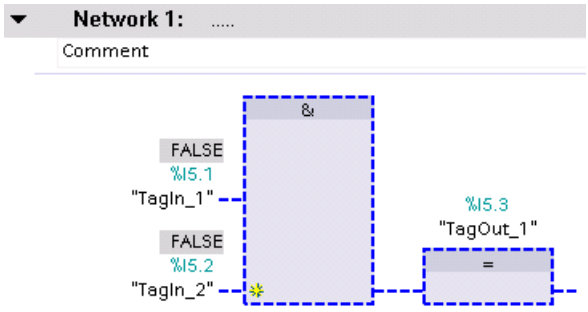
Representation	Status
Green solid	Satisfied
Blue dashed	Not satisfied
Gray solid	Unknown or not executed
Black	Not interconnected
Lines or parameters in a frame with a saturation of 100 %	Value is current
Lines or parameters in a frame with a saturation of 50 %	Value originates from an earlier cycle. The point in the program was not executed in the current cycle.

Program status display for FBD programs

Displays in program status

The display of the program status is updated cyclically.

The following figure shows an example of the program status display for FBD:



Representation of the program status

You can recognize the status of individual instructions and lines of a network quickly based on the color and type of lines and symbols. The following table shows the relationship between representation and status:

Representation	Status
Green solid	Satisfied
Blue dashed	Not satisfied
Gray solid	Unknown or not executed
Black	Not interconnected
Lines or parameters in a frame with a saturation of 100 %	Value is current
Lines or parameters in a frame with a saturation of 50 %	Value originates from an earlier cycle. The point in the program was not executed in the current cycle.

The values of the operands are displayed above the relevant operand name in a gray box.

Note

Program status display for outputs which are not interconnected

Please note that a monitor value cannot be displayed for outputs which are not interconnected.

Program status display for STL programs (S7-300, S7-400, S7-1500)

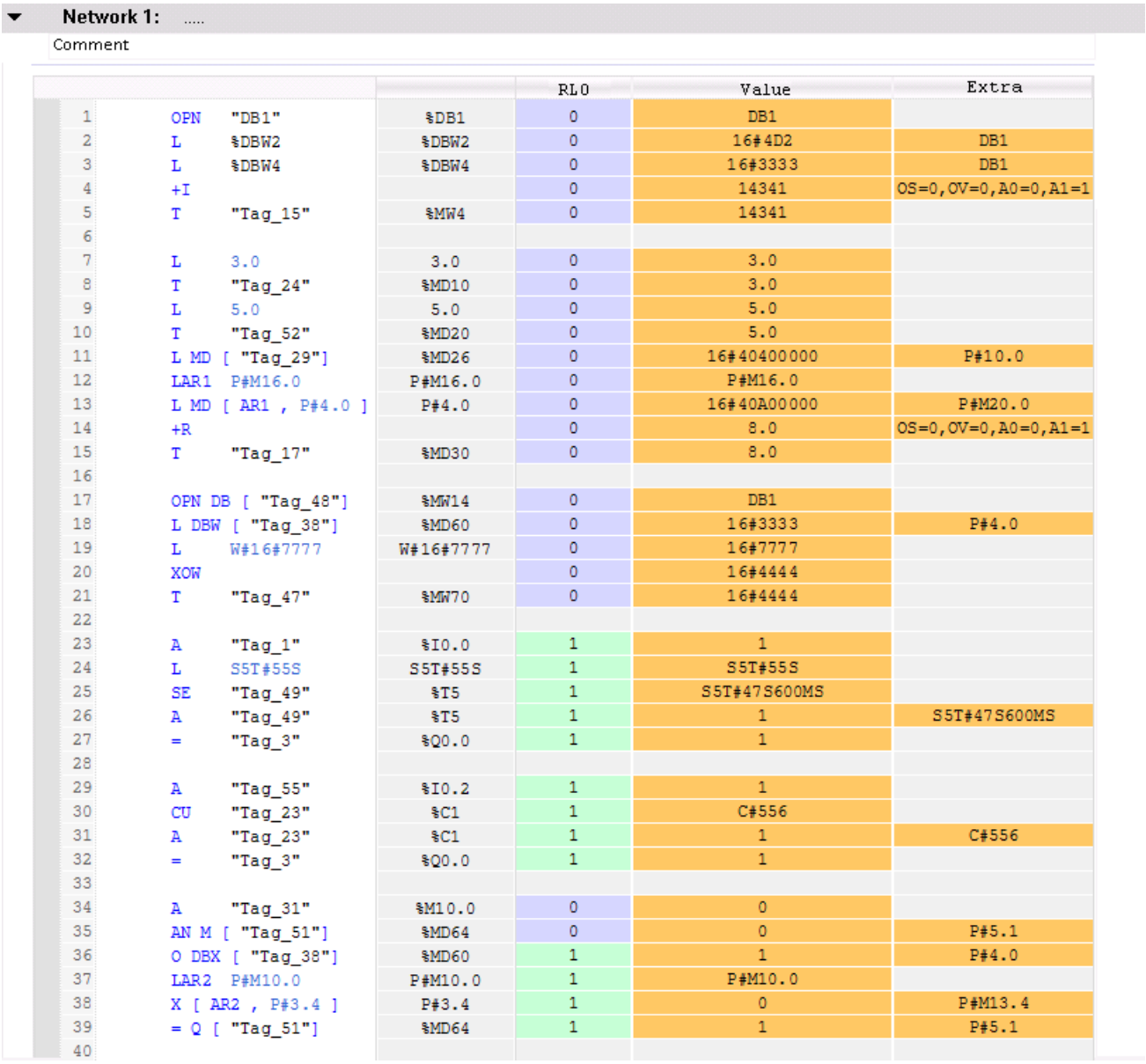
Displays in program status

The display of the program status is updated cyclically and shown in tables. The tables are shown directly next to the STL program. You can read the program status for each line of the program. The display depends on the CPU in use (S7-300, S7-400, or S7-1500).

The tables to be displayed in the program status always contain the following information:

- RLO
  The "RLO" column shows the result of logic operation for each line of program. You can recognize the value of the RLO based on the background color of the table cell. Here, green means an RLO of 1 and lilac an RLO of 0.
- Value
  The current value of the operand is shown in the "Value" column.
- Extra
  The "Extra" column shows additional information depending on the particular operation, for example, relevant status bits for mathematical instructions, time or count values for timers and counters or memory addresses for indirect addressing.

The following figure shows an example of the program status display of a CPU S7-300 under STL:



Network 1: .....

Comment

			RLO	Value	Extra
1	OPN "DB1"	%DB1	0	DB1	
2	L %DBW2	%DBW2	0	16#4D2	DB1
3	L %DBW4	%DBW4	0	16#3333	DB1
4	+I		0	14341	OS=0,OV=0,A0=0,A1=1
5	T "Tag_15"	%MW4	0	14341	
6					
7	L 3.0	3.0	0	3.0	
8	T "Tag_24"	%MD10	0	3.0	
9	L 5.0	5.0	0	5.0	
10	T "Tag_52"	%MD20	0	5.0	
11	L MD ["Tag_29"]	%MD26	0	16#40400000	P#10.0
12	LAR1 P#M16.0	P#M16.0	0	P#M16.0	
13	L MD [AR1 , P#4.0]	P#4.0	0	16#40A00000	P#M20.0
14	+R		0	8.0	OS=0,OV=0,A0=0,A1=1
15	T "Tag_17"	%MD30	0	8.0	
16					
17	OPN DB ["Tag_48"]	%MW14	0	DB1	
18	L DBW ["Tag_38"]	%MD60	0	16#3333	P#4.0
19	L W#16#7777	W#16#7777	0	16#7777	
20	XOW		0	16#4444	
21	T "Tag_47"	%MW70	0	16#4444	
22					
23	A "Tag_1"	%I0.0	1	1	
24	L S5T#55S	S5T#55S	1	S5T#55S	
25	SE "Tag_49"	%T5	1	S5T#47S600MS	
26	A "Tag_49"	%T5	1	1	S5T#47S600MS
27	= "Tag_3"	%Q0.0	1	1	
28					
29	A "Tag_55"	%I0.2	1	1	
30	CU "Tag_23"	%C1	1	C#556	
31	A "Tag_23"	%C1	1	1	C#556
32	= "Tag_3"	%Q0.0	1	1	
33					
34	A "Tag_31"	%M10.0	0	0	
35	AN M ["Tag_51"]	%MD64	0	0	P#5.1
36	O DBX ["Tag_38"]	%MD60	1	1	P#4.0
37	LAR2 P#M10.0	P#M10.0	1	P#M10.0	
38	X [AR2 , P#3.4]	P#3.4	1	0	P#M13.4
39	= Q ["Tag_51"]	%MD64	1	1	P#5.1
40					

Program status display for SCL programs

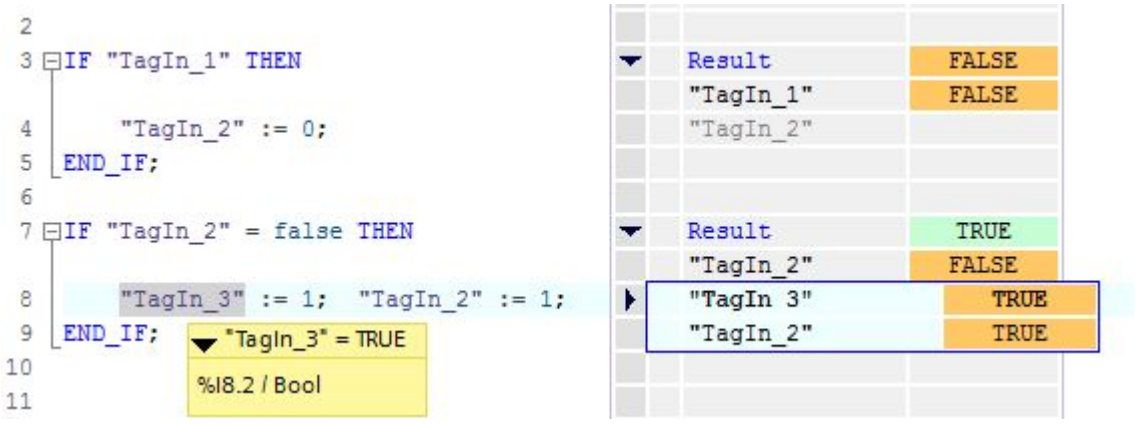
Displays in program status

The display of the program status is updated cyclically and shown in a table. The table is displayed immediately beside the SCL program and you can see the program status for each line of the program. The table contains the following information:

- Tag names
- Value

You can move the table to the left or right at any time.

The following figure shows an example of the program status display for SCL:



Explanation on monitoring in SCL

- If the code of the line is not executed, the tag name is displayed in the values table in gray text.
- The current values of the tags are displayed in the last column. If no values can be displayed for a tag, the corresponding cell has a yellow background and three question marks are shown. In this case, select the "Create extended status information" check box in the properties of the block and download the block to the device again. All values are then displayed.
- In the first column, you can see the name of the tag for which the current value is being displayed.
- If the line includes the "IF", "WHILE" or "REPEAT" instruction, the result of the instruction is displayed in the line as "True" or "False".
- If the line contains more than one tag, the value of the first tag is displayed. In both cases, all tags of these lines are displayed with their values in a separate list as soon as you select a line.
- If you place the cursor in a tag in the program code, this is shown in bold face in the list. You can also display the other tags of a line explicitly by clicking the arrow right located in front of lines containing more than one tag.

Program status display for GRAPH programs (S7-300, S7-400, S7-1500)

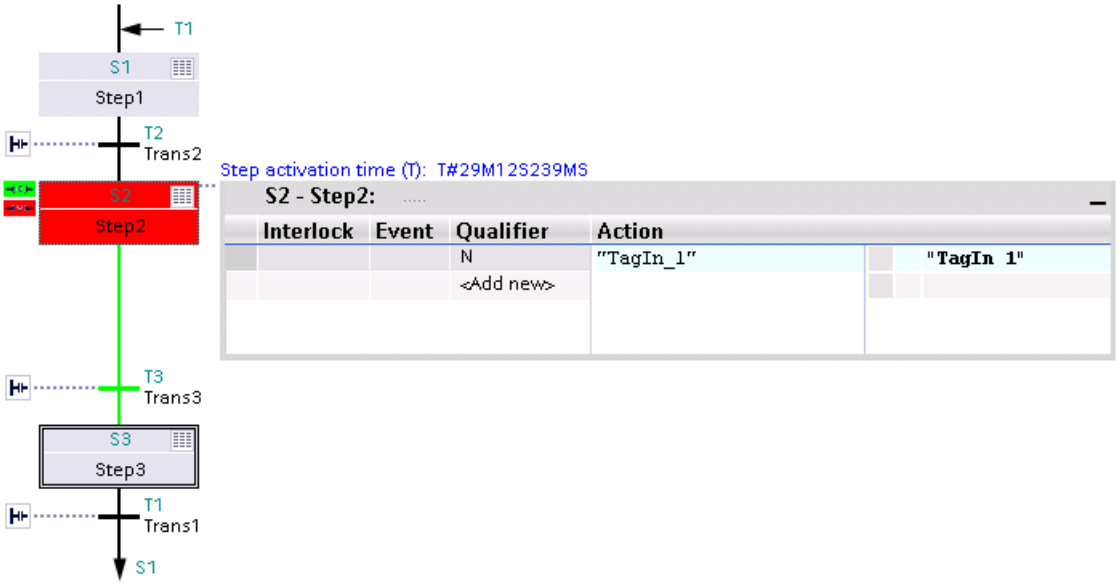
Displays in program status

For GRAPH programs you can have the program status displayed in sequence view and in single step view and for the permanent instructions. The program status display of permanent instructions corresponds to the program status display for LAB/FBD programs. The display of the program status is updated cyclically.

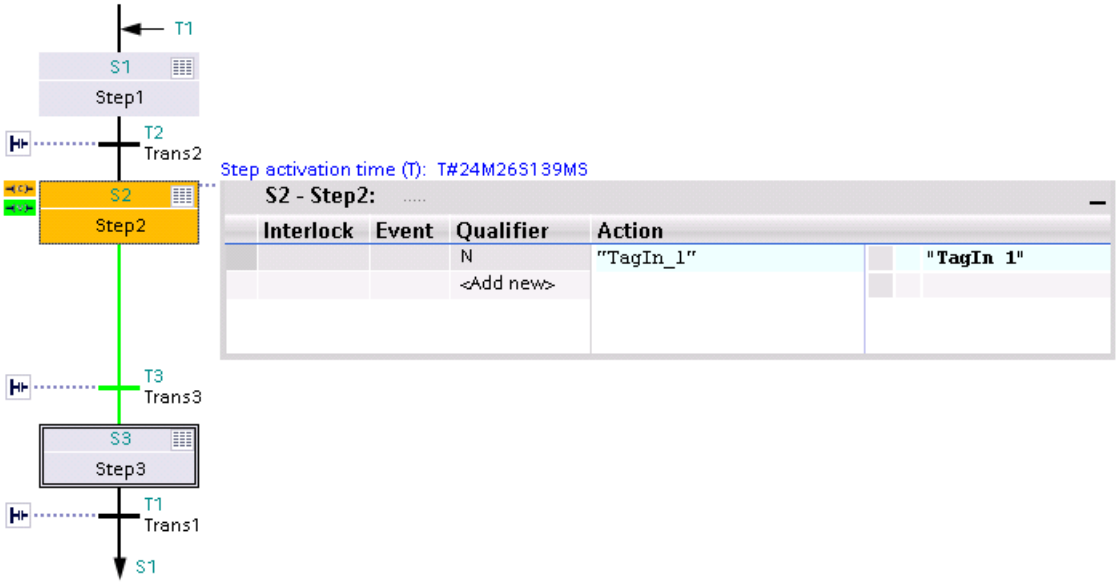
The following table shows the relationship between representation and status:

Representation	Area	Status
Green	Step, sequencer	There are no faults.
Light green	Step, sequencer	There are no faults. The values are from elapsed cycles.
	Condition	The transition is fulfilled.
Red	Step, sequencer	There is a supervision error.
Yellow	Step, sequencer	There is an interlock error.
Black	Condition	The transition is not fulfilled.

The following figures show examples of the display for the program status in the sequence view:

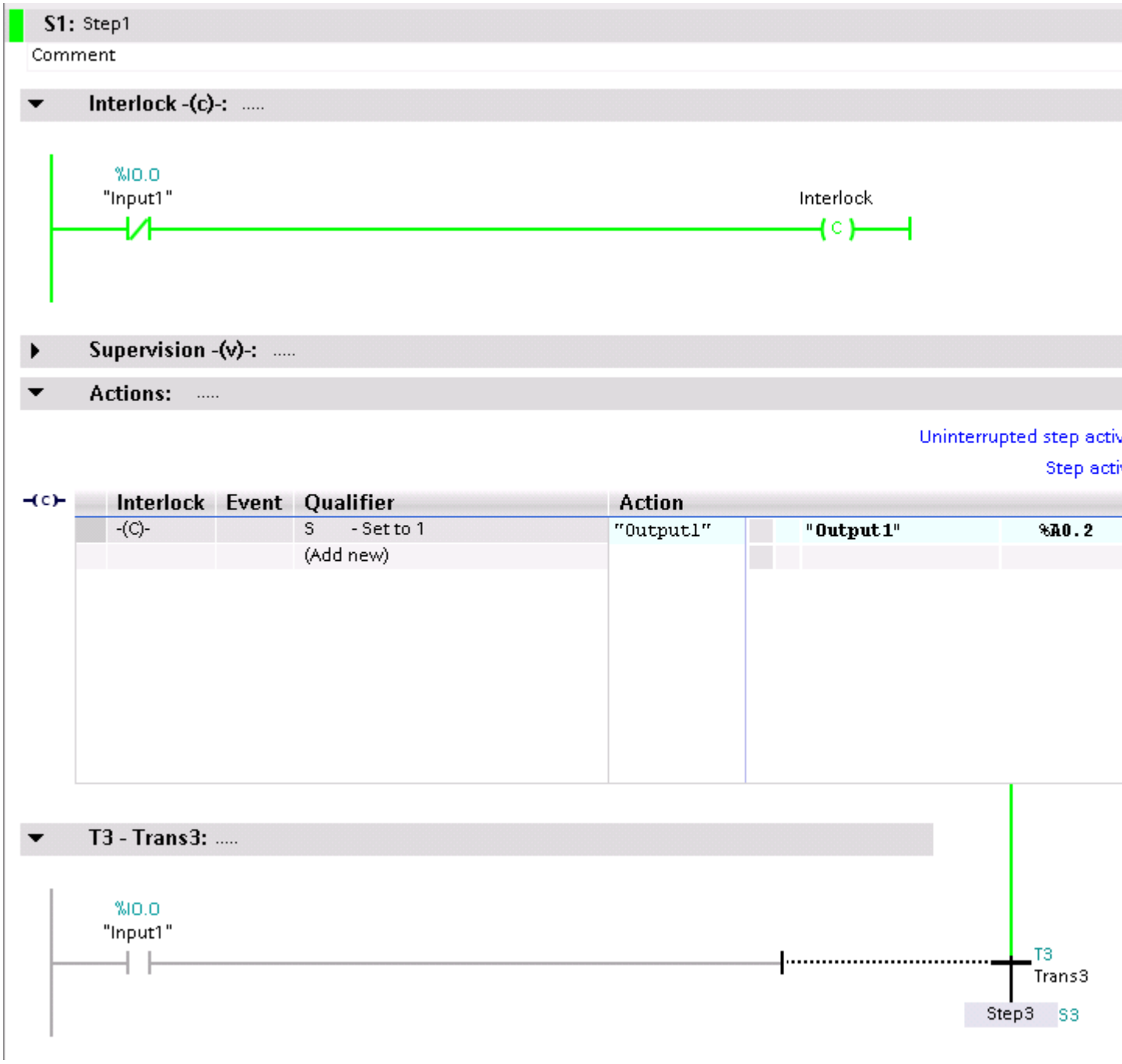


Step 2 contains a supervision error. The transition for switching from step 2 to step 3 is fulfilled.



Step 2 contains an interlock error. The transition for switching to step 3 is fulfilled.

The figure below shows an example of the display for the program status in the single step view of a S7-300:



The step does not contain an interlock error. The subsequent transition is not satisfied.

11.19.3 Testing with the watch table

11.19.3.1 Introduction to testing with the watch table

Overview

The following functions are available in the watch table:

- **Monitoring tags**
  This allows the current values of the individual tags of a user program or a CPU to be displayed on the programming device or PC.
- **Monitoring tags**
  You can use this function to assign fixed values to the individual tags of a user program or CPU. Modifying is also possible with Test with program status .
- **"Enable peripheral outputs" and "Modify now"**
  These two functions enable you to assign fixed values to individual peripheral outputs of a CPU in the STOP mode. You can also use them to check your wiring.

Monitoring and modifying tags

The following tags can be monitored and modified:

- Inputs, outputs, and bit memories
- Contents of data blocks
- Content of UDTs
- I/O

Team engineering in the watch table

As of TIA Portal V13 SP1, within the framework of team engineering with an S7-1500 CPU with firmware version >= 1.7, several Engineering Systems can access the CPU online simultaneously, for example to monitor and modify tags and download blocks. If you use this function, make sure that you keep to the requirements and rules that apply to team engineering as explained in the information system in "Using Team Engineering" in the section "Shared commissioning of projects".

Possible applications

The advantage of the watch table is that a variety of test environments can be stored. This enables you to reproduce tests during commissioning or for service and maintenance purposes.

See also

Layout of the watch table (Page 5197)

Basic mode and expanded mode in the watch table (Page 5198)

Icons in the watch table (Page 5199)

Creating and editing watch tables (Page 5200)

11.19.3.2 Layout of the watch table

Introduction

A watch table contains the tags you defined for the entire CPU. A "Watch and force tables" folder is automatically generated for each CPU created in the project. You create a new watch table in this folder by selecting the "Add new Watch table" command.

Layout of the watch table

The columns displayed in the watch table depend on the mode you are working in: basic mode or expanded mode.

The following additional columns are shown in expanded mode:

- Monitor with trigger
- Modify with trigger

The names of the columns can also be changed dynamically based on the action.

Meaning of the columns

The following table shows the meaning of the individual columns in basic mode and expanded mode:

Mode	Column	Meaning
Basic mode		Identifier column
	Name	Name of the inserted tag
	Address	Address of the inserted tag
	Display format	Selected display format
	Monitor value	Values of the tags, depending on the selected display format.
	Modify value	Value with which the tag is modified.
		Select the tag to be modified by clicking the corresponding check box.
	Comment	Option to enter a comment to document the tags in the watch table.
	Tag comment	Display of the comment on the selected tag that was entered in other editors. This column cannot be edited.
The following additional columns are shown in expanded mode:	Monitor with trigger	Display of selected monitoring mode
	Modify with trigger	Display of selected modify mode

See also

Icons in the watch table (Page 5199)

11.19.3.3 Basic mode and expanded mode in the watch table

Difference between basic mode and expanded mode in the watch table

Depending on the mode specified, the watch table displays different columns and column headings that can be used to perform different actions.

You will find a detailed list of the columns in Layout of the watch table (Page 5197).

Switching between basic mode and expanded mode

You have the following options of toggling between the basic and expanded mode:

- Click the icon "Show/hide advanced setting columns". Click this icon again to return to the basic mode.
  Or:
- In the "Online" menu, select the "Expanded mode" check box. Deactivate this check box to return to the basic mode.

Functionality in expanded mode

The following functionality is only possible in expanded mode:

- Monitor with trigger
- Modify with trigger
- Enable peripheral outputs
- Monitor peripheral inputs
- Modify peripheral outputs

NOTICE
Danger of a time-out while monitoring peripheral inputs and controlling peripheral outputs Note that the monitoring of peripheral inputs and the controlling of peripheral outputs in the watch table can result in a time-out. The CPU assumes the "STOP" mode.

See also

Setting the monitoring and modify mode (Page 5212)

11.19.3.4 Icons in the watch table

Meaning of the icons

The following table shows the meaning of the icons in the watch table:

Icon	Meaning
	Identifies a table inside the project tree as a watch table.
	Shows information in the identifier column.
	Inserts a row before the selected row.
	Inserts a row after the selected row.
	Inserts a comment line above the selected row.
	Modifies the addresses of all selected tags immediately and once. This command is executed once and as quickly as possible without reference to a defined trigger point in the user program.
	Modifies the addresses of all selected tags with reference to a defined trigger point in the user program.
	Disables the command output disable of the peripheral outputs. You can then modify the peripheral outputs when the CPU is in STOP mode.
	Displays all columns of expanded mode. If you click this icon again, the columns of expanded mode will be hidden.
	Displays all modify columns. If you click this icon again, the modify columns will be hidden.
	Starts monitoring of the visible tags in the active watch table. The default setting for the monitoring mode in basic mode is "permanent". In expanded mode, you can set defined trigger points for the monitoring of tags.
	Starts monitoring of the visible tags in the active watch table. This command is executed immediately and the tags are monitored once.
	Displays the check box for the selection of tags to be modified.
	Indicates that the value of the selected tag has been modified to "1".
	Indicates that the value of the selected tag has been modified to "0".
	Indicates that the address is being used multiple times.
	Indicates that the substitute value is being used. Substitute values are values that are output to the process in case of signal output module faults or are used instead of a process value in the user program in case of signal input module faults. The substitute values can be assigned by the user (e.g., retain old value).
	Indicates that the address is blocked because it is already being modified.
	Indicates that the address cannot be modified.
	Indicates that the address cannot be monitored.
	Indicates that an address is being forced.

Icon	Meaning
	Indicates that an address is being partly forced.
	Indicates that an associated I/O address is being fully or partly forced.
	Indicates that an address cannot be fully forced. Example: It is indeed possible to force the address QW0:P, but it is not possible to force the address QD0:P since this address area is eventually not available on the CPU.
	Indicates that a syntax error occurred.
	Indicates that the address is selected but at the moment e.g. has not yet been modified.

See also

Layout of the watch table (Page 5197)

11.19.3.5 Creating and editing watch tables

Creating a watch table

Introduction

The watch table allows you to monitor and modify tags in the user program. Once you have created a watch table, you can save it, duplicate it, and print it and use it again and again to monitor and modify tags.

Requirement

A project is open.

Procedure

To create a watch table, follow these steps:

1. Click "Project view" in the status bar.
   The project view is displayed.
2. In the project tree, double-click the CPU for which you want to create a watch table.
3. Double-click the "Watch and force tables" folder and then the "Add new watch table" command.
   A new watch table is added.
4. In the "Name" column or in the "Address" column, enter the name or the absolute address for the tags that you want to monitor or modify.

5. You can select a display format from the drop-down list in the "Display format" column if you want to change this default setting.
6. Now decide whether you want to monitor or modify the entered tags and, if applicable, enter the desired values for modifying.

Opening a watch table

Requirement

A watch table has been created.

Procedure

To open a watch table, follow these steps:

1. Open the "Watch and force tables" folder below the desired CPU.
2. Double-click on the required watch table in the folder.

Result

The selected watch table opens.

Copying and pasting a watch table

Requirement

A watch table has been created.

Procedure

To copy a watch table, follow these steps:

1. Right-click the watch table that you want to copy.
2. In the context menu, select "Copy".
3. In the project tree, open the folder structure for the CPU in which you want to paste the copied watch table.
4. Right-click on the "Watch and force tables" folder.
5. In the context menu, select "Paste".
6. Alternatively, you can select the entire contents of the watch table and Drag & Drop it onto another watch table.

Result

A copy of the selected watch table is placed in the "Watch and force tables" folder of the relevant CPU.

Saving a watch table

Prerequisite

A watch table has been created.

Procedure

To save a watch table, follow these steps:

1. In the project tree select the watch table you want to save.
2. If you wish to change the preset name of the table, select the "Rename" command in the context menu and enter a new name for the table.
3. In the "Project" menu, select "Save". Note that this save operation will save the entire project.

Result

The contents of the watch table and the project are saved.

Note

You can reuse saved watch tables to monitor and modify tags when retesting your program.

11.19.3.6 Entering tags in the watch table

Basic information on entering tags in the watch table

Recommended procedure

- Select the tags whose values you want to monitor or modify, and enter them in the watch table.
- When entering tags into the watch table, please note that these tags must be previously defined in the PLC tag table.
- When entering tags, work from the outside to the inside. This means that you start by entering the tags for the inputs in the watch table. Then, you enter the tags that are affected by the inputs or that affect the outputs. Finally, you enter the tags for the outputs.

Example of filling out a watch table

- Enter the absolute address to be monitored or modified in the "Address" column.
- Enter the symbolic name for the tag in the "Name" column.

- Select the display format you require from the drop-down list in the "Display format" column, if you do not want to use the default setting.
- Now decide whether you want to monitor or modify the entered tags. Enter the desired values for modifying as well as a comment in the corresponding columns of the watch table.

Create comment row

If required, you can create a comment row by entering the string "//" in the "Name" column.

Syntax check

When you enter the tags in the watch table, the syntax of each cell is checked when you exit the cell. Incorrect entries are marked in red.

Note

When you place the mouse pointer in a cell marked in red, brief information is displayed with additional notes on the error.

See also

Permitted operands for the watch table (Page 5203)

Permissible modify values for the watch table (Page 5204)

Permitted operands for the watch table

Permissible operands for the watch table

The following table shows the operands that are permitted for the watch table:

Permitted operand	Example of data type	Example (International mnemonics)
Input/output/bit memory	BOOL	I1.0, Q1.7, M10.1 I0.0:P; Q0.0:P
Input/output/bit memory	BYTE	IB1/QB10/MB100 IB1:P; QB1:P
Input/output/bit memory	WORD	IW1; QW10; MW100 IW2:P; QW3:P
Input/output/bit memory	DWORD	ID4; QD10; MD100 ID2:P; QD1:P
Timers	TIMER	T1
Counters	COUNTER	C1
Data block	BOOL	DB1.DBX1.0
Data block	BYTE	DB1.DBB1

Permitted operand	Example of data type	Example (International mnemonics)
Data block	WORD	DB1.DBW1
Data block	DWORD	DB1.DBD1

Note

Please observe the following notes to work with the watch table.

- You cannot enter "DB0..." because it is used by the system!
- Peripheral outputs can be modified but not monitored.
- Peripheral inputs can be monitored but not modified.

NOTICE
Danger of a time-out while monitoring peripheral inputs and controlling peripheral outputs
Note that the monitoring of peripheral inputs and the controlling of peripheral outputs in the watch table can result in a time-out.
The CPU assumes the "STOP" mode.

See also

Basic information on entering tags in the watch table (Page 5202)

Overview of the valid data types (Page 2239)

Permissible modify values for the watch table

Entry of modify values in the watch table

The following table shows the operands that are permitted for the entry of modify values in the watch table:

Table 11-87 Bit operands

Possible bit operands	Example for permitted modify values
I1.0	True
M1.7	False
Q1.0	0
Q1.1:P	1
DB1.DBX1.1	2#0
M1.6	2#1

Table 11-88 Byte operands

Possible byte operands	Example for permitted modify values
IB1	2#0011_0011
MB12	B#16#1F
QB10	1F
QB11:P	'a'
DB1.DBB1	10

Table 11-89 Word operands

Possible word operands	Example for permitted modify values
IW1	2#0011_0011_0011_0011
MW12	W#16#ABCD
MW14	ABCD
QW10	B#(12, 34)
QW12:P	12345
DB1.DBW1	'ab'
MW16	S5T#9s_340ms
MW18	C#123
MW9	D#2006-12-31

Table 11-90 Double word operands

Possible double word operands	Example for permitted modify values
ID1	2#0011_0011_0011_0011_0011_0011_0011_0011
QD10	Dw#16#abcdef10
QD12:P	ABCDEF10
DB1.DBD2	b#(12,34,56,78)
MD8	L#-12
MD12	L#12
MD16	123456789
MD20	123456789
MD24	T#12s_345ms
MD28	Tod#1:2:34.567
MD32	P#e0.0

Table 11-91 Timers

Possible operands of the "Timer" type	Permitted control values	Explanation
T1	0 ms	Time value in milliseconds (ms)
T12	20 ms	Time value in milliseconds (ms)

Possible operands of the "Timer" type	Permitted control values	Explanation
T14	12400 ms	Time value in milliseconds (ms)
T16	S5t#12s300ms	Time value 12s 300 ms

Table 11-92 Counters

Possible operands of the "Counter" type	Permitted control values
C1	0
C14	20
C16	C#123

Notes on timers and counters

- Timers

  Note

  Modifying a timer influences only the value, not the status. Timer T1 can be modified to the value "0", but the result of logic operation for A T1 is not changed.

  The time sequences "s5t" and "s5time" can be written in both lower-case and upper-case characters.

- Counters

  Note

  Modifying a counter influences only the value, not the status. Counter C1 can be changed to the value "0", but the result of the logic operation for A C1 is not changed.

Overview of the display formats

Display formats in the watch table

The display format you select specifies the representation of a tag value.

When entering the address a display format is automatically preset. If you want to change this, you can select a display format from the drop-down list in the "Display formats" column. The drop-down list only offers the display formats which are valid for this data type. The display format that appears first in the list is the pre-selected format.

Example

The following table shows the 32-bit data types permitted for all CPU families in the watch table and their possible display formats:

Data type	Possible display formats
BOOL	Bool, Hex, BCD, Octal, Bin, Dec, Dec+/-
BYTE	Hex, BCD, Octal, Bin, Dec, Dec+/-, Character
WORD	Hex, BCD, Octal, Bin, Dec, Dec+/-, Dec_Sequence, Character, Unicode_Character, SIMATIC_Time, Date, Counter
DWORD	Hex, BCD, Octal, Bin, Dec, Dec+/-, Dec_Sequence, Character, Unicode_Character, Floating-point number, Time of day, Timer
SINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character
INT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, SIMATIC_Time, Counter, Date
DINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, Time of day, Timer
USINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character
UINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, SIMATIC_Time, Counter, Date
UDINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, Time of day, Timer
REAL	Floating-point number, Hex, BCD, Octal, Bin, Character, Unicode_Character,Dec, Dec+/-, Dec_Sequence, Time of day, Timer
DATE	Date, Dec, Hex, Bin
TIME_OF_DAY	Time of day, Dec, Hex, Bin
TIME	Timer, Hex, Bin, Dec, Dec+/-
DATE_AND_TIME	Date and time,
TIMER	SIMATIC_Time, Hex, BCD, Bin
CHAR	Character, Hex, BCD, Octal, Bin, Dec, Dec+/-
WCHAR	Unicode_Character, Character, Hex, BCD, Octal, Bin, Dec, Dec+/-
STRING	Character string
WSTRING	Unicode_character string
POINTER	Pointer, Hex
COUNTER	Counter, Hex, BCD, Bin
S5TIME	SIMATIC_Time, Hex, BCD, Bin

For the S7-1200 CPU family, all 32-bit data types are permitted (see table above), as well as the 64-bit data type LREAL with the following possible display formats:

Data type	Possible display formats
LREAL	In a project created with TIA Portal < V12: Floating-point number Note: The display of LREAL is limited to 13 digits plus exponent.
LREAL	In a project created with TIA Portal >= V12: Floating-point number, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec, Dec+/-, Dec_Sequence, Time of day, Time, Date and time Note: The display of LREAL is limited to 13 digits plus exponent.

For the S7-1500 CPU family, in addition to 32-bit data types, the 64-bit data types listed in the table are also permitted with the following possible display formats:

Data type	Possible display formats
LWORD	Hex, Octal, BCD, Bin, Character, Unicode_Character, Dec, Dec+/-, Dec_Sequence, Floating-point number, Time of day, Time, Date and time
LINT	Dec+/-, Dec, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, Time of day, Time, Date and time
ULINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, Time of day, Time, Date and time
LREAL	Floating-point number, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec, Dec+/-, Dec_Sequence, Time of day, Time, Date and time
LTIME	Timer, Dec+/-, Dec, Hex, Bin
LTOD	Time of day, Dec, Hex, Bin
LDT	Date and time, Dec, Hex, Bin

For more information, refer to the description of the valid data types (Page 2239).

Note

Rounding of floating-point numbers

In the watch table, floating-point numbers are stored as binary numbers in IEEE format. Because not every floating point number (real, longreal) that can be displayed on the user interface can be mapped to the IEEE format, there is a possibility that floating-point numbers will be rounded. If a rounded floating-point number in the watch table is copied and, in turn, inserted in another input field, the rounding may cause a slight difference.

Note

Only symbolic addressing is possible

In the watch table, LongDataTypes such as LWORD or LREAL can only be addressed symbolically.

Selecting the display format for tags

Procedure

To select the display format of the tags, follow these steps:

1. Enter the desired address in the watch table.
2. Click the desired cell in the "Display format" column, and open the drop-down list. The permissible display formats are shown in the drop-down list.
3. Select the desired display format from the drop-down list.

Note

If the selected display format cannot be applied, then the last selected display format will be displayed automatically.

See also

Overview of the valid data types (Page 2239)

Creating and editing comment lines

Basic principles of comment lines

In addition to the row related comments in the comment column, you can now also create complete comment lines to enhance the structure of the watch table.

The contents of the comment line are stored in the "Languages & Resources" folder in the "Project texts" tab and can be compiled in other project languages.

Alternatively, you have the following options to enter comment lines:

- Using the corresponding icon in the toolbar of the watch table.
- Using the menu command "Insert > Insert comment line".
- Using the shortcut menu command "Insert comment line".
- By entering the string "//" in the "Name" column of the watch table.

Inserting comment lines

To insert comment lines, follow these steps:

1. Open the watch table and enter the required addresses.
2. Insert a new comment line:
   - To do so, click the "Insert comment line" icon in the toolbar of the watch table.
   - Or click the menu command "Insert > Insert comment line".
   - Or click the shortcut menu command "Insert comment line".
     Result: A new comment line is inserted above the selected row in the watch table.
3. Enter the required comment in the comment line. The entered comment is shown in green.
4. To show all comments you entered, double-click "Project texts" in the project tree under "Languages & Resources".
5. If you are working in multi-lingual projects and want your comment to be translated into other languages, you can set the project languages required in addition to the editing language in the project tree under "Languages & Resources > Project languages".

Creating comment lines

To create comment lines, follow these steps:

1. Open the watch table and enter the required addresses.
2. Enter the character string "//" in a free row in the "Name" column. This turns the row into a comment line.
   No comment lines can be created in the other columns.
3. Enter the required comment in the comment line. The entered comment is shown in green.
4. To show all comments you entered, double-click "Project texts" in the project tree under "Languages & Resources".
5. If you are working in multi-lingual projects and want your comment to be translated into other languages, you can set the project languages required in addition to the editing language in the project tree under "Languages & Resources > Project languages".

Deleting comment lines

To delete comment lines, follow these steps:

1. Open a watch table containing comment lines.
2. Delete the entire comment with the introductory string "//", if you no longer required this.
3. Alternatively, delete only the introductory string "//". In this case the existing comment is retained and is displayed in the "Comment" column in the watch table.

Note

Deleting comment lines

When you delete comment lines the project languages and any existing translations for these comments are also deleted.

11.19.3.7 Monitoring tags in the watch table

Introduction to monitoring tags in the watch table

Introduction

The watch table allows you to monitor the tags of the configured input and output modules in the CPU, depending on the monitoring and modify mode (Page 5212) selected. To monitor tags, an online connection to the CPU must exist.

NOTICE
Danger of a time-out while monitoring peripheral inputs
Note that the monitoring of peripheral inputs can result in a time-out.
The CPU assumes the "STOP" mode.

Options for monitoring tags

The following options are available for monitoring tags:

- Monitor now
  This command starts the monitoring of the visible tags in the active watch table immediately and once only.
- Monitor all
  This command starts the monitoring of all visible tags in the active watch table, depending on the selected watch mode:
  - In basic mode, the monitoring mode is set to "permanent" by default.
  - In expanded mode, you can specify defined trigger points for the monitoring of tags.

  Note

  If the monitoring mode is changed while in expanded mode and then a switch is made to basic mode, the monitoring mode set before will also be applied in basic mode.

CPU-specific limitations when monitoring tags

The following CPU-specific differences exist:

- CPU S7-300/400:
  CPUs from this family can only monitor the first 30 characters of a string.
- CPU S7-1200/1500:
  CPUs from this family can monitor a string up to the total size of 254 characters.

Setting the monitoring and modify mode

Introduction

By selecting the monitoring and modify mode, you specify the trigger point and the duration of the tag monitoring in the watch table and the force table.

Possible monitoring and modify modes (duration of monitoring or modifying)

The following monitoring and modifying modes are available:

Trigger	Execution	CPU status	Duration
Permanent	Permanent During monitoring: The inputs are monitored at the end of the cycle and the outputs at the start of the cycle. During modify: The inputs are modified at the start of the cycle and the outputs at the end of the cycle.	RUN	Will be executed until the user stops the action or the online connection to the CPU is interrupted.
Permanently, at start of scan cycle	Permanently, at start of scan cycle	RUN	Will be executed until the user stops the action or the online connection to the CPU is interrupted.
Permanently, at end of scan cycle	Permanently, at end of scan cycle	RUN	Will be executed until the user stops the action or the online connection to the CPU is interrupted.
Permanently, at transition to STOP	Permanently, at transition from RUN to STOP	RUN > STOP	Will be executed until the user stops the action or the online connection to the CPU is interrupted.
Once only, at start of scan cycle	Once only, at start of scan cycle	RUN	Ends automatically after being executed once.
Once only, at end of scan cycle	Once only, at end of scan cycle	RUN	Ends automatically after being executed once.
Once only, at transition to STOP	Once only, at transition from RUN to STOP	STOP > RUN	Ends automatically after being executed once.

Special features when using "Permanent" mode

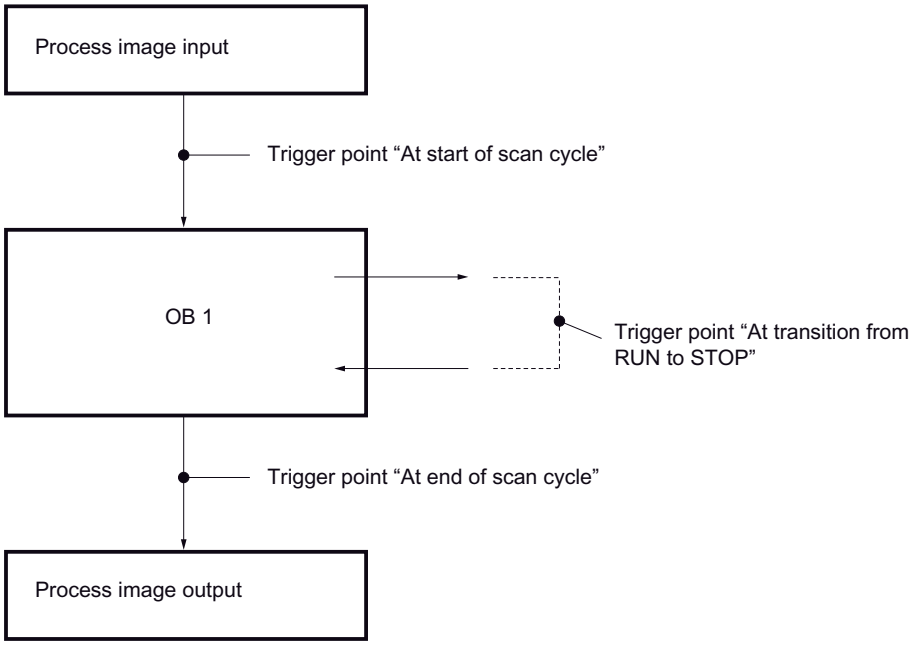
The "Permanent" mode is executed differently for the monitoring and modifying of tags.

- Monitoring: The inputs are monitored at the end of the cycle and the outputs at the start of the cycle.
- Modifying: The inputs are modified at the start of the cycle and the outputs at the end of the cycle.

Selecting the trigger point

The trigger points "Beginning of scan cycle", "End of scan cycle", and "Switch to stop" specify the time at which the tags are to be read from the CPU or updated in the CPU.

The following diagram shows the position of these trigger points:



Position of the trigger points

From the position of the trigger points, it follows that:

- Modifying of inputs is only appropriate at the beginning of the scan cycle (corresponding to the beginning of the user program OB 1), because otherwise the process image input is updated again after modifying and is thus overwritten.
- Modifying of outputs is only appropriate at the end of the scan cycle (corresponding to the end of the user program OB 1), because otherwise the process image output can be overwritten by the user program.
- The modified value is displayed in the "Monitor value" column, provided that monitoring is active and the modified value is not overwritten by the user program.

Monitoring tags

When tags are being modified, the following applies to the trigger points:

- If you have specified the modify mode as "once only", you will receive an alarm if the selected tags cannot be modified.
- In "permanent" modify mode, you do not receive an alarm.

Note regarding the "Modify now" command

You can modify the values of selected tags immediately using the "Online > Modify >Modify now" command. This command is executed once only and as quickly as possible without reference to a defined position (trigger point) in the user program. This function is used mainly for modifying when the CPU is in STOP mode.

"Monitor all" command for tags

Introduction

The "Monitor all" command allows you to start monitoring the visible tags in the active watch table. The default setting for the monitoring mode in basic mode of the watch table is "permanent". In expanded mode, you can specify defined trigger points for the monitoring of tags. In this case, the tags are monitored with reference to the specified trigger points.

NOTICE
Danger of a time-out while monitoring peripheral inputs
Note that the monitoring of peripheral inputs can result in a time-out.
The CPU assumes the "STOP" mode.

Requirements

- A watch table has been created.
- An online connection to the CPU exists.

Procedure

To execute the "Monitor all" command, follow these steps:

1. Enter the tags to be monitored and the corresponding addresses in the watch table.
2. Switch to expanded mode by clicking the icon "Show/hide advanced setting columns" in the toolbar.
3. If you want to change the default monitoring mode for a tag, click the appropriate cell in the "Monitor with trigger" column and select the desired monitoring mode from the drop-down list.
4. Click the "Monitor all" icon in the toolbar.

Result

The tags of the active watch table are monitored using the monitoring mode selected.

See also

Icons in the watch table (Page 5199)

Basic mode and expanded mode in the watch table (Page 5198)

Entering tags in the watch table (Page 5202)

"Monitor now" command for tags

Introduction

The "Monitor now" command starts the monitoring of tags immediately without reference to defined trigger points. The tag values are read out once only and displayed in the watch table.

NOTICE
Danger of a time-out while monitoring peripheral inputs
Note that the monitoring of peripheral inputs can result in a time-out.
The CPU assumes the "STOP" mode.

Requirements

- A watch table has been created.
- An online connection to the CPU exists.

Procedure

To execute the "Monitor now" command, follow these steps:

1. Enter the tags to be monitored and the corresponding addresses in the watch table.
2. Click the "Monitor now" icon in the toolbar.

Result

The tags of the active watch table are monitored immediately and once only.

See also

Icons in the watch table (Page 5199)

Basic mode and expanded mode in the watch table (Page 5198)

Entering tags in the watch table (Page 5202)

11.19.3.8 Modifying tags in the watch table

Introduction to modifying tags

Introduction

The watch table allows you to modify the tags of the configured input and output modules in the CPU, depending on the monitoring and modify mode (Page 5212) selected.

To monitor the tags, an online connection to the CPU must exist.

DANGER

Danger when modifying:

Serious personal injury and material damage can result from changes in the tags or addresses during plant operation in the event of malfunctions or program errors!
Make sure that dangerous conditions cannot occur before you execute the "Modify" function.

NOTICE

Danger of a time-out while controlling peripheral outputs

Note that the controlling of peripheral outputs in the watch table can result in a time-out.

The CPU assumes the "STOP" mode.

NOTICE

Danger due to modifying an identical operand in parallel with different modify values in more than one watch table

When working with more than one watch table, avoid modifying identical operands permanently multiple times with different modify values.

If an identical operand is modified permanently with different modify values at the same time in different watch tables, all watch tables will display the last modified value, because the modify value assigned last will be used in this case.

Options for modifying tags

The following options are available for modifying tags:

- Modify to "0"
  This command modifies the selected address to the modify value "0".
- Modify to "1"
  This command modifies the selected address to the modify value "1".
- Modify once only and immediately
  This command modifies all selected addresses in the active watch table "once only and immediately".
- Modify with trigger
  This command modifies all selected addresses in the active watch table using the monitoring and modify mode (Page 5212) selected.
  The "Modify with trigger" function is only available in expanded mode. You will not receive an alarm indicating whether or not the selected addresses were actually modified with the specified value. You should use the "Modify once only and immediately" function if you require such a confirmation.
- Enable peripheral outputs
  This command disables the command output disable.
  This function can only be executed in expanded mode, when the CPU is in STOP and the option Force (Page 5241) of tags is not enabled. If desired, deactivate this function in the force table.

Note

When modifying, note the following:

Modifying of tags **cannot** be undone.

Modify tags to "0"

Introduction

You can assign one-time values to tags independent of the monitoring and modify mode and modify them. The modify command is executed as fast as possible, similar to a "Trigger now" command, without reference to a defined position in the user program.

⚠ DANGER

Danger when modifying:

Serious personal injury and material damage can result from changes in the tags or addresses during plant operation in the event of malfunctions or program errors!
Make sure that dangerous conditions cannot occur before you execute the "Modify" function.

NOTICE
Danger of a time-out while controlling peripheral outputs
Note that the controlling of peripheral outputs in the watch table can result in a time-out.
The CPU assumes the "STOP" mode.

Requirements

- A watch table has been created.
- An online connection to the CPU exists.

Procedure

To modify tags to "0", follow these steps:

1. Enter the desired address in the watch table.
2. Select the "Online > Modify > Modify to 0" command in order to modify the selected address with the specified value.

Result

The selected address is modified to "0".

Note

When modifying, note the following:

Modifying can **not** be undone!

Modify tags to "1"

Introduction

You can assign one-time values to tags independent of the monitoring and modify mode and modify them. The modify command is executed as fast as possible, similar to a "Trigger now" command, without reference to a defined position in the user program.

⚠ DANGER
Danger when modifying:
Serious personal injury and material damage can result from changes in the tags or addresses during plant operation in the event of malfunctions or program errors! Make sure that dangerous conditions cannot occur before you execute the "Modify" function.

NOTICE
Danger of a time-out while controlling peripheral outputs
Note that the controlling of peripheral outputs in the watch table can result in a time-out.
The CPU assumes the "STOP" mode.

Requirements

- A watch table has been created.
- An online connection to the CPU exists.

Procedure

To modify tags to "1", follow these steps:

1. Enter the desired address in the watch table.
2. Select the "Online > Modify > Modify to 1" command in order to modify the selected address with the specified value.

Result

The selected address is modified to "1".

Note

When modifying, note the following:

Modifying can **not** be undone!

"Modify now" command for tags

Introduction

You can assign one-time values to tags independent of the monitoring and modify mode and modify them immediately. The modify command is executed as fast as possible, similar to a "Trigger now" command, without reference to a defined position in the user program.

⚠ DANGER
Danger when modifying:
Serious personal injury and material damage can result from changes in the tags or addresses during plant operation in the event of malfunctions or program errors! Make sure that dangerous conditions cannot occur before you execute the "Modify" function.

NOTICE
Danger of a time-out while controlling peripheral outputs
Note that the controlling of peripheral outputs in the watch table can result in a time-out.
The CPU assumes the "STOP" mode.

Requirements

- A watch table has been created.
- An online connection to the CPU exists.

Procedure

To modify tags immediately, follow these steps:

1. Enter the desired addresses and modify values in the watch table.
2. Select the addresses to be modified by selecting the check boxes for modifying in the column after the "Modify value".
   A yellow triangle appears behind the selected check box, indicating that the address is now selected for modifying but has not yet been modified.
3. Select the "Online > Modify > Modify once and now" command in order to immediately modify the selected address once only with the specified value.

Result

The selected addresses are modified immediately and once only.

Note

When modifying, note the following:

Modifying can **not** be undone!

"Modify with trigger" command for tags

Introduction

You can assign values to addresses dependent on the defined monitoring and modify mode and modify them. The modify command is executed as specified in the monitoring and modify mode, with reference to the defined trigger position in the user program.

DANGER

Danger when modifying:

Serious personal injury and material damage can result from changes in the tags or addresses during plant operation in the event of malfunctions or program errors!
Make sure that dangerous conditions cannot occur before you execute the "Modify" function.

NOTICE

Danger of a time-out while controlling peripheral outputs

Note that the controlling of peripheral outputs in the watch table can result in a time-out.

The CPU assumes the "STOP" mode.

Requirements

- A watch table has been created.
- An online connection to the CPU exists.
- The watch table has to be in expanded mode.

Procedure

To modify tags "with trigger", follow these steps:

1. Enter the desired addresses and modify values in the watch table.
2. Select the addresses to be modified by selecting the check boxes for modifying in the column after the "Modify value".
   A yellow triangle appears behind the selected check box, indicating that the address is now selected for modifying but has not yet been modified.
3. Switch to expanded mode using the icon "Show/hide advanced settings columns" in the toolbar or the "Online > Expanded mode" command.
   The "Monitor with trigger" and "Modify with trigger" columns are displayed.

4. In the "Modify with trigger" column, select the desired modify mode from the drop-down list box. The following options are available:
   - Permanent
   - Permanently, at start of scan cycle
   - Once only, at start of scan cycle
   - Permanently, at end of scan cycle
   - Once only, at end of scan cycle
   - Permanently, at transition to STOP
   - Once only, at transition to STOP
5. Start modifying using the "Online > Modify > Modify with trigger" command.
6. Confirm the prompt with "Yes" if you want to start modifying with trigger.

Result

The selected tags are modified using the selected monitoring and modify mode. The yellow triangle is no longer displayed.

Note

When modifying, note the following:

Modifying can **not** be undone!

Enable peripheral outputs

Introduction

The "Enable peripheral outputs" function deactivates the command output disable of the peripheral outputs. You can then modify the peripheral outputs when the CPU is in STOP mode.

This function is available in the watch table in "Expanded mode" only and only for CPUs of the S7-300/400 family.

⚠ DANGER

Danger when enabling the peripheral outputs:

Attention, the enabling of the peripheral outputs can cause serious personal injury and material damage!
Make certain that dangerous conditions cannot occur before you execute the "Enable peripheral outputs" function.

Prerequisites

- A watch table has been created.
- An online connection to the CPU exists.
- The CPU is in STOP mode before you can enable the peripheral outputs.
- The watch table has to be in expanded mode.
- The option Force (Page 5241) of tags must not be enabled.

Note

"Enable peripheral outputs" function

- This function is possible only in STOP mode. The function is exited by an operating state change of the CPU and by the termination of the online connection.
- While the function is enabled, forcing is not possible.

Procedure

To enable the peripheral outputs in STOP mode, follow these steps:

1. Enter the desired addresses and modify values in the watch table.
2. Select the addresses to be modified by selecting the check boxes for modifying in the column after the "Modify value".
   A yellow triangle appears behind the selected check box, indicating that the address is now selected for modifying but has not yet been modified.
3. Switch to expanded mode using the icon "Show/hide advanced settings columns" in the toolbar or the "Online > Expanded mode" command.
   The "Monitor with trigger" and "Modify with trigger" columns are displayed.
4. Change the relevant CPU to STOP using the operator panel.
5. Right-click to open the shortcut menu and select "Enable peripheral outputs".
6. Confirm the prompt with "Yes" if you want to unlock the command output disable for the peripheral outputs.
7. Modify the peripheral outputs using the "Online > Modify > Modify now" command.

Result

The peripheral outputs are modified with the selected modify values. The yellow triangle is no longer displayed.

Enabling the peripheral outputs

The "Enable peripheral outputs" function remains active until:

- The "Enable peripheral outputs" command is deactivated again via the shortcut menu or via the "Online > Modify > Enable peripheral outputs" command.
- The CPU is no longer in STOP mode.
- The online connection is terminated.

Note

When modifying, note the following:

Modifying can **not** be undone!

11.19.4 Testing with the force table

11.19.4.1 Introduction for testing with the force table

Overview

You can use the force table to assign permanent values to individual tags

of the user program. This action is referred to as "forcing".

The following functions are available in the force table:

- **Monitoring tags**
  This allows the current values of the individual tags of a user program or a CPU to be displayed on the programming device or PC. Tags can be monitored with or without a trigger condition.
- **Forcing tags**
  This function lets you assign a fixed value to individual I/O tags of the user program.

Monitoring and forcing tags

The monitoring and forcing of tags is always dependent on the operand scope of the CPU used.

The following tags can be monitored:

- Inputs, outputs, and bit memories
- Contents of data blocks
- Peripheral inputs

The following tags can be forced:

- Peripheral inputs
- Peripheral outputs

Example

Independent of the CPU used, only I/O can be forced, such as: "Tag_1":P or "QW0:P" or "IW0:P". Note that "Tag_1":P must not be the symbolic name of a bit memory.

Possible applications

One advantage of the force table is that you can simulate different test environments and overwrite tags in the CPU with a permanent value. This enables you to intervene in the ongoing process for regulating purposes.

See also

Layout of the force table (Page 5226)

Basic mode and expanded mode in the force table (Page 5227)

Icons in the force table (Page 5228)

Open and edit force table (Page 5229)

11.19.4.2 Safety precautions when forcing tags

Safety precautions when forcing tags

Because the forcing function allows you to intervene permanently in the process, observance of the following notices is essential:

DANGER

Prevent personal injury and material damage!

Note that an incorrect action when executing the "Force" function can:

- Harm persons or pose a health hazard.
- Cause damage to machinery or the entire plant.

CAUTION

Prevent personal injury and material damage!

- Before you start the "Force" function, you should ensure that no one else is currently executing this function on the same CPU.
- Forcing can only be stopped by clicking the "Stop forcing" icon or using the "Online > Force > Stop forcing" command. Closing the active force table does **not** stop the forcing!
- Forcing can **not** be undone!
- Review the differences between " modifying tags" (Page 5216) and "forcing tags" (Page 5241).
- If a CPU does not support the "Force" function, the relevant icons cannot be selected.
- If the function "Enable peripheral outputs" is active on your CPU, then forcing is **not** possible on this CPU. If desired, deactivate this function in the watch table.

11.19.4.3 Layout of the force table

Introduction

In the force table, enter the CPU-wide tags that you have defined and selected and which are to be forced in the allocated CPU. Only peripheral inputs and peripheral outputs can be forced.

For each CPU created in the project, a force table will automatically be created in the "Watch and force tables" folder. Only one force table can be allocated to a CPU. This force table displays all the addresses forced in the allocated CPU.

Layout of the force table

The columns displayed in the force table depend on the mode you are working in: basic mode or expanded mode.

In expanded mode the "Monitor with trigger" column is also displayed.

Meaning of the columns

The following table shows the meaning of the individual columns in basic mode and expanded mode:

Mode	Column	Meaning
Basic mode	i	Identification column
	Name	Name of the inserted tag
	Address	Address of the inserted tag
	Display format	Selected display format
	Monitor value	Values of the tags, dependent on the selected display format.
	Force value	Value with which the tag is forced.
	F ("Force")	Select the tag to be forced by activating the corresponding check box.
	Comment	Option to enter a comment to document the tags in the force table.
	Tag comment	Display of the comment on the selected tag that was entered in other editors. This column cannot be edited.
The following additional column is shown in expanded mode:	Monitor with trigger	Display of selected monitoring mode

See also

Icons in the force table (Page 5228)

Basic mode and expanded mode in the force table (Page 5227)

11.19.4.4 Basic mode and expanded mode in the force table

Difference between basic mode and expanded mode in the force table

In expanded mode the "Monitor with trigger" column is also displayed in the force table.

You will find a detailed list of the columns under Layout of the force table (Page 5226).

Switching between basic mode and expanded mode

You have the following options of toggling between the basic and expanded mode:

- Click the icon "Show/hide advanced setting columns". Click this icon again to return to the basic mode.
  Or:
- In the "Online" menu, select the "Expanded mode" check box. Deactivate this check box to return to the basic mode.

Functionality in expanded mode

The following functionality is only possible in expanded mode:

- Monitor with trigger
- Monitor peripheral inputs

11.19.4.5 Icons in the force table

Meaning of the icons

The following table shows the meaning of the icons in the force table:

Icon	Meaning
	Identifies a table inside the project tree as a force table.
	Identification column
	Inserts a row before the selected row.
	Inserts a row after the selected row.
	Inserts a comment line above the selected row.
	Displays all columns of expanded mode. If you click this icon again, the columns of the expanded mode will be hidden.
	Updates all operands and values currently being forced on the CPU in the open force table.
	Starts forcing for all addresses of the selected tags. If forcing is already running, the previous action is replaced without interruption.
	Stops forcing of addresses in the force table.
	Starts monitoring of the visible tags in the force table. The default setting for monitoring in basic mode is "permanent". In expanded mode an additional column is shown and you can set certain trigger points for monitoring tags.
	Starts monitoring of the visible tags in the force table. This command is executed immediately and the tags are monitored once.
	Displays the check box for the selection of tags to be forced.
	Indicates that an address cannot be fully forced. Example: It is possible to force the address QW0:P, but it is not possible to force the address QD0:P because this address area is potentially not available on the CPU.
	Indicates that an address cannot be monitored.
	Indicates that an address is being forced.
	Indicates that an address is being partly forced.
	Indicates that the associated peripheral address is being forced.
	Indicates that a syntax error occurred.

Icon	Meaning
	Indicates that the address is selected but has not been forced yet.
	Indicates that the selected tag was monitored for the value "1".
	Indicates that the selected tag was monitored for the value "0".
	Indicates that the address is being used more than once.
	Indicates that a substitute value is being used. Substitute values are values that are output to the process in case of signal output module faults or are used instead of a process value in the user program in case of signal input module faults. The substitute values can be assigned by the user (e.g., retain old value).

See also

Layout of the force table (Page 5226)

11.19.4.6 Open and edit force table

Display force table

Introduction

You cannot create a new force table; one force table already exists for each CPU. It is permanently allocated to this CPU and cannot be copied or duplicated.

Requirements

A project with an allocated CPU has to be open.

Displaying a force table

The force table is always displayed below a CPU in the "Watch and force tables" folder.

Open force table

Requirements

A project with an allocated CPU must be created.

Procedure

Proceed as follows to open a force table:

1. Open the "Watch and force tables" folder below the desired CPU.
2. Double-click the "Force table" in this folder.

Result

The selected force table opens.

Save force table

Requirements

A project with an allocated CPU has been created.

Procedure

Proceed as follows to save a force table:

1. Enter the desired changes in the force table.
2. Select the "Save" command in the "Project" menu or click the "Save project" icon in the toolbar. Note that this save operation will save the entire project.

Result

The contents of the force table and the associated project are saved.

Note

You cannot rename a force table.

11.19.4.7 Entering tags in the force table

Basic principles for entering tags in the force table

Recommended procedure

Select the tags whose values you want to monitor or force, and enter them in the force table.

When entering tags in the force table, please note that these tags must be previously defined in the PLC tag table.

Example of filling out a force table

- You can enter the absolute address that is to be forced or monitored in the "Address" column or you can enter the symbolic name of the tag in the "Name" column.
- Select the display format you require from the drop-down list in the "Display format" column, if you do not want to use the default setting.
- Now you have to decide whether you want to monitor or force the entered tags. Enter the required force value and a comment in the appropriate columns of the force table.
- Note that only peripheral inputs and peripheral outputs can be forced and review the Safety precautions when forcing tags (Page 5243).

Create comment line

If required, you can create a comment row by entering the string "//" in the "Name" column.

Syntax check

When you enter tags in the force table, the syntax of each cell will be checked when you exit the cell. Incorrect entries are marked in red.

Note

When you place the mouse pointer in a cell marked in red, brief information is displayed with additional notes on the error.

Permitted operands for the force table

Permitted operands for the force table

The following table shows the operands that are permitted for forcing in the force table:

Permitted operand	Example of data type	Example (International mnemonics)
Peripheral input/peripheral output	BOOL	I0.0:P; Q0.0:P
Peripheral input/peripheral output	BYTE	IB1:P; QB1:P
Peripheral input/peripheral output	WORD	IW2:P; QW3:P
Peripheral input/peripheral output	DWORD	ID2:P; QD1:P

The following table shows the operands that are permitted for monitoring in the force table:

Permitted operand	Example of data type	Example (International mnemonics)
Input/output/bit memory	BOOL	I1.0, Q1.7, M10.1 I0.0:P
Input/output/bit memory	BYTE	IB1/QB10/MB100 IB1:P
Input/output/bit memory	WORD	IW1; QW10; MW100 IW2:P
Input/output/bit memory	DWORD	ID4; QD10; MD100 ID2:P
Timers	TIMER	T1
Counters	COUNTER	C1
Data block	BOOL	DB1.DBX1.0
Data block	BYTE	DB1.DBB1
Data block	WORD	DB1.DBW1
Data block	DWORD	DB1.DBD1

Note

You cannot enter "DB0..." because it is used by the system!

Permitted force values for the force table

Entering force values in the force table

The following table shows the operands that are permitted for entering force values in the force table:

Table 11-93 Bit operands

Possible bit operands	Example for permitted force values
I1.0:P	True
I1.1:P	False
Q1.0P	0
Q1.1:P	1
I2.0:P	2#0
I2.1:P	2#1

Table 11-94 Byte operands

Possible byte operands	Example for permitted force values
IB1:P	2#00110011
IB2:P	B#16#1F
QB14:P	1F
QB10:P	'a'
IB3:P	10

Table 11-95 Word operands

Possible word operands	Example for permitted force values
IW0:P	2#0011001100110011
IW2:P	W#16#ABCD
QW10:P	ABCD
QW12:P	B#(12, 34)
IW4:P	'ab'
IW6:P	12300
IW8:P	S5T#9S_300ms
IW10:P	C#123
IW12:P	D#2006-12-31

Table 11-96 Double word operands

Possible double word operands	Example for permitted force values
ID0:P	2#00110011001100110011001100110011
ID4:P	1.2
QD10:P	1.234.e4
QD14:P	Dw#16#abcdef10
ID8:P	16#ABCDEF10
ID12:P	b#(12,34,56,78)
ID16:P	L#-12
ID20:P	L#12
ID24:P	123456789
ID28:P	123456789
ID32:P	T#12s300ms
ID36:P	Tod#14:20:40.645
ID40:P	P#e0.0

Overview of the display formats

Display formats in the force table

The display format you select specifies the representation of a tag value.

When entering the address a display format is automatically preset. If you want to change this, you can select a display format from the drop-down list in the "Display formats" column. The drop-down list only offers the display formats which are valid for this data type. The display format that appears first in the list is the pre-selected format.

Example

The following table shows the 32-bit data types permitted for all CPU families in the force table and their possible display formats:

Data type	Possible display formats
BOOL	Bool, Hex, BCD, Octal, Bin, Dec, Dec+/-
BYTE	Hex, BCD, Octal, Bin, Dec, Dec+/-, Character
WORD	Hex, BCD, Octal, Bin, Dec, Dec+/-, Dec_Sequence, Character, SIMATIC_Time, Date, Unicode_Character, Counter
DWORD	Hex, BCD, Octal, Bin, Dec, Dec+/-, Dec_Sequence, Character, Floating-point number, Time of day, Timer, Unicode_Character
SINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character
INT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, SIMATIC_Time, Counter, Date
DINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, Time of day, Timer
USINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character
UINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, SIMATIC_Time, Counter, Date
UDINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, Time of day, Timer
REAL	Floating-point number, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec, Dec+/-, Dec_Sequence, Time of day, Timer
DATE	Date, Dec, Hex, Bin
TIME_OF_DAY	Time of day, Dec, Hex, Bin
TIME	Timer, Hex, Bin, Dec, Dec+/-
DATE_AND_TIME	Date and time,
TIMER	SIMATIC_Time, Hex, BCD, Bin
CHAR	Character, Hex, BCD, Octal, Bin, Dec, Dec+/-
WCHAR	Unicode_Character, Character, Hex, BCD, Octal, Bin, Dec, Dec+/-
STRING	Character string
WSTRING	Unicode_character string
POINTER	Pointer, Hex
COUNTER	Counter, Hex, BCD, Bin
S5TIME	SIMATIC_Time, Hex, BCD, Bin

For the S7-1200 CPU family, all 32-bit data types are permitted (see table above), as well as the 64-bit data type LREAL with the following possible display formats:

Data type	Possible display formats
LREAL	In a project created with TIA Portal < V12: Floating-point number Note: The display of LREAL is limited to 13 digits plus exponent.
LREAL	In a project created with TIA Portal >= V12: Floating-point number, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec, Dec+/-, Dec_Sequence, Time of day, Time, Date and time Note: The display of LREAL is limited to 13 digits plus exponent.

For the S7-1500 CPU family, in addition to 32-bit data types, the 64-bit data types listed in the table are also permitted with the following possible display formats:

Data type	Possible display formats
LWORD	Hex, Octal, BCD, Bin, Character, Unicode_Character, Dec, Dec+/-, Dec_Sequence, Floating-point number, Time of day, Time, Date and time
LINT	Dec+/-, Dec, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, Time of day, Time, Date and time
ULINT	Dec, Dec+/-, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec_Sequence, Time of day, Time, Date and time
LREAL	Floating-point number, Hex, BCD, Octal, Bin, Character, Unicode_Character, Dec, Dec+/-, Dec_Sequence, Time of day, Time, Date and time
LTIME	Timer, Dec+/-, Dec, Hex, Bin
LTOD	Dec, Hex, Bin, Time of day
LDT	Dec, Hex, Date and time, Bin

For more information, refer to the description of the valid data types (Page 2239).

Note

Rounding of floating-point numbers

In the force table, floating-point numbers are stored as binary numbers in IEEE format. Because not every floating point number (real, longreal) that can be displayed on the user interface can be mapped to the IEEE format, there is a possibility that floating-point numbers will be rounded. If a rounded floating-point number in the force table is copied and, in turn, inserted in another input field, the rounding may cause a slight difference.

Note

Only symbolic addressing is possible

In the force table, LongDataTypes such as LWORD or LREAL can only be addressed symbolically.

Selecting the display format for tags

Procedure

To select the display format of the tags, follow these steps:

1. Enter the desired address in the force table.
2. Click the desired cell in the "Display format" column, and open the drop-down list. The permitted display formats are shown in the drop-down list.
3. Select the desired display format from the drop-down list.

Note

If the selected display format cannot be applied, then the last selected display format will be displayed automatically.

Creating and editing comment lines

Basic principles of comment lines

In addition to the row-related comments in the comment column, you can now also create complete comment lines to enhance the structure of the force table.

The contents of the comment line are stored in the "Languages & Resources" folder in the "Project texts" tab and can be compiled in other project languages.

- Using the corresponding icon in the toolbar of the force table.
- Using the menu command "Insert > Insert comment line".
- Using the shortcut menu command "Insert comment line".
- By entering the string "//" in the "Name" column of the force table.

Inserting comment lines

To insert comment lines, follow these steps:

1. Open the force table and enter the required addresses.
2. Insert a new comment line:
   - To do so, click the "Insert comment line" icon in the toolbar of the force table.
   - Or click the menu command "Insert > Insert comment line".
   - Or click the shortcut menu command "Insert comment line".
     Result: A new comment line is inserted above the selected row in the force table.
3. Enter the required comment in the comment line. The entered comment is shown in green.
4. To show all comments you entered, double-click "Project texts" in the project tree under "Languages & Resources".
5. If you are working in multi-lingual projects and want your comment to be translated into other languages, you can set the project languages required in addition to the editing language in the project tree under "Languages & Resources > Project languages".

Creating comment lines

To create comment lines, follow these steps:

1. Open the force table and enter the required addresses.
2. Enter the character string "//" in a free row in the "Name" column. This turns the row into a comment line.
   No comment lines can be created in the other columns.
3. Enter the required comment in the comment line. The entered comment is shown in green.
4. To show all comments you entered, double-click "Project texts" in the project tree under "Languages & Resources".
5. If you are working in multi-lingual projects and want your comment to be translated into other languages, you can set the project languages required in addition to the editing language in the project tree under "Languages & Resources > Project languages".

Deleting comment lines

To delete comment lines, follow these steps:

1. Open a force table containing comment lines.
2. Delete the entire comment with the introductory string "//", if you no longer required this.
3. Alternatively, delete only the introductory string "//". In this case the existing comment is retained and is displayed in the "Comment" column in the force table.

Note

Deleting comment lines

When you delete comment lines the project languages and any existing translations for these comments are also deleted.

11.19.4.8 Monitoring tags in the force table

Introduction to monitoring tags in the force table

Introduction

Use the force table to monitor the tags of the configured input and output modules in the CPU, dependent on the monitoring mode (Page 5238) you have selected. An online connection to the CPU must exist to monitor tags.

Options for monitoring tags

The following options are available for monitoring tags:

- Monitor all
  This command starts the monitoring of all visible tags in the active force table, dependent on the selected monitoring mode:
  - In basic mode, the monitoring mode is set to "permanent" by default.
  - In expanded mode, you can specify defined trigger points for the monitoring of tags.

  Note

  If the monitoring mode is changed while in expanded mode and then a switch is made to basic mode, the monitoring mode set before will also be applied in basic mode.

- Monitor now
  This command starts the monitoring of the visible tags in the active force table immediately and once only.

CPU-specific limitations when monitoring tags

The following CPU-specific differences exist:

- CPU S7-300/400:
  CPUs from this family can only monitor the first 30 characters of a string.
- CPU S7-1200:
  CPUs from this family can monitor a string up to the total size of 254 characters.

Setting the monitoring mode in the force table

Introduction

By selecting the monitoring mode, you specify the trigger point and the duration of tag monitoring in the force table.

Possible monitoring mode (duration of monitoring)

The following selection options are available:

- Permanent
- Once only, at start of scan cycle
- Once only, at end of scan cycle
- Permanently, at start of scan cycle
- Permanently, at end of scan cycle
- Once only, at transition to STOP
- Permanently, at transition to STOP

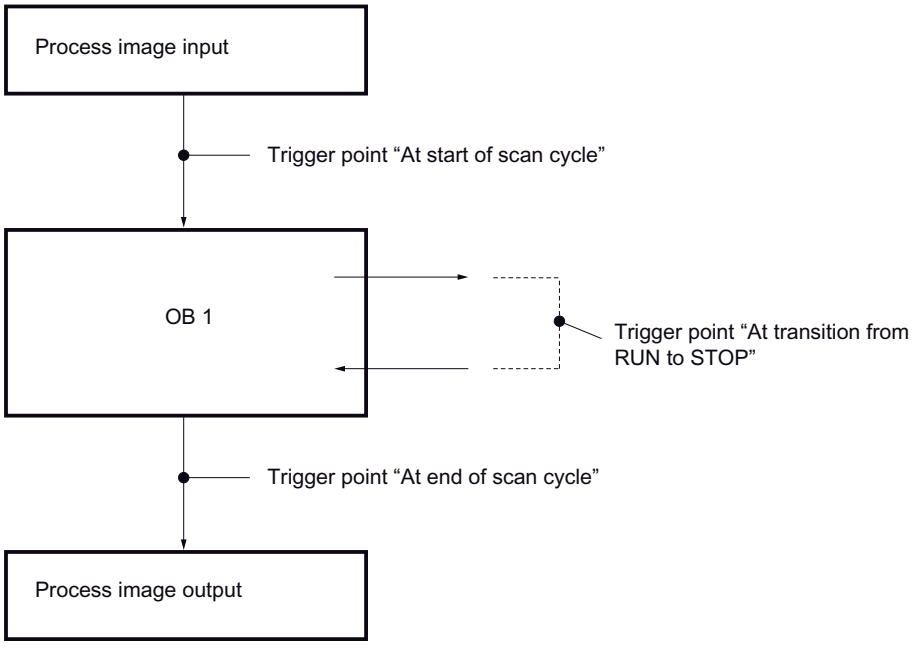
Special features when using "Permanent" mode

The "Permanent" mode is executed as follows for the monitoring of tags: The inputs are monitored at the end of the cycle and the outputs at the start of the cycle.

Selecting the trigger point

The trigger points "Beginning of scan cycle", "End of scan cycle", and "Switch to stop" specify the time at which the tags are to be read from the CPU or updated in the CPU.

The following diagram shows the position of these trigger points:



"Monitor all" command for tags

Introduction

Use the "Monitor all" command to start monitoring the visible tags in the active force table. In basic mode of the force table, the default setting for the monitoring mode is "permanent". In expanded mode, you can specify defined trigger points for the monitoring of tags. In this case, the tags are monitored with reference to the specified trigger points.

Requirements

An online connection to the CPU exists.

Procedure

To execute the "Monitor all" command, follow these steps:

1. Enter the tags to be monitored and the corresponding addresses in the force table.
2. Switch to expanded mode by clicking
   the icon "Show/hide advanced setting columns" in the toolbar.
3. If you want to change the default monitoring mode for a tag, click the appropriate cell in the "Monitor with trigger" column and select the desired monitoring mode from the drop-down list.
4. Click the "Monitor all" icon in the toolbar.

Result

The tags of the active force table will be monitored using the set monitoring mode.

"Monitor now" command for tags

Introduction

The "Monitor now" command starts the monitoring of tags immediately without reference to defined trigger points. The tag values are read out only once and displayed in the force table.

Requirements

An online connection to the CPU exists.

Procedure

To execute the "Monitor now" command, follow these steps:

1. Enter the tags to be monitored and the corresponding addresses in the force table.
2. Click the "Monitor now" icon in the toolbar.

Result

The tags of the active force table are monitored immediately and once only.

11.19.4.9 Forcing tags in the force table

Introduction to forcing tags

Introduction

You can use the force table to assign permanent values to individual tags of the user program. This action is referred to as forcing. Only peripheral inputs and peripheral outputs can be forced.

To use the forcing function, you must have an online connection to the CPU and the utilized CPU must support this functionality.

If you open a force table in the "Watch and force tables" folder below a CPU on which a force job is already running, you will first be prompted to update the display of the forced operand. Forcing or stopping forcing in the open force table is only possible following this command.

Possible applications

By permanently assigning defined values to tags, you can specify defined default settings for your user program and, thus, test the programmed functions. Forcing is possible in basic mode and in expanded mode (Page 5227).

Caution when forcing tags

Before forcing, make sure that you review the safety precautions for this procedure (Page 5243).

DANGER

Prevent personal injury and material damage!

Remember that an incorrect action when executing the "Force" function can:

- Harm persons or pose a health hazard.
- Cause damage to machinery or the entire plant.

Options for forcing tags

The following options are available for forcing tags:

- Force to "0"
  This command forces the selected address in the CPU to the force value "0".
- Force to "1"
  This command forces the selected address in the CPU to the force value "1".
- Force all
  This command starts the forcing of enabled addresses in the active force table or replaces an existing force job without interruption.
- Stop forcing
  This command stops the forcing of all addresses in the active force table.

Constraints when forcing tags

Note the following constraints when forcing:

- Forcing is always dependent on the operand scope of the CPU used.
- In principle, only peripheral inputs and peripheral outputs can be forced.
- If the function "Enable peripheral outputs" is active on your CPU, then forcing is not possible. If desired, deactivate this function in the watch table.

Unique aspects when forcing tags

Note that forcing of tags will overwrite values in the CPU and will continue even after the online connection to the CPU is terminated.

- **Stop forcing**
  Terminating the online connection is not sufficient to stop the forcing operation! To stop forcing, you must select the "Online > Force > Stop forcing" command. Only then will the tags that are visible in the active force table no longer be forced.
- **Stop forcing of individual tags**
  The "Online > Force > Stop forcing" command always applies to all tags displayed in the force table. To stop forcing individual tags, you must clear the check mark for forcing of these tags in the force table and restart forcing using the "Online > Force > Force all" command.

Safety precautions when forcing tags

Safety precautions when forcing tags

Because the forcing function allows you to intervene permanently in the process, observance of the following notices is essential:

DANGER

Prevent personal injury and material damage!

Note that an incorrect action when executing the "Force" function can:

- Harm persons or pose a health hazard.
- Cause damage to machinery or the entire plant.

CAUTION

Prevent personal injury and material damage!

- Before you start the "Force" function, you should ensure that no one else is currently executing this function on the same CPU.
- Forcing can only be stopped by clicking the "Stop forcing" icon or using the "Online > Force > Stop forcing" command. Closing the active force table does **not** stop the forcing!
- Forcing can **not** be undone!
- Review the differences between " modifying tags" (Page 5216) and "forcing tags" (Page 5241).
- If a CPU does not support the "Force" function, the relevant icons cannot be selected.
- If the function "Enable peripheral outputs" is active on your CPU, then forcing is **not** possible on this CPU. If desired, deactivate this function in the watch table.

Updating forced operands

Introduction

If a force job is already running on a CPU, after opening the force table, you first need to make sure that the operands and values currently being forced on the CPU are displayed in the force table.

The command "Online" > "Force" > "Update forced operands" updates all operands and values currently being forced on the CPU in the open force table.

"Force" or "Stop forcing" in the open force table is only possible following this command.

Caution when forcing tags

Before forcing, make sure you are familiar with the safety precautions when forcing tags (Page 5243).

DANGER

Prevent personal injury and material damage!

Remember that an incorrect action when executing the "Force" function can:

- Harm persons or pose a health hazard.
- Cause damage to machinery or the entire plant.

Requirement

- An online connection to the CPU is possible.
- A force job is currently running on the CPU being used.

Procedure

To update the forced operands and values, follow these steps:

1. Open a force table.
2. Establish an online connection to the CPU.
3. Confirm the "Update forced operands" dialog that follows with "Yes".

Result

All forced operands in the open force table are updated with the relevant values. A red "F" is displayed in the first column indicating which operands are being forced.

This enables the "Force all" and "Stop forcing" buttons and you can select these functions.

Note

When forcing, note the following:

- Forcing **cannot** be undone!
- Terminating the online connection does **not** stop the forcing function!
- To stop forcing, the forced address must be visible in the active force table.

Force tags to "0"

Introduction

You can use the force function to assign permanent values to individual tags of a user program.

Caution when forcing tags

Before forcing, you must review the safety precautions when forcing tags (Page 5243).

DANGER

Prevent personal injury and material damage!

Note that an incorrect action when executing the "Force" function can:

- Pose a risk to the life or health of persons.
- Cause damage to machinery or the entire plant.

Requirements

- An online connection to the CPU exists.
- The utilized CPU supports the force function.
- The "Enable peripheral outputs" function is **not** enabled on the CPU on which the tags are to be forced. If desired, deactivate this function in the watch table.

Procedure

To force tags to "0", follow these steps:

1. Open the force table.
2. Enter the desired address in the force table.
3. Select the "Online > Force> Force to 0" command in order to force the selected address with the specified value.
4. Confirm the next dialog with "Yes".

Result

The selected address is forced to "0". The yellow triangle is no longer displayed. A red "F" is displayed in the first column, for example, indicating that the tag is being forced.

Stop forcing

To stop forcing, follow these steps:

1. Open the force table.
2. Select the "Online > Force > Stop forcing" command.
3. Confirm the next dialog with "Yes".

Result

Forcing of the selected values is stopped. The red "F" in the first column is no longer displayed. The yellow triangle reappears behind the check box again to indicate that the address is selected for forcing but is not being forced at the moment.

Note

When forcing, note the following:

- Forcing **cannot** be undone!
- Terminating the online connection does **not** stop the forcing!
- To stop forcing, the forced address must be visible in the active force table.

Force tags to "1"

Introduction

You can use the force function to assign permanent values to individual tags of a user program.

Caution when forcing tags

Before forcing, you must review the safety precautions when forcing tags (Page 5243).

⚠ DANGER

Prevent personal injury and material damage!

Note that an incorrect action when executing the "Force" function can:

- Endanger the life or health of personnel
- Cause damage to machinery or the entire plant.

Requirements

- An online connection to the CPU exists.
- The utilized CPU supports the force function.
- The "Enable peripheral outputs" function is **not** enabled on the CPU on which the tags are to be forced. If desired, deactivate this function in the watch table.

Procedure

To force tags to "1", follow these steps:

1. Open the force table.
2. Enter the desired address in the force table.
3. Select the "Online > Force> Force to 1" command in order to force the selected address with the specified value.
4. Confirm the next dialog with "Yes".

Result

The selected address is forced to "1". The yellow triangle is no longer displayed. A red "F" is displayed in the first column, for example, indicating that the tag is being forced.

Stop forcing

To stop forcing, follow these steps:

1. Open the force table.
2. Select the "Online > Force > Stop forcing" command.
3. Confirm the next dialog with "Yes".

Result

Forcing of the selected values is stopped. The red "F" in the first column is no longer displayed. The yellow triangle reappears behind the check box again to indicate that the address is selected for forcing but is not being forced at the moment.

Note

When forcing, note the following:

- Forcing **cannot** be undone!
- Terminating the online connection does **not** stop the forcing!
- To stop forcing, the forced address must be visible in the active force table.

"Force all" command for tags

Introduction

You can use the force function to assign permanent values to individual tags of a user program.

If forcing is already active, this forcing operation is replaced without interruption by the "Online > Force > Force all" command. Any forced addresses that are not selected will no longer be forced.

Caution when forcing tags

Before forcing, you must review the safety precautions when forcing tags (Page 5243).

DANGER

Prevent personal injury and material damage!

Note that an incorrect action when executing the "Force" function can:

- Pose a risk to the life or health of persons.
- Cause damage to machinery or the entire plant.

Requirements

- An online connection to the CPU exists.
- The utilized CPU supports the force function.
- The "Enable peripheral outputs" function is **not** enabled on the CPU on which the tags are to be forced. If desired, deactivate this function in the watch table.

Procedure

To force tags with the "Online > Force > Force all" command, follow these steps:

1. Open the force table.
2. Enter the desired addresses and force values in the force table.
3. Select the addresses to be forced by selecting the check boxes for forcing in the column after the "Force value".
   A yellow triangle appears behind the selected check box, indicating that the address is selected for forcing but is not being forced at the moment.
4. Select the "Online > Force> Force all" command in order to force the selected addresses with the specified values.
5. Confirm the next dialog with "Yes".

Result

The selected addresses are forced to the specified values. The yellow triangle is no longer displayed. A red "F" is displayed in the first column, for example, indicating that the tag is being forced.

Stop forcing

To stop forcing, follow these steps:

1. Open the force table.
2. Select the "Online > Force > Stop forcing" command.
3. Confirm the next dialog with "Yes".

Result

Forcing of the selected addresses is stopped. The red "F" in the first column is no longer displayed. The yellow triangle reappears behind the check box again to indicate that the address is selected for forcing but is not being forced at the moment.

Note

When forcing, note the following:

- Forcing **cannot** be undone!
- Terminating the online connection does **not** stop the forcing!
- To stop forcing, the forced address must be visible in the active force table.

11.19.4.10 Stop forcing tags

Stop forcing all tags

Introduction

Note the following before you stop forcing tags:

- Stopping forcing **cannot** be undone!
- Terminating the online connection does **not** stop the forcing!
- To stop forcing, the forced address must be visible in the active force table.

Caution when forcing tags

Before forcing, you must review the safety precautions when forcing tags (Page 5243).

DANGER

Prevent personal injury and material damage!

Note that an incorrect action when stopping the "Force" function can:

- Pose a risk to the life or health of persons.
- Cause damage to machinery or the entire plant.

Requirements

- Tags are forced in a force table.
- An online connection to the CPU exists.
- The utilized CPU supports the force function.
- The "Enable peripheral outputs" function is not enabled on the CPU on which the tags are to be forced. If desired, deactivate this function in the watch table.

Procedure

Proceed as follows to stop **forcing all tags** :

1. Open the force table.
2. Select the "Online > Force > Stop forcing" command in order to stop forcing the displayed addresses.
3. Confirm the "Stop forcing" dialog with "Yes".

Result

The forcing of all tags is stopped. The red "F" in the first column is no longer displayed. The yellow triangle reappears behind the check box again to indicate that the address is flagged for forcing but is not being forced at the moment.

Stop forcing individual tags

Introduction

Note the following before you stop forcing tags:

- Stopping forcing **cannot** be undone!
- Terminating the online connection does **not** stop the forcing!
- To stop forcing, the forced address must be visible in the active force table.

Caution when forcing tags

Before forcing, you must review the safety precautions when forcing tags (Page 5225).

DANGER

Prevent personal injury and material damage!

Note that an incorrect action when stopping the "Force" function can:

- Pose a risk to the life or health of persons.
- Cause damage to machinery or the entire plant.

Requirements

- Tags are forced in a force table.
- An online connection to the CPU exists.
- The utilized CPU supports the force function.
- The "Enable peripheral outputs" function is not enabled on the CPU on which the tags are to be forced. If desired, deactivate this function in the watch table.

Procedure

Proceed as follows to stop **forcing individual tags** :

1. Open the force table.
2. Deactivate the check boxes for the addresses that are no longer to be forced.
3. Reselect the "Online > Force" command.

Result

Forcing of the disabled addresses will be stopped. The red "F" in the first column is no longer displayed. The yellow triangle reappears behind the check box again to indicate that the address is flagged for forcing but is not being forced at the moment.

Visualizing processes

12

12.1 Creating screens

12.1.1 Basics

12.1.1.1 Elements and basic settings

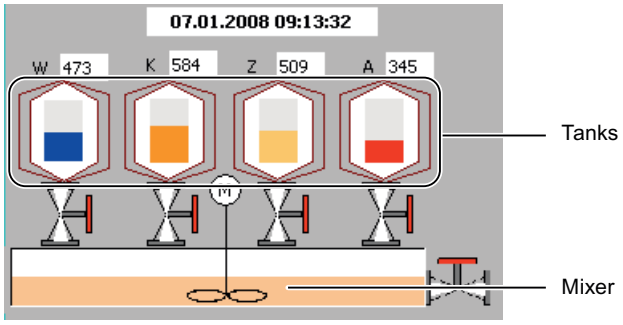
Screen basics (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

In WinCC you create screens that an operator can use to control and monitor machines and plants. When you create your screens, the object templates included support you in visualizing processes, creating images of your plant, and defining process values.

Application example

The figure shows a screen that was created in WinCC. With the help of this screen, the operator can operate and monitor the mixing station of a fruit juice manufacturing system. Fruit juice base is supplied from various tanks to a mixing unit. The screen indicates the fill level of the tanks.



Screen design

Insert an object you need to represent a process into your screen. Configure the object to correspond to the requirements of your process.

A screen may consist of static and dynamic elements.

- Static elements such as text or graphic objects do not change their status in runtime. The tank labels (W, K, Z, A) shown in this example of a mixing unit are static elements.
- Dynamic elements change their status based on the process. Visual current process values as follows:
  - From the memory of the PLC
  - From the memory of the HMI device in the form of alphanumeric displays, trends and bars.

  Input fields on the HMI device are also considered dynamic objects. The fill level values of the tanks in our example of a mixing plant are dynamic objects.

Process values and operator inputs are exchanged between the controller and the HMI device via tags.

Screen properties

The screen layout is determined by the features of the HMI device you are configuring. It corresponds with the layout of the user interface of this device. The screen properties such as the screen resolution, fonts and colors are also determined by the characteristics of the selected HMI device. If the set HMI device has function keys, the screen shows these function keys.

A function key is a key on the HMI device. You can assign one or several functions in WinCC. These functions are triggered when the operator presses the relevant key on the HMI device.

A function key can be assigned global or local functions.

- Global function keys always trigger the same action, regardless of the currently displayed screen.
- Function keys with local assignment trigger different actions, based on the currently displayed screen on the operator station. This assignment applies only to the screen in which you have defined the function key.

Opening screens

In order for the operator to be able to call a screen in runtime, you must integrate each configured screen in the operating process. You have various options of configuring these functions:

- You use the "Screen" editor to configure buttons and function keys for opening other screens.
- You use the "Global Screen" editor to configure globally assigned function keys.

Device-dependent functional scope of screens (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The functions of an HMI device determine the display of the device in WinCC and the scope of functions of the editors.

The following screen properties are determined by the functions of the selected HMI device:

- Device layout
- Screen resolution
- Number of colors
- Fonts
- Objects available

Device layout

The device layout of a screen forms the image of the HMI device in your configuration. The device layout of the screen shows all the function keys available on the HMI device, for example.

Screen resolution

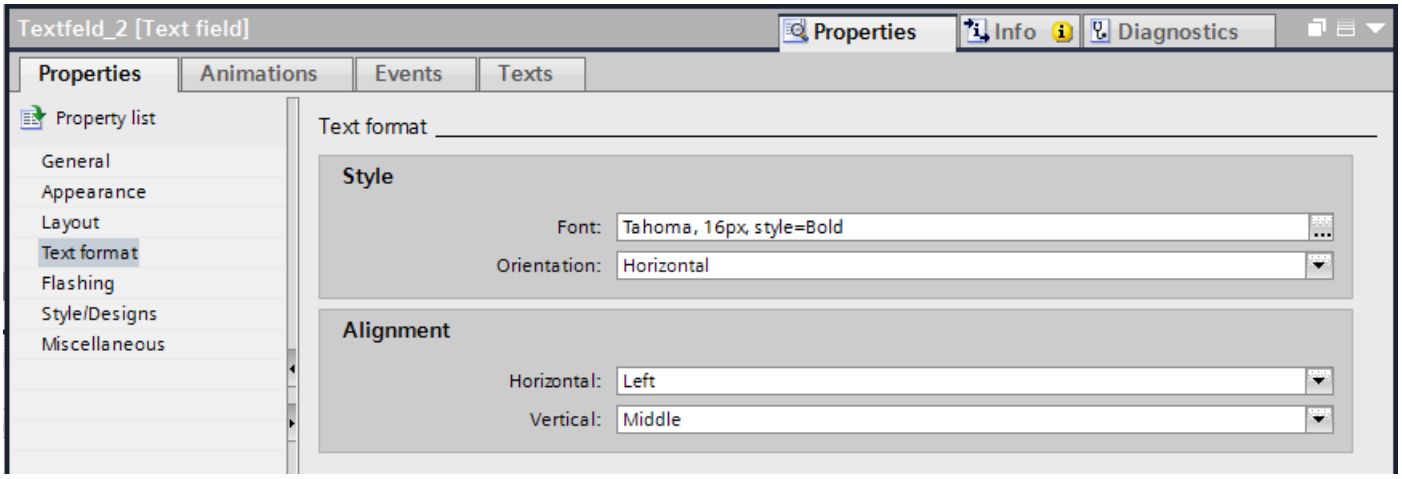
The screen resolution is determined by the different display dimensions of the various operator panels. You can only change the screen resolution if you configure the "WinCC Runtime Advanced" or "WinCC Runtime Professional" HMI device.

Number of colors

You can assign colors to the screen objects. The number of possible colors is determined by the color depth and specific colors supported on the selected HMI device.

Fonts

You can customize the appearance of the texts in all the screen objects that contain static or dynamic text. How to highlight individual texts in a screen. Select the font, font style and size, and set additional effects such as underscoring, for example.



The settings for the text markups such as font style and effects always refer to the entire text of a screen object. That is, you can display the complete title in bold format, but not its individual characters or words, for example.

Objects available

Some of the screen objects can not be configured globally for all HMI devices. These screen objects are not displayed in the "Tools" task card. For a KTP 1000 touch panel unit you can not configure a slider, for example.

Task cards

Introduction

The following task cards are available in the "Screens" editor:

- Toolbox: Display and operating objects
- Animations: Templates for dynamic configuration
- Layout: Aid for customizing the display
- Libraries: Administration of the project library and of the global libraries

Note
WinCC Basic

The "Animations" task card is not available in WinCC Basic.

Toolbox

The "Toolbox" task card contains objects in different panes:

- Basic objects
- Elements
- Controls
- User controls (optional)
- Graphics

You paste objects from the palettes into your screens by drag&drop or a double click. The objects available for selection are determined by the features of the HMI device you are configuring. The following icons are used to change the display mode:

Icon	Meaning
	Displays the objects as a list.
	Displays the objects as a graphic.

Animations

The "Animations" task card contains the possible dynamizations of a screen object in the palettes. You paste the animations to a screen object by drag&drop or a double click from the "Movements", "Display" and "Tag Binding" palettes.

Layout

The "Layout" task card contains the following panes for displaying objects and elements:

- Layers: Serves to manage screen object layers The layers are displayed in a tree view and contain information about the active layer and the visibility of all layers.
- Grid: You specify whether you want to align the objects to a grid or to other objects and set the grid size for a grid.
- Objects out of range: Objects that lie outside the visible area are displayed with name, position and type.

Libraries

The "Libraries" task card show the following libraries in separate panes:

- Project library: The project library is stored together with the project.
- Global library: The global library is stored in a separate file in the specified path on your configuration PC.

Moving the view

Introduction

To display only a section of the entire screen in the work area, use the icon of the "Screens" editor:

Requirement

- A screen is open.
- The view shows only a screen section.

Procedure

To move the view:

1. Click the icon at the bottom right corner of the work area and press the left mouse button. A miniature view of the full screen is shown. An orange frame shows the currently selected area.
2. Hold down the mouse button and drag the frame to the desired area.

Note

The screen is scrolled when you drag a screen object from the visible to a currently hidden section.

Zooming the view

Introduction

To view a small screen section in closer detail, use the zoom tool to magnify the screen in the working area. The maximum zoom amounts to 800%.

You can zoom with the toolbar in the work area or with the "Layout > Zoom" task card.

There are different ways to increase the screen, for example, with the zoom factor or by adapting the work area to the height of the screen.

Requirement

The screen is opened.

Procedure

Proceed as follows to zoom a view with the selection frame:

1. Click the toolbar button.
2. Use the mouse to draw a selection frame in the screen.

After you have released the mouse button, the section enclosed by the selection frame is zoomed to fit the complete work area.

Alternatively, use the slider in the lower right-hand corner of the screen.

Result

The selected screen section is magnified.

12.1.1.2 Working with screens

Steps (Basic Panels, Panels, Comfort Panels, RT Advanced)

Steps

To create screens, you need to take the following initial steps:

- Plan the structure of the process visualization: Number of screens and their layout
  Example: Subprocesses are visualized in separate screens, and merged in a master screen.
- Define your screen navigation control strategies.
- Adapt the templates and the global screen.
  You define objects centrally and assign function keys for example.
- Create the screens. Use the following options of efficient screen creation:
  - Working with libraries
  - Working with layers
  - Working with faceplates

Creating a new screen (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

Create screens to display processes in your system.

Requirement

- The project has been created.
- The Inspector window is open.

Procedure

1. Double-click "Screens > Add New Screen" in the project navigation.
   The screen is generated in the project and appears in your view. The screen properties are shown in the Inspector window.
2. Enter a meaningful name for the screen.
3. Configure the screen properties in the Inspector window:
   - Specify whether and on which template the screen is based.
   - Set the "Background Color" and the "Screen Number."
   - Specify a documenting text under "Tooltip".
   - Specify the layers to be displayed under "Layers" in the engineering system.
   - Select dynamic screen update under "Animations."
   - Select "Events" to define which functions you want to execute in Runtime when you call and exit the screen or at other events.

   Note

   Not all HMI devices support the "Visibility" animation.

Result

You created the screen in your project. You can now paste objects and control elements from the "Tools" task card and assign function keys in further work steps.

Managing screens (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You can move screens within a project to other groups, or copy, rename, and delete them.

Moving screens in a group

1. Select the "Screens" folder in the project tree.
2. Select the "Add group" command from the shortcut menu.
   A folder called "Group_x" is inserted.
3. Select the screen in the project tree.
4. Drag-and-drop the screen to the required group.
   The screen is moved into this group.

Copy screen

1. Select the screen in the project tree.
2. Select the "Copy" command in the shortcut menu to copy the screen to the clipboard.

3. In the project tree, select the screen insert position.
4. Select "Paste" from the shortcut menu to insert the screen.
   A copy of the screen is inserted. A consecutive number is appended to the name of the original in the copy.

Alternatively, press <Ctrl> while you drag the screen to the required position.

Note

If you copy a screen with interconnected template for several devices and projects, the template will also be copied. Any existing matching template is not used. This holds particularly true when you copy the screens with drag-and-drop.

Rename screen

1. Select the screen in the project tree.
2. Select "Rename" from the shortcut menu.
3. Type in a new name.
4. Press <Enter>.

As an option, use the <F2> function key to rename the screen.

Delete screen

1. Select the screen in the project tree.
2. Select "Delete" from the shortcut menu.
   The screen and all its objects are deleted from the current project.

Defining the start screen of the project (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

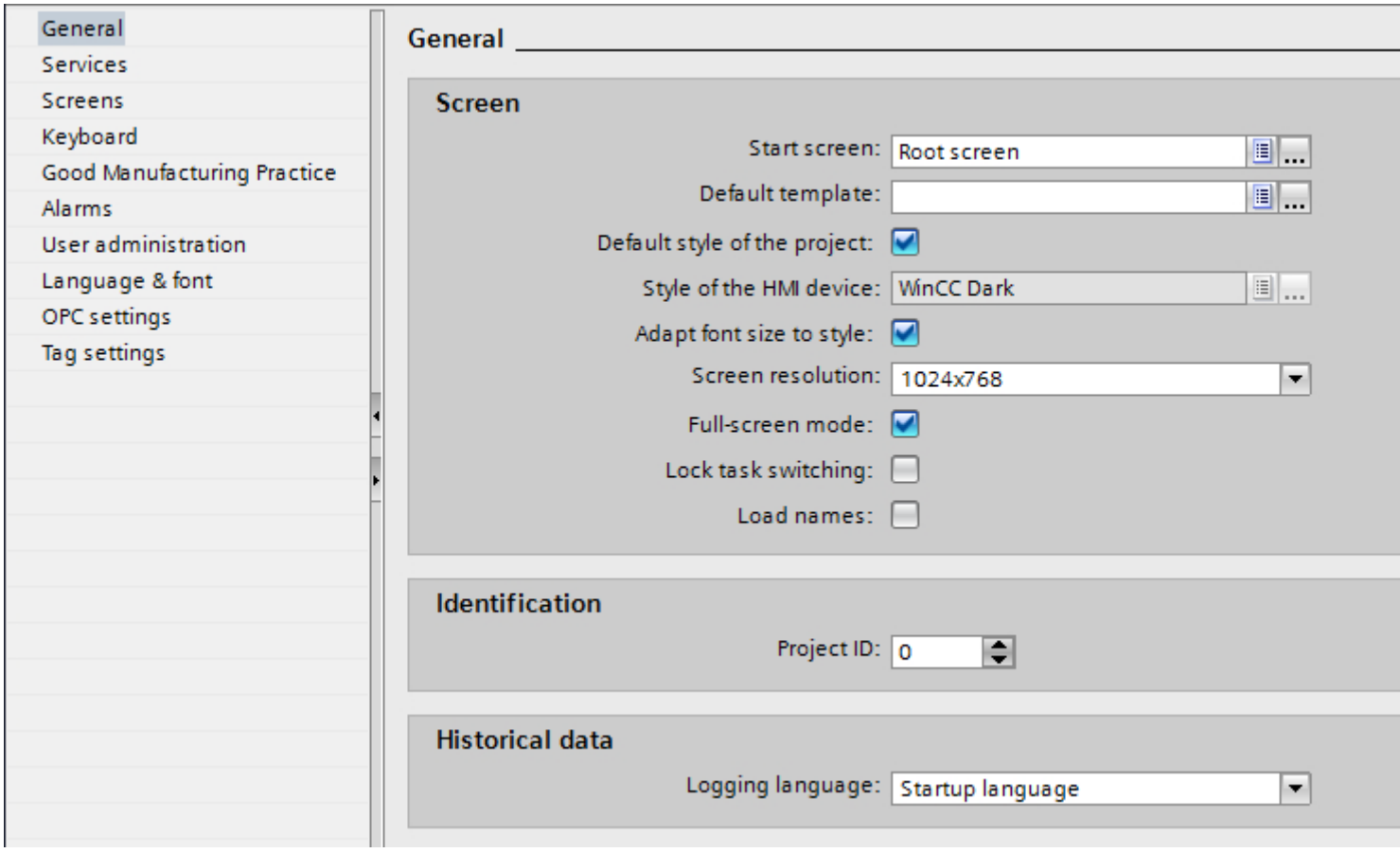
The start screen is the initial screen opened at the start of project in Runtime. You can define a different start screen for each one of the HMI devices. Beginning at this start screen, the operator calls the other screens.

Requirement

The project contains the screen you want to use as the start screen.

Procedure

1. Double-click "Runtime settings > General" in the project tree.



2. Select the desired screen as "Start screen".

   Alternatively select a screen in the project tree and select "Define as start screen" in the shortcut menu.

Result

The start screen opens on the HMI at the start of Runtime.

See also

Steps (Page 5259)

12.1.1.3 Working with templates

Basics on working with templates (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

Configure the objects in a template which are to be displayed in all screens based on this template.

Note the following rules:

- A screen must not be based on a template.
- A screen is only based on one template.
- You can create several templates for one device.
- A template cannot be based on another template.

Objects for a template

You determine functions and objects in the template which are to apply for all screens based on this template:

- Assignment of function keys: You also assign the function keys in the template for HMI devices with function keys. This assignment overwrites a possible global assignment.
- Permanent areas: Some devices support a permanent area for all screens in the top area of the screen. In contrast to the template, the permanent area occupies an area of the screen for itself alone.
- Operator controls: You can paste all screen objects which you also use for a screen into a template.

Application examples

- You want to assign the ActivateScreen" function to a function key in the template. The operator uses this key to switch to another screen in runtime. This configuration applies to all screens that are based on this template.
- A graphic with your company logo can be added to the template. The logo appears on all screens that are based on this template.

Note

If an object from the template has the same position as an object in the screen, the template object is covered.

Global screen (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You define global elements for all screens of an HMI device independently of the used template.

Function keys

For HMI devices with function keys you assign the function keys globally in the "Global Screen" editor. This global assignment applies for all screens of the HMI device.

Proceed as follows to assign function keys locally in screens or templates:

1. Click the function key in your screens or templates.
2. Deactivate "Properties > Properties > General > Use Global Assignment" in the inspection window.

Indicator and control objects for alarms

The "Alarm window" and "Alarm indicator" objects that are available as global objects are configured within the "Global screen" editor.

The "Alarm window" and "Alarm indicator" are always shown in the foreground.

For Comfort Panels you also have the possibility of configuring a "System diagnostic view" in the global screen.

Note

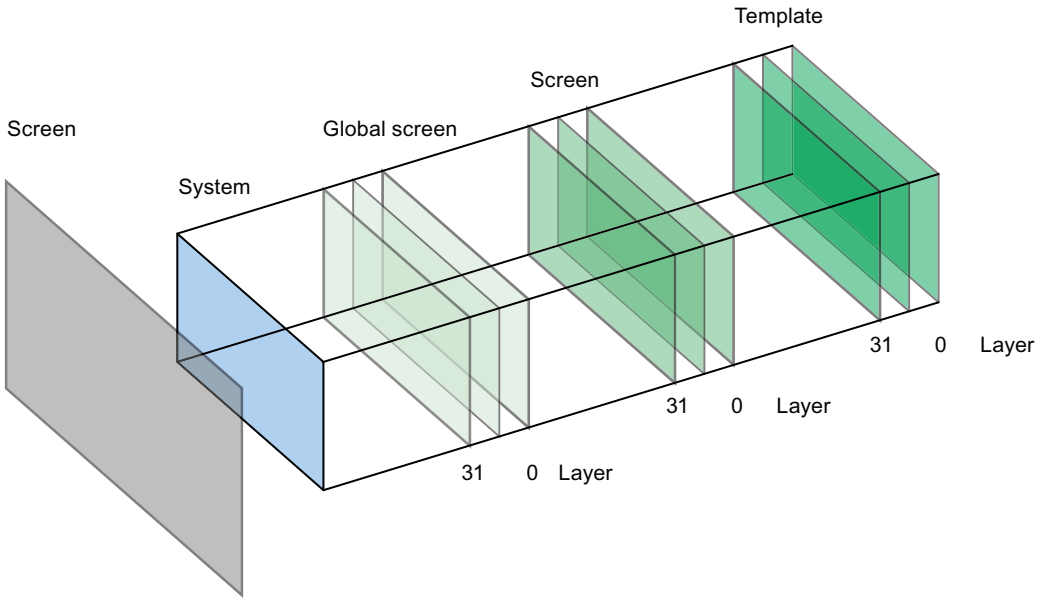
If you have configured a permanent area in a template, do not position the alarm window and alarm indicator in the area of the permanent area. Otherwise, the alarm window and the alarm indicator will not be displayed in Runtime.

However, the permanent area is not visible in the "Global screen" editor.

Order of configuration of screens

The following order applies for the configuration:

- The global screen comes before screens and templates
- Screens come before templates



The system layer is not configurable. This contains

- input dialog boxes
- alarms from the operating system
- the direct keys for touch panels

Creating a new template (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

In a template, you can centrally modify objects and function keys. Changes to an object or of a function key assignment in the template are applied to the object in all the screens which are based on this template.

Note

HMI device dependency

Function keys are not available on all HMI devices.

Requirement

- The project has been created.
- The Inspector window is open.

Procedure

1. Select "Screen management > Templates" in the project tree and then double-click "Add new template".
   The template is created in the project, and appears in your view.
   The properties of the template are displayed in the Inspector window.
2. Define the name of the template under "Properties > Properties > General" in the Inspector window.
3. Specify the layers in the engineering system that are displayed under "Properties >Properties >Layers" in the Inspector window.
4. Add the necessary objects from the "Tools" task card.
5. Configure the function keys.

Result

The template is created in your project.

Managing templates (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You can move, copy, rename, and delete templates within a project in the Project window.

Moving a template into a group

1. Select the templates in the project navigation "Screen management > Templates".
2. Select "Add group" in the shortcut menu.
   A folder called "Group_x" is inserted.
3. Select the template in the project navigation.
4. Drag-and-drop the template to the required group.
   The template is moved to this group.

Copying templates

1. Select the template in the project navigation.
2. Select "Copy" in the shortcut menu.
3. Select the position in the project navigation where you want to paste the template.
4. Select "Paste" from the shortcut menu to insert the template.
   A unique name is assigned automatically to the copy.

Alternatively, you can hold down the <Ctrl> key, and drag the template into position.

Deleting a template

1. In the project navigation, select the template to be deleted.
2. Select "Delete" in the shortcut menu.
   The template, and all its objects are deleted from the current project.

Assigning a template to a screen

1. In the project navigation, select the screen to which you want to assign the template.
2. In the Inspector window, select "Properties > Properties > General".
3. Select the desired template under "Template."
   The selected template and all the objects contained in it are assigned to the screen.

Using a template in the screen (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You use a template in the screen. All the objects configured in the template are also available in the screen.

Requirement

A template has been created.

A screen has been created.

Procedure

Proceed as follows to use a template in a screen:

1. Double click a screen in the project tree. The screen opens in the work area.
2. Open "Properties > Properties > General" in the inspector window.
3. Select a template that is to be applied to the screen under "Template".

Show template in screen

When you edit a screen, you can show an existing template in the screen.

Proceed as follows to show a template in the screen:

1. Activate "Extras > Settings > Visualization > Show template in screens" in the menu.

Result

The screen is based on the selected template. All objects which you have configured in the template are available in the screen. The template is displayed in the screen.

12.1.1.4 Working with styles and style sheets

Basics on working with styles (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

The style editor is a global editor. You define a uniform appearance for display and operating elements in the style editor. In this way, you harmonize the display of objects in Runtime. The style editor offers predefined styles. Additional predefined styles are available for download at Siemens Industry Online Support (http://support.automation.siemens.com/WW/view/en/91174767).

You can choose one of the predefined styles or create your own style. You use the style editor independently of devices and projects.

Views in the style editor

Objects are grouped in the style editor according to their main visual features.

The following groups are available:

- Buttons, lines, and polygon-based objects
- Text-based objects
- Trend-based objects, such as trend view,
- graphic or value-based objects, such as value table.

Style items

You use a style item to define the appearance of a screen object within a style. You can design several objects of the same type differently in one style. You define, for example, different style items for simple buttons and navigation buttons.

You create and manage style items in the style editor.

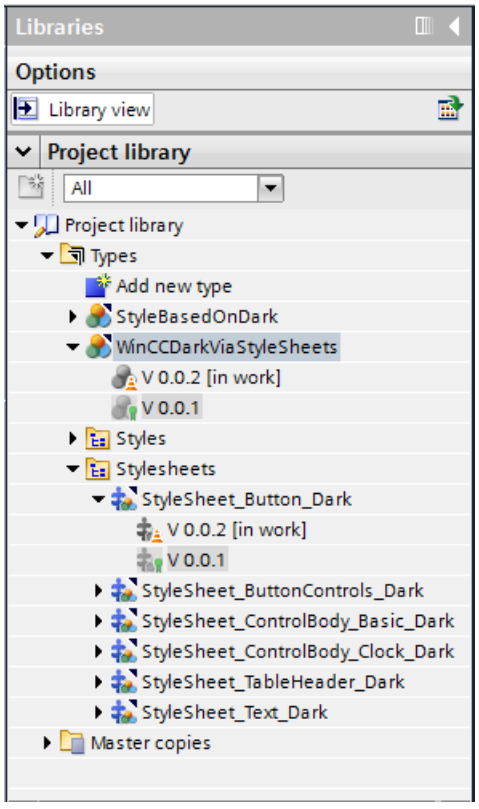
Style sheets

You use style sheets to configure the same properties for a group of objects. In a style sheet you can, for example, define the width and background color of table headers for all table-based objects. Style sheets are divided into predefined types depending on the included properties, for example, buttons, limits, table headers.

You use style sheets to define the design of shared object properties. Some special object properties are not included in style sheets. You use styles to design all objects of a project with visual consistency.

Managing styles and style sheets

You create and edit styles and style sheets in the project library. To get a better overview of types in expansive libraries, you save styles and style sheets in folders with meaningful names. You should also assign meaningful names to the styles and style sheets you create yourself.



Support of styles in faceplates

Styles and style items can be assigned to faceplate instances.

However, there is no direct connection between the faceplate types and styles, so selection and preview of a style item in the faceplate editor is not possible. To assign a style item to a faceplate, enter the name of the style item under "Style item appearance".

The configured style item is used in the instance of a faceplate, provided the name of the style item exists in the style used. Ensure that the name is written correctly.

See also

Displaying predefined styles (Page 5270)

Defining a style (Page 5270)

Managing styles (Page 5272)

Using style items (Page 5275)

Displaying predefined styles (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

In the "Style" editor, you see predefined styles and create your own styles for representing display and operating objects in Runtime.

Open "Styles" editor

1. Double-click "Common data" in the project navigation.
2. Click the "Styles" editor. The editor opens.

Note

Style editor in the project navigation

The style editor is only visible in the project navigation if a device that supports styles, e.g., Comfort Panels, is created in the project.

3. The standard styles are displayed in the work area of the "Style" editor:

- WinCC Light
- WinCC Dark
- WinCC Fresh
- WinCC Wireframe.

Result

The settings for the groups and individual objects are displayed in the work area.

You will assign predefined styles or define your own styles in the next steps.

Predefined styles are write-protected and cannot be changed.

Defining a style (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

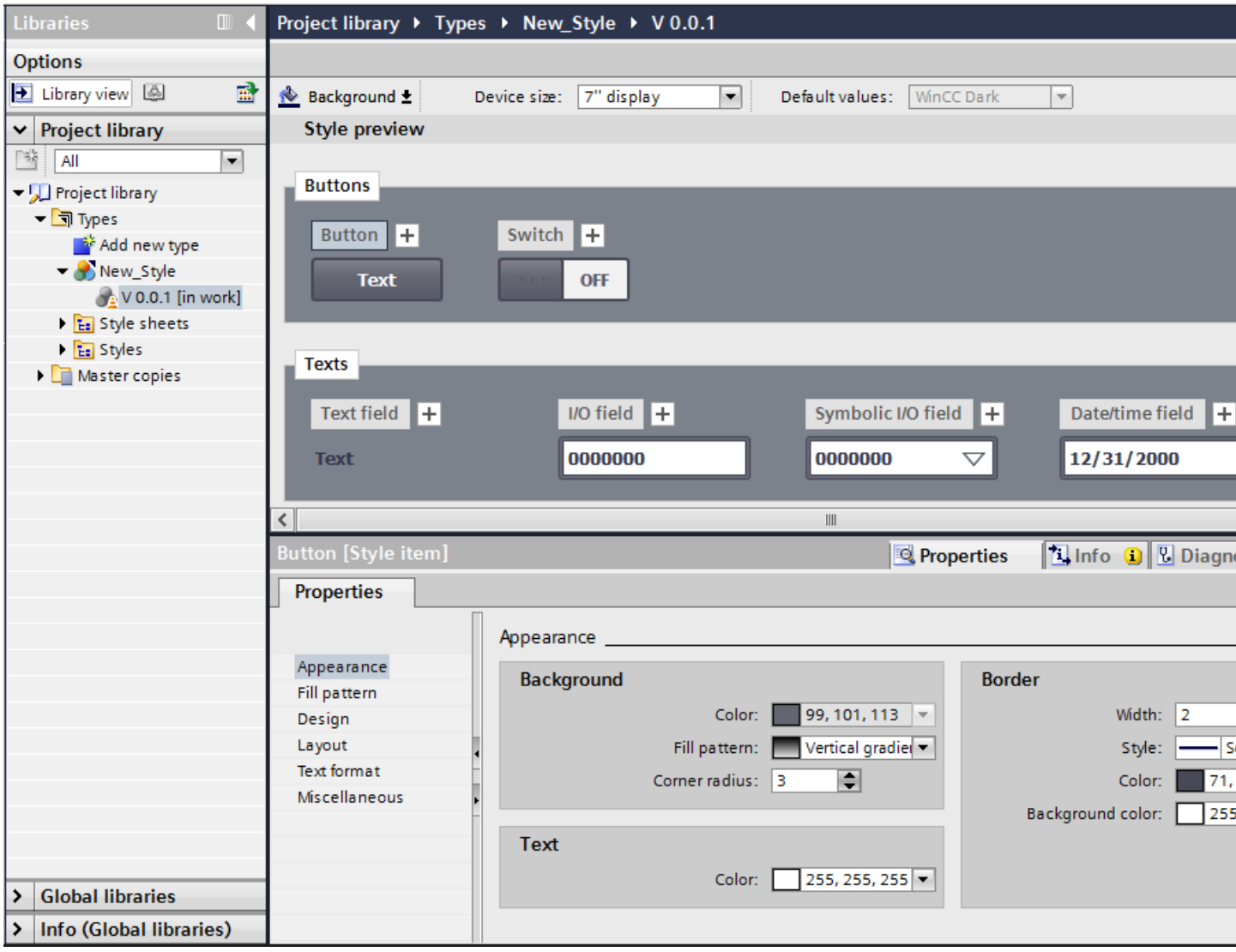
In the "Styles" editor, you see predefined styles. Predefined styles are write-protected and cannot be changed.

In addition, you have the option of defining your own styles and applying them to individual objects, groups, and even projects.

Defining a style

1. Open the "Common data" folder in the project navigation.
2. Open the "Style" editor.

3. Click "Add" in the "Styles" area.
   A new style is created.
   The library view opens. The version of the type is displayed in the "Types" folder. The version 0.0.1 has the status "in progress".
4. In the Inspector window, select the colors, borders and other settings for individual objects.



5. In order to activate the new style, select the current version "in progress" and select "Release version" in the shortcut menu.
   The new style appears in the "Styles" editor.

Result

You have added a new style which can be applied to objects or projects.

Defining a style based on a predefined style

1. Open the "Common data" folder in the project navigation.
2. Open the "Style" editor.
3. Select a predefined style in the work area.
4. Select the "Duplicate type" command in the shortcut menu.
   The "Duplicate type" dialog opens.
5. Overwrite the suggested name with a meaningful name.
6. In the "Comment" column, enter a description of the style, e.g. the intended use.
7. Click "OK" to confirm your input.
8. Select the created type in the project library.
9. Select "Edit new type" in the shortcut menu.
   The library view opens. The version of the type is displayed in the "Types" folder. The version 0.0.2 has the status "in progress".
10. Configure the type as required.
11. Select the "Release version" command from the shortcut menu of the edited version.

Result

They have configured a new style based on the predefined style of WinCC.

Managing styles (Basic Panels, Panels, Comfort Panels, RT Advanced)

Deleting a style

You can delete styles that you have defined yourself.

1. Open the "Common data" folder in the project navigation.
2. Open the "Style" editor.
3. Select the style in the work area that you want to delete.

   Note

   You cannot delete the predefined styles of WinCC.

4. Select the "Delete" command in the shortcut menu.

The style is deleted.

Editing a style

You can edit styles that you have defined yourself.

1. Open the project library.
2. Select the released version of the style that you want to change.

3. Select "Edit type" in the shortcut menu.
   A new version "in progress" is created.
4. Make the required changes in the Inspector window.
5. Select the "Release version" command from the shortcut menu of the edited version to put the changes into effect.
   The changes made take effect.

Determining a standard style for a project

A project can be assigned a standard style to achieve a visual conformity of all display objects in the project.

1. Open the "Common data" folder in the project navigation.
2. Open the "Style" editor.
3. Activate a style in the work area.

The activated style is used as standard style in the project. When you add a device to the project, this device uses the standard style.

Changing the style for a specific device

Within a project, you change the style for a specific device.

1. Open the "Runtime settings" editor of the HMI device.
2. Select a device style under "General".

The selected style is applied to the device.

If the device style is not assigned, the device uses the standard style of the project.

Adjusting the font size to the device size (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You define the size of the device to whose objects you want to apply a style in the style editor. This means you specify device-dependent font sizes in a style. All font sizes are automatically adapted to the device size by selecting a specific reference size.

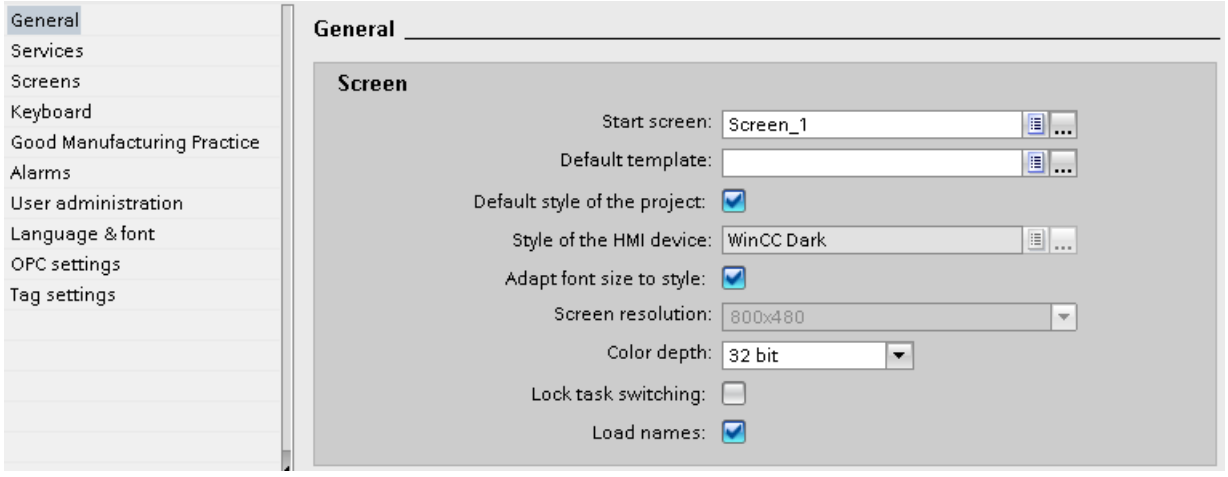
The table below shows the device sizes and the reference font sizes in pixels:

Device size	Font size in pixels
4" display	15 px
7" display	17 px
9" display	15 px
12" display	19 px
15" display	19 px
19" display	17 px

Device size	Font size in pixels
22" display	21 px
WinCC RT Advanced	17 px

Activating automatic adjustment of font sizes

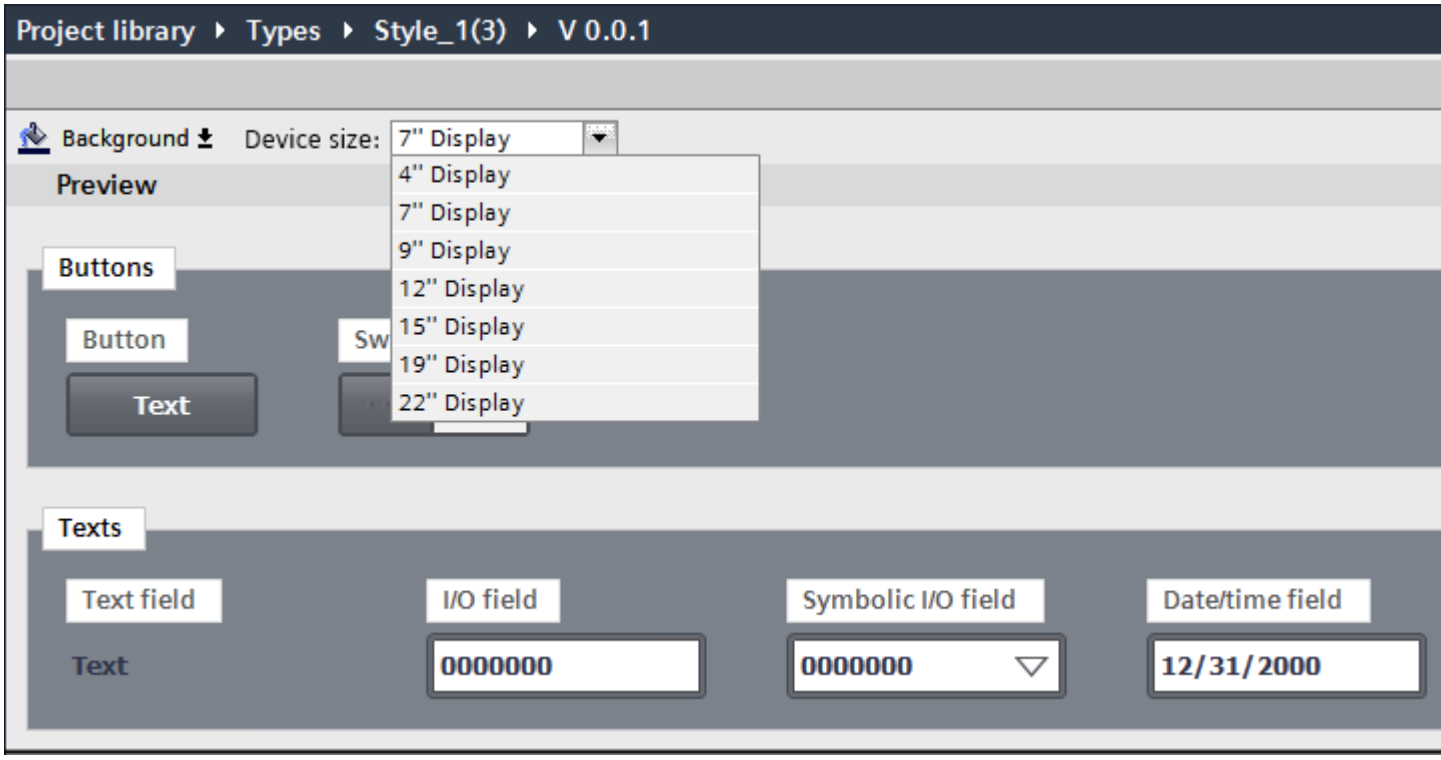
1. Double-click the "Runtime settings" editor in the project window.
2. Click "General".
3. Activate the option "Adjust font size in style" under "General > Screen".



Automatic adjustment of font sizes to the device size was activated in a style.

Adjusting font size in a device style

1. Open the corresponding style for editing in the project library.
2. Select the reference size of the HMI device in the work area.



All font sizes in the style are automatically adjusted to the selected device size.

Using style items (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You use style items to change the style of a screen object without changing the entire device style. You can, for example, define types of buttons with the help of different style items that are to receive the same appearance in one or several devices.

You define style items in the style editor.

Adding a new style item

1. Open the project library.
2. Select the released version of the style that you want to change.
3. Select "Edit type" in the shortcut menu.
   A new version "in progress" is created.
4. Select the object that you want to use in a different style.

5. Click the plus sign next to the object name.



A new style item has been created.
You can also select "Duplicate style item" in the shortcut menu of the object.

6. Make the required changes in the inspector window.
7. Assign a meaningful name to the new style item under "Properties > Miscellaneous > Object > Name".
8. Release the version of the style.

Deleting a style item

You can delete style items that you have defined yourself.

1. Open the project library.
2. Select the style that includes the style item.
3. Open the "in progress" version.
4. Select the style item that you want to delete in the work area.
5. Select the "Delete" command in the shortcut menu.

Note

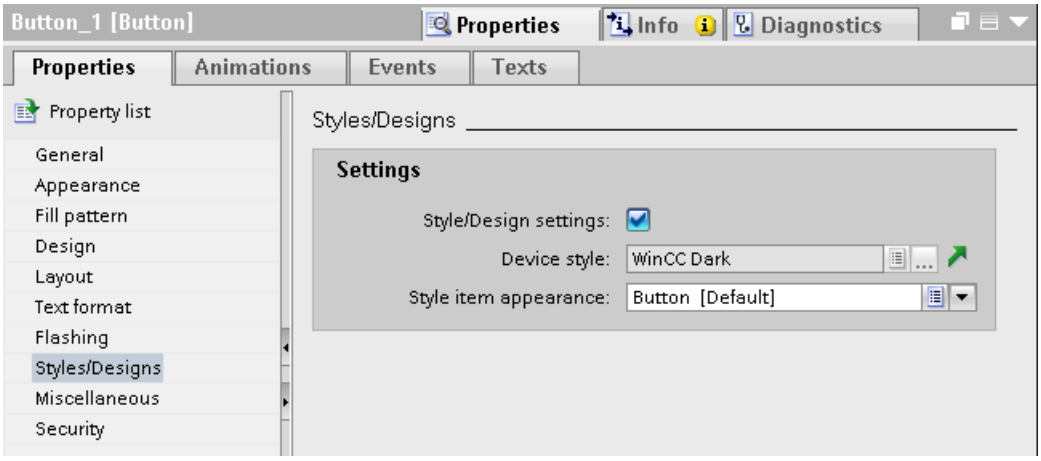
You cannot rename or delete predefined WinCC style items.

The style item is deleted.

Specifying a style item for an object

If you have configured additional style items for an object in the device style, you can use the predefined style item for a screen object.

1. Open "Properties > Styles/Designs > Settings".
2. Enable the option "Use style/design".
3. Select the predefined style item under "Style item appearance".



The object is displayed as selected style item.

Basics on working with style sheets (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

Some screen objects have the same properties which can be configured similarly. These general properties are grouped in style sheets. You can configure several properties for several objects simultaneously and consistently by using style sheets.

Types of style sheets

Style sheets are divided into several predefined categories. Each category contains a specific number of common properties shared by different objects. Select the required category when you create a style sheet. A style sheet can only be applied to objects for which the corresponding category is defined.

The table below shows the available categories and objects that can be configured with them.

Category/Objects	Polylines/ Lines	Text-based objects	Buttons	Value-based objects	Table-based objects	Diagrams
Button	-	-	X	-	X	X
Object body	X	-	-	X	X	X
Diagram body	-	-	-	-	-	X
Focus and selection	-	X	X	-	X	X

Category/Objects	Polylines/ Lines	Text-based objects	Buttons	Value-based objects	Table-based objects	Diagrams
Limits	-	X	-	X	-	-
Labels	-	-	-	X	-	-
Scale	-	-	-	X	-	-
Table body	-	-	-	-	X	X
Table header	-	-	-	-	X	X
Text field	-	X	-	-	-	-

Use of style sheets

You apply a style sheet to a style item when editing a style. The properties configured in the style sheet are transferred to the style item.

To apply a style sheet to a style item, drag the required style sheet from the task card to the style item in the work area using drag-and-drop.

Managing style sheets

Style sheets are library types. You manage style sheets in the project library. Style sheet categories are available in the project library in the "Style sheets > Categories" task card during style editing.

The style sheets that are available for a style sheet type are available in the "Style sheets > Visual attributes" task card. The task card includes predefined style sheets as well as style sheets you have created yourself and released.

Creating a new style sheet (Basic Panels, Panels, Comfort Panels, RT Advanced)

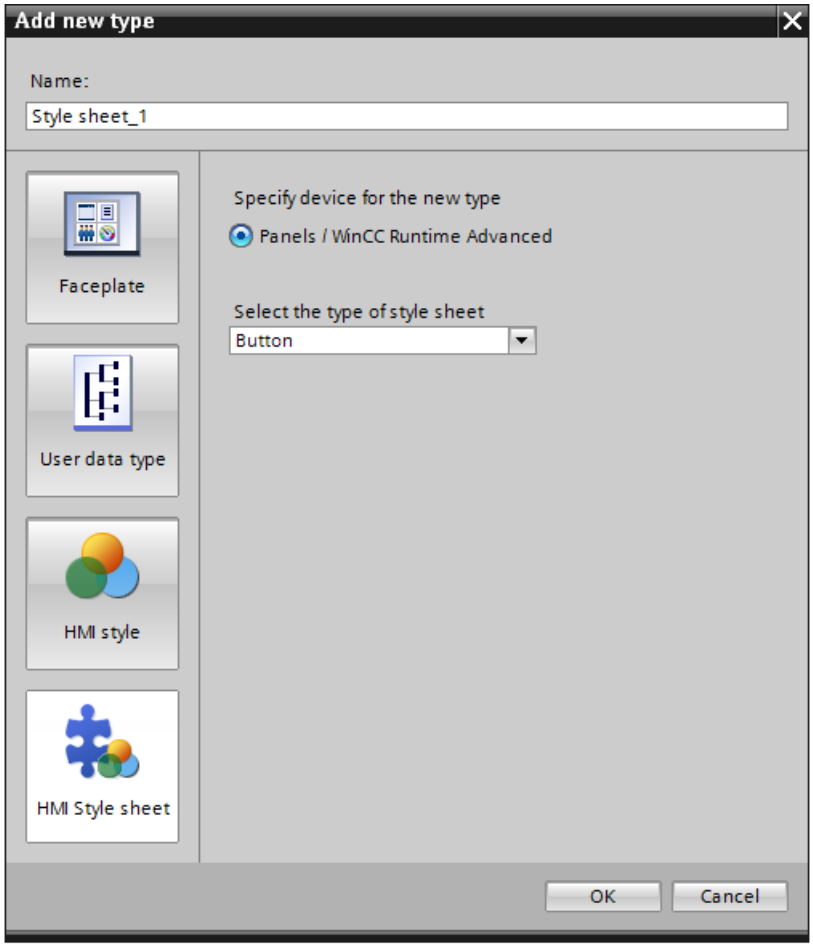
Introduction

You configure the properties for several objects of a similar type in a style sheet.

Create new style sheet

1. Open the project library.
2. Click the "Add new type" button.
   The "Add new type" dialog opens.
3. Select the option "HMI style sheet" in the dialog.

4. Select the type of the style sheet.



5. Assign a meaningful name to the style sheet.
6. Click "OK" to confirm your input.
   The new style sheet was created.

Create new style sheet based on a style

To automatically configure the properties for similar screen objects when editing a style, create a new style sheet based on properties of a style item that have already been configured.

1. Select the style item you want to use to create a style sheet.
2. Select "Create new style sheet" in the shortcut menu.
3. Select the category for the style sheet in the submenu.
   The "Add type" dialog opens.

4. Assign a meaningful name to the style sheet.
5. Click "OK" to confirm.
   The new style sheet based on a style was created and has been added to the "Style sheets > Visual attributes" task card.
   The properties in the style sheet were applied from the selected style item. The properties of the style sheet not included in the selected style have been assigned default values.

Managing style sheets (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You edit, copy and delete the style sheets in the project library.

You can also open, copy, create and delete style sheets with the commands in the shortcut menu of the "Style sheets > Categories" task card.

Edit style sheet

You can edit style sheets that you have created yourself.

1. Open the project library.
2. Select the released version of the style sheet that you want to change.
3. Select "Edit type" in the shortcut menu.
   A new version "in progress" is created.
4. Make the required changes in the Inspector window.
5. Select the "Release version" command from the shortcut menu of the edited version to put the changes into effect.
   The changes made take effect.

Delete style sheet

You can delete style sheets that you have defined yourself.

1. Open the project library.
2. Select the style sheet that you want to delete in the project library.

   Note

   You cannot delete the predefined style sheets of WinCC.

   Note

   You cannot delete the style sheet which includes an "in progress" type. To delete the style sheet, start by releasing all versions of the style sheet.

3. Select the "Delete" command in the shortcut menu.

The style sheet is deleted.

Copy style sheet

1. Open the project library.
2. Select the style sheet that you would like to copy.
3. Select "Duplicate type" in the shortcut menu.
4. Overwrite the suggested name with a meaningful name.
5. In the "Comment" column, enter a description of the style sheet, for example, the intended use.
6. Click "OK" to confirm your input.
   The required style sheet was copied and stored in the same folder.

Using style sheets (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You are using a style sheet while editing a style in the style editor. Style sheets are object-specific and are only applied to style items with which they are compatible.

Only a released version of a style sheet can be applied to a style item.

You have the following options to use a style sheet:

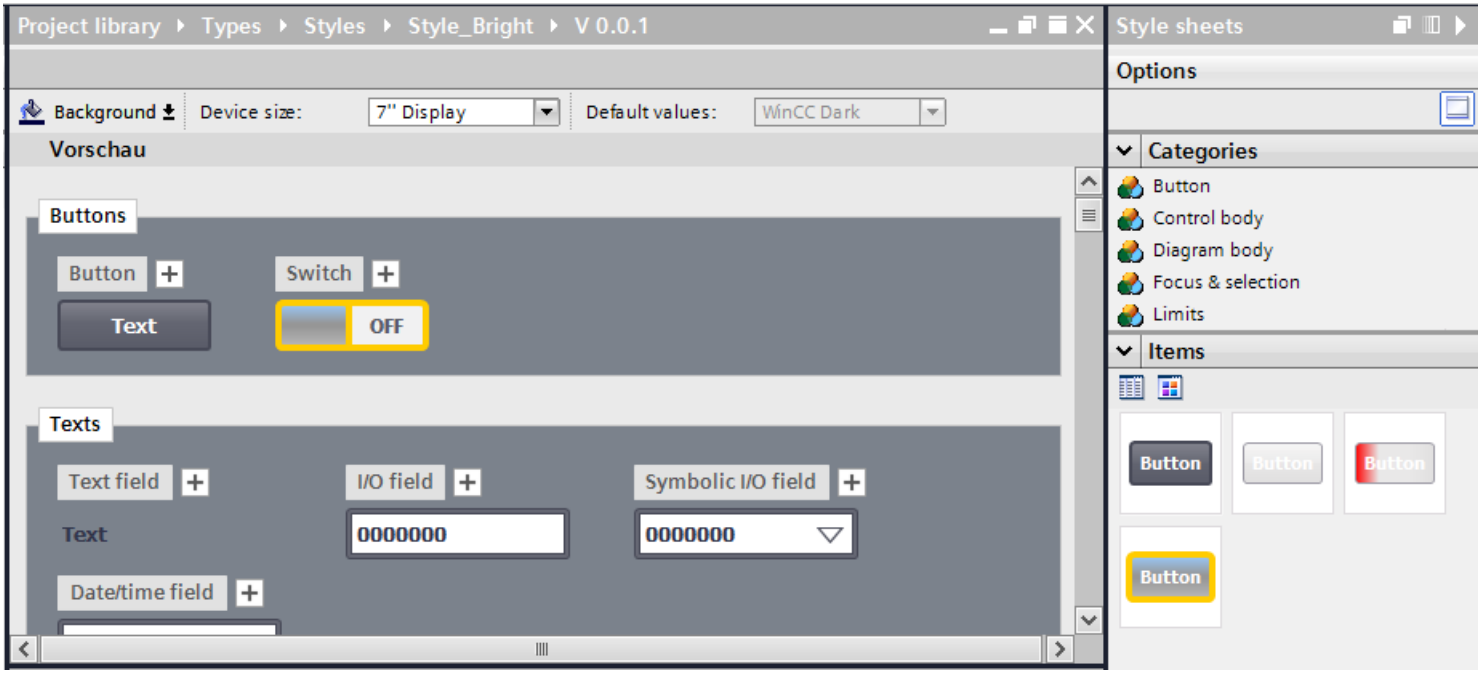
- To design the selected style item, drag the style sheet to a style item in the style editor.
- To design several style items of a group, drag the style sheet to a group of style items.
- To design all style items of the style which are compatible with the style sheet, drag the style sheet to an empty area in the style editor.

Requirement

- You have created a style.
- A version of the style is opened "in progress".

Apply style sheet to style item

1. Select the style sheet category in the "Style sheets > Categories" task card.
2. Select the required style sheet in the "Style sheets > Visual attributes" task card.
3. Drag the selected style sheet to a style item with drag&drop.



When you drop the style sheet on the style item, the style item applies the settings from the style sheet.

Note

There is no connection between the style item and the used style sheet. When you change the style sheet, the changes are not automatically applied in the style item. Apply the style sheet again to make the changes to the style sheet effective in the style item.

Note

Style sheets are not linked with screen objects. Unused or outdated style sheet versions are deleted by cleaning up or versioning the library.

Result

You have applied a style sheet to a style item in a style.

12.1.2 Working with objects and object groups

12.1.2.1 Working with objects

Overview of objects (Basic Panels)

Introduction

Objects are graphics elements which you use to design the screens of your project.

The "Tools" task card contains all objects that can be used for the HMI device. Display the Task Card with menu command "View" by activating the "Task Card" option.

The toolbox contains various palettes, depending on the currently active editor. If the "Screens" editor is open, the toolbox contains the following palettes:

- "Basic objects"
  The basic objects include basic graphic objects such as "Line", "Circle", "Text field" or "Graphic view".
- "Elements"
  The elements include basic operator controls, e.g. "I/O field", "Button" or "Gauge".
- "Controls"
  The controls provide advanced functions. They also represent process operations dynamically, for example Trend view and Recipe view.

- "Graphics"
  Graphics are broken down into subjects in the form of a directory tree structure. The various folders contain the following graphic illustrations:
  - Machine and plant areas
  - Measuring equipment
  - Operator controls
  - Flags
  - Buildings

  You can create links to your own graphic folders. The external graphics are located in these folders and subfolders. They are displayed in the toolbox and incorporated into the project using links.
- "Libraries" task card
  In addition to the display and operating elements, the library objects are available. They are located within the palettes of the "Libraries" task card. A library contains preconfigured objects such as graphics of pipes, pumps, or preconfigured buttons. You can also integrate multiple instances of library objects into your project without having to reconfigure them. The WinCC software package includes libraries, e.g. "HMI Buttons & Switches".
  You can also store customized objects, and faceplates in user libraries.
  Faceplates are objects that you create from existing screen objects, and for which you define the configurable properties.

Note

HMI device dependency

Some of the toolbox objects are either available with restricted functionality, or not at all. This depends on the HMI device you are configuring. Unavailable properties of an object are displayed as deactivated, and cannot be selected.

Basic objects

Symbol	Object	Instructions
	"Line"	-
	"Ellipse"	-
	"Circle"	-
	"Rectangle"	-
	"Text field"	One or more lines of text. The font and layout are adjustable.
	"Graphic view"	Displays graphics from external graphic programs, and inserts OLE objects. The following graphic formats can be used: "*.emf", "*.wmf", "*.dib", "*.bmp", "*.jpg", "*.jpeg", "*.gif" and "*.tif".

Elements

Symbol	Object	Instructions
	"I/O field"	Outputs the values of a tag, and/or writes values to a tag. You can define limits for the tag values shown in the I/O field. To hide the operator input in runtime, configure "Hidden input".
	"Button"	Executes a function list, or a script as configured.
	"Symbolic I/O field"	Outputs the values of a tag, and/or writes values to a tag. A text from a text list is displayed in relation to the tag value.
	"Graphic I/O field"	Outputs the values of a tag, and/or writes values to a tag. A graphic from a graphics list is displayed in relation to the tag value.
	"Date/time field"	Outputs the system date and time, or the time and date from a tag. This allows the operator to enter new values. The display format is adjustable.
	"Bar"	The bar represents a value from the PLC in the form of a scaled bar graph.
	"Switch"	Toggles between two defined states. You can label a switch with text, or a graphic.

Controls

Symbol	Object	Description
	"Alarm view"	Shows currently pending alarms or alarm events from the alarm buffer or alarm log.
	"Trend view"	Represents multiple curves with values from the PLC, or from a log.
	"User view"	Allows an administrator to administer users on the HMI device. It also allows an operator without administrator rights to change his password.
	"HTML Browser"[1)]	Displays HTML pages.
	"Recipe view"	Displays data records, and allows them to be edited.
	"System diagnostics view"	Provides an overview of all devices capable of diagnostics. Displays errors in the plant.

[1)]Available for Basic Panels 2nd Generation.

Options for editing objects

Introduction

Objects are graphics elements which you use to design the screens of your project.

You have the following options for editing objects:

- Copying, pasting or deleting objects using the shortcut menu If you copy an object in a screen and the screen already includes an object of the same name, the name of the object is changed.
- Maintaining the standard size of the objects you are inserting or customizing their size on insertion.
- Moving an object in front of or behind other objects
- Rotating objects
- Mirroring objects
- Defining the tab sequence for objects
- Stamping: Inserting several objects of the same type
- Selecting several objects simultaneously
- Repositioning and resizing multiple objects
- You can assign external graphics to objects, e.g. in the Graphic View.
  You can view only the images you have previously stored in the graphic browser of your WinCC project.
  You can save graphics in the graphic browser:
  - Via drag & drop from the "Graphics" object group to the working area
  - As graphic files in the following formats: *.bmp, *.dib, *.ico, *.emf, *.wmf, *.gif, *.tif, *.jpeg or *.jpg
  - As an OLE object
    You either create a new OLE object or save an existing graphic file as an OLE object. To save an OLE object, an OLE-compatible graphics program must be installed on the configuration computer.

Inserting an object

Introduction

In the "Screens" or "Reports" editor, insert the objects to the "Toolbox" task card. Use the mouse to drag the objects into the work area. You either keep the objects in their original size, or scale them up or down when you paste them.

In addition, you can copy or move objects via the clipboard from one editor to another, for example to transfer a screen object to a report. Alternatively, you can also use the mouse instead of the clipboard for copying and moving:

- Copy: <Ctrl + Drag&Drop>
- Moving: Drag&drop

Note

Basic Panels

The "Reports" editor is not available for Basic Panels.

Requirement

The "Tools" task card is open.

Inserting objects in their standard size

1. In the "Toolbox" task card, select the desired graphic object or the desired graphic in the WinCC graphics folder.
   When you move the cursor across the work area, it turns into a crosshair with an appended object icon.
2. Click the location in the work area where you want to insert the object or graphic.
   The object is inserted with its standard size at the desired position in the work area.

Alternatively, double-click the object in the "Toolbox" task card.

Copying an object

1. Select the desired object.
2. Select "Copy" in the shortcut menu.
3. Click the desired location and select "Paste" in the shortcut menu.

WinCC inserts a copy of the object at the desired location. You can only change the properties that are appropriate for the relevant context.

Example: In the "Screens" editor, you can set the mode for input and output for I/O fields. In the "Reports" editor, the mode is set to "Output".

Original and copy are not interconnected and are configured independently from one another.

Inserting lines

1. Select the desired graphic object in the "Tools" task card.
2. Click on a location in the work area. A line in the standard size is inserted.

Inserting a polygon or polyline

1. Select the desired object "Polyline" or "Polygon" in the "Tools" task card.
2. Click on a location in the work area. This fixes the starting point of the object.
3. Click another location in the work area. A corner point is set.
4. For every additional corner point, click on the corresponding location in the work area.
5. Double-click on a location in the work area. The last corner point is set.
   This fixes all the points of the polygon or polyline.

Note

Basic Panels

The "Polyline" and "Polygon" objects are not available for Basic Panels.

Note

If you want to insert several objects of the same type, use the "Stamp" function. This avoids having to reselect the object in the "Tools" task card every time before inserting it. To do so, select the icon in the toolbar of the "Tools" task card.

Deleting an object

Introduction

You can delete objects individually or with a multiple selection.

Requirement

You have opened the work area containing at least one object.

Procedure

1. Select the object that you want to delete.
   To delete multiple objects, keep the <Shift> key pressed and select the objects to be deleted one after the other. Alternatively, drag and maximize an area around the desired objects with the mouse.
2. Select "Delete" from the shortcut menu.

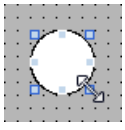
Result

The selected objects are deleted.

Positioning an object

Introduction

When you select an object, it is enclosed by a rectangle with resizing handles. This rectangle is the selection rectangle. The position of an object is defined by the coordinates of the top left corner of the rectangle surrounding the object.



Note

If the position is outside the work area the object is not displayed in Runtime.

Position and align

You have the possibility of having a grid displayed in the work area. You have three options for easier positioning of objects:

- "Snap to grid" When you resposition objects, they are automatically snapped and pasted to the grid. If you hold down the <Alt> key, the object is no longer snapped to the grid.
- "Snap to objects" When you reposition objects, they are displayed with help lines. You can use other objects for orientation during positioning.
- "None": You position the objects in any position.

You activate and deactivate the grid and options as follows:

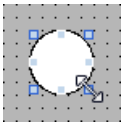
- In the "Options > Settings > Visualization > Screens" menu
- In the "Layout > Grid" task card

Requirement

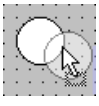
You have opened the work area containing at least one object.

Procedure

1. Select the object you want to move.
   The selected object is framed by a rectangle with resizing handles.



2. Left-click the object and keep the mouse button pressed.
3. Move the mouse pointer onto the new position.
   The contour of the object moves with the mouse and displays the new position for the object.



   The object initially remains at its original position.
4. Now release the mouse button.
   The object is moved into the position indicated by the contour of the selection rectangle.

Alternative procedure

1. In the Inspector window, select "Properties > Properties > Layout".
2. Enter the X and Y values for the position under "Position & Size".

Result

The object appears at its new position.

Resizing objects

Introduction

When you select an object, it is enclosed by a rectangle with handles. You have the following options of resizing an object:

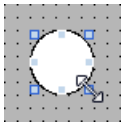
- Drag the handles using the mouse.
- Modify the "Size" property in the Inspector window.

Requirement

You have opened the work area containing at least one object.

Procedure

1. Select the object you want to resize.
   The selection rectangle appears. The following figure shows a selected object:



2. Drag a resizing contact of the rectangle to a new position.
   The size of the object changes.
   - The size of the object is aligned to the grid pattern, provided the "Snap to grid" function is set.
   - Press <ALT> to disable this function while you drag the object.
     In order to scale the object proportionally, keep the <Shift> key pressed while changing the size with the mouse.

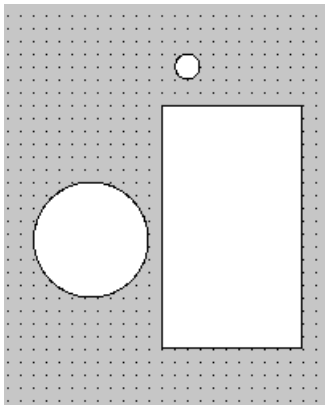
Alternative procedure

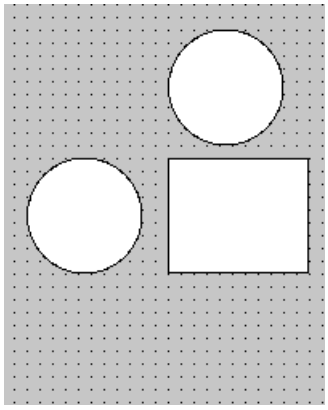
1. In the Inspector window, select "Properties > Properties > Layout".
2. Enter the size of the object under "Position & Size".

Harmonizing the object size

1. Select the objects.
2. Now, click one of the following buttons:  or  or 
   The size of the selected objects is matched to each other.

The following screen shows how the selected objects are adapted to the height of the reference object:





Icon	Description
	Aligns the selected objects to the width of the reference object.
	Aligns the selected objects to the height of the reference object.
	Aligns the selected objects to the width and height of the reference object.

Result

The object now appears with its new size.

Selecting multiple objects

Introduction

Select all objects you want to align with each other or to change global properties. This procedure is called "multiple selection."

The Inspector window shows all the properties of the selected objects.

You now have several options of selecting multiple objects:

- Draw a selection frame around the objects.
- Hold down the <Shift> key, and click the required objects.

Selection frame of a multiple selection

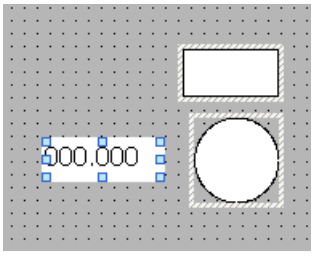
The selection frame surrounds all objects of a multiple selection. The selection frame is comparable with the rectangle that surrounds an individual object.

The selection frame is not visible. When you have made your multiple selection, the following frame is displayed:

- The reference object is indicated by the rectangle around it.
- The other selected objects are indicated by a dashed-line frame.

Specifying a reference object

The reference object is the object upon which the other objects are oriented. The reference object is framed by a rectangle with handles. The following figure shows a reference object with two other selected objects:



You have the following options to specify the reference object:

- Select the objects via multiple selection. The object selected first is then the reference object.
- Draw a selection frame around the objects. The reference object compiled automatically. If you wish to specify a different object within the selection as the reference object, click on the desired object. This action does not cancel your multiple selection.

Requirement

You have opened the work area containing at least two objects.

Selecting multiple objects with a selection frame

1. Position the mouse pointer in the work area close to one of the objects to be selected.
2. Hold down the mouse button, and draw a selection frame around the objects to be selected.

Or:

1. Hold down the <Shift> key.
2. Click the relevant objects, working in succession.
   All the selected objects are identified by frames.
   The object selected first is identified as reference object.

Note

To remove an object from the multiple selection, press <SHIFT>, hold it down and then click the relevant object once again.

Result

Multiple objects are selected. One of those is identified as the reference object. You can now perform the following steps:

- Changing the object properties of all the objects
- Resizing all the objects by the same ratio, by dragging the selection frame to increase or reduce the size
- Moving all the objects in one group
- Aligning the objects to the reference object

Aligning objects

Procedure

1. Select the objects via multiple selection.
2. Specify an object as the reference object.
3. Select the desired command in the toolbar or the shortcut menu - see table below.
   The selected objects will be aligned.

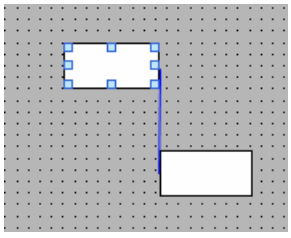
Aligning objects flush

The selected objects will be aligned flush to the reference object.

Icon	Description
	Aligns the selected objects to the left edge of the reference object.
	Aligns the selected objects to the vertical center axis of the reference object.
	Aligns the selected objects to the right edge of the reference object.
	Aligns the selected objects to the upper edge of the reference object.
	Aligns the selected objects to the horizontal center axis of the reference object.
	Aligns the selected objects to the lower edge of the reference object.
	Centers the selected objects to the center points of the reference object.
	Centers the selected objects vertically in the screen.

Snap to object

When you reposition objects, they are displayed with help lines. You can use other objects for orientation during positioning.



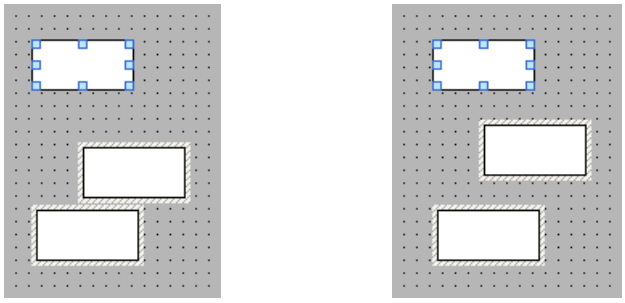
If you are working with the keyboard, press the Alt key. When you move the selected object with the arrow keys, the next anchor point is displayed.

Distributing objects evenly

You need at least three selected objects. A reference object is not required.

1. Select the objects.
2. Click one of the buttons "Distribute horizontally equal" or "Distribute vertically equal". The selected objects are distributed at equal distances.

The following screen shows how you align the vertical spacing of the selected objects:



Icon	Description
	Aligns the horizontal distance between the objects. The position of the objects on the extreme left and right side remains unchanged. All other objects are distributed evenly between them.
	Aligns the vertical distance between the objects. The position of the objects at the extreme top and bottom (right and left) remains unchanged. All other objects are distributed evenly between them.

Moving objects with keyboard shortcuts

Introduction

You can move the objects in the screen by using shortcut keys or select them by type.

Controlling objects with the keyboard

Function	Keyboard shortcut
Moves the selected screen object to the left to the first available vertical alignment position.	<Alt+Shift+Left>
Moves the selected screen object to the right to the first available vertical alignment position.	<Alt+Shift+Right>
Moves the selected screen object up to the first available horizontal alignment position.	<Alt+Shift+Up>
Moves the selected screen object down to the first available horizontal alignment position.	<Alt+Shift+Down>
Selects all screen objects of this type.	<Ctrl+Shift+A>
Selects all objects that support the use styles.	<Ctrl+Shift+D>

Moving an object forwards or backwards

Introduction

You can use the "Order" functions in the shortcut menu of a selected object or in the toolbar to move a selected object in front of or behind other objects within an object layer.

Note

ActiveX controls are always positioned in front of an object layer (.NET property).

Requirement

You have opened a screen which contains a layer with multiple objects.

Procedure

1. Select the object you want to move forward or backward.
2. Select the "Sort" command and one of the following commands from the shortcut menu:

Icon	Description
	Moves the selected object before all the other objects of the same layer
	Moves the selected object to the lowest position in the same layer
	Moves the selected object up by one position
	Moves the selected object down by one position

Alternative procedure

1. Open the "Layers" palette of the "Layout" task card.
2. Navigate to the required object.

3. Hold down the mouse button, and drag the object in the tree topology to the required position in the layer.
4. Now release the mouse button.

Result

The object is moved up or down.

Showing objects outside the screen area

Introduction

If you assign objects to positions that are outside the configurable area, these objects will be hidden. The functions of the "Objects outside the visible area" palette in the "Layout" task card are used to move these objects back into the screen.

Requirement

- You have opened a screen which contains objects that are outside the configurable area.
- The "Layout" task card is open.

Procedure

1. Open the "Layout > Objects outside the area" task card.
   This displays a list of objects that are outside the configurable area.
2. Select the the object which you want to move back into the screen from the list.
3. Select "Move to screen" in the object shortcut menu.

Alternatively open the "Layout > Layer" task card. Objects outside the area are indicated by the  icon. If you click this icon, the object is moved back into the screen.

Result

The objects are moved to the configurable area.

Rotating objects

Introduction

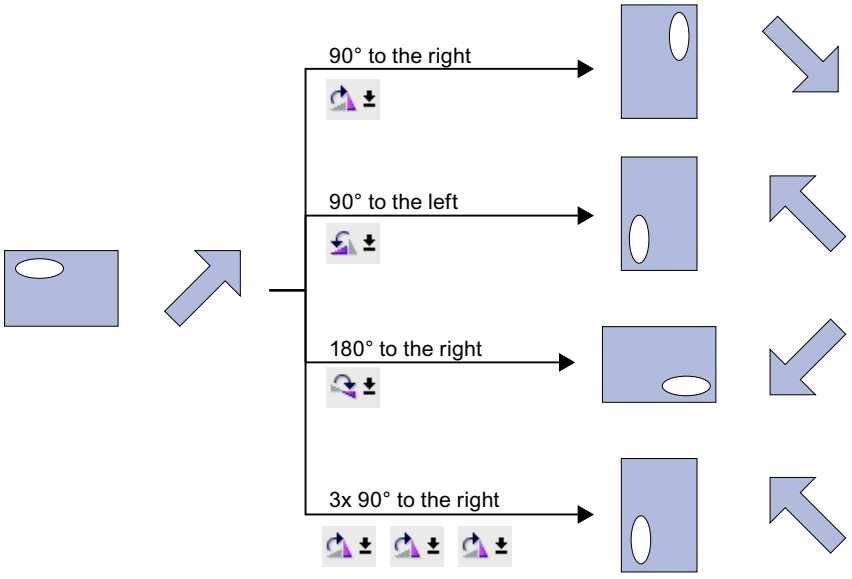
You can rotate a suitable object clockwise or counterclockwise around its center axis in steps of 90°.

Note

Not all the objects can be rotated. Some objects that can be rotated in screens cannot be rotated in reports.

You can also rotate multiple objects using the multiple selection function. Certain WinCC objects such as buttons cannot be rotated.

The alignment of elements in an object will change in a rotated object. The following figure shows how a rectangle and an ellipse behave under the different commands for rotating an object:



Requirement

You have opened the work area containing at least one object.

Procedure

1. Select the object that you want to rotate.
2. Click one of the following toolbar icons:
   , to rotate the object clockwise around its center point. The angle of rotation is 90°.
   , to rotate the object counterclockwise around its center point. The angle of rotation is 90°.
   , to rotate the object clockwise by 180°.

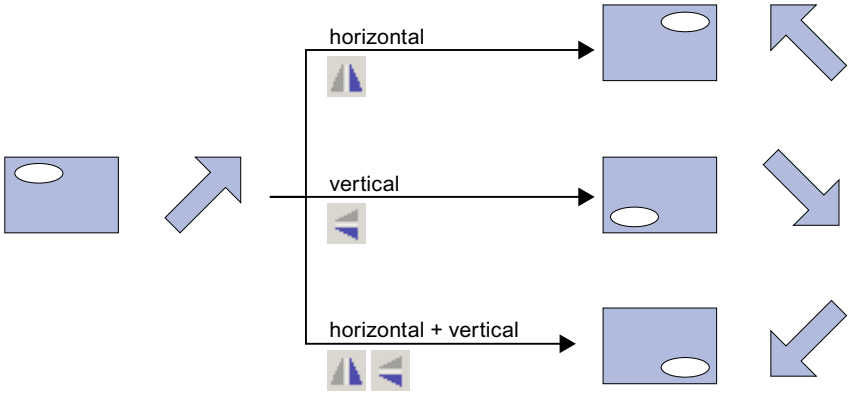
Result

The object is shown at its new angle.

Flipping objects

Introduction

You can flip an object along its vertical or horizontal center axis. The alignment of elements in an object will change when you flip an object. The following figure shows how a rectangle and an ellipse behave under the different commands for flipping an object.



Requirement

You have opened a screen which contains at least one object.

Procedure

1. Select the object that you want to flip.
2. Click the "Flip" command in the shortcut menu and select one of the options displayed:
   - , to flip the selected object along its vertical center axis.
   - , to flip the selected object along its horizontal center axis.

Result

The object is shown at its flipped position.

Designing an object (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You design the border and background of an object.

Requirements

A line has been created in a screen.

Procedure

1. Select the line on your screen.
2. In the Inspector window, select "Properties > Properties > Appearance":
3. Select "Dash" as the style.
4. To display the dashed line in two colors, select the line width "1".
5. Select the setting "Arrow" in the "Line ends" area.

Result

The line is displayed in two colors as a dashed line. The end of the line is an arrow.

Designing the fill pattern

Introduction

WinCC lets you design the background color and the fill pattern of an object. The design options change in the Inspector window depending on object for which you are making the filling pattern.

For certain objects, you can not only define the color, but also a transparent background or a background with a color gradient.

Note

Device-specific nature of the properties

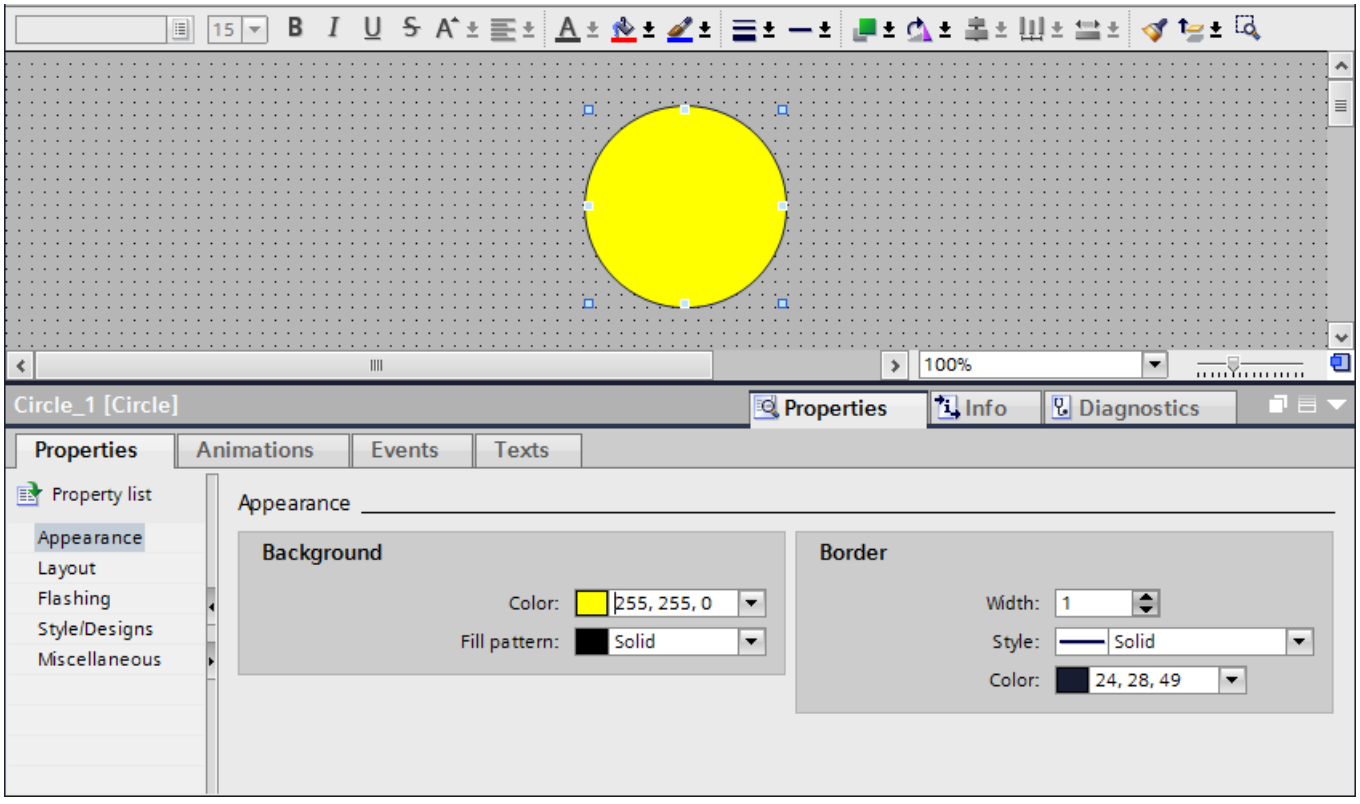
The fill style that is available depends on the object and the HMI device used.

Requirement

The object has been created and selected.

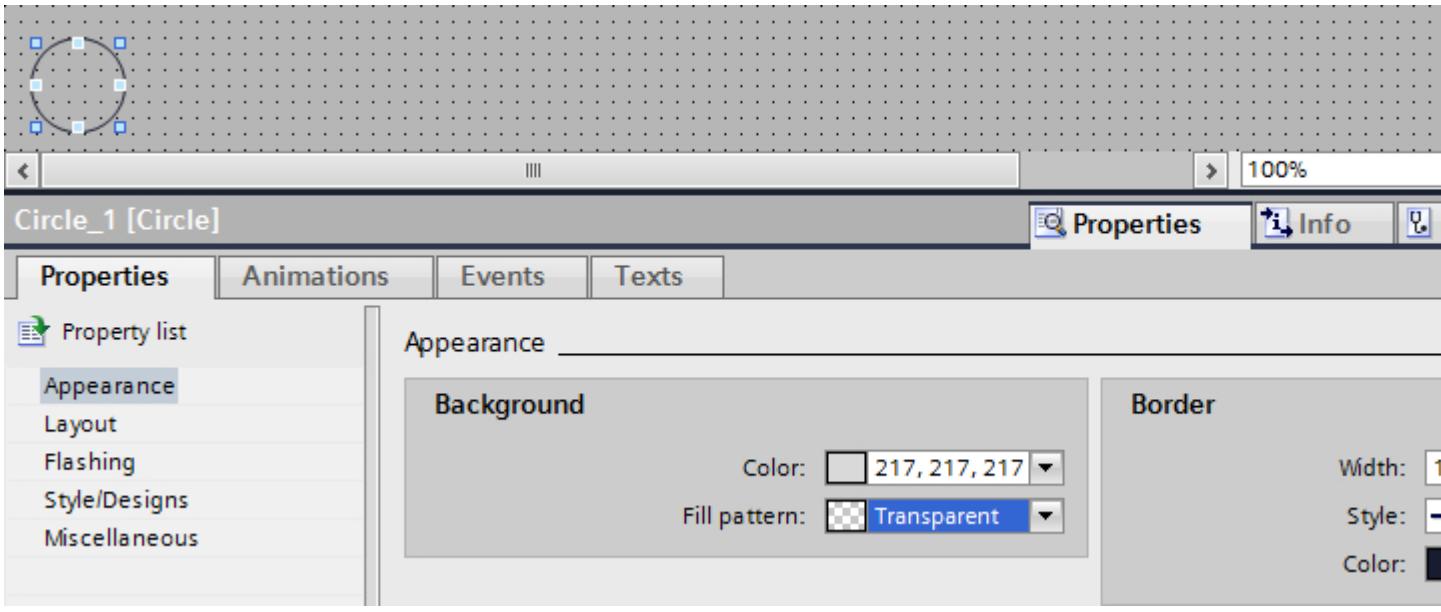
Designing the background color of an object

1. In the Inspector window, select "Properties > Properties > Appearance".
2. Select a color for the background of the object, for example, yellow.
   The object is filled with the selected color.



Designing the fill pattern of an object

1. In the Inspector window, select "Properties > Properties > Appearance".
2. To define a transparent background for the object, select transparent.
   You can find information about designing a fill pattern with a color gradient in the section AUTOHOTSPOT.
   The object is shown as transparent.



You can optionally set the fill pattern in the Inspector window under "Properties > Properties > Fill Pattern".

Specifying scroll bars

You can use the scroll bars for navigation in the table-based objects:

- Use the scroll arrows to scroll line by line either up or down or from left to right in the view.
- Use the scroll box in the scroll bar to scroll slowly to the required position in an object.
- Click on the scroll box to navigate quickly in the object through a large volume of data.

Note

On HMI devices with keys, you can move the scroll bar with arrow keys.

In pop-up screens, use <ALT>+<arrow key> to scroll.

In WinCC version V14 and higher, you can configure different colors for the scroll bars.

Scrolling in controls

Two scroll display options are available under "Runtime settings > Screens > Scrolling in controls":

- Scroll bar with scroll arrows for devices with device version V14 or higher
  The scroll bar is displayed in the screen object with scroll arrows and scroll box.
- Scroll indicator
  A scroll indicator is shown in the screen object

In the Engineering System, you can see the scroll preview in all controls with the exception of HTML browser, PDF display and NC partial program display.

Note

The display of scroll indicators is not supported in the PLC code display or in the pop-up screens.

Showing all data

The scroll bars are used to display content in limited space on the screen. Please remember in configuration that the scroll bars themselves take up space in the control. The width and height of the scroll bars vary depending on the screen resolution.

If you configure a control to display just a few lines, the scroll bars may cover the content in runtime. To avoid hiding the data, take account of the scroll bars in configuration.

Note

When configuring the user display, avoid the setting "Fit object to contents". This setting calculates the object size in runtime automatically and cannot be changed.

Configuring scroll bar colors

You can configure the color of the scroll bar in conjunction with other colors for certain objects.

- The scroll bar arrows and scroll box are shown in the text color of the table
- The scroll bar background color is the alternative color of the table

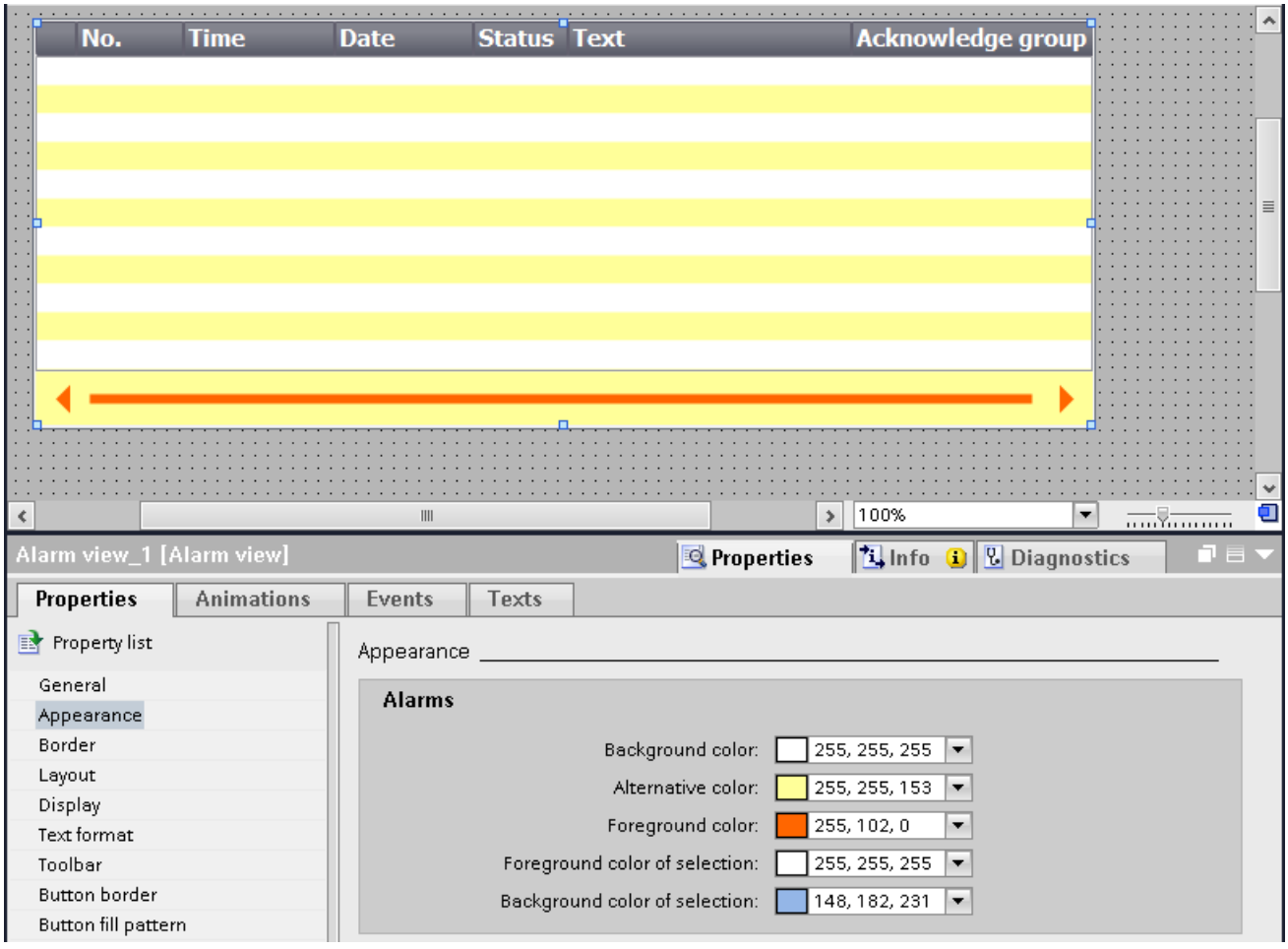
Note

In the objects graphic I/O field, HTML browser, PDF display and file browser, the scroll bar colors cannot be changed.

Example

This example shows the configuration of scroll bar colors for an alarm view.

1. Open "Properties > Properties > Appearance".
2. Select a color for the table foreground, for example "blue" for the scroll arrows and the scroll box in the scroll bar.
3. Select the alternative color, for example "yellow" for the even rows in the table and the background of the scroll bar.
   The scroll bar is displayed in the configured colors.



Formatting graphics in an object

Introduction

WinCC lets you insert graphics into some objects and format them. You can change the size, orientation and distances of a graphic to the object border. You change the properties of the graphic in the Inspector window of the relevant object.

Requirement

- The selected object is in graphic mode.
- The selected object contains at least one graphic.

Resizing a graphic

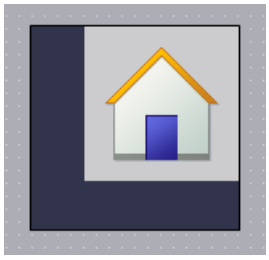
1. In the Inspector window, select "Properties > Properties > Layout".
2. Select "No stretching of picture" or "Stretch picture".
   The contained graphic is displayed in its original size or stretched to the size of the object.

Note

To adjust the object size to the contained graphic, select "Fit object to contents".

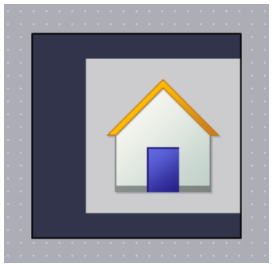
Aligning a graphic horizontally and vertically

1. In the Inspector window, select "Properties > Properties > Layout".
2. To determine the horizontal position of the graphic, select "Right", for example.
3. To determine the vertical position of the graphic, select "Top", for example.
   The graphic is then shown at the top right of the object.



Defining the distance from the object border

1. In the Inspector window, select "Properties > Properties > Layout".
2. Enter the value for the distance to the object border, for example, "20" for the top border.
   The graphic is shown with a distance of 20 pixels to the top border of the object.



Icon	Description
	Sets the distance from the left border of the object.
	Sets the distance from the right border of the object.
	Sets the distance from the top border of the object.
	Sets the distance from the bottom border of the object.

Formatting text in an object

Introduction

Some objects, for example, the I/O field, support text within the object.

You have several options to align text.

Configuring the distance from the object border

1. In the Inspector window, select "Properties > Properties > Layout".
2. Enter the value "5" for the left border, for example.

The text is aligned five pixels from the left border of the object.

Aligning the text position

1. In the Inspector window, select "Properties > Properties > Text format".
2. Select the horizontal alignment, for example, centered.
3. Select the vertical alignment, for example, top.
   The text is shown at the top center in the object.

Defining the text orientation

1. In the Inspector window, select "Properties > Properties > Text format".
2. Select the orientation of the text, for example, vertical, right.
   The text flow is shown from bottom to top.

Designing table-based objects

Introduction

WinCC offers various properties for designing display and operating objects.

You change the properties of table-based objects in the Inspector window of the relevant object.

Designing colors

Select the colors for font, selections and areas under "Properties > Properties > Appearance".

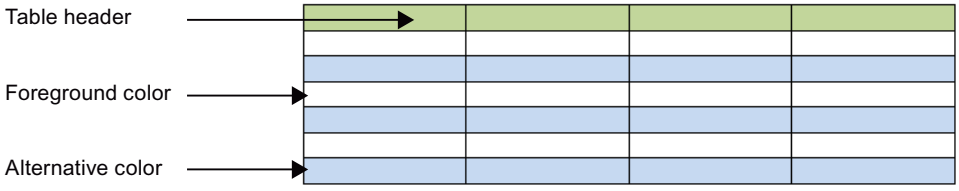


Figure 12-1 Selecting table properties

Designing the border of the table header

1. In the Inspector window, open "Properties > Properties > Table header border".
2. Select "Solid", for example, as the style.
3. Select 5, for example, as the "Width".
4. Select red, for example, as a foreground color.

The border of the table header is shown with a red border and width of 5 pixels. With "Solid" border style, only the foreground color is visible.

Colors and color gradient in the table header

1. Select the table-based object in the screen.
2. In the Inspector window, open "Properties > Properties > Table header fill pattern".
3. Select the fill pattern, for example "Horizontal gradient".
4. Select a background color, for example, blue, under "Gradient".
5. Activate "Gradient 1".
6. Select a color for "Gradient 1", for example, white.
7. Select a "Width" for the color gradient, for example, 12.
8. Activate Gradient 2.
9. Select a color for Gradient 2, for example, yellow.
10. Select a "Width" for the color gradient, for example, 10

The table header is displayed with a color gradient.

Designing a border

Introduction

WinCC offers various properties for designing display and operating objects.

The I/O field example below shows the possible settings for designing borders.

You change the appearance in the Inspector window of the relevant object.

Designing a border

1. Open "Properties > Properties > Appearance".
2. Select the width of the border, e.g. "5".
3. Select the style of the border, e.g. double line.
4. Select the foreground color.
5. Select the background color.



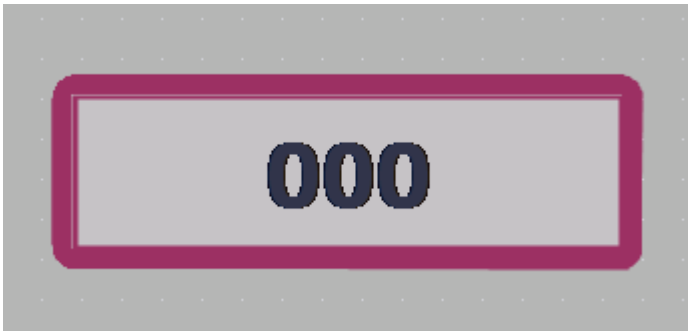
Figure 12-2 "Double line" border style

Border style

The figure below shows a border in "3D style".



The figure below shows a border in "Solid" style.



Note

Device-specific nature of the properties

The border style that is available depends on the object and the HMI device used.

Defining color gradients

Introduction

For the objects in WinCC, color gradients can be specified as background for various surfaces.

Change the category in the Inspector window, depending on which surface you fill with a color gradient. The procedure remains the same.

The following section describes how to configure the color gradient of buttons.

Horizontal color gradient with two colors

1. Select an object with buttons, for example, a button.
2. In the Inspector window, select "Properties > Properties > Fill Pattern".
3. Select "Fill Pattern > Horizontal gradient".
4. Select a background color for the vertical color gradient, for example, orange.
5. Activate "Gradient 1".
6. Select a "Color" for Gradient 1, for example, red.
7. Select a "Width" for Gradient 1, for example, "10".



The background of the button is displayed in orange.

For the horizontal color gradient, Gradient 1 is shown from the left border. Gradient 1 is 10 pixels wide.

Vertical color gradient with three colors

1. Select a button in the screen.
2. In the Inspector window, select "Properties > Properties > Fill Pattern".
3. Select the "Vertical gradient" background.
4. Select a background color under "Gradient", for example, orange.
5. Activate "Gradient 1".
6. Select a color for "Gradient 1", for example, red.
7. Select a "Width" for the color gradient, for example, 8.
8. Activate Gradient 2.
9. Select a color for Gradient 2, for example, yellow.
10. Select a "Width" for the color gradient, for example, 10.

	← Color Gradient 1
	← Background color
	← Color Gradient 2

For the vertical color gradient, Gradient 1 is shown from top to bottom. Gradient 1 is 8 pixels wide.

The background of the button is displayed in orange.

Gradient 2 is shown at the bottom border. Gradient 2 is 10 pixels wide.

Using predefined styles

Introduction

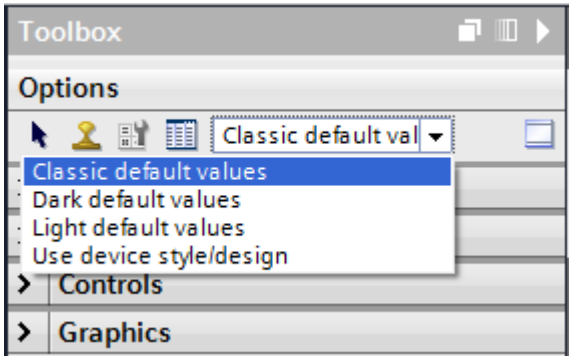
You can assign predefined styles to objects and faceplates in Runtime. You can change the background color of the display and operating elements using predefined styles. In this way, you harmonize the display in Runtime.

Requirement

The "Tools" task card is open.

Setting predefined styles

1. Select the object that you want to insert in the "Toolbox" task card.
2. Select one of the following options in the "Toolbox" task card:
   - "Classic default values" to use the default style
   - "Dark default values" to use the dark color scheme
   - "Light default values" to use the light color scheme
   - "Use device style/design" to use the settings from the current device design



3. Insert the required object in the work area.
   The object is displayed in the selected style.
4. The objects are created in the selected style as long as the style remains activated in the toolbar.
   To return to the default style, select the "Default" option in the toolbar.

Inserting multiple objects of the same type (stamping)

Introduction

WinCC offers the possibility to "stamp" several objects of the same type directly one after the other, i.e. paste without having to reselect the object every time. In addition you have the possibility of multiplying an object that has already been inserted.

Requirement

The "Tools" task card is open.

Inserting several objects of one type

1. Select the object that you want to insert in the "Tools" task card.
2. Click the icon in the toolbar of the "Tools" task card.
   The "Stamp" function is activated.

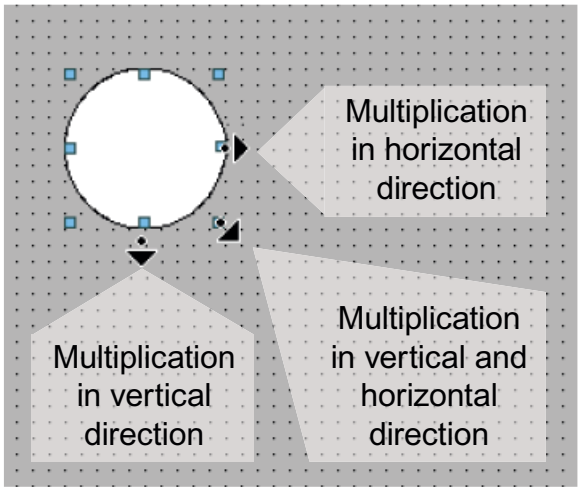
3. To insert the object with its standard size, click the relevant insertion position in the work area.
   To insert the object with another size, position the mouse pointer at the desired location in the work area. Press the left mouse button and drag the object to the required size.
   The object is inserted in the work area as soon as you release the mouse button.
4. Repeat step 3 to insert further objects of the same type.
5. Click the icon again.
   The "Stamp" function is deactivated.

Note

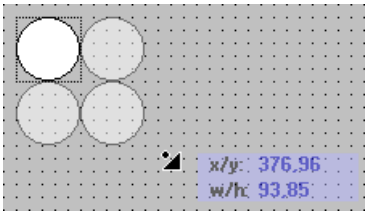
You can copy existing objects using the drag-and-drop +<CTRL> function. The existing object is not moved in this case. You paste a copy of this object into the new position instead.

Inserting and multiplying an object

1. Insert the desired object from the "Tools" task card.
2. Press the <Ctrl> key and position the cursor on one of the handles displayed in the figure shown below.



3. Drag the handles to the right and/or down while keeping the left mouse button pressed.
4. The object is multiplied depending on available space if you keep moving the cursor.



Result

You have pasted and stamped an object in a screen.

Repositioning and resizing multiple objects

Possible modifications

After you have selected multiple objects, you edit them:

- Shift using the mouse
  - To change the absolute position of the marked objects, position the mouse pointer over an object, and shift the multiple selection with the mouse button pressed.
  - To resize all the objects by the same ratio, grab the resizing handles of the reference object.
- Move over the work area with the icons of the toolbar
  - Change the position of the marked objects with respect to each other
  - Align the height and width of the marked objects
- Moving with the shortcut menu commands of the work area
  - Change the position of the marked objects with respect to each other
  - Align the height and width of the marked objects

External graphics

Introduction

You can use graphics created with an external graphic program in WinCC. To use these graphics you store them in the graphic browser of the WinCC project.

You can save graphics in the graphic browser:

- When you drag-and-drop graphics objects from the "Graphics" pane into the work area, these are stored automatically in the graphic browser. The graphic names are numbered in the order of their creation, for example, "Graphic_1." Use the <F2> function key to rename the graphic.
- As a graphic file with the following formats:
  *.bmp, *.ico, *.emf, *.wmf, *.gif, *.tif, *.png, *.jpeg or *.jpg
- As an OLE object that is embedded in WinCC and is linked to an external graphic editor. In the case of an OLE link, you open the external graphic editor from WinCC. The linked object is edited using the graphic editor. An OLE link only works if the external graphic editor is installed on your PC, and supports OLE.

Use of graphics from the graphic browser

Graphics from the graphic browser are used in your screens:

- In a Graphic view
- In a graphics list
- As labeling for a button/function key

Transparent graphics

In WinCC you also use graphics with a transparent background. When a graphic with a transparent background is inserted into a graphic object of WinCC, the transparency is replaced by the background color specified for the graphic object. The selected background color is linked firmly with the graphic. If you use the graphic in another graphic object of WinCC, this object is displayed with the same background color as the graphic object that was configured first. If you want to use the graphic with different background colors, include this graphic in the graphic browser again under a different name. The additional background color is configured when the graphic is used at the corresponding graphic object of WinCC.

Managing graphics

An extensive collection of graphics, icons and symbols is installed with WinCC, for example:

In the Toolbox window of the "Graphic" pane the graphic objects are structured by topic in the "WinCC graphics folder." The link to the WinCC graphics folder cannot be removed, edited or renamed.

The "Graphics" pane is also used to manage the external graphics. The following possibilities are available:

- Creating links to graphics folders
  The external graphic objects in this folder, and in the subfolders, are displayed in the toolbox and are thus integrated in the project.
- Editing folder links
- You open the program required for editing of the external graphic in WinCC.

Managing external graphics

Introduction

External graphics that you want to use in WinCC are managed in the "Screens" editor by using the "Tools" task card in the "Graphics" pane.

Requirement

- The "Screens" editor is open.
- The "Tools" task card is open.
- The graphics are available.
- The graphics have the following formats:
  *.bmp, *.ico, *.emf, *.wmf, *.gif, *.tif, *.jpeg or *.jpg

Creating a folder link

1. Click "My graphics folder."
2. Select "Link" in the shortcut menu.
   The "Create link to folder" dialog is opened. The dialog suggests a name for the folder link.
3. Edit the name as required. Select the path containing the graphic objects.
4. Click "OK" to confirm your input.
   The new folder link is added to the "Graphics" object group. The external graphics that are located in the target folder and in sub-folders are displayed in the toolbox.

Editing folder links

1. Select the folder link to edit.
2. Select the "Edit link..." command from the shortcut menu.
   The "Create link to folder" dialog is opened.
3. Edit the name and path of the folder link as required.
4. Click "OK" to confirm your input.

Renaming the folder link

1. Select the folder link to rename.
2. Select "Rename" from the shortcut menu.
3. Assign a name to the new folder link.

Removing a folder link

1. Select the folder link you want to delete.
2. Select "Remove" in the shortcut menu.

Edit external graphics

1. Select the graphic you want to edit.
2. Select the "Edit graphic" command from the shortcut menu.
   This opens the screen editor associated with the graphic object file.

Editing graphics folders from WinCC

1. Select the graphic you want to edit.
2. Select the "Open parent folder" command from the shortcut menu.
   The Windows Explorer opens.

Storing an external image in the graphics library (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

To display graphics that have been created in an external graphics program in your screens, you will first have to store these graphics in the graphics browser of the WinCC project.

Requirement

- A screen has been created.
- A Graphic View is inserted in the screen.
- The Inspector window of the Graphic view is open.

To store an external graphic in the image browser:

- A graphic is available.

To store an OLE object in the browser:

- An OLE-compatible graphics program is installed on your configuration computer.

Save graphics file

1. Open the Windows Explorer.
2. Select the graphic that you want to store.
3. Drag-and-drop the graphic into the graphic browser.

Creating and saving a new graphic as an OLE object

1. Select the Graphic view on your screen.
2. In the Inspector window, select "Properties > Properties > General":
3. Open the graphic selection list.
4. Click .
5. The "Insert object" dialog box opens.

   Note

   In addition, the dialog "External application running..." will open. The dialog will not close until you exit the external application.

6. From the "Insert object" dialog box, select "New" and an object type. The settings in "Settings > "OLE settings" determine which object types are shown.
7. Click "OK." The associated graphic program is opened.
   When you are finished creating graphics, end the graphic programming software with "File > Close" or "File > Close & return to WinCC."
   The graphic will be stored in the graphic programming software standard format and added to the graphic browser.

Inserting created graphics in WinCC

Note

A new graphic object created as OLE object may not be directly accepted in WinCC when you save it to an external graphics program.

1. Reopen the dialog for inserting a graphic.
2. From the "Insert object" dialog box, select "Create from file."
3. Click the "Browse" button.
4. Navigate to the created graphic and select it.

Saving an existing graphic object as an OLE object

1. In the Inspector window, select "Properties > Properties > General":
2. Open the graphic selection list.
3. Click .
4. The "Insert object" dialog box opens.

Note

In addition, the dialog "External application running..." will open. The dialog will not close until you exit the external application.

5. From the "Insert object" dialog box, select "Create from file."
6. Click the "Browse" button.
7. Use the dialog to help you navigate to the folder in which the graphic file is saved.

Note

To import graphics files, note the following size restrictions:

*.bmp, *.tif, *.emf, *.wmf ≤4 MB

*.jpg, *.jpeg, *.ico, *.gif "*≤1 MB

Result

The image file is now stored in your image browser. It is shown in a screen with a Graphic view , or is added as a list element in an image list.

You can double-click OLE objects in your library to open them for editing in the corresponding graphic editor. When you have finished editing graphics, end the graphic programming software with "File > Close" or "File > Close & return to WinCC." The changes are applied to WinCC.

12.1.2.2 Working with object groups

Basics on groups (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Groups are several objects that are grouped together with the "Group" function. You edit a group in the same way as any other object.

Overview

WinCC offers the following methods for editing multiple objects:

- Multiple selection
- Grouping objects

Editing mode

To edit an individual object in a group, select the object in the "Layout > Layers" task card.

Alternatively select "Group > Edit group" in the object group's shortcut menu.

Hierarchical groups

You add further objects or groups to extend a group. The group is enlarged by the new objects and is structured hierarchically in main and sub-groups. Such hierarchical groups must be broken up in stages. You also break up the group in the same order in which you grouped the objects or groups. It takes exactly the same number of steps to break up these hierarchical groups as it did to create them.

Rectangle surrounding the object

Only one surrounding rectangle is now displayed for the whole group. The surrounding rectangles of all objects are displayed for a multiple selection on the other hand.

Layers

All objects of a group are located in the same layer.

Properties of a group

A group has its own properties, such as authorization. If you set a property at a group, all objects of the group inherit this setting. If the same property is configured differently at an object in the group, the value of the property at the object and not the value in the group applies in Runtime.

Grouping objects (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The "Group" command combines multiple objects to form a group.

You can change the size and position of the group. The following rules apply:

- The system automatically adapts the position coordinates of the grouped objects when you reposition the group. The relative position of the grouped objects to the group is not affected.
- The system automatically adapts the height and width of the grouped objects in proportion to a change of the group size.
- To change the size of the group proportionately, hold down the <Shift> key and drag the rectangle around the object until has the required size.

Note

To create a hierarchical group, organize the individual groups like objects.

Requirement

- You have opened a screen which contains at least two objects.

Creating object groups

1. Select all the objects you want to organize in a group.
2. Select the command "Group > Group" from the shortcut menu.

The objects of the group are displayed with a rectangle around the objects.

Grouping objects within a group

1. Select the group you want to edit.
2. Select the command "Group > Edit group" from the shortcut menu.
   The group that you are editing is highlighted by a red frame.
3. Select the objects of a group that you want to combine into a subgroup.
4. Select the command "Group > Group" from the shortcut menu.

A subgroup with the objects is created.

Adding objects into an existing group

1. Select the group to which you want to add objects.
2. Press the <Shift> key and select the object you want to add to the group.
3. Select the "Group > Add to group" command from the shortcut menu.

The object is added to this group.

Alternative procedure

You can also edit groups in the "Layout" task card. Using drag-and-drop you can also easily edit nested groups in the "Layers" pane.

Result

The selected objects are combined in a group. The multiple selection rectangle becomes the rectangle surroundings the objects in the group. The handles are shown only for the group. The group in in the active layer.

Ungroup (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Select the "Ungroup" command to split a group into the original object fractions.

Requirement

- You have opened a screen that contains a group.

Ungroup

1. Select the group.
2. Select the "Group > Ungroup" command from the shortcut menu.

Ungrouping a group within a group

1. Select the higher-level group.
2. Select the command "Group > Edit group" from the shortcut menu.
   The group that you are editing is highlighted by a red frame.
3. Select the lower-level group.
4. Select the "Group > Ungroup" command from the shortcut menu.

Result

The lower-level group is ungrouped. The objects are assigned to the next higher group.

Alternative procedure

You can also edit groups in the "Layout" task card. Using drag-and-drop you can also easily edit nested groups in the "Layers" pane.

Adding objects to a group (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The "Add objects to group" command is used to add objects to a group, without ungrouping it first.

Requirements

A screen with one group and at least one other object is opened.

Procedure

1. Select the group.
2. Press the <Shift> key and select the object you want to add to the group.
3. Select the "Group > Add to group" command from the shortcut menu.

Result

The group consists of the original objects, and the newly-added objects. The added objects are arranged at the front of the group.

Alternative procedure

You can also edit groups in the "Layout" task card. Using drag&drop you can also easily edit hierarchical groups in the "Layers" palette.

Removing objects from the group (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You use the "Remove objects from group" command to remove individual objects from a group, without ungrouping it first.

You do not have to remove the object from the group to edit an object in a group. You can edit the objects of a group individually.

Requirement

- You have opened a screen that contains a group.

Removing objects from a group

To remove an object from a group:

1. Select the group.
2. Select the command "Group > Edit group" from the shortcut menu.
   The group you want to edit is highlighted by a red frame.
3. Select all objects in the group that you want to remove from the group.
4. Select the "Group > Remove from group" command from the shortcut menu.

The objects are removed from the group.

Note

If there are only two objects in the group, the menu command "Remove from group" is not available.

Deleting objects from a group

To remove an object from the group, and from the screen:

1. Select the group.
2. Select the command "Group > Edit group" from the shortcut menu.
   The group you want to edit is highlighted by a red frame.
3. Select the objects in the group that you want to delete.
4. Select "Delete" from the shortcut menu.

Note

If there are only two objects in the group, the menu command "Delete" is not available.

Alternative procedure

You can also edit groups in the "Layout" task card. Using drag-and-drop you can also easily edit nested groups in the "Layers" pane.

Showing objects outside the screen area (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

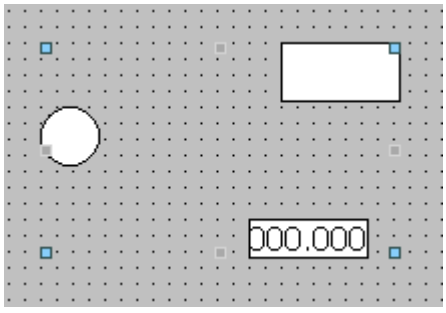
You can edit the objects of a group individually.

Requirement

You have opened a screen that contains a group.

Editing grouped objects

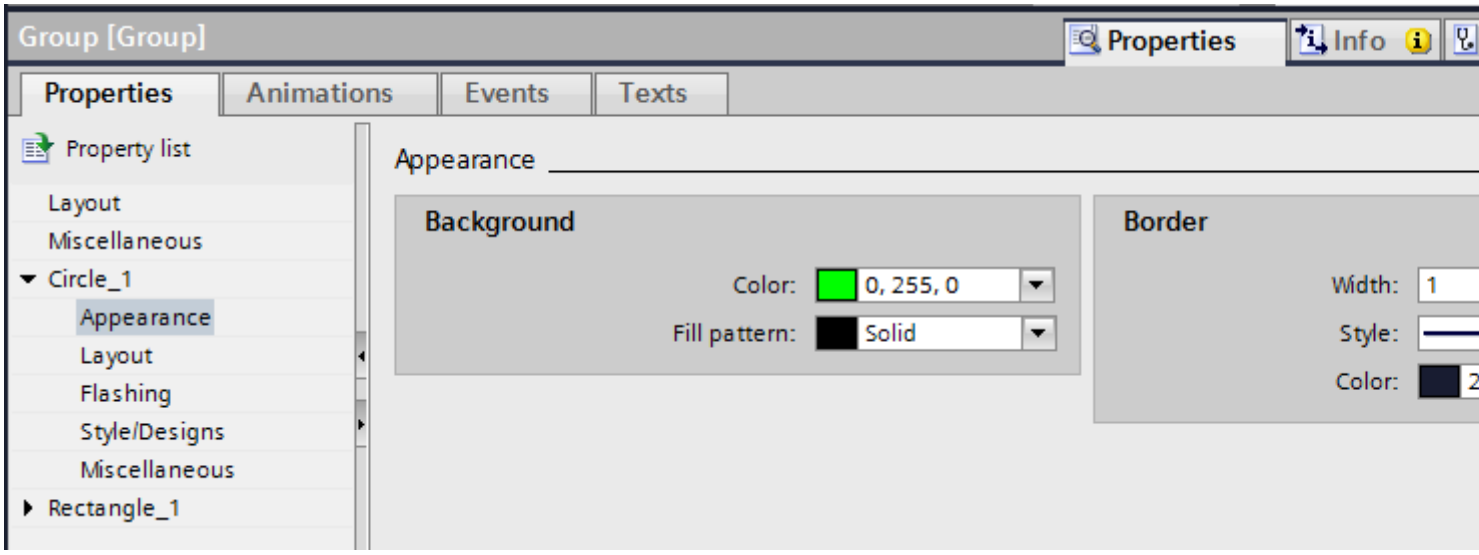
1. Select the group.



   The properties of the group are displayed in the Inspector window.
2. Change the position and size of the grouped objects in "Properties > Properties >Layout."
3. Change the name of the group in "Properties > Properties > Miscellaneous."

Modifying the properties of an object within a group

1. Select the group.
2. Select the object whose properties you want to change in the Inspector window.



   The properties of the object are displayed.
3. Change the object properties.

Note

The dynamization of properties for all objects of the group which have these properties is not possible in a group. You can only dynamize the properties of the objects associated with a group for each object itself.

Result

Although you have edited the object, it is still an element of the group. These changes do not affect the other objects of the group.

12.1.2.3 Configuring the keyboard access

Overview of keyboard access (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

For keyboard units without a mouse, the operator activates control objects using the <Tab> key. You can set up keyboard input to make the process easier to run, and to make sure that the operator enters all the necessary values. If you are using the keyboard, use the <Tab> key to activate the objects in a certain order, and to enter the necessary values.

For HMI devices without key, you can simulate the <Tab> key by configuring the "SimulateSystemKey" system function to a function key.

Operator authorizations and operator control enables

If you configure an object for operator input with the <Tab> key, the object must have both an operator authorization, and an operator control enable.

Editing the tab sequence

The tab sequence is determined automatically when the control objects are created. The numbers of the tab sequence are assigned in the sequence in which the libraries are created.

It makes sense to change the tab sequence in the following cases:

- The operator changes directly to a specific control object.
- The screen requires a specific sequence

Change to the tab sequence mode to change the tab sequence. In this mode, the tab sequence number is displayed at the top left of the control objects. The tab sequence numbers of hidden objects are also shown. The distribution of these numbers is edited using the mouse.

Note

Other functions are not available in the tab sequence mode.

Defining the operator authorization and operator control enable for an object (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

If you configure an object for operator input with the <Tab> key, the object must have both an operator authorization, and an operator control enable.

Requirement

You have opened a screen which contains at least one object.

Procedure

1. Select the object.
2. Select "Properties > Properties > Security" in the Inspector window.
3. Select the operator authorization under "Authorization."
4. Activate the authorization to operate.

Result

The operator can use the <Tab> key in Runtime to select the object.

Setting the tab sequence order

Introduction

All operable objects can be reached in runtime with the <Tab> key. You use the "Tab sequence" command to define the order in which the operator can activate objects in runtime.

Note

You cannot reach objects with the "Output" or "Two states" mode in runtime with the <Tab> key.

You can operate the screen in runtime:

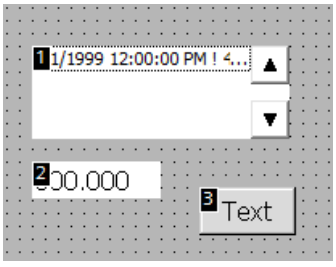
- Using the <Tab> key
- Using the mouse
- Using a configured hotkey

Requirement

- The active screen contains operable objects.
- No object is selected.
- The objects have been enabled for use in runtime, and have operator authorization.

Procedure

1. Select "Edit tab sequence" in the "Edit" menu.
   Tab sequence mode is activated. The tab sequence number is displayed for all objects that can be used. The tab sequence number is also displayed for hidden objects.
2. Use edit the tab sequence mode, click the accessible objects in the order in which you want them to be activated using <Tab> in runtime.
   The following figure shows how the tab sequence is defined in the screen. In runtime, the <Tab> key first activates the alarm view (number 1), then the I/O field (number 2), and then the button (number 3):



3. To exclude a screen object from the tab sequence, press the key combination <Shift+Ctrl> and click on the desired object.
   The tab sequence number is no longer displayed in the screen object. The screen object is now excluded from the tab sequence. The remaining tab sequence numbers are automatically decreased by 1.
4. To reenter an excluded screen object in the tab sequence, repeat step 3.
   The screen object entered as the first object in the tab sequence.

Result

The operator selects the objects in the specified order in Runtime with the <Tab> key.

Note

Changes after upgrades to a new software version

If you have configured a tab sequence in screens with faceplates in an older WinCC version, you should check the tab sequence of these screens after an upgrade to a newer WinCC version. The tab sequence may have been changed in both the screen and the faceplate.

12.1.2.4 Examples

Example: Inserting and configuring a rectangle

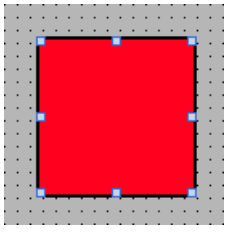
Task

In this example, you insert a rectangle in a screen. You can configure the following properties:

- Name = "MyRectangle"
- Position = (20, 20)
- Size = (100,100)
- Color = red
- Black frame 2 pixels wide

Principle

The rectangle is a closed object which can be filled with a color or pattern. The height and width of a rectangle can be adjusted to allow its horizontal and vertical alignment.



Overview

Carry out the following steps in order to create a rectangle:

- Inserting a rectangle
- Configuring a rectangle

Example: Inserting a rectangle

Task

In this example, you insert and rename a rectangle. Do not use the special characters ?, ", /, \, *, <, > for the name.

Requirement

- A screen is open.
- The Inspector window is open.
- The "Tools" task card is open.

Procedure

1. Click the "Basic objects" palette in the "Tools" task card.
2. Drag the "Rectangle" object into the screen.
3. In the Inspector window, select "Properties > Properties > Miscellaneous".
4. Type in the new name "MyRectangle".

Result

The rectangle is now inserted and named "MyRectangle". The rectangle has the default properties of the "rectangle" object.

Example: Configuring a rectangle

Task

In this example you configure a rectangle:

- Color = red
- Black frame 2 pixels wide
- Position = (20, 20)
- Size = (100,100)

Changing the color of the rectangle

To change the color of the rectangle:

1. Select the rectangle.
2. Define the background color in "Properties > Properties > Appearance > Background > Color" in the Inspector window.
3. Select "Solid" as the fill pattern.
4. Define the color for the border in "Properties > Properties > Appearance > Border > Color" in the Inspector window.
5. Enter the value "2" for "width".
6. Select "Solid" as the "Style".

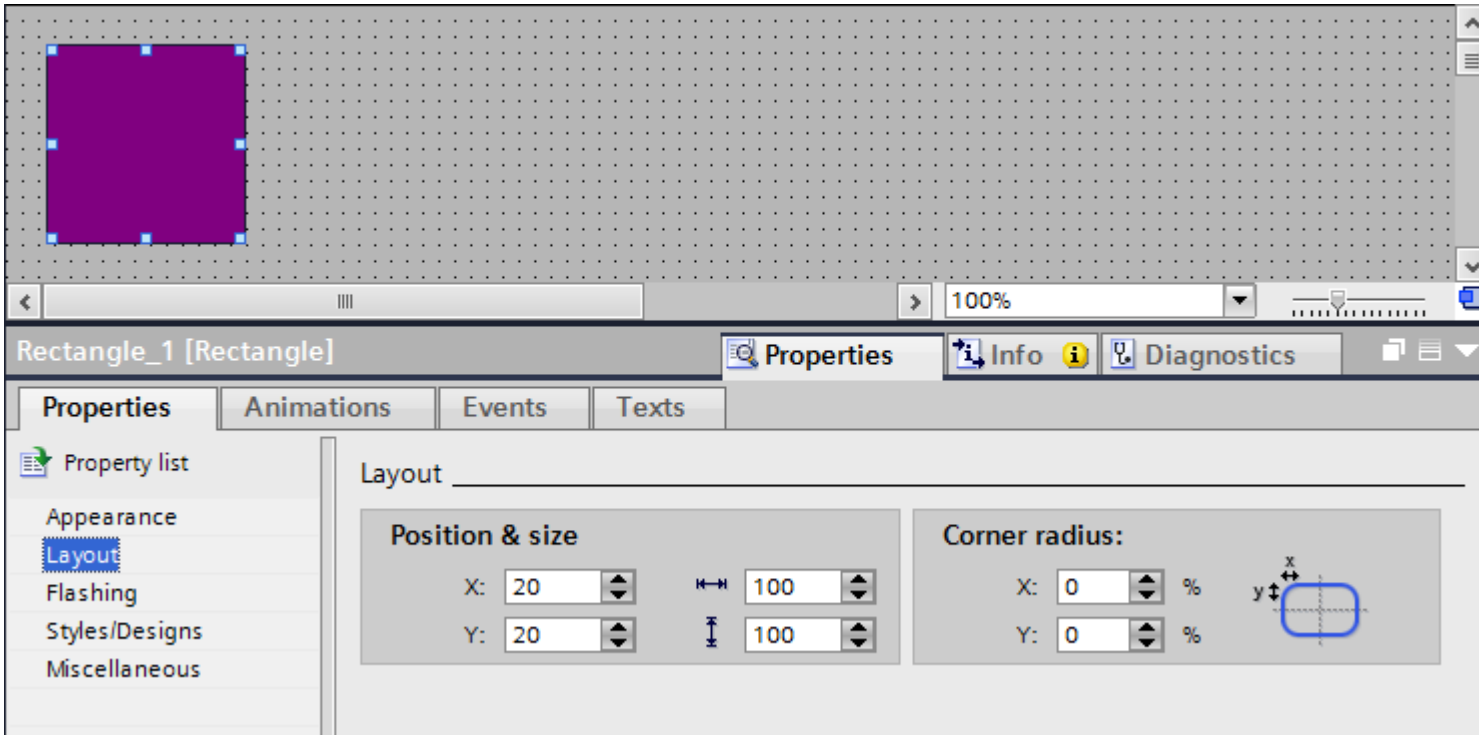
Interim result

The rectangle is red and has a black frame with a width of two pixels.

Repositioning and resizing the rectangle

To change the position and size of the rectangle:

1. Select the rectangle.
2. In the Inspector window, select "Properties > Properties > Layout".



3. Set "20" for the both the X and Y coordinates under "Position & Size".
4. Set "100" for the height and for the width.

Result

The rectangle is positioned at the coordinates (20, 20), and has a width and height of 100 pixels.

12.1.3 Working with text lists and graphics lists

12.1.3.1 Working with text lists

Basics on text lists (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Texts are assigned to the values of a tag in a text list. Assign the text list to a symbolic I/O field for example in the configuration. This supplies the text to be displayed to the object. The text lists are created in the ""Text List" editor. You configure the interface between the text list and a tag at the object that uses the text list.

The selection of objects that can have a text list assigned depends on the Runtime.

Application

The text list is used, for example, to display a drop-down list in a symbolic I/O field.

If the symbolic I/O field is a display field, the associated texts will differ according to the value of the configured tags. If the symbolic I/O field is an input field, the configured tag assumes the associated value when the operator selects the corresponding text in Runtime.

Note

Display of tag values without text

The display of tag values to which no text has been assigned depends on the Runtime:

- The display and operating element remains empty.
- Three asterisks *** are displayed.

Ranges for the text list

Three types are available for the text lists:

- Value/Range
  This setting assigns text entries from the text list to integer values or value ranges of a tag. You can select the number of text entries as needed. The maximum number of entries depends on the HMI device you are using.
  You specify a default value which is shown if the value of the tag lies outside the defined range.
- Bit (0, 1)
  This setting assigns text entries from the text list to two states of a binary tag. You can create a text entry for each state of the binary tag.
- Bit number (0 - 31)
  This setting assigns a text entry from the text list to each bit of a tag. The maximum number of text entries is 32. This form of text list can be used, for example, in a sequence control when processing a sequence in which only one bit of the used tag may be set. You influence the behavior of the bit number (0 - 31) with the set bit of the least significance and a default value.

Multilingual texts

You can configure multiple languages for the texts in a text list. The texts will then be displayed in the set language in Runtime. To this purpose you set the languages in the Project window under "Language support > Project languages."

Configuration steps

The following steps are necessary to display texts in a symbolic I/O field for example:

1. Creating the text list
2. Assignment of the texts to values or value ranges of a text list
3. Assignment of a text list in the display object, for example, to the symbolic I/O field

Creating a text list (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

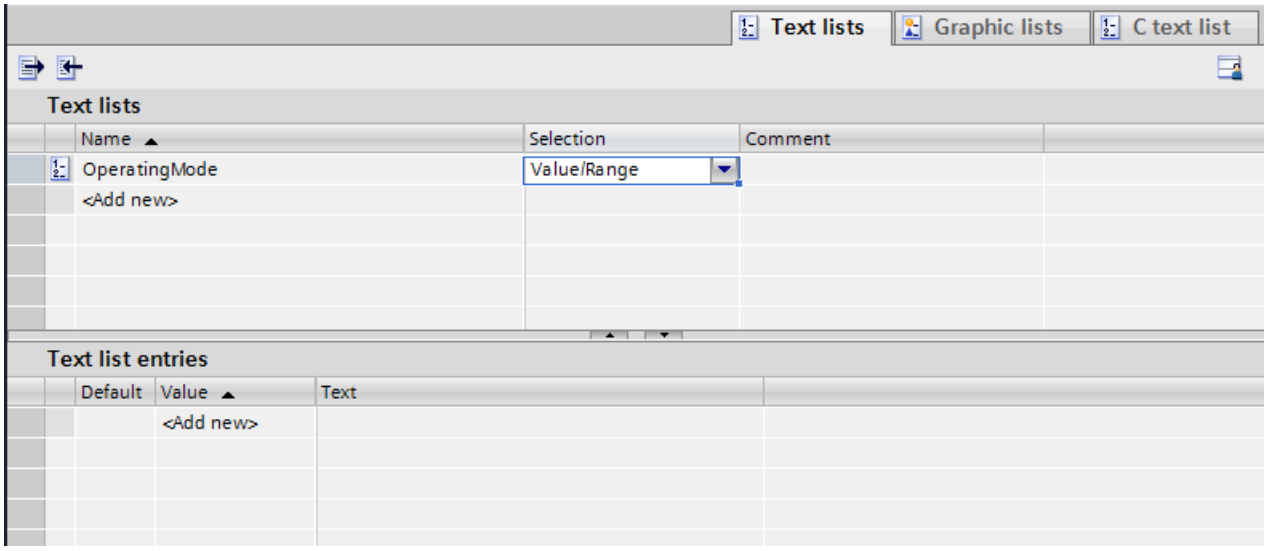
The text list allows you to assign specific texts to values and to output these in Runtime, for example in a symbolic I/O field. The type of symbolic I/O field can be specified, for example as a pure input field.

The following types of list are available:

- Value/Range
- Bit
- Bit Number

Procedure

1. Double-click "Text and graphics lists" in the project window.
2. Open the "Text lists" tab.



3. Click "Add" in the "Text lists" table.
   The Inspector window of the text list is open.
4. Assign a name to the text list that indicates its function.
5. Select the text list type under "Selection":
   - Value/Range: Text from the text list is displayed when the tag has a value that lies within the specified range.
   - Bit (0,1): A text from the text list is displayed when the tag has the value 0. A different text from the text list is displayed when the tag has the value 1.
   - Bit number (0-31): Text from the text list is displayed when the tag has the value of the assigned bit number.
6. Enter a comment for the text list.

Note

You should not use semicolons in the texts in a text list in WinCC Runtime Professional. The semicolon is a control character and is automatically deleted from a text.

Result

A text list is created.

Assigning texts and values to an area text list (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

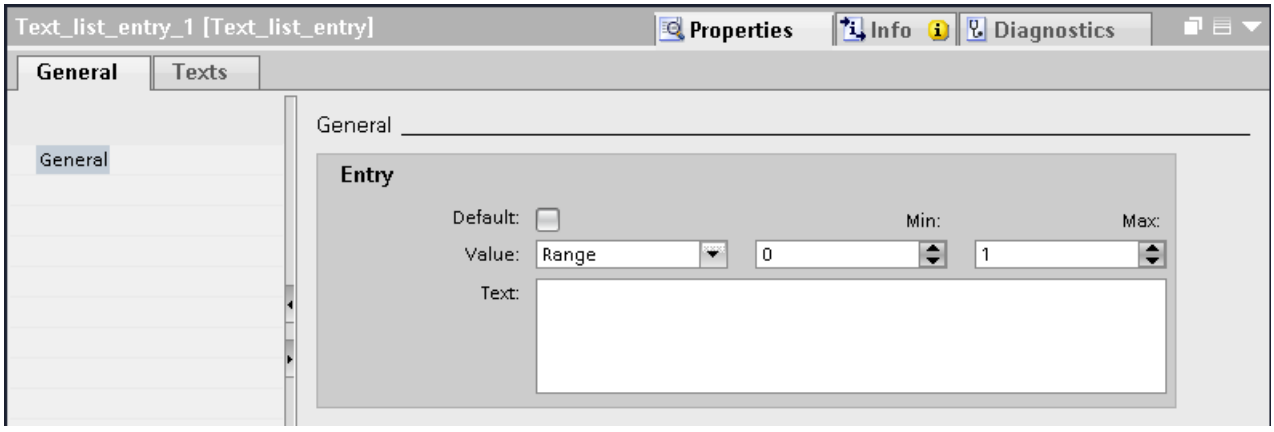
For each area text list you specify which texts are displayed at which value range.

Requirement

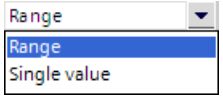
- The "Text and graphics list" editor is open.
- The "Text lists" tab is open.
- An area text list has been created and selected.

Procedure

1. Click "Add" in the "Text list entries" table.
   The Inspector window for this list entry opens.



2. Select the setting "Range" in "Properties > Properties > General > Value" in the Inspector window.



   – Enter the value "1" for "Min" for example.
   – Enter the value "20" for "Max" for example.
   – For "Text", enter the text that is displayed in Runtime if the tag is within the specified value range.

   Note

   Use a maximum of 320 characters for the text.

3. Click "Add" in the "Text list entries" table. A second list entry is created.

4. Select the setting "Range" in "Properties > Properties > General > Value" in the Inspector window.
   - Enter the value "21" for "Min" for example.
   - Enter the value "40" for "Max" for example.
   - For "Text", enter the text that is displayed in Runtime when the tag is within the specified value range.
5. If required, activate the "default entry".
   The entered text is always displayed when the tag has an undefined value. Only one default entry is possible per list.

Result

An area text list is created. Texts have been assigned to the possible value ranges.

Assigning texts and values to a bit text list (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

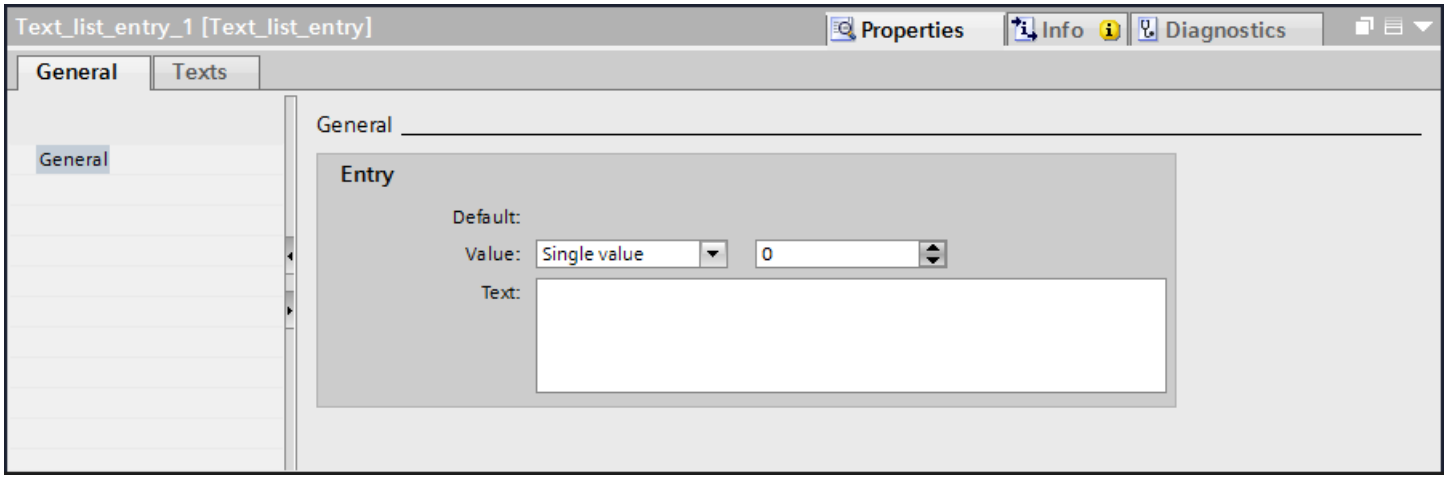
For each text list, you specify which text is displayed at which bit value.

Requirement

- The "Text and graphics list" editor is open.
- The "Text lists" tab is open.
- A bit text list has been created and selected.

Procedure

1. Click "Add" in the "Text list entries" table.
   The Inspector window for this list entry opens.



2. Select the setting "Single value" in "Properties > Properties > General > Value" in the Inspector window.
   - Enter "0" for "Value."
   - Enter the text which is displayed in Runtime under "Text" if the bit tag is set to "0".
3. Click "Add" in the "Text list entries" table. A second list entry is created.
4. Select the setting "Single value" in "Properties > Properties > General > Value" in the Inspector window.
   - Enter "1" under "Value."
   - Enter the text which is to be displayed in Runtime under "Text" if the bit tag is set to "1".

Note

Use a maximum of 320 characters for the text.

Use a maximum of 255 characters and no semicolons for the text in WinCC Runtime Professional.

Result

A bit text list is created. Texts that appear in Runtime are assigned to the possible values "0" and "1".

Assigning texts and values to a bit number text list (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

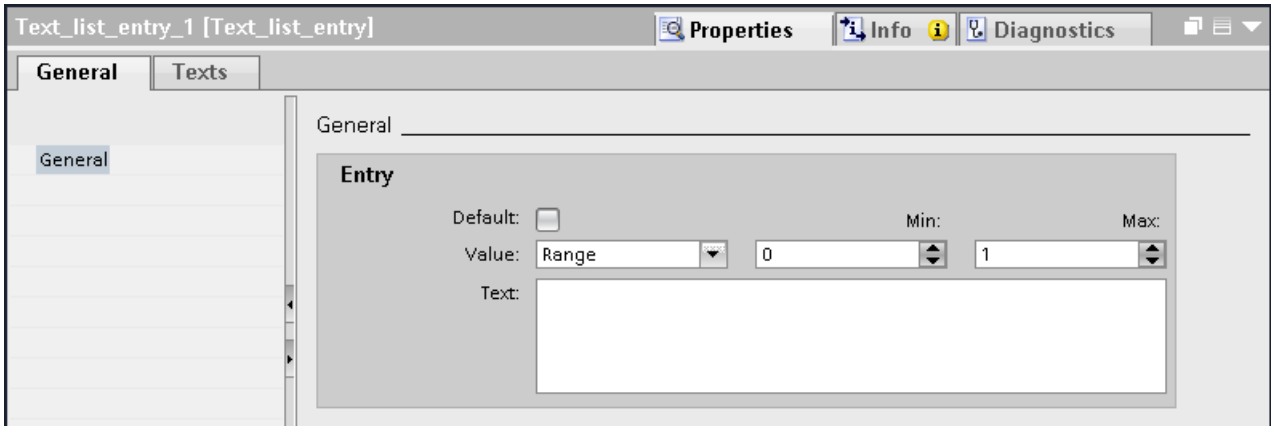
For each bit number text list you specify which texts are displayed at which bit number.

Requirement

- The "Text and graphics list" editor is open.
- The "Text lists" tab is open.
- A bit number text list has been created and selected.

Procedure

1. Click "Add" in the "Text list entries" table.
   The Inspector window for this list entry opens.



2. Select the setting "Single value" in "Properties > Properties > General > Value" in the Inspector window.
   - Enter "10", for example, for "Value".
   - Under "Text", enter the text that is displayed in Runtime when the tag has the value "10".
3. If required, activate the "default entry".
   The entered text is always displayed when the tag has an undefined value. Only one default entry is possible per list.
4. Create further list entries for additional bit numbers of the same text list.

Note

Use a maximum of 320 characters for the text.

Use a maximum of 255 characters and no semicolons for the text in WinCC Runtime Professional.

Note

Bit selection for text lists

The output of the text list depends on the "Bit selection for text and graphics lists" option in the Runtime settings. Use this option to specify that the bit selection for text lists should be used on your HMI device. When you enable the option, the text displayed is the one configured for the set bit with the least significance. When you disable this option, only the text that has been configured for the set bit is displayed .

Result

A bit number text list is created. Texts that appear in Runtime are assigned to the specified bit numbers.

Configuring objects with a text list (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

The output value and value application for text lists are specified in the display and operating object that displays the texts of the text list in Runtime. The properties of these objects are configured as required.

Requirement

- A text list is created.
- You have created a tag.
- The "Screens" editor is open.
- A screen with a symbolic I/O field is open. The object is edited.

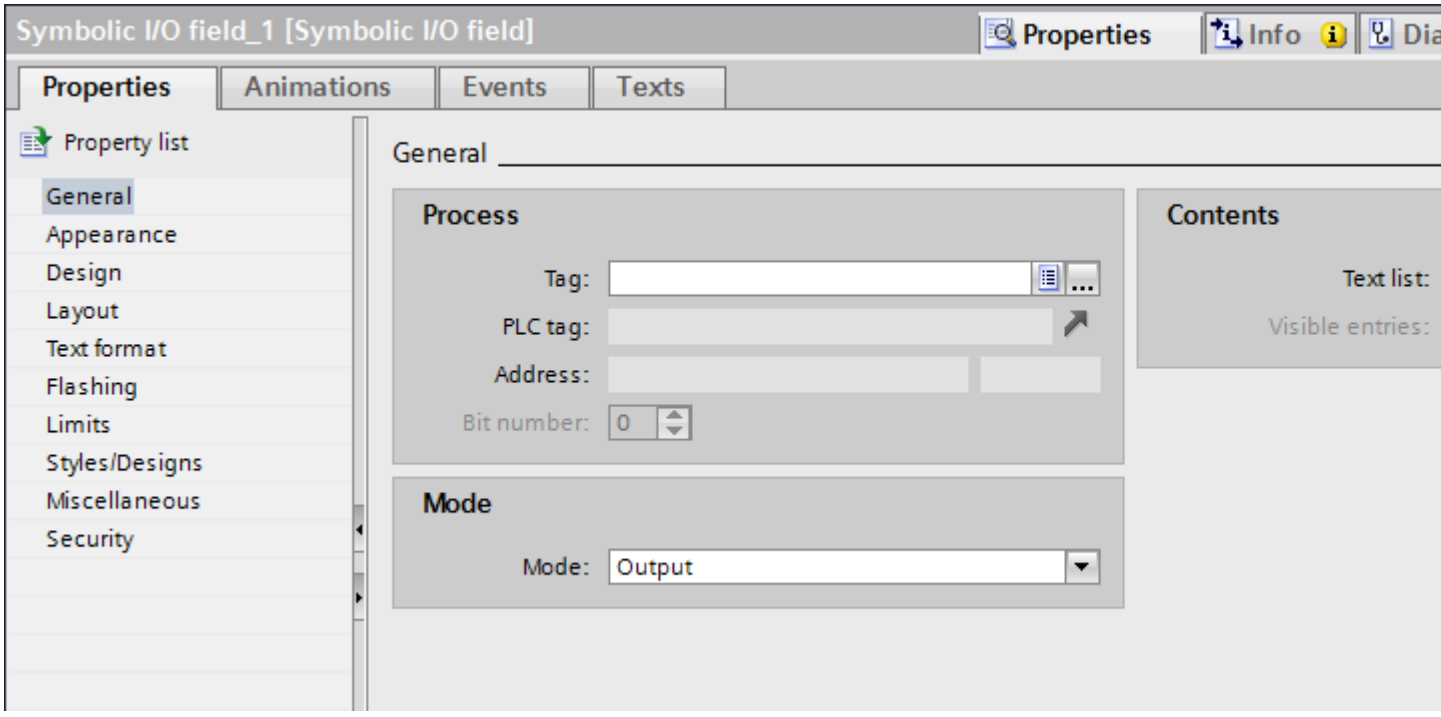
Procedure

1. Select the text list which you want to have displayed in Runtime in "Properties > Properties > General > Text list" in the Inspection window.
2. Select the setting "Output" as the "Mode".

   Note
   Runtime dependency

   Different field types are available for a symbolic I/O field depending on the Runtime.

3. Select the tag the value of which determines the display in the symbolic I/O field as "Tag".



Result

The defined texts of the text list are displayed in the symbolic I/O field in Runtime when the tag has the specified value.

12.1.3.2 Working with graphics lists

Basic principles of graphics lists (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The possible values of a tag are assigned to specific graphics in a graphics list. During configuration, you can create a graphics list for a button or a graphic I/O field. This supplies the graphics to be displayed to the object.

The graphics lists are created with the "Text and graphics list" editor. You configure the interface between the graphics list and a tag at the object that uses the graphics list. The availability of the graphics list is determined by the HMI device used.

Application

You can configure the graphics list for the following situations:

- Drop-down list with a graphic I/O field
- State-specific graphic for a button

The graphics in a graphics list can be configured as multilingual. The graphics will then be displayed in the set language in Runtime.

Graphic sources

Graphics can be added to the graphics list from the following sources:

- By selecting from a graphic browser
- By selecting an existing file
  You can use the following file types:
  *.bmp, *.ico, *.emf, *.wmf, *.gif, *.tiff, *.png, *.jpeg and *.jpg.
- By creating a new file

Function

If the graphic I/O field is a display field, the associated graphics will differ according to the value of the configured tags. If the graphic I/O field is an input field, the configured tag assumes the associated value when the operator selects a graphic in Runtime.

Ranges for the graphics list

Three types are available for the graphics lists:

- Value/Range
  This setting assigns graphic entries from the graphics list to integer values or value ranges of a tag. You can select the number of graphic entries as needed. The maximum number of entries depends on the HMI device you are using.
  You specify a default value which is shown if the value of the tag lies outside the defined range.
- Bit (0, 1)
  This setting assigns graphic entries from the graphics list to two states of a binary tag. You can create a graphic entry for each state of the binary tag.
- Bit number (0 - 31)
  This setting assigns a graphic entry from the graphics list to each bit of a tag. The maximum number of graphic entries is 32. This form of graphics list can be used, for example, in a sequence control when processing a sequence in which only one bit of the used tag may be set. You influence the behavior of the bit number (0 - 31) with the set bit of the least significance and a default value.

Configuration steps

The following tasks are required to display graphics, for example, in a graphic I/O field:

1. Creating the graphics list
2. Assignment of the graphics to values or value ranges of a graphics list
3. Assignment of a graphics list in the display object, for example, the graphic I/O field

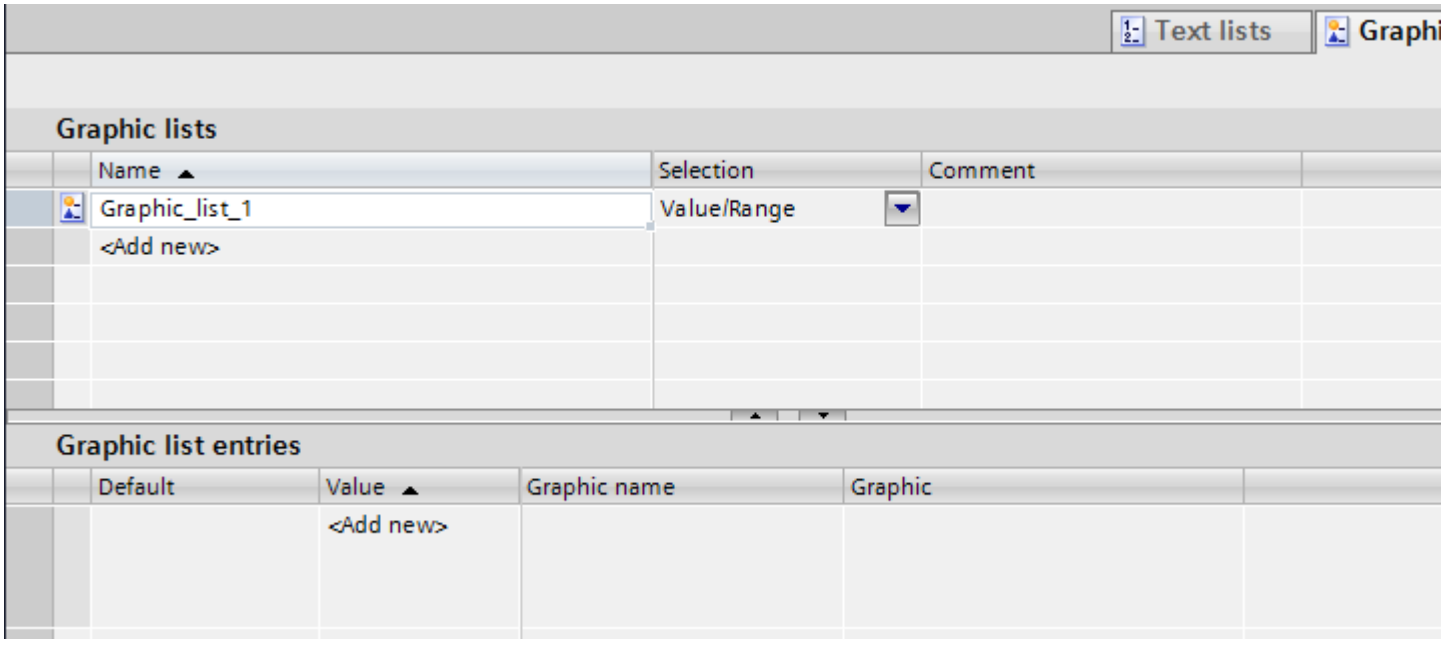
Creating a graphics list (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

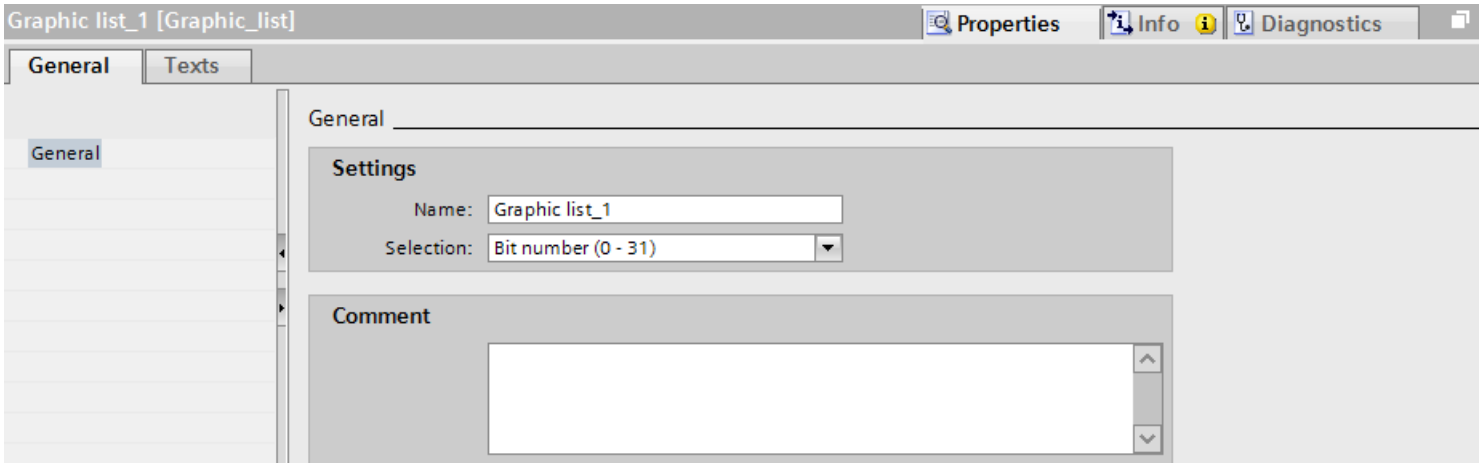
The graphics list allows you to assign specific graphics to variable values and to output these in a graphic IO field in Runtime. You can specify the type of graphic I/O field, for example as a pure output field.

Procedure

1. Double-click "Text and graphics lists" in the project window.
2. Open the "Graphics lists" tab.



3. Click "Add" in the "Graphics lists" table. The Inspector window of the graphics list will open up.



4. Assign a name to the graphics list that indicates its function.
5. Select the graphics list type "Bit number (0 - 31)" for example under "Select".
6. Enter a comment for the graphics list.

Result

A graphics list of the type "Range (0 - 31)" is created.

Assigning a graphic and values to an area graphics list (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

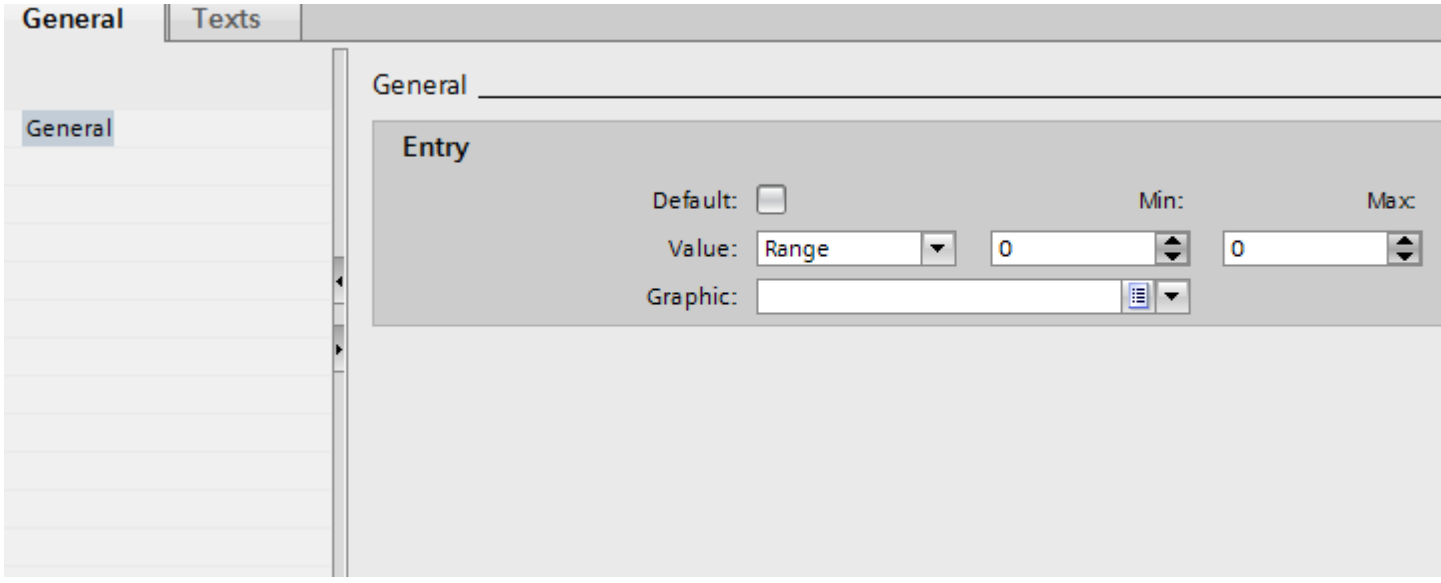
For each area graphics list you specify which graphics are displayed at which value range.

Requirement

- The "Text and graphics list" editor is open.
- The "Graphics list" tab is open.
- An area graphics list has been created and selected.

Procedure

1. Click "Add" in the "Graphics list entries" table.
   The Inspector window for this list entry opens.



2. Select the settings "Range" in "Properties > Properties > General > Value" in the Inspector window:
   - Enter the value "1" for "Min" for example.
   - Enter the value "20" for "Max" for example.
   - Select a graphic that is displayed in Runtime when the tag is within the specified value range.



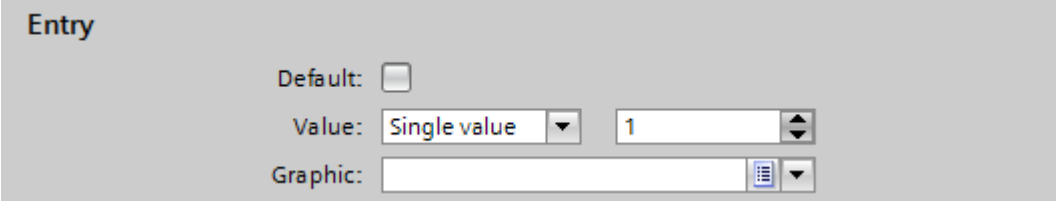
Note

As an alternative to the drop-down menu, you can insert graphics from libraries or from your file system:

1. Select a graphic in the library or in your file system.
2. Drag-and-drop the graphic into the "Graphics list entries > Graphic" table.

3. Click "Add" in the "Graphics list entries" table. A further list entry is created.

4. Select the settings "Single value" in "Properties > Properties > General > Value" in the Inspector window:
   - Enter the value "21" for example.
   - Select a graphic which is displayed in Runtime if the bit "21" is set in the tag.



5. If required, activate the "default entry".
   The graphic is always displayed when the tag has an undefined value. Only one default entry is possible per list.

Result

An area graphics list is created. Graphics that appear in Runtime are assigned to the possible values.

Assigning graphics and values to a bit graphics list (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

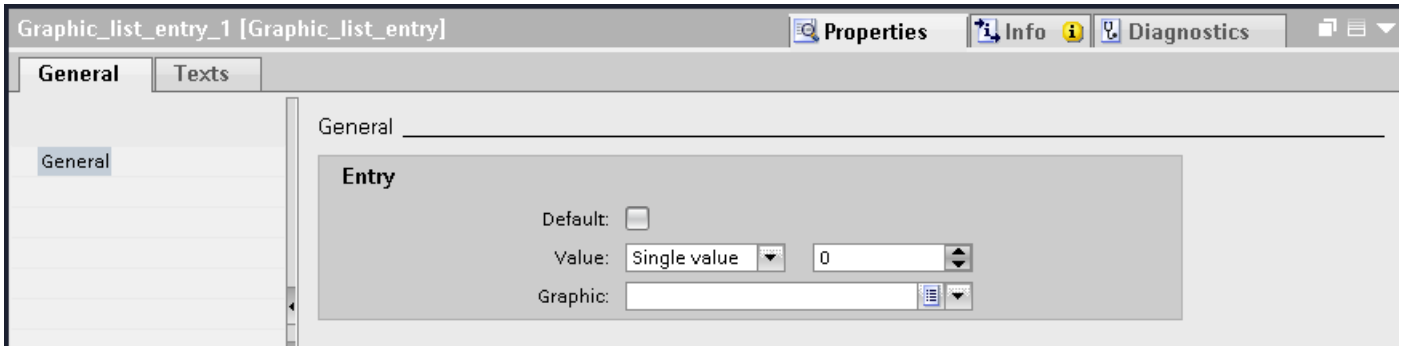
For each graphics list you specify which graphic is displayed at which bit value.

Requirement

- The "Text and graphics list" editor is open.
- The "Graphics list" tab is opened.
- A bit graphics list has been created and selected.

Procedure

1. Click "Add" in the "Graphics list entries" table.
   The Inspector window for this list entry opens.



2. Select the settings "Single value" in the inspector window "Properties > Properties > General > Value":
   - Enter "0" as the value.
   - Select a graphic which is displayed in Runtime if the bit "0" is set in the tag.

   Note

   As an alternative to the drop-down menu, you can insert graphics from libraries or from your file system:
   1. Select a graphic in the library or in your file system.
   2. Drag-and-drop the graphic into the "Graphics list entries > Graphic" table.
3. Click "Add" in the "Graphics list entries" table. A new list entry is created.
4. Select "Properties > Properties > General > Value > Single value": in the Inspector window.
   - Enter "1" as the value.
   - Select a graphic which is displayed in Runtime if the bit "1" is set in the tag.

Result

A bit graphics list is created. Graphics that appear in Runtime are assigned to the values "0" and "1".

Assigning graphics and values to a bit number graphics list (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

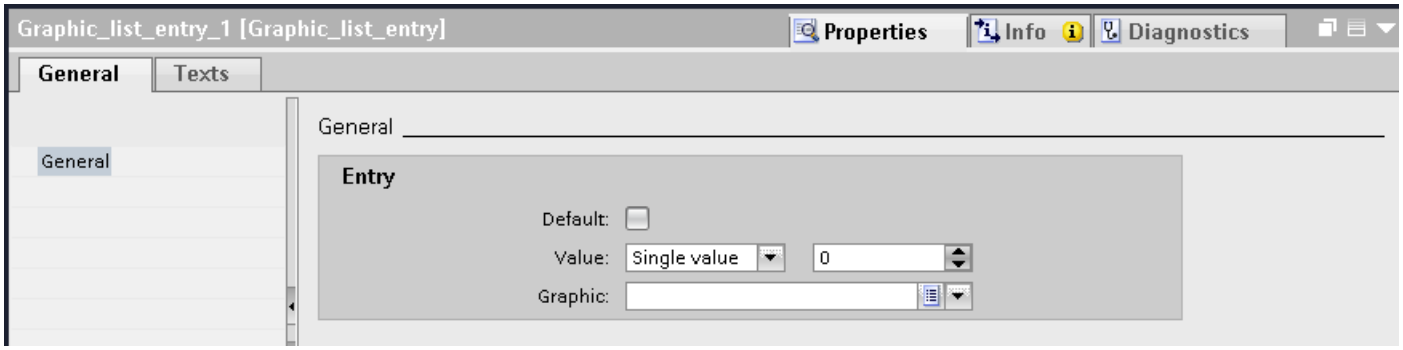
For each bit number graphics list you specify which graphics are displayed at which bit number.

Requirement

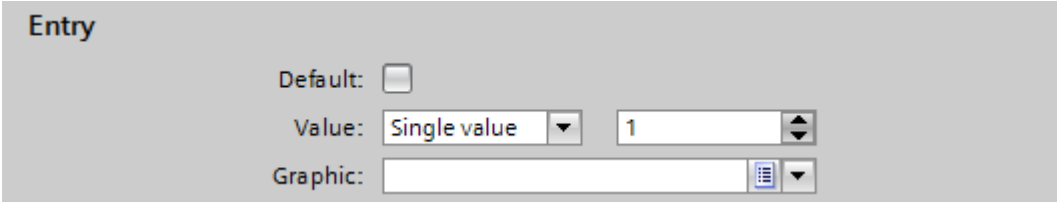
- The "Text and graphics list" editor is open.
- The "Graphics list" tab is open.
- A bit number graphics list has been created and selected.

Procedure

1. Click "Add" in the "Graphics list entries" table.
   The Inspector window for this list entry opens.



2. Select the settings "Single value" in the Inspector window "Properties > Properties > General > Value":
   - Enter the value "1" for example.
   - Select a graphic which is displayed in Runtime if the bit "0" is set in the tag.



Note

As an alternative to the drop-down menu, you can insert graphics from libraries or from your file system:

1. Select a graphic in the library or in your file system.
2. Drag-and-drop the graphic into the "Graphics list entries > Graphic" table.

3. If required, activate the "default entry".
   The graphic is always displayed when the tag has an undefined value. Only one default entry is possible per list.
4. Create further list entries for additional bit numbers of the same graphics list.

Note

Bit selection for graphic lists

The output of the graphic list depends on the "Bit selection for text and graphics lists" option in the Runtime settings. Use this option to specify that the bit selection for graphic lists should be used on your HMI device. When you enable the option, the graphic displayed is the one configured for the set bit with the least significance. When you disable this option, the graphic that has been configured only for the set bit is displayed.

Result

A bit number graphics list is created. Graphics that appear in Runtime are assigned to the specified bit numbers.

Configuring objects with a graphics list (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

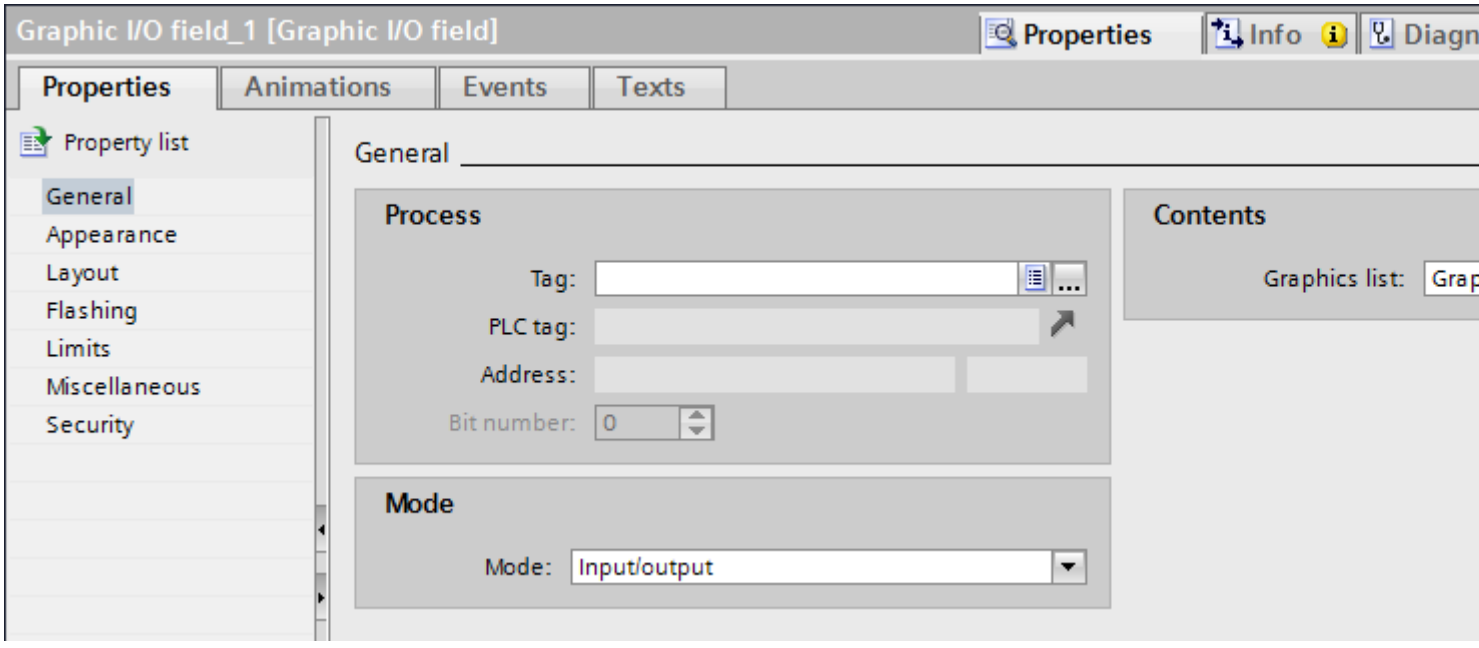
The output value and value application for graphics list are specified in the display and operating object that displays the graphics of the graphics list in Runtime. The properties of these objects are configured as required.

Requirement

- A graphics list is created. The values have been defined. Graphics have been assigned to the values.
- You have created a tag.
- The "Screens" editor is open.
- A screen with a graphics I/O field is open. The object is edited.

Procedure

1. Select the graphics list the graphics of which you want to have displayed in Runtime in "Properties > Properties > General > Graphics list" in the Inspector window.
2. Select the "Input/output" setting for "Mode".



Note

Runtime dependency

Different field types are available for a graphic I/O field depending on the Runtime.

3. As "Tag", select the tag whose values are defined by the display in the graphic I/O field.

Result

The defined graphics of the graphics list are displayed in the graphic I/O field in Runtime when the tag has the specified value.

12.1.4 Dynamizing screens

12.1.4.1 Basics on dynamizing screens

Basics on dynamization (Basic Panels)

Dynamizing objects

In WinCC you dynamize objects to map your system and show processes on HMI devices.

You implement dynamizations by

- Animations
- Tags
- System functions

One example is the mapping of a tank, the liquid level of which rises or falls in relation to a process value.

The options for dynamization depend on the object involved. When you copy an object, its dynamization functions are included.

Dynamization in the inspector window (Basic Panels)

Introduction

Basically, you can dynamize all the screen objects which you have configured in a screen. Which dynamization possibilities and which events are available depends on the device and the selected object.

Animations

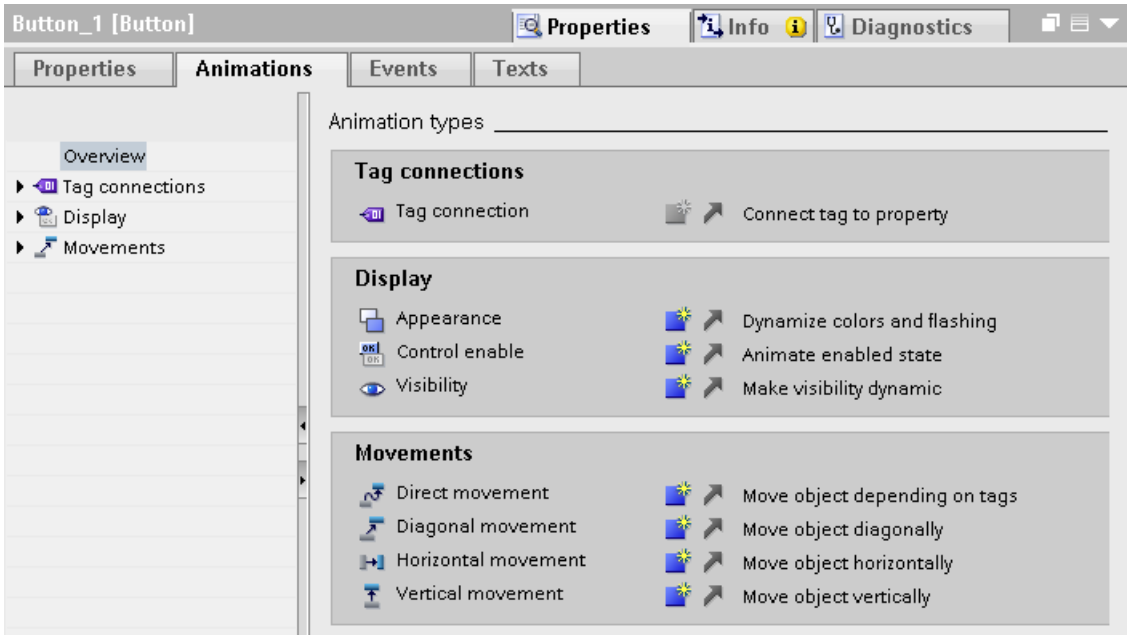
WinCC helps you to implement dynamization using predefined animations. If you want to animate an object, first configure the desired animation in the object's inspector window. Then adapt the animation to the requirements of your project.

The selection of the supported animations depends on the HMI device and the selected object. You choose between the following types of animation:

- Layout: Appearance, visibility
- Movements: direct, diagonal, horizontal and vertical movement
- Variable binding

You can configure the "Variable binding" type of animation several times for one object.

You configure animations in the "Properties > Animations" inspector window.

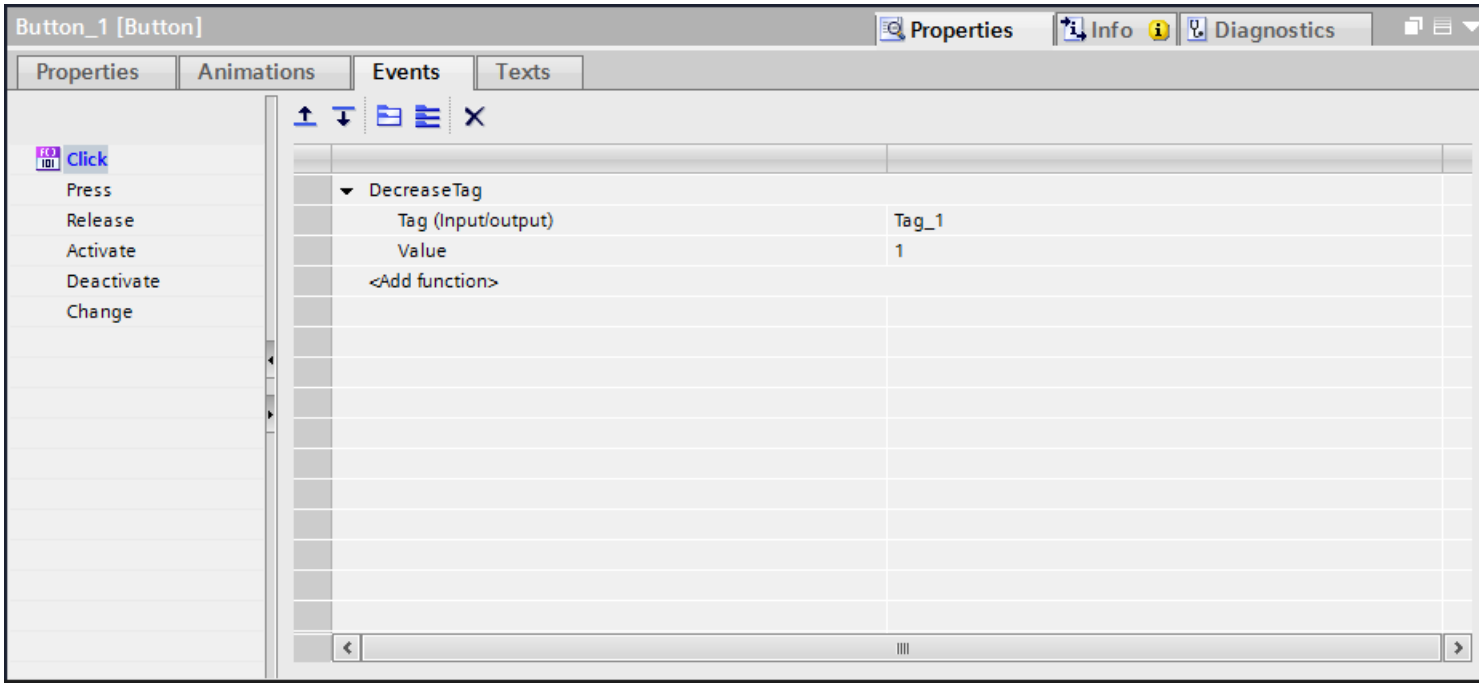


Events

Operable objects also react to events, such as a mouse click.

You configure a function list with system functions on an event. The system functions are processed as a reaction to the triggered event.

You configure events in the "Properties > Events" inspector window.



You can find more detailed information on this under "Working with Function Lists".

12.1.4.2 Dynamization with animations

Configuring a new animation (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Use predefined animations to dynamize screen objects.

Requirement

- You have opened a screen which contains at least one dynamic object.
- The Inspector window is open.
- The toolbox window is displayed.

Procedure in the inspector window

1. In the Inspector window, select "Properties > Animations".
2. Select the animation you want to use.
3. Click the  button.

Procedure in the "Animations" task card.

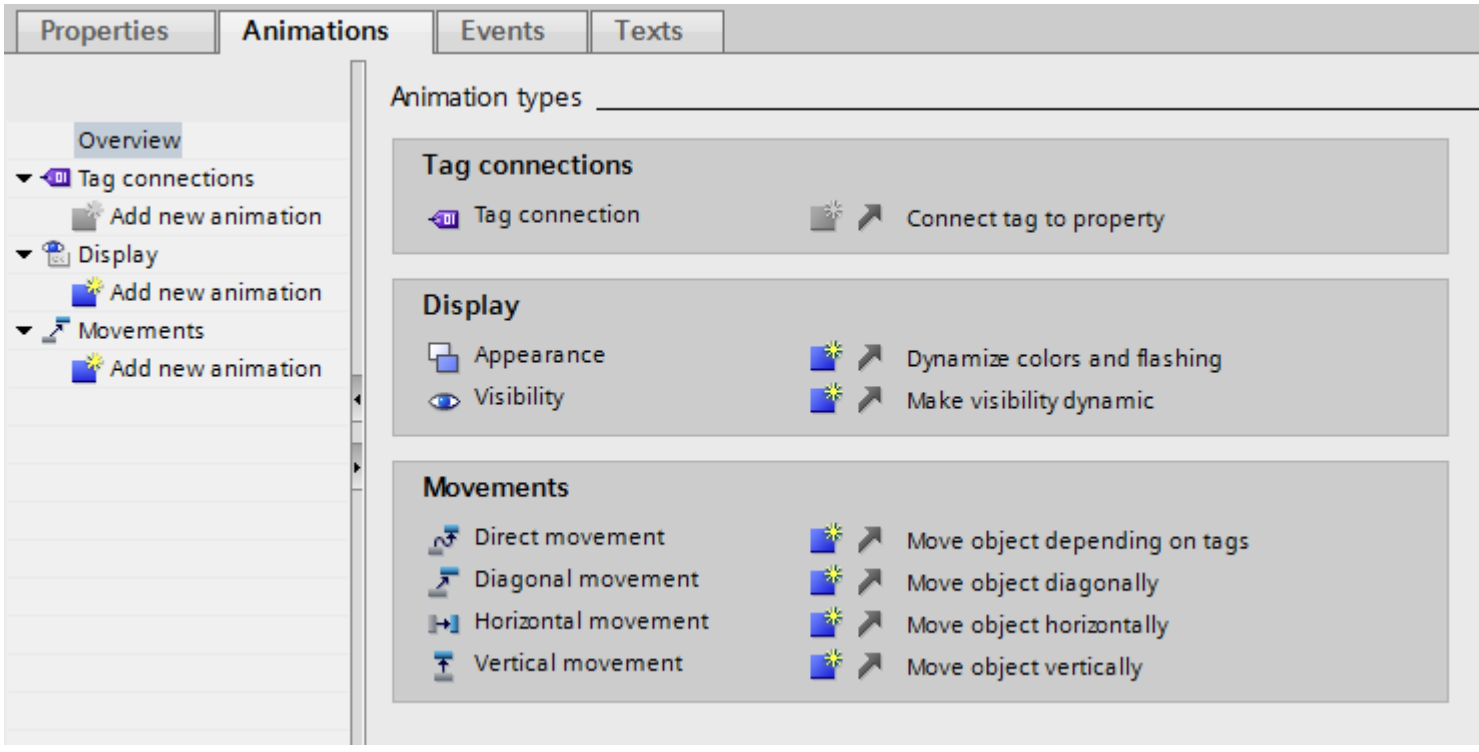
1. Open the object group containing the required animation in the "Animations" task card.
2. Drag the animation onto the object that you want to make dynamic.

Alternatively you select the object in the screen and double click the desired animation in the "Animation" task card.

Result

The animation appears in the Inspector window of the object. You configure the animation in the following steps.

In the animations overview the green arrow indicates which animation is already configured. The configured animation opens in the inspector window when you click the green arrow.



Dynamizing the appearance of an object (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The appearance of a screen object is controlled by changing the value of a tag in runtime. When the tag adopts a certain value, the screen object changes its color or flashing characteristics according to the configuration.

Type

Areas or single values of the tag are observed in Runtime depending on the selection. The appearance of the object changes according to the configuration.

Note

"Multiple bits" selection

If you use the "Multiple bits" option, also select "Bit selection for appearance" option in the runtime settings under "Screens > General".

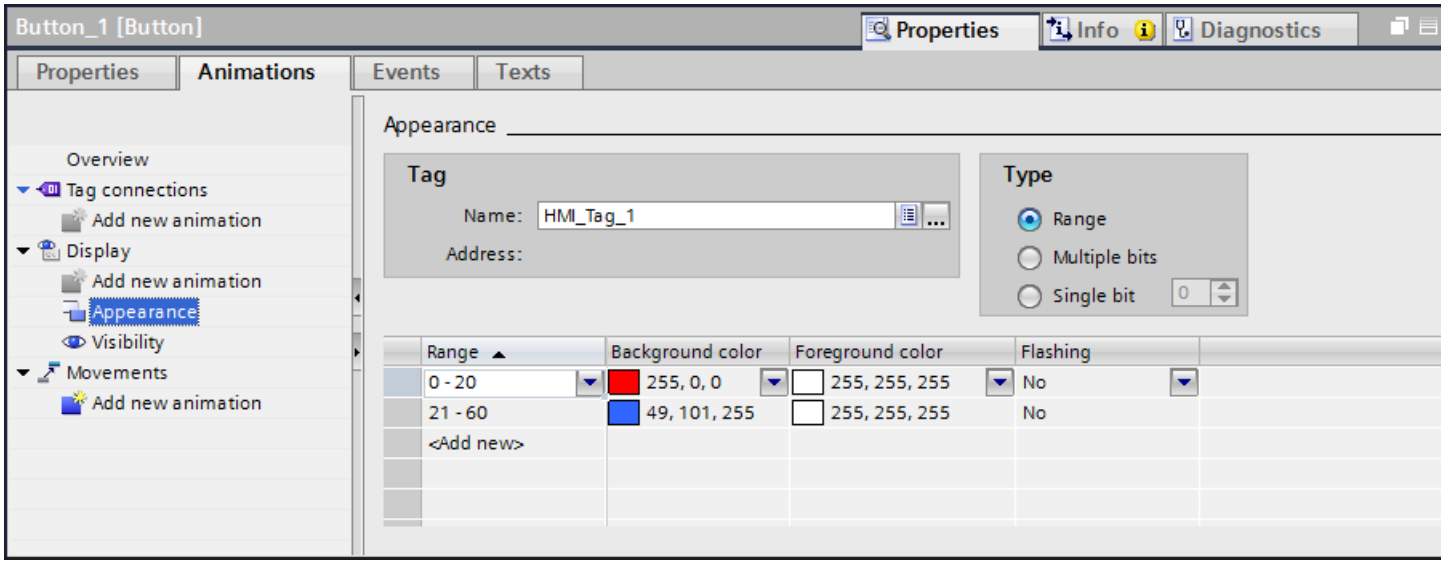
Requirement

- A screen is open.
- A dynamic object is contained and selected in the screen.

- The Inspector window is open.
- The toolbox window is displayed.

Procedure

1. In the Inspector window, select "Properties > Animations".
   The animations available for the selected object are displayed.
2. Select the animation "Appearance" and click the button.
   The parameters of the animation are displayed.
3. Select a tag in "Tag > Name".
4. Select "Type > Range" for example.
5. Click "Add" in the table.
6. Enter the tag interval "0 - 20" in the "Range" column for example.
7. For "Foreground color" and "Background color", select the color the object is to acquire in Runtime when the tag reaches the interval.
8. Select a flashing mode for the object from the "Flashing" list.
9. Repeat steps 5 to 8 to create another tag interval, e.g. "21 - 60".



Result

In Runtime, the flashing response, and color of the object change dynamically according to the process value returned in the tag.

Configuring movements (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You can configure dynamic objects in such a way that they move on a certain track. The movement is controlled via tags. The object moves every time the tag is updated.

You can only program one type of movement for each object.

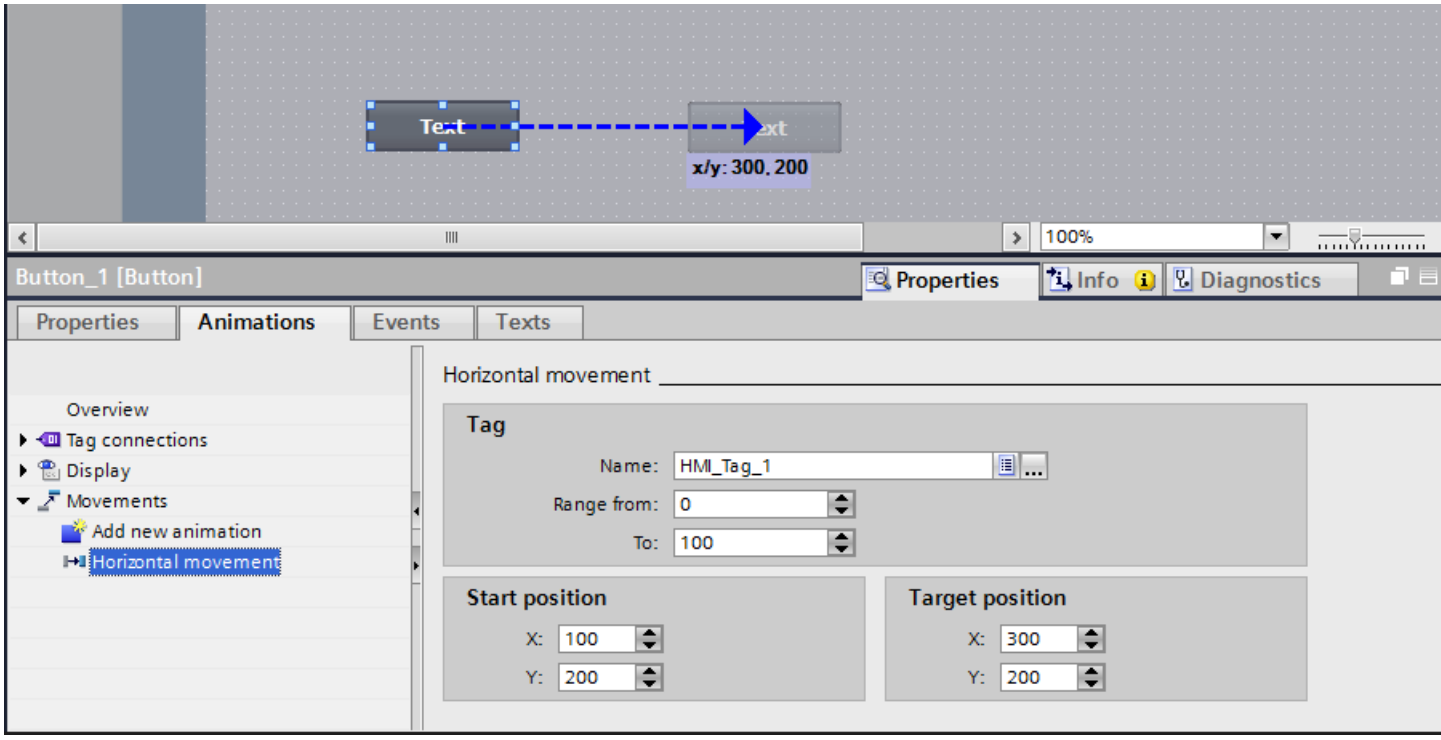
Requirement

- You have created a tag.
- You have opened a screen which contains at least one dynamic object.
- The Inspector window is open.
- The toolbox window is displayed.

Procedure

1. Select the screen object you want to control dynamically.
   The object properties are displayed in the Inspector window.
2. In the Inspector window, select "Properties > Animations".
   The animations available for the selected object are displayed.
3. Select "Horizontal movement" and click the  button.
   The parameters of the animation are displayed.
   A transparent copy of the object is shown in the work area, which is connected to the source object by means of an arrow.
4. Select a tag for control of movement.

5. Move the object copy to the relevant destination. The system automatically enters the pixel values of the final position in the Inspector window.
6. Customize the range of values for the tag as required.



Result

In Runtime, the object moves in response to every change in value of the tag that is used to control the movement. The direction of movement corresponds to the configured type of movement "horizontal".

Note

You configure vertical and diagonal movements similar to horizontal movements

Configuring direct movement (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

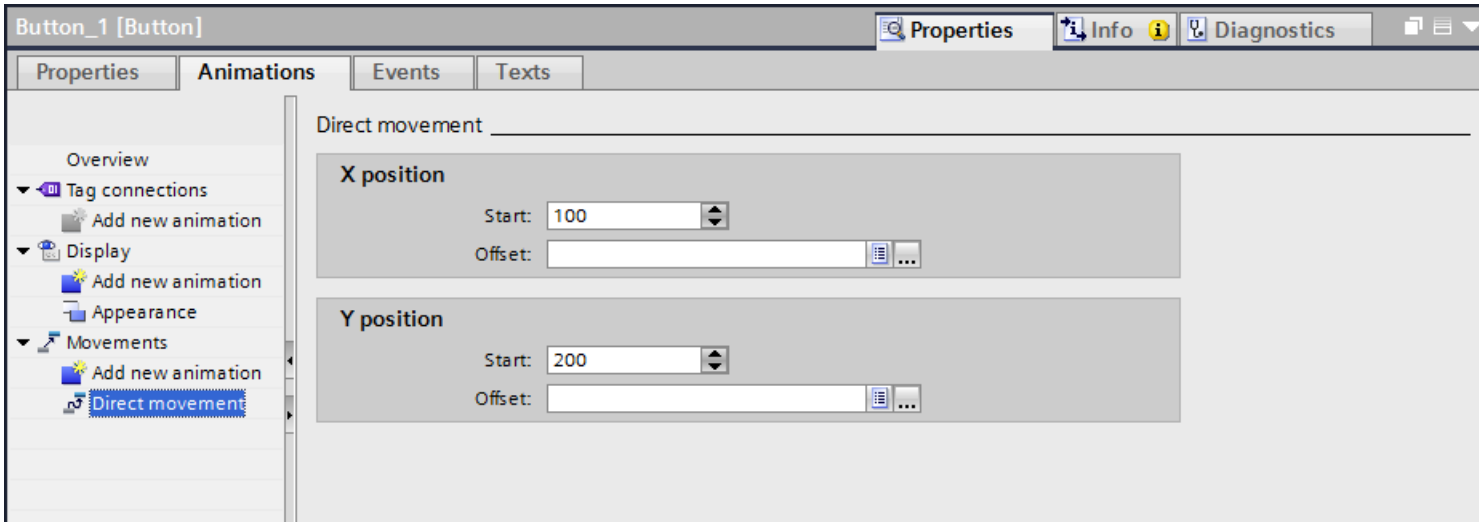
The object is moved respectively in x direction and y direction with "Direct movement". Two tags define the number of pixels by which the object moves from its original static start position.

Requirement

- Two tags are set up.
- You have opened a screen which contains at least one dynamic object.
- The Inspector window is open.
- The toolbox window is displayed.

Configuring "Direct movement"

1. Select the screen object you want to control dynamically.
   The object properties are displayed in the Inspector window.
2. In the Inspector window, select "Properties > Animations".
3. Select "Direct movement" and click the button.
   The parameters of the animation are displayed.
4. Select a tag for the X position with which the movement in x direction is controlled.
5. Select a tag for the Y position with which the movement in y direction is controlled.



Result

In Runtime, the object moves in response to every change in value of the tag that is used to control the movement.

Dynamizing the visibility of an object (Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Dynamization of the "Visibility" property allows you to output an alarm to your screen, which is triggered when the tag value exceeds a critical range, for example. The alarm is cleared when the tag value returns to the non-critical range.

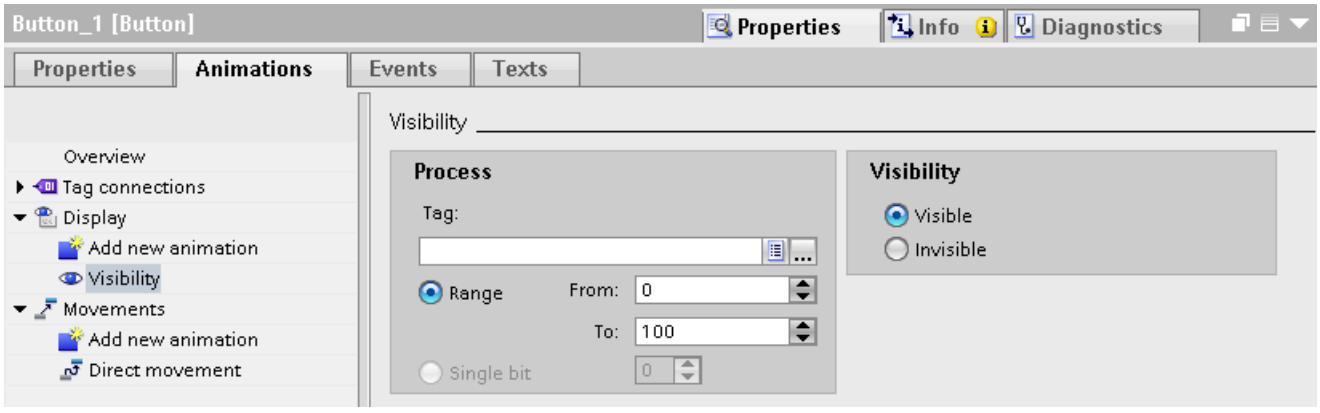
The "Simple recipe view" and "Simple alarm view" objects are always visible.

Requirement

- You have created a tag.
- You have opened a screen containing an object that you want to show or hide in Runtime.
- The Inspector window is open.

Procedure

1. Select the screen object you want to control dynamically.
   The object properties are displayed in the Inspector window.
2. In the Inspector window, select "Properties > Animations".
   The animations available for the selected object are displayed.
3. Select "Visibility" and click the button.
   The parameters of the animation are displayed.
4. Select a tag.
5. Activate "Range".
6. Select, for example, "from 20" and "to 40"
7. Activate "Visible".



Result

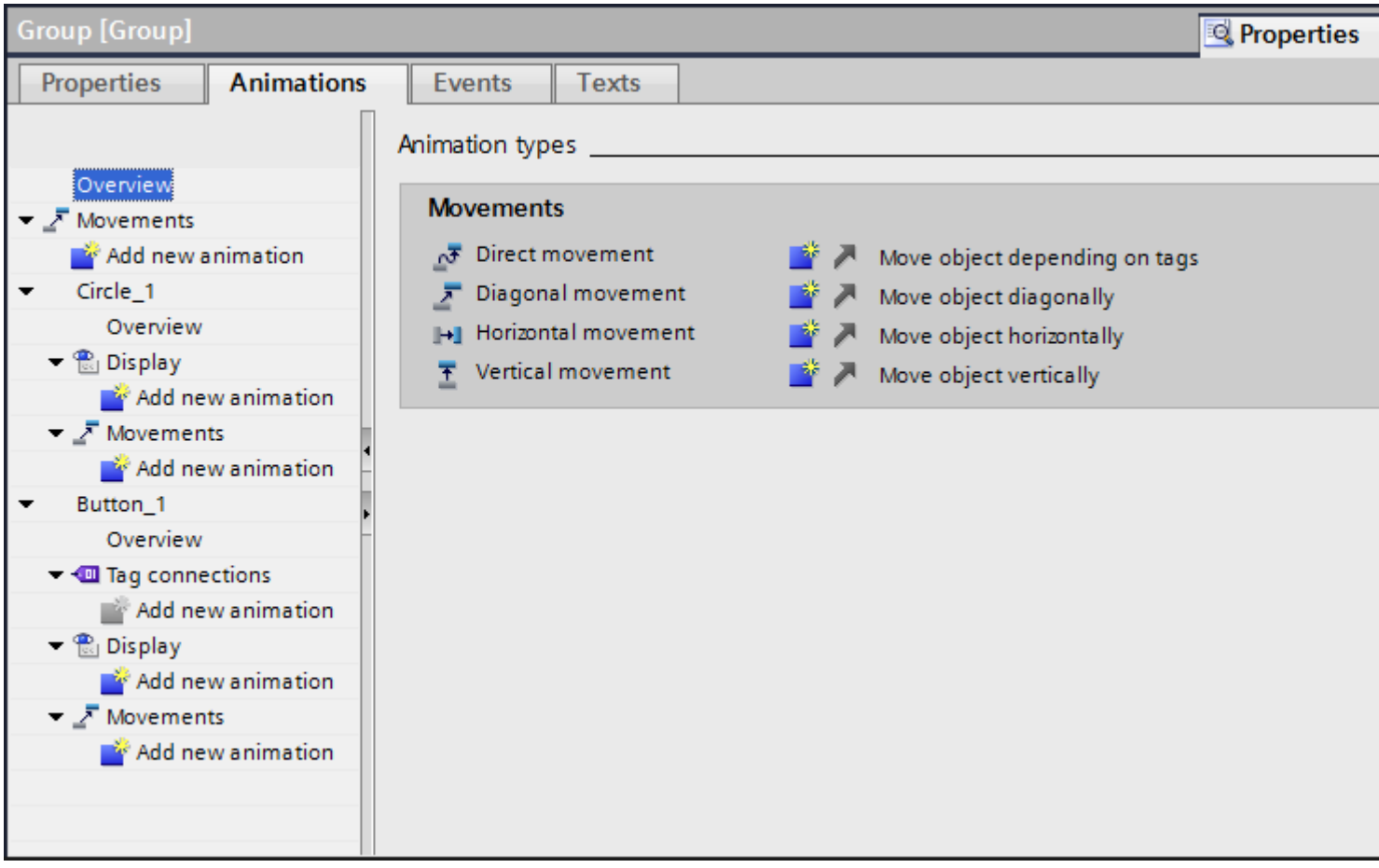
The screen object is shown or hidden in Runtime depending on the tag value.

- If the tag value matches the configured range between 20 and 40, the screen object is shown.
- If the tag value is outside the configured range, the screen object is hidden.

Animations of object groups (Basic Panels)

Applying animations to object groups

The inspector window shows all objects of a group and their possible animations. The animation types which are supported by all objects in the group are also listed separately.



If you configure an animation for an object group, this animation will apply to all individual objects that support this animation.

The animation of nested groups is not supported.

Application example

The "horizontal movement" animation is configured for the object of an object group. The "direct movement" animation is configured for the whole object group. Only the animation of the object group i.e. "direct movement" is executed in Runtime. This also applies for object groups within object groups. Only the animation of the group on the top layer is listed.

Animations for object groups and for multiple selection (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Changing animations for multiple objects

For a multiple selection, the animations that are configured for the reference object appear in the Inspector window. You can change the animations as usual. The changes will apply to all the objects in the multiple selection that support the configured animation. This means that even objects for which you have not yet configured an animation will have the reference object's animation.

Application example

Select a button, and a circle at the same time. The button is the reference object. The "Appearance" animation is already configured for the button, so it appears in the Inspector window of the multiple selection. When you activate "Properties > Animations >Appearance > Flashing" in the inspector window, the settings of the "Appearance" animation apply for the button and for the circle.

Configuring new animations for multiple objects

If you configure a new animation for the objects of a multiple selection, this animation will apply to all selected objects that support the configured animation. If the new animation replaces an existing animation, a security prompt appears.

Application example

You select a circle, and a rectangle. The "Diagonal movement" animation is already configured for the circle. You configure the "Horizontal movement" animation for the multiple selection. This animation applies to the rectangle since no animation of the Movement type is yet configured for it. For the circle, you are asked to confirm that you want to replace the existing "Diagonal movement" animation with the new "Horizontal movement" animation.

12.1.4.3 Dynamizing with system functions

Basic on events

Introduction

Screen objects react to events. You configure a function list with system functions on the events of an object.

Events

Which events and system functions are available depends on the object used.

If the operator activates a screen object for example, the configured system function is executed.

You can find additional information at "Basics on Function Lists (Page 5651)".

See also

Basic of the functions list (Page 5651)

Configure system function on the "Click" event (Basic Panels)

Introduction

You configure a function list on an object event. The linked system function is executed when the event occurs in runtime.

Requirements

A screen is open.

A button has been created in the screen.

The inspector window is open.

Procedure

1. Select the button.
2. Click "Properties> Events" in the Inspector window.
3. Select the "Click" event.
4. Click "Add function" in the table.
5. Select the "ShowAlarmWindow" system function.

Result

The alarm window opens in the screen if the operator clicks the button in runtime.

Example: Configuring a button for language switching

Introduction

In this example, you configure a button that can be used to toggle between multiple runtime languages during runtime.

Requirements

- You have completed the "Configuring a button in multiple languages" example.
- The "Screen_1" screen is open.
- The button on the screen has been selected.

Procedure

1. In the Inspector window, select "Properties > Events > Press".
2. Click on "Add function" in the table.
3. Select the "SetLanguage" system function and the "Switch" setting.

Result

You have assigned the button the function "SetLanguage". Pressing the button during runtime will switch the runtime language. The runtime languages are switched in the order specified by the number sequence in the "Languages and fonts" editor.

12.1.5 Working with function keys

12.1.5.1 Working with function keys

Introduction

The function key is a physical key on your HMI device and its functions can be configured. A function list can be configured for the events "Key pressed" and "Release key".

A function key can be assigned global or local functions.

Note

Availability for specific HMI devices

Function keys are not available on all HMI devices.

Global function keys

Global function keys always trigger the same action, regardless of the displayed screen.

Global function keys are configured in the "Global Screen" editor. The global assignment applies for all the screens of the set HMI device.

Global function keys reduce programming considerably, because there is no need to assign these global keys to each individual screen.

Local function keys in screens

Local function keys in screens can trigger a different action in each screen. This assignment applies only to the screen in which you have defined the function key.

Within a screen, a function key has only one function assignment, either a global or local one. The project planner specifies which assignment has priority.

Note

If a screen with local function keys is overlapped by an alarm view or an alarm window, then the function keys are still active in runtime. This may happen in particular on HMI devices with a small display.

Local function keys in templates

Local functions keys that are assigned in templates are valid for all the screens based on this template. They can trigger a different action in every screen. Function keys for templates are assigned in the template of the "Screens" editor. Within a template, a function key has only one function assignment, either a global or local one. The project planner specifies which assignment has priority.

Hotkey assignment

You can assign hotkeys, such as buttons, to control objects. The available hotkeys depend on the HMI device.

Note

The function key has a local or global action assigned to it. If the function key also has a hotkey assigned to it, then the hotkey function will be executed in Runtime.

Graphics

When a function key is placed directly next to the display, you can assign a graphic to it to make the function of the function key more clear.

Display of assignment

Table 12-1 The following table shows which symbols display the assignment of the function keys:

Function key	Description
	Not assigned
	Global assignment
	Assigned locally in the template

Function key	Description
	Local assignment
	Local assignment (local assignment of the template overwrites global assignment)
	Local assignment (local assignment overwrites global assignment)
	Local assignment (local assignment overwrites local assignment of the template)
	Local assignment (local assignment overwrites local assignment of the template, which already overwrites global assignment)
	Assigning buttons with screen navigation

Note

Basic Panels

The "Screen Navigation" editor is not available for Basic Panels.

12.1.5.2 Assigning function keys globally

Introduction

You define the global assignment of a function key in the "Global Screen" editor. The global assignment applies to all screens of the set HMI device.

Note

Availability for specific HMI devices

Function keys are not available on all HMI devices.

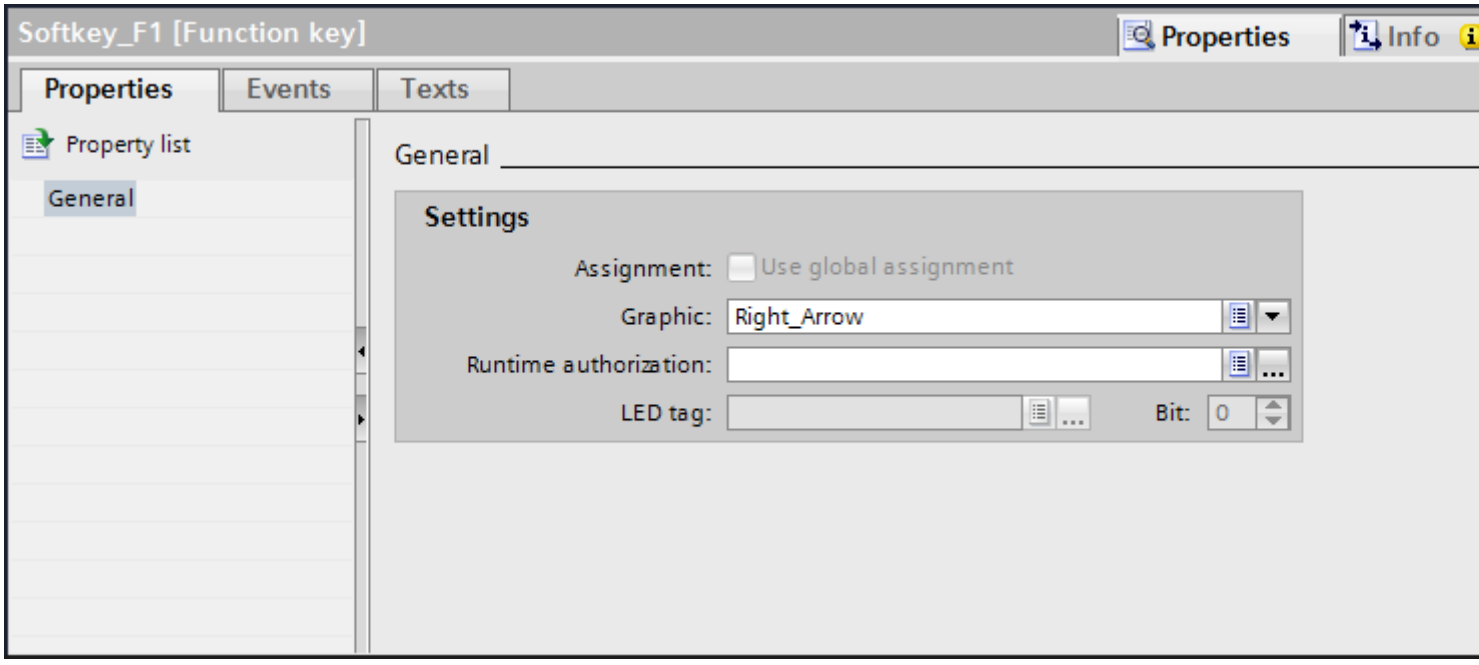
Requirement

- You have opened the project.
- The Inspector window is open.

Procedure

Proceed as follows to assign a screen-based function to a function key:

1. To open the "Global Screen" editor, double-click "Global Screen" in the "Screen management" group of the Project window.
2. Select the desired function key.
   The properties of the function key are shown in the Inspector window.



3. Under "General", configure a graphic and an operator authorization for the function key.
4. Configure a function list for the required event under "Events".

Result

If a local assignment does not overwrite the global assignment, the assignment of the function key changes in all the screens of the set HMI device in accordance with your entry.

12.1.5.3 Local assignment of function keys

Introduction

Function keys are assigned globally and locally. A local assignment of the function keys is only valid for the screen or the template in which it was defined. The following local function keys are available:

- Local function keys of a screen
  Different functions are assigned to the function key for each screen. This assignment applies only to the screen in which you have defined the function key.
- Local function keys of a template
  You assign the function keys in a template. The assignment applies to all the screens that are based on this template and are not overwritten by a local assignment in a screen.

A local assignment overwrites the global assignment of a function key.

Note

Availability for specific HMI devices

Function keys are not available on all HMI devices.

Using existing assignments

The option for using existing assignments is referred to as follows in the Inspector window:

- In a template: "Use global assignment"
- In a screen:
  - Screen based on a template: "Use local template"
  - Screen not based on a template: "Use global assignment"

The "Use local template" option includes the use of the local assignment in the template and the global assignment.

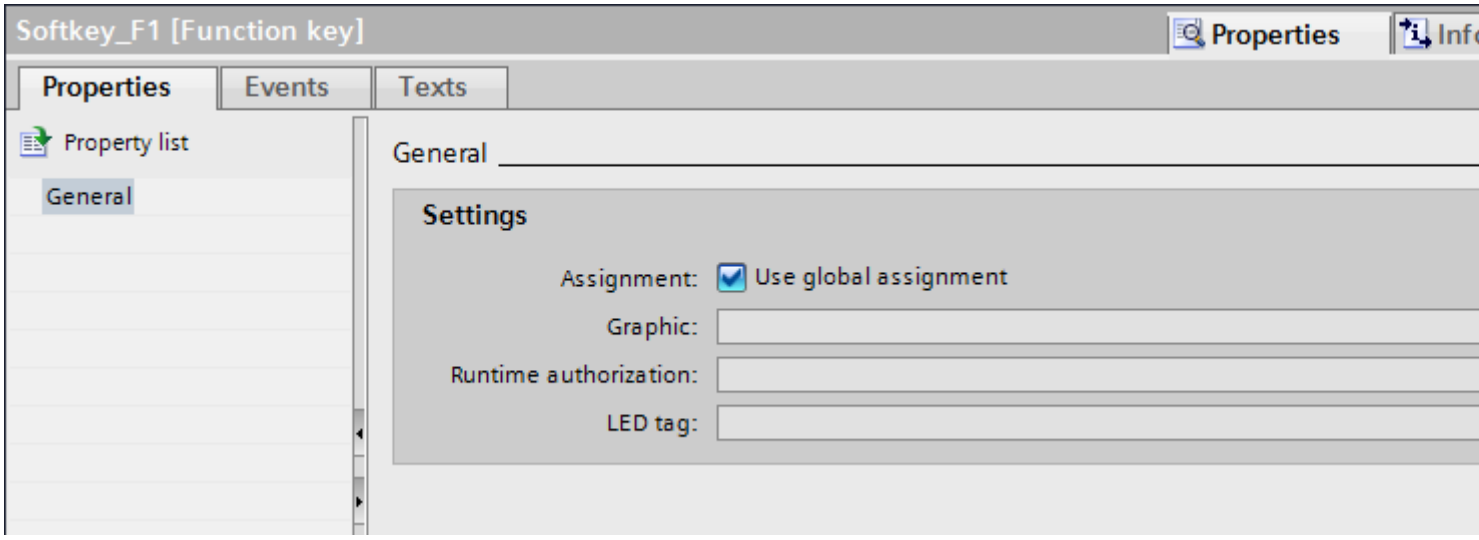
Requirement

- You have opened the screen or the template in which you want to assign a function key locally.
- The Inspector window is open.

Procedure

Proceed as follows:

1. Select the desired function key in the screen or in the template.
   The properties of the function key are shown in the Inspector window.
2. Click "General" in the Inspector window.



3. Disable the "Use local template" or "Use global assignment" option.
4. Under "General", configure a graphic and an operator authorization for the function key.
5. Configure a function list for the required event under "Events".

Result

The function key is assigned the configured function in the screen or in the template.

12.1.5.4 Assigning a function to a function key

Introduction

A function key can have two states:

- Pressed: Defined by the "Key pressed" event.
- Released: Defined by the "Release key" event.

Both of these events are configured in the Inspector window of the function key. You can assign any event a function list which contains system functions or scripts. Execution of this function list is event-driven in runtime.

Note
Availability for specific HMI devices

Function keys are not available on all HMI devices.

Note
Basic Panels

Scripts are not available for Basic Panels.

Requirement

To assign the function key a global function:

- The "Global Screen" editor is open.

To assign the function key a local function:

- The screen in which you want to assign a function key is open.

If you want to assign a function key locally in a template:

- The template in which you want to assign a function key is open.
- The Inspector window is open.

Procedure

Proceed as follows:

1. Select the function key you want to define.
   The properties of the function key are shown in the Inspector window.
2. Configure the function list for the desired result in the Inspector window under "Properties" in the "General" group.

Result

The function list is executed in runtime when the operator presses or releases the function key.

12.1.5.5 Assigning operator authorization for a function key

Introduction

In WinCC you can assign an operator authorization for a function key in runtime. This allows you to restrict access to the function keys to specific persons or operator groups when you configure your project. Only authorized personnel can then change important parameters and settings in runtime.

You configure access protection in order to avoid operator errors and increase plant or machine security.

Note

Availability for specific HMI devices

Function keys are not available on all HMI devices.

Requirement

- The user groups have been defined.

To protect a global function key:

- The "Global Screen" editor is open.

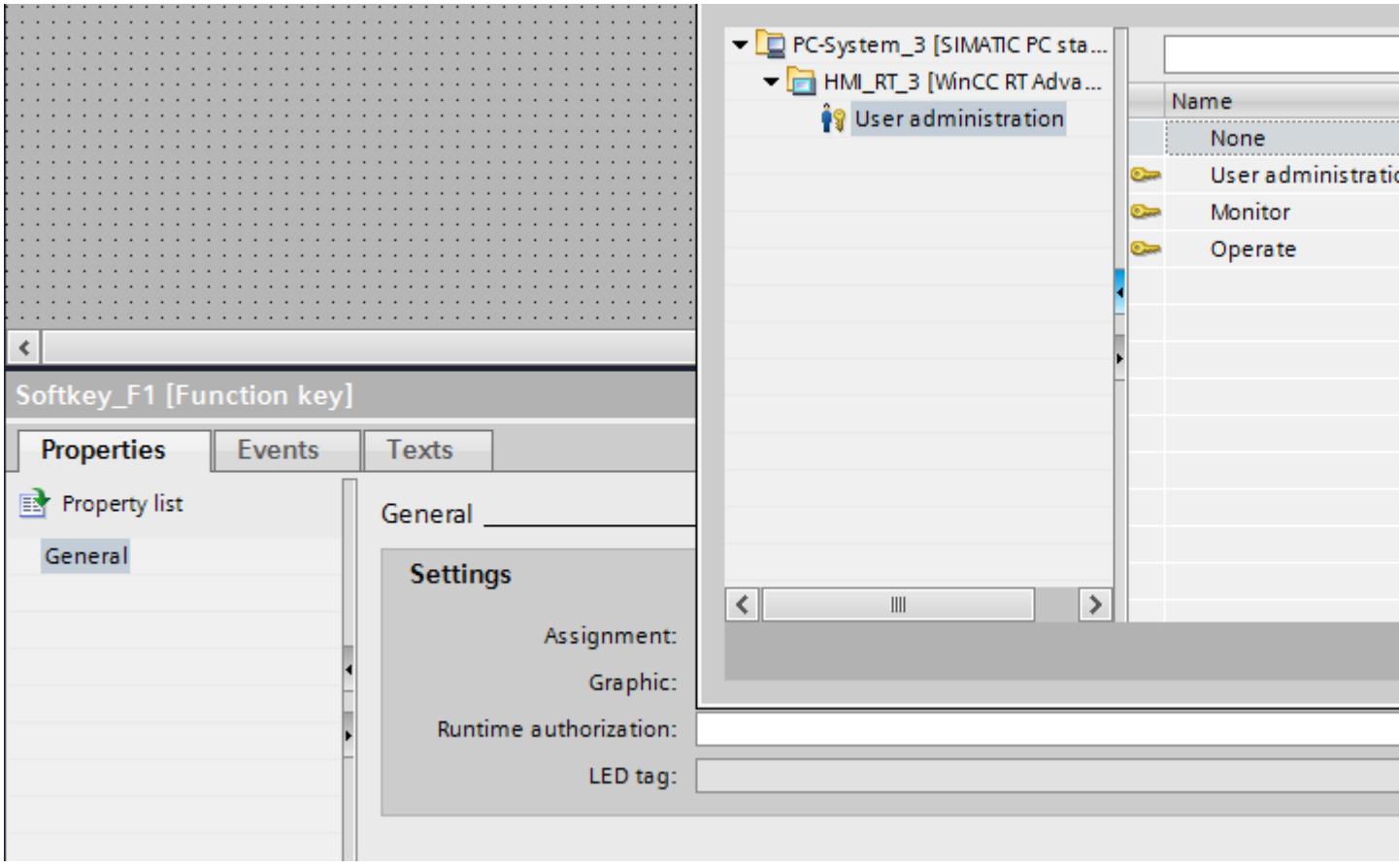
If you want to protect a local function key of a screen or of a template:

- The screen or the template which contains the function key is open.
- The Inspector window is open.

Procedure

Proceed as follows:

1. Select the relevant function key.
   The properties of the function key are shown in the Inspector window.
2. Click "General" in the Inspector window.



3. In the "Authorization" list, select the user group you want to allow runtime access to the function key.

Result

The operator authorization is configured.

12.1.5.6 Assigning a graphic to a function key

Introduction

In order to make the function of a key more clear, you can insert a graphic in the screen alongside the function key. Graphics can only be assigned to function keys that border the screen margin of the HMI device.

Note

HMI device dependency

Function keys are not available on all HMI devices.

Note

Assigning graphics to a function key

A graphic can only be assigned to a function key if the bottom edge of the permanent window does not conceal the area of the function key graphic.

Requirement

To assign a graphic to a global function key:

- The "Global Screen" editor is open.

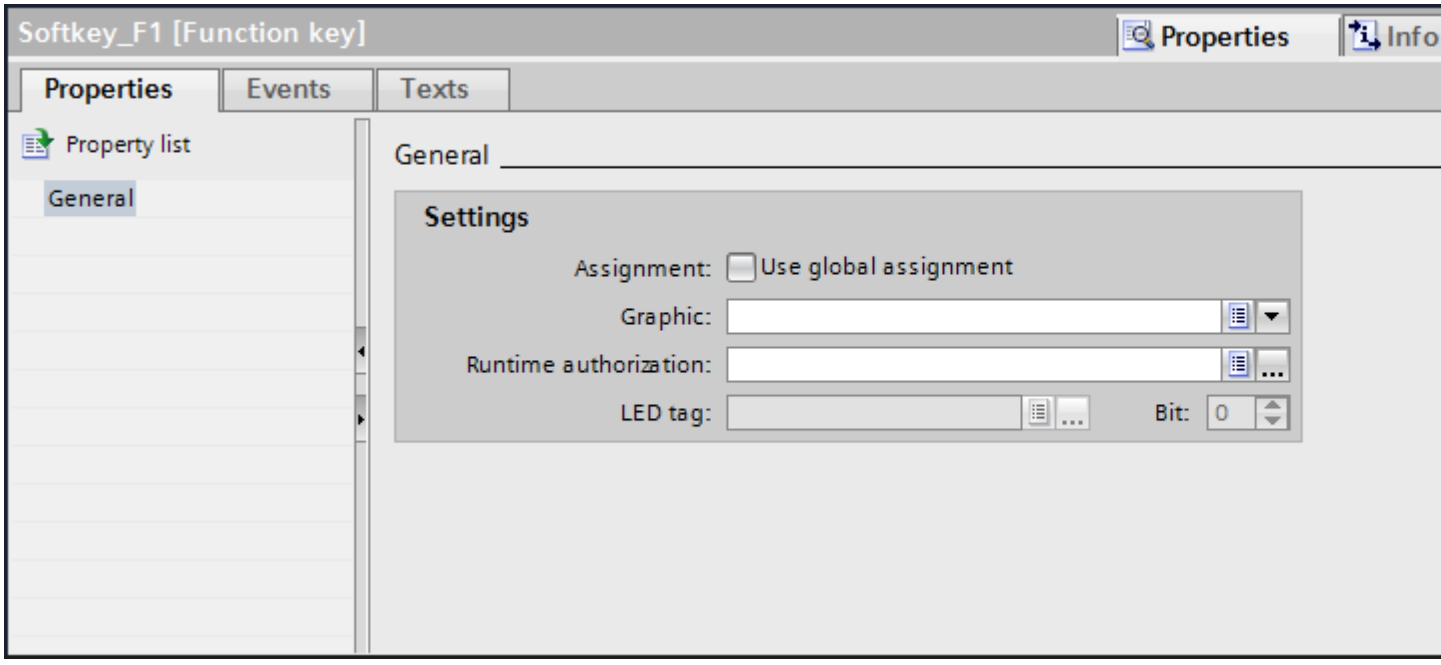
If you want to assign a graphic to a local function key in a screen or template:

- The screen or the template that contains the corresponding function key is open.
- The Inspector window is open.
- You have created the graphic for the function key.

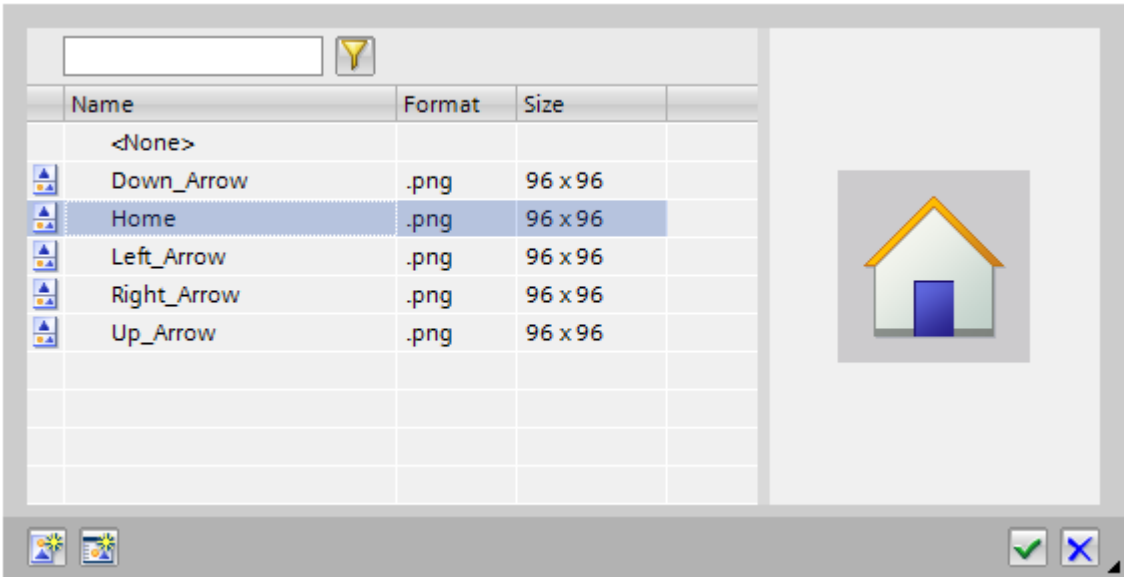
Procedure

Follow these steps:

1. Select the relevant function key.
   The properties of the function key are shown in the Inspector window.
2. Click "General" in the inspector window.



3. Click in the "Graphic" area of the list.
   The graphic browser of your WinCC project appears. The pane on the left side shows the external graphics which are stored in the browser. The pane on the right side shows you a preview of the graphic you have selected in the browser.



   Using the  and  icons, you can display the collection either in form of thumbnails or as a list.
   In order to open and edit OLE objects with the associated graphics program, double-click on the object.
4. In the browser click the desired graphic or store the relevant graphic in the graphic browser.
   The graphic preview is shown in the right pane.
5. Click "Select" to add the graphic to the screen.
   Click "Clear" to remove the graphic from the screen.

Result

The graphic is displayed next to the function key.

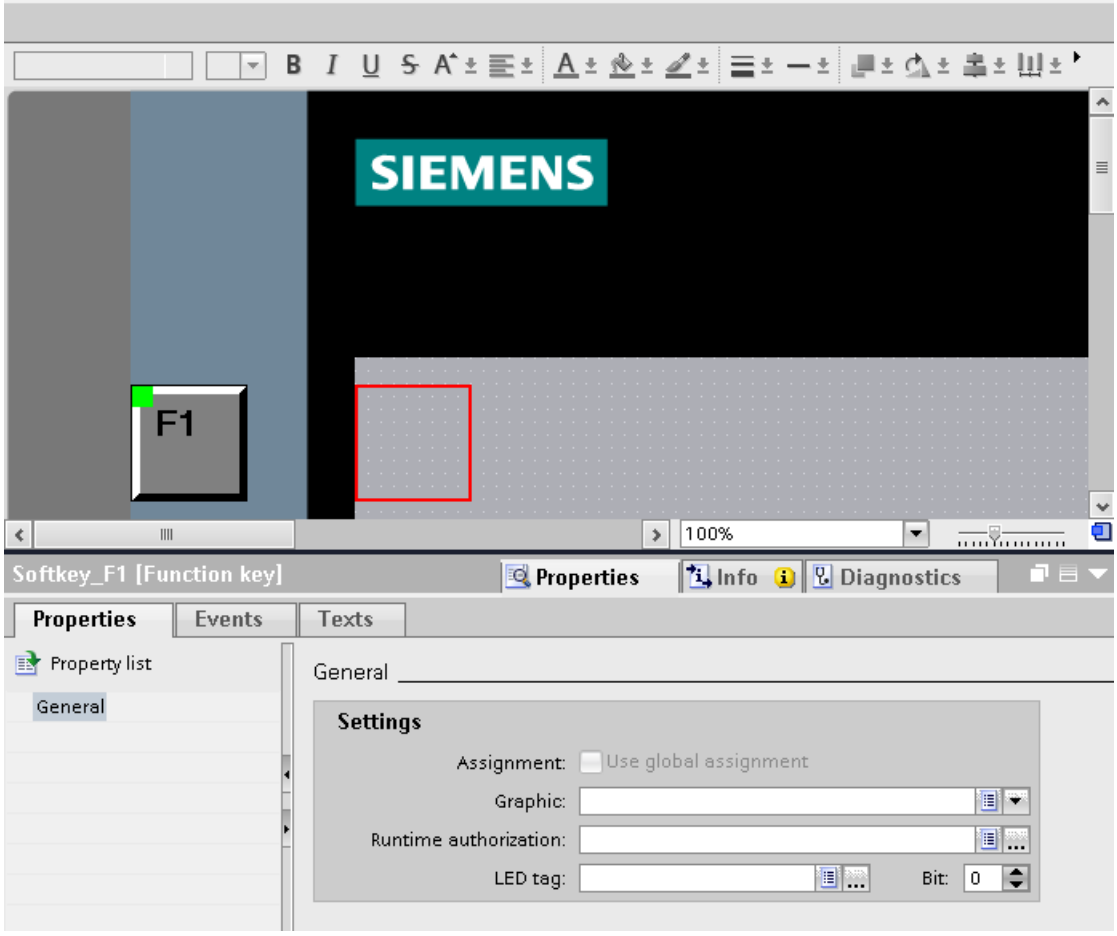
12.1.5.7 Configuring LED tags

Requirements

- An HMI device with key operation has been created.
- You have created an LED tag.
- The global screen is open.

Procedure

1. Click an F-key of the HMI device.
2. In the Inspector window, click "Properties > Properties > General".



3. Select a tag under "LED tag" in the "General > Settings" area.
4. Under "Bit" enter the correct bit number.
   The correct bit number depends on the HMI device and the input and output assignments on the HMI device.

12.1.5.8 Example: Using function keys for screen navigation

Task

In this example you create a local function key in a screen. When the operator presses this function key, a screen change to a predefined screen is triggered, for example "Boiler 2".

Note

Availability for specific HMI devices

Function keys are not available on all HMI devices.

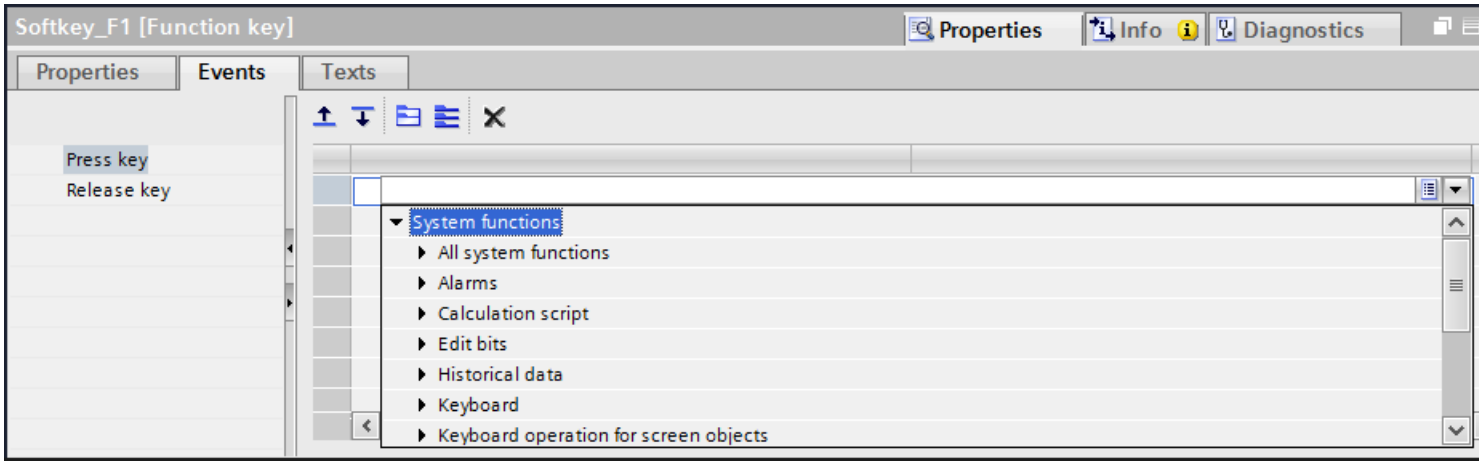
Requirement

- The screen in which you want to assign the function key is open.
- You have created the "Boiler 2" screen.
- The Inspector window is open.

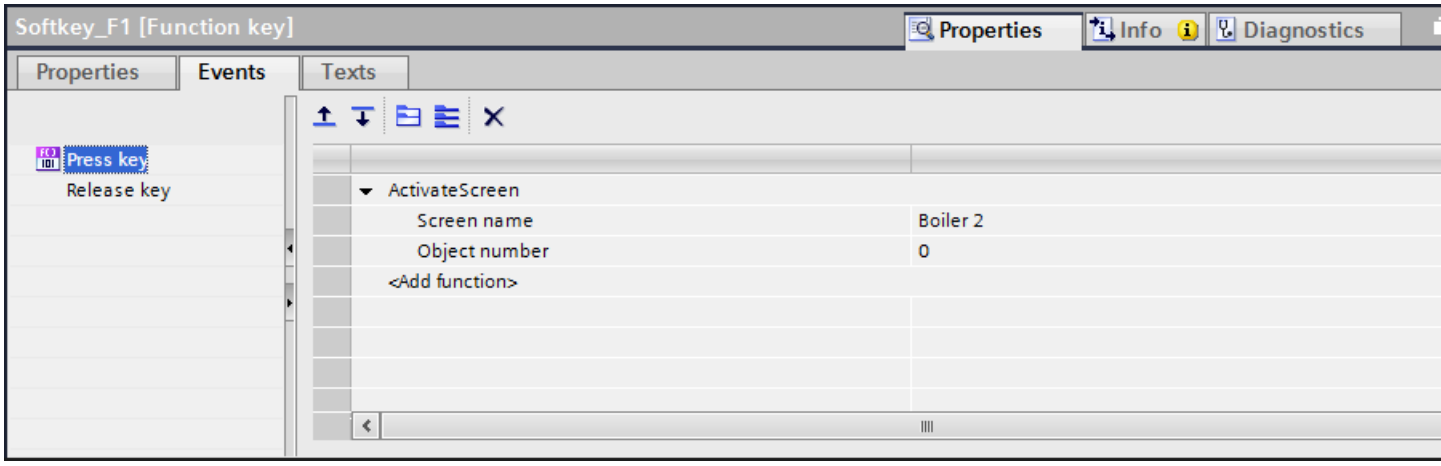
Procedure

Proceed as follows to use the "ActivateScreen" function:

1. Select the desired function key.
   The properties of the function key are shown in the Inspector window.
2. Click "General."
3. To overwrite a global assignment, disable the "Use local template" option.
4. Click "Key pressed" under "Events".



5. Select the "ActivateScreen" system function from the list.
   The "ActivateScreen" function appears in the "Function list" dialog box, including the "Screen name" and "Object number" parameters.



6. Select the "Boiler 2" screen from the "Screen name" list.

Result

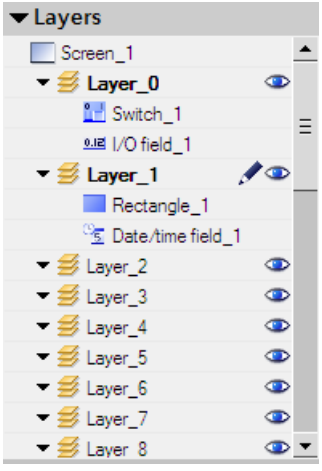
The operator changes to the "Boiler 2" screen in runtime by pressing the selected function key.

12.1.6 Working with layers

12.1.6.1 Basics on working with layers

Layers

Use layers in order to achieve differentiated editing of the objects in a screen. A screen consists of 32 layers that you can give any names. If you assign objects to the layers, you thereby define the screen depth. Objects of layer 0 are located at the screen background, while objects of layer 31 are located in the foreground.



The objects of a single layer are also arranged hierarchically. When you create a screen, the object inserted first is located at the rear within the layer. Each further object is placed one position towards the front. You can shift objects forwards and backwards within a layer.

Principle of the layer technique

Always one layer of the 32 layers is active. New objects you add to the screen are always assigned to the active layer. The number of the active level is displayed in the inspector window of the screen and in the "Layout > Layers" task card.

When you open a screen, all 32 layers of the screen are displayed. You can hide all the layers except for the active layer in the inspector window of the screen and in the "Layout > Layers" task card. You then explicitly edit objects of the active layer.

In the tree view of the "Layers" palette in the "Layout" task card, you administer layers and objects with drag-and-drop and the context menu.

Application examples

Use layers, for example, in the following cases:

- To hide the labeling of objects when editing,
- To hide objects, e.g. alarm windows, while configuring other objects

Restrictions on the objects in RT Professional

Some controls in WinCC RT Professional appear in a separate window in Runtime.

The following controls are displayed in Runtime regardless of the levels:

- Table view
- F(x) trend view
- F(t) trend view
- Value table
- Recipe view
- Alarm view
- .Net control
- HTML browser
- Screen window

The assignment of these controls to certain levels has no effect in Runtime.

12.1.6.2 Moving objects on layers

Introduction

By default, a new object is inserted on the active layer. You can, however, assign an object to another layer at a later time.

Requirement

- A screen with an object is open.
- The Inspector window is open.

Procedure

1. Select the object in the screen.
   The object properties are displayed in the Inspector window.
2. Enter the layer to which you want to move the object in "Properties > Properties > Miscellaneous > Layer" in the Inspector window.

Alternatively, select the object from the "Layout" task card and drag it to the required layer.

Changing the order of objects

1. Select the object in the screen.
   The object properties are displayed in the Inspector window.
2. To move the object to the front or back, select the "Order" > "Move backward" or "Move forward" command from the shortcut menu.
   Alternatively, use the  or  button in the toolbar.

Result

The object is assigned to the selected layer, and positioned at the top of the layer.

12.1.6.3 Setting the active layer

Introduction

The screen objects are always assigned to one of the 32 layers. There is always an active layer in the screen. New objects you add to the screen are always assigned to the active layer.

The number of the active layer is indicated in the "Layer" toolbar. The active layer is indicated by the icon in the "Layout > Layers" task card.

Layer 0 is the active layer when you start programming. You can activate a different layer during configuration, if necessary.

Requirement

- You have opened a screen which contains at least one object.
- The Inspector window of the active screen is open.

Procedure

1. Click "Properties > Properties > Layers" in the Inspector window of the current screen.
2. Enter the layer number in "Settings > Active layer".

Alternative procedure

1. Select "Layout > Layers" in the "Layout" task card.
2. Select the "Set to active" command from the shortcut menu of a layer.

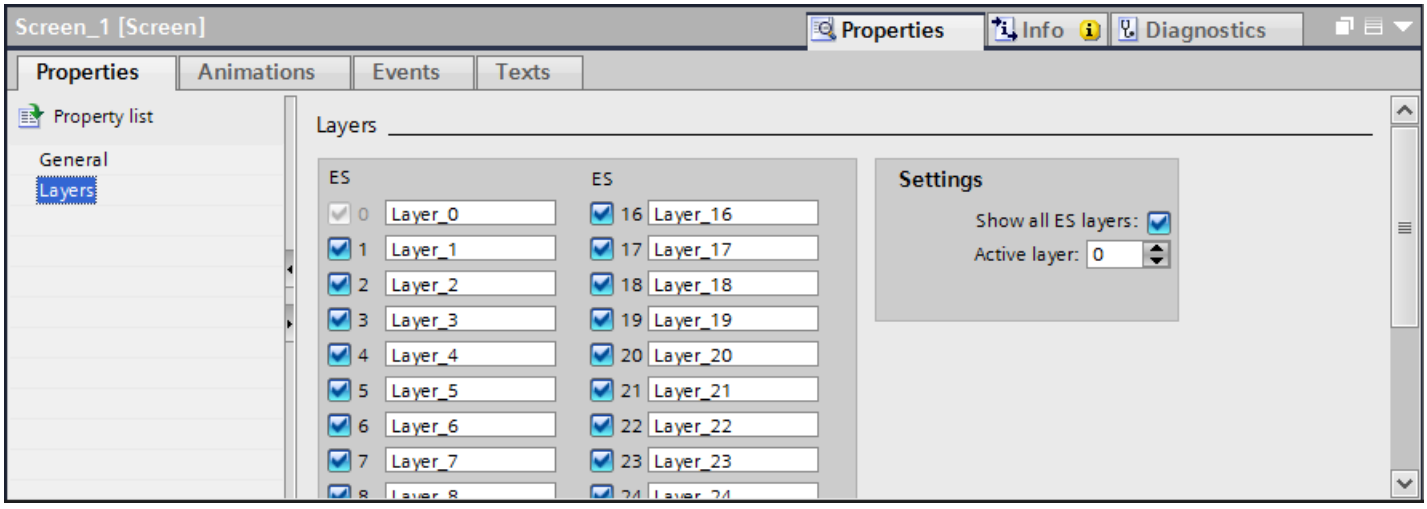
Result

The layer with the specified number is now active.

12.1.6.4 Showing and hiding layers (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You can show or hide the layers of a screen as required. You specify the layers that are shown in the Engineering System. When you open a screen, all the layers are always shown.



Requirement

- The screen is opened.
- The "Layout" task card is open.

Procedure

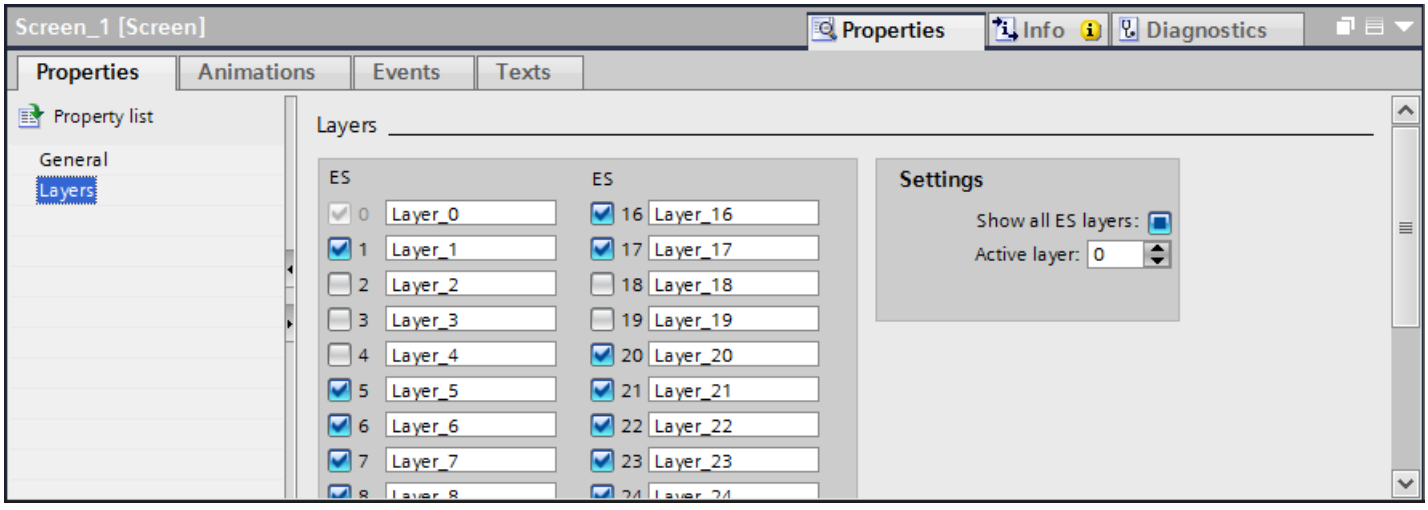
1. Select the layer that you want to hide or show in the "Layout > Layers" task card.
2. Click one of the icons next to the corresponding layer:
   - A shown layer is hidden
   - A hidden layer is shown

   Note

   The active layer cannot be hidden.

Alternative procedure

1. Click in an area of the screen that does not contain an object.
   The screen properties are shown in the Inspector window.
2. In the Inspector window, select "Properties > Properties > Layers":



3. In the list, disable the levels you wish to hide.
   If you activate "All ES layers" for a layer, the objects in this layer will be shown in the Engineering System.

Result

The layers are shown according to your settings.

12.1.6.5 Renaming layers

Introduction

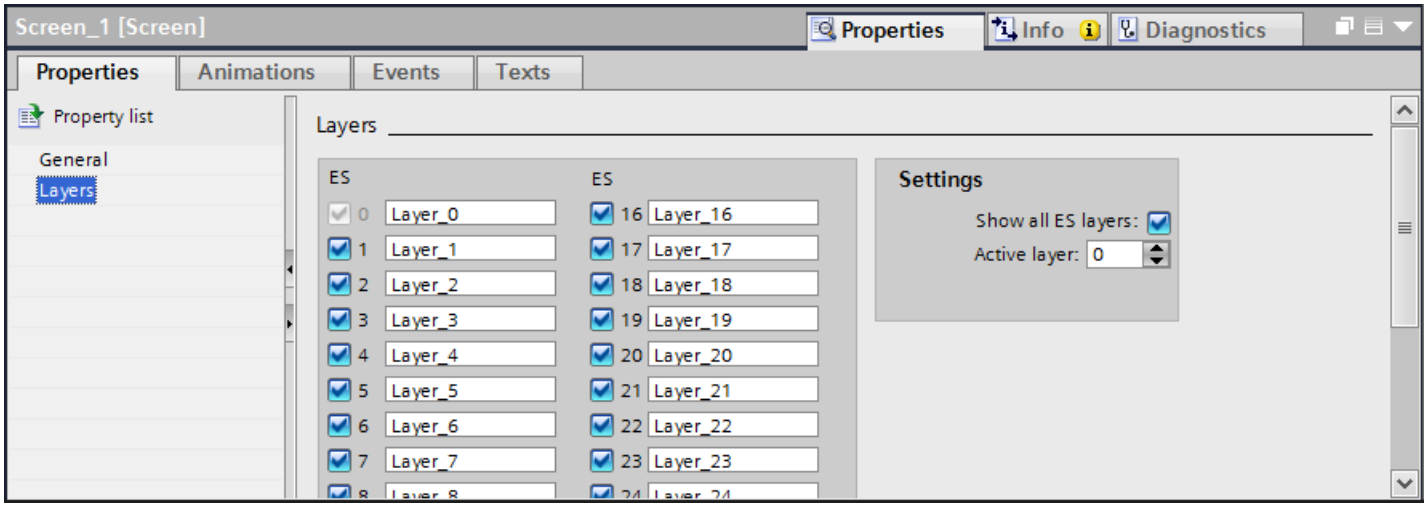
When you create a screen, the 32 layers are numbered consecutively by default. To improve clarity, you can rename the layers to suit your requirements.

Requirement

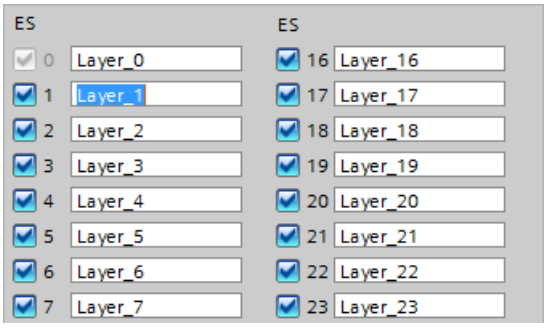
- The screen is opened.

Procedure

1. Click in an area of the screen that does not contain an object.
   The screen properties are shown in the Inspector window.
2. In the Inspector window, select "Properties > Properties > Layers".



3. Enter the new layer name.



Result

The layer is displayed with the new name.

12.1.7 Configuring screen navigation

12.1.7.1 Basics on screen navigation (Basic Panels, Panels, Comfort Panels, RT Advanced)

Types of navigation for the screen change

For a production process consisting of multiple subprocesses, you will configure multiple screens. You have the following options to enable the operator to switch from one screen to the next in Runtime:

- Assign buttons to screen changes
- Configuring screen changes at local function keys

Procedure

Before you create a screen change, define the plant structure and derive from it the screen changes that you want to configure.

Create the start screen under "Runtime Settings > General > Start screen".

12.1.7.2 Assigning screen change to button (Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Configure a button in the screen to switch between the screens on the HMI device during operation.

Note

If you have set the "Visibility" of animations to "Hidden" in the Inspector window of a screen, this screen cannot be called up in Runtime.

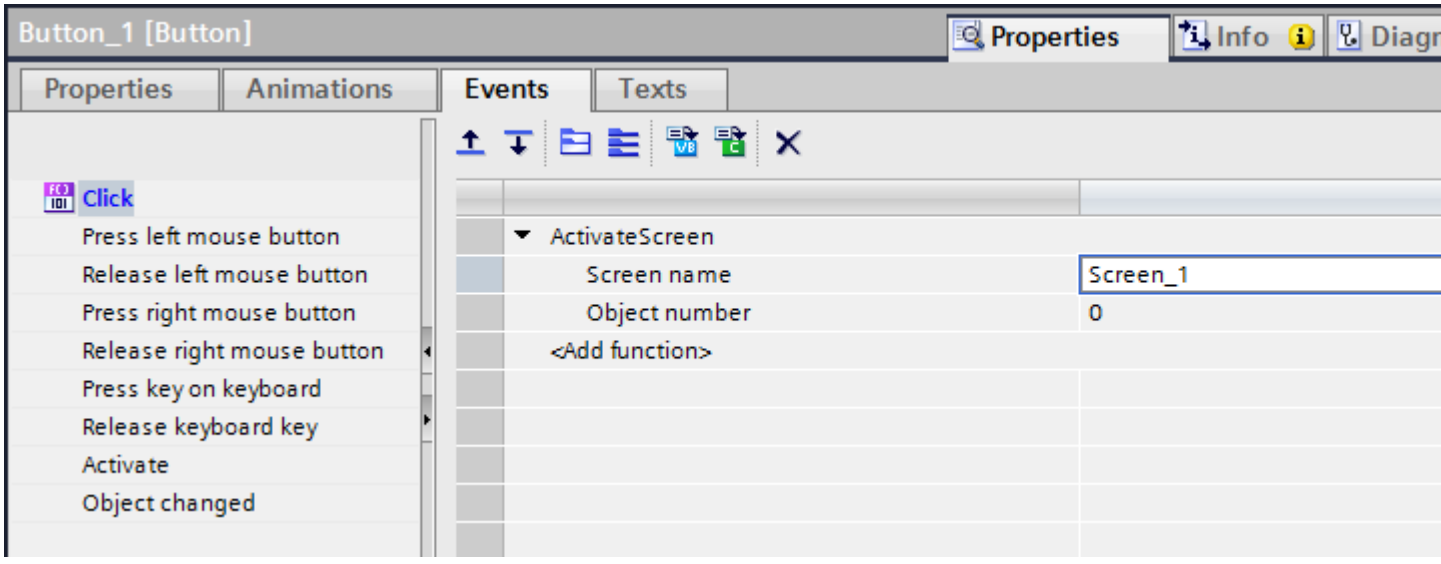
Requirements

- You have created the project.
- You have created the "Screen_2" screen.
- "Screen_1" is created.

Procedure

1. Double-click "Screen_1" in the project navigation. The screen is displayed in the work area.
2. Move "Screen_2" from the project tree to the open screen by drag&drop.
   A button with the name "Screen_1" is inserted.

3. In the Inspector window, select "Properties > Events > Click".
   The "ActivateScreen" system function is displayed in the "Function list".



4. At the "Object number" attribute, define, if required, the tab sequence number of the object on which the focus is to be set after a screen change. You can also specify a tag that contains the object number.

Alternative procedure

1. Move a button from the "Tools" task card to "Screen2" by drag&drop.
2. In the Inspector window, select "Properties > Events > Click".
3. Select the "ActivateScreen" system function.
4. Select "Screen_2" for the "Screen number".

Result

The operator goes to "Screen_1 with the button in Runtime. If you have specified an object number, the object with this object number has the focus following a screen change.

12.1.7.3 Assigning screen change to function key (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Configure a screen change function key in the screen to switch between the screens on the HMI device during operation.

Note

If you have set the "Visibility" of animations to "Hidden" in the inspector window of a screen, this screen cannot be called up in Runtime.

Requirements

- You have created a project.
- You have created the "Screen_2" screen.
- You have created the "Screen_1" screen.

Procedure

1. Double click "Screen_1" in the project tree. The screen is displayed in the work area.
2. Move "Screen_2" from the project tree to a function key, e.g. "F2".
   The configured function key displays a yellow triangle.
3. Click "Properties > Events > Press key" in the inspector window.
   The "ActivateScreen" system function is displayed.

Result

The operator goes to the specified "Screen_2" with function key "F2" in Runtime.

12.1.8 Display and operating elements

12.1.8.1 Device dependency of the objects

Objects for Basic Panels (Basic Panels)

Availability of display and operating objects for Basic Panels

Only the objects which can be used for the device you are configuring will be shown in the object window. The following table shows the availability of indicator and operating objects for the Basic Panels.

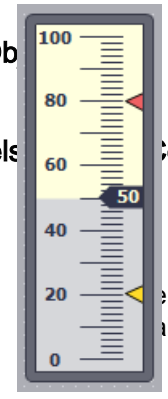
On devices with the device version earlier than V13, configure simple display objects. On devices with the device version V13 or later, configure table-based display objects.

Overview

	KP300 Basic KP400 Basic	KTP400 Basic KTP600 Basic KTP1000 Basic TP1500 Basic	KTP700 Basic PN KTP900 Basic	KTP700 Basic DP KTP1200 Basic DP	KTP1200 Basic PN
Bar	Yes	Yes	Yes	Yes	Yes
User view	Yes	Yes	Yes [1]	Yes [1]	Yes [1]
Date/time field	Yes	Yes	Yes	Yes	Yes
I/O field	Yes	Yes	Yes	Yes	Yes
Ellipse	Yes	Yes	Yes	Yes	Yes
Graphic view	Yes	Yes	Yes	Yes	Yes
Graphic I/O field	Yes	Yes	Yes	Yes	Yes
Help indicator	Yes	No	No	No	No
HTML browser	No	No	Yes	No	Yes
Circle	Yes	Yes	Yes	Yes	Yes
Trend view	Yes	Yes	Yes [1]	Yes [1]	Yes [1]
Line	Yes	Yes	Yes	Yes	Yes
Alarm view Alarm window	Yes	Yes	Yes [1]	Yes [1]	Yes [1]
Alarm indicator	Yes	Yes	Yes	Yes	Yes
Rectangle	Yes	Yes	Yes	Yes	Yes
Recipe view	Yes	Yes	Yes [1]	Yes [1]	Yes [1]
Button	Yes	Yes	Yes	Yes	Yes
Switch	Yes	Yes	Yes	Yes	Yes
Symbolic I/O field	Yes	Yes	Yes	Yes	Yes
System diagnostic view	Yes	Yes	Yes [1]	Yes [1]	Yes [1]
Text field	Yes	Yes	Yes	Yes	Yes

1) The objects are displayed on the panels as table-based objects.

12.1.8.2 Ob



Bar (Basic Panels omfort Panels, RT Advanced)

Use

displayed graphically using the "Bar" object. The bar graph can be labeled with lues.

Layout

In the Inspector window, you customize the settings for the position, shape, style, color, and font types of the object. You can adapt the following properties in particular:

- Color transition: Specifies the change in color display when limit values are exceeded.
- Displaying the limit lines / limit markers: Shows the configured limit as a line or marker.
- Define bar segments: Defines the gradations on the bar scale.
- Define scale gradation: Defines the subdivisions, scale markings and intervals of a bar scale.

Color transition

You define how the color change is represented in "Properties > Properties > Appearance" in the Inspector window.

Color transition	Description
"Segmented"	If a particular limit was reached, the bar changes color segment by segment. With segment by segment representation, you visualize, for example, which limits are exceeded by the displayed value.
"Entire bar"	If a particular limit was reached, the entire bar changes color.

Displaying limit lines and limit markers

You display the configured limit in the bar as a line or marking in Runtime using the "Lines" and "Markings" property:

1. In the Inspector window, select "Properties > Properties > Appearance":
2. Activate "Lines" and "Markings".

Define bar segments

Use the "Subdivisions" property to define the number of segments into which the bar is divided by the main gradations on the scale.

Use the "Interval" property to divide the distance between the main gradations. The value appears as the difference in value between two adjacent main gradations:

1. In the Inspector window, select "Properties > Properties > Scales":
2. Activate "Show scale."
3. Select the corresponding value for "Settings > Subdivisions".
4. Select the corresponding value for "Settings > Marks label".
5. Select the corresponding value for "Large interval > Interval".

Configuring limits/ranges

You can represent limits and ranges in different colors. Define the colors in the Inspector window.

1. In the Inspector window, select "Properties > Properties > Limits/Ranges".
2. Activate the limits/ranges that are to be displayed in Runtime
3. If necessary, change the default colors for the limits/ranges.

Note

If you select the option "Show ranges of tags", you can display up to five ranges in a slider with values that can be specified via a process tag. You define the values for the ranges with a process tag, which you interconnect with the screen object.

The "Show ranges of tags" option is available for Comfort Panels, KTP Mobile Panels and RT Advanced.

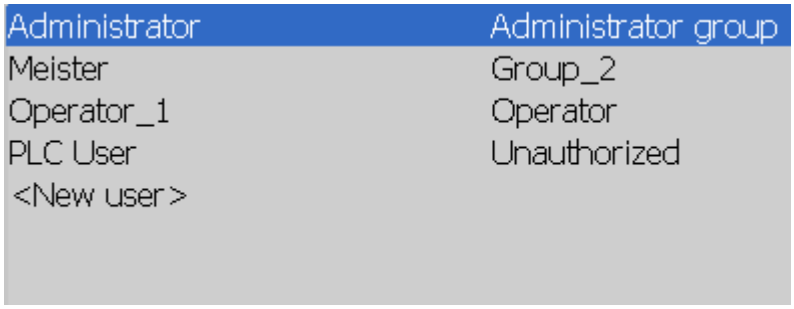
Note

If the "Show ranges of tags" option is enabled, the setting "Color gradient Segmented" under "Properties > Appearance > Bar" has no effect in Runtime.

User view (Basic Panels, Panels, Comfort Panels, RT Advanced)

Application

The "User view" object is used to set up and administer users and authorizations.



Note

Do not use the user view in a group.

Note

The "Simple user view" object cannot be operated dynamically with a script.

Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object. You can adapt the following properties in particular:

- Number of lines: Specifies the maximum number of visible entries.

Number of lines

The number of lines in the user view displayed in Runtime is specified in the Inspector window. The setting for the number of lines is only effective if the property "Fit object to contents" is active.

1. In the Inspector window, select "Properties > Properties > View".
2. Enter an integer value under "Number of lines".
3. In the Inspector window, activate "Properties > Properties > Layout".
4. Select "Fit object to contents".

Display in Runtime

The appearance depends on the authorizations:

- All users on the HMI device are displayed in the User view to the administrator or to a user with administrator authorizations.
- For a user without administrator rights, only the user's own entry is shown.

Operation

Depending on the configuration you can:

- Manage users, e.g. create, delete.
- Change existing user data.
- Export or import user data.

Note

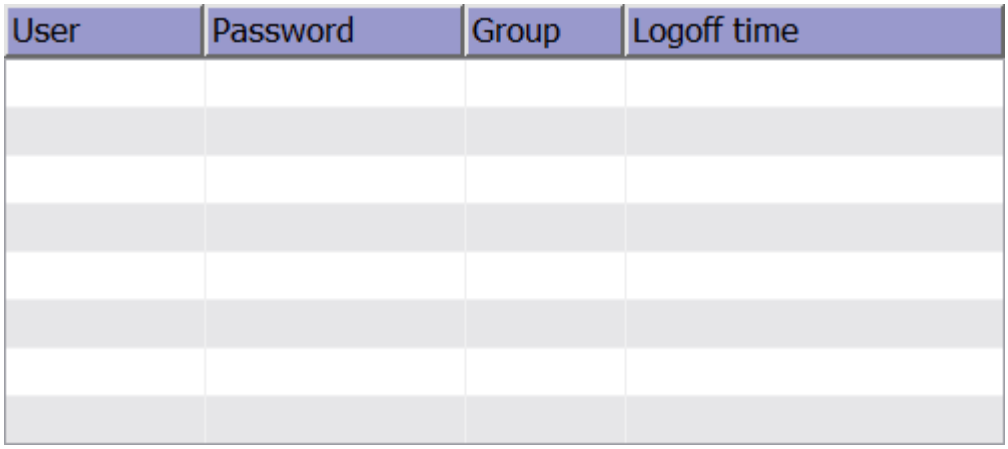
The number is limited to 100 users and one PLC user on an HMI device. This restriction does not apply to PCs. On a PC, the maximum number of users is restricted by the physical memory.

User view (as of V13) (Basic Panels, Panels, Comfort Panels, RT Advanced)

Use

The "User view" object is used to set up and administer users and authorizations. The user view is used, for example, to create new users in Runtime and assign them to a user group.

On HMI devices with device version V13 or later, the table-based user view is available for managing users and authorizations.



Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object. You can adapt the following properties in particular:

- Number of lines: Specifies the maximum number of visible entries.
- Columns moveable: Specifies whether the operator can change the sequence of columns in Runtime.

In addition, you set the borders, the fill pattern and the colors of the table header in the Inspector window.

Number of lines

The number of lines in the user view displayed in Runtime is specified in the Inspector window. The setting for the number of lines is only effective if "Fit object to contents" is active.

1. Click "Properties > Properties > View" in the Inspector window.
2. Enter an integer value under "Number of lines".
3. Click "Properties > Properties > Layout" in the Inspector window.
4. Select "Fit object to contents".

Columns moveable

The operator can change the sequence of columns in Runtime with the "Columns moveable" property.

1. Click "Properties > Properties > Layout" in the Inspector window.
2. Activate "Columns moveable".

Column width

In the Inspector window, you can change the width of columns displayed in Runtime.

1. Click "Properties > Properties > Layout" in the Inspector window.
2. Enter integer values for the column width under "Columns".

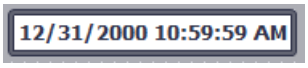
Note

The "Password" column is a dynamic column which adapts to the outer dimensions of the object so that the object content occupies the entire width. The column width you assign is applied as the minimum size. In a dynamic column, the minimum size is changed; however, the width is displayed dynamically to fill the display.

Date/time field (Basic Panels, Panels, Comfort Panels, RT Advanced)

Use

The "Date/time field" object shows the system time and the system date. The appearance of the "Date/time field" depends on the language set on the HMI device.



Layout

In the Inspector window, you customize the position, style, colors and font types of the object. You can adapt the following properties in particular:

- Display system time: Specifies that the system time is displayed.
- Include tag: Specifies that the time of the connected tag is displayed.
- Long date/time format: This setting defines the format displayed for the data and time.

Display system time

The time displayed in the "Date/time field" on the HMI device is specified in the Inspector window.

1. In the Inspector window, select "Properties > Properties > General".
2. Select "Format > System time".

Using tags

The time of the interconnected tag is displayed in the date/time field.

1. In the Inspector window, select "Properties > Properties > General".
2. In the "Format" area, select a tag with the "DateTime" data type, e.g. an internal tag.

Long date/time format

Visualization of the date and time is specified in the "Format" area under "General" in the Inspector window.

Option	Description
"Enabled"	Date and time are displayed in full, e.g. "Sunday, December 31, 2000 10:59:59 AM"
"Disabled"	Date and time are displayed in short form, e.g. "12/31/2000 10:59:59 AM"

Behavior during operation

If the operator ignores the syntax when entering values or enters invalid values in Runtime, these entries will not be applied. Instead, the original value (plus the time that has elapsed in the meantime) appears in the date/time field and a system event is displayed on the HMI device.

I/O field

Application

The "I/O field" object is used to enter and display process values.



Layout

In the Inspector window, you customize the position, shape, style, color and font types of the object. You can adapt the following properties in particular:

- Mode: Specifies the response of the object in Runtime.
- Display format: Specifies the display format in the I/O field for input and output of values.
- Hidden input: Specifies whether the input value is displayed normally or encrypted during input.

Note

Reports

In reports, I/O fields only output data. "Output" mode is preset. Properties for configuring input are not available, e.g. "hidden input".

Mode

The response of the I/O field is specified in the Inspector window in "Properties > Properties > General > Type".

Mode	Description
"Input"	Values can only be input into the I/O field in Runtime.
"Input/output"	Values can be input and output in the I/O field in Runtime.
"Output"	The I/O field is used for the output of values only.

Layout

The "display format" for the input and output of values is specified in "Properties > Properties > General > Format" in the Inspector window.

Layout	
"Binary"	Input and output of values in binary form
"Date"	Input and output of date information. The format depends on the language setting on the HMI device.
"Date/time"	Input and output of date and time information. The format depends on the language setting on the HMI device.
"Decimal"	Input and output of values in decimal form
"Hexadecimal"	Input and output of values in hexadecimal form
"Time"	Input and output of times. The format depends on the language setting on the HMI device.
"Character string"	Input and output of character strings.

Note

Data formats

Not all data formats are available for selection for Runtime Professional.

Hidden input

In Runtime the input can be displayed normally or encrypted, for example for hidden input of a password. A "*" is displayed for every character during hidden input. The data format of the value entered cannot be recognized.

1. In the Inspector window, select "Properties > Properties > Response":
2. Select "Hidden input".

Avoid overlaps in output fields

If several I/O fields are configured as output fields with a transparent background in a screen, these I/O fields may overlap. The transparent part of the one field covers the digits of the other field. This may cause display problems in Runtime. In order to avoid such overlaps, set the borders of the I/O fields to zero in the object properties under "Properties > Properties > Appearance". Select "Properties > Properties > Layout > Fit object to contents."

Limits

Under "Properties > Properties > Limits" in the Inspector window, set the colors for the values that violate the high or low limits.
You define the limits via the properties of a tag. In Runtime Professional you can specify if the colors should be used.

When there is a limit violation during Runtime, the background color of the I/O field changes according to your configuration, even if the I/O field is in "Input" mode.

In Runtime Professional, you can also define the limit range for the entry in the I/O field via "Properties > Properties > Limits".

If you enter a numeric value outside this limit, it is not applied; for example, 80 with a limit of 78. In this case, the HMI device generates a system event, if an alarm window is configured. The original value is displayed again.

Decimal places for numerical values

The configuration engineer can define the number of decimal places for a numerical input field. The number of decimal places is checked when you enter a value in this type of I/O field. Decimal places in excess of the limit are ignored. Empty decimal places are filled with "0".

Behavior when switching between input fields

When you change to another input field within the same screen, and the screen keyboard appears, the "Exit field" event is not executed for the previous field unless you close the screen keyboard.

Ellipse (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Application

The "Ellipse" is an enclosed object that can be filled with a color or pattern.



Layout

In the Inspector window you can customize the settings for the object position, geometry, style, frame and color. You can adapt the following properties in particular:

- Horizontal radius: Specifies the horizontal radius of the elliptical object.
- Vertical radius: Specifies the vertical radius of the elliptical object.

Horizontal radius

The horizontal radius of the "Ellipse" object is specified in the Inspector window. The value is entered in pixels.

1. In the Inspector window, select "Properties > Properties > Layout".
2. Enter a value between 0 and 2500 under "Horizontal."

Vertical radius

The vertical radius of the "Ellipse" object is specified in the Inspector window. The value is entered in pixels.

1. In the Inspector window, select "Properties > Properties > Layout".
2. Enter a value between 0 and 2500 at "Vertical."

Graphic view (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Use

The "Graphic view" object is used to display graphics.



Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object. You can adapt the following properties in particular:

- Graphic: Specifies the graphic file that is displayed in the object.
- Adjust graphic: Specifies the automatic size stretching for objects with graphics.
- Transparent color: Specify whether or not the transparent color is used for the graphic.

Inserting graphics

Use the following image formats in the "Graphic Display" object: *.bmp, *.tif, *.png, *.ico, *.emf, *.wmf, *.gif, *.jpg or *.jpeg. You may also use graphics as OLE objects in the Graphic view .

1. In the Inspector window, select "Properties > Properties > General".
2. Select the graphic that you wish to insert.
   The graphic preview is shown in the right pane.
   You have the option to create the graphic from a file using the  button, or to create a new graphic from an OLE object using the  button.
3. Click "Apply" to insert the graphic in the Graphic view .

Adjust graphic

Whether a graphic displayed in a Graphic view is stretched to the size of the Graphic view in runtime is specified in the Inspector window.

Option	Description
"No auto-sizing"	The size is not automatically adapted in Runtime.
"Fit graphic to object size"	The graphic is automatically adapted to the object size in Runtime.
"Fit object size to graphic"	Graphic display is adapted to the largest used graphic and other graphics are stretched.
"Adjust object size to graphic"	Graphic display is adapted to the largest used graphic but the other graphics retain their original size.

1. Click "Properties > Properties > Layout" in the Inspector window.
2. Select the required size adjustment for the graphic.

Transparent color

This property defines whether the transparent color is used for the graphic to be displayed.

1. In the Inspector window, select "Properties > Properties > Appearance":
2. Select "Background > Transparent".
3. Select a transparent color.

Note

When using bitmaps in WinCC screens the "Transparent color" setting demands a high character performance in the layout on the panel. Visualization performance is enhanced by disabling the "Transparent" setting in the properties of the relevant display object. This restriction applies in particular when bitmaps are used as background image.

Note

Basic Panels

The "Transparent" property is not available for Basic Panels.

Note

If the selected transparent color of the graphic is not shown properly although the "Transparent" property is activated, the color in the graphic is not entirely consistent with the selected color. For example, the selected transparent color [R=255;G=255;B=255] looks almost exactly like the color in the graphic that is expected to be transparent [R=255;G=251;B=255]. In this case, the graphics used in the object must be reedited. Under "Project Navigation > Languages & Resources > Graphics", select the graphic that you want to display transparently. From the shortcut menu of the default graphic, select the "Edit" command and fill in the background of the graph with the desired color that appears transparent in the screen. Save the graphic.

Graphic I/O field (Basic Panels, Panels, Comfort Panels, RT Advanced)

Application

The "Graphic I/O field" object can be used to configure a list for display and selection of graphic files.



Layout

In the Inspector window, you customize the position, shape, style, color and font types of the object. You can adapt the following properties in particular:

- Scroll bar type: Specifies the graphic layout of the scroll bar.
- Mode: Specifies the response of the object in Runtime.

Note

Scroll bar

The "Scroll bar > Type" property is available for Panels and Runtime Advanced up to device version V12.

Note

Reports

Graphic I/O fields output exclusively graphics in reports. "Output" mode is preset. Properties for configuring the selection of graphics are not available, e.g. "scroll bar".

Note

Border

You can configure the width and the style of the border of a graphic I/O field in the "Two states" mode.

Mode

The response of the "Graphic I/O field" object is specified under "Properties > Properties > General > Type > Mode" in the Inspector window.

Mode	Description
"Input"	The "Graphic I/O field" object is only used to select graphics.
"Input/output"	The "Graphic I/O field" object is used to select and display graphics.

Mode	Description
"Output"	The "Graphic I/O field" object is used to display graphics only.
"Two states"	The "Graphic I/O field" object is only used to display graphics and can have a maximum of two states. You use no graphics list but insert one graphic each for the "ON" and "OFF" state.

Adjust graphic

Whether a graphic displayed in a graphic I/O field is stretched to the size of the view in Runtime is specified in the Inspector window.

1. Click "Properties > Properties > Layout" in the Inspector window.
2. Select the required size adjustment for the graphic.

Scroll bar type

The response for the graphic representation of the scroll bar is specified under "Properties > Properties > Appearance > Scroll Bar > Type" in the Inspector window.

Type	Description
"Permanent"	The scroll bar is always visible.
"No scrollbar"	The scroll bar is not visible.
"Visible after clicking"	The scroll bar is made visible by a mouse click.

Behavior during operation

If the graphic I/O field displays a cactus image, a graphic to be output for a specific value has not been defined in the project. The contents of the graphic I/O field change color to show that it is now activated. The frame in 3D is only shown graphically in an output field.

Help indicator (Basic Panels)

Application

The object "help indicator" is available for the HMI devices OP 73 and KP300 Basic. If a a tooltip exists for the selected object, the help indicator is displayed during runtime. If a tooltip was configured for the opened screen, the help indicator always remains visible.



You configure the object "help indicator" exclusively in the global screen.

Layout

You can adapt the following properties in the Inspector window:

- Position: Determines the position of the object "Help indicator."

Position

You can use this property to set the position of the object "Help indicator."

1. Select the object "Help indicator" in the template.
2. In the Inspector window, select "Properties > Properties > Layout".
3. Enter a value for X and Y. You can also use the cursor keys to position the selected object.

If you have configured a screen object at this position, the visible help indicator covers the screen object. The help indicator is covered only by incoming system alarms and dialogs.

Circle (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Application

The "Circle" object is a closed object which can be filled with a color or pattern.



Layout

In the Inspector window you can customize the settings for the object position, geometry, style, frame and color. You can adapt the following properties in particular:

- Radius: Specifies the size of the circle.

Radius

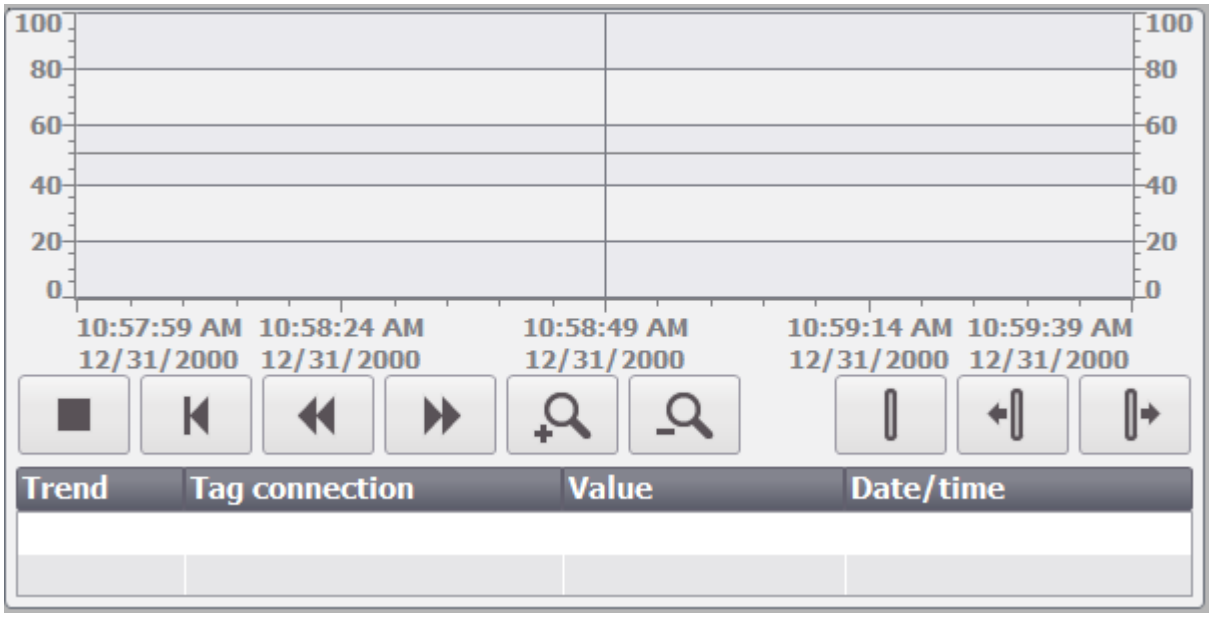
The radius of the "Circle" object is specified in the Inspector window. The value is entered in pixels.

1. In the Inspector window, select "Properties > Properties > Layout".
2. Enter a value between 0 and 2500 in the "Radius" area.

Trend view (Basic Panels, Panels, Comfort Panels, RT Advanced)

Application

The trend view is meant for the graphical representation of tag values from the current process or from a log in form of trends.



Layout

In the Inspector window, you customize the position, geometry, style, color, and font types of the object. You can adapt the following properties in particular:

- Display value table, ruler and grid: Specifies whether a value table, a ruler or a grid is displayed in addition to the coordinate system to improve legibility.
- Toolbars: Defines the display of the operator controls.

Display value table, ruler and grid

For improved legibility a value table, a ruler and a grid can be displayed in Runtime.

1. Activate "Properties > Properties > Appearance > Show ruler".
2. Activate "Properties > Properties > Table > Show table".
3. Activate "Properties > Properties > Table > Show grid".

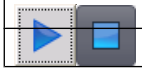
Toolbars

The layout of the operator controls is defined in the "Properties > Properties > Toolbar" inspector window.

Note

Basic Panels

Because archiving with Basic Panels is only possible for devices as of the 2nd Generation, the operator controls are not available on all Basic Panels.

Toolbar button	Brief description	Description
	"Go to start"	Scroll to the current value of the trend.
	"Zoom in"	Reduces the displayed time slice.
	"Zoom out"	Zooms into the displayed time section.
	"Ruler backward"	Moves the ruler back.
	"Ruler forward"	Moves the ruler forward.
	"Backward"	Scrolls back one display width.
	"Forward"	Scrolls forward one display width.
	"Ruler"	Shows or hides the ruler. The ruler displays the X-value associated with a Y-value.
	"Start/stop"	Stops trend recording or continues trend recording.

Operation on Basic Panels and devices with small display size

The trend view buttons are not displayed on Basic Panels and HMI devices with a display size of less than 6". You can operate the trend view using the function keys of the HMI device that are assigned corresponding system functions.

Displaying column headers

The layout of the table in the trend view depends on the view settings in the Control Panel. Depending on the setting, the column headers may be truncated. This setting is found under "Display > Appearance" in the control panel. To display column headers correctly, set the display in "Windows and buttons" to "Windows Classic" style.

This behavior only occurs during configuration. The column headers are displayed correctly in Runtime.

Consistency test

If warnings or errors are displayed in the output window during a consistency check in connection with trend views, clicking "Go to Error/Tag" on the shortcut menu will not always take you to the exact error position. In some cases only the trend view is shown as cause of error.

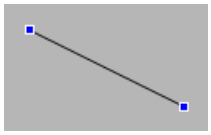
Adding, configuring, and removing trends

The trends of the trend view are managed in the Inspector window under "Properties > Properties > Trend". You can copy trends between different trend views.

Line

Application

The "Line" object is an open object. The line length and gradient slope are defined by the height and width of the rectangle enclosing the object.



Layout

In the Inspector window, you customize the settings for the object position, shape, style, and color. You can adapt the following properties in particular:

- Line style
- Line start and end

Line style

The representation of the line is specified under "Properties > Properties > Appearance" in the Inspector window. The line is shown without interruption if you select "Solid", for example.

Note

The line styles available depend on the selected HMI device.

Line start and end

The start and end points of the line are specified under "Properties > Properties > Appearance > Line ends" in the Inspector window.

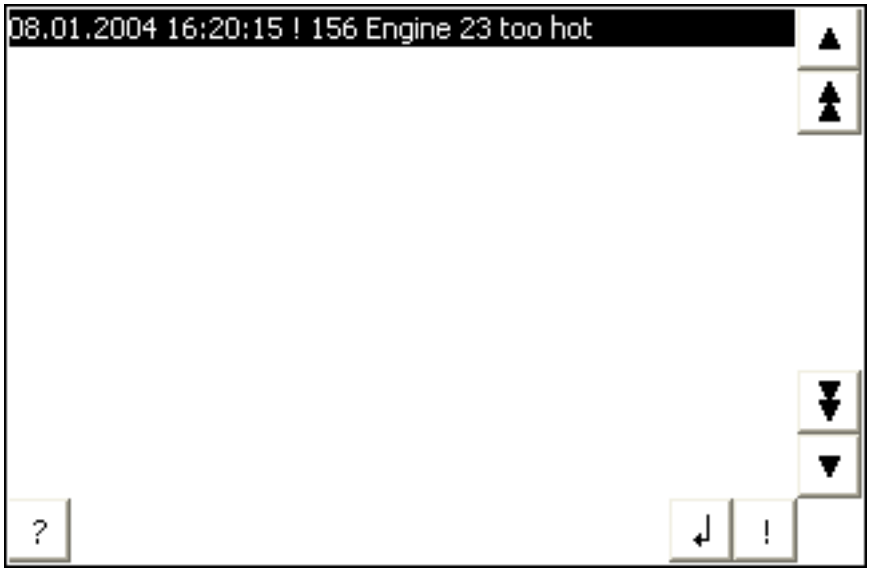
Use arrow point, for example, as start and end point. The available start and end points depend on the device.

Alarm view (Basic Panels, Panels, Comfort Panels, RT Advanced)

Use

Alarms are indicated in the Alarm view or in the Alarm window of the HMI device.

The following screen contains a simple alarm view:



Note

The "Single alarm view" object cannot be operated dynamically with a script.

Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object.

Note

The fonts available for selection depend on the fonts you have configured in the Runtime settings under "Language & font" and the fonts supported by your HMI device.

You can adapt the following properties in particular:

- Operator controls: Defines the operator controls of the alarm view.
- Alarm classes: This setting defines which alarm classes are displayed in the alarm view.
- Columns: Specifies the displayed columns in Runtime.

  Note

  If you have different alarm classes output, these will be initially sorted into alarm classes in Runtime, and then by the time at which the alarm occurred.

Operator controls

The operator controls that can be used to control the alarm display in Runtime are specified in the Inspector window under "Display > Settings". The following table shows the operator controls in the alarm view, and what they do:

Button		Function
"Info text"	?	Displays info text for an alarm.
"Acknowledge"	!	Acknowledges an alarm.
"Loop-In-Alarm"	↵	Executes the configured function (for example: "ActivateScreen") with each keystroke. It is hereby possible to call a screen containing your information about the selected alarm. Also acknowledges an alarm requiring acknowledgment. [1]

[1] Only if a function is configured for the selected alarm for the "loop-in-alarm" event.

Select alarm classes

1. Click "Properties" in the Inspector window.
2. Under "Alarm classes", activate the alarm classes to be displayed in the alarm view in Runtime.

Define columns

Define the columns to be displayed in the alarm view in Runtime in the Inspector window.

1. In the Inspector window, click "Properties > Columns".
2. Activate the columns that are to be displayed in Runtime under "Columns".

Displaying column headers

The layout of the alarm view is dependent on the view settings in the control panel. Depending on the setting, the column headers may be truncated. This setting is found under "Display > Layout tab" in the control panel. To display column headers correctly, set the display in "Windows and buttons" to "Windows Classic" style.

This behavior only occurs during configuration. The column headers are displayed correctly in Runtime.

Note

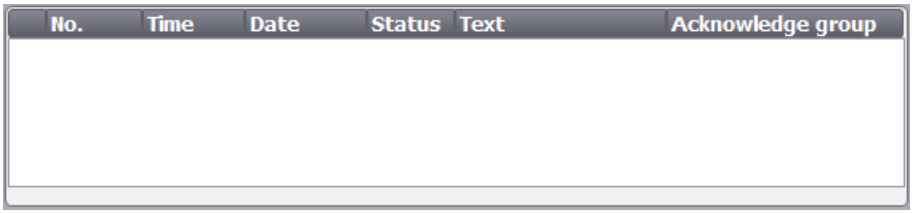
In the engineering system, you can dynamically control the visibility of an object, for example, in the "Animations" group of the Inspector window. In Runtime, the "Simple alarm view" does not support animations. If you have configured an animation and, for example, wish to perform a consistency check of the project, then an error alarm is issued in the Output window.

Alarm view (as of V13) (Basic Panels, Panels, Comfort Panels, RT Advanced)

Use

Advanced alarm view is used to display alarms on the HMI device.

On HMI devices with device version V13 or later, the advanced table-based alarm view is available for managing alarms.



Alarm line

The alarm line allows display of the most current pending alarm in Runtime. The following figure shows an alarm line:



How to configure the alarm line:

1. Click "Properties > Properties > Layout" in the Inspector window.
2. Select "Mode > Alarm line".

Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object.

Note

The fonts available for selection depend on the fonts you have configured in the Runtime settings under "Language & font" and the fonts supported by your HMI device.

You can adapt the following properties in particular:

- Toolbar: Defines the operator controls of the alarm view.
- Alarm classes: This setting defines which alarm classes are displayed in the alarm view.
- Columns: Specifies the displayed columns in Runtime.
- Columns moveable: Specifies whether the sequence of columns can be changed in Runtime.
- Identification of alarm classes: To be able to distinguish various alarm classes, they are identified in the first column of the alarm view.

- Filter: Defines that only alarms which contain a specific character string in the alarm text will be displayed.
- Define display area: This setting specifies the date from which alarms are displayed in the alarm view.
- Sorting by date/time possible: Specifies whether the alarms can be sorted in Runtime by date and time.

Note

If you have different alarm classes output, these will be initially sorted into alarm classes in Runtime, and then by the time at which the alarm occurred.

Operator controls

The operator controls that can be used to control the alarm view in Runtime are specified in the Inspector window under "Properties > Properties > Toolbar". The following table shows the operator controls in the alarm view and their function:

Button	Name	Function
	"Tooltip"	Displays a tooltip for an alarm.
	"Acknowledge"	Acknowledges an alarm.
	"Loop-in-alarm"	Executes the configured function (for example: "ActivateScreen") with each keystroke. It is hereby possible to call a screen containing your information about the selected alarm. Also acknowledges an alarm requiring acknowledgment. [1)]

[1)] Only if a function is configured for the selected alarm for the "loop-in-alarm" event.

Select alarm classes

1. Click "Properties > Properties > General" in the Inspector window.
2. Under "Alarm classes", activate the alarm classes to be displayed in the alarm view in Runtime.

Access protection in Runtime

You can configure access protection in the "Properties > Security" group in the properties of the alarm view. A logged-on user with the required authorization can acknowledge and edit alarms using the operator controls in the alarm view. If the logged-on user does not have the required authorization, or if no user is logged in, pressing the "Acknowledge" or "Edit" buttons or double-clicking an alarm line opens the logon dialog.

Define columns

Define the columns to be displayed in the alarm view in Runtime in the Inspector window.

1. In the Inspector window, select "Properties > Properties > Columns".
2. Activate the columns that are to be displayed in Runtime under "Visible columns".

Sequence of columns

If this property is enabled, the column sequence in the alarm view can be changed in Runtime.

1. In the Inspector window, select "Properties > Properties > Columns".
2. Select "Column properties > Columns moveable".

Sorting

If this property is enabled, the alarms displayed in the alarm view in Runtime can be sorted by date and time.

1. In the Inspector window, select "Properties > Properties > Columns".
2. Select "Column properties > Sorting by date/time possible".

Filter alarms

This property is used to define that only alarms that contain a configured string in the alarm text will be displayed in Runtime for the extended alarm view.

1. In the Inspector window, select "Properties > Properties > Filter".
2. In the "Filter string" field, enter the desired term for filtering.
   Alternatively, you can configure a filter tag using the "Filter tag" field. The contents of the filter tag serve as the filter criterion in Runtime.

Identify alarm classes

An icon is displayed in the first column of the alarm view. This symbol allows the alarm to be allocated to an alarm class.

1. In the Inspector window, select "Properties > Properties > Columns".
2. Select "Visible columns > Alarm class name".
3. Open the "HMI alarms" editor and click on the "Alarm classes" tab.
4. Specify an icon for use to identify the alarms of this alarm class for an alarm class in the "Display name" column.

Note

The "alarm view" object cannot be grouped.

Set colors for alarm classes

You have the option of having alarms of selected alarm classes shown in different colors in Runtime. You set the background colors of the alarms in the "HMI alarms" editor.

1. Open the "HMI alarms" editor and click on the "Alarm classes" tab.
2. Set the colors for alarms under "Properties > Properties > Colors".
3. Depending on the HMI device, also change the flashing characteristics if required.
4. To show alarms in the alarm view in the defined colors, activate the option "Alarm class colors" under "Runtime settings > Alarms > General".

Define display area

You select a tag which defines the time from which alarms should be displayed. The following data types are permitted:

- External tags: Date, Date_and_Time, Time_of_Day
- Internal tags: DateTime

To define the display area, proceed as follows:

1. Click "Properties > Properties > Display" in the Inspector window.
2. Define the tag in which the time is specified under "Control tag for display area".

Switch to PLC code view

You configure the "ActivatePLCCodeView" system function on a button to enable the jump from a monitoring alarm in the display to the affected program code in the PLC code view. In the alarm view, you define a control tag that you use to give the button dynamic properties. Control tag is used to evaluate whether the entry point of the selected monitoring signal is possible in the PLC code view. To define the control tag, proceed as follows:

1. Click "Properties > Properties > Display" in the Inspector window.
2. Specify a Boolean tag under "Properties > Display > Control tag for PLC code view" that will control the jump from the last active ProDiag alarm or GRAPH alarm to the PLC code view.

You have the option make the button dynamic using animations, for example, so that the button changes color with an incoming alarm.

Displaying column headers

The layout of the alarm view is dependent on the view settings in the control panel. Depending on the setting, the column headers may be truncated. This setting is found under "Display > Layout tab" in the control panel. To display column headers correctly, set the display in "Windows and buttons" to "Windows Classic" style.

This behavior only occurs during configuration. The column headers are displayed correctly in Runtime.

Alarm window (Basic Panels, Panels, Comfort Panels, RT Advanced)

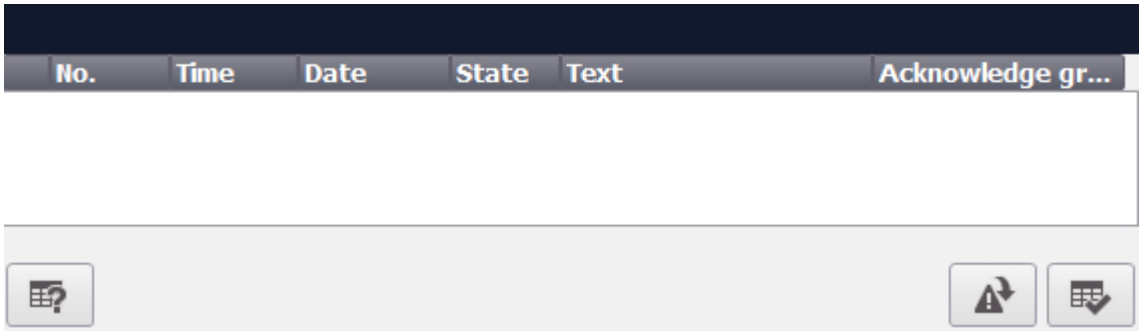
Use

Alarms are indicated in the Alarm view or in the Alarm window of the HMI device. The layout and operation of the Alarm window are similar to that of the Alarm view. The Alarm window has the following characteristics that are the same as in the Alarm view:

- Alarm window
- Alarm line: The alarm line is not available on Basic Panels.

The Alarm window is configured in the "Global screen" editor.

The alarm window is not assigned to any screen. Depending on the configuration the alarm window is opened when an alarm that belongs to a specific alarm class is active. Based on the configuration, it is not closed until the alarm is acknowledged.

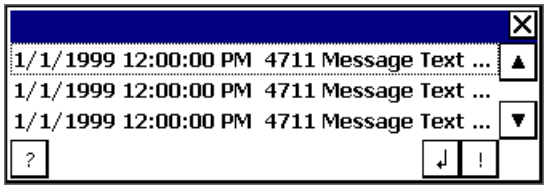


Note

In the engineering system you dynamize, for example, the visibility of an object in "Properties > Animations" in the inspector window. In Runtime, the "Simple alarm window" object does not support animations. If you have configured an animation and, for example, wish to perform a consistency check of the project, then an error alarm is issued in the Output window.

Simple alarm window

The simple alarm window is available on Basic Panels with a device version prior to V13 to display the alarms.



Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object. You configure the Alarm window in the same way as the Alarm view. In addition you adapt the following properties:

- Fixed alarm window: Specifies that the Alarm window retains the focus after a screen change.
- Window: You define the operator input and response of the Alarm window in Runtime.

Note

If you have different alarm classes output, these will be initially sorted into alarm classes in Runtime, and then by the time at which the alarm occurred.

Operator controls

The operator controls that can be used to control the alarm view in Runtime are specified in the inspector window under "Properties >Display > Settings". The following table shows the operator controls in the Alarm window, and what they do:

Button		Function
"Tooltip"	?	Displays a tooltip for an alarm.
"Acknowledge"	!	Acknowledges an alarm.
"Loop-In-Alarm"	↵	Executes the configured function (for example: "ActivateScreen") with each keystroke. It is hereby possible to call a screen containing your information about the selected alarm. Also acknowledges an alarm requiring acknowledgment. [1)]

[1)] Only if a function is configured for the selected alarm for the "loop-in-alarm" event.

Access protection in Runtime

Configure access protection under "Properties > Properties > Security" in the inspector window of the alarm view. If a logged-on user has the required authorization, he can acknowledge and edit alarms using the operator controls in the alarm view. If the logged-in user does not have the required authorization, or if no user is logged in, clicking the "Acknowledge" or "Edit" buttons or double-clicking an alarm line opens the login dialog box.

Note

Basic Panels

Access protection is not available for Basic Panels.

Activating the focus of the Alarm window

Select the following option so that the Alarm window does not lose the focus after a screen change:

1. In the Inspector window, select "Properties > Properties > Mode".
2. Enable "Label".

Window

Define the response of the Alarm window under "Properties > Properties > Mode > Window" in the inspector window. The following table shows the possible properties:

Option	Function
Automatic display	The Alarm window is automatically displayed when a system event occurs, for example.
Closable	The window closes again after a set time has elapsed. You define the display duration in the alarm settings.
Modal	The alarm window is linked to a confirmation, e.g. the alarm must be acknowledged. If the modal alarm window has the focus, the buttons in the screen behind it cannot be used. The functions configured on a function key are carried out.
Sizeable	You can change the size of the Alarm window in Runtime.

Alarm indicator (Basic Panels, Panels, Comfort Panels, RT Advanced)

Application

The alarm indicator is a graphic symbol that shows current errors or errors that need to be acknowledged, depending on the configuration. The alarm indicator is configured in the "Global screen" editor. The following figure shows an alarm indicator:



Alarm indicator OP 73

A "simple" alarm indicator is available for the HMI OP 73. The following diagram shows the alarm indicator for the OP 73 HMI devices:



The "simple" alarm indicator shows alarms to be acknowledged or alarms which have already been acknowledged and have not yet gone. Only the position can be defined for the "simple" alarm indicator. The alarm indicator is displayed on the device at the selected position. If you have configured a screen object at this position, the visible alarm indicator covers the screen

object. The alarm indicator is covered by system dialogs, such as the login dialog, Help dialog, and alarm windows.

Alarm classes

You define which alarm classes are shown with an alarm indicator in "General > Alarm classes" in the Inspector window. Alarm classes, such as "Warnings" or "Errors".

Define operator control in Runtime

1. Select the alarm indicator in the screen.
2. Click "Events > Click" or "Click when flashing" in the Inspector window.
3. The "function list" opens. Click the first line of the function list. The list of system functions, and scripts available in the project opens.
4. Select the "ShowAlarmWindow". system function under "Alarms."
5. Under "object name" select the name of the Alarm window from the selection list. Under "Layout", define whether the Alarm window should be visible, hidden, or should toggle between the two states.

Rectangle

Application

The "Rectangle" is a closed object which you can fill with a color.



Layout

In the Inspector window, you customize the position, shape, style, color and font types of the object. You can adapt the following properties in particular:

- Corner radius: Specifies the horizontal and vertical distance between the corner of the rectangle and the start point of a rounded corner.

Corner radius

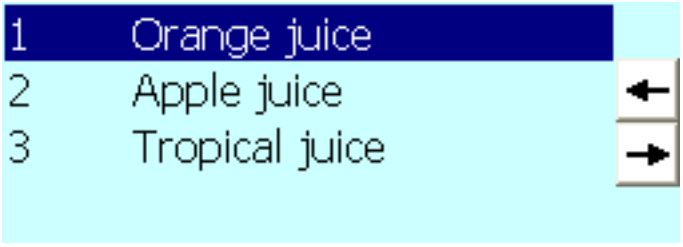
The corners of the "Rectangle" object can be rounded to suit your requirements. When the properties "X" and "Y" are set to the 100 % value, the rectangle is displayed as an ellipse. As soon as one of the properties has the value 0%, a normal rectangle without a rounded corner is shown.

1. Click "Properties > Properties > Layout" in the inspector window.
2. Enter a value for "X" in the "Corner radius" area.
   The input value is the percentage proportion of half the width of the rectangle.
3. Enter a value for "Y" in the "Corner radius" area.
   The input value is the percentage proportion of half the height of the rectangle.

Recipe view (Basic Panels, Panels, Comfort Panels, RT Advanced)

Use

The "Recipe view" object is used to display and modify recipes.



Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object. You can adapt the following properties in particular:

- Toolbar: Specifies the menu commands of the recipe view.

Toolbar

The menu items with which the recipe view is operated in Runtime are configured under "Properties > Properties > Toolbar" in the Inspector window.

Menu command	Description
"Tooltip"	Calls up the configured tooltip for the selected recipe.
"Add data record"	Creates a new recipe data record in the recipe.
"Delete data record"	Deletes the selected data record.
"Save"	Saves the modified record with its current name.
"Save as"	Saves the modified record with a new name.
"Write to PLC"	Sends the current value to the PLC.
"Read from PLC"	Reads the current value from the PLC.
"Rename"	Specifies that the "Rename" button is shown.

Menu command	Description
"Forward"	Specifies that the menu buttons are visible.
"Back"	Specifies that the "Back" button is shown.

Recipe view (as of V13) (Basic Panels, Panels, Comfort Panels, RT Advanced)

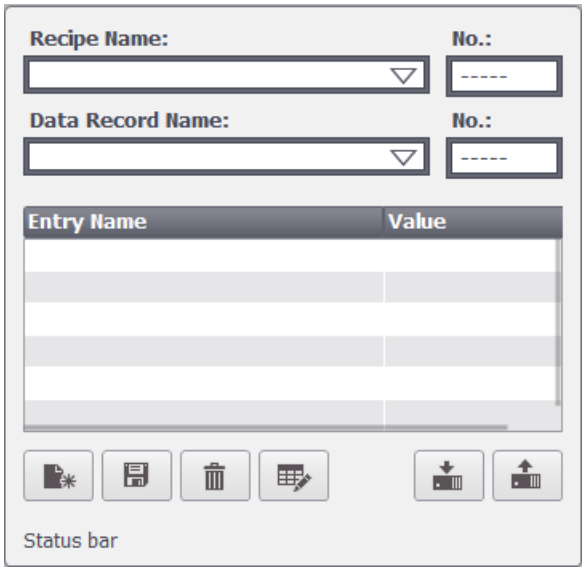
Use

The "Recipe view" object is used to display recipes on the HMI device.

Note

Device dependency of the "Advanced recipe view" object

The "Advanced recipe view" object is available on the HMI devices Basic Panels 2nd Generation, Comfort Panels, Mobile Panels with the device version V13.



Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object. You can adapt the following properties in particular:

- Toolbar: Defines the operator controls of the recipe view.
- Display number: Defines whether the number of the recipe and the number of the recipe data record is displayed.
- Label: Sets the label for the name of the recipe and the recipe data record.

Operator controls

The operator controls with which the recipe view is operated in Runtime are configured under "Properties > Properties > Buttons" in the Inspector window. In the simple recipe view the operating elements are based on the functions of the menu.

Operator control		Description
	"Tooltip"	Calls up the configured tooltip for the selected recipe.
	"Add data record"	Creates a new recipe data record in the recipe.
	"Delete data record"	Deletes the selected data record.
	"Rename"	Changes the name of selected data record.
	"Save"	Saves the modified record with its current name.
	"Save as..."	Saves the modified record with a new name.
	"Write to PLC"	Sends the current value to the PLC.
	"Read from PLC"	Reads the current value from the PLC.
	"Synchronize recipe tags"	Compares the values of the selected data record with the values on the PLC.

Display number

Specifies whether the number of the recipe and the number of the recipe data record are displayed in Runtime. The number of the recipe uniquely identifies the recipe in the project.

1. Click "Properties > Properties > View" in the Inspector window.
2. Select "Display > Display numbers".

Display label

Use this property to specify the name to be shown for recipes and data records in the recipe view.

1. Click "Properties > Properties > Label" in the Inspector window.
2. Select "Display labels".
3. Enter the labels for the recipe and the recipe data record.

Behavior during configuration

Displaying column headers

The layout of the recipe view is dependent on the view settings in the control panel. Depending on the setting, the column headers may be truncated. This setting is found under "Display > Appearance" in the control panel. To display column headers correctly, set the display in "Windows and buttons" to "Windows Classic" style.

This behavior only occurs during configuration. The column headers are displayed correctly in Runtime.

Scrolling in the recipe view

A recipe view contains an element that is shown as a text list in the recipe table. You can only scroll between the elements in the table of the recipe view, if you have previously selected an element that does not contain a text list.

See also

Advanced recipe view (as of V13) (Page 5578)

Switch (Basic Panels, Panels, Comfort Panels, RT Advanced)

Application

The "Switch" object is used to configure a switch that is used to switch between two predefined states in runtime. The current state of the "Switch" object can be visualized with either a label or a graphic.

The following figure shows a "Switch" type switch.



Layout

In the Inspector window, you customize the position, shape, style, color and font types of the object. In particular, you can customize the following property:

- Type: Defines the graphic representation of the object.

Type

Button visualization is specified at "Properties > Properties > General >Type" in the Inspector window.

Type	Description
"Switch"	The two states of the "Switch" are displayed in the form of a switch. The position of the switch indicates the current state. The state is changed in runtime by sliding the switch. You specify the direction of movement of the switch in "Switch orientation" with this type.
"Switch with text"	The switch is shown as a button. The current state is visualized with a label. In runtime click on the button to actuate the switch.
"Switch with graphic"	The switch is shown as a button. The current state is visualized with a graphic. In runtime click on the button to actuate the switch.

Note

Basic Panels

The "Switch" type is not available for Basic Panels.

Button (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Application

The "Button" object allows you to configure an object that the operator can use in Runtime to execute any configurable function.



Layout

In the Inspector window, you customize the position, geometry, style, color and font types of the object. You can adapt the following properties in particular:

- Mode: Defines the graphic representation of the object.
- Text / Graphic: Defines whether the Graphic view is static or dynamic.
- Define hotkey: Defines a key, or shortcut that the operator can use to actuate the button.

  Note

  You can only define a hotkey for HMI devices with keys.

Mode

The button display is defined in "Properties > Properties > General >Mode" in the Inspector window.

Mode	Description
"Invisible"	The button is not visible in Runtime.
"Text"	The button is displayed with text. This text explains the function of the button.
"Graphic"	The button is displayed with a graphic. This graphics represents the function of the button.
"Graphics or text"	The button is displayed with text or graphics. If the graphics cannot be displayed, the corresponding text is displayed.
"Graphics and text"	The button is displayed with text and graphics.

Different options are available depending on the device.

Text / Graphic

The "Mode" property settings are used to define whether the display is static or dynamic. The display is defined in "Properties > Properties > General >Text" or "Graphic" in the Inspector window.

You can, for example, select the following options for the "Graphic" or "Text" type.

Type	Option	Description
"Graphic"	"Graphic"	Use "Graphic when button is "not pressed"" to specify a graphic displayed in the button for the "OFF" state. Use "Graphic when button is "pressed"" to specify a graphic for the "ON" state.
	"Graphics list"	The graphic in the button depends on the state. The corresponding entry from the graphics list is displayed depending on the state.
"Text"	"Text"	Use "Text when button is "not pressed"" to specify the text displayed in the button for the "OFF" state. Use "Text when button is "pressed"" to specify text for the "ON" state.
	"Text list"	The text in the button depends on the state. The entry from the text list corresponding to the state is displayed.

Note

You cannot use text lists and graphics lists in buttons in WinCC Runtime Professional.

Define hotkey

In the Inspector window, a key or key combination is defined that the operator can use to control the button in Runtime.

1. In the Inspector window, select "Properties > Properties > General".
2. Select a key or key combination from the selection list in the "Hotkey" area.

Symbolic I/O field (Basic Panels, Panels, Comfort Panels, RT Advanced)

Application

The "Symbolic I/O field" object can be used to configure a selection list for input and output of texts in Runtime.



Layout

In the Inspector window, you customize the position, shape, style, color and font types of the object. You can adapt the following properties in particular:

- Mode: Specifies the response of the object in Runtime.
- Text list: Specifies the text list that is linked to the object.
- Button for selection list: Specifies that the object has a button to open the selection list.

Note

Reports

In reports, symbolic I/O fields only output data. "Output" mode is preset. Properties for configuring the selection of graphics are not available, e.g. "button for selection list".

Mode

The response of the symbolic I/O field is specified in the Inspector window in "Properties > Properties > General > Type".

Mode	Description
"Output"	The symbolic I/O field is used to output values.
"Input"	The symbolic I/O field is used to input values.
"Input/output"	The symbolic I/O field is used for the input and output of values.
"Two states"	The symbolic I/O field is used only to output values and has a maximum of two states. The field switches between two predefined texts. This is used, for example, to visualize the two states of a valve: closed or open.

Note

The behavior possible for the symbolic I/O field depends on the Runtime.

Text list

In the Inspector window, you specify which text list is linked to the symbolic I/O field.

1. In the Inspector window, select "Properties > Properties > General".
2. Under "Contents" open the selection list for "Text list".
3. Select a text list.

Button for selection list

The "Button for selection list" property is used to display a button for opening the selection list.

1. Select "Properties > Properties > Layout" in the Inspector window.
2. Select "Response > Button for selection list".

Note

Basic Panels

The "Button for selection list" option is not available for Basic Panels.

Behavior during operation

The selection list of the symbolic I/O field displays an empty text line if a corresponding entry was not defined in the project. The active state is indicated on the HMI device by changing the color of contents of the symbolic I/O field.

System diagnostics view (Basic Panels)

Introduction

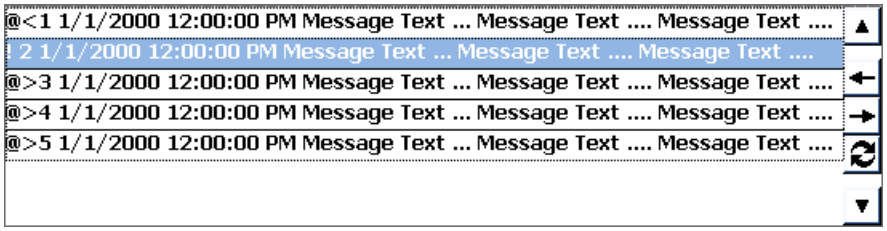
The system diagnostics view offers you an overview of all the available devices in your plant. You navigate directly to the cause of the error and to the relevant device. You have access to all diagnostics-capable devices which you have configured in the "Devices & networks" editor.

Use

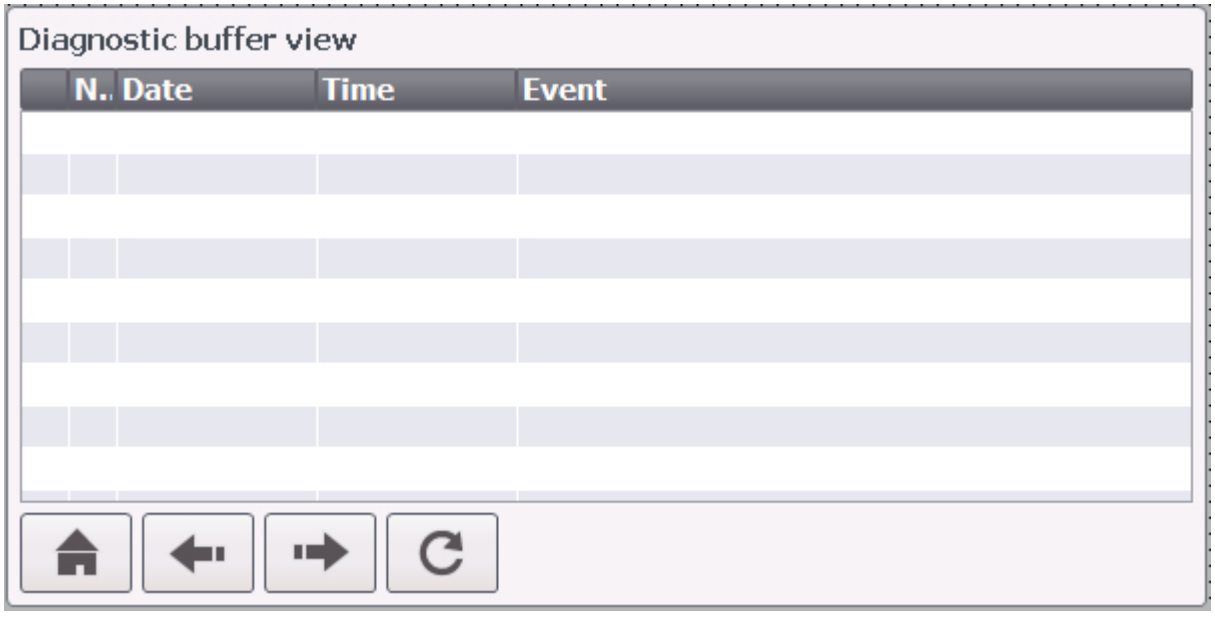
The system diagnostics view enables you to achieve the maximum level of detail of the diagnostics data. A precise diagnosis is possible, as all the available data is displayed. You have the system status of the entire plant at one glance.

Views in the system diagnostics view

A simple system diagnostics view is available on Basic Panels with a device version prior to V13.



The advanced system diagnostics view is available on Basic Panels with a device version of V13 or later.



Three different views are available in the system diagnostics view.

- Device view
- Diagnostic buffer view
- Detail view

Device view

The device view of the system diagnostics view shows all the available connections in tabular form. Double-clicking a connection opens the detail view. The device view is only displayed if more than one connection has been created in the "Devices & Networks" editor.

Diagnostic buffer view

In the diagnostic buffer view, the current data from the diagnostic buffer is displayed.

Detail view

The detail view gives detailed information about the selected connection. You cannot sort error texts in the detail view. The detail view is only available if there is an integrated connection to an S7-1200 or S7-1500.

Layout

You change the settings for the position, geometry, style, color, and font of the object in the Inspector window. You can adapt the following properties in particular:

- Lines per entry: Specifies the number of lines that are shown for an entry.

Configuring the system diagnostics view

1. Drag-and-drop the system diagnostics view from the toolbox.
2. In the Inspector window, select "Properties > Layout".
3. Enter a number under "Lines per entry", i.e. 5.
4. Select an authorization for operation in "Properties > Properties > Security".

See also

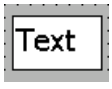
System diagnostics basics (Page 5788)

Configuring the system diagnostic view (Page 5791)

Text field (Panels, Comfort Panels, RT Advanced, RT Professional)

Application

The "Text field" is a closed object which you can fill with a color.



Layout

In the Inspector window, you customize the position, shape, style, color and font types of the object. You can adapt the following properties in particular:

- Text: Specifies the text for the text field.
- Size of text field: Defines whether the size of the object is adapted to the space required by the largest list entry.

Text

Specify the text for the text field in the Inspector window.

1. In the Inspector window, select "Properties > Properties > General".
2. Enter a text.
   For texts over several lines you can set a line break by pressing the key combination <Shift + Enter>.

Size of text field

In the Inspector window, you can define whether the size of the object is adapted to the space required by the largest list entry.

1. In the Inspector window, select "Properties > Properties > Layout".
2. Activate "Resize > Fit to contents".

12.2 Working with Tags (Basic Panels)

12.2.1 Basics (Basic Panels)

12.2.1.1 Basics of tags (Basic Panels)

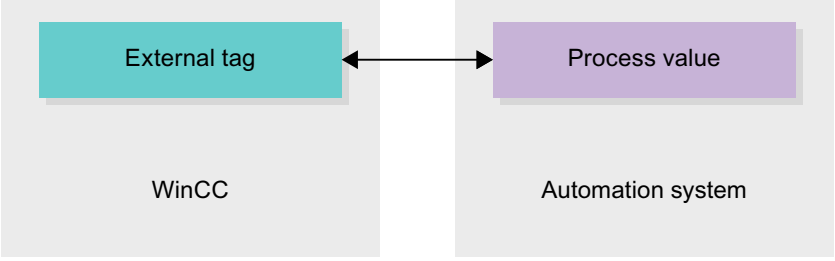
Introduction

Process values are forwarded in runtime using tags. Process values are data which is stored in the memory of one of the connected automation systems. They represent the status of a plant in the form of temperatures, fill levels or switching states, for example. Define external tags for processing the process values in WinCC.

WinCC works with two types of tag:

- External tags
- Internal tags

The external tags form the link between WinCC and the automation systems. The values of external tags correspond to the process values from the memory of an automation system. The value of an external tag is determined by reading the process value from the memory of the automation system. It is also possible to rewrite a process value in the memory of the automation system.



Internal tags do not have a process link and only convey values within the WinCC. Tag values are only available while Runtime is running.

Tags in WinCC

For external tags, the properties of the tag are used to define the connection that the WinCC uses to communicate with the automation system and form of data exchange.

Tags that are not supplied with values by the process - the internal tags - are not connected to the automation system. In the tag's "Connection" property, this is identified by the "Internal tag" entry.

You can create tags in different tag tables for greater clarity. You then directly access the individual tag tables in the "HMI tags" node in the project tree. The tags from all tag tables can be displayed with the help of the table "Show all tags".

See also

Overview of HMI tag tables (Page 5422)

Internal tags (Page 5428)

External tags (Page 5423)

Addressing external tags (Page 5425)

Creating external tags (Page 5430)

Basics on arrays (Page 5463)

Cycle basics (Page 5468)

12.2.1.2 Overview of HMI tag tables (Basic Panels)

Introduction

HMI tag tables contain the definitions of the HMI tags that apply across all devices. A tag table is created automatically for each HMI device created in the project.

In the project tree there is an "HMI tags" folder for each HMI device. The following tables can be contained in this folder:

- Standard tag table
- User-defined tag tables
- All tags

The following tables are also available in an HMI tag table:

- Discrete alarms
- Analog alarms

With the help of these tables you configure alarms for the currently selected HMI tag.

In the project tree you can create additional tag tables in the HMI tags folder and use these to sort and group tags and constants. You can move tags to a different tag table using a drag-and-drop operation or with the help of the "Tag table" field. You activate the "Tag table" field using the shortcut menu of the column headings.

Standard tag table

There is one standard tag table for each HMI device of the project. It cannot be deleted, renamed or moved. The standard tag table contains HMI tags and, depending on the HMI device, also system tags. You can declare all HMI tags in the standard tag table, or create additional user-defined tag tables as you want.

User-defined tag tables

You can create multiple user-defined tag tables for each HMI device in order to group tags according to your requirements. You can rename, gather into groups, or delete user-defined tag tables. To group tag tables, create additional subfolders in the HMI tags folder.

All tags

The "All tags" table shows an overview of all HMI tags and system tags of the HMI device in question. This table cannot be deleted, renamed or moved.

Discrete alarms table

In the "Discrete alarms" table, you configure discrete alarms to the HMI tag selected in the HMI tag table. When you configure a discrete alarm, multiple selection in the HMI tag table is not possible. You configure the discrete alarms for each HMI tag separately.

Analog alarms table

In the "Analog alarms" table, you configure analog alarms to the HMI tag selected in the HMI tag table. When you configure an analog alarm, multiple selection in the HMI tag table is not possible. You configure the analog alarms for each HMI tag separately.

See also

Basics of tags (Page 5421)

12.2.1.3 External tags (Basic Panels)

Introduction

External tags allow communication (data exchange) between the components of an automation system, for example, between an HMI device and a PLC.

Principle

An external tag is the image of a defined memory location in the PLC. You have read and write access to this storage location from both the HMI device and from the PLC.

Since external tags are the image of a storage location in the PLC, the applicable data types depend on the PLC which is connected to the HMI device.

If you write a PLC control program in STEP 7, the PLC tags created in the control program will be added to the PLC tag table. If you want to connect an external tag to a PLC tag, access the PLC tags directly via the PLC tag table and connect them to the external tag.

Data types

All the data types which are available at the connected PLC are available at an external tag in WinCC. Information about data types which are available for connection to other PLCs can be found in the documentation about the respective communication drivers.

See "AUTOHOTSPOT" for additional information.

Note

As well as external tags, area pointers are also available for communication between the HMI device and PLC. You can set up and enable the area indicators in the "Connections" editor.

Central tag management in STEP 7

You can connect also connect DB instances of user-defined PLC data types (UDT) to the HMI tags.

The PLC data type and the corresponding DB instances are created and updated centrally in STEP 7. In WinCC, you can use the following sources as PLC tag (DB instances):

- Data block elements that use a UDT as data type
- Data block instances of a UDT

The data type is taken from STEP 7 and is not converted to an HMI data type. The access type is always "Symbolic access". Depending on the release for WinCC in STEP 7, elements and structured elements of the PLC data type are applied to WinCC. Elements of a structured UDT are applied and displayed in the PLC tag table if the instance-specific properties "Visible in HMI" and "Accessible from HMI" have been set.

Note

Accessing PLC data types

Access to PLC data types is only available in conjunction with SIMATIC S7-1200 or S7-1500.

Synchronization with PLC tags

A variety of options for synchronizing external tags with the PLC tags are available in the Runtime settings under "Settings for tags".

When you perform synchronization, you have the option of automatically applying the tag names of the PLC to external tags and reconnecting the existing tags.

The generated tag name is derived from the position of the data value in the hierarchical structure of the data block.

Update of tag values

For external tags, the current tag values are transmitted in Runtime via the communication connection between WinCC and the connected automation systems and then saved in the runtime memory. Next, the tag value will be updated to the set cycle time. For use in the runtime project, WinCC accesses tag values in the runtime memory that were read from the PLC at the previous cycle time. As a result, the value in the PLC can already change while the value from the runtime memory is being processed.

Note

PLC array elements in conjunction with S7-1200 or S7-1500

The index of the PLC array elements can begin with any number. In WinCC, indexing always starts with 0.

A PLC tag "Array [1..3] of Int", for example, is mapped to "Array [0..2] of Int" in WinCC.

When you access an array in a script, pay attention to the correct indexing.

See also

Addressing external tags (Page 5425)

Basics of tags (Page 5421)

12.2.1.4 Addressing external tags (Basic Panels)

Introduction

The options for addressing external tags depend on the type of connection between WinCC and the PLC in question. A distinction must be made between the following connection types:

- Integrated connection
  Connections of devices which are within a project and were created with the "Devices & Networks" editor are referred to as integrated connections.
- Non-integrated connection
  Connections of devices which were created with the "Connections" editor are referred to as non-integrated connections. It is not necessary that all of the devices be within a single project.

The connection type can also be recognized by its icon.

	Integrated connection
	Non-integrated connection

You can find additional information in the section "AUTOHOTSPOT".

Addressing with integrated connections

An integrated connection offers the advantage that you can address a tag both symbolically and absolutely.

For symbolic addressing, you select the PLC tag via its name and connect it to the HMI tag. The valid data type for the HMI tag is automatically selected by the system. You have to distinguish between the following cases when you address elements in data blocks:

Symbolic addressing of data blocks with optimized access and standard access:

During the symbolic addressing of a data block with optimized access and standard access, the address of an element in the data block is dynamically assigned and is automatically adopted in the HMI tag in the event of a change. You do not need to compile the connected data block or the WinCC project for this step.
For data blocks with optimized access, only symbolic addressing is available.

For symbolic addressing of elements in a data block, you only need to recompile and reload the WinCC project in case of the following changes:

- If the name or the data type of the linked data block element or global PLC tag has changed.
- If the name or the data type of the higher level structure node of a linked element in the data block element or global PLC tag has changed.
- If the name of the connected data block has changed.

Symbolic addressing is currently available with the following PLCs:

- SIMATIC S7-1200
- SIMATIC S7-1500
- SIMATIC ET 200 CPU
- SIMATIC S7-1500 software controller

Symbolic addressing is also available if you have an integrated link.

You can also use absolute addressing with an integrated connection. You have to use absolute addressing for PLC tags from a SIMATIC S7-300/400 PLC. If you have connected an HMI tag with a PLC tag and the address of the PLC tag changes, you only have to recompile the control program to update the new address in WinCC. Then you recompile the WinCC project and load it onto the HMI device.

In WinCC, symbolic addressing is the default method. To change the default setting, select the menu command "Options > Settings". Select "Visualization > Tags" in the "Settings" dialog. If required, disable the "Symbolic access" option.

The availability of an integrated connection depends on the PLC used. The following table shows the availability:

PLC	Integrated connection	Comments
S7-300/400	Yes	The linking of tags is not checked in Runtime. If the tag address changes in the PLC and the HMI device is not compiled again and loaded, the change is not registered in runtime.
S7-1200	Yes	A validity check of the tag connection is performed in runtime during symbolic addressing. If an address is changed in the PLC, the change is registered and an error message is issued. In the case of absolute addressing, the behavior described for the S7-300/400 applies.
S7-1500	Yes	A validity check of the tag connection is performed in runtime during symbolic addressing. If an address is changed in the PLC, the change is registered and an error message is issued. In the case of absolute addressing, the behavior described for the S7-300/400 applies.
SIMATIC ET 200 CPU	Yes	A validity check of the tag connection is performed in runtime during symbolic addressing. If an address is changed in the PLC, the change is registered and an error message is issued. In the case of absolute addressing, the behavior described for the S7-300/400 applies.
SIMATIC S7-1500 software controller	Yes	A validity check of the tag connection is performed in runtime during symbolic addressing. If an address is changed in the PLC, the change is registered and an error message is issued. In the case of absolute addressing, the behavior described for the S7-300/400 applies.

Create an integrated connection in the "Devices & Networks" editor. If the PLC is contained in the project and integrated connections are supported, you can then also have the connection created automatically. To do this, when configuring the HMI tag, simply select an existing PLC tag to which you want to connect the HMI tag. The integrated connection is then automatically created by the system.

Addressing with non-integrated connections

In the case of a project with a non-integrated connection, you always configure a tag connection with absolute addressing. Select the valid data type yourself. If the address of a PLC tag changes in a project with a non-integrated connection during the course of the project, you also have to make the change in WinCC. The tag connection cannot be checked for validity in Runtime, an error message is not issued.

A non-integrated connection is available for all supported PLCs.

Symbolic addressing is not available in a non-integrated connection.

With a non-integrated connection, the control program does not need to be part of the WinCC project. You can perform the configuration of the PLC and the WinCC project independently

of each other. For configuration in WinCC, only the addresses used in the PLC and their function have to be known.

See also

External tags (Page 5423)

Basics of tags (Page 5421)

12.2.1.5 Internal tags (Basic Panels)

Introduction

Internal tags do not have any connection to the PLC. Internal tags transport values within the HMI device. Tag values are only available while Runtime is running.

Principle

Internal tags are stored in the memory of the HMI device. Therefore, only this HMI device has read and write access to the internal tags. You can create internal tags to perform local calculations, for example.

You can use the HMI data types for internal tags.

The following HMI data types are available:

HMI data type	Data format
Array	One-dimensional array
Bool	Binary tag
DateTime	Date/time format
DInt	Signed 32-bit value
Int	Signed 16-bit value
LReal	Floating-point number 64-bit IEEE 754
Real	Floating-point number 32-bit IEEE 754
SInt	Signed 8-bit value
UDInt	Unsigned 32-bit value
UInt	Unsigned 16-bit value
USInt	Unsigned 8-bit value
WString	Text tag, 16-bit character set

See also

Basics of tags (Page 5421)

12.2.1.6 User-defined PLC data types (UDT) (Basic Panels)

Overview

You can connect with the HMI tags and DB instances of user-defined PLC data types (UDT).

The PLC data type and the corresponding DB instances are created and updated centrally in STEP 7. In WinCC, you can use the following sources as PLC tag (DB instances):

- Data block elements that use a UDT as data type
- Instance data blocks of a UDT

The data type is taken from STEP 7 and is not converted into an HMI data type. The access type is always "Symbolic access".

Elements of a PLC data type

You have access to the following elements in WinCC with a structured PLC data type:

- Elements that have been released for WinCC in STEP 7.
- Elements whose data types are supported in WinCC.

Note

Invalid elements of a PLC data type in WinCC

Invalid elements generate an error in WinCC.

If you disable the "Accessible from HMI" option for the corresponding elements of the associated PLC data type in STEP 7, these elements are excluded in WinCC.

Naming conventions

The following characters are invalid in the name of the PLC data type and generate an error in WinCC:

- Period: "."
- Brackets: "(" and ")"

Properties

The properties of the PLC data type and its elements are adopted in WinCC. Depending on the data type used, the properties are read-only or can be written to in WinCC.

If you change the connection of the PLC data type in WinCC, all elements of the PLC data type are deleted and the properties of the newly connected PLC tag are used.

In WinCC, you have access to STEP 7 comments on elements of the PLC data type.

You have limited access to properties in WinCC for the following elements of PLC data types:

- Elements of the data type "Struct"
- PLC data type

- Multidimensional arrays
- Array of complex data types except "DTL"

Mapping of the data type "DTL"

If a PLC data type contains elements of the data type "DTL", these elements are mapped in WinCC without lower-level elements. The data type "DTL" turns into "DateTime" in WinCC.

Tags with elements of the "DTL" data type

Tags that use the "DTL" data type element by element can only be used as read-only with symbolic addressing, e.g. with SIMATIC S7 1200 and SIMATIC S7 1500. With absolute addressing, write access is also possible.

12.2.2 Working with tags (Basic Panels)

12.2.2.1 Creating tags (Basic Panels)

Creating external tags (Basic Panels)

Introduction

You can access an address in the PLC via a PLC tag using an external tag. The following options are available for addressing:

- Symbolic addressing
- Absolute Addressing

You can find further details on symbolic addressing in section "Addressing external tags (Page 5425)". If possible, use symbolic addressing when configuring a tag. You create tags either in the standard tag table or in a tag table you defined yourself.

Requirement

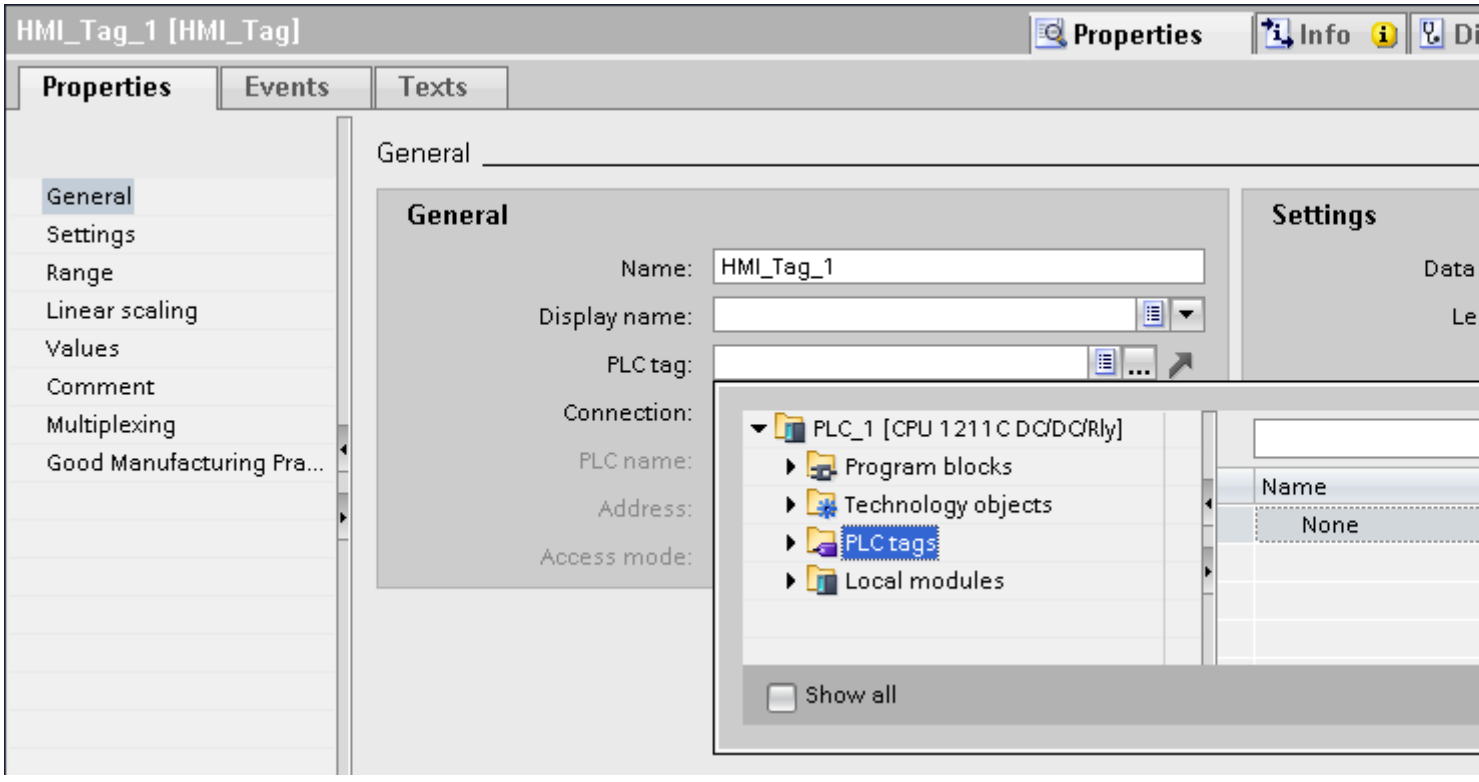
- You have opened the project.
- A connection to the PLC is configured.
- The Inspector window is open.

Procedure

To create an external tag, proceed as follows:

1. Open the "HMI tags" folder in the project tree and double-click the standard tag table. The tag table open.
   Alternatively, create a new tag table and then open it.
2. In the "Name" column, double-click "Add" in the tag table. A new tag is created.
3. Select the "Properties > Properties >General" category in the Inspector window and, if required, enter a unique tag name in the "Name" field. The tag name must be unique throughout the device.
4. If required, select the "Display name" field to enter a name to be displayed in Runtime. The name to be displayed is language-specific and can be translated for the required Runtime languages. The display name is available for Basic Panels, Panels and Runtime Advanced.
5. Select the connection to the required PLC in the "Connection" box. If the connection you require is not displayed, you must first create the connection to the PLC. You create the connection to a SIMATIC S7 PLC in the "Devices & Networks" editor. You create the connection to external PLCs in the "Connections" editor.
   If the project contains the PLC and supports integrated connections, you can also have the connection created automatically. To do this, when configuring the HMI tag, simply select an existing PLC tag to which you want to connect the HMI tag. The integrated connection is then automatically created by the system.

6. If you are working with an integrated connection, click the ... button in the "PLC tag" field and select an already created PLC tag in the object list. Click the ✓ button to confirm the selection.



Alternatively, use the autocomplete to select a PLC tag for an integrated connection. If you select a PLC tag from the autocomplete list, it is entered in the input path. The elements of the PLC tags are displayed in the autocomplete list. If you have selected a PLC tag that can be connected to the HMI tags, the PLC tag is connected to the HMI tags.

7. If you are working with a non-integrated connection, enter the address from the PLC in the "Address" field. The "PLC tag" field remains empty.
8. Configure the other properties of the tag in the inspector window.

You can also configure all tag properties directly in the tag table. To view hidden columns, activate the column titles using the shortcut menu.

You can also create tags alternatively directly at the application point, e.g. on an I/O field. To do this, click the Add new button in the object list. You then configure the new tag in the Inspector window.

Result

An external tag has been created and linked to a PLC tag or an address in the PLC.

Alternative procedure

You can also create external HMI tags by dragging and dropping data block elements or global PLC tags in an HMI tag table.

See also

Creating internal tags (Page 5433)

Creating multiple tags (Page 5434)

Editing a Tag (Page 5435)

Tag limits (Page 5443)

Basics of tags (Page 5421)

Addressing external tags (Page 5425)

Creating internal tags (Basic Panels)

Introduction

You must at least set the name and data type for internal tags. Select the "Internal tag" item, rather than a connection to a PLC.

For documentation purposes, it is a good idea to enter a comment for every tag.

Requirement

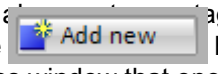
You have opened the project.

Procedure

1. Open the "HMI tags" folder in the project tree and double-click the entry "Standard tag table". The tag table opens.
   Alternatively, create and then open a new tag table.
2. Double-click "Add" in the "Name" column of the tag table. A new tag is created.
3. If the Inspector window is not open, select the "Inspector window" option in the "View" menu.
4. Select the "Properties > Properties >General" category in the Inspector window and, if required, enter a unique tag name in the "Name" field. This tag name must be unique throughout the project.
5. If required, select the "Display name" field to enter a name to be displayed in Runtime. The name to be displayed is language-specific and can be translated for the required Runtime languages. The display name is available for Basic Panels, Panels and Runtime Advanced.
6. Select "Internal tag" as the connection in the "Connection" field.
7. Select the required data type in the "Data type" field.

8. In the "Length" field, you must specify the maximum number of characters to be stored in the tag according to the selected data type. The length is automatically defined by the data type for numerical tags.
9. As an option, you can enter a comment regarding the use of the tag. To do so, click "Properties > Properties > Comment" in the Inspector window and enter a text.

You can also configure all tag properties directly in the tag table. To view hidden columns, activate the column titles using the shortcut menu.

You can a ags alternatively directly at the application point, e.g. on an I/O field. Click the 

button in the object list. You can then configure the new tag in the Properties window that opens.

Result

An internal tag is created. You can now use this in your project.

In additional steps you can configure the tag, for example, by setting the start value and limits.

See also

Creating external tags (Page 5430)

Creating multiple tags (Basic Panels)

Introduction

In a tag table, you create additional identical tags by automatically filling the rows of the table below a tag.

The tag names are incremented automatically when filling in automatically.

You can also use the auto fill function to fill table cells below a tag with a single tag property and thus modify the corresponding tags.

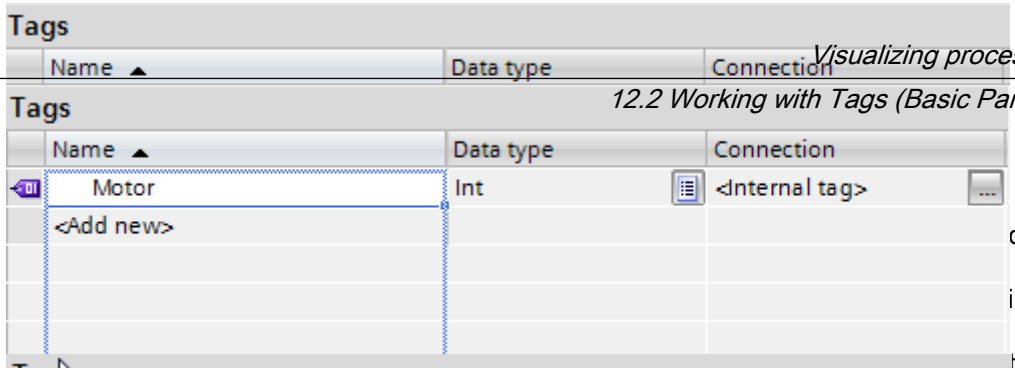
If you apply the automatic filling in to already filled cells of a tab table, you will be asked whether you want to overwrite the cells or insert new tags.

If you do not want to overwrite already configured tags, activate insert mode. Activate insert mode by keeping the <Ctrl> key pressed during insertion. Already existing entries in the tag table are moved down if insert mode is activated.

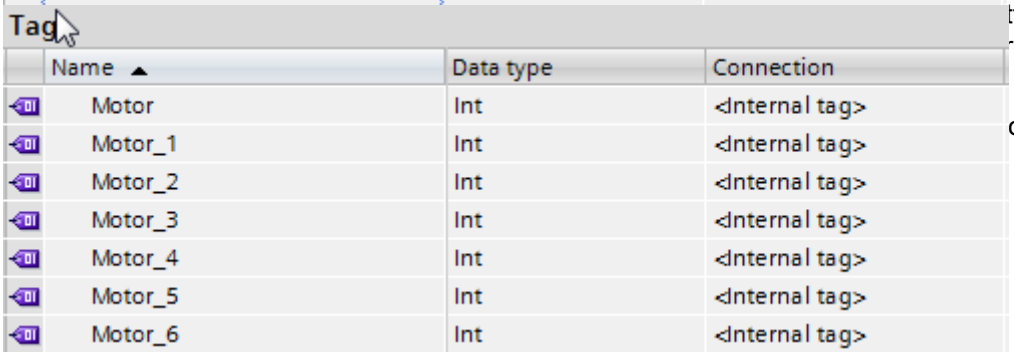
Requirement

- You have opened the project.
- A tag table is open.
- The tag which is to serve as a template for other tags is configured.

Procedure

1. 

d as
ins
ttom
ross.
2. o fill
3.



Result

Depending on which cells were selected, the function may automatically fill individual properties or create new tags.

See also

Creating external tags (Page 5430)

12.2.2.2 Editing tags (Basic Panels)

Editing a Tag (Basic Panels)

Introduction

You can always rename, copy or delete tags.

When a tag is renamed, the new name must be unique for the whole device.

If you use the "Copy" command to copy a tag to the clipboard, the objects and references linked to the tag are copied as well.

If you use the "Insert" command to add a tag to another device, the tag will be added without the connected references. Only the object name of the reference will be inserted. If a reference of the same name and valid properties exists in the target system, the existing reference will then be connected to the copied tag.

If you copy a tag, the alarms linked to the tag are copied as well. If you add the copied tag to another device, the tag is added together with the linked alarms.

Requirement

- The tag which you wish to rename, copy or delete must exist.
- The tag table is open.

Renaming tags

Note

During renaming, the connection to the tags may be interrupted in Runtime under the following conditions:

- HMI tag is used in a type, for example, make an object property dynamic via a script.
- One or more instances of the type are used in the project.
- Project is in Runtime.
- After renaming, select the command "Compile > Software (only changes)".

Remedy: Exit Runtime and rename the tag. In the shortcut menu, select the "Compile > Software (compile all)" command.

1. In the "Name" field, select the tag in the tag table.
2. Select "Rename" from the shortcut menu.
3. Type in a new name.
   The tag appears under its new name.

Copying tags

1. Select one or more tags in the tag table or in the Detail window.
2. Select "Copy" from the shortcut menu.
3. Click on the point at which you want to insert the tag. For example, click another tag table in the same device or the tag table in a second device.
4. Select the "Insert" or "Extended insert" command from the shortcut menu. The tag is inserted as described above.

Deleting a tag

1. Select one or more tags in the tag table.
2. Select the "Cross-reference" command from the "Tools" menu. In the "Cross-reference" editor, check to see where the tags are used. In this manner, you can see what impact the deletion of the tag will have on your project.
3. Select "Delete" in the pop-up menu of the tag.
   All marked tags will be deleted.

Export and import of tags

WinCC gives you the option to export and import tags. With Export and Import, you have the option to export tags from one project and import them into another project. There is also the option to create larger numbers of tags outside of WinCC, edit them and subsequently import into any WinCC project. See AUTOHOTSPOT for additional information.

See also

Edit tags (Page 5437)

Configuring multiple tags simultaneously (Page 5438)

Using multiple tags simultaneously in a screen (Page 5438)

Synchronizing tags (Page 5442)

Creating external tags (Page 5430)

Edit tags (Basic Panels)

Introduction

You can modify tags at any time to adapt them to changed requirements in the project.

Edit tags

You have the following options to change the configuration of a tag:

- You open the tag table in which the tag is contained.
- Open the tag table "Show all tags".
- You open the Inspector window of a tag with the "Edit" button in the object selection on the display and operating object.

In the tag tables, you can perform such tasks as comparing and adjusting the properties of multiple tags or sorting the tags by their properties.

You can change the properties either directly in the table or in the Inspector window.

If you change a tag property and this change causes a conflict with another property, it will be highlighted in color to draw your attention to this fact. If you link the tag with another controller, for example, which does not support the set data type, this property is highlighted in color.

See also

Editing a Tag (Page 5435)

Configuring multiple tags simultaneously (Basic Panels)

Introduction

In WinCC, you can assign the same properties to multiple tags in a single operation. This facilitates efficient programming.

Requirement

- You created the tags you want to configure.
- The tag table is open.
- The Inspector window is open.

Procedure

1. In the tag table, select all the tags that you want to configure at the same time.
   If the selected property is identical for all the tags, the setting for this property will appear in the Inspector window. The associated field will remain blank otherwise.
2. You can define the shared properties in the Inspector window or directly in the tag table.

if you change a property commonly on several tags, only this one property is changed. The other properties of the tag remain unchanged.

Result

All marked tags will be reconfigured.

To edit tag properties which differ from one tag to the other, simply clear the multiple selection.

See also

Editing a Tag (Page 5435)

Using multiple tags simultaneously in a screen (Basic Panels)

Introduction

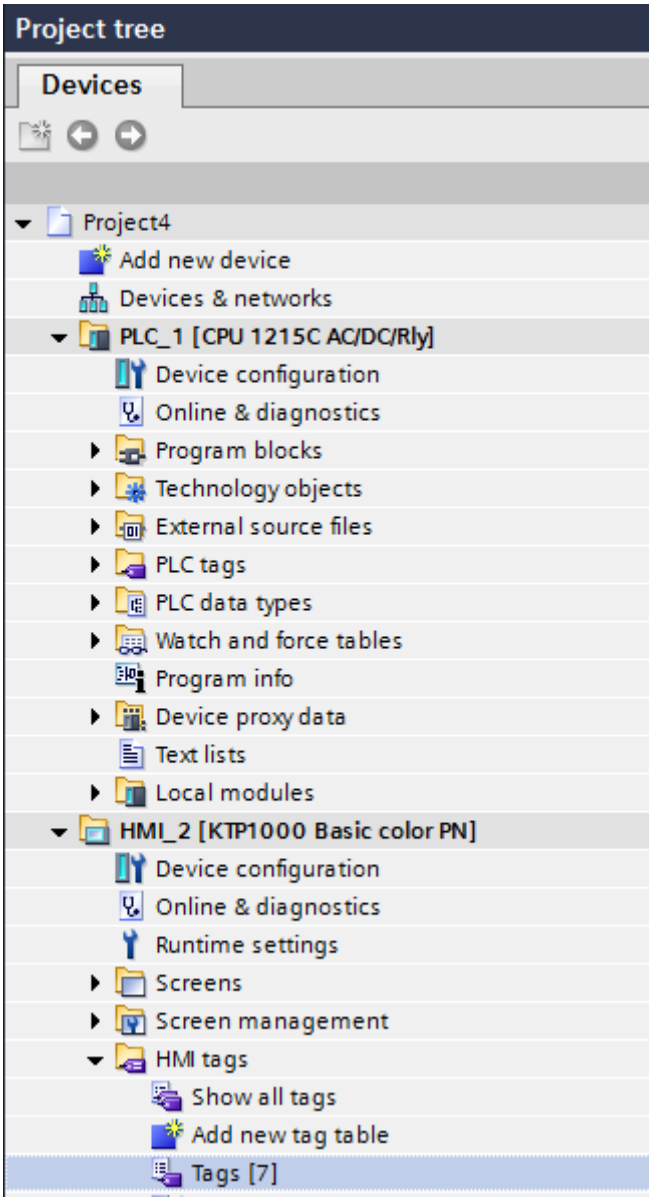
In WinCC, you can create multiple I/O fields that are linked with tags in one screen in a single operation. This facilitates efficient programming.

Requirement

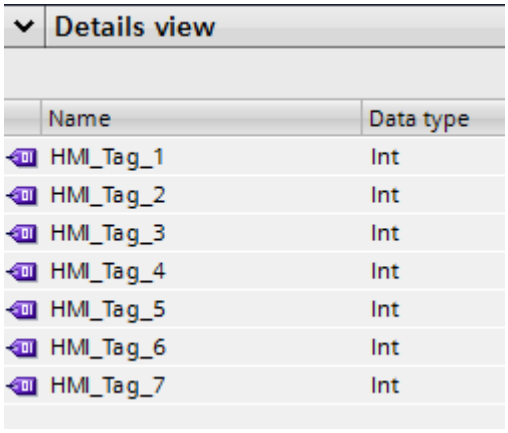
- Several tags are set up.
- A screen is open.

Procedure

1. In the project tree, select the required tag table under "HMI tags".



2. Select the detail view at the bottom of the project tree. The detail view shows the tags that exist in the selected tag group.



3. Mark the tags in the detail window.
4. Drag the tags to the screen. For each tag, this creates an I/O field that is connected to the tag.

Note

When you move a PLC tag from the detail window to the work area by drag&drop, a network and a connection are created additionally in the "Devices & Networks" editor.

See also

Editing a Tag (Page 5435)

Settings for synchronization (Basic Panels)

Introduction

In WinCC, you can map the location of a data value in the structure of a data block in the WinCC tag name. To do this, you synchronize the WinCC tags with the connected PLC tags. If necessary, the PLC name is set as a prefix.

You configure the name matching based on the requirements of your project set under "Runtime Settings > Settings for Tags". WinCC offers you several options for this.

You specify whether the names are matched and the conditions for this to occur during synchronization.

Settings for tags

To avoid conflicts within complex tag types that you configure under "Settings for tags", the delimiters of the path statement from STEP 7 are replaced in WinCC similar to name matching:

- Compatibility mode
  In compatibility mode, an underscore is set between the names of the data block and the element name of the first hierarchy level. The delimiters in the lower hierarchy levels are retained. When you enable this option, all other options are disabled.
- Replace delimiter
  Depending on your selection, the delimiters of all hierarchy levels are replaced during synchronization.
- PLC prefix
  The PLC name is set as a prefix to the WinCC tag name. You set the option for each HMI connection.

Note

Duplicate tag names

If the newly generated tag name is already in use, a number is added in parentheses, for example, Datablock_1_Static_2{1}(1).

Example

The "PLC1" controller contains the structured data block "DB1". The "Db1.a[1].b.c[3]" data block element is used in a picture. Depending on your settings, the WinCC tag name is generated as follows:

WinCC tag name	Selected option	Comment
Db1_a[1].b.c[3]	Compatibility mode	
Plc1.Db1.a[1].b.c[3]	PLC prefix	
Db1;a(1);b;c(3)	Dot and parenthesis replaced with ; ( )	
Plc1_Db1_a{10}_b_c{3}	Dot and parenthesis replaced with _ { } PLC prefix	
Db1.a[1].b.c[3]	Delimiter replaced without character selection	The tag name is enclosed in quotes at the point of use in the picture: "Db1.a[1].b.c[3]"

Synchronizing tags (Basic Panels)

Introduction

To synchronize the PLC and HMI tags, WinCC offers the following options:

- Synchronizing tags with or without name matching between PLC and WinCC
  The following options are available for this:
- Link tags with addresses in the PLC
  This procedure is suitable, for example, if changes were made to the connection between the HMI device and the PLC and the tag connections were lost. The function can also be used if you have configured the control program and HMI project separately.

Requirement

- External HMI tags have been created.
- PLC tags have been created.
- An HMI connection to a PLC in the project has been established.

Procedure

To synchronize HMI tags with PLC tags, follow these steps:

1. In the project tree, select the directory that contains the tags in question.
2. Select "Connect HMI tags to matching PLC tags" from the shortcut menu.
   A dialog opens.



3. Select the option you want to use.
   If you want to synchronize the tags without name matching, disable "Replace WinCC tag name with PLC tag name".
   If you want to connect HMI tags new with absolute access, select "Data type and absolute address match".
4. Confirm with "Synchronize".
   The system searches for a suitable PLC tag according to the selected option.

Result

The external HMI tags are synchronized with the PLC tags.

If you have selected the option "Data type and absolute address match", the tag connection is established as soon as a suitable PLC tag is found.

If you have selected a different option, the WinCC tags are updated accordingly and the PLC tag names are applied in WinCC.

See also

Editing a Tag (Page 5435)

12.2.2.3 Configuring tags (Basic Panels)

Tag limits (Basic Panels)

Introduction

You can restrict the value range with limits for numerical tags.

Principle

You can specify a value range defined by a high and low limit for numerical tags.

If the process value violates the value range, you trigger a function list. If the operator enters a value for the tag that is outside the configured value range, the input is rejected. The value is not accepted.

Note

You enter the text of the analog alarms for limit violations in the "Analog alarms" editor.

Application example

Use the limit to warn the operator in good time when the value of a tag enters a critical range, for example.

See also

Applying linear scaling to a tag (Page 5449)

Defining the start value of a tag (Page 5446)

Defining the acquisition cycle for a tag (Page 5453)

Defining Limits for a Tag (Page 5444)

Start value of a tag (Page 5445)

Updating the tag value in Runtime (Page 5446)
Linear scaling of a tag (Page 5447)
Connecting a tag to another PLC (Page 5449)
Indirect addressing of tags (Page 5450)
Addressing tags indirectly (Page 5451)
Using tags to trigger functions (Page 5452)
Address multiplexing (Page 5454)
Configuring address multiplexing with absolute addressing (Page 5457)
Configuring address multiplexing with symbolic addressing (Page 5460)
Creating external tags (Page 5430)

Defining Limits for a Tag (Basic Panels)

Introduction

For numerical tags, you can specify a value range by defining a low and high limit.

Additionally, you configure the system to process a function list whenever a tag value drops below or exceeds its configured value range.

Requirement

- You created the tag for which you want to set limits.
- The Inspector window with the properties for this tag is open.

Procedure

To define limits of a tag, follow these steps:

1. In the Inspector window select "Properties > Properties > Limits." If you want to define one of the limits as a constant value, select "Constant" using the button. Enter a number in the relevant field.
   If you want to define one of the limits as a tag value, select "HMI tag" using the button. Use the object list to define the tag for the limit.
2. To set an additional limit value for the tag, repeat step 1 with the appropriate settings.

Alternative procedure

You can also configure the high and low limit directly in the tag table. To view hidden columns, activate the column titles using the shortcut menu.

Configuring a function list

You can configure a function list for exceeding the value range as follows:

1. If you want to start a function list when the value drops below the value range, click "Properties > Events > Minimum violated" in the Inspector window. Create a function list in this dialog.
2. If you want to start a function list when the value exceeds the value range, click "Properties > Events > Maximum violated" in the Inspector window. Create a function list in this dialog.

Result

You have set a value range defined by a high and low limit for the selected tag. If the value range is exceeded or undershot, a function list is carried out.

See also

Tag limits (Page 5443)

Start value of a tag (Basic Panels)

Value of a tag at start of Runtime

You can configure a start value for numeric tags and tags for date/time values. The tag will be preset to this value when Runtime starts. In this way, you can ensure that the tag has a defined status when Runtime starts.

For external tags, the start value will be displayed on the HMI device until it is overwritten by the PLC or by input.

If no start value was configured, the tag contains the value "0" when starting Runtime.

In WinCC Runtime Professional you can enter a tag value in place of the start value on a tag with the "String" data type. The tag value is saved in the "Project texts" editor and is multilingual. After the text has been translated, it is displayed in Runtime as a language-dependent start value.

Application example

You can assign a default value to an I/O field. Enter the desired default value as start value for the tag that is linked to the I/O field.

See also

Defining the start value of a tag (Page 5446)

Tag limits (Page 5443)

Defining the start value of a tag (Basic Panels)

Introduction

In WinCC you can configure a start value for a numeric tag and a tag for date/time values which this adopts at Runtime start.

Requirement

- You have created the tag for which you want to define a start value.
- The Inspector window with the tag properties is open.

Procedure

To configure a start value, proceed as follows:

1. In the Inspector window select "Properties > Properties > Values."
2. Enter the desired "Start value."

Alternative procedure

You can also configure the start value directly in the tag table. To view hidden columns, activate the column titles using the shortcut menu.

Result

The start value you selected for the tag is transferred to the project.

See also

Tag limits (Page 5443)

Start value of a tag (Page 5445)

Updating the tag value in Runtime (Basic Panels)

Introduction

Tags contain process values which change while Runtime is running. Value changes are handled differently at internal and external tags.

Principle

When Runtime starts, the value of a tag is equal to its start value. Tag values change in Runtime.

In Runtime, you have the following options for changing the value of a tag:

- A value change in an external tag in the PLC.
- By input, for example, in an I/O field.
- By running a system function, such as "SetValue."

Updating the Value of External Tags

The value of an external tag is updated as follows:

- Cyclic in operation
  If you select the "Cyclic in operation" acquisition mode, the tag is updated in runtime as long as it is displayed in a screen. The acquisition cycle determines the update cycle for tag value updates on the HMI device. Cyclic acquisition is based on the selected scan cycle time.
- Cyclic continuous
  If you select the "Cyclic continuous" acquisition mode, the tag will be updated continuously in Runtime, even if it is not in the currently-open screen. This setting is activated for tags that are configured to trigger a function list when their value changes, for example.
  Only use the "Cyclic continuous" setting for tags that must truly be updated. Frequent read operations increase communication load.
- On demand
  If you select the "On demand" acquisition mode, the tag is not updated cyclically. It will only be updated on request, for example, by using the "Update Tag" system function.

See also

Tag limits (Page 5443)

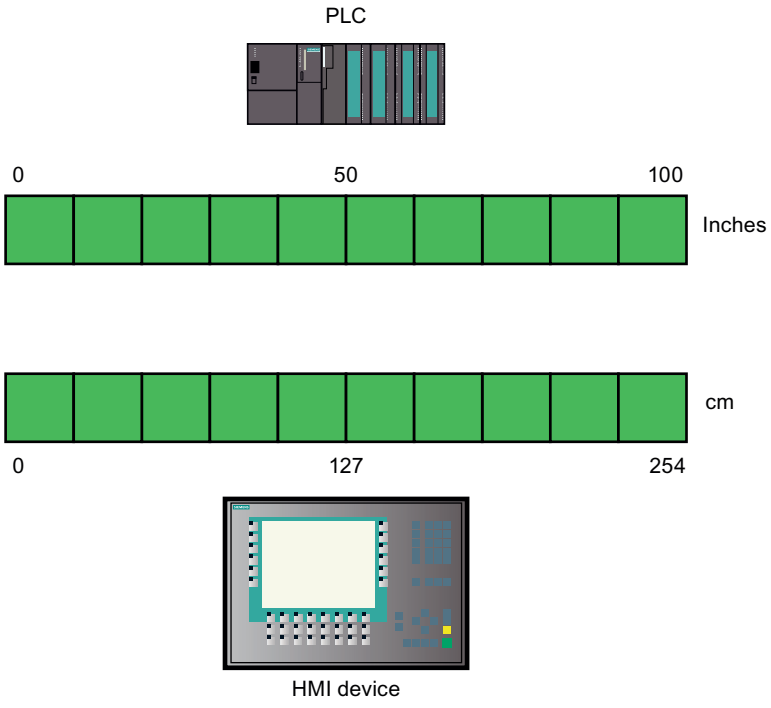
Linear scaling of a tag (Basic Panels)

Introduction

Numeric data types can be processed with linear scaling. The process values in the PLC for an external tag can be mapped onto a specific value range in the project.

Principle

To apply linear scaling to a tag, you must specify one value range on the HMI device and one on the PLC. The value ranges will be mapped to each other linearly.



As soon as data from the HMI device is written to an external tag, it will be automatically mapped to the value range of the PLC. As soon as data from the HMI device is read from the external tag, a corresponding transformation will be performed in the other direction.

Note

You can also use the system functions "LinearScaling" and "InvertLinearScaling" to automatically convert process values.

Application example

The user enters length dimensions in centimeters but the PLC is expecting inches. The entered values are automatically converted before they are forwarded to the controller. Using linear scaling, the value range [0 to 100] on the PLC can be mapped onto the value range [0 to 254] on the HMI device.

See also

Tag limits (Page 5443)

Applying linear scaling to a tag (Basic Panels)

Introduction

To apply linear scaling to a tag, you must specify one value range on the HMI device and one on the PLC. The value ranges will be mapped to each other linearly.

Requirement

- The external tag to which linear scaling is to be applied must exist.
- The Inspector window with the properties for this tag is open.

Procedure

To apply linear scaling to a tag, follow these steps:

1. In the Inspector window select "Properties > Properties > Linear scaling."
2. Click on "Enable" to switch on linear scaling.
   Using this option, you can temporarily switch off linear scaling for testing purposes, for example. Settings which were made earlier for linear scaling remain unchanged.
3. In the "PLC" area, enter the start and end values of the value range to be applied to the process values on the PLC.
4. In the "HMI device" area, enter the end and start values of the value range to be applied to the process values on the HMI device.

Result

In Runtime the data will be automatically mapped from one value range to the other.

Note

You can also use the "LinearScaling" and "InvertLinearScaling" system functions to automatically convert process values.

See also

Tag limits (Page 5443)

Connecting a tag to another PLC (Basic Panels)

Introduction

In WinCC, you can change the PLC connection of a tag at any time. This is needed when you change the configuration of your plant, for example.

Depending on the PLC selected, you may need to modify the configuration of the tag. The tag properties which must be changed will be highlighted in color.

Requirement

- The external tag, whose connection you wish to change, must already exist.
- The connection to the PLC must already exist.
- The Properties window for this tag is open.

Procedure

To change the PLC connection of a tag, proceed as follows:

1. In the Inspector window select "Properties > Properties > General."
2. Select the new connection in the "Connection" field.
   The tag properties that you must change will be highlighted in color in the tag table and in the Inspector window.
3. Change all highlighted properties of the tag to suit the requirements of the new PLC.

Result

The external tag is connected to the new PLC.

See also

Tag limits (Page 5443)

Indirect addressing of tags

Principle

In multiplexes, a type of indirect addressing, the tag used is first determined at runtime. A list of tags is defined for the multiplex tag. The relevant tag is selected from the list of tags in runtime. The selection of the tag depends on the value of the index tag.

In Runtime, the system first reads the value of the index tag. Then the tag which is specified in the corresponding place in the tag list is accessed.

Application example

Using indirect addressing, you could configure the following scenario:

The operator selects one of several machines from a selection list. Depending on the operator's selection, data from the selected machine will be displayed in an output field.

To configure such a scenario, configure the index tag for a symbolic I/O field. Configure the multiplex tag at an I/O field. Configure the tag list of the multiplex tag to reflect the structure of the selection list.

If the operator selects another machine, the value of the index tag will change. The selection field then displays the content of the tag that is pointed to in the tag list in the multiplex tag by the new index value.

See also

Tag limits (Page 5443)

Addressing tags indirectly

Introduction

With indirect addressing, the tag used is first determined at runtime. Instead of a single tag, a list of tags is defined. The list entries consist of an index value and the name of the tag to be used. You can control which entry in the tag list is accessed using an index tag.

Note

Data type of the index tags

If you use tags in the tag list with different data types, the properties of the largest data type are used. If you use tags of the INT and REAL data types, for example, the values are shown in floating point numbers.

Requirement

- The tag which you wish to address indirectly must already exist.
- The index tag must exist.
- The tags which will be contained in the tag list must already exist.
- The Inspector window with the tag properties is open.

Procedure

To address tags indirectly, proceed as follows:

1. In the Inspector window select "Properties > Properties > Multiplexing".
2. Select the "Multiplexing" option to activate indirect addressing.
   Using this option, you can temporarily switch off indirect addressing for testing purposes, for example. Settings which were made earlier for indirect addressing remain unchanged.
3. Select an "Index tag" or define a new tag using the object list.
4. Click the first entry in the "Tags" column in the tag list.
5. Select a tag as a list entry or define a new tag using the object list.
   The entry in the "Index" column will be generated automatically.
6. Repeat step 5 for all tags that you wish to add to the tag list.
7. If necessary, you can use drag-and-drop to change the order of the entries in the list.

Result

In runtime, the system will dynamically access the tag in the tag list which has the same index value as the value currently in the index tag.

See also

Tag limits (Page 5443)

Using tags to trigger functions (Basic Panels)

Introduction

You can use the values of variables as the triggering event for an action in runtime. To start an action in Runtime, configure a function list for a tag. Include one or more system functions in the function list. The function list is processed when the configured event occurs.

The following events are available for a tag:

- Change in value of the tag
  Function list processing is triggered by each change in the value of the variable.
  If the tag contains arrays, the function list is processed whenever an element of the array changes.
- Violation of the tag's high limit
  The function list is processed when the high limit is violated.
- Violation of the tag's low limit
  The function list is processed when the low limit is violated.

Requirement

- The tag whose value you wish to use as an event already exists.
- The Inspector window with the properties for this tag is open.

Procedure

To configure a tag with a function list, proceed as follows:

1. Under "Properties > Events >" in the Inspector window, select the event for which you want to create a function list.
   The function list associated with the selected event is shown.
2. Click "<Add function>". The second table column contains a selection button.
3. Click the selection button and select a system function.
4. Define the parameter values.

Result

The function list is processed when the configured event occurs in Runtime.

See also

Tag limits (Page 5443)

Defining the acquisition cycle for a tag (Basic Panels)

Introduction

The value of an external tag can be changed in Runtime by the PLC to which the tag is linked. To ensure that the HMI device is informed of any changes in tag values by the PLC, the values must be updated on the HMI. The value is updated at regular intervals while the tag is displayed in the process screen or is logged. The interval for regular updates is set with the acquisition cycle. The update can also be made continuous.

Requirement

- You have created the tag for which you want to define an acquisition cycle.
- The Inspector window with the tag properties is open.

Procedure

To configure an acquisition cycle for a tag, follow these steps:

1. In the Inspector window select "Properties > Properties > General."
2. If you want to update the tag at regular intervals as long as it is being displayed on the screen or logged, select "Cyclic in operation" as the acquisition mode.
   Or:
   If you want to update the tag at regular intervals even though it is not being displayed on the screen or logged, select "Cyclic continuous" as the acquisition mode.
   The "Cyclic continuous" setting is selected for a tag, for example, that has a function list configured for a change of its value and that is not directly visible in a screen.
3. Select the required cycle time in the "Acquisition cycle" field or define a new acquisition cycle using the object list.

Alternatively, you can configure the acquisition cycle directly in the work area of the tag table. To view hidden columns, activate the column titles using the shortcut menu.

Note

Only use the "Cyclic continuous" acquisition mode for tags that really have to be continuously updated. Frequent read operations generate a heavy communication load.

Note

For structured HMI tags, the acquisition mode can be selected for the respective structured HMI tag as well as their individual subordinate elements. The acquisition mode of the elements of the array tag cannot be changed. The elements of the array tag is automatically assigned to the acquisition mode of the higher-level array tag.

When the acquisition mode of structured HMI tags is changed, it is applied to all subordinate elements. This means changing the acquisition mode may overwrite subordinate elements.

Result

The configured tag is updated in Runtime with the selected acquisition cycle.

See also

Tag limits (Page 5443)

Address multiplexing (Basic Panels)

Introduction

Using address multiplexing, you can use a single tag to access a multitude of memory locations within the PLC's address range. You read and write to the addresses without defining a tag for each individual address.

The square brackets indicate address multiplexing.

Multiplexing with absolute addressing

When using multiplexing with absolute addressing, you configure tags as placeholders for the address in the PLC to be addressed.

If you want to access, for example, an address of the format "%DBx.DBWy", the expression for multiplexing is as follows:

"%DB[HMITag1].DBW[HMITag2]"

In Runtime, you supply the tag "HMITag1" with the required value for the data block you want to address.

In Runtime, you supply the tag "HMITag2" with the required address from the data block.

Tags are supplied with values, for example, with the help of values from the PLC or via a script.

Multiplexing with absolute addresses is supported for the following PLCs and communication drivers.

- SIMATIC S7-300/400
- SIMATIC S7-1200
- SIMATIC S7-1500

Multiplexing with absolute addresses is not available for data blocks with optimized access.

Multiplexing with symbolic addressing

When multiplexing with symbolic addressing, you access various array elements of an array tag in a data block of the connected controller by means of a multiplex tag and an index tag.

First, connect an element of the desired array tags in a data block to the multiplex tags in the HMI device. Then replace the array index (leftmost in the symbolic address) with the index tag.

If you want to access, for example, the various elements of the array tag "Arraytag_1" in the data block "Datablock_1", the expression for symbolic addressing is as follows:

"Datablock_1.Arraytag_1["HMITag_1"]

You control the access to the index of the array elements with the HMI tag "HMITag_1". In Runtime, you supply the tag with the index of the array element that you want to access.

Multiplexing with symbolic addressing is only available if the following components support symbolic addressing:

- HMI device
- PLC
- Communication driver

Symbolic addressing is supported by the following communication drivers:

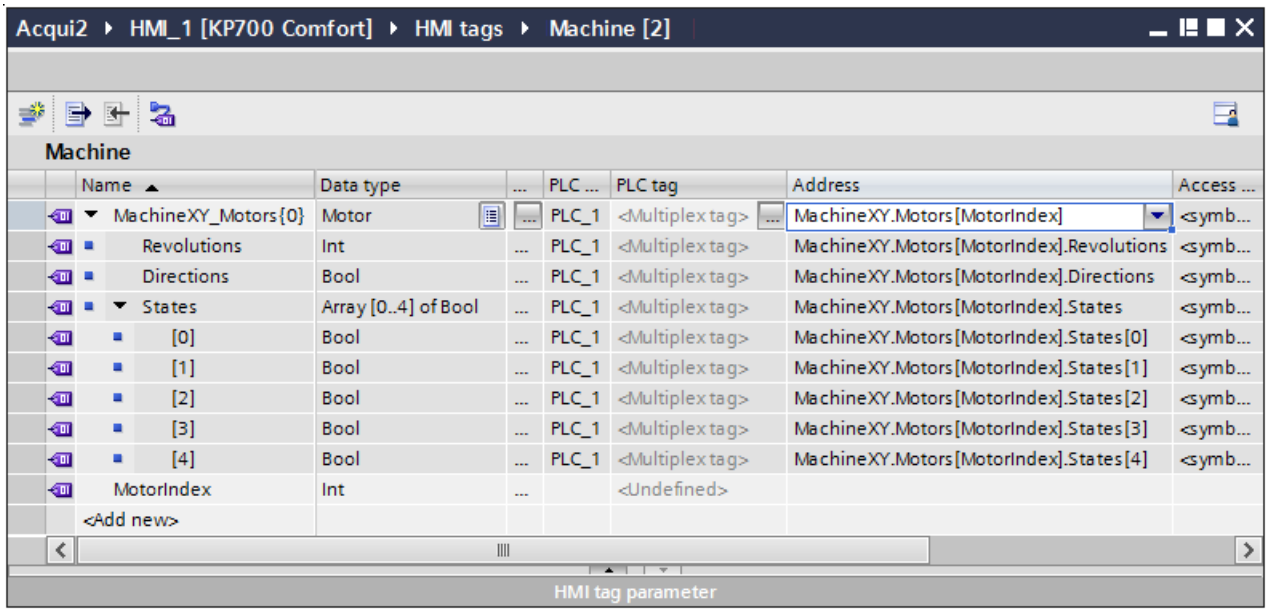
- SIMATIC S7-1200
- SIMATIC S7-1500

Structured HMI tags with user-defined PLC data types (UDT) and arrays can be used for multiplexing with symbolic addressing.

Multiplexing is possible under the following conditions:

- The path of the connected PLC tag includes at least one array index.
- The array, which is left in the path of the PLC tags, is one-dimensional.
- If there are multiple arrays in the path of the PLC tags, only the array at the far left is available for multiplexing.
- The index tag must be of the "USInt", "UInt", "UDInt", "SInt", "Int" or "DInt" data type.

The following figure shows multiplexing with symbolic addressing using a structured HMI tag with custom PLC data types (UDT). The data block element "db1.udt3Arr[0]" is linked to the HMI tag "db1_udt3Arr{0}". Symbolic addressing of the HMI tag "db1_udt3Arr{0}" is multiplexed by the "index" HMI tag:



Name	Data type	...	PLC ...	PLC tag	Address	Access ...
MachineXY_Motors{0}	Motor	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex]	<symb...
Revolutions	Int	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex].Revolutions	<symb...
Directions	Bool	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex].Directions	<symb...
States	Array [0..4] of Bool	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex].States	<symb...
[0]	Bool	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex].States[0]	<symb...
[1]	Bool	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex].States[1]	<symb...
[2]	Bool	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex].States[2]	<symb...
[3]	Bool	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex].States[3]	<symb...
[4]	Bool	...	PLC_1	<Multiplex tag>	MachineXY.Motors[MotorIndex].States[4]	<symb...
MotorIndex	Int	...		<Undefined>		
<Add new>						

The following tags can not be used as index tags for multiplexing with symbolic addressing:

- HMI tags that are not of the "USInt", "UInt", "UDInt", "SInt", "Int" or "DInt" data type
- HMI tags that are already multiplexed
- Array elements
- HMI tags that are elements of structured HMI tags

Note

In Runtime, the first array element is always addressed with the index value "0". Have your array element in the data blocks also start with the index "0".

Note

You can export the HMI tags that are used for multiplexing with symbolic addressing to MS Excel. You can also import such HMI tags from MS Excel into an HMI tag table that contains all HMI tags of the HMI device.

See also

Tag limits (Page 5443)

Configuring address multiplexing with absolute addressing (Basic Panels)

Introduction

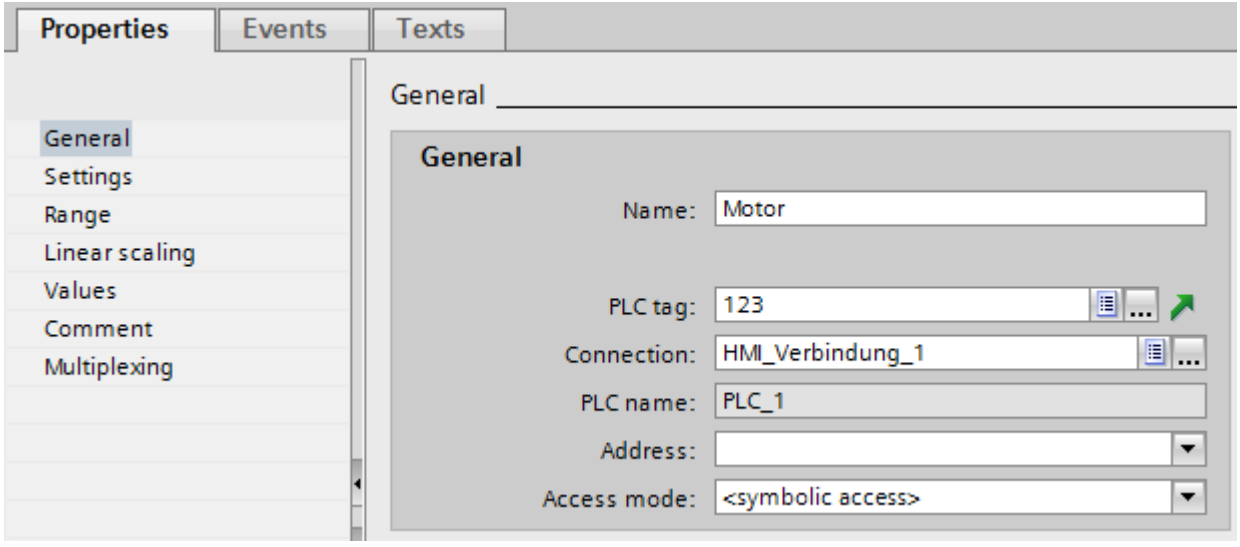
When using address multiplexing, you can efficiently access different addresses in the PLC with the help of a small number of tags. Instead of the absolute address in the PLC, you use tags in order to be able to change the address in Runtime.

Requirement

- The tag for address multiplexing is created and connected to the PLC.
- The Properties window for this tag is open.

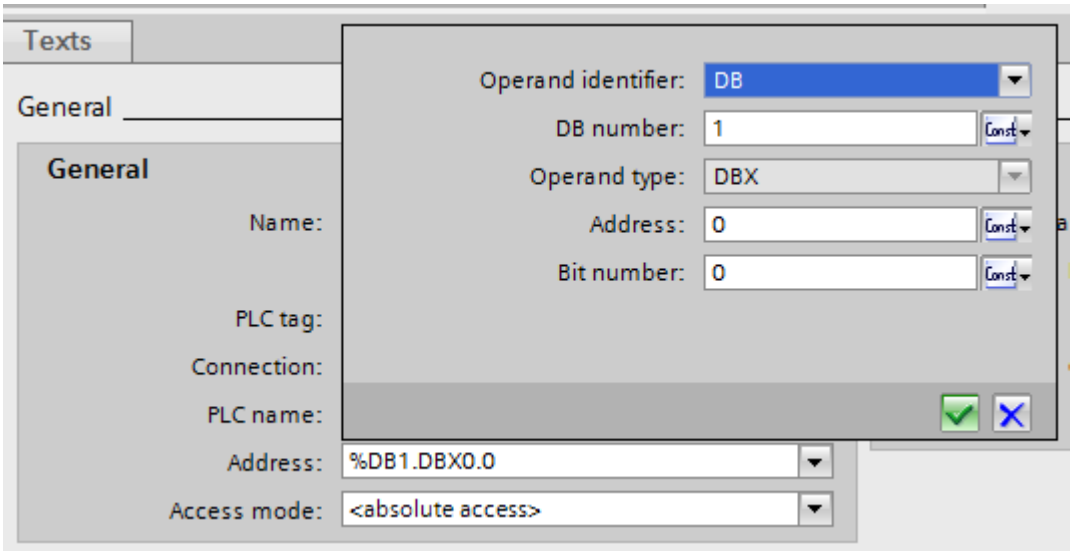
Procedure

1. Select the tag for address multiplexing in the tag table, and select "Properties > Properties > General" in the Inspector window. The general properties of the tag are displayed.

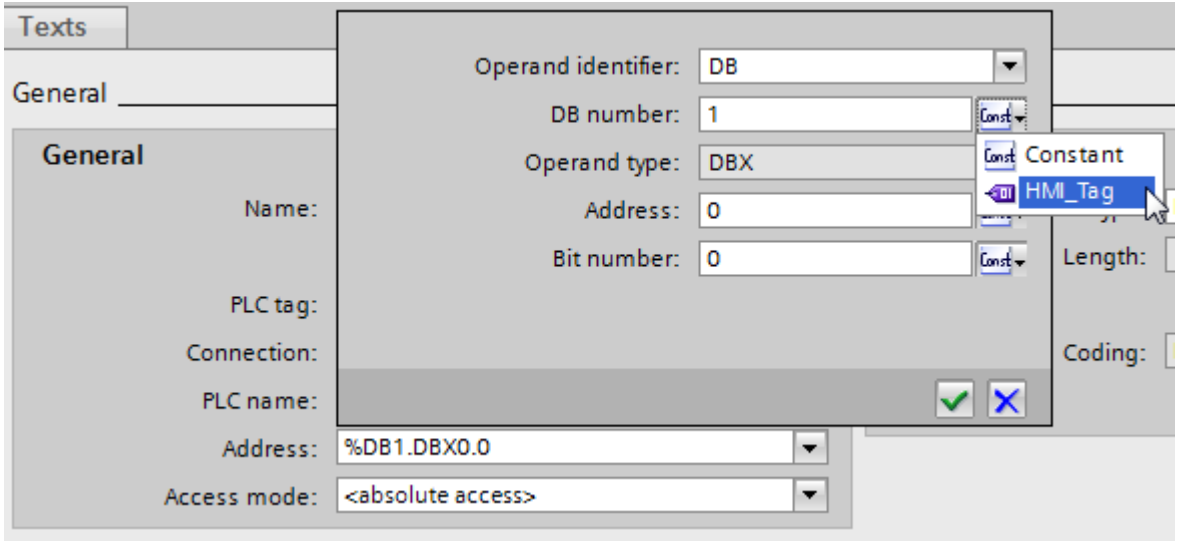


2. Select the "Int" data type for this example.
3. Select the access type "Absolute addressing".

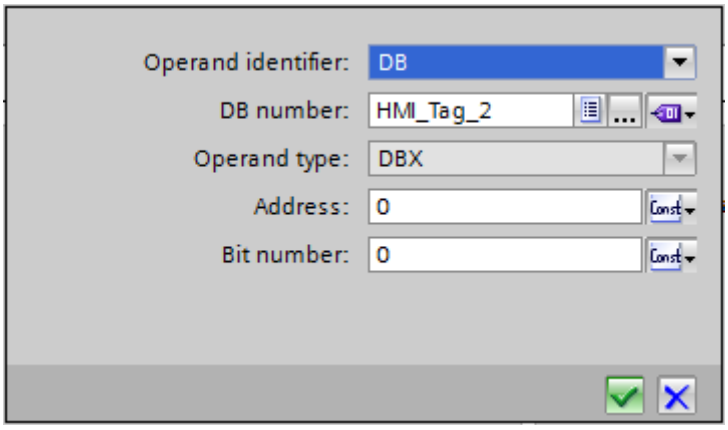
4. Click the selection button in the "Address" field. The address dialog opens.



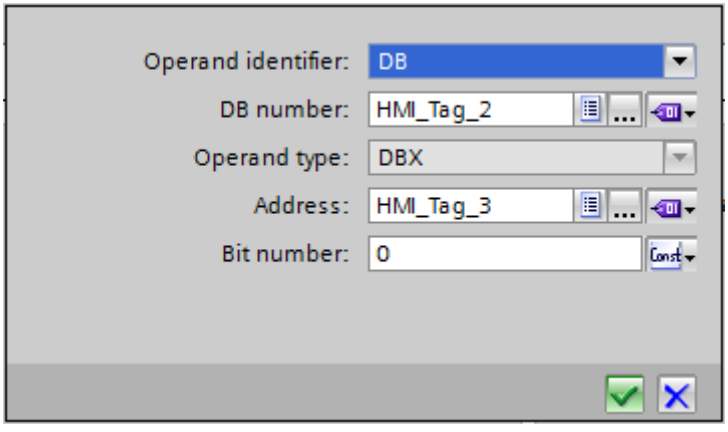
5. Click the selection button in the "DB number" field and select the entry "HMI tag".



6. In the "DB number" field, click the ... button and select a tag for the DB number in the object list. Or create a new tag with the help of the object list. Accept the tag by clicking the button.



7. Repeat steps 3 and 4 for the "Address" field and configure a further tag for calling the address area in the data block.



The selection options in the Address dialog depend on the selected data type of the multiplex tag. The Address dialog offers only address settings that can be configured with the selected data type.

Result

In runtime, the multiplex tag is used to access the memory location corresponding to the address currently found in the tag. You control access to the data block with the tag in the DB number field. You control access to the address in the selected data block with the tag in the "Address" field.

Note

The value in the memory location will only be read at the next update cycle for the addressed tag.

If, for example, you use a multiplex tag in a script, do not attempt to access contents of the memory location directly after changing it.

See also

Tag limits (Page 5443)

Configuring address multiplexing with symbolic addressing (Basic Panels)

Introduction

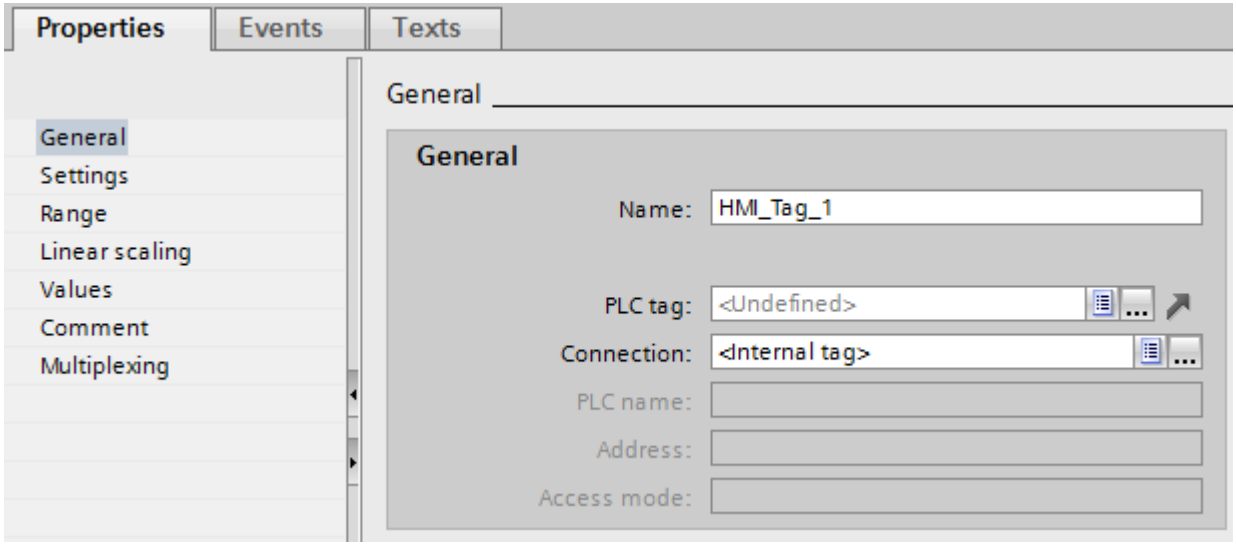
When using address multiplexing, you can efficiently access different addresses in the PLC with the help of a small number of tags. Instead of the symbolic address in the PLC, you use tags in order to be able to change the address in Runtime.

Requirement

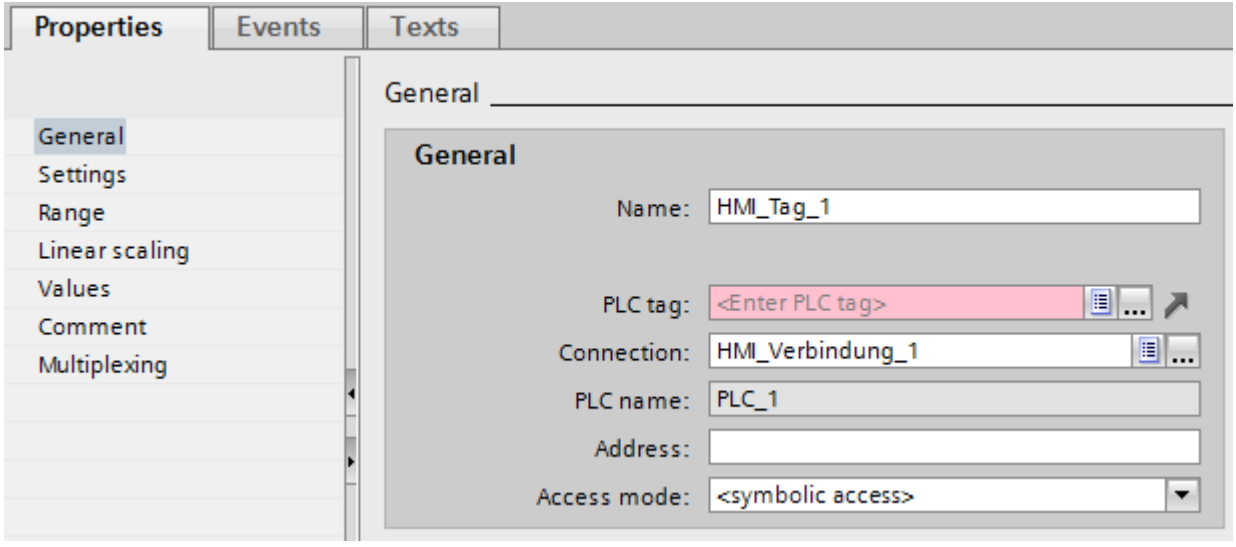
- The tag for address multiplexing is created.
- The Properties window for this tag is open.
- A data block with an array tag is created in the connected PLC.
- The data block was compiled.

Procedure

1. Select the tag for address multiplexing in the tag table, and select "Properties > Properties > General" in the Inspector window. The general properties of the tag are displayed.

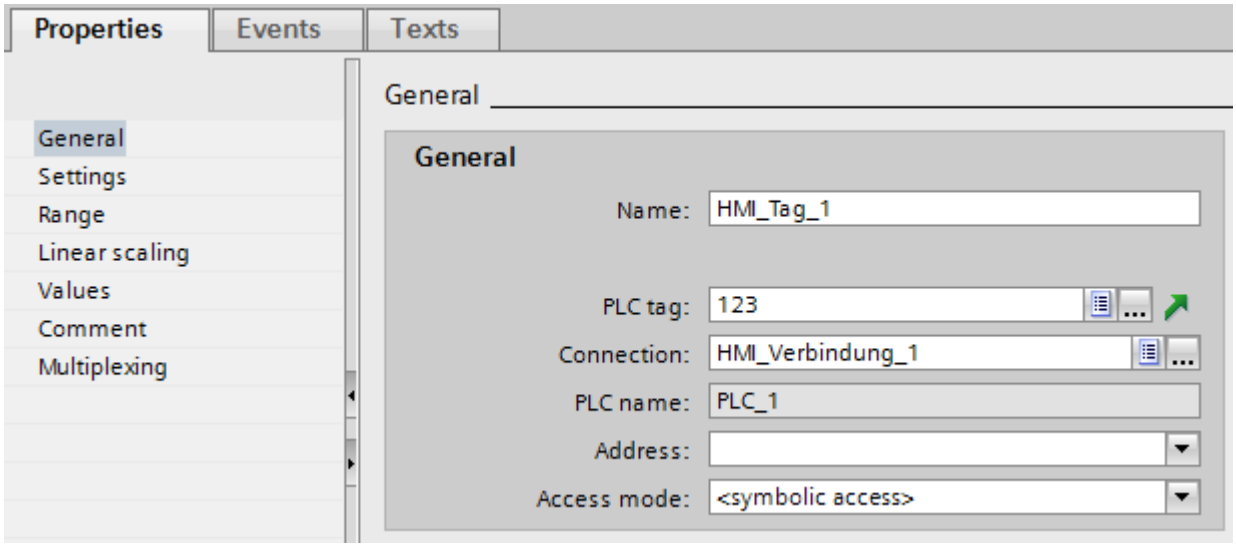


2. Select the connection to the PLC via the "Connection" field.

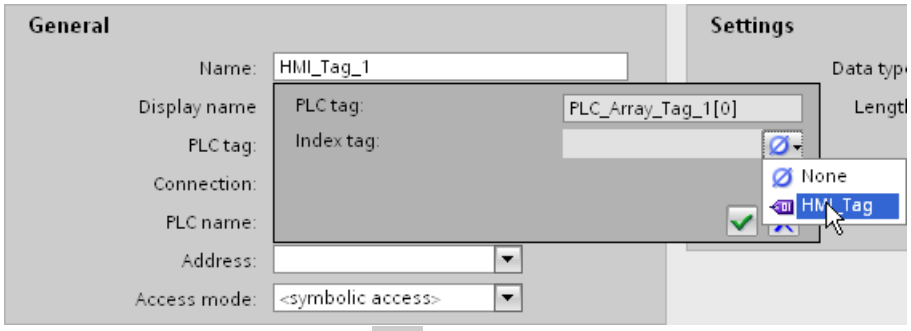


3. Select the access type "Symbolic access".

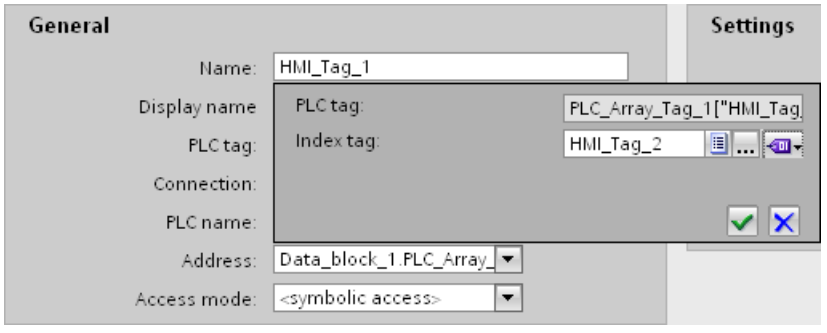
4. Navigate to the data block of the PLC via the "PLC tag" field and select an array element of the array tag.



5. Click the selection button in the "Address" field. The address dialog opens.
6. Click the selection button in the "Index tag" field and select the entry "HMI tag".



7. In the "Index tag" field, click the ... button and select a tag for the array index in the object list. Or create a new tag with the help of the object list. Accept the tag by clicking the button.



Result

In Runtime, the array element whose index value is contained in the index tag is accessed.

See also

Tag limits (Page 5443)

Making characters available for HMI tags of the type WString/WChar

Introduction

In tags of the type WString or WChar, you specify strings that can be displayed with specific fonts. All characters used in the project are available to you.

The Tahoma font is fully pre-installed on the Basic Panels. The characters of most languages can be displayed with the Tahoma font. The characters of the Chinese, Taiwanese, Japanese and Korean languages cannot be displayed with Tahoma.

Only those characters that are used in the texts in the configuration are transferred to the Basic Panels via the configuration. Therefore, it is possible that characters that are displayed via tags of the WString or WChar type are not shown.

To use fonts that are not installed on an HMI device in Runtime, provide the characters as configured texts in a screen.

Procedure

To use fonts that are not installed on an HMI device, provide the characters as configured texts in a screen.

1. Create a new screen.
2. Insert a text box.
3. In the properties of the text box, set a font that contains characters that you wish to use later.
4. Insert the characters of the font that you wish to use later into the text box.

After the project is compiled, you can use the characters configured in the text in HMI tags of the type WString/WChar.

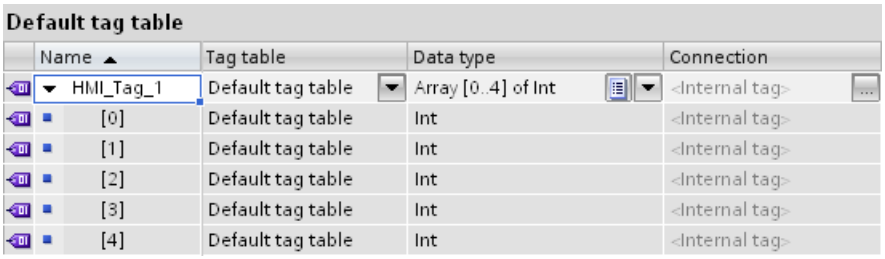
12.2.3 Working with arrays (Basic Panels)

12.2.3.1 Basics on arrays (Basic Panels)

Definition

A data list with data elements of one single data type is known as an array. This data list is termed an array tag or array for short. The data elements are termed array elements.

The individual array elements can be addressed over one or more integer indices. The properties of an array element are largely identical and can be configured centrally in the array tags.



Advantages

You can combine a large number of similar array elements in a structured manner with just one array tag. The common properties of array elements of the same kind can be defined centrally in the array tag. In the configuration, you then use each individual array element as an individual tag with the same data type.

Restrictions

The following restrictions apply to the use of arrays:

- Not all HMI devices support array tags.
- An array can contain only one dimension. Multi-dimensional arrays of controllers are mapped to individual tags on the HMI systems.
- Only elementary data types with no subordinate structures can be used as the data types of arrays. In other words, only scalar arrays are supported.
- The lower index of an array must begin with "0". With other lower limits on the controllers, the indexes are shifted accordingly.
- Unlike individual tags, scalar arrays do not support areas, linear scaling or tag multiplexing.

Application examples

Array tags can be used in the following situations:

- To group process values in profile trends: You map process values to trends which are acquired at different points in time, for example.
- To access specific values which are grouped in trends: You output all values of the profile trend by stepping up the index tag, for example.
- To configure discrete alarms with successive bit number.
- To save the complete machine data records in a recipe.

Tag counting with arrays

- Scalar arrays on HMI devices and RT Advanced are counted as one tag.
- In RT Professional, however, each array element they contain is counted separately.
- Multi-dimensional arrays of controllers are mapped to individual tags of the same data type and counted like these.
- Structured controller arrays are mapped to and counted like individual tags of the corresponding data type.

Special features

On HMI devices and in RT Advanced, all array elements are always read or written by the controller at once when an element in a scalar array is accessed (with the exception of arrays of String/WString).

Example for HMI devices and RT Advanced:

- An array tag consisting of 100 array elements of the data type "Real" has been configured.
- If an array element with a length of 4 bytes changes, 100 x 4 bytes are written to the controller.

WARNING

Increased system load and performance losses

As a result of the simultaneous transfer of all array elements in a scalar array, writing to the controller in particular when a large array is accessed takes longer than writing a tag with the same data type. Take this into consideration when designing communication.

Please also note that with scalar arrays, the controller reading entire array tags is usually faster than the sum of instances of read access to the same number of elementary tags of the same data type.

CAUTION

Data inconsistency at array tags

If the value of a single element is to be changed in a scalar array tag, the whole array is read, changed and rewritten as a complete array. Changes carried out in the meantime to other array elements in the PLC are overwritten during rewriting.

You should always prevent different parts of the system, for example the HMI device and the controller, from writing values to the same array tag at the same time. Use synchronous transfer of recipe data records, for example, to synchronize an array tag with the controller.

Note

Array tags as a list entry of multiplex tags

You can use array tags of the Char or WChar data type like tags of the String data type.

The use of an array tag of the Char or WChar data type as the list entry of a multiplex tag in the "HMI tags" editor is not, however, supported.

See also

Creating array tags (Page 5466)

Examples of arrays (Page 5467)

Basics of tags (Page 5421)

Basic Panel (Page 5962)

Basic Panel 2nd Generation (Page 5966)

12.2.3.2 Creating array tags (Basic Panels)

Introduction

Create an array tag to configure a large number of tags of the same data type. The array elements are saved to a consecutive address space.

You can create an array tag as an internal tag or as an external tag.

Note

Connecting HMI tags to PLC array tags

If you want to create an array tag as an external tag, first configure an array tag in a data block of the connected PLC. You then connect the array tag to an HMI tag.

Note

HMI array tags of the following data types are only supported with PLC array tags:

- Char
- WChar
- String
- WString

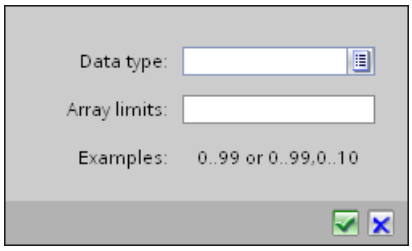
Creating HMI array tags

To create an array tag, follow these steps:

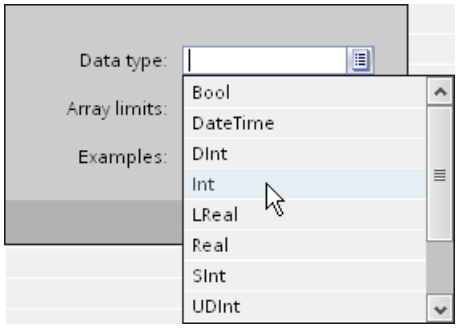
1. Open the HMI tag table.
2. Double click <Add> in the "Name" column of the HMI tag table.
   A new HMI tag is created.

Configuring HMI array tags

1. Click in the Data type column of the HMI tag table and select the "Array" data type.
2. Click in the data type column.
   The dialog box for configuring the array is opened.



3. Select the desired data type for the array tag in the "Data type" field.



4. Define the number of array elements in the "Array limits" field. The lower limit must begin with "0".
5. Click . The settings for the array are saved.
6. As with tags with elementary data types, you still need to assign the address of the array tag in the PLC.
7. Save the project.

Result

An array tag is created. The properties of the array elements are inherited from the parent array tag.

See also

Basics on arrays (Page 5463)

12.2.3.3 Examples of arrays (Basic Panels)

Introduction

Array tags combine a large number of tags, e.g. 100 array elements.

The common properties of array elements of the same kind can be defined centrally in the array tag. In the configuration, you then use each individual array element as an individual tag with the same data type. However, array elements cannot be used as parameters in function lists.

You use array tags as complete arrays in the following places:

- In the "Alarms" editor
- In the "Recipes" editor
- For address multiplexing
- In function lists

Examples

You can drag a PLC array tag straight from the PLC tag table to the HMI tag table.

Alternatively, you can configure an array tag separately with the corresponding number of array elements to handle multiple tags of the same data type.

- The individual array elements can be accessed indirectly by means of an index tag, for example.
- Use these index tags to operate and monitor the array elements.

See also

Basics on arrays (Page 5463)

12.2.4 Working with cycles (Basic Panels)

12.2.4.1 Cycle basics (Basic Panels)

Introduction

Cycles are used to control actions that regularly occur in runtime. Common applications are the acquisition cycle and the update cycle.

Principle

In Runtime, actions that are performed regularly are controlled by cycles. Typical applications for cycles:

- Acquisition of external tags
  The acquisition cycle determines when the HMI device will read the process value of an external tag from the PLC. Set the acquisition cycle to suit the rate of change of the process values. The temperature of an oven, for example, changes much more slowly than the speed of an electrical drive.
  Do not set the acquisition cycle too low, since this will unnecessarily increase the communication load of the process.

The smallest possible value for the cycle depends on the HMI device that will be used in your project. For most HMIs, this value is 100 ms. The values of all other cycles are always an integer multiple of the smallest value.

Application example

You can use cycles for the following tasks:

- To regularly update a tag.
- To draw attention to maintenance intervals.

See also

Basics of tags (Page 5421)

12.2.5 Archive tags (Basic Panels)

12.2.5.1 Basic principles for data logging

Introduction

Data logging is used to collect, process and log process data from an industrial system.

When you analyze the logged process data, you can extract important business and technical information regarding the operational state of the system.

Application of the data logging

You can use data logging to analyze error statuses and to document the process. By analyzing data logs, you can extract the information necessary to allow you to optimize maintenance cycles, increase product quality and ensure that quality standards are met.

12.2.5.2 Working with data logs (Basic Panels)

Logging Tags (Basic Panels)

Introduction

In Runtime, you store tag values in logs. You can then analyze the logged data at a later date. You define the following conditions for logging a tag:

- The log tag, through which the values of the connected tag are logged.
- The data log in which the tag is stored.
- The cycle or the event with which the tag is stored.
- The value range within which the tag is stored.

Note

You use data logging mainly to log the values of external tags. You can also log the values of internal tags.

Principle

Several steps come together during data logging:

- Creating and configuring the data log
  When you create a data log, you define the following settings:
  - General settings, such as name, size and storage location
  - Runtime start characteristics
  - Behavior when the log is full
- Configuring the logging of tags
  For every log tag, specify a data log in which the values of the connected tags and other information are logged, such as the time of logging.
  When and how often the values of a log tag are logged are also defined. You have the following options:
  - "On demand":
    The tag values are logged by calling the "LogTag" system function.
  - "On change":
    The tag values are logged as soon as the HMI device detects a change in the value of the tag.
  - "Cyclic":
    The tag values are logged at regular intervals. You can supplement the default cycles in WinCC with your own cycles based on the default cycles. The smallest value that can be set is 1 s. All other values are integer multiples of this value.
    You can also restrict the logging of values within or outside a tolerance band. In this way, the logging of values within a relevant range of values is restricted.

  Note the following points if you want to log a tag on demand:
  - Do not log this type of tag in a segmented circular log in which tags are logged in a continuous cycle or in response to changes.

  Background:
  - If logging takes place on demand only rarely, the log segment will be filled by cyclically-logged values, for example, and the next log segment will be created. If you then attempt to access the tag that was logged on demand, it will not be possible to display the tag since it is the current log segment that is accessed in Runtime. To remedy this, you should create a separate data log for tags that are logged only rarely.
- Further processing logged tag values
  You can analyze the logged tag values directly in your project, such as in a trend view, or in another user program, such as Excel.

Creating a data log (Basic Panels)

Introduction

Data logs are used to store values from external and internal tags in runtime. When you create a log, you must give it a name and define its size and storage location. In addition you can enter comments for each log.

Requirement

- A project is open.
- The Inspector window is open.

Procedure

To create a data log, proceed as follows:

1. Double-click on the "Historical Data" entry in the project tree.
   The editor for data logs and alarm logs opens.
2. Open the "Data logs" tab and double-click "Add" in the "Name" column of the "Data logs" editor.
   A new data log is created.
3. In the Inspector window select "Properties > Properties > General."
4. Enter a unique name for the log in the "Name" field.
5. Define the number of data records to be logged in each log in the "Number of data records per log" field.
   The size of a log is calculated as follows: The number of items * the length of each tag value to be logged.
   In the Inspector window, the maximum size that the log can reach if the currently selected number of data records is observed is displayed beneath the "Number of data records" input field.
6. In the "Storage location" field you select where the log entries are saved.
7. Depending on the selected "Storage location", you either select the "Path" or the "Name of the data source".
8. If necessary, enter a descriptive text in the "Comment" category to document your configuration.

Alternatively you can configure log properties directly in the "Data log" editor. To view hidden columns, activate the column titles using the shortcut menu.

Result

The data log is created.

You can now configure tags so that their values are stored in this log.

To continue the log configuration, perform the following steps:

- Define the restart characteristics of the log when Runtime is started.
- Define how the log should behave when it is full.
- Configure a function list for the "Overflow" event.

See also

Storage locations of logs (Page 5563)

Managing logging behavior when Runtime starts (Basic Panels)

Introduction

When you configure a log, you define the restart characteristics of the log when Runtime is started. You define whether the logging should start when Runtime starts in the log properties. You can also define whether an existing log will be continued or overwritten.

You define the restart characteristics separately for each log.

Requirement

- You have created a log.
- The "Historical Data" editor is open.
- The Inspector window with the log properties is open.

Procedure

To configure the restart characteristics of a data log, proceed as follows:

1. Select the log for which you want to define the restart characteristics in the "Historical Data" editor.
2. In the Inspector window select "Properties > Properties > Restart behavior".
3. If you want logging to start when Runtime starts, enable the "Enable logging at runtime start" option in the "Logging" area.
   You can also start logging in Runtime using the "StartLogging" system function, for example.
4. Select the restart behavior of the log in the "Log handling at restart" area.
   - The logged values are deleted and logging is started again with the option "Reset log".
   - The option "Append data to existing log" is used to append the values to be logged to the existing log.

Alternatively you can configure the restart characteristics of a log directly in the "Historical Data" editor. To view hidden columns, activate the column titles using the shortcut menu.

Result

Logging will start in runtime according to your settings.

Log response to language switching in runtime (Basic Panels)

Introduction

In the Runtime settings of the HMI device, select the language to be used for writing to logs in runtime.

Requirements

- The languages used in your project are activated in the "Project languages" editor, for example, "German (Germany)" and "English (USA)" .

Procedure

1. Open the "Project languages" editor in the project tree under "Languages & resources".
2. Activate the runtime language, for example, "German (Germany)" and "English (USA)".
3. Open the runtime settings of your HMI device and select the entry "Language and font".
4. Activate the runtime languages in the "Activate" column.
5. Use the arrow buttons to set the "Language switch order":
   - German 0
   - English 1
     With "0", German is specified as the "Startup language".
6. Select the "General" entry from the "Runtime settings" dialog.
7. Select the "Logging language" in the "Logs" area.

Result

By selecting the logging language, the logs are always written in the selected language. This is true regardless of the language selected by the operator in runtime.

Controlling the Logging in relation to the Fill Level (Basic Panels)

Introduction

The size of a log is determined by the number of entries. You use the logging method to determine how the log responds when it is full.

Logging methods

The following logging methods are available:

-  Circular log
  When the configured log size has been reached, the oldest entries are deleted. When the configured log size has been reached, approximately 20% of the oldest entries are deleted. It is therefore not possible to display all the configured entries. During configuration, select an appropriate size for the circulation log. Alternatively, configure a segmented circular log.
-  Segmented circular log
  In a segmented circular log, multiple log segments of the same size are filled in succession. When all logs are completely full, the oldest log is overwritten.

-  Log that sends a system alarm when it is full
  When a defined level is reached, such as 90 %, a system alarm is triggered. When the log is 100% full, new tag values are not logged.
-  Log with level-dependent triggering of an event.
  When the log is completely full, the "Overflow" event is triggered. Configure a function list for the event that will be carried out when the "Overflow" event occurs. When the configured size of the log is reached, new tag values are not logged.
  The following system functions are available for further processing of full logs:

Requirement

- You have created a log.
- The "Historical Data" editor is open.
- The Inspector window with the log properties is open.

Procedure

1. Select the log for which you want to define the logging method in the "Historical Data" editor.
2. Select "Properties > Properties > Logging method" in the Inspector window and select the required logging method.
3. If you have selected the "Segmented circular log" type, enter the number of log segments.
   If you selected a log with the "Display system alarm on" setting, specify the level as a percentage at which a system alarm is to be triggered.
   If you selected the "Trigger event" setting, configure the function list in the "Events" group.

Alternatively you can configure the logging method directly in the "Historical Data" editor table. To view hidden columns, activate the column titles using the shortcut menu.

The "Overflow" event is not available in the editor table. You must therefore configure the function list in the Inspector window.

Result

The selected log responds according to the settings in Runtime.

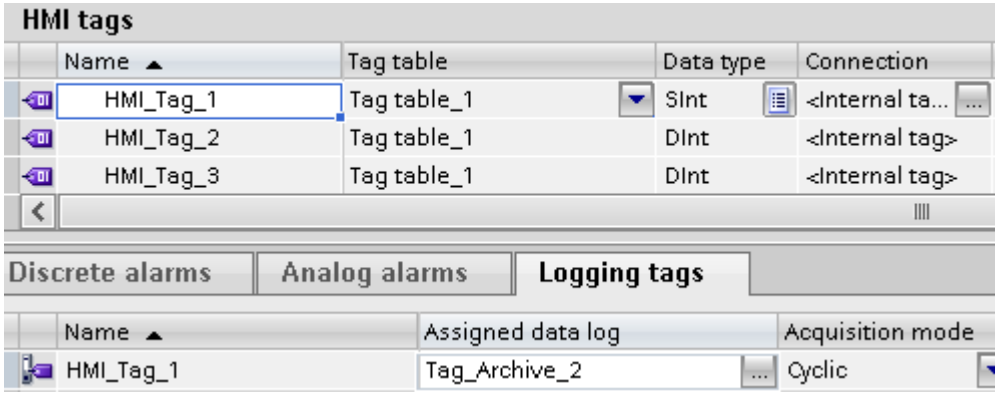
Logging process values (Basic Panels)

Introduction

You can save the process values of a tag in a data log in Runtime. You define the following conditions for logging a tag:

- The log tag, through which the values of the connected tag are logged.
- In which log the values are stored
- The conditions under which the values are stored
- If only process values for a certain range of values are stored

To log the tag values, assign a logging tag to an HMI tag. The logging tag is stored in the data log and logs the values of the connected HMI tag. You can configure logging tags directly in the "HMI tags" editor. The " HMI tags" editor contains the "Logging tags" editing table.



If the view of the "Logging tags" table is minimized, click the arrow button below the tag table.

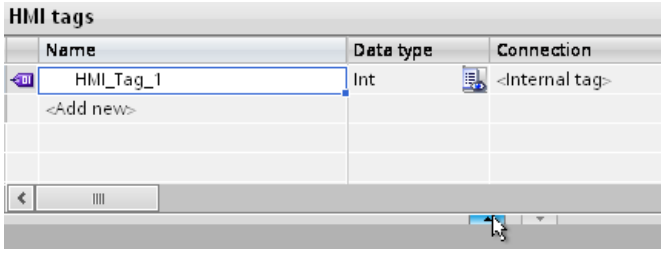


Figure 12-3 The "Logging tags" table is displayed.

Requirement

- The data log has been created.
- The tag for which you wish to configure the logging must already exist.
- The "Tags" editor is open.
- The "Logging tags" table is displayed.
- The Inspector window with the tag properties is open.

Procedure

To log process values in a tag, proceed as follows:

1. Select a tag in the tag table.
2. Double-click "Add" in the "Name" field in the "Logging tags" table.
   A new logging tag is created; it is given the same name as the associated HMI tag.
3. Select the data log in which the values of the tags are to be logged under "Properties > Properties > General" in the Inspector window.

4. Select "Properties > Properties > Logging type" in the Inspector window and select the log type for logging.
   - "Cyclic": The tag values are logged in accordance with the set logging cycle.
   - "On change": The tag values are logged, as soon as the operator device detects a change in value.
   - "On demand": The tag values are logged by calling the "LogTag" system function.
5. If you want to log tag values cyclically, select a cycle time in the "Logging cycle" area. Alternatively, you can define your own cycle using the object list. The smallest value that can be set is 1 s. All other values are integer multiples of this value.
6. If you only want to log tag values outside or inside a defined value range, select "Properties > Properties > Deadband for logging" in the Inspector window. Define the values for the high and low limits.
   If you want to configure a dynamic limit, select "HMI tag" using the selection button. In the second field, select the tag that contains the limit.
   If you want to configure a fixed limit, select "Constant." Enter the limit value in the second field.
   If you want to leave a limit undefined, select "None".
7. Under "Scope", specify whether tag values are to be logged only if they are within the defined limits or only if they are outside the defined limits.

Alternatively, you can configure the logging of a tag directly in the "Logging tags" editor table. To view hidden columns, activate the column titles using the shortcut menu.

You can also fully configure a logging tag in the "Historical Data" editor.

Result

The process values of the configured tag are logged in Runtime according to the selected settings.

Note

To have the tag values actually logged in runtime, you must ensure that the data log is started. Logging can be started automatically at start up or by means of a system function. To automatically start the log, use the setting in the log Properties window.

Triggering a system function when log is full (Basic Panels)

Introduction

You can select a logging method that executes a function list as soon as the log is full.

Application example

You can use system functions, for example, to swap the data from a full log file out to another log before the full log is overwritten. You can then continue to process the transferred data in another program. Assign the "CopyLog" system function to the "Overflow" event accordingly.

Requirement

- A log with the "Trigger event" logging method is created.
- The "Data log" editor is open.
- The Inspector window with the data log properties is open.

Procedure

To configure a system function for the "Overflow" event, proceed as follows:

1. Select the required log in the table in the "Data Log" editor.
2. In the Inspector window select "Properties > Events > Overflow."
   The function list will open.
3. Double-click "Add function" and select the desired system function.
4. Configure the required parameters of the selected system function.

Result

During runtime, the function list will be executed as soon as the log is full.

Logging tags in the "Historical Data" editor (Basic Panels)

Configuring logging tags (Basic Panels)

Introduction

You can also create and edit logging tags in the "Historical Data" editor in WinCC. You also directly edit the properties of logging tags in the "Historical Data" editor.

Note

If you delete, move or copy in the "Historical Data" editor, the changes also take effect in the tag table.

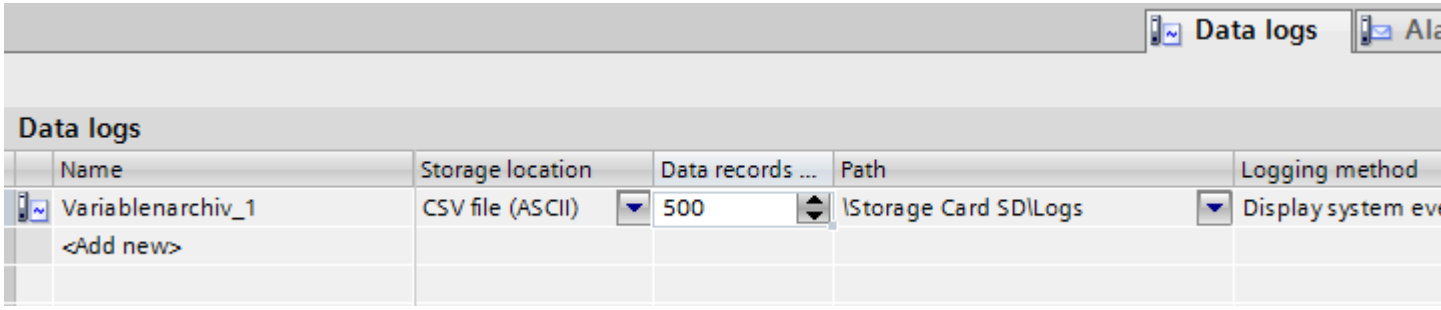
Requirement

- The "Data logs" tab is open in the "Historical Data" editor.
- A data log has been created.

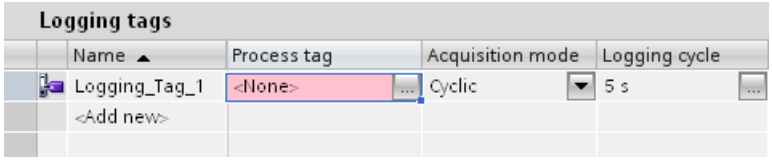
Procedure

Proceed as follows to configure a logging tag in the "Historical Data" editor:

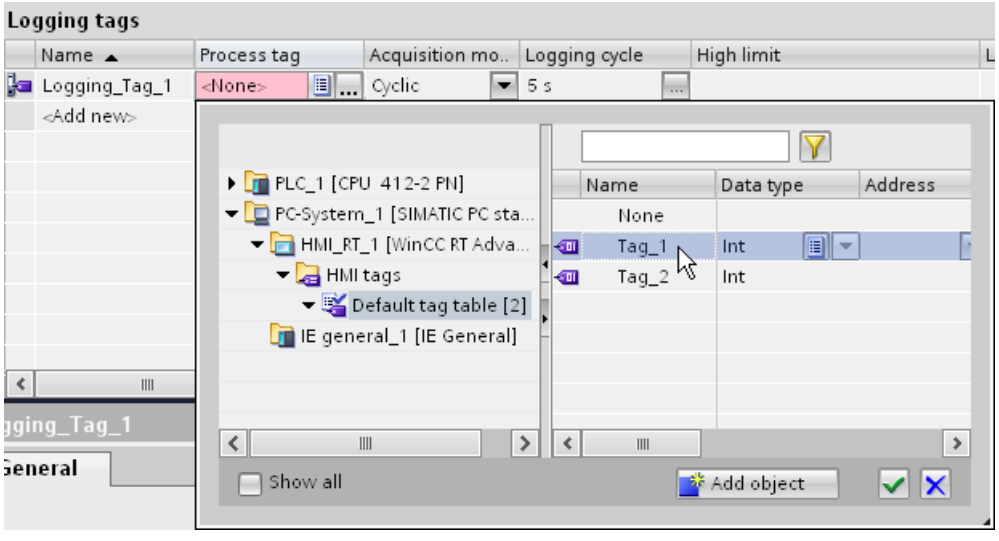
1. Select an existing data log in the "Data logs" table of the editor.
   Alternatively, double-click on "Add..." in the "Name" column to create a new data log.



2. Double-click "Add ..." in the "Name" column of the editor "Logging tags" table.



3. Enter a unique name for the logging tag in the "Name" field.
4. In the "Process tag" field, click on the selection button and select the process tag for logging in the object list.



5. Select the desired trigger mode in the "Log type" field:
   - "Cyclic": The tag values are logged in accordance with the set logging cycle.
   - "On change": The tag values are logged, as soon as the operator device detects a change in value.
   - "On demand": The tag values are logged by calling the "LogTag" system function.

6. If you want to log tag values cyclically, select the desired cycle time in the "Logging cycle" area. Alternatively, you can define your own cycle using the object list. The smallest value that can be set is 1 s. All other values are integer multiples of this value.
7. Configure additional parameters for logging in the table of the editor or in the Inspector window.

Result

The configured logging tag is created in the "Historical Data" editor and is also displayed in the tag table.

12.2.6 Displaying tags (Basic Panels)

12.2.6.1 Outputting tag values in screens (Basic Panels)

Introduction

In runtime you can output tag values in the screens of the operator device in the form of a trend. A trend is a graphic representation of the values that a tag takes during runtime. Use the "Trend display" graphic object to represent it. Process values for the trend display are loaded by the PLC from the ongoing process.

The values to be displayed are determined individually within a fixed, configurable cycle. Cyclically-triggered trends are suitable for representing continuous curves, such as the changes in the operating temperature of a motor.

Displayed values

You will need to configure a trend view in a screen so that tag values are displayed on the HMI device. When configuring the trend view, specify which tag values are to be displayed.

You can control the updating of the trend display by defining the cycle time.

12.2.6.2 Configuring trend displays for values from the PLC (Basic Panels)

Introduction

You use a trend view to graphically represent values that a tag assumes during the process.

Requirement

- A screen is open.
- The Inspector window with the trend view properties is open.

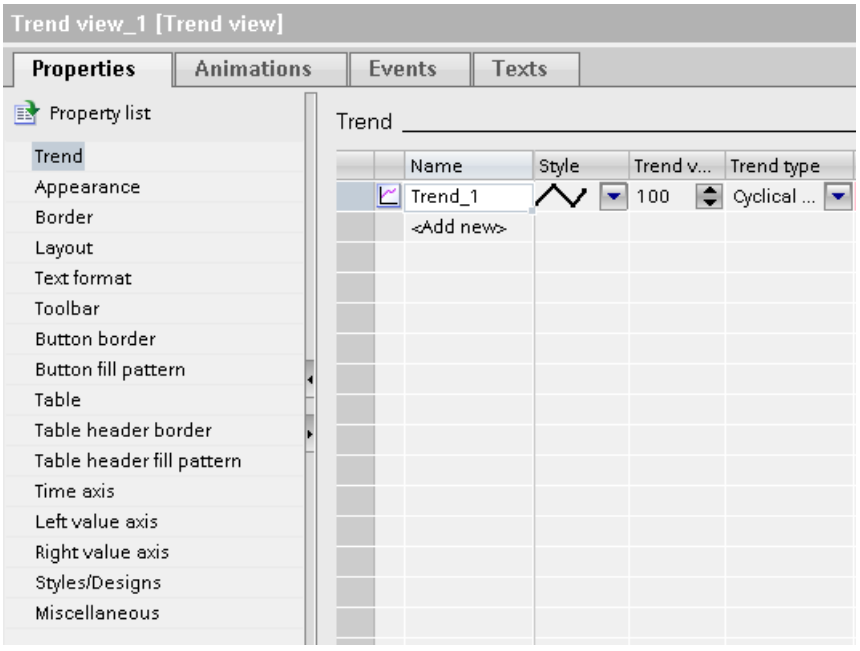
Procedure

To configure a trend view, follow these steps:

1. Add the "Trend view" object from the toolbox in the "Control" group to the screen.

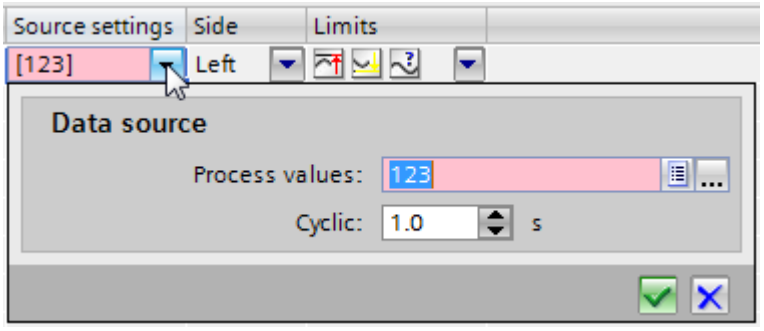


2. Select the "Trend" category from the "Properties" group in the Inspector window and double-click "<Add>" in the "Name" column.



3. Assign a name to the trend in the "Name" column.
4. In the "Style" column, use the selection button to open the "Style" dialog and select the style of the line.
5. Select the number of trend values in the "Trend values" column.

6. In the "Settings" column, use the selection button to open the "Data source" dialog and select the tags to supply the trend with values.
   Specify the cycle for reading the tags from the PLC.



7. You can make other settings in the dialogs of the Inspector window. For example, you can select the "Display table" option in the "Table" category to display a value table beneath the trend view.

Note

If you hold down the <CTRL> key and double-click the trend view, the trend view is activated. You set the column width and the position of the columns in the table header of the values table in active mode. In order to activate the trend view the zoom factor has to be set to 100 %.

Result

In runtime, the values of the selected tags are displayed in the configured trend view.

12.3 Working with alarms

12.3.1 Basics

12.3.1.1 Alarm system in WinCC (Basic Panels)

Introduction

The alarm system allows you to display and record operating states and faults on the HMI device that are present or occur in a plant.

An alarm may have the following content:

No.	Time	Date	Alarm text	Status	Alarm class
5	12:50:24 :590	24.02. 2007	Boiler pressure above high limit.	Incoming Outgoing	Warning: Color Red

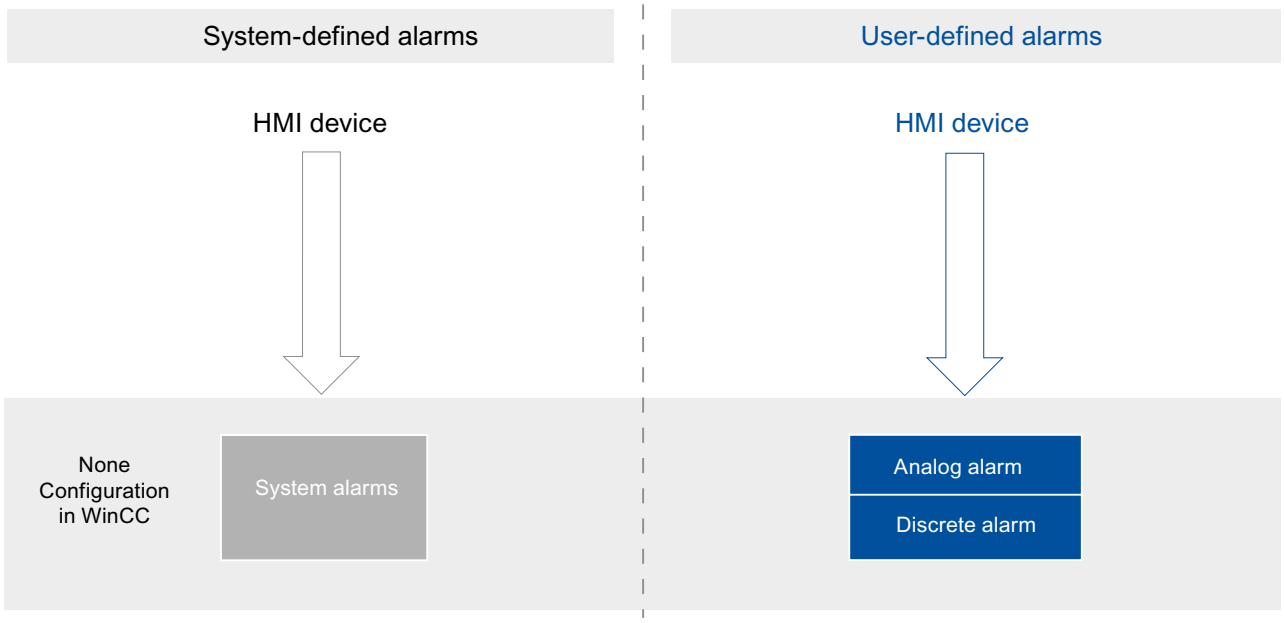
Alarm system in WinCC

The alarm system processes a variety of alarm types. The alarm procedures can be broken down into system-defined alarms and user-defined alarms:

- User-defined alarms are used to monitor the plant process.
- System-defined alarms monitor the HMI device.

The detected alarm events are displayed on the HMI device. Targeted access to the alarms combined with supplementary information about individual alarms ensures that faults are localized and cleared quickly. This reduces stoppages or even prevents them altogether.

The following figure shows the alarm system structure:



12.3.1.2 Alarm types

Overview of the alarm types (Basic Panels)

Introduction

The alarm types serve various purposes for monitoring the plant. The alarms from the individual alarm types are configured and triggered in different ways.

Select the relevant tab in the "HMI alarms" editor to configure alarms based on the individual alarm types.

Alarm types in WinCC

WinCC supports the following alarm types:

User-defined alarms

- **Analog alarms**
  - Analog alarms are used to monitor limit violations.
- **Discrete alarms**
  - Discrete alarms are used to monitor states.

System-defined alarms

- **System events**
  - System events belong to the HMI device and are imported into the project.
  - System events monitor the HMI device.

System-defined alarms

System alarms (Basic Panels)

Examples for alarms

- "An online connection to the PLC is established."

Description

A system alarm indicates the status of the system, plus communication errors between the HMI device and system.

Under "Runtime settings > Alarms" you specify how long a system alarm is shown on the HMI device.

Support

The reference contains a list of the possible system events, along with the cause and possible remedies. If you contact online support because of a system alarm on the HMI device, you will need the alarm number and tags used in the system alarm.

User-defined alarms

Analog alarms (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Analog alarms signal limit violations during the process.

Example

The speed of the mixer in a fruit juice mixing plant must not be too high or too low. You can configure analog alarms to monitor the speed of the mixer. If the high or low limit for the speed of the mixer is violated, an alarm is output on the HMI device containing the following alarm text, for example: "Mixer speed is too low".

Discrete alarms (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Discrete alarms indicate a status in the current process.

Example

A fruit juice mixing plant consists of several tanks containing the ingredients. To ensure the correct mixing ratio of water, fruit concentrate, sugar, and flavoring, the valves in the intakes open and close at the right moment. This operation should be monitored.

You configure a suitable discrete alarm for all the valve states. If a valve on one of the four tanks opens or closes, an alarm is displayed, such as "Water valve closed".

The operator can thus monitor whether the plant is producing correctly.

12.3.1.3 Alarm states (Basic Panels)

Introduction

An alarm assumes various alarm states in Runtime. The user analyzes and reports on the process execution with reference to the alarm states.

Description

Every alarm has an alarm status. The alarm states are made up of the following events:

- Incoming
  The condition for triggering an alarm is satisfied. The alarm is displayed, such as "Boiler pressure too high."
- Outgoing
  The condition for triggering an alarm is no longer satisfied. The alarm is no longer displayed as the boiler was vented.
- Acknowledge
  The operator acknowledges the alarm.

Alarms without acknowledgment

The following table shows the alarm states for alarms that do not have to be acknowledged:

Status	Description
Incoming	The condition for an alarm is satisfied.
Outgoing	The condition for an alarm is no longer satisfied.

Alarms with acknowledgment

The following table shows the alarm states for alarms that have to be acknowledged:

Status	Description
Incoming	The condition for an alarm is satisfied
Outgoing, not acknowledged	The condition for an alarm is no longer satisfied. The operator has not acknowledged the alarm.
Outgoing, subsequently acknowledged	The condition for an alarm is no longer satisfied. The operator has acknowledged the alarm after this time.
Incoming, acknowledged	The condition for an alarm is satisfied. The operator has acknowledged the alarm.
Outgoing, acknowledged first	The condition for an alarm is no longer satisfied. The operator acknowledged the alarm while the condition was still satisfied.

If necessary, the HMI device shows you each time one of these states occurs.

Note

The display text for alarm states depends on the language and configuration.

12.3.1.4 Alarm classes

Basics on alarm classes (Basic Panels)

Introduction

Many alarms occur in a plant. These are all of different importance. You can assign the alarms of your project to alarm classes to clearly show the user which of the alarms are most important.

Description

The alarm class defines how an alarm is displayed. The alarm class specifies if and how the user has to acknowledge alarms of this alarm class.

A new alarm class with mandatory acknowledgment is generated in WinCC.

Note

The choice of display modes for alarm classes depends on the options on your HMI device.

Examples of how to use alarm classes

- The alarm "Speed of fan 1 in upper tolerance range" has alarm class "Warnings". The alarm is displayed with a white background. The alarm does not have to be acknowledged.
- The alarm "Speed of fan 2 has exceeded upper warning range" is assigned to the "Errors" alarm class. The alarm is displayed with a red background and flashes at high frequency in runtime. The alarm is displayed until the user acknowledges it.

Using alarm classes

Use the following alarm classes to define the state machines and display of alarms for your project:

- Predefined alarm classes
  You cannot delete predefined alarm classes and edit them only to a limited extent. Predefined alarm classes have been created for each HMI device under "HMI alarms > Alarm classes".
- Custom alarm classes
  You can create new alarm classes under "HMI alarms > Alarm classes", configure how you want the alarms to be displayed, and define an acknowledgment model for alarms of this alarm class. The possible number of user-defined alarm classes depends on the Runtime used in your project.

Note

Alarm classes from STEP 7

If an HMI device is connected to the controller, alarm classes from STEP 7 will also be displayed in WinCC.

Display of system-defined controller alarms (configured in STEP 7) is not supported by Basic Panels. You can find additional information on the topic of "Controller alarms" and "Alarm logging" under Alarm system in WinCC (Page 5483).

See also

Creating alarm classes (Page 5494)

Alarm system in WinCC (Page 5483)

Predefined alarm classes (Basic Panels)

Predefined alarm classes

The following alarm classes already created in WinCC for every HMI device:

Alarm classes for user-defined alarms

- "Warnings"
  The "Warnings" alarm class is intended to show regular states and routines in the process. Users do not acknowledge alarms from this alarm class.
- "Errors"
  The "Errors" alarm class is intended to show critical or dangerous states or limit violations in the process. The user must acknowledge alarms from this alarm class.

Alarm class for system-defined alarms

- "System"
  The "System" alarm class contains alarms that display states of the HMI device and the PLCs.

See also

Creating alarm classes (Page 5494)

12.3.1.5 Acknowledgment

Acknowledging alarms (Basic Panels)

Introduction

To make sure that an alarm was registered by the plant operator, configure this alarm so that it is displayed until acknowledged by the operator. Alarms that display critical or hazardous states in the process have to be acknowledged.

Description

Acknowledging an alarm changes the alarm status from "Incoming" to "Acknowledged". When the operator acknowledges an alarm, the operator confirms that he or she has processed the status that triggered the alarm.

Triggering acknowledgment of an alarm

In Runtime, you trigger alarm acknowledgments in various ways:

- Acknowledgement by the authorized user at the HMI device
- Automatic acknowledgment by the system without operator action, e.g. by means of
  - Tags
  - PLC
  - System functions in function lists

Acknowledging alarms that belong together

To make the alarm system clearer and easier to use in Runtime, you can configure an alarm group. You can acknowledge all alarms belonging to this alarm group in a single pass.

Acknowledgment by the PLC

Discrete alarms will be automatically acknowledged by the PLC, if necessary. The acknowledgment is triggered by a bit in the "Acknowledgment tag PLC". You define the bit and tag at the configuration stage.

Acknowledgment of an alarm on the HMI device

In Runtime, the user acknowledges an alarm in one of the following ways, depending on the configuration:

- Using the acknowledgment button <ACK> on the HMI device
- Using the button in the alarm view
- Using configured function keys or buttons in screens

Note

Acknowledgment button <ACK> on the HMI device

To ensure that critical alarms are processed only by authorized users, protect the "ACK" button on the HMI devices, including the operating controls and display objects of the alarms. Use the appropriate operator authorization for this.

Note

HMI device dependency

The acknowledgement key <ACK> is not available on all HMI devices.

Acknowledgment model (Basic Panels, Panels, Comfort Panels, RT Advanced)

Overview

You define the state machines for an alarm class. Alarms that are assigned to this alarm class will be acknowledged on the basis of these state machines. The following state machines are used in WinCC:

- Alarm without acknowledgment
  This alarm comes and goes without having to be acknowledged. There is no visible response from the system.
- Alarm with simple acknowledgment
  This alarm must be acknowledged as soon as the event that triggers the alarm occurs. The alarm remains pending until it is acknowledged.

12.3.1.6 Alarm groups (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

Many alarms from different areas and processes occur in a plant. You can compile associated alarms into alarm groups.

Alarm groups

You can use the alarm groups to monitor the parts of the plant and to acknowledge the associated alarms together as required.

Alarm groups can contain alarms from different alarm classes. You only assign alarms that require acknowledgment to alarm groups.

Using alarm groups

It is a good idea to compile alarm groups for alarms such as the following:

- Alarms that are caused by the same fault.
- Alarms of the same type
- Alarms from a machine unit, such as "Fault in drive XY"
- Alarms from an associated part of the process, such as "Fault in cooling water supply"

Display in Runtime

In Runtime, the "Alarm group" column displays the number of the alarm group to which the alarm belongs.

12.3.1.7 Alarm number (Basic Panels, Panels, Comfort Panels, RT Advanced)

Assigning alarm numbers

The system assigns unique alarm numbers within an alarm type.

Note

When adapting alarm numbers, observe the uniqueness of the alarm number within an alarm type.

12.3.2 Working with alarms

12.3.2.1 Alarm components and properties (Basic Panels)

Overview

You configure the components of alarms in WinCC. The following table shows the basic components of alarms:

Alarm class	Alarm number	Time of day	Date	Alarm status	Alarm text	Alarm group	Tooltip	Trigger tag	Limit value
Warning	1	11:09:14	06.08.2007	IO	Maximum speed reached	2	This alarm is ...	speed_1	27
System	110001	11:25:58	06.08.2007	I	Switch to "Online" mode	0	This alarm is ...	PLC-Variable_1	–

Alarm class

Alarm classes, such as "Warnings" or "Errors." The alarm class defines the following for an alarm:

- State machine
- Appearance in Runtime (e.g. color)

Alarm number

An alarm is identified by a unique alarm number. The alarm number is assigned by the system. You can change the alarm number to a sequential alarm number, if necessary, to identify alarms associated in your project.

Time and date

Every alarm has a time stamp that shows the time and date at which the alarm was triggered.

Alarm status

An alarm has the events "Incoming," "Outgoing," "Acknowledge." For each event, a new alarm is output with the current status of the alarm.

Alarm text

The alarm text describes the cause of the alarm.

The alarm text can contain output fields for current values. The values you can insert depend on the Runtime in use. The value is retained at the time at which the alarm status changes.

Alarm group

The alarm group bundles individual alarms.

Tooltip

You can configure a separate tooltip for each alarm; the user can display this tooltip in Runtime.

Trigger tag

Each alarm is assigned a tag as trigger. The alarm is output when this trigger tag meets the defined condition, e.g. when its state changes or it exceeds a limit.

Limit value

Analog alarms indicate limit violations. Depending on the configuration, WinCC outputs the analog alarm as soon as the trigger tag exceeds or undershoots the limit value.

12.3.2.2 Configuring alarms

Overview of alarm configuration tasks (Basic Panels)

Steps to configure alarms

Configuring alarms in WinCC involves the following steps:

1. Edit and create alarm classes
   You use the alarm class to define how an alarm will be displayed in runtime and to define the state machines for it.
2. Creating tags in the "HMI tags" editor
   - Configure the tags for your project.
   - You create range values for the tags.

3. Creating tags in the "HMI alarms " editor
   - Create custom alarms and assign these the tag to be monitored, alarm classes, alarm groups, and other properties.
   - You can also assign system functions or scripts to the alarm events.
4. Output of configured alarms
   To output configured alarms, configure an alarm view or an alarm window in the "Screens" editor.

Additional configuration tasks

Additional tasks may be necessary for configuring alarms, depending on the requirements of your project:

1. Creating alarm groups
   You assign the alarms of your project to alarm groups according to their association, such as by the cause of the problem (power failure) or source of the error (Motor 1).
2. Configuring Loop-In-Alarm
   A Loop-In-Alarm is configured in order to change to a screen containing relevant information on an alarm received.

Creating alarm class

Introduction

Cre
cla
You

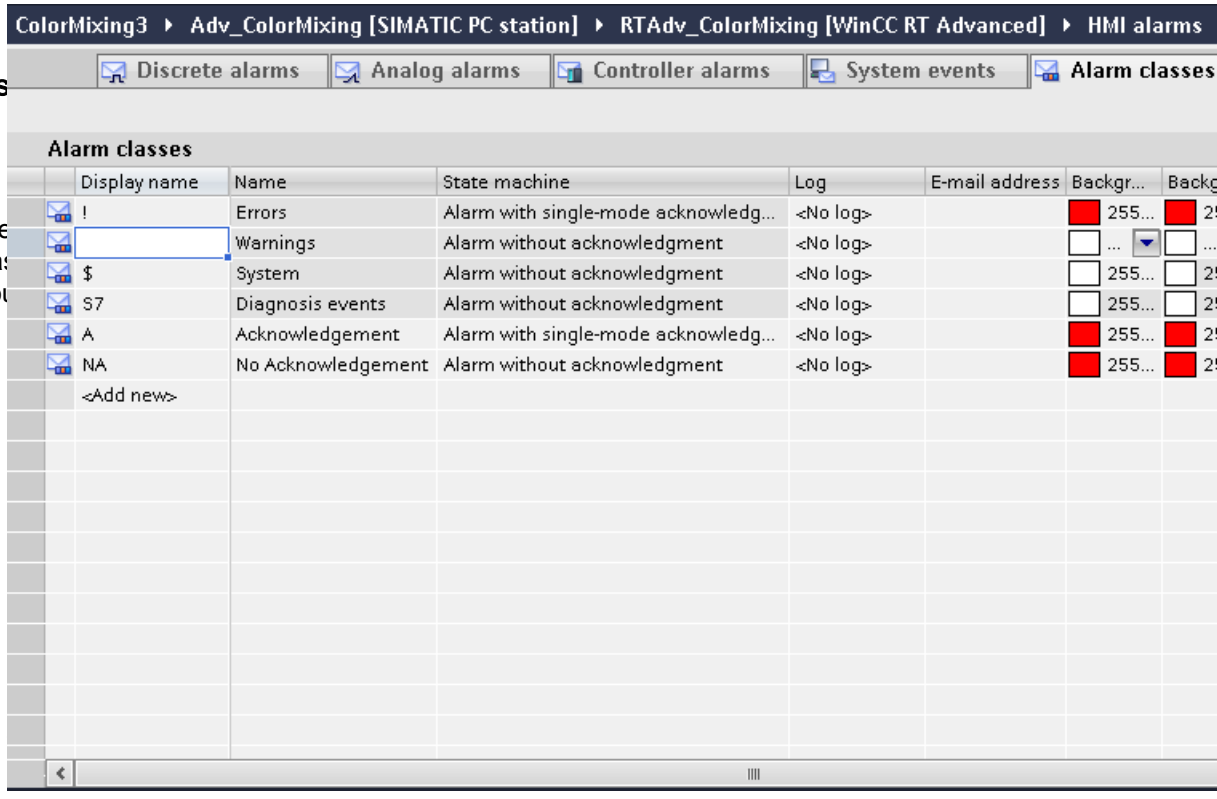
Requirement

-
-

Procedure

To

1.



1. Double-click "<Add>" in the table.
   A new alarm class is created. Each new alarm class is automatically assigned a static ID. The properties of the new alarm class are shown in the Inspector window.
2. Configure the alarm class under "Properties > Properties >General" in the Inspector window.
   - Enter a "Name" and the "Display name".
   - Depending on the HMI device, you can also activate logging, or automatic sending of e-mails.
3. Define the state machines for the alarm class under "Properties > Properties > Acknowledgment" in the Inspector window.
4. Change the default text under "Properties > Properties > Status texts" in the Inspector window.
   This text indicates the status of an alarm in Runtime.
5. Change the default colors under "Properties > Properties > Colors" in the Inspector window. Depending on the HMI device, also change the flashing characteristics.

These settings define how alarms from this alarm class are displayed in Runtime.

Note

To display the alarm classes in color in Runtime, the "Use alarm class colors" option must be activated. In the project navigation, enable "Runtime settings > Alarms > General > Use alarm class colors" accordingly.

Configuring alarm groups (Basic Panels, Panels, Comfort Panels, RT Advanced)

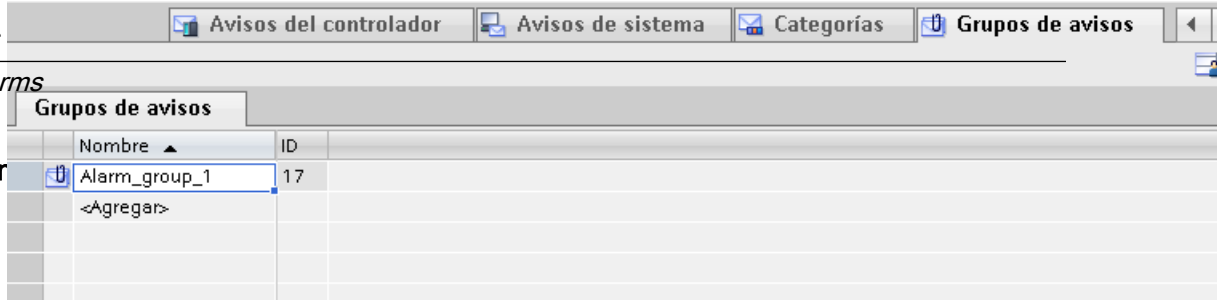
Introduction

Create alarm groups on the "Alarm Groups" tab in the "HMI alarms" editor. An alarm group is a compilation of single alarms. You assign alarms in an alarm group by association, such as cause of the problem or source of the error. If you acknowledge an alarm from this alarm group in Runtime, all other alarms in the alarm group are acknowledged automatically.

Requirement

- You have created a project.
- The "HMI alarms" editor is open.
- The Inspector window is open.

Creating a new alarm

1. 

2. In the work area of the table, double-click "<Add>" in the first free row.
   A new alarm group is created.
3. You can overwrite the proposed "Name".

Result

An alarm group is created. For the group acknowledgment of alarms in Runtime, assign the associated alarms that require acknowledgment to an alarm group.

Configuring discrete alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

Discrete alarms triggered by the PLC indicate status changes in a plant. They indicate the opened or closed state of a valve, for example.

The following sections describes the configuration procedures in the "HMI alarms" editor. You can also configure discrete alarms in the "HMI tags" editor.

Requirements

- The "HMI alarms" editor is open.
- The Inspector window is open.
- You have created the required alarm classes and alarm groups.

Procedure

To configure a discrete alarm, proceed as follows:

1. Open the "Discrete alarms" tab.
2. To create a new discrete alarm, double-click in the work area on "<Add>".
   A new discrete alarm is created.

3. To configure the alarm, select "Properties > Properties >General" in the Inspector window:
   - Enter an alarm text as event text.
     Use the functions of the shortcut menu to format the text on a character-by-character basis, or to insert output fields for HMI tags, or texts from the text lists.
   - You can renumber the alarm.
   - Select the alarm class and the alarm group, if necessary.
4. In the Inspector window, select the tag and the bit that triggers the alarm under "Properties > Properties > Trigger". Note the following information:
   - Use the data types "Int" or "UInt" to select an HMI tag.
   - When you select a PLC tag, use the data types "Int" or "Word". The input and output area of a PLC tag is unsuitable as trigger.
   - Use trigger tag bits only for alarms.
   - Do not use trigger tags for anything else.
   - If you want to acknowledge the alarm via the PLC, use this tag also as PLC acknowledgment tag.

Note

Note the method used to count bits in the utilized PLC when specifying the bit. For more information, refer to the "Communication" section in the PLC Online Help.

Note

If the object does not yet exist in the selection list, create it directly in the object list and change its properties later.

Status-dependent alarm texts

To display a different text independent of the alarm status, link a text list to the alarm text. You control the text list with a tag.

Additional settings for discrete alarms

Creating a tooltip

To configure a tooltip for the alarm, follow these steps:

- Enter your text under "Properties > Properties > Tooltip".

Configuring event-driven tasks

To configure event-driven tasks, such as a loop-in alarm, follow these steps:

1. Select the discrete alarm.
2. Select "Properties > Events" in the Inspector window and configure a new function list for the relevant event.

See also

Configuring loop-in alarm (Page 5503)

Configuring analog alarms (Basic Panels)

Introduction

Analog alarms indicate limit violations. For example, if the speed of a motor drops below a certain value, an analog alarm is triggered.

Requirement

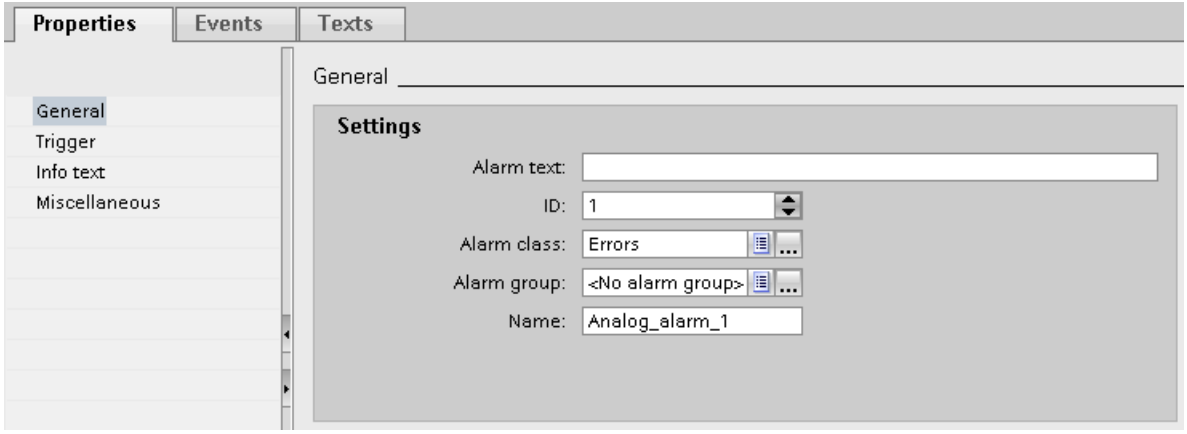
- The "HMI alarms" editor is open.
- The Inspector window is open.
- You have created the required alarm classes and alarm groups.

Procedure

To configure an analog alarm, proceed as follows:

1. Click the "Analog alarms" tab.
2. To create a new analog alarm, double-click in the table on "<Add>".
   A new analog alarm is created.

3. To configure the alarm, select "Properties > Properties >General" in the Inspector window:
   - Enter an alarm text as event text.
     Format the text character-for-character using the shortcut menu.
     Using the shortcut menu, you can insert output fields for HMI tags, or text from text lists.
   - You can renumber the alarm.
   - Select the alarm class and the alarm group, if necessary.



4. Configure the tag that triggers the alarm under "Properties > Properties > Trigger > Settings".
   Do not use trigger tags for anything else.

Configure limit values for an analog alarm

1. In the Inspector window, click the button under "Properties > Properties > Trigger > Limit > Value".
   - To use a constant as limit value, select "Constant".
     Enter the required limit value.
   - To use a tag as limit value, select "HMI tag".
     The button is shown. Use this button to select the tag you want to use.

   Note

   If the tag included in the selection does not yet exist, create it in the object list and change its properties later.

2. Select the mode:
   - "Higher": The alarm is triggered when the limit is exceeded.
   - "Lower": The alarm is triggered when the limit is undershot.

Optional settings for analog alarms

Setting the delay time

To set the delay time, proceed as follows:

- Enter a time period in the Inspector window under "Properties > Properties> Trigger > Settings > Delay".
  The alarm is only triggered when the trigger condition is still present after the delay time has elapsed.

Setting the deadband

Note

If a process value fluctuates around the limit, the alarm associated with this fault may be triggered multiple times. To prevent this from happening, configure a deadband or delay time.

To enter the deadband, follow these steps:

1. Under "Properties > Properties> Trigger > Deadband > Mode", select the change in alarm status for which the deadband is to be taken into account.
2. Enter a constant value under "Value".
3. To define the deadband value as a percentage of the limit, set the "in %" check box.

Creating a tooltip

To configure a tooltip for the alarm, follow these steps:

- Select "Properties > Properties > Tooltip" in the Inspector window and enter your text.

Configuring event-driven tasks

To configure event-driven tasks, such as a loop-in alarm, follow these steps:

1. Select the analog alarm.
2. Select "Properties > Events" in the Inspector window and configure a new function list for the relevant event.

See also

Configuring loop-in alarm (Page 5503)

Adding an output field to alarm text (Basic Panels)

Introduction

In WinCC, you can insert output fields into the alarm text which display the content of tags.

Requirements

- The "HMI alarms" editor is open.
- The alarm is selected.

Output of a tag value in the alarm text

To insert an output field for a tag value in the alarm text, proceed as follows:

1. Place the cursor onto the required position in the event text.
2. Select "Insert tag output field" in the shortcut menu.
3. Open the object list under "Tag" and select a tag.
   You can also create the tag in the object list.
4. Under "Format", specify the length of the output field and the format for tag value output in the alarm text.
   Configure an output field of sufficient size. Otherwise, the tag content is not output to the full extent in the alarm.
5. Click  to save your entries.

WinCC inserts a placeholder for the output field into the alarm text: "<tag: n, [tag name]>" whereby n = text string length.

Editing output field properties

To edit the properties of an output field, proceed as follows:

- Double-click on the output field in the alarm text and edit the settings.

Deleting an output field from the alarm text

To delete an output field from the alarm text, proceed as follows:

- Select the output field in the alarm text and then select the "Delete" command from the shortcut menu.

Note

The sequence of the tag output fields in the alarm text depends on the language. Changing the tag of an output field in one language causes the modified output field to appear at the end of the alarm text in all other languages.

Formatting alarm text (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

The alarm texts can be formatted or special characters inserted in the alarm texts to emphasize specific events or processes.

Requirement

- The "HMI alarms" editor is open.
- An alarm has been created.

Formatting alarm texts

To format an alarm text, proceed as follows:

1. Select the alarm to edit.
2. In the Inspector window, select the characters to format under "Properties > Properties > General > Alarm text".
3. Select the formatting from the shortcut menu, e.g. "Underscored" or "Uppercase".
   The selected characters are displayed in Runtime with the selected formatting.

Note

It is not possible to underline tags and text list entries.

Note

Displaying special characters in alarm texts

When configuring alarm texts, a fixed character set is used in the Engineering System. This character set allows you to use numerous special characters in alarm texts.

Language-dependent fonts are used in runtime to display the texts, for example MS PGothic, SimSun. The fonts used in runtime do not support all special characters. As a result, some special characters are not displayed in runtime.

Removing format settings

To remove all text formats, proceed as follows:

1. In the Inspector window, select the characters whose formatting you want to remove in the alarm text.
2. Select "Delete formatting characters" from the shortcut menu.
   The selected characters are displayed in unformatted notation in Runtime.

Editing system events (Basic, Advanced) (Panels, Comfort Panels, RT Advanced)

Editing system events

When you open the "System events" tab in the "HMI alarms" editor, WinCC prompts you to import or update the system events. Import the system events and specify the display period. You cannot delete or create new system events. You can only edit the alarm text with system events.

Requirements

- You have imported the system events.

Specifying the display period

To determine the display period, proceed as follows:

1. Double-click "Runtime settings" in the project navigation.
2. Enter a value under "Alarms > System events > Display period".
   This value specifies a time in seconds during which the system events are displayed on the HMI device. If you want to display system events permanently, enter "0".

Changing the alarm text

To change the alarm text of a system event, follow these steps:

1. Open the "HMI alarms" editor and then click the "System events" tab.
2. Select the system event that you want to edit.
1. Change the alarm text under "Properties > Properties > Properties > General" in the Inspector window.

Note

If you change an alarm text for a system event that contains a placeholder, leave the number of placeholders unchanged. A placeholder may be a "%1".

Configuring event-driven tasks

The "Incoming" event is available for certain system events, depending on the HMI device. To configure event-driven tasks, proceed as follows:

1. Select the system event.

2. Select "Properties > Properties > Incoming" in the inspector window and configure a function list.

Configuring loop-in alarm (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

A Loop-In-Alarm is configured to change to your screen that contains the relevant information for the error that has occurred.

Note

You configure the screen that contains information about the error that occurred. No standard screens are available that are automatically called depending the selected error alarm.

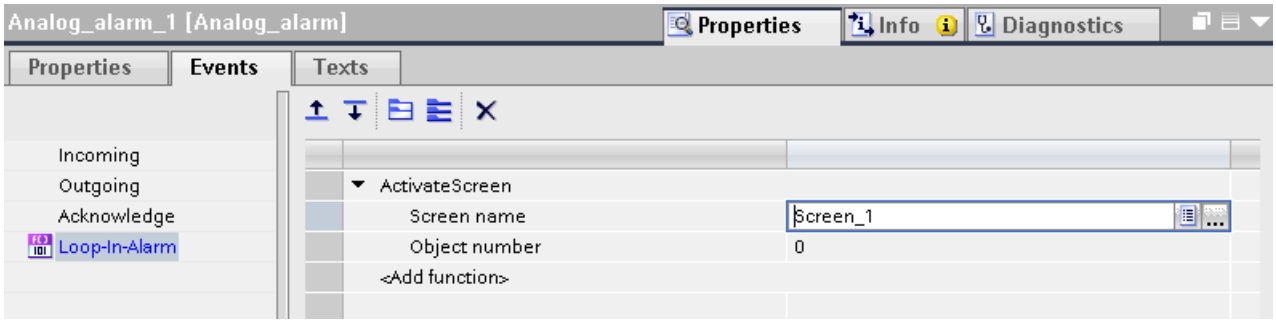
Requirement

- The screen called by the Loop-In-Alarm has been created.
- The "HMI alarms" editor is open.

Procedure

To configure a Loop-In-Alarm for an alarm, proceed as follows:

1. Click the tab that contains the alarm for which you want to configure the Loop-In-Alarm.
2. Select the alarm.
3. In the Inspector window, select "Properties > Events > Loop-In-Alarm".
4. Select the "ActivateScreen" system function.
5. Select the screen to be called by the Loop-In-Alarm as its parameter.



Note

To configure the Loop-In-Alarm for an alarm view with an "alarm line" format, use a button with the following system functions:

- "EditAlarm" for HMI devices with keys
- "AlarmViewEditAlarm" for HMI devices without keys

The system functions trigger the "Loop-In-Alarm" event. The alarm line has no buttons.

Result

If you select an alarm in the alarm view in Runtime and click on the "Loop-In-Alarm" button, a screen is opened with information on the selected alarm.

Even if multiple alarms are selected, exactly one alarm requiring acknowledgment is acknowledged (provided it requires acknowledgment) when the "Loop-In-Alarm" button is pressed for the first time. The system functions stored for the "Loop-In-Alarm" alarm event are always executed, regardless of whether the alarm requires acknowledgment or not.

The next time the "Loop-In-Alarm" button is pressed, only the system functions stored for the "Loop-In-Alarm" alarm event are executed.

Note

If no Loop-In-Alarm is configured for the selected alarm, pressing the "Loop-In-Alarm" button has no effect.

See also

Configuring analog alarms (Page 5498)

Configuring discrete alarms (Page 5496)

Alarms in the "HMI tags" Editor

Configuring discrete alarms in the "HMI tags" editor (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

In WinCC, you can create and edit discrete and analog alarms, including the trigger tags, in the "HMI tags" editor.

Note

If you delete, move or copy objects in the "HMI tags" editor, the changes also take effect in the "HMI alarms" editor.

Requirements

The "HMI tags" editor is open.

Procedure

To configure a discrete alarm, proceed as follows:

1. To create a tag, click on "<Add>" in the table at the top of the work area.
   A new tag is created.
2. Configure an internal or external tag as required.
   - Use the data types "Int" or "UInt" to select an HMI tag.
   - When you select a PLC tag, use the data types "Int" or "Word". The input and output area of a PLC tag is unsuitable as trigger.
3. Select the tag at the top of the work area.

4. Click on "<Add>" in the table on the "Discrete alarms" tab at the bottom of the work area. A new discrete alarm is created for the tag. If you have selected the incorrect data type, the tag will be marked in the discrete alarm.
5. Configure the discrete alarm in the Inspector window:
   - Enter the alarm text under "Properties > Properties > General > Alarm text". You can also insert output fields into the alarm text.
   - Select an alarm class.
   - Select the trigger bit of the tag that triggers the discrete alarm under "Properties > Trigger".
6. You can create additional discrete alarms to monitor the tags.

Note

A tag is monitored using only one alarm type. You should therefore create either analog alarms **or** discrete alarms for a tag.

Result

The configured discrete alarms are created in the "HMI tags" editor and displayed in the "HMI alarms" and "HMI tags" editors.

Configuring analog alarms in the "HMI tags" editor (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

In WinCC, create the discrete and analog alarms, including the trigger tags, in the "HMI tags" editor. You can also edit the alarms as in the "HMI alarms" editor. You can create up to two range values for a tag. You monitor these limits with analog alarms.

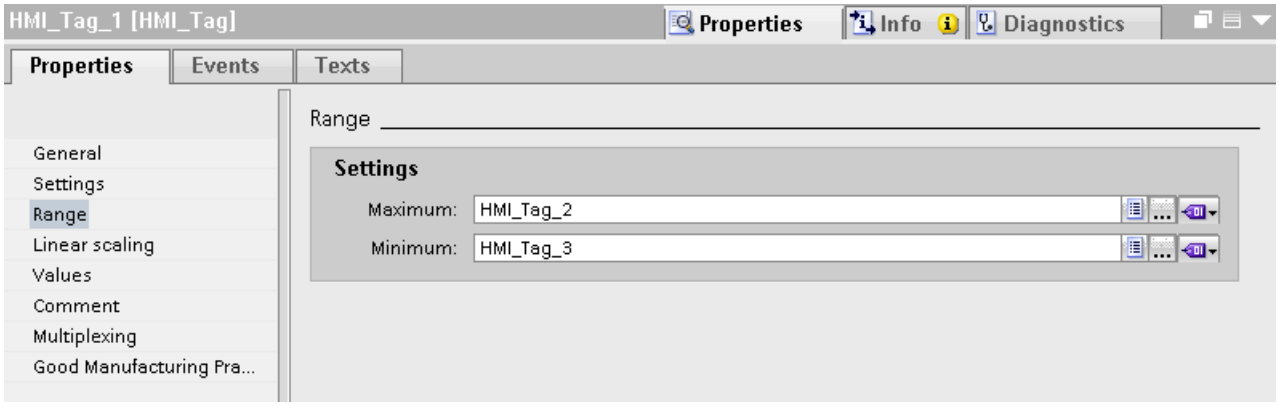
Requirements

The "HMI tags" editor is open.

Procedure

To configure an analog alarm in the "HMI tags" editor, proceed as follows:

1. To create a tag, click on "<Add>" in the table at the top of the work area. A new tag is created.
2. Configure an internal or external tag as required.
3. In the Inspector window, configure the range values of the tags under "Properties > Properties > Range":
   - Select whether to use a "Constant" or an "HMI tag" as limit value for your range values. The object list opens when you select "HMI tag". Select the tag you want to use.



1. Click the "Analog Alarms" tab at the bottom of the work area.
   Create an analog alarm for both range values.
2. Select an analog alarm and configure it in the Inspector window:
   - Enter the alarm text under "Properties > Properties > General > Alarm text".
   - You can also insert output fields into the alarm text.
   - You can change the default alarm class.
3. Configure the analog alarms as in the "HMI alarms" editor.
4. Complete the configuration of all analog alarms.

Note

A tag is monitored using only one alarm type. You should therefore create either analog alarms **or** discrete alarms for a tag.

Result

The configured analog alarms are created in the "HMI tags" editor and displayed in the "HMI alarms" and "HMI tags" editors.

12.3.2.3 Configuring alarm output

Overview of configuring alarm output (Basic Panels)

Steps to complete when configuring alarm output

You configure the alarm output in WinCC in the following steps:

1. Create alarm view
   Use the display and control objects in the "Screens" editor to display alarms in Runtime.
2. Configure acknowledgment
   In the "Screens" editor, you can set the operator action that will trigger the acknowledgment.

Additional configuration tasks

Additional tasks may be necessary for configuring alarm views, depending on the requirements of your project:

1. Setting up authorizations
   To make sure only authorized operators process the alarms, assign authorizations for the alarm view and the function keys of the HMI device.
2. Configuring the filtering of the alarm view
   You configure the filtering of the alarms in Runtime in the "Screens" editor. You can also configure alarm views that only display selected alarms.
3. Configure operator input alarms
   Configure the operator input alarms on the operator controls of the HMI device in the "Screens" editor. A preconfigured operator input alarm is output for an operator action. An operator input is, e.g. the acknowledgment of an alarm.

Displaying alarms (Basic Panels)

Options for displaying alarms on the HMI device

WinCC offers the following options for displaying alarms on the HMI device:

- Alarm view
  The alarm view is configured in a screen. More than one alarm can be displayed simultaneously, depending on the configured size. You can configure multiple alarm views with different contents.
- Alarm window
  The Alarm window is configured in the "Global screen" editor. The alarm window can display multiple alarms at the same time, depending on the configured size. The alarm window is not assigned to any screen. Depending on the configuration, the alarm window is opened when an alarm of a specific alarm class is pending. Based on the configuration, it is not closed until the alarm is acknowledged.

Additional signals

- Alarm indicator
  The alarm indicator is a configurable, graphical icon. When an alarm comes in, the alarm indicator is displayed on the HMI device. You configure the alarm indicator in the "Global screen" editor.
  The alarm indicator has two states:
  - Flashing: At least one alarm that requires acknowledgment is pending.
  - Static: The alarms are acknowledged but at least one of them has not gone out yet. The alarm indicator also displays the number of pending alarms according to the HMI device.
- System functions
  You can configure a list of functions for the event associated with an alarm. These functions must be executed in Runtime when the event occurs.
  Use system functions for alarms in WinCC to control the alarm view or the alarm window other than via the toolbar.

Displaying the predefined alarm classes in Runtime

The following table shows the symbols used to display the predefined alarm classes in the alarm view:

Alarm class	Displayed icon
"Errors"	!
"System"	$
"Warnings"	<No symbol>

Configuring an alarm view (RT Professional)

Introduction

Current alarms are displayed in Runtime in an alarm view or alarm window.

Requirement

- A screen is open in the "Screen" editor.
- The "Tools" task card is open.

Configuring alarms for the alarm view

To specify the alarms that will be shown in the alarm view, proceed as follows:

1. Insert an "Alarm view" object from the "Tools" task card into the screen.
2. Select the alarm view.
   - In the Inspector window, select "Properties > Properties > General > View > Current alarm states".
     Set whether to display alarms with and/or without mandatory acknowledgment.
   - To display all alarms in the alarm buffer, enable "Alarm buffer".
3. In the table, activate the alarm classes to be displayed in the alarm view.

Configuring the layout of the alarm view

To specify how the alarms are shown in the alarm view, proceed as follows.

1. Under "Properties > Properties > Layout > Settings > Lines per alarm" in the Inspection window, specify the number of lines to display for each alarm.
2. In "Properties > Properties > View", select the control elements that are available on the HMI device.
3. Configure the columns under "Properties > Properties > Columns":
   - Under "Visible columns" select the columns to be output in the alarm view.
   - Under "Properties Column", define the properties of the columns.
   - Under "Sort", select the sorting order of the alarms.

Result

Alarms of various alarm classes are output in the alarm view during runtime.

Configuring an alarm window (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

The alarm window displays current alarms. The alarm window is configured in the "Global screen" editor. The alarm window is not assigned to any screen. Depending on the configuration the alarm window is opened when an alarm that belongs to a specific alarm class is active. Depending on the configuration, it is not closed until the alarm is acknowledged. The HMI device can still be used, even if alarms are pending and displayed. An alarm window is displayed and configured like an alarm view.

To hide an alarm window during configuration, create it on its own level.

Requirement

- The "Global Screen" editor is open.
- The "Tools" task card is displayed.
- The Inspector window is open.

Procedure

Proceed as follows to configure an alarm window:

1. Insert an "Alarm window" object from the "Tools" task card into the global screen.
2. Configure the alarm window like an alarm view.
3. Under "Properties > Properties > Mode > Window" in the Inspector window, select how the alarm window reacts and is operated in Runtime.
   - Activate "Modal" if the alarm window is to retain the focus in Runtime after a screen change.
     This option is important, as switching back and forth between the screen and different windows with <Ctrl+TAB> is not supported.

Result

During runtime, the alarms of the selected alarm class are displayed in the alarm window.

Configuring an alarm indicator (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

The alarm indicator uses a warning triangle to indicate that alarms are pending or require acknowledgment. If an alarm of the configured alarm class occurs, the alarm indicator is displayed.

The alarm indicator has two states:

- Flashing: At least one alarm that requires acknowledgment is pending.
- Static: At least one of the acknowledged alarms has not gone out yet.

During configuration, specify whether Runtime has to open an alarm window when you operate the alarm indicator.

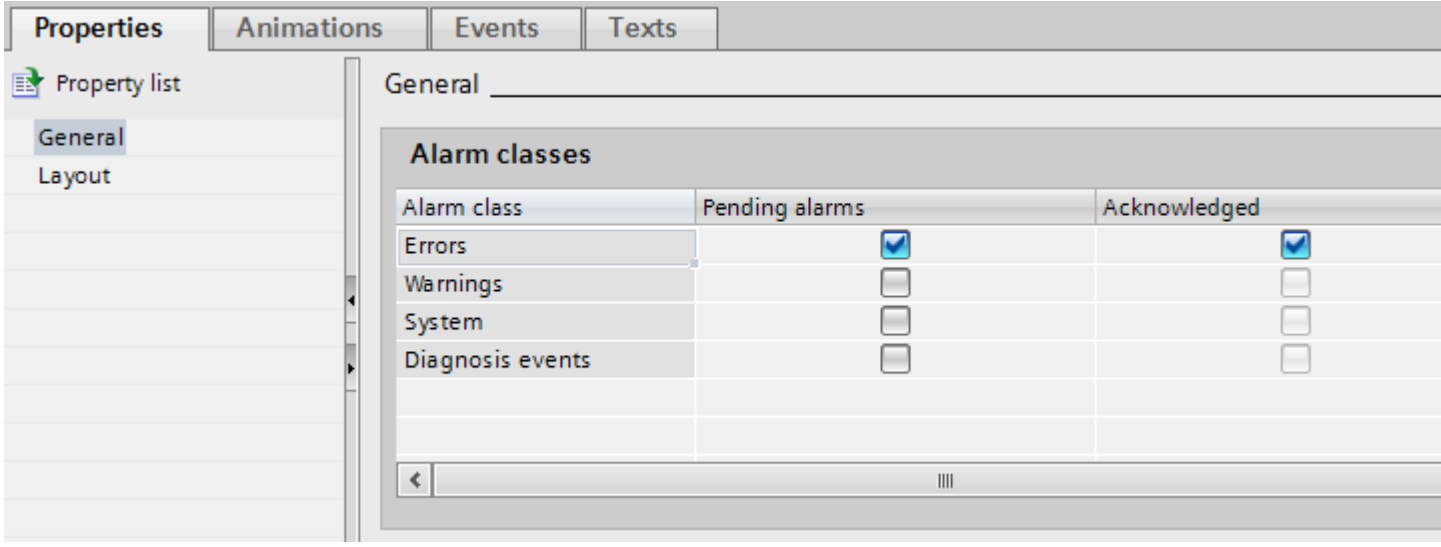
Requirement

- The "Global Screen" editor is open.
- The "Tools" task card is open.
- The Inspector window is open.

Procedure

Proceed as follows to configure the alarm indicator:

1. Insert the "Alarm view" object from the "Tools" task card into the work area.
2. Select the alarm indicator.
3. Under "Properties > Properties > General" in the Inspector window, select the alarm classes to be displayed by the alarm view.
   Specify whether to display pending and/or acknowledged alarms in the alarm view.



4. Under "Properties > Event", assign the system function "ShowAlarmWindow" to an event of the alarm view.

Note

If you have configured a permanent area in the screen or template, do not position the alarm window and alarm view in the vicinity of the permanent area. Otherwise, the alarm window and the alarm view will not be displayed in Runtime. However, the permanent area is not visible in the "Global screen" editor.

Result

The alarm view is displayed if alarms from the selected alarm class are pending or need to be acknowledged in Runtime. The alarm window opens when the user operates the alarm view.

12.3.2.4 Acknowledging alarms

Configuring alarm acknowledgment by means of alarm class (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

To configure an alarm with alarm acknowledgment, assign it to an alarm class with the "Alarm with single acknowledgment" state machine.

Requirement

- The "HMI alarms" editor is open.
- The required alarm class has been created.
- The required alarm has been created.

Selecting the state machines for an alarm class

The state machine for a predefined alarm class has already been set. You can only set the state machines for user-defined alarm classes. Proceed as follows:

1. In the "HMI alarms" editor, click the "Alarm class" tab and select the alarm class.
2. Select the required acknowledgment model under "Properties > Properties > Acknowledgment" in the Inspector window.

Assign alarms to an alarm class requiring acknowledgment

Proceed as follows to assign an alarm to an alarm class requiring acknowledgment.

1. In the "HMI alarms" editor, click the tab for the alarm type and select the alarm.
2. Under "Properties > Properties > General" in the Inspector window, select the alarm class of the alarm.

Result

The alarm will not disappear in Runtime until it is acknowledged by the operator.

Configuring trigger for alarm acknowledgment (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You always specify the acknowledgment requirement for an alarm using the alarm class. Then the operator acknowledges the alarm using the "ACK" function key of the HMI device or the "Acknowledgment" button of the alarm view.

The following options are also available to trigger acknowledgment:

- Configuring a button to acknowledge an alarm
- Acknowledgment of a Discrete Alarm by the PLC

Requirements

- The "HMI alarms" editor is open.
- The required alarm class has been created.
- The required alarm has been created.
- An alarm view and a button are created in the "Screens" editor.

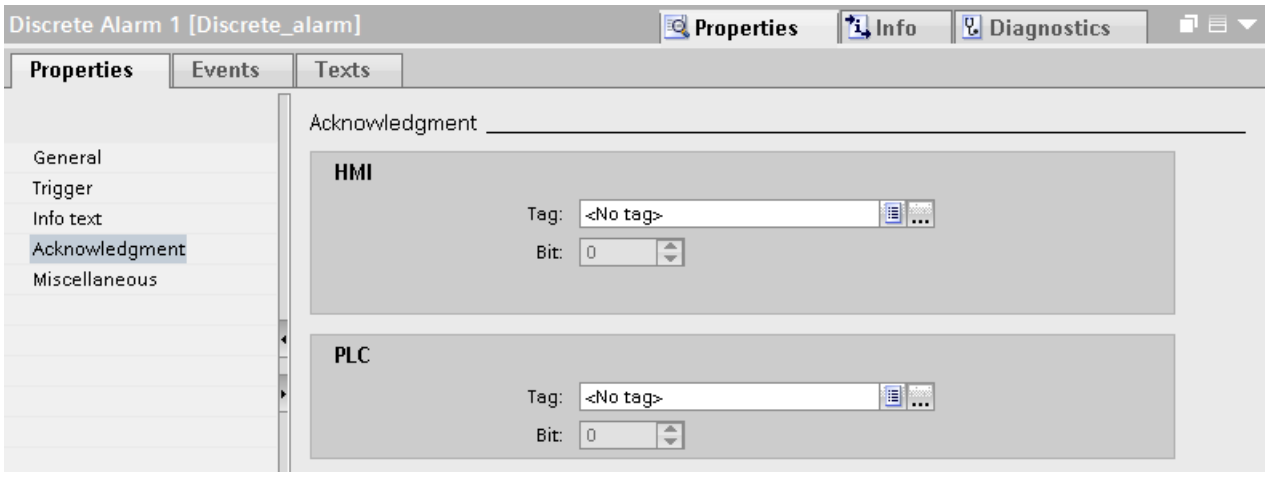
Configuring a button to acknowledge an alarm

To configure a button for acknowledging an alarm, proceed as follows:

1. Select the button in the "Screens" editor.
2. Under "Properties > Events" in the Inspector window, assign the "AlarmViewAcknowledgeAlarm" system function to the "Click" event.
3. Select the alarm view as parameter.

Acknowledgment of a Discrete Alarm by the PLC

1. In the "HMI alarms" editor, click the "Discrete alarm" tab and select the discrete alarm.
2. In the Inspector window, select the tag and the bit that acknowledges the PLC alarm under "Properties > Properties > Acknowledgment > PLC".



Acknowledging alarms in an alarm group

The various acknowledgment options have the following effect in alarm groups:

- Operator acknowledges the alarm
  You acknowledge the alarm of an alarm group, for example, using the <ACK> acknowledgment key or a function key. All alarms of the alarm group are acknowledged.
- The PLC acknowledges the alarm
  The alarm of an alarm group is acknowledged by setting a bit in the tag.
  Only the alarm to be acknowledged is acknowledged.

Sending alarm acknowledgments to the PLC (Basic Panels, Panels, Comfort Panels, RT Advanced)

Requirement

- The "HMI alarms" editor is open.
- The required alarm has been created and assigned to an alarm class requiring acknowledgment.

Note

You cannot send the acknowledgment of analog alarms to the PLC.

Sending alarm acknowledgments to the PLC

To configure that acknowledgment of an alarm is sent to the PLC, follow these steps:

1. In the "HMI alarms" editor, click the "Discrete alarm" tab and select the discrete alarm.
2. In the Inspector window, select "Properties > Properties > Acknowledgment".
3. Under "HMI", select the tag and the bit set by the alarm acknowledgment function.

Note

The HMI device and PLC only have read access to the acknowledgment tag memory area.

Result

If the operator acknowledges the alarm in Runtime, the operating step is forwarded to the PLC.

12.3.3 Logging Alarms

12.3.3.1 Basics on alarm logging

Introduction

An alarm log is used to record project alarms.

Note

Alarm logging is not available for every HMI device.

Configuration steps

To log an alarm, follow these configuration steps:

1. Creating an alarm log
   You define the following properties for the alarm log:
   - Logging method
     Behavior of the log when reaching a specific fill level
   - Storage location and file format
   - Log size
   - Runtime start characteristics
2. Assigning an alarm log to an alarm class
   You can log the alarms from multiple alarm classes in an alarm log.
3. Assigning an alarm to a loggable alarm class
4. Configure display of logged alarms in an alarm view

Content of the alarm log

All states are logged for configured alarms. The following three entries, for example, are stored in the log for an alarm that requires acknowledgment:

- 04.08.2007 10:00:25:520, analog alarm, ID5, **K**, error, fill level exceeded by 10%
- 04.08.2007 10:01:20:442, analog alarm, ID5, **Q**, error, fill level exceeded by 10%
- 04.08.2007 10:01:30:112, analog alarm, ID5, **G**, error, fill level exceeded by 10%

In the example, the alarm states are identified by the following letters:

K = Incoming

Q = Acknowledged

G = Outgoing

All data belonging to an alarm is stored in the alarm log, including configuration data such as the alarm class, time stamp, and alarm text.

The potential number of logged alarms depends on the data medium used. You can further process the logged alarms in other programs for analysis purposes, for example.

Note

Alarm text and point of error are only logged if you have configured such a setting in the log properties.

Logged alarms that contain alarm text and the point of error exceed the estimated size of the configured alarms. Check to see whether the specified storage location still has sufficient space.

Note

The time stamp of a logged alarm is always specified in standard UTC format (Universal Time Coordinated).

Logging methods

The logging method determines how the alarm log responds when the configured size is reached. WinCC supports the following logging methods:

- Circular log
  When the configured log size has been reached, the oldest entries are deleted. When the configured log size has been reached, approximately 20% of the oldest entries are deleted. It is therefore not possible to display all the configured entries. During configuration, select an appropriate size for the circulation log. Alternatively, configure a segmented circular log.
- Segmented circular log
  In a segmented circular log, multiple single logs of the same size are filled in succession. When all logs are completely full, the oldest log is overwritten.

- Log with level-dependent system alarm
  When a defined level is reached, such as 90 %, a system alarm is triggered. When the configured size of the log has been reached, new alarm events are not logged.
- Log with level-dependent execution of system functions
  When the log is completely full, the "Overflow" event is triggered. You configure a function list for the event.
  When the configured size of the log has been reached, new alarm events are not logged.

Displaying logged data

On the HMI device, the logged data is displayed in an alarm view that is configured for this purpose.

See also

Storage locations of logs (Page 5563)

12.3.3.2 Creating an alarm log

Introduction

In Runtime, you save alarms to the alarm logs. Specify the alarm log in the alarm class. An alarm log contains the alarms of several alarm classes. Create the alarm log in the "Logs" editor. When creating an alarm log, define the following parameters:

- Name
- Size
- Storage location
- Runtime start characteristics
- Log type
- Advanced content
- Checksum

You can also enter comments for each log.

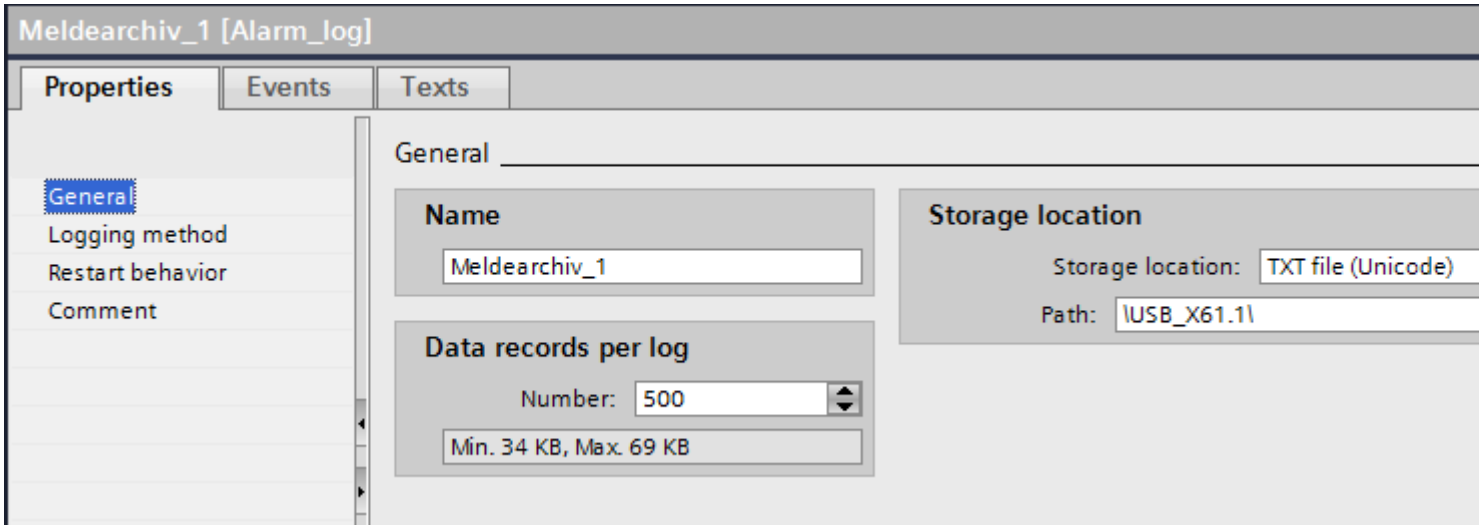
Requirement

- The "Alarm log" tab is open in the "Log" editor.
- The Inspector window is open.

Procedure

To create an alarm log, proceed as follows:

1. Double-click "<Add>" in the table.
   A new alarm log is created.
2. Under "Properties > Properties > General" in the Inspector window, enter a unique name for the alarm log.



3. Under "Number of data records per log", define the number of alarms to be saved to a log file.
   The approximate space required on the storage medium is displayed. Memory space requirements will increase accordingly if you log alarm texts with tag values.
4. In the "Storage location" field you select where the log entries are saved.
5. Depending on the selected "Storage location", you either select the "Path" or the "Name of the data source".
6. To determine if audit trail data were changed at a later time, activate "Properties > Properties > General > Checksum".

   Note

   Device dependency

   The "Checksum" option is only available for display and HMI devices which support "Configuration conforms to GMP".

7. If necessary, enter a descriptive text in the "Comment" category to document your configuration.

Result

The alarm log is created. You can assign one or several alarm classes to this alarm log.

See also

Basics on alarm logging (Page 5515)

12.3.3.3 Logging alarms

Overview

To log alarms, follow these steps:

- Create an alarm log.
- Assign the created alarm log to an alarm class.
- Assign the alarm class with the created alarm log to an alarm.
- In the Runtime settings specify the language in which you want to write the logs.
- You evaluate logged alarms.
  You can analyze the logged alarms directly in your WinCC project, such as in an alarm view, or in another user program, such as Microsoft Excel.

Note

Tag fields in the alarm text

The sequence of tag fields in the alarm text is language dependent. The sequence of the Runtime language is used for logging alarms to a "*.csv" log file.

Changing the tag of an output field in one language causes the modified output field to appear at the end of the alarm text in all other languages. This means the sequence of the output fields in the log file can change.

Requirements

- You have created an alarm log.
- The "HMI alarms" editor is open.

Assigning an alarm log to an alarm class

To assign the alarm log to an alarm class, proceed as follows:

1. Open the "Alarm classes" tab in the "HMI alarms" editor.
2. Select the required alarm class.
3. Under "Properties > Properties > General > Log" in the Inspector window, select the alarm log.

Assign alarm to an alarm class

To assign an alarm to the alarm class, proceed as follows:

1. In the "HMI alarms" editor, open the "Analog alarms" tab, or the "Discrete alarms" tab.
2. Select the required alarm.
3. Under "Properties > Properties > General > Alarm class" in the Inspection window, select the alarm class for which the alarm log was configured.

Result

The alarm is saved to the configured alarm log.

See also

Basics on alarm logging (Page 5515)

12.3.3.4 Configuring an alarm view for logged alarms (Advanced)

Introduction

In Runtime, logged alarms are displayed in an alarm view or alarm window.

Requirements

- An alarm view or alarm window is configured in the "Screens" editor
- An alarm log was created in the "Log" editor.
- The alarms are assigned the "loggable" attribute in the "HMI alarms" editor.

Configuring an alarm view for logged alarms

To configure an alarm view for logged alarms, proceed as follows:

1. Open the screen with the alarm view and select the alarm view.
2. In the Inspector window, select "Properties > Properties > General > Alarm log".
3. Click the "...." button and select the alarm log.
4. Continue to configure the alarm view as for the display of current alarms.

Result

In Runtime, the logged alarms are output in the alarm view.

See also

Basics on alarm logging (Page 5515)

12.3.3.5 Configuring alarm buffer overflow (Advanced) (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

The size of the alarm buffer depends on the type of HMI device. In WinCC, specify the percentage of the alarm buffer to be deleted on alarm buffer overflow.

Requirements

- You have created a project.

Procedure

To configure the response to an alarm buffer overflow, follow these steps:

1. Double-click "Runtime settings" in the project navigation under your HMI device.
2. Under "Alarms > General > Buffer clearance in percent upon buffer overflow", enter a value between 1 and 100.
   This value specifies the percentage of the alarm buffer that is deleted on alarm buffer overflow.

Result

If data in the alarm buffer exceeds the resources of alarm buffer memory, the configured percentage of data will be deleted from the oldest alarms in the alarm buffer.

Note

You cannot check the alarm buffer overflow separately by alarm method. You can use the "ClearAlarmBuffer" system function to delete specific alarms of specific alarm classes from the alarm buffer.

12.3.3.6 Log response to language switching in runtime

Introduction

In the Runtime settings of your HMI device, select the language to be used for writing to logs in runtime.

Requirement

- The languages used in your project are activated in the "Project languages" editor, for example, "German (Germany)" and "English (USA)" .

Procedure

To determine the startup language, follow these steps:

1. Select "Runtime settings > Language and fonts" in the project navigation.
2. Activate the runtime language, for example, "German (Germany)" and "English (USA)".
3. Set the "Language switch order". Use 0 to determine the startup language, for example:
   - German 0.
   - English 1
     With "0", German is specified as the "Startup language".
4. Select "Runtime settings > General" in the project navigation.
5. Select the "Logs > Logging language > Startup language".

Result

The project starts after it is transferred. German is specified as the "Startup language" with "Language switch order". The logs are now written in German. During runtime, the operator switches the runtime language to English. The logs will still to be written in German.

The operator closes runtime. Due to the previously performed language switch, the next time the system starts the "startup language" is English. Since English is the startup language, the logs are now written in English.

The logging language remains English even when the language is switched again in runtime until runtime is closed once again.

If you use another option instead of "Startup language" to select the language, the logs are always written in the same language. This is true regardless of the language selected by the operator in runtime.

See also

Basics on alarm logging (Page 5515)

12.3.3.7 Managing logging behavior when Runtime starts

Introduction

When you configure a log, you define the restart characteristics of the log when Runtime is started. You define whether the logging should start when Runtime starts in the log properties. You can also define whether an existing log will be continued or overwritten.

You define the restart characteristics separately for each log.

Requirement

- You have created a log.
- The "Historical Data" editor is open.
- The Inspector window with the log properties is open.

Procedure

To configure the restart characteristics of a data log, proceed as follows:

1. Select the log for which you want to define the restart characteristics in the "Historical Data" editor.
2. In the Inspector window select "Properties > Properties > Restart behavior".
3. If you want logging to start when Runtime starts, enable the "Enable logging at runtime start" option in the "Logging" area.
   You can also start logging in Runtime using the "StartLogging" system function, for example.
4. Select the restart behavior of the log in the "Log handling at restart" area.
   - The logged values are deleted and logging is started again with the option "Reset log".
   - The option "Append data to existing log" is used to append the values to be logged to the existing log.

Alternatively you can configure the restart characteristics of a log directly in the "Historical Data" editor. To view hidden columns, activate the column titles using the shortcut menu.

Result

Logging will start in runtime according to your settings.

See also

Basics on alarm logging (Page 5515)

12.3.3.8 Controlling the Logging in relation to the Fill Level

Introduction

The size of a log is determined by the number of entries. You use the logging method to determine how the log responds when it is full.

Logging methods

The following logging methods are available:

-  Circular log
  When the configured log size has been reached, the oldest entries are deleted. When the configured log size has been reached, approximately 20% of the oldest entries are deleted. It is therefore not possible to display all the configured entries. During configuration, select an appropriate size for the circulation log. Alternatively, configure a segmented circular log.
-  Segmented circular log
  In a segmented circular log, multiple log segments of the same size are filled in succession. When all logs are completely full, the oldest log is overwritten.
-  Log that sends a system alarm when it is full
  When a defined level is reached, such as 90 %, a system alarm is triggered. When the log is 100% full, new tag values are not logged.
-  Log with level-dependent triggering of an event.
  When the log is completely full, the "Overflow" event is triggered. Configure a function list for the event that will be carried out when the "Overflow" event occurs. When the configured size of the log is reached, new tag values are not logged.
  The following system functions are available for further processing of full logs:

Requirement

- You have created a log.
- The "Historical Data" editor is open.
- The Inspector window with the log properties is open.

Procedure

1. Select the log for which you want to define the logging method in the "Historical Data" editor.
2. Select "Properties > Properties > Logging method" in the Inspector window and select the required logging method.
3. If you have selected the "Segmented circular log" type, enter the number of log segments.
   If you selected a log with the "Display system alarm on" setting, specify the level as a percentage at which a system alarm is to be triggered.
   If you selected the "Trigger event" setting, configure the function list in the "Events" group.

Alternatively you can configure the logging method directly in the "Historical Data" editor table. To view hidden columns, activate the column titles using the shortcut menu.

The "Overflow" event is not available in the editor table. You must therefore configure the function list in the Inspector window.

Result

The selected log responds according to the settings in Runtime.

See also

Basics on alarm logging (Page 5515)

12.3.4 Operating alarms in Runtime

12.3.4.1 Alarms in Runtime (Basic Panels)

Alarms

Alarms indicate events and states on the HMI device which have occurred in the system, in the process or on the HMI device itself. A status is reported when it is received.

An alarm could trigger one of the following alarm events:

- Incoming
- Outgoing
- Acknowledge
- Loop-in-alarm

The configuration engineer defines which alarms must be acknowledged by the user.

An alarm may contain the following information:

- Date
- Time
- Alarm text
- Location of fault
- Status
- Alarm class
- Alarm number
- Alarm group

Alarm classes

Alarms are assigned to various alarm classes.

- "Warnings"
  Alarms of this class usually indicate states of a plant such as "Motor switched on". Alarms in this class do not require acknowledgement.
- "Errors"
  Alarms in this class must always be acknowledged. Error alarms normally indicate critical errors within the plant such as "Motor temperature too high".

- "System"
  System alarms indicate states or events which occur on the HMI device.
  System alarms provide information on occurrences such as operator errors or communication faults.
- Custom alarm classes
  The properties of this alarm class must be defined in the configuration.

Alarm buffer

Alarm events are saved to an internal buffer. The size of this alarm buffer depends on the HMI device type.

Alarm view

The alarm view shows selected alarms or alarm events from the alarm buffer. Whether alarm events have to be acknowledged or not is specified in your configuration.

Alarm window

The alarm window is not assigned to any screen. Depending on the configuration the alarm window is opened when an alarm that belongs to a specific alarm class is active.

You can configure the order in which the alarms are displayed. You can choose to display the alarms in ascending or descending order of their occurrence. The alarm window can also be set to indicate the exact location of the fault, including the date and time of the alarm event. By means of configuration, the display can be filtered in such a way that only alarms that contain a specific character string will be shown.

Alarm indicator

The alarm indicator is a graphic symbol that is displayed on the screen when an alarm of the specified alarm class is activated.

The alarm indicator can have one of two states:

- Flashing: At least one unacknowledged alarm is pending.
- Static: The alarms are acknowledged but at least one of them is not yet deactivated. The number indicates the number of queued alarms.

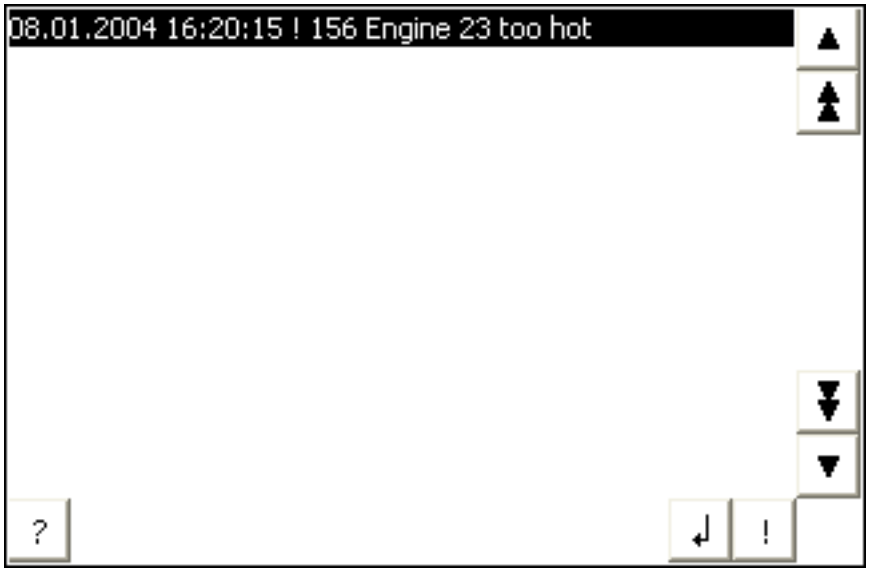
12.3.4.2 Simple alarm view, simple alarm window in runtime (Basic Panels)

Application

The simple alarm view shows selected alarms or alarm events from the alarm buffer. The layout and operation of the simple alarm window correspond to that of the simple alarm view.

Note

In the Engineering System, for example, dynamize the visibility of an object in the "Animations" tab of the Inspector window. In Runtime, the "Simple alarm view" does not support animations. If you configured an animation and, for example, run a consistency check on the project, an error alarm is displayed in the output window.



Layout

Depending on the configuration, in the alarm view different columns with information regarding an alarm or an alarm event are displayed.

To differentiate between the different alarm classes, the first column in the alarm view contains an icon:

Icon	Alarm class
!	"Errors"
empty	"Warnings"
depends on the configuration	Custom alarm classes
$	"System"

Operation

You use the alarm view as follows, depending on how it is configured:

- Acknowledging alarms
- Editing alarms

Control elements

The buttons have the following functions:

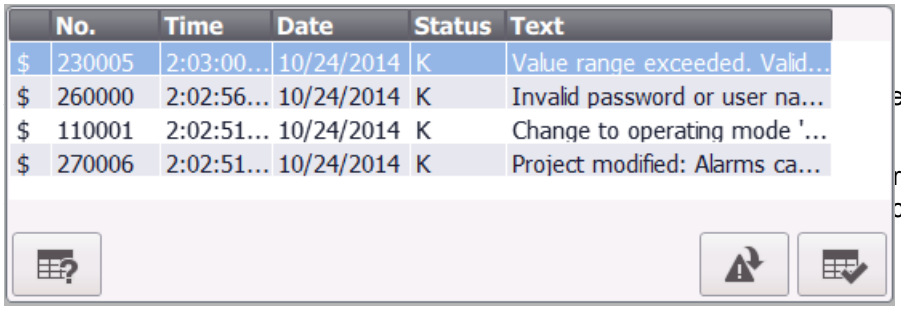
Button	Function
!	Acknowledge alarm
↲	Loop-In-Alarm Changes to the screen that contains information about the error event
?	Displaying a tooltip for an alarm
▶	Displays the full text of the selected alarm in a separate window, namely the alarm text window In the alarm text window, you can view alarm texts that exceed the space available in the Alarm view. Close the alarm text window with the ✕ button.
▲	Scrolls one alarm up.
▲▲	Scrolls one page up in the alarm view.
▼▼	Scrolls one page down in the alarm view.
▼	Scrolls one alarm down.

Format of the control elements

The display of the buttons for using the simple alarm view depends on the configured size. You should therefore check on the HMI device whether all the required buttons are available.

12.3.4.3 Alarm window, alarm view in Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced)

Application



evice. The layout

ration the alarm

ctive. Depending

Layout

Depending on the configuration, in the alarm view different columns with information regarding an alarm or an alarm event are displayed. If a filter is configured, only alarms that contain a specific string in the alarm text will be displayed.

To differentiate between the different alarm classes, the first column in the alarm view contains an icon:

Symbol	Alarm class
!	"Errors"
Empty	"Warnings"
depends on the configuration	Custom alarm classes
$	"System"
S7	"Diagnosis Event"
!!	"Safety Warnings"

Operation

You use the alarm view as follows, depending on how it is configured:

- Change the order of the columns.
- Change the order in which the alarms are displayed.
- Acknowledging alarms
- Editing alarms

Operator controls

The buttons have the following functions:

Button	Function
	Displaying a tooltip for an alarm
	Loop-In-Alarm Executes the configured function (for example: "ActivateScreen") with each key-stroke. It is hereby possible to call a screen containing your information about the selected alarm. Also acknowledges an alarm requiring acknowledgment. [1)]
	Acknowledge alarm

[1)] Only if a function is configured for the selected alarm for the "loop-in-alarm" event.

Operation behavior

Linked alarm window for touch panels

When configuring the alarm window for keyboard units, enable the "Connected" (modal) property under "Properties > Mode". This ensures that the alarm window does not defocus during screen changes. Switching back and forth between the screen and different windows with <Ctrl+TAB> is not supported. If the linked alarm window has the focus, then the buttons in the screen behind it cannot be operated. The functions configured on a function key are carried out.

Changing the order of the displayed alarms

When you click on the column, first of all alarms requiring acknowledgment are sorted according to date and time. Then those alarms that do not require acknowledgment are sorted according to date and time.

12.3.4.4 Alarm indicator in Runtime (Basic Panels)

Application

The alarm indicator is displayed if alarms of the specified alarm class are pending or require acknowledgment.



Layout

The alarm indicator can have one of two states:

- Flashing: At least one unacknowledged alarm is pending.
- Static: The alarms are acknowledged but at least one of them has not gone out yet. The displayed number indicates the number of queued alarms.

Operation

Depending on the configuration, when operating the alarm indicator an alarm window is opened. The alarm indicator can only be operated with the touch screen.

12.3.4.5 Acknowledging alarms (Basic Panels)

Introduction

You can acknowledge alarms in Runtime according to your project configuration settings. You can acknowledge alarms as follows:

- Using the display and control object buttons
- Using the "ACK" key on your HMI device
- Using individually-configured function keys or buttons

If an operator authorization is configured for an individual control, the alarms can only be acknowledged by authorized users.

To automatically acknowledge alarms in Runtime, use the system functions and the option "Acknowledgment by the PLC".

Acknowledgment variants

You acknowledge individual alarms or multiple alarms together in Runtime. They are distinguished as follows:

- Single acknowledgment
  Acknowledgment of an alarm using a button or a function key.
- Acknowledge alarm groups
  Acknowledgment of all the alarms of an alarm group using a button or a function key.

Requirement

- An alarm is displayed on the HMI device.

Procedure

To acknowledge an alarm, proceed as follows:

1. Select the alarm.
2. Click on the ! button.

Result

The alarm status changes to "Acknowledged". If the condition for triggering an alarm no longer applies, the alarm status also changes to "Outgoing", and it is no longer displayed on the HMI device.

12.3.5 Reference

12.3.5.1 System functions for alarms (Basic Panels)

System functions

System functions are predefined functions you can use to implement many tasks in runtime, even with no programming knowledge. You use system functions in a function list.

The table shows all the system functions available for displaying and editing alarms.

System function	Effect
EditAlarm	Triggers the "Loop-In-Alarm" event for the selected alarm or, with multiple selection, the last selected alarm.
ClearAlarmBuffer	Deletes alarms from the alarm buffer on the HMI device.
ClearAlarmBufferProtoolLegacy	Function such as "ClearAlarmBuffer". This system function has been retained to ensure compatibility and uses the old ProTool numbering.
AlarmViewEditAlarm	Triggers the event Loop-In-Alarm for all alarms selected in the specified alarm view.
AlarmViewAcknowledgeAlarm	Acknowledges the alarms that are selected in the specified alarm view.
AlarmViewShowOperatorNotes	Displays the configured tooltip for the alarm selected in the specified alarm view.
AcknowledgeAlarm	Acknowledges all selected alarms.
ShowAlarmWindow	Hides or shows the alarm window on the HMI device.

12.3.5.2 System events

Basics on system events (Basic Panels, Panels, Comfort Panels, RT Advanced)

System events

System events on the HMI device provide information about internal states of the HMI device and PLC.

The following overview illustrates when a system event occurs and how to eliminate the cause of error.

Note

HMI device dependency

Some of the system events described in this section apply to the individual HMI devices based on their scope of functions.

Note

System events are output in an alarm view. System events are output in the language currently set on your HMI device.

The time format (AM/PM or 24-hour format) is based on the selected language. If there is no translation of the alarm texts in this language, English or Russian is selected as a replacement language and displays the appropriate time format.

System event parameters

System events may contain encrypted parameters. The parameters are of relevance when troubleshooting because they provide a reference to the source code of the Runtime software. These parameters are output after the "Error code:"

9999 - Global alarm (Panels, Comfort Panels, RT Advanced)

Meaning of the system event

The most important system events are listed below.

Table 12-2 9999 - Global alarm

Number	Effect/cause	Remedy
9999	The system start was interrupted. Check the following: • Line voltage • Cable connections • System settings • Configurations • Conversions	Check the power supply and cable connections. Check the system settings. Check the configurations and conversions. If the error occurs repeatedly, contact the hotline!

30000 - Alarms errors when using system functions (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below. The system events are divided into different ranges.

Table 12-3 30000 - Alarms errors when using system functions

Number	Effect/causes	Remedy
30010	The tag could not accept the function result, e.g. when it has exceeded the value range.	Check the tag types of the system function parameters.
30011	A system function could not be executed because the function was assigned an invalid value or type in the parameter.	Check the parameter value and tag type of the invalid parameter. If a tag is used as a parameter, check its value.
30012	A system function could not be executed because the function was assigned an invalid value or type in the parameter.	Check the parameter value and tag type of the invalid parameter. If a tag is used as a parameter, check its value.

40000 - Linear scaling alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below. The system events are divided into different ranges.

Table 12-4 40000 - Linear scaling alarms

Number	Effect/causes	Remedy
40010	The system function could not be executed since the parameters could not be converted to a common tag type.	Check the parameter types in the configuration.
40011	The system function could not be executed since the parameters could not be converted to a common tag type.	Check the parameter types in the configuration.

50000 - Data server alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below. The system events are divided into different ranges.

Table 12-5 50000 - Data server alarms

Number	Effect/causes	Remedy
50000	The HMI device is receiving data faster than it is capable of processing. Therefore, no further data is accepted until all current data have been processed. Data exchange then resumes.	--
50001	Data exchange has been resumed.	--

70000 - Win32 function alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below.

Table 12-6 70000 - Win32 function alarms

Number	Effect/causes	Remedy
70010	The application could not be started because it could not be found in the path specified or there is insufficient memory space.	Check whether the application exists in the specified path or close other applications.
70011	The system time could not be modified. The error message only appears in connection with area pointer "Date/time PLC". Possible causes: • An invalid time was transferred in the job mailbox. • The Windows user has no right to modify the system time. If the first parameter in the system event is displayed with the value 13, the second parameter indicates the byte containing the incorrect value.	Check the time that is to be set. Under Windows: Users running WinCC Runtime must be granted the right to set the system time of the operating system.
70012	Error when executing the system function "StopRuntime" with the "Runtime and operating system" option. Windows and WinCC Runtime are not closed. The error may have been generated because other programs cannot be closed.	Close all programs currently running. Now close Windows.
70013	The system time could not be modified because an invalid value was entered. An incorrect delimiter may have been used.	Check the time which is to be set.

Number	Effect/causes	Remedy
70014	The system time could not be modified. Possible causes: • An invalid time was transferred. • The Windows user has no right to modify the system time. Windows rejects the setting request.	Check the time that is to be set. Under Windows: Users running WinCC Runtime must be granted the right to set the system time of the operating system.
70015	The system time could not be read because Windows rejects the reading function.	--
70016	An attempt was made to select a screen by means of a system function or job. This is not possible because the configured screen number does not exist. Or: A screen could not be generated due to insufficient system memory. Or: The screen is blocked. Or: Screen call has not been executed correctly.	Check the screen number in the system function or job against the screen numbers configured. Assign the number to a screen if necessary. Check the details for the screen call and whether the screen is blocked for specific users.
70017	Date/time is not read from the area pointer because the address set in the PLC is either not available or has not been set up.	Change the address or set up the address in the PLC.
70018	Acknowledgment that the password list has been successfully imported.	--
70019	Acknowledgment that the password list has been successfully exported.	--
70020	Acknowledgment for activation of alarm reporting.	--
70021	Acknowledgment for deactivation of alarm reporting.	--
70022	Acknowledgment to starting the Import Password List action.	--
70023	Acknowledgment to starting the Export Password List action.	--
70024	The tag value range was exceeded during system function execution. No calculation of the system function.	Check and correct the calculation.
70025	The tag value range was exceeded during system function execution. No calculation of the system function.	Check and correct the calculation.
70026	No other screens are stored in the internal screen memory. No other screens can be selected.	--
70027	The backup of the RAM file system has been started.	--
70028	RAM file system backup is complete. The files from the RAM have been copied to the Flash memory. Following a restart, these saved files are copied back to the RAM file system.	--
70029	Backup of the RAM file system has failed. No backup copy of the RAM file system has been made.	Check the settings in the "Control Panel > OP" dialog and save the RAM file system using the "Save Files" button in the "Persistent Storage" tab.
70030	The parameters configured for the system function are faulty. The connection to the new PLC was not established.	Compare the parameters configured for the system function with the parameters configured for the PLCs and correct them as necessary.

Number	Effect/causes	Remedy
70031	The PLC configured in the system function is not an S7 PLC. The connection to the new PLC has not been established.	Compare the S7 PLC name parameter configured for the system function with the parameters configured for the PLC and correct them as necessary.
70032	The object configured with this number in the tab sequence is not available in the selected screen. The screen changes but the focus is set to the first object.	Check the number of the tab sequence and correct it if necessary.
70033	An e-mail could not be sent because a TCP/IP connection to the SMTP server no longer exists. This system event is only generated at the first failed attempt. All subsequent unsuccessful attempts to send an e-mail will no longer generate a system event. The event is only generated again if an e-mail has been successfully sent in the meantime. The central e-mail component in WinCC Runtime attempts to connect to the SMTP server at regular intervals (1 minute) in order to transmit the remaining e-mails.	Check the network connection to the SMTP server and re-establish it if necessary.
70034	Following a disruption, the TCP/IP connection to the SMTP server was successfully re-established. The queued e-mails are then sent.	--
70036	No SMTP server for sending e-mails is configured. An attempt to connect to an SMTP server therefore fails and it is not possible to send e-mails. WinCC Runtime generates the system event at the first attempt to send an e-mail.	Configure an SMTP server: In the WinCC Engineering System using "Device settings > Device settings" In the Windows CE operating system using "Control Panel > Internet Settings > E-mail > SMTP Server"
70037	An e-mail could not be sent for unknown reasons. The contents of the e-mail are discarded.	Check the e-mail parameters (recipient etc.).
70038	The SMTP server has rejected the sending or forwarding of an e-mail because the domain of the recipient is unknown to the server or because the SMTP server requires authentication. The contents of the e-mail are discarded.	Check the domain of the recipient address or disable the authentication on the SMTP server if possible. SMTP authentication is currently not used in WinCC Runtime.
70039	The syntax of the e-mail address is incorrect or contains invalid characters. The contents of the e-mail are discarded.	Check the e-mail address of the recipient.
70040	The syntax of the e-mail address is incorrect or contains illegal characters.	--
70041	The import of the user management has been aborted due to an error. Nothing has been imported.	Check your user administration or download it again to the panel.
70042	The range of values of the tag was exceeded while executing the system function. The system function was not calculated.	Check and correct the calculation.
70043	The range of values of the tag was exceeded while executing the system function. The system function was not calculated.	Check and correct the calculation.
70044	An error occurred while sending the e-mails. The e-mails were not sent.	Check the SMTP settings and the error message in the system event.

Number	Effect/causes	Remedy
70045	Cannot load a file required for encrypting the e-mail.	Update the operating system and Runtime.
70046	The server does not support encryption.	Select an SMTP server that supports encryption.
70047	The SSL versions of the HMI device and SMTP server may not be compatible.	Contact your network administrator or the operator of the SMTP server.

80000 - Log alarms (Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below.

Table 12-7 80000 - Log alarms

Number	Effect/causes	Remedy
80001	The log specified is filled to the size defined (in percent) and must be stored elsewhere.	You swap out the file or table by executing a ‘move’ or ‘copy’ function.
80002	A line is missing in the specified log.	--
80003	The copying process for logging was not successful. In this case, it is advisable to check any subsequent system events, too.	--
80006	Since logging is not possible, this causes a permanent loss of the functionality.	In the case of databases, check whether the corresponding data source exists and start up the system again.
80009	A copying action has been completed successfully.	--
80010	An incorrect storage location was entered in WinCC and causes permanent loss of functionality.	Configure the storage location for the respective log again and restart the system when the full functionality is required.
80012	Log entries are stored in a buffer. If the values are read to the buffer faster than they can be physically written (using a hard disk, for example), overloading may occur and recording is then stopped.	Archive fewer values. Or: Increase the logging cycle.
80013	The overload status no longer exists. Archiving resumes the recording of all values.	--
80014	The same action was triggered twice in quick succession. Since the process is already in operation, the action is only carried out once.	--
80015	This system event is used to report DOS or database errors to the user.	--
80016	The logs are separated by system function "CloseAllLogs" and the incoming entries exceed the defined buffer size. All buffer entries are deleted.	Reconnect the logs.
80017	The number of incoming entries cause a buffer overflow. This can be caused, for example, by several copying actions being activated at the same time. All copy jobs in the buffer are deleted.	Stop the copy action.

Number	Effect/causes	Remedy
80019	The connection between WinCC and all log files was shut down, for example, after execution of the "CloseAllLogs" system function. Entries are written to the buffer and written to the logs when the connection is recovered. There is no connection to the storage location and the storage medium can be changed.	--
80020	The maximum number of simultaneously copy operations has been exceeded. Copying is not executed.	Wait until the current copying actions have been completed, then restart the last copy action.
80021	An attempt was made to delete a log which is still busy with a copy action. Deletion has not been executed.	Wait until the current copying actions have been completed, then restart the last action.
80022	An attempt was made to use system function "StartSequenceLog" to start a sequence log for a log which is not configured as sequence log. No sequence log file is created.	In the project, check • whether the "StartSequenceLog" system function was properly configured. • if the tag parameters are properly provided with data on the HMI device.
80023	An attempt was made to copy a log to itself. The log is not copied.	In the project, check • whether the "CopyLog" system function was properly configured. • if the tag parameters are properly provided with data on the HMI device.
80024	In your configuration, you specified that the "CopyLog" system function should not allow copying if the destination log already contains data ("Mode" parameter). The log is not copied.	Edit the "CopyLog" system function in your configuration, if necessary. Before you initiate the system function, delete the destination log file.
80025	You have canceled the copy operation. Data written up to this point are retained. The target log file (if configured) was not deleted. The cancellation is reported by an error entry RT_ERR at the end of the target log.	--
80026	This alarm is output after all logs are initialized. Values are written to the logs from then on. No entries are written to the logs before this time has expired, irrespective of the active state of WinCC Runtime.	--
80027	The internal Flash memory has been specified as the storage location for a log. This is not permissible. No values are written to this log and the log file is not created.	Configure "Storage Card" or network path as the storage location.
80028	The alarm serves as status report indicating that the logs are currently being initialized. No values are logged until the alarm 80026 is output.	--
80029	The number of logs specified in the alarm could not be initialized. Initialization of logs was finished. The faulty log files are not available for logging jobs.	Evaluate the additional system events related to this alarm. Check the configuration, the ODBC (Open Database Connectivity), and the specified drive.
80030	The structure of the existing log file does not match the expected structure. Logging is stopped for this log.	Delete the existing log data manually, in advance.
80031	The log in CSV format is corrupted. The log cannot be used.	Delete the faulty file.

Number	Effect/causes	Remedy
80032	Logs can be configured with events. These are triggered as soon as the log is full. Assuming WinCC Runtime is started and the log is already full, the event would never be triggered. The specified log is full and no longer accepts data.	Close WinCC Runtime, delete the log, and restart WinCC Runtime. Or: Configure a button which contains the same actions as the event and press it.
80033	"System Defined" is set in the data log file as the data source name. This causes an error. No data is written to the database logs, whereas the logging to the CSV logs works.	Reinstall SQL Sever 2005 Express.
80034	An error has occurred in the initialization of the logs. An attempt has been made to create the tables as a backup. This action was successful. A backup copy of the tables of the corrupted log file was generated and the cleared log restarted.	No action is necessary. However, it is recommended to save the backup files or delete them in order to make the space available again.
80035	An error has occurred in the initialization of the logs. An attempt has been made to create backups of the tables and this has failed. No logging or backup has been performed.	It is recommended to save the backups or to delete them in order to release memory.
80044	The export of a log was interrupted because Runtime was closed or due to a power failure. At the restart of Runtime, it was detected that the export must be resumed.	The export resumes automatically.
80045	The export of a log was interrupted due to an error in the connection to the server or at the server itself.	The export is repeated automatically. Check: • The connection to the server. • If the server is running. • If there is enough free space on the server.
80046	The destination file or the associated directory could not be created on the server.	Check whether there is enough space on the server and it you have permission to create the log file.
80047	The log could not be read while exporting it.	Check whether the storage medium is correctly inserted.
80049	The log could not be renamed while preparing to export it. The job can not be completed."	Check whether the storage medium is correctly inserted and if there is sufficient space on the medium.
80050	The log which shall be exported is not closed. The job can not be completed.	Make sure to call the "CloseAllLogs" system function before using the "ExportLog" system function. Change the configuration as required.
80051	The log to be copied contains an invalid checksum. The log was not copied.	Select a log with a valid checksum. The selected log may have been manipulated.
80052	The log cannot be read.	Check the log and the specified path.
80053	The closed log cannot be read.	Open the log.

110000 - Offline function alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below.

Table 12-8 110000 - Offline function alarms

Number	Effect/causes	Remedy
110000	The operating mode was changed. "Offline" mode is now set.	--
110001	The operating mode was changed. "Online" mode is now set.	--
110002	The operating mode was not changed.	Check the connection to the PLCs. Check whether the address range for the “Coordination" area pointer is present in the PLC.
110003	The operating mode of the specified PLC was changed by the "SetConnectionMode" system function. The "offline" operating mode is now set.	--
110004	The operating mode of the specified PLC was changed by the "SetConnectionMode" system function. The "online" operating mode is now set.	--
110005	An attempt was made to use the "SetConnectionMode" system function to set the specified PLC to "online" mode, although the entire system is in "offline" mode. This changeover is not allowed. The PLC remains in "offline" mode.	Switch the complete system to "online" mode and repeat execution of the system function.
110006	The content of the "project ID" area pointer does not match the project ID configured in WinCC. The WinCC Runtime is therefore terminated.	Check: • the project ID entered on the PLC. • the project ID entered in WinCC.

120000 - Trend alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below.

Table 12-9 120000 - Trend alarms

Number	Effect/causes	Remedy
120000	The trend is not displayed because you configured an incorrect axis to the trend or an incorrect trend.	Change the configuration.
120001	The trend is not displayed because you configured an incorrect axis to the trend or an incorrect trend.	Change the configuration.
120002	The trend is not displayed because the tag assigned attempts to access an invalid PLC address.	Check whether the data area for the tag is available on the PLC, whether the configured address is correct, or whether the value range for the tag is correct.

140000 - Connection alarms: Connection + device (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below.

Table 12-10 140000 - Connection alarms: Connection + device

Number	Effect/causes	Remedy
140000	An online connection to the PLC is established.	--
140001	The online connection to the PLC was shut down.	--
140003	No tag update or write operations are executed.	Check whether the connection is up and the PLC is switched on. In the Control Panel, check the set parameters using the "Set PG/PC interface" function. Restart the system.
140004	No tag update or write operations are executed due to an incorrect access point, or incorrect module configuration.	Check the connection and whether the PLC is switched on. Check the access point or the module configuration (MPI, PPI, PROFIBUS) in the Control Panel with "Set PG/PC interface". Restart the system.
140005	No tag update or write operations are executed due to an incorrect HMI device address (possibly too high).	Use a different address for the HMI device. Check the connection and whether the PLC is switched on. Check the parameters set in the "Set PG/PC interface" dialog of the Control Panel. Restart the system.
140006	No tag update or write operations are executed due to an incorrect baud rate setting.	Select a different baud rate in WinCC (according to module, profile, communication peer, etc.).
140007	An incorrect bus profile prevents tag updates or write operations (see %1). The following parameters cannot be written to the registry database: 1: Tslot 2: Tqui 3: Tset 4: MinTsdr 5: MaxTsdr 6: Trdy 7: Tid1 8: Tid2 9: Gap factor 10: Retry limit	Check the custom bus profile. Check the connection and whether the PLC is switched on. Check the parameters set in the "Set PG/PC interface" dialog of the Control Panel. Restart the system.

Number	Effect/causes	Remedy
140008	An incorrect baud rate prevents tag updates or write operations. The following parameters cannot be entered in the registry database. 0: general error 1: incorrect version 2: Profile cannot be entered in the registry database. 3: Subnet type cannot be entered in the registry database. 4: Target rotation time cannot be entered in the registry database. 5: Highest address (HSA) is incorrect.	Check whether the connection is up and the PLC is switched on. In the Control Panel, check the set parameters using the "Set PG/PC interface" function. Restart the system.
140009	No tag updates or write operations because the S7 communication module was not found.	To reinstall the module, open the Control Panel and select "Set PG/PC interface".
140010	No S7 communication partner found because the PLC is shut down. DP/T: The option "PG/PC is the only master" is not set in the Control Panel under "Set PG/PC interface".	Switch on the PLC. DP/T: If only one master is connected to the network, disable "PG/PC is the only master" in "Set PG/PC interface". If several masters are connected to the network, enable these. Do not change any settings, for this will cause bus errors.
140011	No tag updates or write operations because communication is down.	Check the connection and whether the communication partner is switched on.
140012	Initialization error (e.g. if WinCC Runtime was closed in Task Manager). Or: Another application (e.g. STEP 7) with different bus parameters is active and the driver cannot be started with the new bus parameters (baud rate, for example).	Restart the HMI device. Or: Start WinCC Runtime and then start your other applications.
140013	The MPI cable is disconnected and, therefore, there is no power supply.	Check the connections.
140014	The configured bus address is in use by another application.	Change the HMI device address in the PLC configuration.
140015	Incorrect baud rate Or: Incorrect bus parameters (e.g. HSA) Or: OP address > HSA or: Incorrect interrupt vector (interrupt not registered by the driver)	Correct the parameters.
140016	The hardware does not support the configured interrupt.	Change the interrupt number.
140017	The set interrupt is in use by another driver.	Change the interrupt number.
140018	SIMOTION Scout disabled the consistency check. Only a corresponding note appears.	In SIMOTION Scout, reactivate the consistency check and once again download the project to the PLC.
140019	SIMOTION Scout is downloading a new project to the PLC. Connection to the PLC is canceled.	Wait until the end of the reconfiguration.
140020	Mismatch of the PLC and project (FWX file) versions. The connection to the PLC is canceled.	The following remedies are available: Download the current version to the PLC using SIMOTION Scout. Recompile the project using WinCC ES, close WinCC Runtime, and restart with the new configuration.

Number	Effect/causes	Remedy
140021	Setup of the connection to the PLC failed. Incorrect configuration of the "Access password" for the connection to the PLC.	Select the "Access password" area in the "Connections" editor to check the password entered for the connection to the PLC. Assign the correct password. The "Password" for the connection to the PLC is assigned in the "Security" area of the PLC properties.
140022	Setup of the connection to the PLC failed. Incorrect configuration of the access password for the connection to the PLC.	Select the "Access password" area in the "Connections" editor to check the password entered for the connection to the PLC. The "Password" for the connection to the PLC is assigned in the "Security" area of the PLC properties.
140023	Error during time synchronization: Unable to read system time of PLC %1.	Check whether the PLC is switched on. Select the "Access password" area in the "Connections" editor to check the password entered for the connection to the PLC.
140025	Setup of the connection to the PLC failed. The access password of the PLC is disabled in the PLC display.	Enable the access password in the PLC display.

180000 - General alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below.

Table 12-11 180000 - General alarms

Number	Effect/causes	Solution
180000	A component/OCX received configuration data with a version ID which is not supported.	Install a newer component.
180001	System overload because too many actions are running in parallel. Certain actions can be executed, while others are discarded.	Several remedies are available: • Generate alarms at a slower rate (polling). • Initiate scripts and functions at greater intervals. If the alarm appears more frequently: Restart the HMI device.
180002	The screen keyboard could not be activated. Possible causes: "TouchInputPC.exe" was not registered due to faulty Setup.	Reinstall WinCC Runtime.

Note

Alarms without additional information

Alarm 180003 is not described since the alarm text contains all the necessary information.

190000 - Tag alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Meaning of the system events

The most important system events are listed below.

Table 12-12 190000 - Tag alarms

Number	Effect/causes	Remedy
190000	It is possible that the tag is not updated.	--
190001	The tag is updated after the cause of the last error state has been eliminated (recovery of normal operation).	--
190002	The tag is not updated because communication with the PLC is down.	Select system function "SetOnline" to enable communication.
190004	The tag is not updated because the address configured for this tag does not exist.	Check the configuration.
190005	The tag is not updated because the configured PLC type does not exist for this tag.	Check the configuration.
190006	The tag is not updated because it is not possible to map the PLC type in the data type of the tag.	Check the configuration.
190007	The tag value is not modified because the connection to the PLC is interrupted or the tag is offline.	Set online mode or reconnect to the PLC.
190008	The configured tag limits were violated due to one of the following events: • Value input • System function • Script	Observe the configured or current tag limits.
190009	An attempt was made to assign this tag a value that is outside the valid range of values for this data type. For example, input of the value 260 for a byte tag, or input of the value -3 for an unsigned word tag.	Observe the range of values for the data type of the tags.
190010	The rate at which values are written to the tag is too high (e.g. initiated in a script loop). Values will be lost because buffer capacity is limited to 100 operations.	The following remedies are available: • Extend the interval between multiple write actions. • Do not use an array tag longer than 6 words when configuring an acknowledgment on the HMI device using "HMI acknowledgment tag".
190011	Possible cause 1: The value entered could not be written to the configured PLC tag because the high or low limit was exceeded. The system discarded the entry and restored the original value. Possible cause 2: The connection to the PLC was interrupted.	Note that the value entered must be within the range of values of the control tag. Check the connection to the PLC.

Number	Effect/causes	Remedy
190012	It is not possible to convert a value from a source format to a target format, for example: An attempt is being made to write a counter value that is outside the valid, PLC-specific range of values. A tag of the type Integer should be assigned a value of the type string.	Check the range of values, or the data type of the tag.
190013	You entered a string that exceeds the tag length. The string is truncated automatically to a valid length.	Always enter strings that do not exceed the valid tag length.

190100 - Area pointer alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

190100 - Area pointer alarms

The most important system events are listed below.

Number	Effect/causes	Remedy
190100	The area pointer is not updated because the address configured for this pointer does not exist. Type 1 Warnings 2 Errors 3 PLC acknowledgment 4 HMI device acknowledgment 5 LED image 6 Trend request 7 Trend transfer 1 8 Trend transfer 2 No.: Consecutive number displayed in WinCC ES.	Check the configuration.
190101	The area pointer is not updated because it is not possible to map the PLC type to the area pointer type. Parameter type and no.: see alarm 190100	--
190102	The area pointer is updated after the cause of the last error state has been eliminated (recovery of normal operation). Parameter type and no.: See alarm 190100.	--

200000 - PLC coordination alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

200000 - PLC coordination alarms

The most important system events are listed below.

Number	Effect/causes	Remedy
200000	Coordination is not executed because the address configured in the PLC does not exist/is not set.	Change the address or set up the address in the PLC.
200001	Coordination is canceled because the write access to the address configured in the PLC is not possible.	Change the address or set the address in the PLC at an area which allows write access.
200002	Coordination is not carried out at the moment because the address format of the area pointer does not match the internal storage format.	Internal error
200003	Coordination can be executed again because the last error state is eliminated (return to normal operation).	--
200004	The coordination may not be executed.	--
200005	No more data is read or written. Possible causes: • The cable connection is interrupted. • The PLC does not respond or has failed, etc. • System overload	Ensure that the cable is plugged in and the PLC is operational. Restart the system if the system event persists.

200100 - Area pointer alarms project ID (Basic Panels, Panels, Comfort Panels, RT Advanced)

200100 - Area pointer alarms project ID

The most important system events are listed below.

Number	Effect/causes	Remedy
200101	The project ID of the area pointer cannot be described. No entry possible.	Check the configuration.
200102	Error in the type conversion.	Check the type conversion.
200103	The project ID of the area pointer is updated as the cause of the last error state has been eliminated (recovery of normal operation).	--

Number	Effect/causes	Remedy
200104	Unknown error	Check the • Configuration • HMI tags • HMI alarms • System alarms • Controller alarms If the error occurs repeatedly, contact the hotline!
200105	The online connection to the PLC was interrupted. The connection is interrupted.	Check the following conditions, for example • Are the access correct? • Correct server name? • Correct station name? • Is the server registered? • Is the configuration correct?

210000 - PLC job alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

210000 - PLC job alarms

The most important system events are listed below.

Number	Effect/causes	Remedy
210000	Jobs are not processed because the configured address does not exist/has not been set up in the PLC.	Change the address or set up the address in the PLC.
210001	Jobs are not processed because read/write access to the configured address is not possible in the PLC.	Change the address, or set up the address in a PLC area at which read/write access is possible.
210002	Jobs are not executed because the address format of the area pointer does not match the internal storage format.	Internal error
210003	The job buffer is processed again because the last error status has been eliminated (recovery of normal operation).	--
210004	The job buffer is possibly not going to be processed.	--
210005	A job mailbox with invalid number was initiated.	Check the PLC program.
210006	An error occurred while executing the job mailbox. As a result, the control job is not executed. Observe the next/previous system event.	Check the parameters of the control job. Recompile the configuration data.

220000 - WinCC communication driver alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

220000 - WinCC communication driver alarms

The most important system events are listed below.

Number	Effect/causes	Remedy
220001	The tag is not downloaded because write access to data type Bool/Bit is not supported by the sublevel communication driver/HMI device.	Change the configuration.
220002	The tag is not downloaded because write access to data type Byte is not supported by the sublevel communication driver/HMI device.	Change the configuration.
220003	The communications driver could not be loaded. The driver might not be installed.	Install the driver by reinstalling WinCC Runtime.
220004	Communication is down and no update data is transferred because the cable is not connected or defective etc.	Check the connection.
220005	Communication is up.	--
220006	The connection between the specified PLC and the specified port is active.	--
220007	The connection to the specified PLC is interrupted at the specified port.	Check the following: • Is the cable plugged in? • Is the PLC OK? • Is the right port being used? • Is your configuration OK (port parameters, protocol settings, PLC address)? Restart the system if the system event persists.
220008	The communication driver cannot access or open the specified port. This port might be in use by another application, or a port that does not exist on the target device is being used. No communication with the PLC.	Close all applications that access the port and restart the computer. Use another port that exists in the system.
220009	The communication path is either empty, incomplete or has an invalid format.	Enter the communication path in the correct format.

230000 - Screen object alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

230000 - Screen object alarms

The most important system events are listed below.

Number	Effect/causes	Solution
230000	The value entered could not be used. The system discards this entry and restores the previous value. Possible causes: • The range of values is exceeded. • You entered invalid characters • The valid maximum number of users has been exceeded.	Enter a practical value, or delete a user that is no longer required.
230002	The user currently logged on does not have the necessary authorization, so the system discards the entry and restores the previous value.	Log on as user with appropriate authorization.
230003	Change to the specified screen failed because this screen is not available/configured. The current screen remains selected.	Configure the screen and check the selection function.
230005	The range of values of the tag has been exceeded in the I/O field. The original tag value is retained.	Observe the range of values for the tag when entering a value.
230100	During navigation in the Web browser, the system returned a message which may be of interest to the user. The Web browser continues to run but may not (fully) show the new page.	Navigate to another page.
230200	The HTTP channel connection was interrupted due to an error. This error is explained in detail by another system event. Data is no longer exchanged.	Check the network connection. Check the server configuration.
230201	The HTTP channel connection is set up. Data is exchanged.	--

Number	Effect/causes	Solution
230202	WININET.DLL has detected an error. Usually, this error occurs if it is not possible to connect to the server, or the server denies a connection because the client lacks proper authorization. An unknown server certificate may also be the cause if the connection is encrypted by means of SSL. The alarm text provides details. This text is always in the language of the Windows installation because it is returned by the Windows OS. Process values are no longer exchanged. The part of the alarm returned by the Windows OS might not be displayed, e.g. "An error has occurred". WININET.DLL returns the following error: Number: 12055 Text:HTTP: <no error text available>."	Depending on the cause: When an attempt to connect fails, or a timeout occurs: • Check the network connection and the network. • Check the server address. • Check whether the WebServer is actually running on the target station. Incorrect authorization: • The configured user name and/or password do not match the entries on the server. Set consistent data. If the server certificate is rejected: Certificate signed by an unknown CA ( ): • Ignore this point, or install a certificate that has been signed with one of the root certificates known to the client station. The date of the certificate is invalid: • Ignore this point, or install a certificate with valid date on the server. Invalid CN (Common Name or Computer Name): • Ignore this point, or install a certificate with a name that corresponds to the server address.
230203	Although a connection can be made to the server, the HTTP server refused to connect. Possible causes: • WinCC Runtime does not run on the server • The HTTP channel is not supported (503 Service unavailable). Other errors can only occur if the Webserver does not support the HTTP channel. The language of the alarm text depends on the Webserver. Data is not exchanged.	On 503 Service unavailable error: Check whether WinCC Runtime is running on the server and whether the HTTP channel is supported.
230301	Internal error. An English text explains the error in more detail. The error may be caused by insufficient memory. OCX does not work.	--
230302	The name of the remote server cannot be resolved. An attempt to connect has failed.	Check the configured server address. Check whether the DNS service is available on the network.
230303	The remote server is not running on the addressed computer. incorrect server address. An attempt to connect has failed.	Check the configured server address. Check whether the remote server is running on the target computer.
230304	The remote server on the addressed computer is incompatible with VNCOCX. An attempt to connect failed.	Use a compatible remote server.

Number	Effect/causes	Solution
230305	Authentication has failed due to incorrect password. An attempt to connect failed.	Configure the correct password.
230306	Error in the connection to the remote server. This may occur as a result of network problems. An attempt to connect failed.	Check whether the network cable is plugged in, or whether there are network problems.
230307	The connection to the remote server was shut down. Possible causes: • The remote server was shut down • The user instructed the server to close all connections. The connection is cancelled.	--
230308	This alarm provides information on the connection status. An attempt is made to connect.	--

Note

Alarms without additional information

Alarms 230400-230409 and 230500-230501 are not described since the alarm text contains all the necessary information.

260000 - Password system alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

260000 - Password system alarms

The most important system events are listed below.

Number	Effect/causes	Solution
260000	An unknown user or an unknown password has been entered in the system. The current user is logged off from the system.	Log on to the system as a user with a valid password.
260001	The logged in user does not have sufficient authorization to execute the protected functions on the system.	Log on to the system as a user with sufficient authorization.
260002	This alarm output when the "TrackUserChange" system function is triggered.	--
260003	The user has logged off from the system.	--
260004	The user name entered into the user view already exists in the user management.	Select another user name because user names have to be unique in the user management.
260005	The entry is discarded.	Enter a shorter user name.
260006	The entry is discarded.	Use a shorter or longer password.
260007	The logoff time entered is outside the valid range from 0 to 60 minutes. The new value entered is discarded and the original value is retained.	Enter a logon timeout value between 0 and 60 minutes.

Number	Effect/causes	Solution
260008	An attempt was made in WinCC to read a PTProRun.pwl file created with ProTool V 6.0. Reading of the file was canceled due to incompatibility of the format.	--
260009	You have attempted to delete the user "Administrator" or "PLC User". These users are fixed components of the user management and cannot be deleted.	If you need to delete a user, because perhaps you have exceeded the maximum number permitted, delete another user.
260012	The password entries in the "Change Password" dialog and in the confirmation field do not match. The password is not changed. User will be logged off.	You have to log on to the system again. Then enter the identical password twice to be able to change the password.
260013	The password entered in the "Change Password" dialog is invalid because it is already in use. The password is not changed. User will be logged off.	You have to log on to the system again. Then enter a new password that has not been used before.
260014	You have tried to log on with an incorrect password three times in a row. You will be locked out and assigned to group no. 0.	You can log on to the system with your correct password. Only an administrator can change the assignment to a group.
260024	The password you entered does not meet the necessary security guidelines.	Enter a password that contains at least one number.
260025	The password you entered does not meet the necessary security guidelines.	Enter a password that comprises at least three characters.
260026	The password you entered does not meet the necessary security guidelines.	Enter a password that contains at least one special character.
260028	Upon system start-up, an attempt to log on, or when trying to change the password of a SIMATIC log-on user, the system attempts to access the SIMATIC Logon Server. If attempting to log on, the new user is not logged in. If a different user was logged on before, then this user is logged off.	Check the connection to the SIMATIC Logon Server and its configuration; for example: 1. Port number 2. IP address 3. Server name 4. Functional transfer cable. Or use a local user.
260030	The SIMATIC Logon user could not change his password on the SIMATIC Logon Server. The new password is possibly noncompliant with password rules set on the server, or the user is not authorized to change his password. The old password remains and the user is logged off.	Log in again and choose a different password. Check the password rules on the SIMATIC Logon Server.
260033	The action change password or log on user could not be carried out.	Check the connection to the SIMATIC Logon Server and its configuration; for example: 1. Port number 2. IP address 3. Server name 4. Functional transfer cable Or use a local user.

Number	Effect/causes	Solution
260034	The last logon operation has not yet ended. A user action or a logon dialog can therefore not be called. The logon dialog is not opened. The user action is not executed.	Wait until the logon operation is complete.
260035	The last attempt to change the password was not completed. A user action or a logon dialog can therefore not be called. The logon dialog is not opened. The user action is not executed.	Wait until the procedure is complete.
260036	There are insufficient licenses on the SIMATIC Logon Sever. The logon is not authorized.	Check the licensing on the SIMATIC Logon Server.
260037	There is no license on the SIMATIC Logon Sever. A logon is not possible. It is not possible to log on via the SIMATIC Logon Server, only via a local user.	Check the licensing on the SIMATIC Logon Server.
260040	The system attempts to access the SIMATIC Logon Server upon system start-up or when trying to change the password. If attempting to log on, the new user is not logged in. If a different user was logged on before, then this user is logged off.	Check connection to the domain and its configuration in the Runtime security settings editor. Or use a local user.
260043	It was not possible to log the user on to the SIMATIC Logon Server. The user name or the password could be incorrect or the user does not have sufficient rights to log on. The new user is not logged in. If a different user was logged on before, then this user is logged off.	Try again. If necessary, check the password data on the SIMATIC Logon Server.
260044	It was not possible to log the user on to the SIMATIC Logon Server as his account is blocked. The new user is not logged in. If a different user was logged on before, then this user is logged off.	Check the user data on the SIMATIC Logon Server.
260045	The SIMATIC Logon user is not associated to any or several groups. The new user is not logged in. If a different user was logged on before, then this user is logged off.	Check user data on the SIMATIC Logon Server and the configuration in your WinCC project. A user may only be assigned to one group.

Note

Alarms without additional information

Alarms 260010, 260011, 260015-260023, 260029, 260031, 260032, 260038, 260039, 260041, 260042 and 260048 are not described since the alarm text contains all the necessary information.

270000 - System alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

270000 - System Alarms

The most important system events are listed below.

Number	Effect/causes	Remedy
270000	The alarm does not indicate the tag because it is accessing an invalid address in the PLC.	Check whether the data area for the tag exists on the PLC, whether the configured address is correct, and whether the value range for the tag is correct.
270001	There is a device-specific limit as to how many alarms may be queued for viewing (see the operating instructions). This limit has been exceeded. The view no longer contains all the alarms. However, all alarms are written to the alarm buffer.	--
270002	The view shows alarms of a log for which there is no data in the current project. Placeholders are output for the alarms.	Delete old log data, if necessary.
270003	The service cannot be set up because too many devices want to access this service. A maximum of four devices can execute this action.	Reduce the number of HMI devices which want to use the service.
270004	Access to the persistent alarm buffer is not possible. Alarms cannot be restored or backed up.	If the problems persist at the next restart, contact Customer Support (delete Flash).
270005	Persistent alarm buffer corrupted: Alarms cannot be restored.	If the problems persist at the next restart, contact Customer Support (delete Flash).
270006	Project modified: Alarms cannot be restored from the persistent alarm buffer.	The project was compiled and downloaded again to the HMI device. The error should no longer occur at the next restart of the HMI device.
270007	A configuration problem is preventing you from the restoring the data (e.g. a DLL was deleted, or unknown directory).	Update the operating system and download your project again to the HMI device.

280000 - Alarms SIMATIC 505 DP Communication Driver (CSP)

Alarms SIMATIC 505 DP Communication Driver (CSP)

The most important system events are listed below.

Number	Effect/causes	Solution
280000	The connection between the controller and port is active.	--
280001	The connection to the PLC at the port is interrupted.	Check the following: • Is the cable plugged in? • Is the controller active? • Is the correct port being used? • Is the configuration OK?
280002	The parameters of the controller are correct, the XSUB and .REC files are available.	--
280003	• The parameters of the controller are correct, the XSUB and .REC files are not available. • The controller is in STOP mode.	• Download the XSUB and .REC files to the controller. • Verify that the controller is in STOP mode.
280004	• The communication cable between HMI device and controller is disconnected. • The controller is in STOP mode.	Check the following: • The communication cable between operating unit and controller • Mode of the controller (Run/STOP)

290000 - Recipe system alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

290000 - Recipe system alarms

The most important system events are listed below.

Number	Effect/causes	Solution
290000	The recipe tag could not be read or written. It is assigned the start value. The alarm can be entered in the alarm buffer for up to four more faulty tags. After that, alarm 290003 is output.	Check the configuration to see whether the address has been set up in the PLC.
290001	An attempt was made to assign the recipe tag a value that is outside the valid range of values for this type. The alarm can be entered in the alarm buffer for up to four more faulty tags if necessary. After that, alarm 290004 is output.	Observe the range of values for the tag type.
290002	It is not possible to convert a value from a source format to a target format. The alarm can be entered in the alarm buffer for up to four more faulty recipe tags if necessary. After that, alarm 290005 is output.	Check the range of values or type of the tag.

Number	Effect/causes	Solution
290003	This alarm is output after alarm 290000 was triggered more than five times. In this case, no further separate alarms are generated.	Check the configuration to see whether the tag addresses have been set up in the PLC.
290004	This alarm is output after alarm 290001 was triggered more than five times. In this case, no further separate alarms are generated.	Observe the range of values for the tag type.
290005	This alarm is output after alarm 290002 was triggered more than five times. In this case, no further separate alarms are generated.	Check the range of values or type of the tag.
290006	The limits configured for the tag have been exceeded by the values entered.	Observe the configured or current tag limits.
290007	There is a difference between the source and target structure in the recipe currently being processed. The source structure contains an additional recipe tag that is not available in the target structure and cannot be assigned. The value is discarded.	Remove the specified recipe tag in the specified recipe from your configuration.
290008	There is a difference between the source and target structure in the recipe currently being processed. The target structure contains an additional recipe tag that is not available in the source structure. The recipe tag specified is assigned its start value.	Insert the specified recipe tag into the source structure.
290010	The storage location configured for the recipe is invalid. Possible causes: Invalid characters, write protection, data carrier out of space or not available.	Check the configured storage location.
290011	A data record of the specified number does not exist.	Check the source for the number (constant or tag value).
290012	A recipe of the specified number does not exist.	Check the source for the number (constant or tag value).
290013	An attempt was made to save a data record under a data record number that already exists. The operation is not executed.	The following remedies are available: • Check the source for the number (constant or tag value). • First, delete the data record. • Modify the "Overwrite" function parameter.
290014	The specified import file was not found.	Check the following: • The file name • Ensure that the file is in the specified directory.
290020	Check back to verify that the download of data records from the HMI device to the PLC has started.	--
290021	Check back to verify that the download of records from the HMI device to the PLC was completed.	--

Number	Effect/causes	Solution
290022	Check back to indicate that the download of data records from the HMI device to the PLC was canceled due to an error.	Check the following conditions in the configuration: • Are the tag addresses configured in the PLC? • Does the recipe number exist? • Does the data record number exist? • Is the "Overwrite" function parameter set?
290023	Check back to verify that the download of data records from the PLC to the HMI device has started.	--
290024	Check back to verify that the download of data records from the PLC to the HMI device was completed.	---
290025	Check back to indicate that the download of data records from the PLC to the HMI device was canceled due to an error.	Check the following conditions in the configuration: • Are the tag addresses configured in the PLC? • Does the recipe number exist? • Does the data record number exist? • Is the "Overwrite" function parameter set?
290026	An attempt was made to read/write a data record that is not free at present. This error can occur if recipes were configured for download with synchronization.	Set the mailbox status to zero.
290027	Unable to connect to the PLC at present. As a result, the data record cannot be read or written. Possible causes: No hardware connection to the PLC (no cable plugged in, cable is defect), or the PLC is switched off.	Check the connection to the PLC.
290030	This alarm is output after you selected screen which contains a recipe view in which a data record is already selected.	Reload the data record from the storage location, or retain the current values.
290031	While saving, it was detected that a data record with the specified number already exists.	Overwrite the data record, or cancel the action.
290032	During the export of data records, a file with the specified name was found.	Overwrite the file, or cancel the process.
290033	Confirmation prompt before deleting data records.	--
290040	A data record error with error code %1 that cannot be described in more detail occurred. The action is canceled. It is possible that the mailbox was not installed correctly on the PLC.	Check the storage location, the data record, the "Data record" area pointer, and the connection to the PLC. Restart the action after a short waiting time. If the error persists, contact Customer Support. Forward the relevant error code to Customer Support.
290041	A data record or file cannot be saved because the storage location is out of sufficient space.	Delete files no longer required.
290042	An attempt was made to execute several recipe actions simultaneously. The last action is not executed.	Retrigger the action after a short waiting time.
290043	Confirmation prompt before saving data records.	--
290044	The database for the recipe was corrupted and will be deleted.	--

Number	Effect/causes	Solution
290050	A check back indicates that the export of data records was started.	--
290051	A check back indicates successful completion of the export of data records.	--
290052	A check back indicates that the export of data records was canceled due to an error.	Ensure that the structure of the data records at the storage location and the current recipe structure on the HMI device are identical.
290053	A check back indicates that the import of records was started.	--
290054	A check back indicates successful completion of the import of data records.	--
290055	A check back indicates that the import of data records was canceled due to an error.	Ensure that the structure of the data records at the storage location and the current recipe structure on the HMI device are identical.
290056	Error when reading/writing the value in the specified row/column. The action was canceled.	Check the specified row/column.
290057	The tags of the recipe specified were toggled from "offline" to "online" mode. Each change of a tag in this recipe is now immediately transferred to the PLC.	--
290058	The tags of the specified recipe were toggled from "online" to "offline" mode. Modifications to tags in this recipe are no longer immediately transferred to the PLC but must be transferred there explicitly by downloading a data record.	--
290059	A check back indicates that the specified record was successfully saved.	--
290060	A check back indicates that the specified data record memory was cleared.	--
290061	A check back indicates that deletion of the data record memory was canceled due to an error.	--
290062	The data record number exceeds the maximum of 65536. This data record cannot be created.	Select another number.
290063	This occurs when you execute system function "ExportDataRecords" while the "Overwrite" parameter is set to "No". An attempt was made to save a recipe under a file name that already exists. The export is canceled.	Check the parameters of the "ExportDataRecords" system function.
290064	A check back indicates that the deletion of data records was started.	--
290065	A check back indicates successful completion of the deletion of data records.	--
290066	Confirmation prompt before deleting data records.	--
290068	Confirmation prompt for deletion of the selected recipe data records.	--

Number	Effect/causes	Solution
290069	Confirmation prompt for deletion of all recipe data records.	--
290070	The data record specified was not found in the import file.	Check the source of the data record number, or the data record name (constant, or tag value).
290071	When the editing data record values, you entered a value that is below the low limit of the recipe tag. The entry is discarded.	Enter a value within the recipe tag limits.
290072	When editing data record values, you entered a value that exceeds the high limit of the recipe tag. The entry is discarded.	Enter a value within the recipe tag limits.
290073	An action (e.g. saving a record) failed for an unknown reason. The error corresponds to status alarm IDS_OUT_CMD_EXE_ERR in the large recipe view.	--
290074	While saving, a data record with the specified number but with different name was found.	Overwrite the record, change the record number or cancel the action.
290075	A data record of this name already exists. The data record is not saved.	Select a different data record name.
290110	The default values could not be set due to an error.	--
290111	The recipes subsystem cannot be used. Recipe views have no content and recipe-specific functions will not be executed. Possible causes: • Error when loading the recipes. • The recipe structure was changed in the ES. The recipes were not included in the latest project download. This means that the new configuration data no longer matches the old recipes on the device.	Download the project to the device again, including the recipes (the corresponding check box in the download dialog must be check marked).

Note

Alarms without additional information

Alarms 290067 and 290076-290109 are not described since the alarm text contains all the necessary information.

12.4 Working with logs

12.4.1 Log basics

Introduction

WinCC provides the following types of log for logging process data for HMI Runtime:

- Data logs
- Alarm logs

A data log is used to log process data from an industrial plant.

An alarm log is used to log alarms that occur in the monitored process.

Principle

The two types of log both have roughly the same structure and largely work in the same way. They are very clear and easy to configure. With both types of log you define the same properties for the log. The same logging methods are also available for both types of log.

The following logging methods are available:

- Circular log
  If a circular log is totally full, the oldest entries are overwritten.
- Segmented circular log
  In a segmented circular log, multiple single logs of the same size are filled in succession. When all log segments are filled, the oldest log segment is overwritten.
- Log with level-dependent system alarm
  A system alarm is triggered when a defined level is reached.
- Log with level-dependent triggering of an event
  The "Overflow" event is triggered when the log is full. The "Overflow" event triggers a system function.

12.4.2 Properties of logs

Introduction

You define the properties of a data log in the "Data logs" editor.

You define the properties of an alarm log in the "Alarm log" editor.

The properties of the data log and alarm log are configured in the same way. You configure the properties either directly in the table of the respective editor or in the log properties of the Inspector window.

General properties

- Name
  You can assign any name to the log. The name must contain at least one letter or one number.
- Storage location
  The storage location specifies where the log is saved. The HMI device determines which storage locations are available.
- Size
  The size of a log depends on the type of log and the selected settings.
  - Size of a data log
    The size of a data log is calculated as follows:
    The number of items * the length of each tag value to be logged.
    In the "Properties" window, the maximum size that the log accepts for retention of the currently selected number of data records is displayed in the input field "Number of data records". The maximum log size is limited by the volume of the storage medium.
  - Size of an alarm log
    The size of an alarm log is calculated from the number of data records and the approximate size of an entry. The size of an entry depends on whether the alarm text and the associated tag values are to be logged as well.
- Restart characteristics
  Under Restart characteristics you can specify that the logging starts when Runtime starts. Check the "Enable logging at runtime start" check box.
  You can also control the behavior when Runtime starts. Enable "Reset log" if you want to overwrite existing logged data with the new data. Select the "Append data to existing log" option if you want to retain the existing logged data. This setting adds the data to be logged to an existing log.

Note

You can use system functions to control the restarting of a log in Runtime.

Automatic log entries

In Runtime, the following log entries are created as standard:

Entry	File format	Log type	Meaning
RT_DIS	Any	Data log	Indicates that the connection to the log was interrupted at this point in time. (A bold line is shown in the trend view for this time period.)
RT_OFF	Any	Data log	Indicates that Runtime was shut down at this point in time. (No line is shown in the trend view for this time period.)

Entry	File format	Log type	Meaning
RT_ERR	Any	Data log Alarm log AuditTrail[1]	Indicates in the destination log that a copy operation was not successful or was interrupted. (The log copy was not fully created.)
$RT_COUNT	*.CSV *. TXT	Data log Alarm log AuditTrail[1]	This entry was created at the end of the log and serves to increase the system performance when Runtime starts.

[1] The "AuditTrail" logging method is not available for all HMI devices.

See also

Storage locations of logs (Page 5563)

12.4.3 Storage locations of logs (Basic Panels)

Storage location of a log

When you configure a log in WinCC, the available storage locations depend on the HMI device in use.

HMI devices	Supported logs			Supported storage locations	Supported storage locations
	Alarms	Tags	Audit Trail		
Basic Panels[1]	No	No	No	-	-
Basic Panels 2nd Generation[2]	Yes	Yes	No	TXT file (Unicode)	USB memory (at USB port)

[1] KP 300, KP 400, KTP 1000, TP 1500

[2] KTP 400, KTP 700, KTP 900, KTP 1200

Note

Logging on network drives

Do not log alarms, tags and Audit Trail directly on a network drive. Power supply can be interrupted at any time. This means there is no guarantee for a reliable operation of logs and audit trails.

Save the logs on your local hard drive or a local storage card. Use the system function "ArchiveLogFile" to save the logs long-term on a network drive. This step ensures reliable operation.

Syntax examples for storage locations

Storage location USB memory:

- <\USB_X61.1\My_Archives\TagLogs>: Saves the log on a USB storage medium connected to the USB port "X61.1" in the subdirectory "My_Archives\TagLogs".

Naming conventions

The log names must be unambiguous in a project. The name of a log must always be unique, regardless of whether different storage locations are selected for the log.

Note

The characters which can be used in the name of the data source depend on the storage location.

The \ / * ? : " < > | characters are not allowed at the following locations:

- File - TXT (Unicode)

The characters _ @ # $ cannot be used as the first character of a name.

Note

Applicable for Basic Panels 2nd Generation

Unicode characters are not supported for the log names and the log paths.

File - TXT (Unicode)

Data is stored in Unicode.

This file format supports all characters that can be used in WinCC and WinCC Runtime. For editing, you will need software that can save files in Unicode, such as Notepad.

Note

Use "File - TXT (Unicode)" as the storage location to log Asian languages.

See also

Basics on alarm logging (Page 5515)

Creating a data log (Page 5471)

Properties of logs (Page 5561)

Log basics (Page 5561)

12.5 Working with recipes

12.5.1 Basics

12.5.1.1 Definition and applications

Introduction

Related data, e.g. machine parameter assignments or production data, are combined in recipes.

Examples:

- Machine parameter settings that are needed to convert production to a different product variant.
- Product components that result in different compositions for different end products.

A recipe has a fixed data structure. The structure of a recipe is defined in the configuration. A recipe contains recipe data records. These differ in terms of their values, but not their structure.

Recipes are saved on the HMI device. A recipe data record is always transferred completely and in a single pass between the HMI device and the PLC.

Note

Restrictions in the import/export

It is not possible to export or import the recipes for Basic Panels.

Complete recipe data but not individual recipe data records can be exported and imported with ProSave to the CSV format and transmitted to the HMI device. Runtime is stopped in the meantime.

Using recipes

Recipes can be used in the following situations:

- Manual production
  You select the required recipe data and display it on the HMI device. You modify the recipe data as required and save it on the HMI device. You transfer the recipe data to the PLC.
- Automatic production
  The control program starts transfer of the recipe data between the PLC and HMI device. You can also start the transfer from the HMI device. Production is then implemented automatically. It is not essential to display or modify the data.
- Teach-in mode
  You optimize production data that was optimized manually on the system, e.g. axis positions or filling volumes. The values thus determined are transferred to the HMI device and saved in a recipe data record. You can then transfer the saved recipe data back to the PLC at a later date.

Entering and modifying the recipe data

You enter the data in the individual recipe data records and modify it as required. The following options are available:

- Data entry during configuration
  If the production data exists already, you enter the data in the "Recipes" editor during recipe configuration.
- Entering the data in Runtime
  If you have to frequently modify production data, you can do this directly in Runtime as follows:
  - Enter the data directly on the HMI device.
  - Set the parameters directly on the machine. You then transfer the data from the PLC to the HMI device and save it in the recipe.

12.5.1.2 Examples for using recipes

Recipes are used in the manufacturing industry and mechanical engineering, for example. The following recipes show typical applications which you can implement with the recipe function of WinCC:

- Machine parameter assignment
  One field of application for recipes is the assignment of machine parameters in the manufacturing industry: A machine cuts wooden boards to a certain size and drills holes. The guide rails and drill have to be moved to new positions according to the board size. The required position data are stored as data records in a recipe. You reassign the machine parameters using "Teach in" mode if, for example, a new board size is to be processed. You transfer the new position data directly from the PLC to the HMI device and save it as a new data record.
- Batch production
  Batch production in the food processing industry represents another field of application for recipes: A filling station in a fruit juice plant produces juice, nectar, and fruit drinks in a variety of flavors. The ingredients are always the same, differing only in their mixing ratios. Each flavor corresponds to a recipe. Each mixing ratio corresponds to a data record. All of the required data for a mixing ratio can be transferred to the machine control at the touch of a button.

12.5.1.3 Structure of recipes

Introduction

The basic structure of a recipe is illustrated with reference to the filling station in a fruit juice plant.

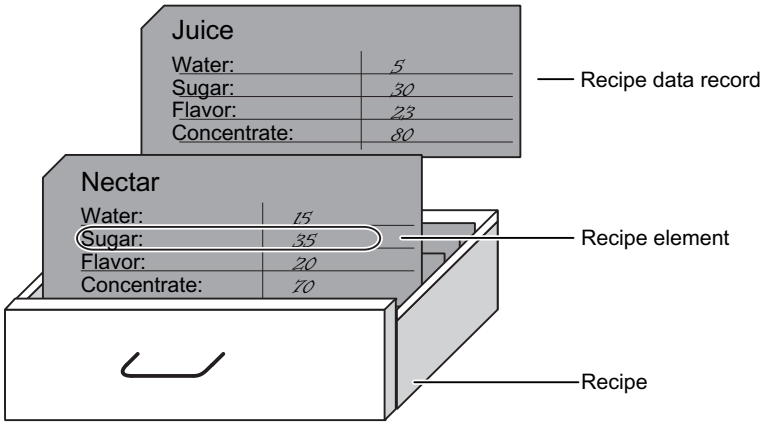
There may be several different recipes in an HMI device. A recipe can be compared to an index card box that contains several index cards. The index card box contains several variants for manufacturing a product family. All the data for each manufacturing variant is contained on a single index card.

Example:

In a soft drinks production plant, a recipe is needed for different flavors. Drink variants include fruit juice drink, juice and nectar.

Recipe

The recipe contains all the recipe data records for the different drink variants.



Recipe data records

Each index card represents a recipe data record needed to manufacture a product variant.

Recipe entries

Each index card in a drawer has the same structure. All the index cards contain fields for the different ingredients. Each field corresponds to a recipe entry. All the records of a recipe thus contain the same entries. The records differ, however, in the value of the individual entries.

Example:

All the drinks contain the following ingredients:

- Water
- Concentrate
- Sugar
- Flavoring

The records for juice drink, fruit juice or nectar differ, however, in the quantity of sugar used in production.

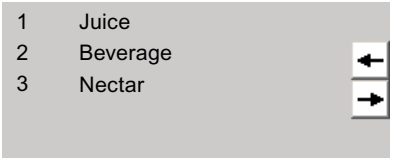
12.5.1.4 Displaying recipes

Introduction

You need to configure the recipe view to display recipes. You can change the values of a recipe in the recipe view and thereby influence the manufacturing process or a machine.

Recipe view

The recipe view is an off-the-shelf WinCC display and operator control for managing recipe data records. The recipe view is always part of a screen. The recipe view shows recipe data records in tabular form. You adapt the appearance and the possible operations to suit your specific needs.



If you are editing recipes with a recipe view in your project, the values are saved in recipe data records. The values are not transferred between the HMI device and PLC until you use the relevant operator control.

12.5.1.5 "Recipes" editor

Introduction

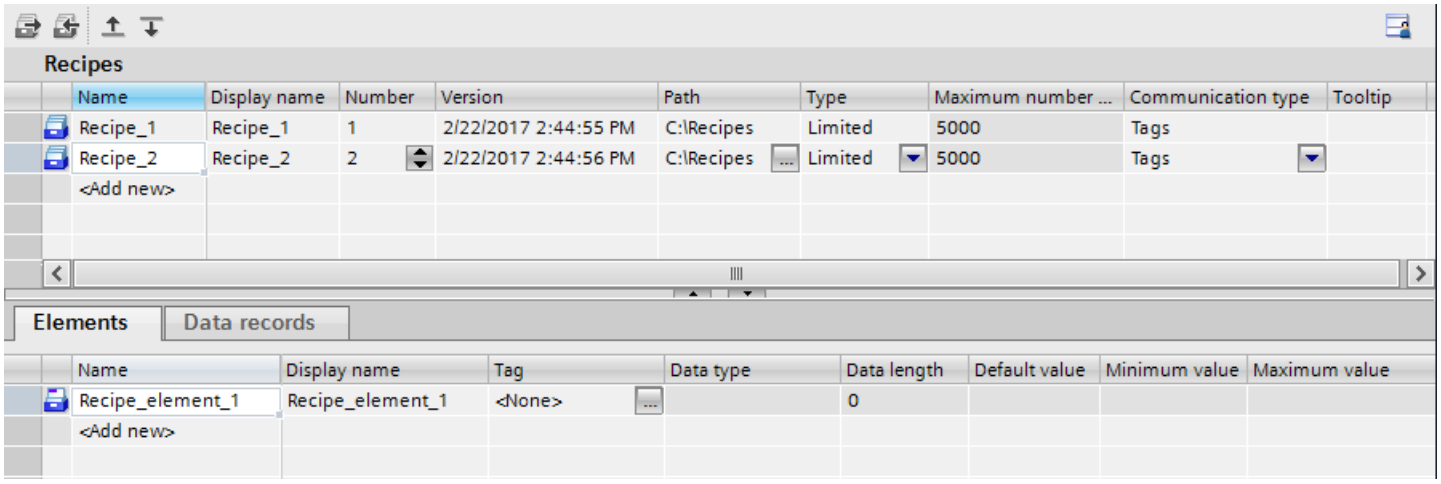
You can create, configure and edit recipes, recipe entries and recipe data records in the "Recipes" editor. The "Recipes" editor also allows you to enter values in recipe data records.

Structure of the "Recipes" editor

You create recipes in the top part of the table editor. You can also configure them there or in the Inspector window.

The bottom part of the table editor has the following tabs:

- Elements
  Define the recipe elements of the selected recipe using the table cells provided here. You can move recipe elements within the table with the shortcut menu commands, "Up" and "Down".
- Data records
  Define the values of the data records of the selected recipe using the table cells provided here.



You can then configure the selected recipe, the recipe element or the recipe data record in the Inspector window. You will find further notes on configuring the components of a recipe under "Configuring Recipes".

Recipe settings

The following settings are available for recipes:

Setting	Description
Name of the recipe	This is a unique identification for the recipe within the HMI device.
Display name	Appears in the recipe view, for example, in Runtime. You can configure display names in multiple languages. Assign descriptive names or designations which the operator can associate directly with a recipe, e.g. "fruit juice drink".
Recipe number	This is a unique identification for the recipe within the HMI device.
Version	Information about the recipe. The date and time of the last change to the recipe is set by default.
Path	Defines the storage location for recipes. The recipes are stored as a file.
Size type [fixed]	The recipe data records are limited to a predetermined number by default.
Number of data records [fixed]	Maximum number of data records in a recipe in Runtime. The number is limited by the recipe memory of the HMI device.
Communication type [fixed]	The recipe data records are written directly to the addresses of the recipe tags and read from there.
Tooltip	Tooltip for the recipe which is shown to the operator in Runtime.

Note

Path

The storage location depends on the storage media available on the HMI device.

Basic Panels and OP77A, TP177A (Portrait)

These HMI devices have no external memory. Recipes are always saved in the internal Flash memory. The "Path" setting is therefore not available.

Recipe element settings

You can make the following settings on the "Elements" tab:

Setting	Description
Name of the recipe element	Identifies a recipe element uniquely within the recipe. Enter meaningful names or labels that you can allocate uniquely, such as axis labels on a machine or ingredients such as "Flavoring".
Display name	Appears in the recipe view, for example, in Runtime. You can configure display names in multiple languages. Assign meaningful names or designations which the operator can associate directly, e.g. "fruit juice flavoring".
Recipe tag	An assigned tag in Runtime stores the current value of the recipe element in the recipe data record.
Data type	Data type of the recipe tag.
Data length [fixed]	Data length of the recipe tag, depending on the data type.
Text list	Text is assigned to a value or range of values in a text list. You can display this text in an output field, for example. The assigned recipe tag must have the data type of a number. The tag value must be within the range of values of the text list.
Default value	This is used as the default entry when you create a new recipe data record.
Minimum value [fixed]	The smallest representable value of a number-based recipe tag, depending on the type of data.
Maximum value [fixed]	The largest representable value of a number-based recipe tag, depending on the type of data.
Decimal places	Determines how many places a decimal number is rounded to, e.g. 3 decimal places and vice versa by what power of ten an integer value is multiplied, e.g. 1,000.
Tooltip	Tooltip about the recipe element which is shown to the operator in Runtime.

Recipe data record settings

You can make the following settings on the "Data records" tab:

Setting	Description
Name of the recipe data record	Identifies a recipe data record uniquely within the recipe.
Display name	Appears in the recipe view, for example, in Runtime. You can configure display names in multiple languages. Assign meaningful names or product numbers which the operator can associate directly with a product, e.g. "yellow fruit juice E231".
Recipe data record number	Identifies a recipe data record uniquely within the recipe.
Recipe elements 1 to n	You can store various values for each recipe element even during configuration. Together with the values of the other recipe elements, a value always forms a recipe data record. You can store multiple recipe data records. If enabled in the transfer settings, the recipe data records are transferred to the HMI device when downloading the project and existing data records on the HMI device are overwritten.
Comment	Comment about the recipe data record

12.5.1.6 Flow of data for recipes

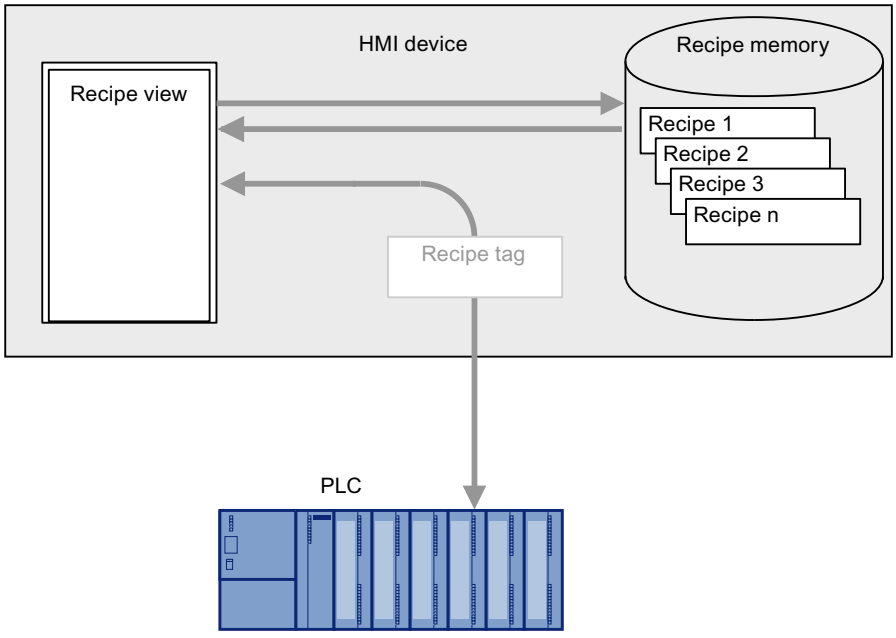
Interaction between the components

There is interaction between the following components at runtime:

- Recipe view
  Recipes are displayed and edited in the recipe view on the HMI device.
  The recipe data records from the internal memory of the HMI device are displayed and edited in the recipe view.
- HMI device recipe memory
  Recipes are saved in the form of recipe data records in the HMI device's recipe memory.
- Recipe tags
  The recipe tags contain recipe data.

Overview of the flow of data

The following figure illustrates the flow of data in recipes:



To transfer recipe data records to the PLC, use the "To PLC" button in the recipe view or an operator control with the system function "RecipeViewSetDataRecordToPLC".

Data are exchanged with the PLC by recipe tags. On Basic Panels you cannot use recipe tags outside a recipe, e.g. not in I/O fields.

12.5.1.7 Synchronization of recipe data records with the PLC

Overview

When recipe data records are transferred between the HMI device and PLC, both communication peers access common communication areas on the other peer.

Recipe data records are always transferred directly. The values of the tags are written directly to or read directly from the configured addresses without being placed on the clipboard.

Data transfer types

There are two ways to transfer recipe data records between the HMI device and PLC:

- Transfer without coordination
- Coordinated transfer via the "Data record" area pointer.

Note

Coordinated transfer

Transfer with coordinated transfer is used to prevent the uncontrolled overwriting of data in either direction in your control program.

Requirements for coordinated transfer

The following requirements apply to coordinated transfer:

- The "Data record" area pointer must be set up for the required connection in the "Communication > Connections" editor.
- In the properties of the recipe "Coordinated transfer of data records" is activated.
- The connection to the PLC is specified in the properties of the recipe with which the HMI device coordinates the transfer.

Coordinated transfer

In the case of coordinated transfer, both the PLC and the HMI device set the status bits in the shared data compartment.

Coordinated transfer of recipe data records can be a useful solution in the following cases:

- The PLC is the "active partner" for the transfer of recipe data records.
- The PLC evaluates information about the recipe number and name, as well as the recipe data record number and name.
- The transfer of recipe data records is started by the following PLC jobs:
  - "Set_data_record_in_PLC"
  - "Get_data_record_from_PLC"

12.5.2 Displaying and editing recipes in Runtime

12.5.2.1 Simple recipe view

Recipe view

The simple recipe view is a ready-made display element and operator control that is used to manage recipe data records. The recipe view shows recipe data records in tabular form.

The displayed buttons and information in the columns are adjustable.

The values displayed or entered in the recipe view are saved in recipe data records. The displayed recipe data record can be written into the PLC by buttons or values can be read in from the PLC.

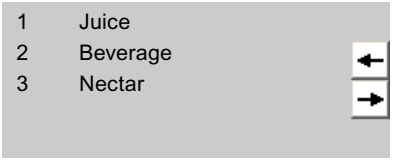
Layout of the display

The simple recipe view consists of three areas:

- Recipe list
- Data record list
- Element list

In the simple recipe view, each area is shown separately on the HMI device. Depending on the configuration, the simple recipe view starts with the recipe list.

The figure below shows an example of the data record list.



Display of values

Note

Processed recipe data record is changed in the background

Only applies for Basic Panels: if an operator has changed a recipe data record and a PLC job wants to read or write any recipe data record of this recipe, the PLC job is stopped and a system alarm is output. On the other hand, the changed value is displayed immediately if only the PLC job and no operator has changed recipe data.

Does not apply for Basic Panels: If an operator has changed a recipe data record and a PLC job has changed the values of the recipe data record concerned, the recipe view is not updated automatically. To update the recipe view, reselect the respective recipe data record.

12.5.2.2 Configuration options of the simple recipe view

The response of the simple recipe view can be defined in the recipe view inspector window. Some devices, for example basic panels, only support the simple recipe view.

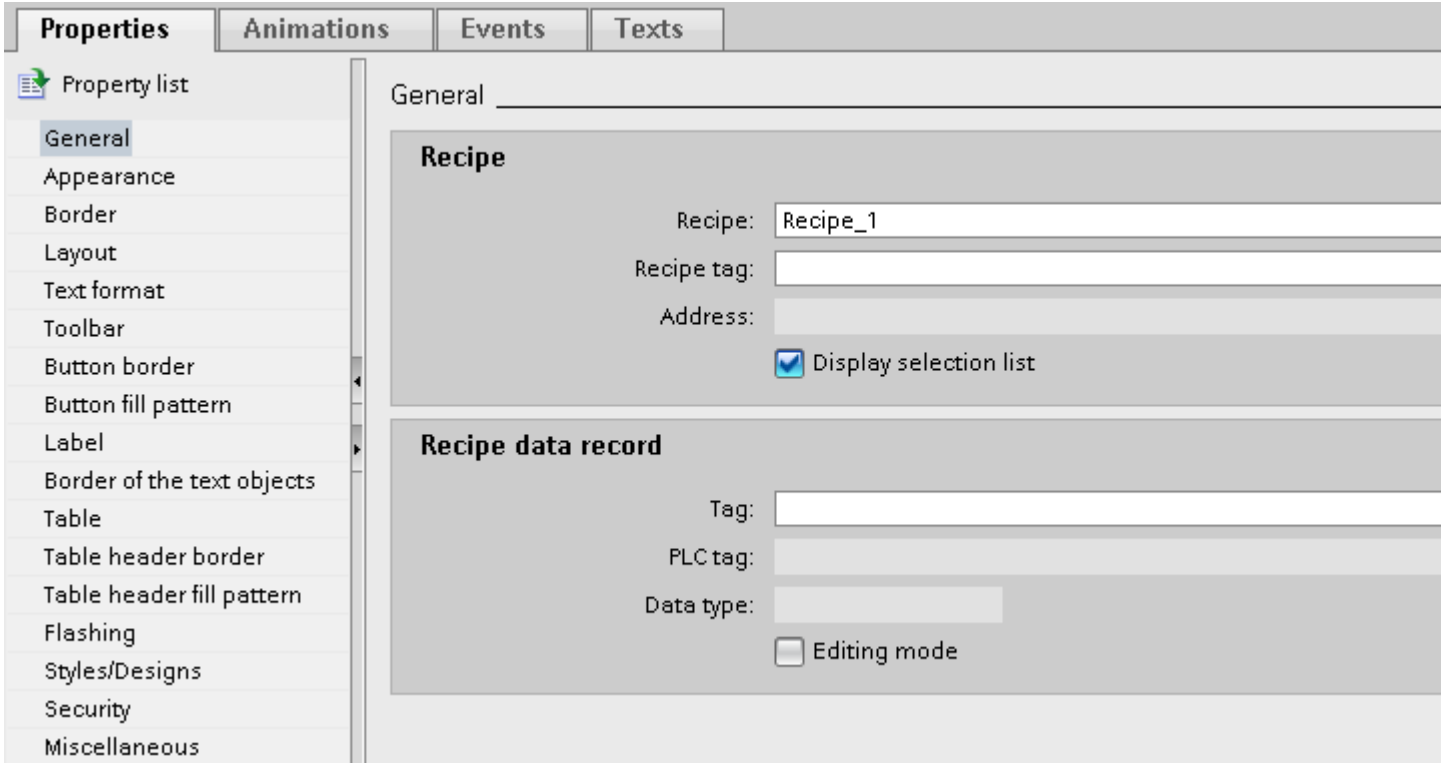
Please note the following for devices with a simple recipe view:

1. In the Inspector window, under "Properties > Display > Mode", select "Simple view" as "View type".
2. The "Properties > Simple view" area contains additional properties only applicable to the simple recipe view.
3. All other properties are also applicable to the advanced recipe view.

Displaying recipe data record values only

To display the recipe data in a recipe view for checking only, proceed as follows:

1. Deselect "Edit mode" in the "General" group.



You cannot create, rename, edit or delete recipe data.

Writing a recipe data record number or name to a tag

A tag to each recipe data record can be configured in the advanced recipe view. Depending on the "String" or "Int" data type of the tag, the name or number of the recipe data record is stored in the tag. Conversely, the tag can also be used to select a recipe data record by entering the corresponding value. For example, the tag can be transferred as a parameter for a system function.

Follow these steps:

1. Enter a tag of the "Int" type in the field "Tag", under "Properties > General > Recipe data record".

The number of the recipe data record is stored in a tag.

Configuring an event on the recipe view

Note

Events and buttons

The "Events" register is faded out when a minimum of one button is enabled.

To configure an event on the recipe view, follow these steps:

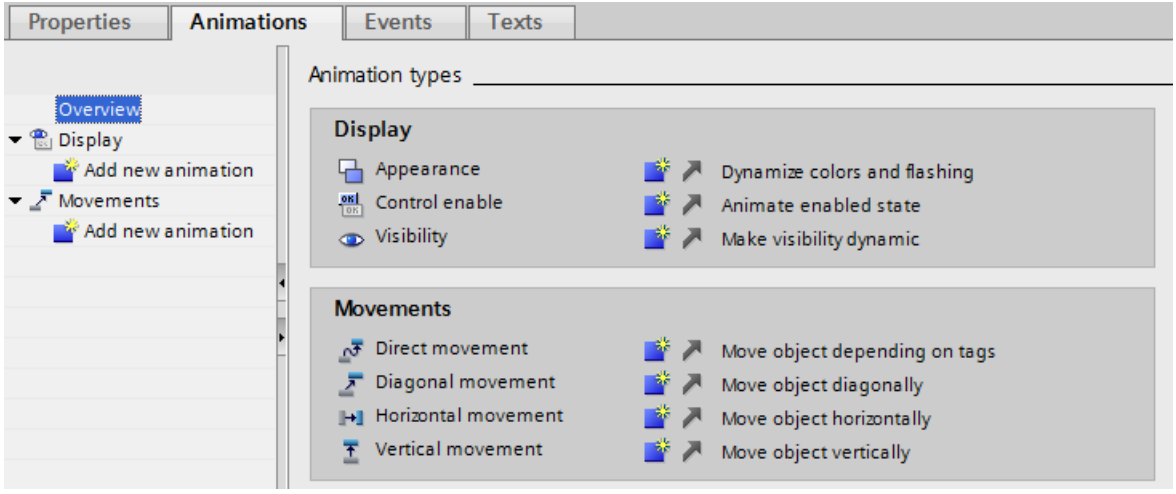
1. Select the recipe view that was added to the screen in the "Screens" editor.
   The properties of the recipe view are displayed in the Inspector window.
2. Deactivate all buttons under "Properties > Toolbar" and "Properties > Simple view".
3. In the Inspector window, under "Properties > Events", click the event to be configured, e.g. "Enable".
4. Configure a function list for the event.

The function list is processed when the operator enables the recipe view.

Animation properties of the recipe view

To configure an animation of a recipe view, proceed as follows:

1. Select the recipe view that was added to the screen in the "Screens" editor. The properties of the recipe view are displayed in the Inspector window.
2. Click under "Properties > Animations" in the Inspector window.



3. Link a tag to one or more of the following properties.
   - X-Position and y-Position
   - Design: Colors, flashing
   - Operability
   - Visibility
     In the "Animations> Overview" area, all animations are summarized in tabular form. Under "Animations> Tag links > Tag linkage", as well as visibility and position a tag can also be linked to height and width.

Note

Animations and buttons

If all buttons are not disabled, an error message appears whilst compiling the project for windows-CE HMI devices.

Constraints on the simple recipe view

The following functions are not possible in the simple recipe view:

- Synchronizing recipe view and recipe tags
- Writing a recipe number / name to a tag
- Display status bar
- Display data record number
- Display label
- Display table

12.5.2.3 Advanced recipe view (as of V13) (Basic Panels, Panels, Comfort Panels, RT Advanced)

Recipe view

The advanced recipe view is an off-the-shelf display and operator control object that is used to manage recipe data records. The recipe view shows recipe data records in tabular form.

The buttons, headers and information displayed in the columns are variable.

The values displayed or entered in the recipe view are saved in recipe data records.

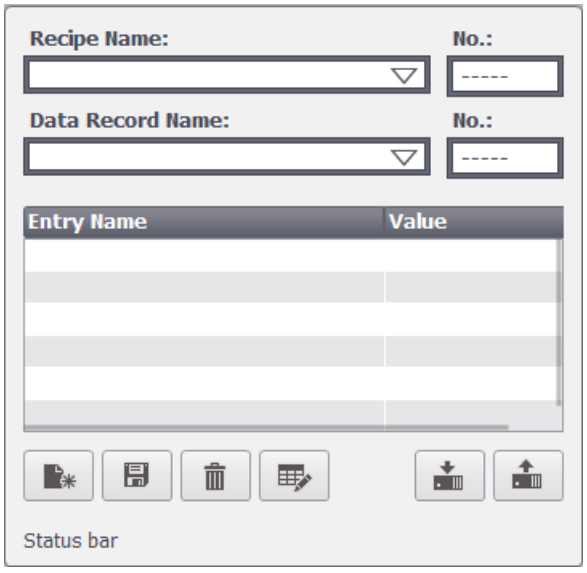
Note

Device dependency of the "Advanced recipe view" object

The "Advanced recipe view" object is available on the HMI devices Basic Panels 2nd Generation, Comfort Panels, Mobile Panels with the device version V13.

Layout of the display

The figure below shows an example of the advanced recipe view:



Show value

Note

Changing the recipe data record in the background

Applies to the processing of a recipe data record:
If values of the corresponding recipe data record are changed by a job mailbox, the recipe view is not updated automatically.

To update the recipe view, you must select the respective recipe data record again.

See also

Behavior of the recipe view in Runtime (Page 5581)

Recipe view (as of V13) (Page 5411)

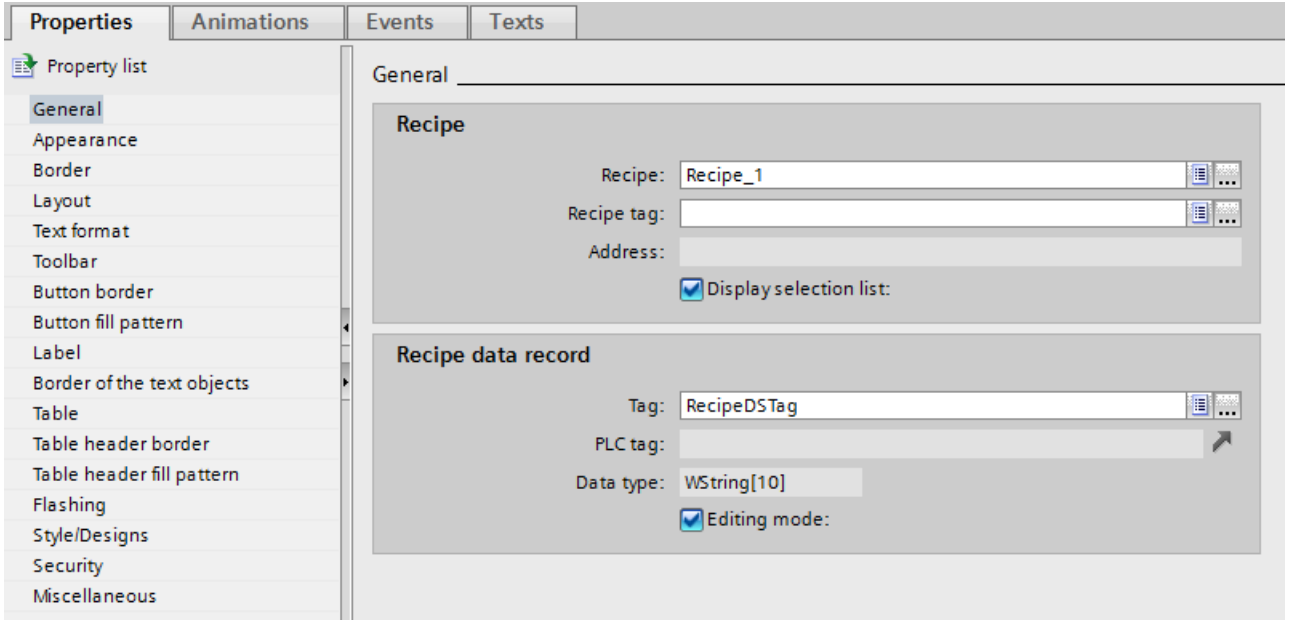
12.5.2.4 Configuration options of the advanced recipe view (V13 or higher) (Basic Panels, Panels, Comfort Panels, RT Advanced)

You can specify the behavior of the advanced recipe view in the Inspector window of the recipe view.

Displaying a recipe

To only allow access to the recipe data records of a specific recipe in a screen, follow these steps:

1. Enter the required recipe or select an existing recipe in the "Recipe" field under "Properties > General".
2. When a recipe is entered in the "Recipe" field, you will have to enable the "Selection list" if you want to display the recipe name in runtime.



The selected recipe appears in the recipe view.

Writing a recipe number or name and recipe data record number or name to a tag

Both the recipe and the recipe data record can each be configured to a tag in the recipe view. Depending on the "String" or "Int" data type of the tag, the name or number of the recipe or recipe data record is stored in the tag. Conversely, the tag can be used to select the recipe or recipe data record by entering the corresponding value. For example, the tag can be transferred as a parameter for a system function.

Follow these steps:

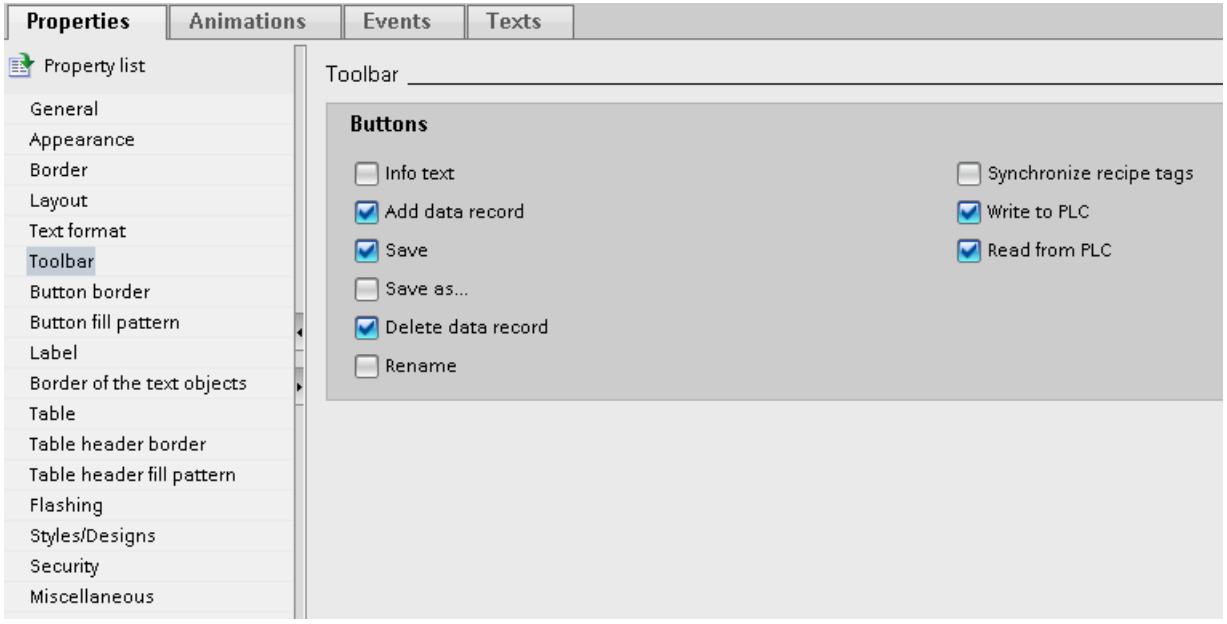
1. Enter a tag of the "String" type in the field "Recipe tag", under "Properties > General > Recipe".
2. Enter a tag of the "Int" type in the field "Tag", under "Properties > General > Recipe data record".

The name of the recipe and the number of the recipe data record are each stored in a tag.

Using the recipe view as a drop-down list

To use the recipe view as a selection field for recipes and recipe data records in a recipe screen, follow these steps:

1. Select the tag for the recipe name under "General > Recipe > Recipe tag".
2. Select the tag for the recipe data record name under "General > Recipe data record >Tag".
3. Disable the "Editing mode". You cannot create, rename, edit or delete recipe data.
4. In order to select recipes, enable "Display selection list" and make sure that no recipe is selected under "Properties > General > Recipe".
5. Deactivate all buttons under "Properties > Toolbar".



Configuring an event on the recipe view

To configure an event on the recipe view, follow these steps:

1. Select the recipe view that was added to the screen in the "Screens" editor.
   The properties of the recipe view are displayed in the Inspector window.
2. In the Inspector window, click the event you want to configure under "Properties > Events".
3. Configure a function list for the selected event.
   The function list is processed when the configured event occurs.

See also

Example of creating a recipe (Page 5609)

12.5.2.5 Behavior of the recipe view in Runtime

Screen change

If you change to another screen and have not yet saved changes to the recipe data in the recipe view, you will be prompted to save the recipe data. The recipe name and the name of the recipe data record are displayed to show which recipe data have not been saved yet.

Create, change, copy or delete recipe data records

If you attempt to create a new recipe data record and a recipe data record already exists, a system alarm will appear on screen.

Operating the recipe view with function keys

The Recipe view can be operated with function keys, e.g. if the HMI device does not have touch functionality. You can assign functions such as "SaveDataRecord" to the function keys on the HMI device.

Display after import of recipe data

Note

Availability

Import and export of recipe data is not available for Basic Panels and OP77A, TP177A (Portrait).

If you open the recipe view during the import of recipe data, only the recipe data that is already completely imported will be displayed. The recipe view is not automatically updated with a data import. In order to have a complete view of all the recipe data, do not open the recipe view until the system prompts you that the recipe data has been imported successfully. Alternatively, update the recipe view after successful completion of the import procedure.

Updating tag for recipes and recipe data records

Note

Availability

Tags for recipes and recipe data records are not available for Basic Panels and OP77A, TP177A (Portrait).

The current recipe data record or its number can be saved to a tag, depending on the configuration. The tag will be updated under the following conditions:

- The recipe data record has been loaded.
- The screen with the recipe view was not exited during loading.

This operation may take some time.

See also

Advanced recipe view (as of V13) (Page 5578)

12.5.2.6 Exporting and importing recipe data records (Panels, Comfort Panels, RT Advanced)

Introduction

You can export recipe data records to a file or import them from a file.

Recipe export and import depends on the configuration of the device and the device version.

List separators

A list separator is used to separate the data records during importing and exporting. The list separator used as the default depends on the setting for formats and numbers in the operating system.

Select "Start > Settings > Control Panel > Regional and Language Options".

If you want to import or export recipe data records, do not use this list separator in the display name of the recipe data records.

The following operating elements can, for example, be configured for using the export/import function:

- Selection field for the recipe
- Selection field for the recipe data record
- Operating element with the "ExportDataRecords" functionality
- Operating element with the "ImportDataRecords" functionality

Exporting a recipe data record

To export a recipe data record, proceed as follows:

1. Select the relevant recipe and recipe data record from the selection boxes.
2. Click the control element with the "ExportDataRecords" functionality.

Importing recipe data records

To import a recipe data record, proceed as follows:

1. Select the relevant recipe and recipe data record from the selection boxes.
2. Click the control element with the "ImportDataRecords" functionality.

Display after import of recipes

If you open the recipe view during the import of recipe data, only the recipe data that is already completely imported will be displayed. The recipe view is not automatically updated following a data import. In order to have a complete view of all the recipe data, do not open the recipe view until the system reports that the recipe data has been imported successfully.

Alternatively, you can update the recipe view after successful completion of the import process.

12.5.3 Configuring recipes

12.5.3.1 Creating and Editing Recipes

General configuration procedure

Carry out the following configuration steps when you create a new recipe:

Step	Description
1	Define the structure of the recipe.
2	Create tags according to the recipe structure. Assign process names to these tags.
3	Create the recipe.
4	Enter the required properties for the recipe: • Language-dependent view name of the recipe • "Coordinated transfer of data records" option Not for Basic Panels: • Recipe storage location • "Synchronize recipe view and recipe tags" option • "Manual transfer of individual modified values (teach-in mode)" option

Step	Description
5	Create the recipe elements and enter the required properties: • Language-dependent view name of the recipe elements • Tag binding of the recipe elements • Standard values and decimal places (power of ten) for the recipe elements
6	Create the recipe data records. Enter the language-specific display names for the recipe data records.
7	Configure a screen with recipe view or a recipe screen.

Note

Basic Panels and OP77A, TP177A (Portrait)

The selection of the storage location is not available for these devices. The recipes are always saved in the internal Flash memory.

Recipe tags cannot be used outside a recipe, e.g. not in I/O fields, not in alarms as trigger tags, not in systems functions as parameters, etc.

Note

Restrictions recipe view and recipe image

Only the simple recipe view is available in Basic Panels and OP77A, TP177A. Recipe images are not available in Basic Panels and OP73, OP77A, TP177A (Portrait).

Creating a new recipe

Introduction

To create a complete recipe, start by creating a new recipe, assign the corresponding recipe elements and then define the associated values in a recipe data record.

Note

Character strings that can be imported

Recipe data that contains semicolons or commas cannot be imported.

Only ever used character strings without semicolons or commas for recipes and recipe data records.

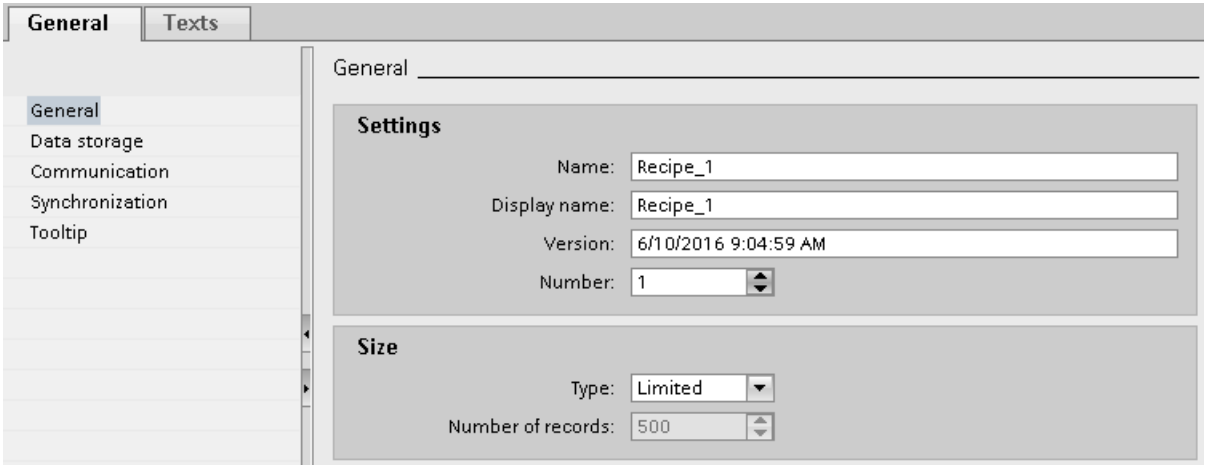
Requirement

- The tags for the recipe have been created.
- The "Recipes" editor is open.

Creating a recipe

Proceed as follows to create a recipe:

1. Click "Add" in the first free row of the table in the "Recipes" editor.
   The new recipe is created and displayed on a line.



2. Enter a descriptive name for the recipe under "Name" in the "General" area.
   This name identifies the recipe unambiguously within the project.
3. Select "Display name" to enter the language-specific name to be displayed in runtime.
4. Select a recipe number in "Number".
   The number identifies the recipe unambiguously within the HMI device.
   The recipe is automatically assigned a version that indicates the date and time of the last change. As an alternative, you can enter specific information relating to the recipe.
5. Specify the storage location for recipe data records in "Data medium". The options offered depend on the specific HMI device used.

   Note

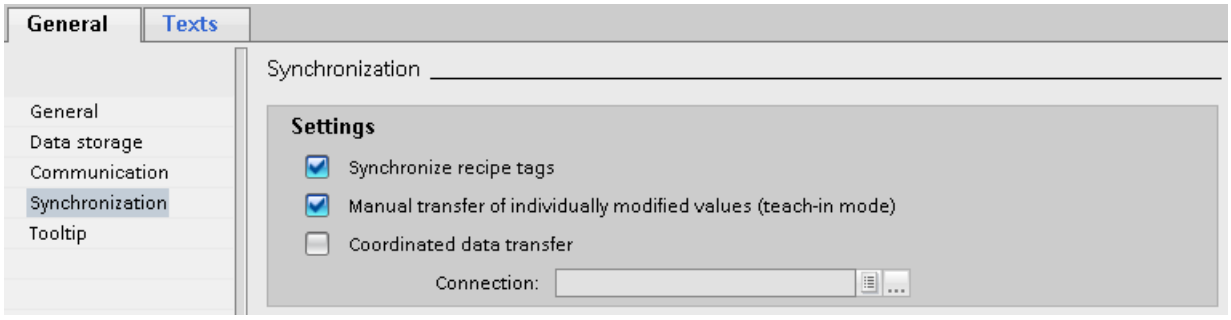
   Basic Panels and OP77A, TP177A (Portrait)

   The selection of the storage location is not available for these devices. The recipes are always saved in the internal Flash memory.

   Recipe tags cannot be used outside a recipe, e.g. not in I/O fields, not in alarms as trigger tags, not in systems functions as parameters, etc.

6. Enter a tooltip that is shown to the operator in runtime.

7. To compare recipe tags which are configured in I/O fields with the recipe view in Runtime, activate "Synchronize recipe view and recipe tags" in the Inspector window under "Properties > Synchronization".



Note

Basic Panels and OP77A, TP177A (Portrait)

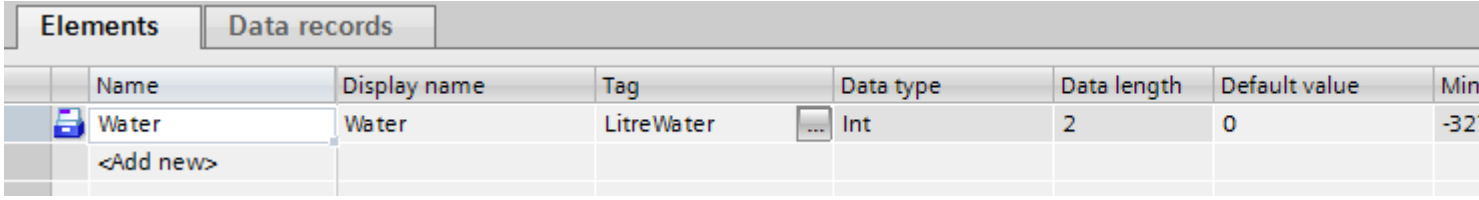
Because the recipe tags cannot be additionally used in I/O fields in screens for Basic Panels, the "Synchronize recipe view and recipe tags" is not available; you will also not be able to use the "Manual transfer of individual modified values (teach-in mode)" option.

8. Deactivate "Manual transfer of individual modified values (teach-in mode)" to specify that the recipe tags are automatically transferred to the PLC when editing the I/O fields.
9. Activate "Coordinated transfer of data records" to monitor the transfer of recipe data in Runtime using area pointers.
10. Select the appropriate connection to the PLC for coordinated transfer under "Synchronize with".

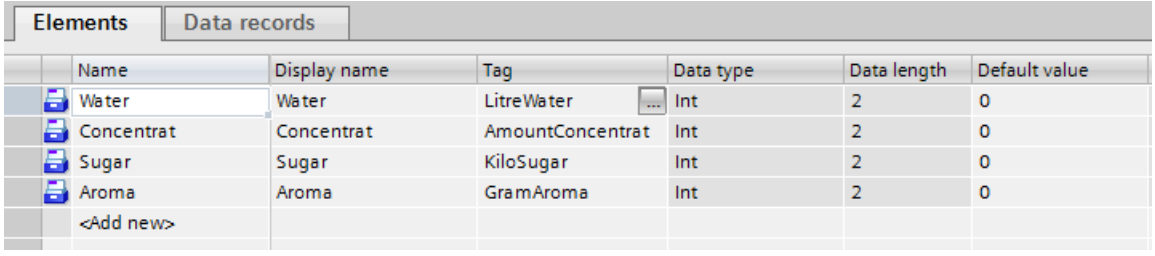
Create recipe element

To create recipe elements, proceed as follows:

1. Click the "Elements" tab.
2. Click "Add" in the first free line of the table editor.
   A new recipe element is created.
3. Enter a descriptive name for the element under "Name".
   The name identifies the element uniquely within the recipe.
4. Enter a language-specific display name for the element under "Display name".
   The display name appears in the recipe view, for example, in runtime.
5. Select the tag you want to link to the recipe element under "Tag".
   The value of the recipe data element is saved in Runtime in this tag, which is stored in a recipe data record.



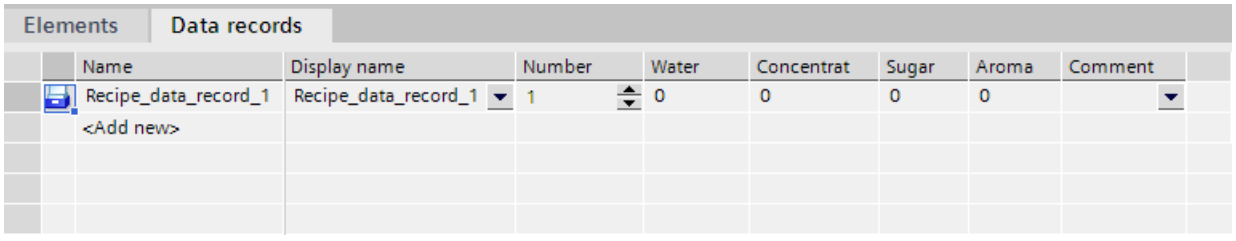
6. Enter a tooltip.
   The tooltip is shown to the operator in Runtime.
7. Under "Default value", enter the value that you want to use as the default entry when you create a new recipe data record.
8. To assign text to a value or range of values, select the relevant text list here. The assigned recipe tag must have the data type of a number. The tag value must be within the range of values of the text list.
   The text stored in the text list is displayed in an output field, for example, in Runtime.
9. Determine exactly how many places a decimal number is rounded to in the "Decimal places" column, e.g. 3 decimal places and vice versa by what power of ten an integer value is multiplied, e.g. 1,000.
   Examples for 3 decimal places: Entering "5" for a recipe element with the "Integer" data type gives the value "5000". Entering "5.6789" for a recipe element with the "Real" data type gives the value "5.679".
10. Create as many recipe entries as needed for the recipe. The maximum number of recipe entries possible depends on the HMI device being used.



Create recipe data record with known recipe values

To create recipe elements, proceed as follows:

1. Click the "Data records" tab.
2. Click "Add" in the first free line of the table editor.
   A new recipe data record is created. The recipe data record has a separate column for every recipe element created in the recipe.



3. Enter a descriptive name under "Name".
   The name identifies the data record uniquely within the recipe.
4. Enter a language-specific name under "Display name".
   The display name appears in the recipe view, for example, in runtime.
5. Enter a recipe data record number under "Number".
   The recipe data record number identifies the recipe data record uniquely within the recipe.

6. If you already know the recipe values at the configuration stage, you can enter the relevant value for each recipe element.

Elements | Data records

	Name	Display name	Number	Water	Concentrat	Sugar	Aroma	Comment
	Beverage	Beverage	1	30	70	45	600	
	<Add new>							

7. Create as many data records as you need for the recipe.

Elements | Data records

	Name	Display name	Number	Water	Concentrat	Sugar	Aroma	Comment
	Beverage	Beverage	1	30	70	45	600	
	Nectar	Nectar	2	50	50	10	300	
	Juice	Juice	3	5	95	3	100	
	<Add new>							

Enter the values in runtime

The following options are available for entering values in the recipe data records at runtime:

- Transfer data directly from the PLC (Teach-in mode)
- Import of values from a CSV file
- Input values on the HMI device

Note

Basic Panels and OP77A, TP177A (Portrait)

The import of values is not available for these devices.

Result

The complete recipe is configured.

Recipe data records with date or time stamp

If you use date or time data, make sure that the system setting for time and date on the configuring computer match those on the target system. Example: You load a recipe data record on the target system at 13:55 in which 14 h is stored as the processing time. If it is already 14:05 on the target computer, the recipe will not be processed. If an operator processes the recipe, change information will not written back correctly into the database.

After loading to the target system, check the recipes with date or time stamps on the target system.

Editing a recipe

Purpose

You want to modify, extend or delete parts of a recipe.

Requirement

- You have created at least one recipe.
- The "Recipes" editor is open.

Changing recipe settings

To change the recipe settings, proceed as follows:

1. Select the recipe that you want to change in the "Recipes" editor.
   The Inspector window opens.
2. Change the recipe configuration in the Inspector window.

You change recipe elements and recipe data records in the same way.

Change recipe values

To change recipe values, proceed as follows:

1. Select the recipe whose values you want to change.
2. Click the "Data records" tab.
3. Enter the new values in the value columns.

Adding a recipe element

To add more recipe elements to a recipe, proceed as follows:

1. Select the recipe to which you want to add more elements in the "Recipes" editor.
2. Click the "Elements" tab.
3. Click "Add" in the first free line.
   The recipe element is created.
4. Configure the recipe element.

You add recipe data records in the same way.

Managing recipes

Requirement

- You have created a recipe with recipe elements and recipe data record.
- The "Recipes" editor is open.

Renaming recipes

We distinguish between internal names and display names for recipes, recipe entries and recipe data records.

To rename recipe elements, proceed as follows:

1. Select the recipe that you want to rename.
   The Inspector window opens.
2. Select the "Rename" command from the shortcut menu.
3. Enter the new name.
   You rename recipe elements and recipe data records on the relevant tab in the same way.

Note

The view names in the "Recipes" editor can also be renamed under "Languages & Resources > Project Texts". This possibility is useful when you have already configured in several languages for example.

Copying and pasting recipes

To copy and paste recipes, proceed as follows:

1. Select the recipe that you want to copy.
2. Select the "Copy" command from the shortcut menu.
3. Select the "Paste" command from the shortcut menu in the first free table row.

The copied recipe is pasted into the table. The recipe elements and recipe data records are also copied in the appropriate tab with the recipe.

You also copy the recipe elements and recipe data records on the appropriate tab in the same way.

If a recipe data record of the same name already exists, the name of the copied recipe data record is extended by one digit. This ensures that the name is unique. Recipe data records can only be copied or pasted within the same recipe.

Deleting a recipe

To delete a recipe, proceed as follows:

1. Select the recipe that you want to delete.
2. Select the "Delete" command from the shortcut menu.
   The recipe is deleted.

You delete recipe elements and recipe data records on the relevant tab in the same way.

Note

When a recipe is deleted, the recipe data records contained in the recipe are also deleted.

Note

When you delete a recipe element, the associated values in the recipe data records are also deleted. The assigned tags are retained.

12.5.3.2 Configuring the display of recipes

Configuring the simple recipe view

Requirement

- The recipe has been created.
- The "Screens" editor is open.
- The screen has been created and opened.

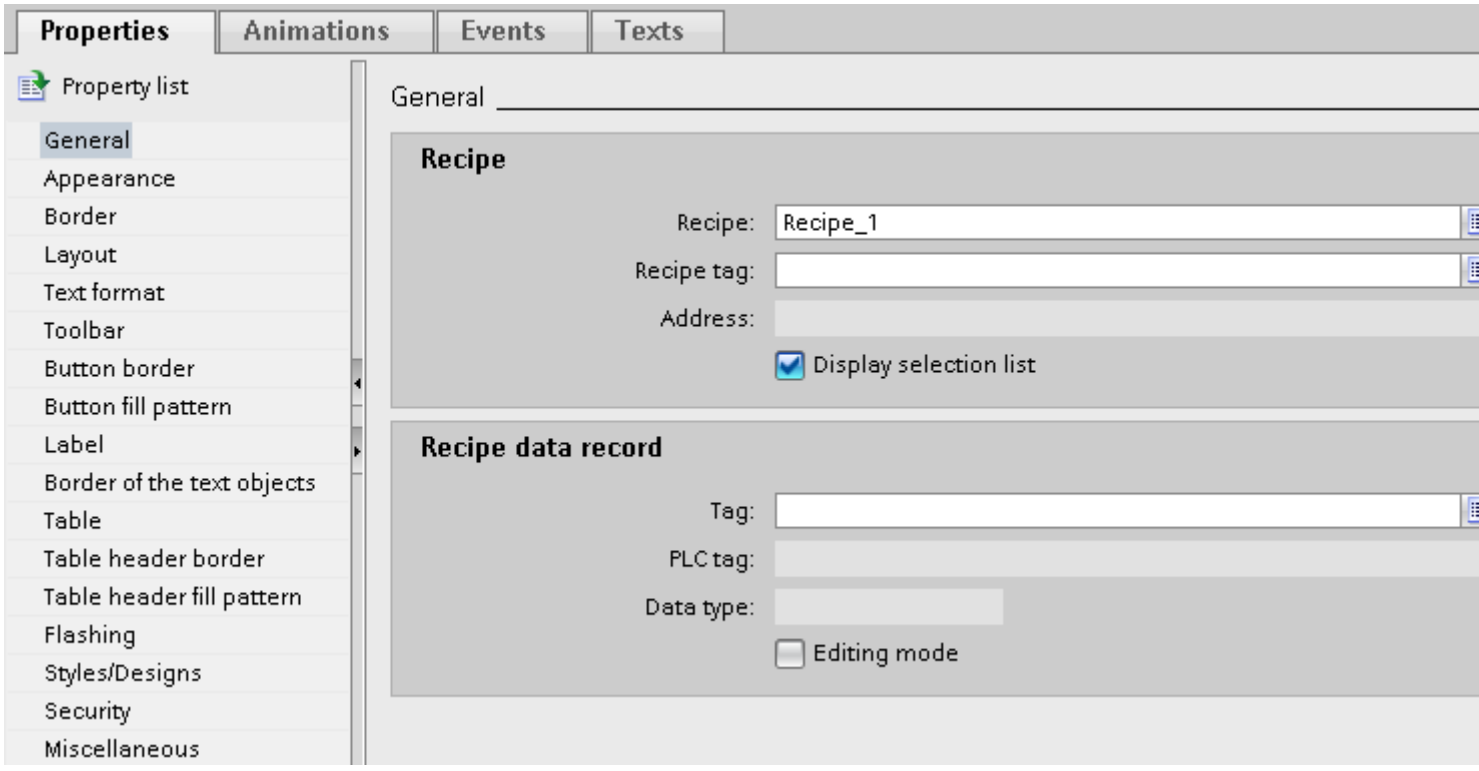
NOTICE
Data loss with several recipe views in the screen Applies only to Basic Panels, OP73, OP77A, TP177A and TP177A (Portrait): If two or more recipe views show the same recipe in a screen, you have a conflict when accessing the data. The result is data loss and unpredictable status of recipe data. Make sure the operators do not select and edit the same recipe in different recipe views. • Display only one recipe in a recipe view. • Display a different recipe in each recipe view.

Procedure

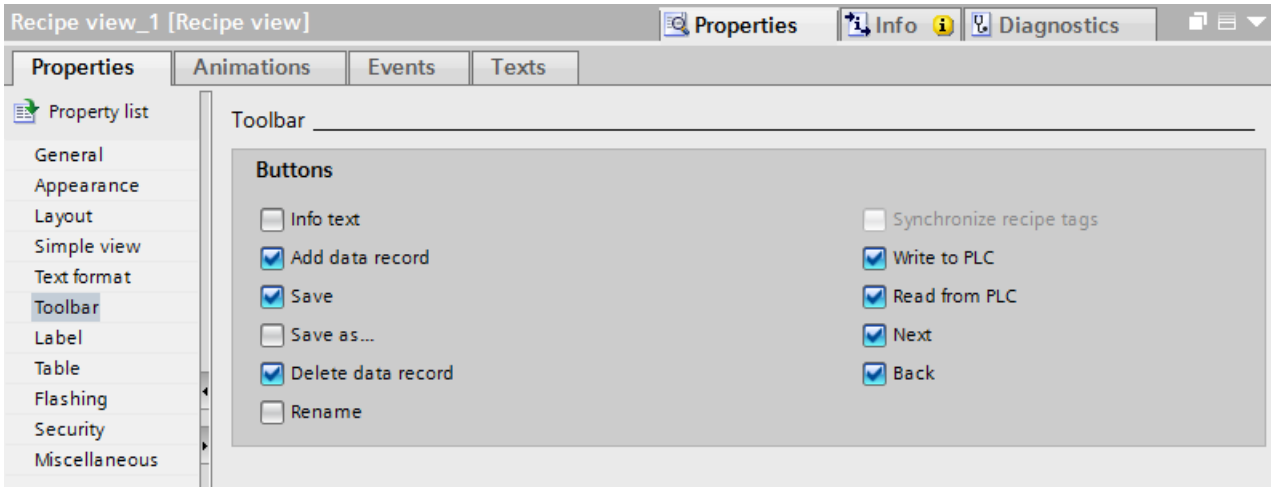
To configure a simple recipe view, proceed as follows:

1. Paste the recipe view into the screen. You will find the recipe view under "Controls" in the "Tools" task card.
2. Only in devices which also support the extended recipe view: Activate "Simple view" under "Properties > Display > Mode".

3. If you want to display only the recipe data records of a specific recipe in the recipe view, select the specific recipe under "Properties > General > Recipe".



4. If you only want to display the recipe data in the recipe view, deactivate "Processing mode" in the "Recipe data record" area.
5. You can define additional options for the recipe view under "Properties > Appearance" and "Properties > Layout".
6. Under "Properties" > Toolbar" specify which menu commands are available in the recipe view in Runtime.



Result

The simple recipe view is configured. You can use the recipe view to display and edit recipe data during runtime.

Deactivation of the editing mode in "Properties > Properties > General" has no impact on the toolbar icons. The buttons activated under "Properties > Toolbar" can still be used even if editing mode is disabled.

Configuring the Advanced Recipe View (Panels, Comfort Panels, RT Advanced)

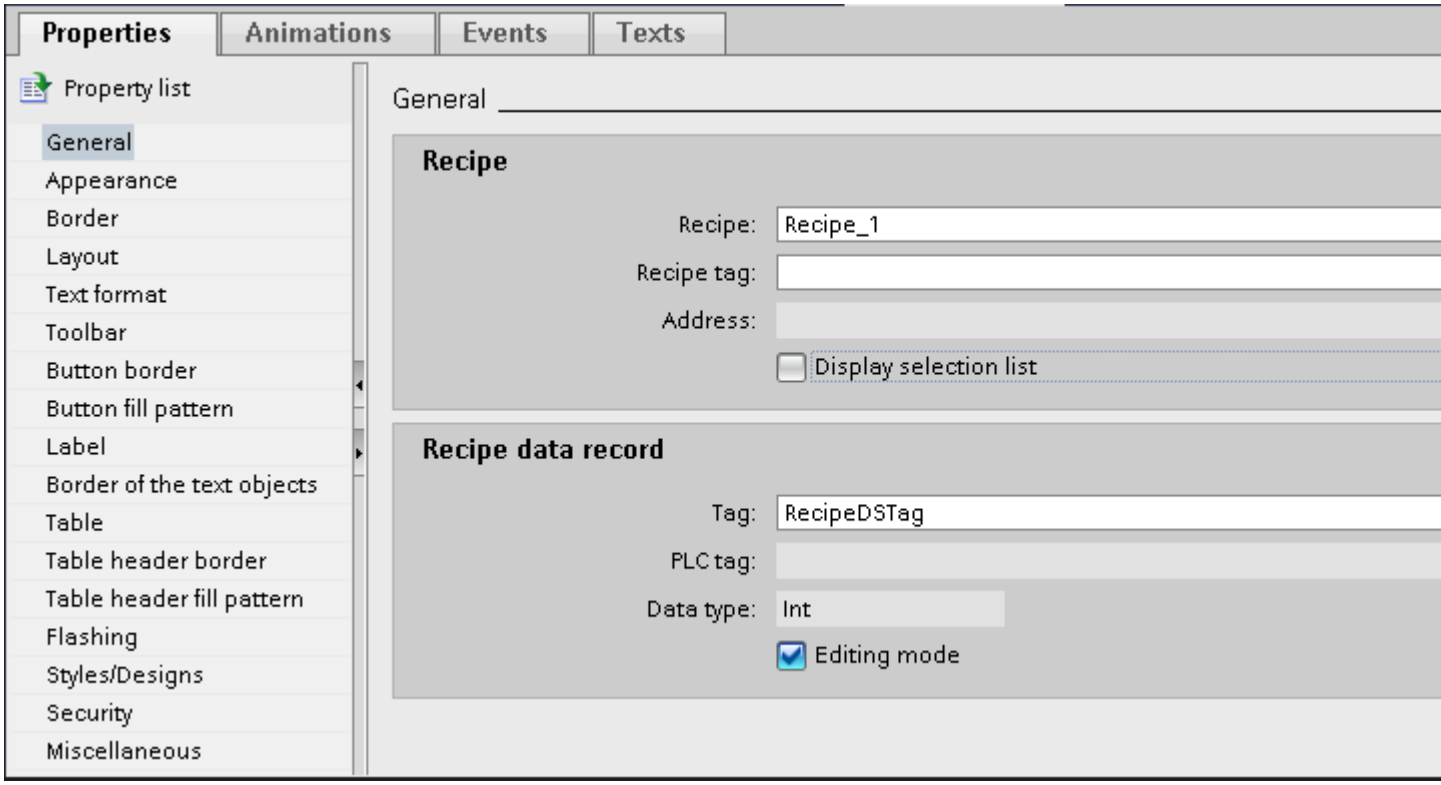
Requirement

- The recipe has been created.
- The "Screens" editor is open.
- The screen has been created and opened.

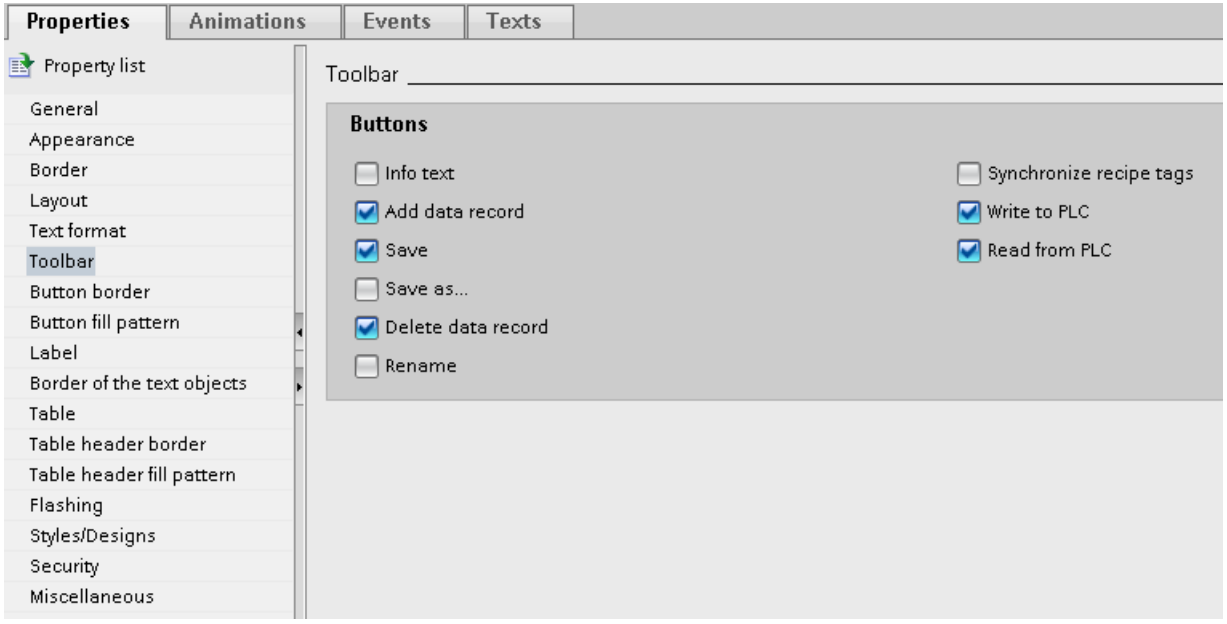
Procedure

To configure an advanced recipe view, proceed as follows:

1. Paste the recipe view into the screen. The recipe view is found in the task card "Tools" > "Controls".
2. Select "Properties > Display > Mode > Advanced view" in the Inspector window.



3. Select the required settings from the "General" group in the Inspector window.
   - If you want to display only the recipe data records for a particular recipe in the recipe view, select the recipe under "Recipe" in the "Recipes" area.
   - If you want to save the recipe name or the recipe number in a tag, select the tag under "Recipe tag" in the "Recipe" area.
   - If you want to save the recipe data record name or number in a tag, select the tag under "Tag" in the "Recipe data record" area.
   - If you only want to display the recipe data in the recipe view, deselect "Edit mode".
   - If you only want to use the recipe view to select recipes, disable "Display table" under "Properties > Table".
4. You can define additional options for the recipe view under "Properties > Appearance" and "Properties > Layout".
5. If you want to change the label in the recipe view, enter a suitable text under "Properties" > "Label".
6. Under "Properties" > Toolbar" specify which buttons are available in the recipe view in Runtime.



Note

If you select the "Edit" command in the context menu of the recipe view, the recipe view becomes active. To activate the recipe view, the zoom factor must be set to 100%.

You can set the column width and the position of the "Entry name" and "Value" columns in active mode.

Result

The recipe view is configured. You can use the recipe view to display and edit recipe data during runtime.

Deactivation of the editing mode in "Properties > Properties > General" has no impact on the toolbar icons. The buttons you activated in "Properties > Toolbar" can still be used even if editing mode is disabled.

12.5.4 Using recipes in Runtime

12.5.4.1 Using the simple recipe view

Description of the simple recipe view

Recipe view

The simple recipe view is a display and operating object that is used to manage recipe data records. The recipe view shows recipe data records in tabular form.

The buttons and information displayed in the columns are adjustable.

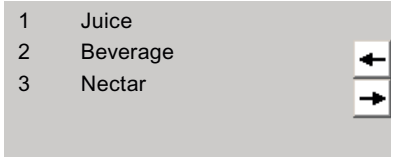
The values displayed or entered in the recipe view are saved in recipe data records. The recipe data record displayed can be written to the PLC and values read in from the PLC using buttons.

Layout of the display

The simple recipe view consists of the following display areas:

- Recipe list
- Data record list
- Element list

This application is illustrated below:



In the simple recipe view, each area is shown separately on the HMI device. You can use the shortcut menu to operate each of these display areas.

The simple recipe view always begins with the recipe list.

Operation

You have the following options for using the simple recipe view, according to the configuration:

- Create, change, copy or delete recipe data records
- Read recipe data records from the PLC or transfer to the PLC

Using the display area and shortcut menu

Toggle between the display areas and the shortcut menus to operate the simple recipe views.

Table 12-13 Operation of the display area

Button	Key	Function
	<Enter>	The next lowest display area is opened, i.e. the data record list or the element list.
	<Esc>	The previous display area opens.
	<INS>	Creates a new data record for the selected recipe if the list of recipes or recipe data records is displayed. Then changes to the list of recipe element. Requirement: "Properties >General > Processing mode" is activated. The button can be simulated with the "Key SimulateSystem-Key" function even on devices without keys.
	<DEL>	Deletes the selected recipe data record in the list of recipe data records. Requirement: "Properties >General > Processing mode" is activated.
	<Up>/<Down>	Selects the previous/next entry.
	<Pg Up>/<Pg Down>	Moves the display up or down one page.
	<Home>/<End>	Selects the first/last entry. The first/last entry is selected.

Table 12-14 Operation of the shortcut menu

Button	Key	Function
	<Right>	The shortcut menu of the display area opens.
	<Esc>	The menu is closed. The display area opens.
	Input of the number of the menu command	The menu command is executed.

Shortcut menus of the simple recipe view

You can click the  button in each display area to call up a selection of commands. The command selection lists those commands that are available in the current display area. A number is assigned to each command. The command is executed when you enter this number. Alternatively select the command and press the <Return> key.

Shortcut menus in the recipe list

Menu command	Function
New	A new recipe data record is created for the selected recipe. If a start value is configured, it is displayed in the input field.
Display tooltip	The tooltip configured for the recipe is displayed.
Open	The record list of the selected recipe opens.

Shortcut menus of the recipe data record list

Menu command	Function
New	Creates a new recipe data record. If a start value is configured, it is shown in the input field.
Delete	The displayed record is deleted.
Save as	The selected data record is saved under a different name. A dialog box opens where you can enter the name.
Rename	Renames the selected data record. A dialog box opens where you can enter the name.
Open	The element list of the selected data record opens.
Previous	The recipe list opens.

Shortcut menus of the recipe element list

Menu command	Function
Save	The selected data record with the recipe element is saved.
To PLC	The displayed values of the selected data record are transferred from the HMI device to the PLC.
From PLC	The recipe values from the PLC are displayed in the recipe view of the HMI device.
Save as	The data record is saved under a new name. A dialog box opens where you can enter the name.
Display tooltip	The tooltip configured for the recipe element is displayed.
Rename	The selected recipe element is renamed. A dialog box opens where you can enter the name.
Previous	The data record list opens.

Shortcut menus in the data record list

Note

HMI device dependency

The following menu commands are configured in Basic Panels and in OP 77A, TP 177A, TP 177A (Portrait) and TP 177B.

Menu command	Function
To PLC	The displayed values of the selected data record are transferred from the HMI device to the PLC.
From PLC	The recipe values from the PLC are displayed in the recipe view of the HMI device.

Updating and displaying values

Note

Processed recipe data record is changed in the background

Only applies for Basic Panels: If an operator has changed a recipe data record and a PLC job wants to read or write any recipe data record of this recipe, the PLC job is stopped and a system alarm is output. On the other hand, the changed value is displayed immediately if only the PLC job and no operator has changed recipe data.

Does not apply for Basic Panels: If an operator has changed a recipe data record and a PLC job has changed values of the recipe data record concerned, the recipe view is not updated automatically. To update the recipe view, reselect the relevant recipe data record.

Using the simple recipe view

Controlling the simple recipe view with mouse or touchpad

To control the simple recipe view with mouse or touchpad, proceed as follows:

1. Select the desired recipe from the recipe view.
2. Click the → button.
   The shortcut menu is opened.
3. Select the desired menu command.
   The menu command is executed.

Controlling the simple recipe view with the keyboard

To control the simple recipe view with the keyboard, proceed as follows:

1. Press the <Tab> key until the simple recipe view is selected.
2. Select the desired recipe with the cursor keys.
3. Press <Right>.
   The shortcut menu is opened.
4. Press the <Down> key until the desired menu command is selected.
5. Press <Enter> to confirm the command.

Key shortcuts for the simple recipe view

The following key shortcuts are activated for the simple recipe view in Runtime if "Activate keyboard operation" is enabled in the ES.

Key shortcut	Effect	Menu command
<Insert>	Generates a new recipe data record	New
<Del>	Deletes the recipe data record displayed.	Delete

Operating the recipe view with function keys

The recipe view can be operated with function keys, for example if the HMI device does not have touch functionality. You can assign functions such as "SaveDataRecord" to the function keys on the HMI device.

Managing recipe data records

Administration of recipe data records

You have the following options for managing the simple recipe view, according to the configuration:

- Creating new recipe data records
- Copy recipe data records
- Edit recipe data records
- Delete recipe data records

Creating new recipe data records

To create a new recipe data record, proceed as follows:

1. Select the recipe on the HMI device in which you want to create a new recipe data record.
2. Select the "New" command from the shortcut menu for the recipe list.
   A new data record with the next available number will be created.
   The element list of the new recipe data record opens.
3. Enter values for the elements of the recipe data record.
   The configuration data may already contain default values for the recipe data record.
4. Select the "Save" command from the shortcut menu for the element list.
   The dialog "Save as" opens.
5. Enter the name and number of the recipe data record.
6. Click the "OK" button.
   The data record is saved under the new name.
   If the recipe data record already exists, a dialog is opened. In this dialog, specify whether the existing data record is to be overwritten.

Copying a recipe data record

To copy a recipe data record, proceed as follows:

1. Select the recipe on the HMI device in which you want to copy an existing recipe data record.
2. On the HMI device, select the recipe data record of which you want to save a copy.
3. Select the "Save As" command from the shortcut menu for the data record list.
   The dialog "Save as" opens. The recipe data record is automatically given the next free recipe data record number.
4. Under name, enter the name of the record.
5. Click the "OK" button.

Modifying a recipe data record

To change a recipe data record, proceed as follows:

1. Select the recipe on the HMI device in which you want to edit an existing recipe data record.
2. Select the recipe data record that you want to edit on the HMI device.
3. Select the recipe data record.
   The element list of the recipe data record is displayed.
4. Replace the old values with new ones.
5. Select the "Save" command from the shortcut menu for the element list.

Deleting a recipe data record

To delete a recipe data record, proceed as follows:

1. Select the recipe on the HMI device from which you want to delete an existing recipe data record.
2. Select the recipe data record that you want to delete on the HMI device.
3. Select the "Delete" command from the shortcut menu for the data record list.
4. Confirm this security prompt to delete the data record.

Read recipe data record from PLC

Introduction

In Runtime, you can change values directly in the plant that are also stored in recipes in the HMI device. This applies if a valve was opened further directly in the plant than was specified in the recipe. The values of the recipe data records saved in the HMI device possibly no longer match the values in the PLC.

You can read the values of the recipe tags from the PLC and write them to a recipe data record.

The read values are written to the recipe data record that is currently displayed on the HMI device.

Procedure

To read a recipe data record from the PLC, proceed as follows:

1. Open the recipe on the HMI device.
   The data record list opens.
2. Select the element list of the recipe data record to which you want to apply the values from the PLC.
3. Select the "From PLC" command from the shortcut menu for the element list.
   The values are read from the PLC and displayed in the current recipe data record.
4. If you want to save the values, select the "Save" or "Save As" command.

Result

The values are read from the PLC, visualized on the HMI device and saved to the recipe data record.

Note

Basic Panels

With Basic Panels, the "From PLC" menu command can also be configured for the data record list. In this case, you can also select the "From PLC" menu command in the data record list.

Transferring a recipe data record to the PLC

Introduction

For the values of a data record that was changed in the recipe view to take effect, you must transfer the values to the PLC.

The values displayed in the recipe view are always transferred to the PLC.

Procedure

To transfer a recipe data record to the PLC, proceed as follows:

1. Open the recipe you want to use.
   The data record list opens.
2. Select the element list of the recipe data record whose values you want to transfer to the PLC.
3. Select the "To PLC" command from the shortcut menu for the element list.

Result

The values of the recipe data record are transferred to the PLC.

Note

Basic Panels

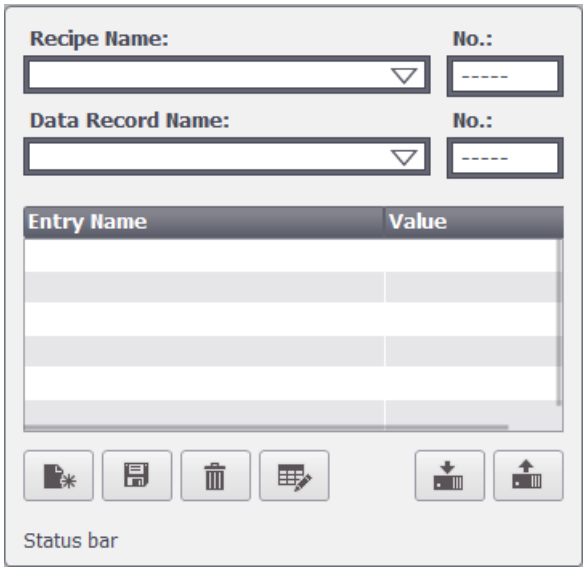
With Basic Panels, the "From PLC" menu command can also be configured for the data record list. In this case, you can also select the "From PLC" menu command in the data record list.

12.5.4.2 Using the advanced recipe view

Description of the advanced recipe view

Application

The recipe view is used to display, edit and manage data records.



Operation

Depending on the configuration you can:

- Create, change, copy or delete recipe data records
- Synchronizing recipe data records with the associated recipe tags
- Read recipe data records from or transfer to the PLC

Operating elements

The following operating elements can be configured in the recipe view:

Operating element	Key combination	Function
		The configured tooltip is displayed.
	<Ctrl+Space Bar>	Creates a new recipe data record. If a start value is configured, it is shown in the input field.
	<Ctrl+Enter>	Saves the displayed values of the recipe data record. The storage location is predefined by the project.
	<Ctrl+*>	The recipe data record is saved under a different name regardless of the recipe view. A dialog box opens where you can enter the name.
	<Ctrl+Del>	The displayed recipe data record is deleted.
	<Ctrl+Shift+T>	The selected recipe data record is renamed. A dialog box opens where you can enter the new name.
	<Ctrl+=>	The system always updates the current value of the recipe view with the up-to-date recipe tag value. When the value shown in the recipe view is more recent than the current recipe tag value, the system writes this value to the recipe tag. "Synchronize recipe view and recipe tags" must be activated in the recipe properties before this function can be used.
	<Ctrl+Down>	The values of the set recipe data record displayed in the recipe view are transferred to the PLC.
	<Ctrl+Up>	The recipe values from the PLC are displayed in the recipe view.

See also

Managing recipe data records (Page 5605)

Synchronizing recipe data record (Page 5607)

Read recipe data record from PLC (Page 5608)

Transferring a recipe data record to the PLC (Page 5608)

Using the advanced recipe view

Introduction

You can control the recipe view using both mouse or touchpad or with the keyboard.

Controlling the recipe view with mouse or touchpad

To control the recipe view with mouse or touchpad, proceed as follows:

1. Select the recipe you want to use.
   The records for the recipe are displayed.
2. Click on the data record you wish to edit.
3. Click the button whose function you want to run.

Controlling the recipe view with the keyboard

To control the recipe view with the keyboard, proceed as follows:

1. Press the <Tab> key until the cursor reaches the field for selecting the recipe.
2. Press <Enter>.
   The drop-down list box for the recipes opens.
3. Select a recipe. You navigate between the next or previous entry in the list by using the cursor keys <Left>, <Right>, <Up> and <Down>.
4. Select a data record.
5. Press the <Tab> key until the operator control object you wish to use is selected.

Alternatively you can control the recipe view using certain key combinations.

Operating the recipe view with function keys

The recipe view can be operated with function keys, for example if the HMI device does not have touch functionality. You can assign functions such as "SaveDataRecord" to the function keys on the HMI device.

Key shortcuts for the advanced recipe view

The following key shortcuts are activated for the advanced recipe view in Runtime if "Activate keyboard operation" is enabled in the ES.

Keys shortcut	Effect	Menu command	Button
<HELP>	Calls up the configured tooltip for the selected recipe.	Tooltip	
<Ctrl+Space-bar>	Generates a new recipe data record. Any configured start value is displayed in the input field.	Add data record	
<Ctrl+Enter>	Saves the modified record with its current name.	Save	
<Ctrl+*>	Saves the modified record with a new name.	Save as	
<Ctrl+Del>	Deletes the recipe data record displayed.	Delete data record	
<Ctrl+Shift +T>	Changes the name of selected data record.	Rename data record	

Keys shortcut	Effect	Menu command	Button
<Ctrl+=>	Compares the values of the selected data record with the values on the PLC. Any value in the recipe view which is more recent compared to the current recipe tag value is written to the recipe tag. This function is only available if enabled in the ES.	Synchronizing recipe view and recipe tags	
<Ctrl+Down Arrow>	Sends the current value to the PLC.	Write to PLC	
<Ctrl+Up Arrow>	Reads the current value from the PLC.	Read from PLC	

Screen change

If you change to another screen and have not yet saved changes to the recipe data in the recipe view, you will be prompted to save the recipe data. The recipe name and the name of the recipe data record are displayed to show which recipe data have not been saved yet.

Managing recipe data records

Administration of recipe data records

You have the following options for managing recipe data records, according to the configuration:

- Creating new recipe data records
- Copy recipe data records
- Edit recipe data records
- Delete recipe data records
- Rename recipe data records

Creating new recipe data records

To create a new recipe data record, proceed as follows:

1. Select the recipe on the HMI device in which you want to create a new recipe data record.
2. Click  or press <Ctrl+Spacebar>.
   A new recipe data record with the next available number is created.
   If you change the new data record number to an existing data record number, the existing data record is overwritten.
3. Enter values for the elements of the data record.
   The elements of the recipe data record can be assigned default values depending on the configuration.
4. Enter a name for the data record.

5. Click on the button  ess the keyboard shortcut <Ctrl+Enter>.
   Alternatively, click on the  button to save the data record or press the keyboard shortcut <Ctrl+*>.
   The "Save as" dialog opens. Enter the name for the data record and confirm your entries with "OK".
6. The data record is saved under the new name.
   If the recipe data record already exists, a dialog is opened. In this dialog, specify whether the existing data record is to be overwritten.

Copying a recipe data record

To copy a recipe data record, proceed as follows:

1. Select the recipe on the HMI device in which you want to copy an existing recipe data record.
2. Select the recipe data record that you want to copy on the HMI device.
3. Click  in the recipe view or press <Ctrl+*>.
   The dialog "Save as" opens.
4. Enter a name for the data record.
5. Click "OK" to confirm your input.

Modifying a recipe data record

To change a recipe data record, proceed as follows:

1. Select the recipe on the HMI device in which you want to edit an existing recipe data record.
2. Select the recipe data record that you want to edit on the HMI device.
3. Replace the old values with new ones.
4. Click  in the recipe view or press <Ctrl+Enter>.

Deleting a recipe data record

To delete a recipe data record, proceed as follows:

1. Select the recipe on the HMI device in which you want to delete an existing recipe data record.
2. Select the recipe data record that you want to delete on the HMI device.
3. Click  in the recipe view or press <Ctrl+Del>.

Renaming recipe data records

Proceed as follows to rename a recipe data record:

1. Select the recipe on the HMI device in which you want to rename an existing recipe data record.
2. Select the recipe data record that you want to rename on the HMI device.

3. Click  in the recipe view or press <Ctrl+Shift+T>.
4. Enter a new name in the dialog window.

See also

Description of the advanced recipe view (Page 5602)

Synchronizing recipe data record

Introduction

Differences between the following values may occur during runtime:

- The values displayed in the recipe view
- The actual values of the recipe tags

Depending on the configuration, the values displayed in the recipe view are synchronized with the recipe tags. Synchronization encompasses all tags of a recipe data record.

Note

Changed tag name

The tag and the value of the recipe data record cannot be associated if you have renamed the tag you want to synchronize. The tags in question are not synchronized.

Note

Recipe tags can only be synchronized in the advanced recipe view.

Requirement

- A recipe data record is displayed in the recipe view.
- The values of recipe tags were modified by teaching, for example.

Procedure

To synchronize a recipe data record, proceed as follows:

- Click  in the recipe view or press <Ctrl+=>.

The system always updates the current value of the recipe view with the up-to-date recipe tag value.

When the value shown in the recipe view is more recent than the current recipe tag value, the system writes this value to the recipe tag.

See also

Description of the advanced recipe view (Page 5602)

Read recipe data record from PLC

Introduction

In runtime, you can change values directly in the plant that are also stored in recipes in the HMI device. This applies for example if a valve was opened right at the plant wider than specified in the recipe. The values of the recipe data records saved in the HMI device may then possibly no longer match the values in the PLC.

You can read the values of the recipe tags from the PLC and write them to a recipe data record.

The values read are written to the recipe data record that is currently displayed on the HMI device.

Procedure

Proceed as follows to read a recipe data record from the PLC:

1. Select the recipe on the HMI device.
2. On the HMI device, select the recipe data record whose values you want to read from the PLC.
3. Click  in the recipe view or press <Ctrl+Up Arrow>.

See also

Description of the advanced recipe view (Page 5602)

Transferring a recipe data record to the PLC

Introduction

For the values of a data record that was changed in the recipe view to take effect, you must transfer the values to the PLC.

The values displayed in the recipe view are always transferred to the PLC.

Procedure

Proceed as follows to transfer a recipe data record to the PLC:

1. Select the recipe on the HMI device.
2. On the HMI device, select the recipe data record whose values you want to transfer to the PLC.
3. Click  in the recipe view or press <Ctrl+Down Arrow>.

See also

Description of the advanced recipe view (Page 5602)

12.5.5 Example

12.5.5.1 Example of creating a recipe (Basic Panels, Panels, Comfort Panels, RT Advanced)

Task

In this example, you create three recipes for a fruit juice mixing machine. The fruit juice mixing machine produces drinks with "orange", "apple" and "tropical" flavors. You create a recipe for each flavor.

Each recipe contains a recipe data record for the following mixing ratios:

- Beverage
- Nectar
- Juice

Settings

The settings relate to an HMI device which is connected to a SIMATIC S7-300 or SIMATIC S7-400.

In this example, you will need the following tags, recipes, recipe entries and recipe data records:

Tags:

Name	PLC connection	Address	Type
Liter water	Yes	DB 120, DBW 0	Integer
Liter concentrate	Yes	DB 120, DBW 4	Integer
Kilo sugar	Yes	DB 120, DBW 8	Integer
Gram flavoring	Yes	DB 120, DBW 12	Integer

Recipes:

- Orange
- Apple
- Tropical

Recipe entries:

Recipe element	Associated tag
Liter water	Liter water
Liter concentrate	Liter concentrate
Kilo sugar	Kilo sugar
Gram flavoring	Gram flavoring

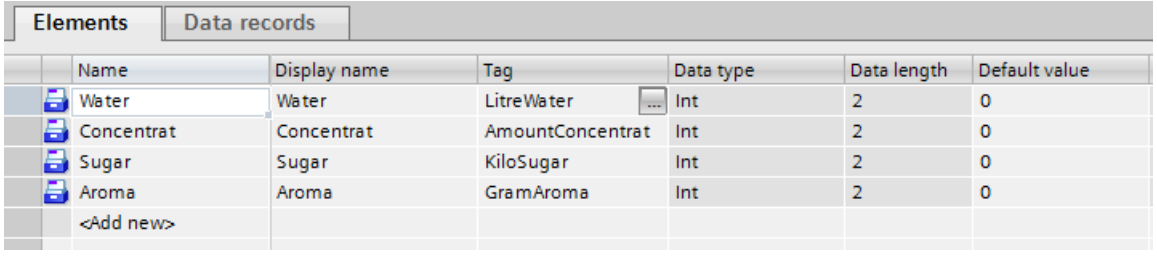
Recipe data records for drink, nectar and juice:

Data record name	Liter water	Liter concentrate	Kilo sugar	Gram flavoring
Beverage	30	70	45	600
Nectar	50	50	10	300
Juice	5	95	3	100

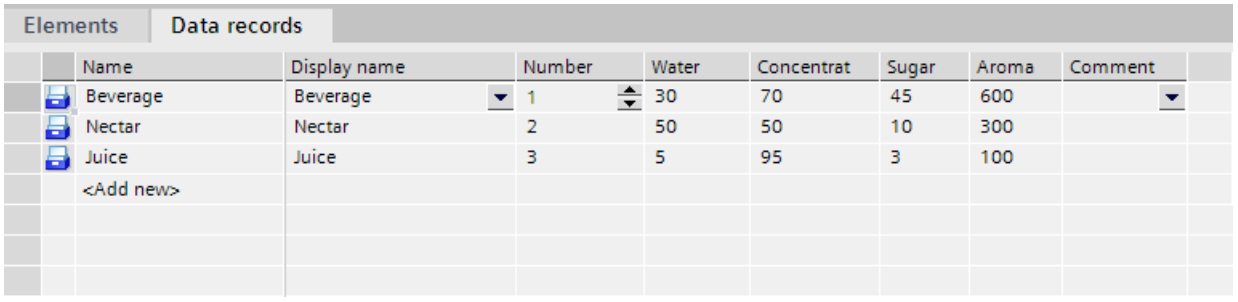
Procedure

To create a recipe, proceed as follows:

1. Create the following tags with the settings specified above: "LiterWater", "LiterConcentrate", "KiloSugar" and "GramFlavoring".
2. Create the "Orange", "Apple" and "Tropical" recipes with the settings indicated above. Create the recipe entries in each recipe.



3. Not for Basic Panels: Configure each recipe so that you can synchronize the recipe data records between the recipe screen and recipe view. The values of the recipe tags should not be transferred automatically to the PLC.
   You will have to make the following settings in the Properties dialog for the recipe concerned: Under "Properties > Options":
   - Activate the "Synchronize recipe view and recipe tags" option.
   - Activate the "Manual transfer of individual modified values (teach-in mode)" option.
4. Create the data records indicated above in each recipe. Enter the values indicated above in each of the data records.



Result

The "Orange", "Apple" and "Tropical" recipes have been created.

See also

Configuration options of the advanced recipe view (V13 or higher) (Page 5579)

12.6 Configuring user administration

12.6.1 Basics

12.6.1.1 Field of application of the user administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Principle

The access protection controls access to data and functions in Runtime. This feature protects your applications against unauthorized operation. Safety-related operations are already limited to specific user groups when a project is being created. To this purpose you set up users and user groups that you equip with characteristic access rights, so-called authorizations. You then configure the authorizations required for operation of safety-related objects. Operators only have access, for example, to specific operator controls. Commissioners, for example, have unlimited access in Runtime.

Definition

You administer users, user groups and authorizations centrally in the user administration of WinCC. You transfer users and user groups together with the project to the HMI device. The users and passwords are managed on the HMI device in the User view.

Application example

You configure the "Service" authorization so that only service technicians have access to the configuration parameters. You assign the authorization to the "Service technician" user group. This allows all members of this group to set the protected configuration parameters.

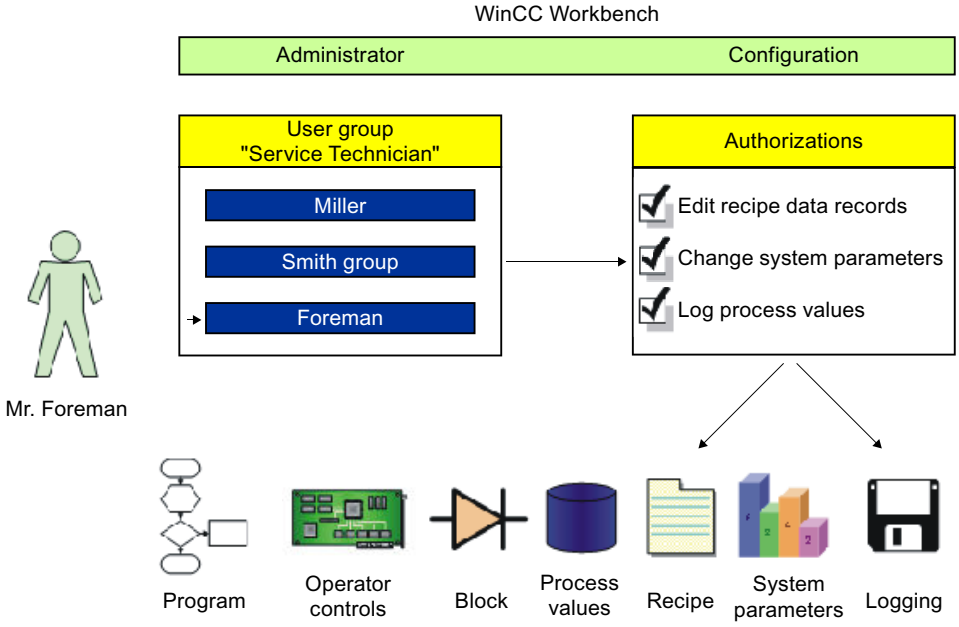
NOTICE
Access protection does not protect against incorrect operations. It is your job to ensure that only authorized personnel with appropriate training will design, commission, operate and maintain plants and machines. Access protection is not suitable for defining work routines and monitoring their observance.

12.6.1.2 Form of the user administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

In case of a project in manufacturing engineering, the environment at the equipment manufacturer has to be differentiated from the environment at the end customer as plant operator.

The equipment manufacturer allows users, for example Mr. Foreman, a specific access within the application or HMI device. However, a user Foreman does not exist at the end customer. The machine manufacturer cannot know the end users and the tasks they have to perform for configuration. The final users are usually set after commissioning at the end customer.



Principle

To minimize the work required for management, authorizations are assigned via user groups and not directly to individual users.

A user group assembles configured authorizations according to common jobs. For example, all permissions required for a service job are collected in a "Service technician" group. When you create a user who should be responsible for servicing, you simply assign him to the "Service technician" group.

The user view enables user administration in Runtime. Use user view to create, delete and assign an authorization to users in Runtime.

The user administration separates the administration of the users from the configuration of the authorizations. This ensures flexibility at the access protection.

Defaults can be set for the user administration during the configuration phase in the Engineering System.

12.6.1.3 Users (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You can create users in the "Users" tab of the "User administration" editor and assign them to user groups. The "Users" tab is part of the user administration in WinCC.

Open

To open the "Users" tab, double-click "User administration" in the project window.

Work area

The users are managed in the work area:

- You create or delete users.
- You assign users to user groups.

Note

You can assign a user to exactly one user group.

Note

On an HMI device the number is limited to 100 users and one PLC user. This restriction does not apply to PCs. On a PC, the maximum number of users is restricted by the physical memory.

Inspector window

When you select a user, you can change the password in the "General" group. Under "Automatic logoff" you can specify if the user is to be automatically logged off by the HMI device when there is no operator activity after the specified time.

Backup and restore

The user data is encrypted and saved on the HMI device to protect it from loss due to power failure.

The users, passwords, group assignments and logoff times set up on the HMI device can be backed up and restored. This prevents you having to enter all of the data again on another HMI device.

Note

The currently valid user data is overwritten in the following cases:

- Depending on settings, the next time the project is downloaded
- Upon restore of a backed-up project
- Upon import of the user administration via an operator control. The newly downloaded or restored user data and passwords take effect immediately.

Note

The user data from earlier Runtime versions can be backed up and restored in the V14.

The user data of the V14 cannot be restored to earlier versions.

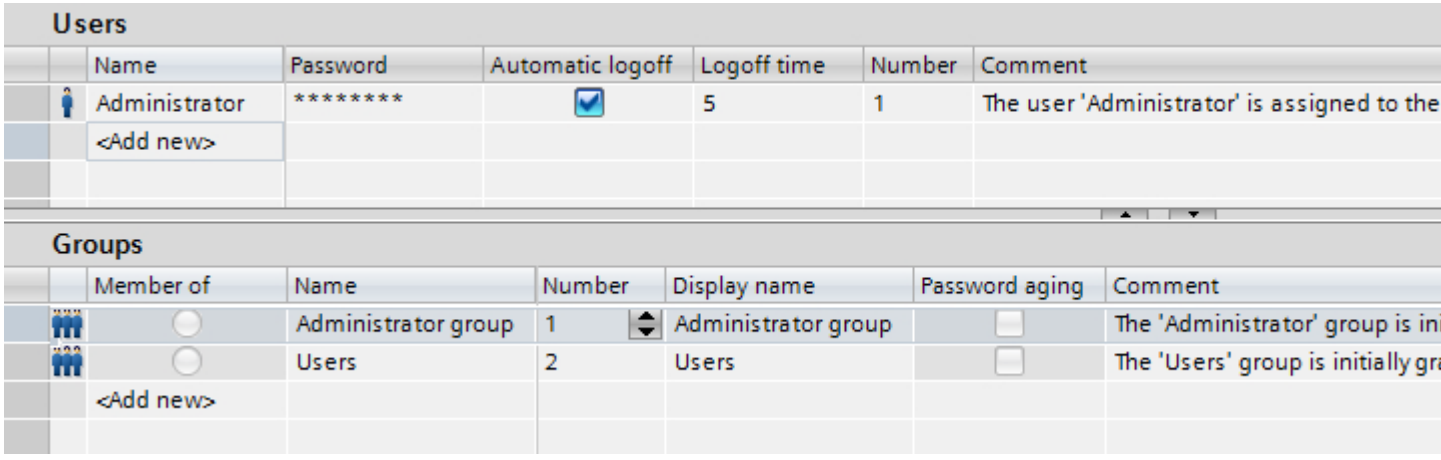
12.6.1.4 Users work area (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The "Users" work area lists the users and user groups in table form. You administrate the users and assign them to a user group.

Principle

The work area consists of the "Users" and "Groups" tables.



The "Users" table shows the existing users. When you select a user in this table, the "Groups" table shows the user group to which the user belongs.

Note

For the user "Administrator", the default password is "administrator". For security reasons, you should change the password of this user.

12.6.1.5 User groups (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You can create users and authorizations in the "User groups" tab of the "User Administration" editor. The "User groups" tab is part of the user administration in WinCC.

Open

Double-click the "User administration" in the project window. Open the "User groups" tab.

Work area

The user groups and authorizations are managed in the work area:

- You create new user groups and authorizations or delete them.
- You assign the authorizations to the user groups.

Inspector window

When a user group or an authorization is selected, you can edit the name in the "General" group. You can also enter a brief description in the "Comment" group.

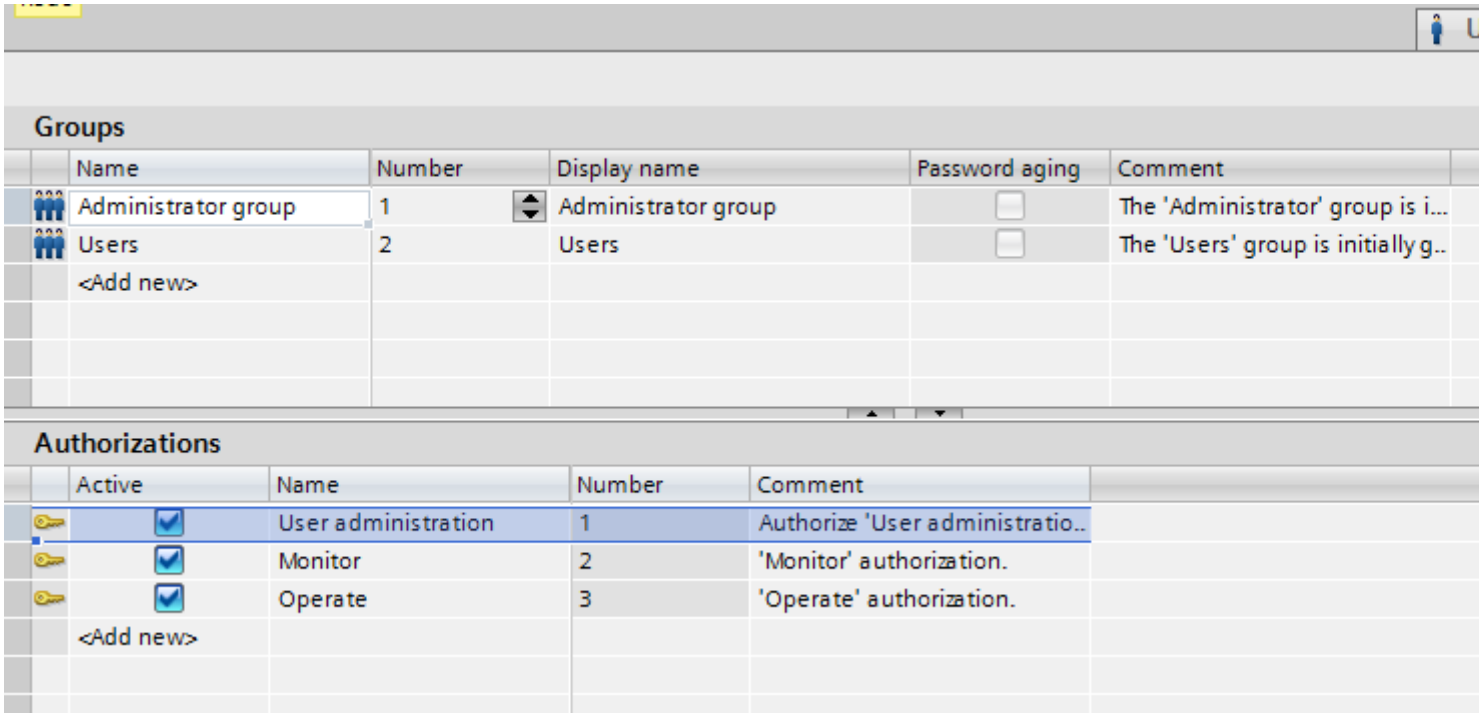
12.6.1.6 User groups work area (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The "User groups" work area shows a table of the groups and their authorizations. You administer the user groups and assign authorizations to them.

Principle

The work area consists of the "Groups" and "Authorizations" tables.



The "Groups" table shows the existing user groups. When you select a user group in this table, the "Active" column of the "Authorizations" table shows the authorizations which were assigned to the user group.

The number of the user group and of the authorization is assigned by the user administration. The designations and descriptions are assigned by you.

The number of predefined authorizations are fixed. Authorizations that you create can be freely edited. Ensure that the assigned numbers are unique.

12.6.1.7 Settings for the user administration (Basic Panels)

Introduction

In the "Runtime settings > User administration" editor, configure the security settings for users and their passwords in runtime.

Open

Double-click the "Runtime settings" editor in the project window. Click "User administration".

Work area

You carry out the settings for the validity of passwords in runtime in the work area. You determine the complexity of the password, for example.

Note

If the password from a previous project no longer corresponds to the policy, the user is prompted to enter a new password.

Effects in Runtime

The security settings have the following effects in runtime, depending on the configuration.

- "General" group
  - "Enable limit for logon attempts" check box selected
    The number entered in the "Number of incorrect logon attempts" box defines the number of logon attempts a user is allowed before being assigned to the "Unauthorized" group.
    "Enable limit for logon attempts" check box not selected
    The user has an unlimited number of logon attempts in runtime.
  - "Number of incorrect logon attempts" field
    If you enter "4" in the field, for example, and the fourth logon attempt fails, the user is automatically assigned to the "Unauthorized" group.
    You can specify 1 to 9 attempts.
  - "Logon only with password" check box
    When the check box is selected, the user will be authenticated by the password. The user name is not required.
    To match users to passwords, you cannot configure passwords more than once.

    Note

    If the "Logon only with password" option is enabled, the system sets the passwords for users that have already been created. It is therefore recommended to always enable the "Logon only with password" option before creating users.

    If you transfer user data from previous device versions in which the "Logon only with password" option was disabled, there may be multiple users with identical passwords. These users are listed in the user view. However, only users whose user name comes first by alphabetical order can log on in Runtime.

- "Hierarchy level" group
  - "Group-specific rights for user administration" check box
    When this check box is selected, administrators only manage users whose group number is less than or equal to their own.
    For example, an administrator whose group number is 5 can only manage users whose group number is less than or equal to 5. This means that the administrator can assign users only to groups with a group number less than or equal to 5.

- "Password" group
  - "Enable password aging" checkbox selected
    The password expires after the number of days set in the "Validity of the password (days)" field.
    The "Password aging" column is selected in the "User groups" editor. This enables you to specify group-by-group, if the passwords should expire or if the password generations should be saved. Passwords never expire for groups for which password aging is not enabled.
  - "Prewarning time (days)" field
    The user is informed that the password will expire the specified number of days before the password expires.
  - "Password generations" field
    If the user changes the password, the new password must be different from the specified number of previous passwords. The number of password generations ranges from 1 to 5.
- "Password complexity" group
  - "Must include special characters" check box selected
    The user must enter a password containing at least one special character at any position.
  - "Must include number" check box selected
    The user must enter a password containing at least one number at any position.
  - "Minimum password length" field
    The user must enter a password with a minimum length, as specified in the "Minimum password length" field.
    The minimum length of the password is 3 characters.

Note

If the "Must include special characters" option is enabled, the system sets the passwords for users that have already been created. It is therefore recommended to always enable the "Must include special characters" option before creating users.

The following special characters are permitted for passwords:

-+.,/*:\"?_()=^!€%~`§'{}[]<>#$;&|@

12.6.1.8 Target groups in the user administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Principle

The target groups serve as examples for different groups of persons who use the user administration. We can distinguish between the following four groups in the user administration:

1. Administrator OEM
2. Administrator RT
3. Project engineer
4. Operator

As Administrator OEM, you create the user groups, users and authorizations for Runtime in the Engineering System of, for example, an equipment manufacturer.

As Administrator RT, you administer users in Runtime by means of the "User view."

As the project engineer, you assign the authorizations to the user groups in the Engineering System. You also configure authorizations for objects.

As Operator, you log on in Runtime. You can only access a protected object if you have the required authorization.

Note

The Administrator RT target group already exists in the Runtime user administration as the predefined user group "Administrator group." For ease of understanding, the predefined user groups and authorizations are not used below.

12.6.2 Setting up the user administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

12.6.2.1 Administering users for Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Creating an authorization (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You create an authorization and assign it to one or more user groups.

Requirements

The "User groups" work area is open.

Procedure

1. Double-click "Add" in the "Authorizations" table.
2. Enter "Stop runtime" as the name of the authorization.
3. Enter a brief description as the "Comment".

Creating a user group (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

User groups are created so that you do not have to assign an authorization to every single user. You create a user group, assign authorizations and then assign users to it.

Note

The name of the user group has to be unique within the project. Otherwise the input is not accepted.

Note

Using SIMATIC Logon

Ensure that the names of the user groups in Windows and WinCC are identical.

Requirements

The "User groups" work area is open.

Procedure

1. Double-click "Add" in the "Groups" table.
2. Enter "Operators" as the "Name" of the user group.
3. Change the "Number" of the user group as required.
4. Enter "Display name" of the "Operators" user group.
5. Enter a brief description as the "Comment".

In runtime, the user view shows the display name of the user group. The display name of the user group depends on the language. You can specify the name in several languages and switch between languages in runtime.

See also

Assigning an authorization (Page 5622)

Creating users (Page 5622)

Assigning a user to a user group (Page 5624)

Managing user groups (Page 5625)

User administration (Page 5624)

Assigning an authorization (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

When you allocate an authorization to a user group, all assigned users have this authorization.

Requirements

- A "Stop runtime" authorization has been created.
- An "Operators" user group has been created.
- The "User groups" work area is open.

Procedure

1. Click on the "Operators" user group in the "Groups" table. The "Authorizations" table shows all authorizations.
2. Enable the "Stop runtime" authorization in the "Authorizations" table.

Note

The "Stop runtime" authorization is only a designation and does not have any relation to the function "StopRuntime". You have to establish this relation yourself. To do so, configure the "StopRuntime" system functions at a button and select "Stop runtime" as the "Authorization".

See also

Creating a user group (Page 5621)

Creating users (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You create a user so that users can log on with their user names in runtime after loading to the device.

As an alternative, you can create and change users by means of the "User view" in Runtime.

Note

Applies only to Runtime Professional:

New users who want to access through a Web client or a WebUX client on the system, can only be created in the engineering system.

When you create a user in Runtime with the user view, it is not possible to assign a Web startup screen to this user. This user is therefore not authorized to access the Web.

Neither can existing users be granted web access rights in Runtime.

In order for a created user to have authorizations you have to assign him to a user group and allocate authorizations to the user group.

The logon is successful when the user name entered during the logon matches a user in Runtime. In addition, the entered password must agree with the stored password of the user.

Note

Note that the entry is case-sensitive.

Requirements

The "Users" work area is open.

Procedure

1. Double-click "Add" in the "Users" table.
2. Enter "Foreman" as the user name.

   Note

   The user name must be unique within the project. Otherwise the input is not accepted.

   Note

   Do not use special characters such as / " § $ % ? ' & when you enter a user name and a password.

3. Click the  button in the "Password" column. A dialog box for entering the password is displayed.
4. Enter the password of the user.
5. To confirm the password enter it a second time in the lower field.
6. Close the dialog box using the  icon.
7. If a user is to be logged off after a specific time period, activate "Automatic logoff".
8. Click in the "Logoff time" column. The preset value for "Logoff time" is 5 minutes.
9. Enter a brief description as the "Comment".

See also

Creating a user group (Page 5621)

Assigning a user to a user group (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

When you assign a user to a user group, the user has the authorizations of the user group.

Note

You have to assign a user to exactly one user group. The assignment is checked during the consistency check and generation of the project.

Requirements

- The user "Foreman" has been created.
- An "Operators" user group has been created.
- The "Users" work area is open.

Procedure

1. Click on the "Foreman" user in the "Users" table. The "Groups" table shows all user groups.
2. Activate the "Operators" user group in the "Groups" table.

See also

Creating a user group (Page 5621)

User administration (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

In the work area, you can administer users and assign them to user groups.

Requirements

The "Users" work area is open.

Changing the user name

1. In the "Users" table, double-click the field in the "Name" column to change the user name.
2. Change the user name.
3. Confirm your entry with <Return>.

Alternatively, select the user in the work area. Change the user name under "Properties > Properties > General" in the Inspector window.

Changing the password of the user

1. Click the button in the "Password" column of the "Users" table. A dialog for entering the password opens.
2. In the "Enter password" field, enter the new password.
3. Reenter the new password in the "Confirm password" field.
4. Confirm your entry with <Return>.

Alternatively, select the user in the work area. Change the password under "Properties > Properties > General" in the Inspector window.

Displaying invisible columns

1. Position the mouse cursor on the title of the "Users" table.
2. Right-click to open the shortcut menu and enable the display of the "Logoff time" column, for example.

Changing the logoff time of the user

1. In the "Users" area, double-click on the field in the "Logoff time" column to change the logoff time.
2. Change the logoff time.
3. Confirm your entry with <Return>.

Alternatively, select the user in the work area. Change the logoff time under "Properties > Properties > Automatic logoff" in the Inspector window.

Deleting a user

1. Select the line of the user to be deleted.
2. Open the shortcut menu with the right mouse button and select the "Delete" command.

Note

Predefined users cannot be deleted.

See also

Creating a user group (Page 5621)

Managing user groups (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

In the workplace you administer user groups and assign authorizations for use in runtime.

Requirements

The "User groups" work area is open.

Changing the name of the user group

1. In the "Groups" table, double-click the field in the "Name" column to change the name of the user group.
2. Change the name of the user group.
3. Confirm your entry with <Return>.

Alternatively, select the user group in the work area. Change the name under "Properties > Properties > General" in the Inspector window.

Note

Predefined user groups cannot be deleted.

Displaying invisible columns

1. Position the mouse cursor on the title of the "Users" table.
2. Right-click to open the shortcut menu and enable the display of the "Display name" column, for example.

Changing the displayed name of the user group

1. In the "Groups" table, double-click the field in the "Display name" column to change the display name of the user group.
2. Change the displayed name of the user group.
3. Confirm your entry with <Return>.

Alternatively, select the user group in the work area. Change the display name under "Properties > Properties > General" in the Inspector window.

Deleting a user group

1. Mark the line of the user group to be deleted.
2. Open the shortcut menu with the right mouse button and select the "Delete" command.

Note

Predefined user groups cannot be deleted.

Changing the name of the authorization

1. In the "Authorizations" table, double-click the field in the "Name" column to change the name of the authorization.
2. Change the name of the authorization.
3. Confirm your entry with <Return>.

Alternatively, select the authorization in the work area. Change the name under "Properties > Properties > General" in the Inspector window.

Deleting authorizations

1. Mark the line of the authorization to be deleted.
2. Open the shortcut menu with the right mouse button and select the "Delete" command.

Note

Predefined authorizations cannot be deleted.

See also

Creating a user group (Page 5621)

12.6.2.2 Managing users in Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Users in Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Principle

You create users and user groups and assign authorizations to them. You configure objects with authorizations. After download to the HMI device, all objects which were configured with an authorization are protected against unauthorized access in Runtime.

User view

When you configure a user view in the Engineering System, you administer users in this user view following download to the HMI device.

NOTICE
Changes in the user view are effective immediately in Runtime. Changes in runtime are not updated in the engineering system. When downloading the user administration to the HMI device, all changes in the user view are overwritten after a security prompt and based on the settings.

Users who have a "User administration" authorization have unlimited access to the user view. This allows them to administer all users. Any other user has only limited access to the user view for self administration.

User view (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Purpose

You configure a user view in the engineering system to also administer the users in Runtime.

Structure

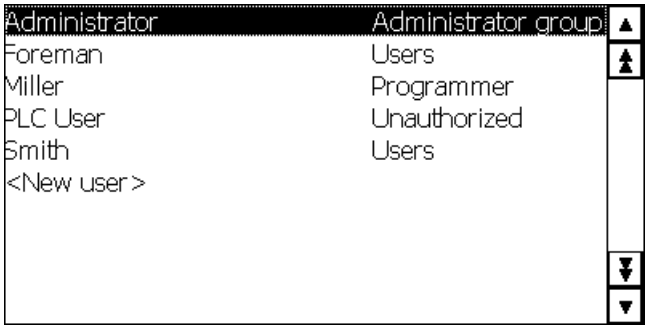
The user view shows the following in each line:

- The user
- The corresponding user group.

If no user is logged on, the user view is empty. The content of the individual fields is displayed after logon.

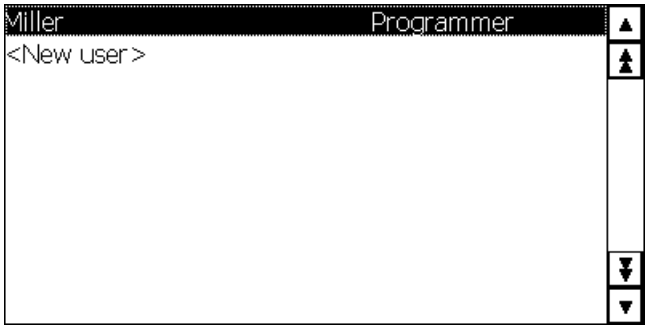


User view of an administrator



When an administrator is logged on, the user view shows all the users. The administrator changes the user name and the password. The administrator creates new users and assigns them to an existing user group.

User view of a user



When no administrator is logged on, the user view shows only the logged-on user. Users can change their own passwords.

User view (Panels, Comfort Panels, RT Advanced, RT Professional)

Purpose

You configure a user view in the engineering system to also administer the users in Runtime.

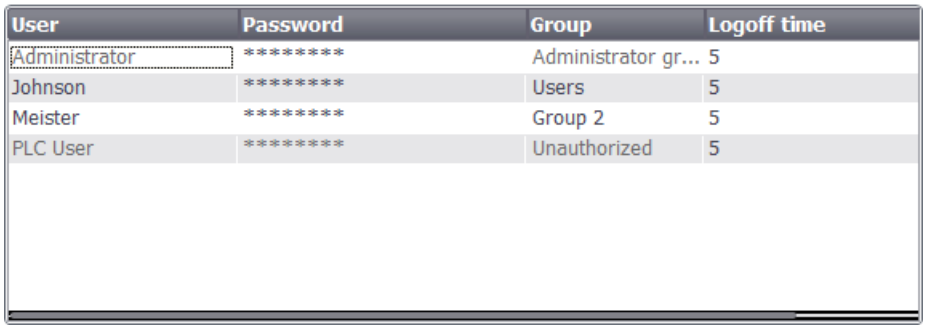
Structure

The user view shows the following in each line:

- The user
- The encrypted password
- The corresponding user group
- The logoff time

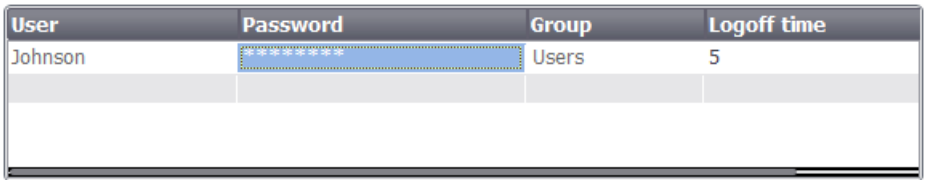
If no user is logged on, the user view is empty. The content of the individual fields is displayed after logon.

User view of an administrator



When an administrator is logged on, the user view shows all the users. The administrator changes the user name and the password. The administrator creates new users and assigns them to an existing user group.

User view of a user



When no administrator is logged on, the user view shows only the logged-on user. Users can change their own passwords and logoff time.

Configuring a user view (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You configure a user view in the Engineering System to also administer users in Runtime.

Requirements

A screen has been created.

Procedure

1. Select the "User view" object from the "Controls" category in the toolbox.
2. Drag-and-drop the "User view" object into the screen.
3. Click on "Properties > Properties" in the Inspector window.
4. Specify the appearance of the "User view".
5. You can, for example, select " "Display mode > Fit to size > Fit object to contents".

Result

You have created a user view in the screen.

Creating users (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You create a user so that users can log on under their user name in runtime.

As an alternative, you can create users in the engineering system and download them to the HMI device.

Note

Applies only to Runtime Professional:

New users who want to access through a Web client or a WebUX client on the system, can only be created in the engineering system.

When you create a user in Runtime with the user view, it is not possible to assign a Web startup screen to this user. This user is therefore not authorized to access the Web.

Neither can existing users be granted web access rights in Runtime.

The logon is successful only when the user name entered during the logon matches a user in runtime. In addition, the password entered at logon has to match that of the user.

Note

Note that the entry is case-sensitive.

Note

Do not use special characters such as / " § $ % ? ' & when you enter a user name and a password.

You assign the user to a user group. The user then has the authorizations of the user group.

Note

Runtime users must be assigned to a user group. The user group is created in the engineering system. The designation of the user group is language-dependent.

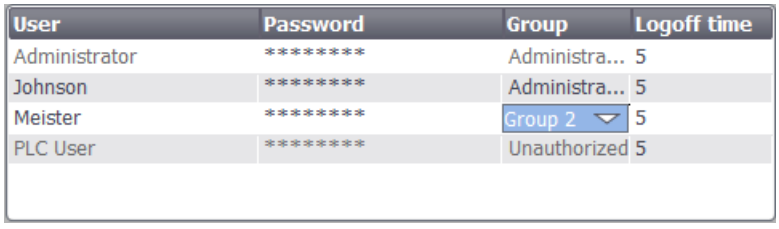
Requirements

- The user view is open.
- A "Group 2" user group has been created.

Procedure

1. Click "<New User>" in the user view. A dialog opens.
2. Enter "Foreman" as the user name.
3. Press the <Return> button.
4. Click "Password."
5. Enter the password of the user.

6. Press the <Return> button. The password is hidden.
7. Click in the "Group" column.
8. Select "Group 2" as the "Group".



9. Press the <Return> button.
10. Click in the "Logoff time" column.
11. Enter the time after which the user is logged off automatically.

Managing users in the user view (Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

If you have configured a user view in the engineering system, the users and user groups can be managed in runtime.

NOTICE
Changes in the user view are effective immediately in runtime. Changes in runtime are not updated in the engineering system. When downloading the user administration to the HMI device, all changes in the user view are overwritten after a security prompt and based on the settings.

Requirements

- Runtime is enabled.
- A complex user view has been created.
- The screen with the user view is open.
- You have the default "User administration" authorization.

Note

If you do not have a "User administration" authorization, you can only change your own password and your logoff time.

Changing the user name

1. Enter a new user name in the "Users" column in the user display.
2. Confirm your entry with <Return>.

Note

The user can then no longer log on with his previous password in runtime. If you delete the name and press <Return>, the user is deleted.

Changing the password of the user

Availability of complex user display in selected devices only.

1. Enter a new password in the "Password" column in the user display.
2. Confirm your entry with <Return>.

Note

The user can then no longer log on with his previous password in runtime.

If you delete the password in the complex user view and press <Return>, the user will be deleted.

Changing the logoff time of the user

1. Enter a new logoff time in the "Logoff time" column in the user display.
2. Confirm your entry with <Return>.

Deleting a user

1. Click the name of the user to be deleted.
2. Delete the name.
3. Press the <Return> button.

Note

The user can no longer log on in runtime.

Assigning a user to a different user group

1. Activate the user group field for the corresponding user.
2. Select a user group.
3. Confirm your selection with <Return>.

Unlocking users (Basic Panels, Panels, Comfort Panels, RT Advanced)

Unlock locked out users

The check box "Activate limit for login attempts" is activated in the "Runtime settings > User administration".
The number 3 is entered in the field "Number of invalid login attempts".

If users have three failed attempts at login, e.g. by entering an incorrect password, they are assigned to the "Unauthorized" group. The user loses all authorizations. The user can still log on, but no longer has any authorizations. Only a user with administrator rights re-assigns the unauthorized user to a user group.

Logging on as a user (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

As a rule you log-on as a user by means of a special button. The logon dialog box is displayed to this purpose.

The logon dialog box is displayed by default during access to a protected object if

- No user is logged on in Runtime.
- The logged-on user does not have the required authorization.

Note

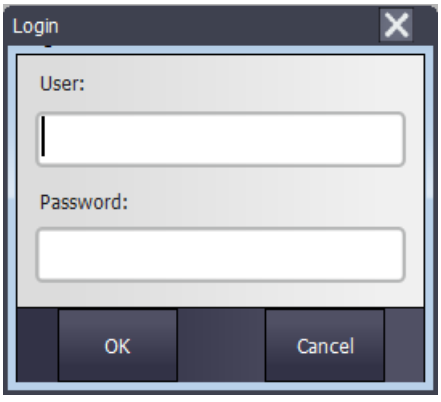
The system always opens the logon dialog on the OP 73, OP 77A, TP 177A and Basic Panels HMI devices when you press an access-protected button.

Requirements

- Under "Runtime settings > User administration" the
  - "Enable limit for logon attempts" check box has been selected.
  - The number 3 is entered in the field "Number of incorrect login attempts".
- The "ShowLogonDialog" system function is configured on a button called "Logon".

Procedure

1. Click the "Logon" button. The logon dialog box is displayed.



2. Enter your user name as it was specified in the user administration, for example "Foreman". If someone has logged on before you, the name of the user will be displayed.
3. Enter the corresponding password. The input is concealed.

   Note

   When entering the password, note that it is case sensitive.

4. Click "OK" to close the dialog box.

Note

When you change the HMI device, you do not need to re-enter the password if the permitted special characters are included in the password on all HMI devices.

The following special characters are permitted for passwords:

()€&@$%+#^[]|*-"<>!,.;:/_=\'? `~{}$

If you switch to V14 from an older device version and prohibited special characters were used for the password, the password is deleted. You are informed about this with an error message and have to save a new password.

Logon was successful

If you have entered the user name "Foreman" and the entered password corresponds with the stored password, you are logged on as the user "Foreman" in Runtime. You have the authorizations of the user "Foreman".

When the user "Foreman" accesses a protected object such as the "Logging" button, access to this protected object will only be authorized if the user "Foreman" has the required authorization. The programmed function is executed immediately.

If you do not have the required authorization after the successful log-on, a corresponding error message is displayed. However, you remain logged on in Runtime.

Logon was unsuccessful

An error message is displayed.

In order to maintain security, you or the user logged-on before you no longer has any authorizations. However, access to unprotected objects remains possible. The user view does not, however, show any entries. The user view and the authorizations change after the next successful log-on.

If the third login attempt has failed, the user will be assigned to the "Unauthorized" group. For this reason, do not configure a user group with this display name.

If the "Log off" function is called up or the logoff time of the user has expired, the user is logged off.

12.6.2.3 Configuring access protection (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Access protection (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You configure an authorization at an object in order to protect it against access. All logged-on users who have this authorization can then access the object. If a user does not have authorization to operate an object, the logon dialog is displayed automatically.

Note

Several system functions are available under "User administration" so that user, password, and user group can be edited, for example, in the PLC.

Configuring operating authorizations (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You configure the "Stop runtime" authorization on a button. Then only those users who have the appropriate authorization have access to this button, for example all the users of the "Operators" user group.

This ensures that access to the command button is protected. If a logged-on user who belongs to the "Operators" user group and has the required authorizations clicks the button, runtime is stopped.

An example describes in detail how to configure a command button with access protection.

Requirements

- The "Operators" user group has been created.
- The "Stop runtime" authorization has been created.
- A screen has been created and opened.
- The screen contains a button.

Procedure

1. Click the button in the screen.
2. Click "Properties > Properties > Security" in the Inspector window.
3. Select "Stop runtime" as the "Authorization".
4. In the Inspector window, select "Properties > Events > Click".
5. Select a system function from the function list, for example, "StopRuntime".

Note

The "Enable" and "Disable" events are only used to detect whether an object was selected or deselected. The events do not, however, trigger a password prompt.

Do not use the "Enable"or "Disable" event if you want to configure access protection at the function call of the object.

12.6.3 Reference

12.6.3.1 Objects with access protection (Basic Panels)

Introduction

The following objects can be configured with an authorization:

- Date/time field
- I/O field
- Graphic I/O field
- Recipe view
- Switch
- Button
- Symbolic I/O field
- System diagnostic view

12.6.3.2 Default user groups and authorizations (Basic Panels, Panels, Comfort Panels, RT Advanced)

Principle

The predefined user groups and authorizations have the following numbers:

User group	Number
"Administrator group"	1
"Users"	2

Authorization	Number
"User administration"	1
"Monitor"	2
"Operate"	3

12.6.4 Examples

12.6.4.1 Example: Configuring a button with logon dialog box (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Task

In the following example, you configure the function "ShowLogonDialog" for a button. A different user can then log on in runtime when the shift changes, for example. In the process the user previously logged on is logged off.

Note

In runtime the logon dialog box is not displayed by default until you access a protected object. Either no user is logged on or the logged-on user does not have the required authorization.

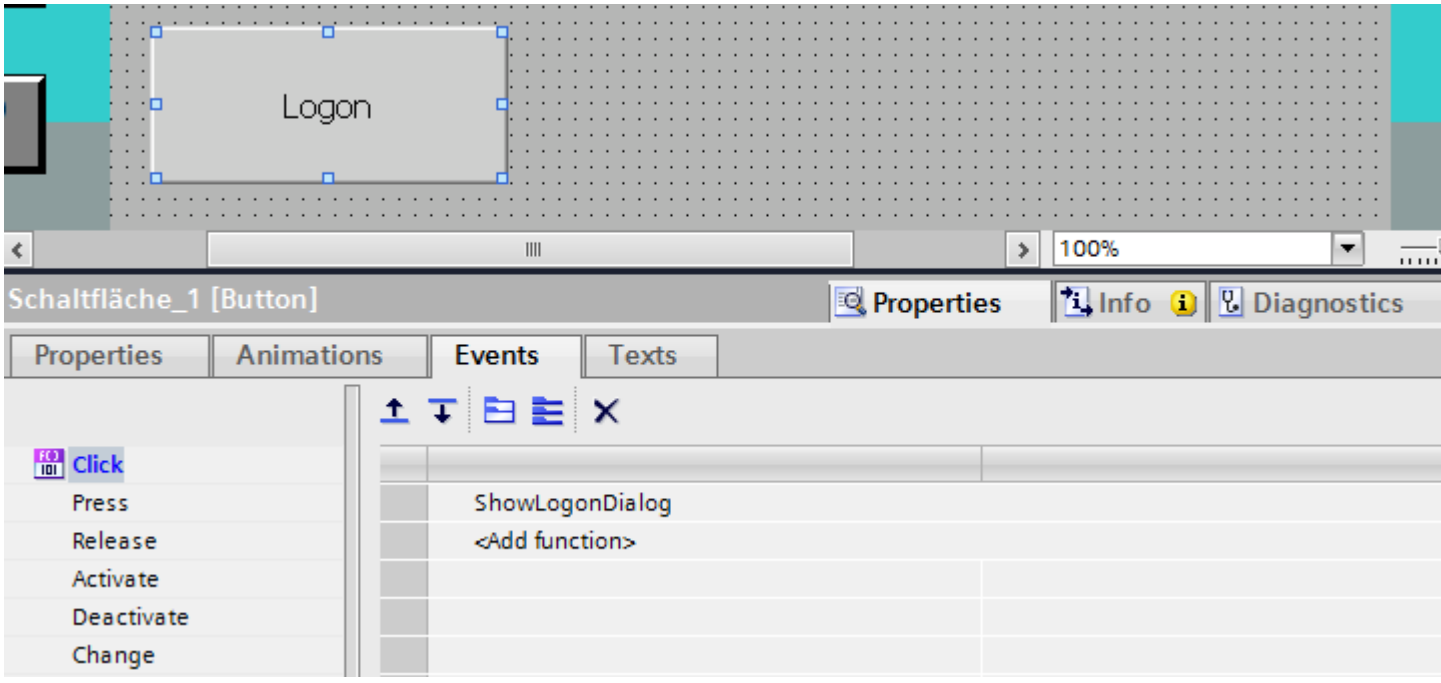
Requirements

- A screen has been created.
- A button has been created in the screen.

Procedure

1. Click the button in the screen.
2. Click "Properties > Events > Release" in the Inspector window.

3. Click the entry "Add function" in the "Function list" table.
4. Select the system function "ShowLogonDialog" from the "User administration" group.



Result

If the user clicks on the button in runtime, the function "ShowLogonDialog" is called up. When the function "ShowLogonDialog" is called up, the logon dialog box is displayed. The user logs on with his user name and password.

12.6.4.2 Example: Logging the logon and logoff events (Basic Panels, Panels, Comfort Panels, RT Advanced)

Task

In the following example, you configure the function "TraceUserChange" to the event "User change".

Principle

The "TraceUserChange" function is called when a user logs on or off. When a function is called up, a system message with information about the corresponding user is output.

This system message can be archived. When archiving, the system message is provided with a date stamp and time stamp. This ensures that you can track which user was logged on at the HMI device at which time and for how long.

Requirements

- You have created an HMI device with Runtime Advanced.
- The Inspector window is open.

Procedure

1. Double-click the "Scheduler" in the Project view.
2. Double-click "Add" in the table of the tasks.
3. Enter "Logon-Protocol" as the "Name".
4. Select "User change" as the "Trigger".
5. Open "Properties > Events" in the Inspector window.
6. Click the entry "Add function" in the "Function list" table.
7. Select the "TraceUserChange" system function.

Result

A system message is output when a user logs on or logs off.

12.6.4.3 Example: Structure of user management (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Task

In the following example you set up a user administration for different users and user groups. The example orientates itself to a typical requirement profile from manufacturing engineering.

Principle

Completely different groups of persons are involved in a plant or project. Each group of persons protects its respective data and functions against access by others. For this purpose, users are created and assigned to a user group.

You can reproduce different views through user groups.

Example:

- Organizational view: Commissioners, Operators, Shift I, Shift II
- Technological view: Axis control, Tool changers, Plant North, Plant South

The following example orientates itself to the organizational view.

Every user group has characteristic requirements regarding access protection: A user group has operating authorizations for specific application cases. A programmer changes, for example, recipe data records.

In the example the users Miller, Group Smith and Foreman are created and assigned to different user groups.

Ms. Miller works as a programmer in the engineering system. The Group Smith are commissioners. Mr. Foreman is an operator.

Requirements

- A new project has been created.
- The "User administration" editor is open.

Procedures overview

Working with user administration has the following procedure in the example:

1. Creating authorizations The planner specifies which authorizations are required for access protection.
2. Configuring authorizations: The planner specifies which objects may be operated and which functions may be executed.
3. Creating user groups and allocating authorizations: The administrator creates the user groups together with the planner. The planner uses the authorizations to specify who may operate objects and change parameters.
4. Creating users and assigning them to a user group: The administrator administers the users.

Result

The aim is the following structure of the user administration of users, user groups and authorizations:

Users			User groups	Authorizations			
Miller	Smith	Foreman	Roles	Changing recipe records	Changing system parameters	Changing process parameters	Managing
			Administrator group				x
X			Programmer	X			
	X		Commissioning engineers	X	X	X	
		X	Operators	x			

The user "Foreman" who belongs to the "Operators" user group has access to the configured "To Recipe view" button.

Note

Alternatively, you can create several users as operators with different operating authorizations, for example, Operator Level 1, Operator Level 2.

12.6.4.4 Example: Creating and configuring authorizations (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

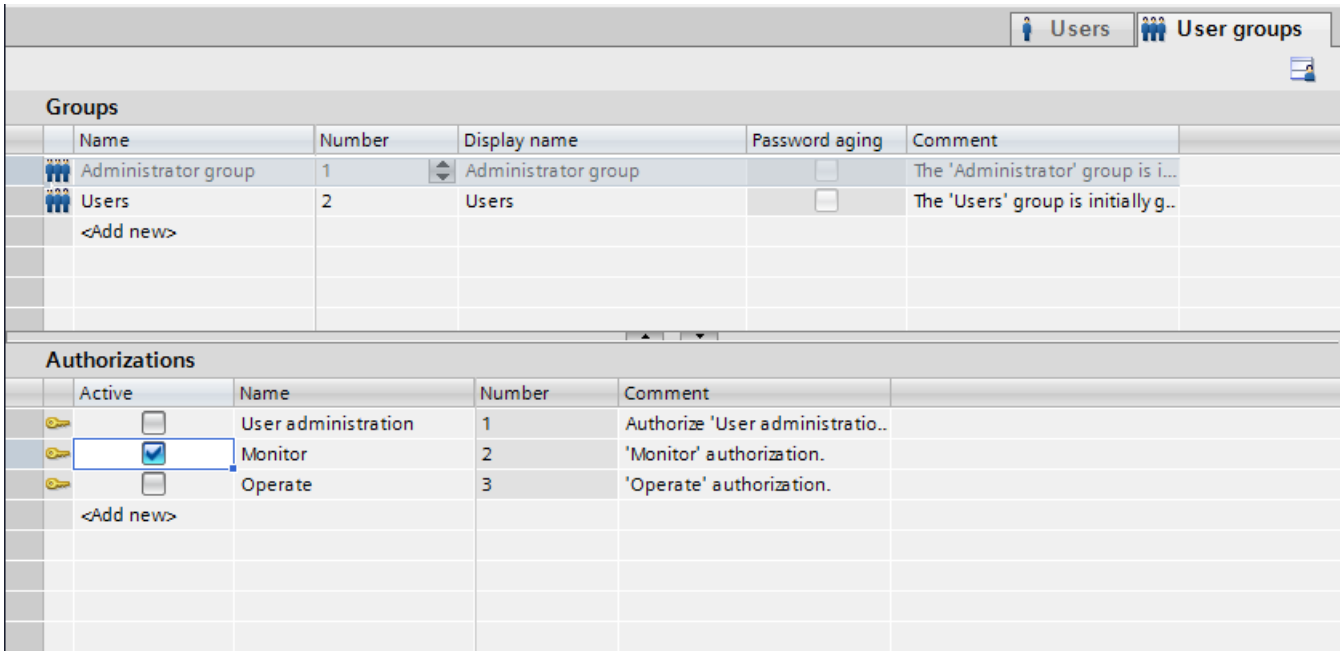
Task

The following example shows you how to create the authorizations

Procedure

1. Open the "User groups" work area.
2. Double-click "Add" in the "Authorizations" table.
3. Enter "Change recipe data records" as the "Name" of the authorization.
4. Repeat steps 2 and 3 to create additional authorizations: "Change system parameters", "Change process parameters".

Result



12.6.4.5 Example: Configuring a button with access protection (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Task

In the following example you use a system function to create a button for a screen change. You protect the "To Recipe view" button against unauthorized operation. To do so, you configure the "Change recipe data records" authorization at the "To Recipe view" button.

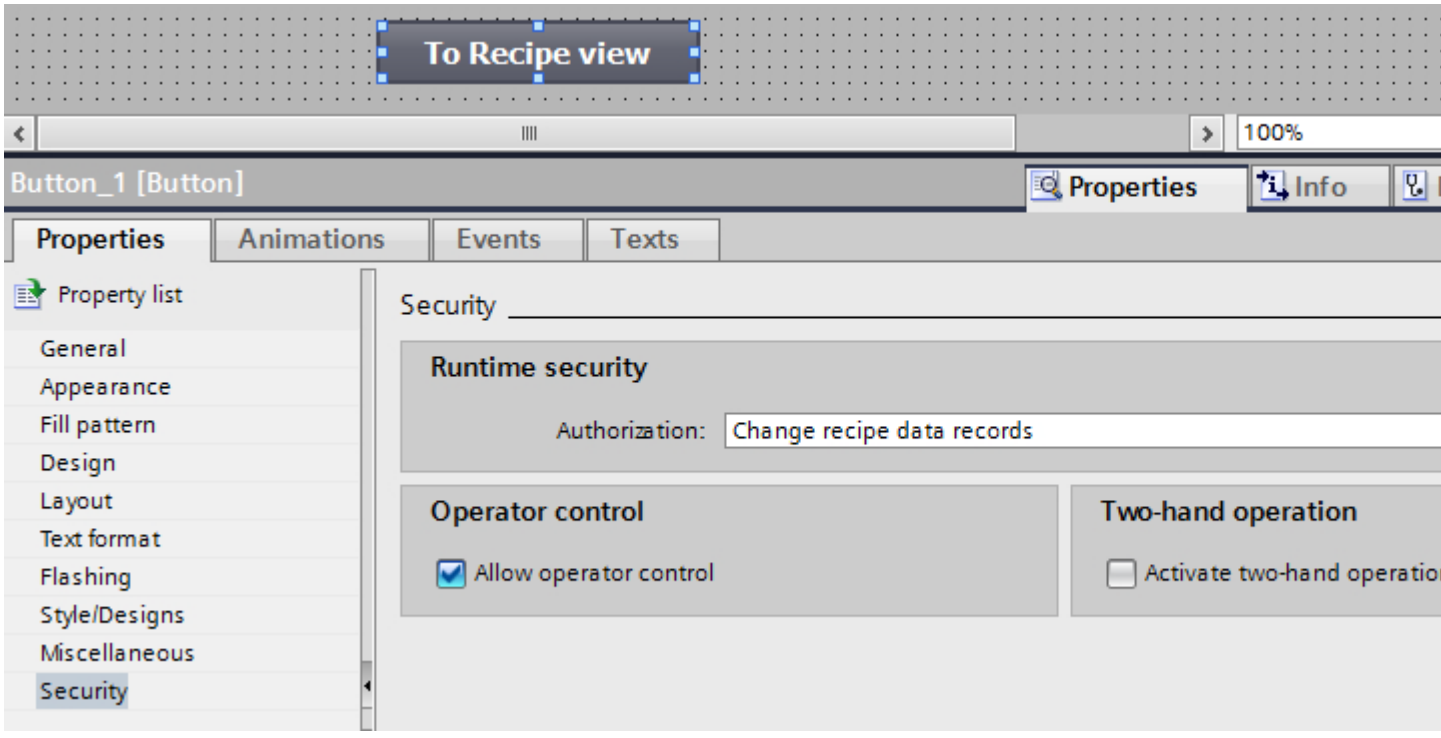
Requirements

- A "Change recipe data records" authorization has been created.
- A "Recipes" screen has been created.
- A "Start" screen has been created and opened.
- A button has been created and marked in the "Start" screen.

Procedure

1. Click "Properties > Properties > General" in the Inspector window.
2. Enter "To Recipe view" as the text.
3. Click "Properties > Events > Click" in the Inspector window.
4. Click the "Add function" entry in the first line of the "Function list" table.
5. Select the "ActivateScreen" system function in the "Screens" group.
6. Click the ... button in the "Screen name" field. A dialog box for selecting the screen opens.
7. Select the "Recipes" screen and use the ✓ button to close the dialog box.
8. Click "Properties > Properties > Security" in the Inspector window.
9. Select "Change recipe data records" as the "Authorization."

Result



Access to the "To Recipe view" button is protected. If, for example, the user "Smith" clicks the button in Runtime, the function "Recipe view" screen is called up. Prerequisite is that the user "Smith" has logged on correctly and has the required authorization. The "Recipes" screen contains a recipe view and other screen objects.

Note

If the logged-on user does not have the required authorization or if no user is logged on, the "Logon dialog box" is displayed. An alarm indicating that no operator authorization is available for this user appears in Runtime Professional in this case.

12.6.4.6 Example: Creating user groups and assigning authorizations: (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

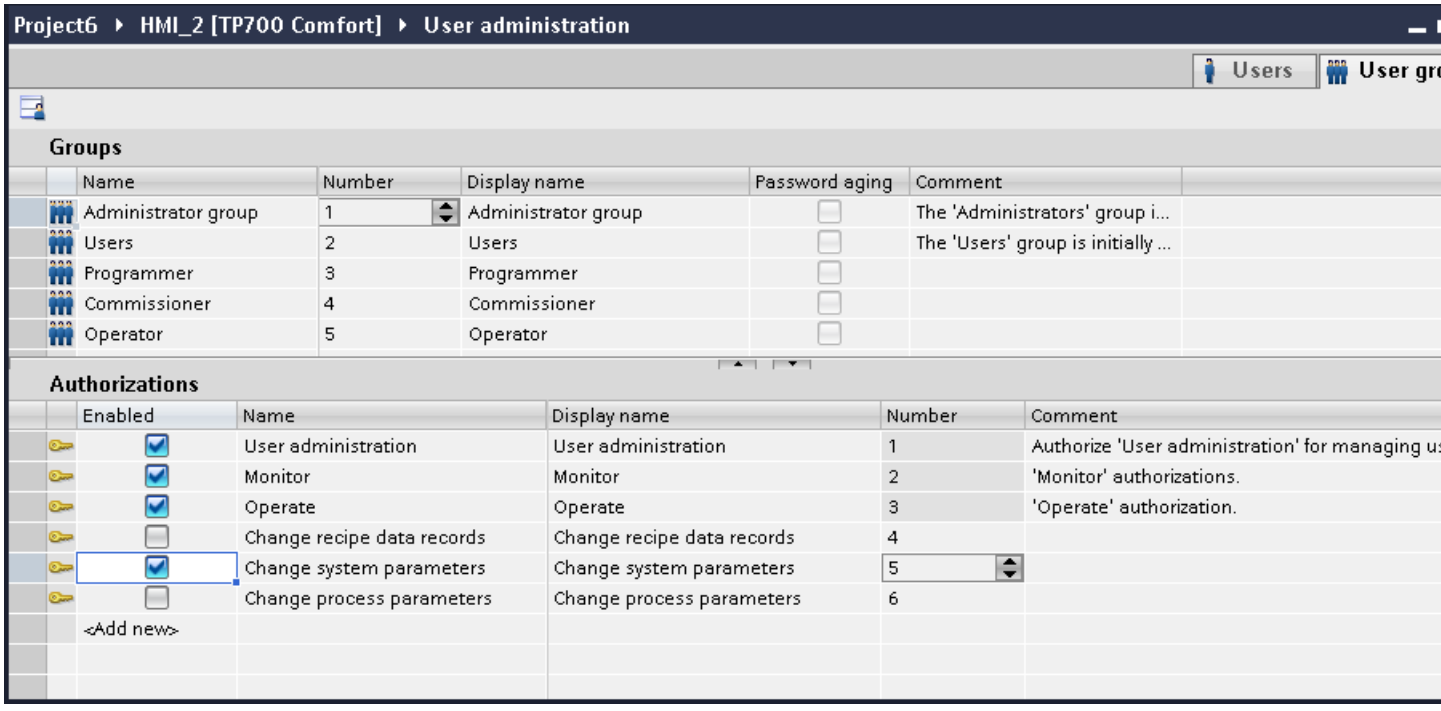
Task

In the following example you create the user groups and assign authorizations to them.

Procedure

1. Open the "User groups" work area.
2. Double-click "Add" in the "Groups" table.
3. Enter "Programmer" as the "Name".
4. Repeat steps 2 and 3 to create the "Commissioner" and "Operator" user groups.
5. Click "Administrator" in the "Groups" table.
6. Activate the "Change system parameters" authorization in the "Authorizations" table.

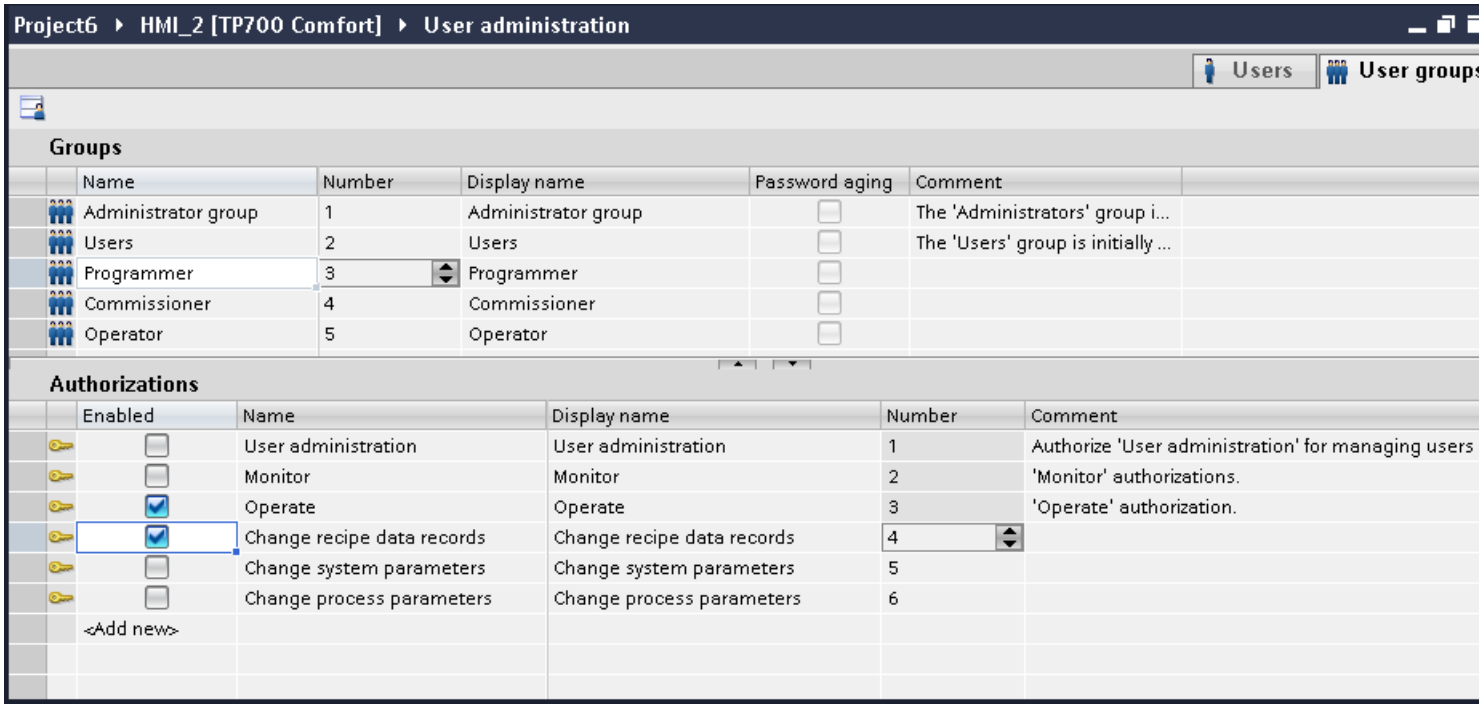
Interim result



Procedure

1. Click "Operator" in the "Groups" table.
2. Activate the "Change recipe data records" authorization in the "Authorizations" table.
3. Click "Commissioner" in the "Groups" table.
4. Activate the authorizations "Change recipe data records", "Change system parameters" and "Change process parameters" in the "Authorizations" table.
5. Click "Programmer" in the "Groups" table.
6. Activate the "Change recipe data records" authorization in the "Authorizations" table.

Result



12.6.4.7 Example: Creating users and assigning them to a user group (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

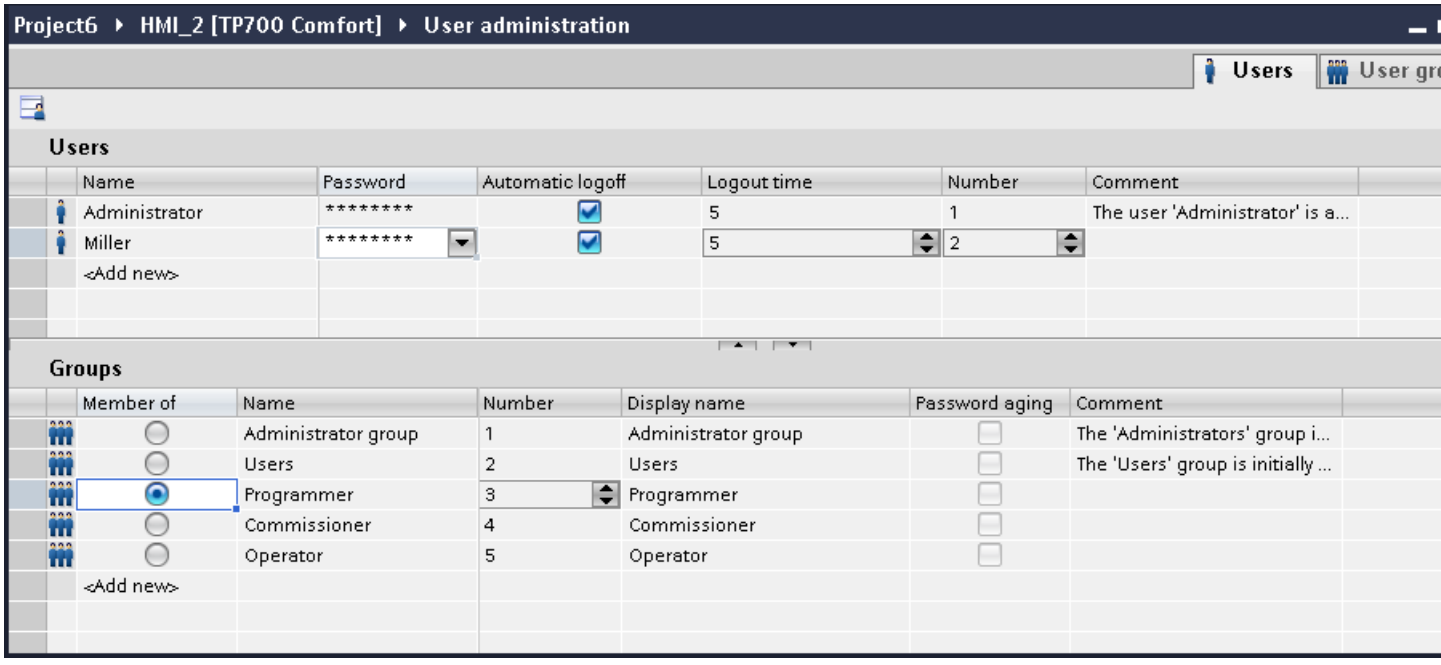
Task

In the following example you create the users and assign user groups to them. The users are sorted alphabetically immediately after the name has been entered.

Procedure

1. Open the "Users" work area.
2. Double-click "Add" in the "Users" table.
3. Enter "Miller" as the user name.
4. Click the button in the "Password" column. The dialog box for entering the password is displayed.
5. Enter "miller" as the password.
6. To confirm the password enter it a second time in the lower field.
7. Close the dialog box by using the icon.
8. Activate the "Programmer" user group in the "Groups" table.

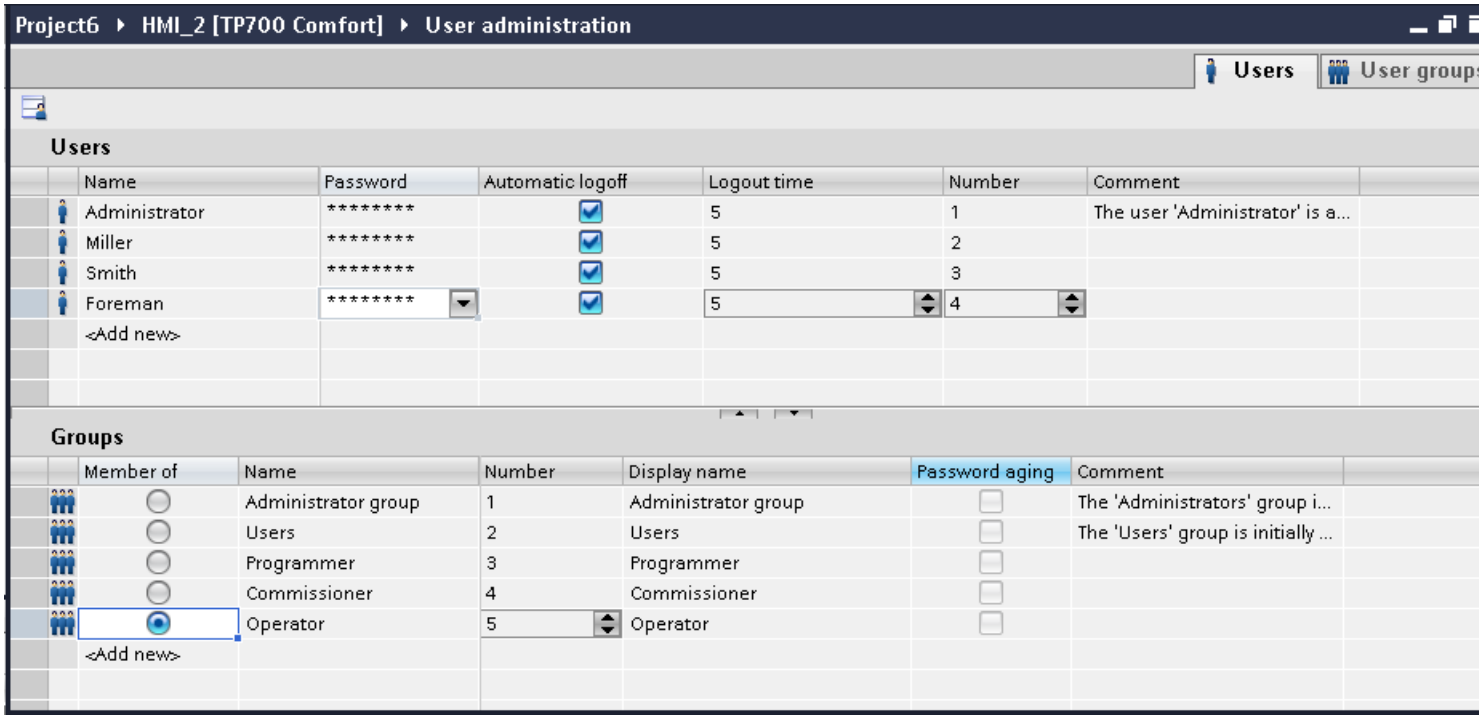
Interim result



Procedure

1. Double-click "Add" in the "Users" table.
2. Enter "Smith" as the user name.
3. Click the button in the "Password" column. The dialog box for entering the password is displayed.
4. Enter "smith" as the password.
5. To confirm the password enter it a second time in the lower field.
6. Close the dialog box by using the icon.
7. Activate the "Commissioner" user group in the "Groups" table.
8. Repeat steps 2 to 6 for the user "Foreman."
9. Activate the "Operators" user group in the "Groups" table.

Result


Project6 ▸ HMI_2 [TP700 Comfort] ▸ User administration
Users
User groups
Users
Name
Password
Automatic logoff
Logout time
Number
Comment
Administrator
****
5
1
The user 'Administrator' is a...
Miller
****
5
2
Smith
****
5
3
Foreman
****
5
4
<Add new>
Groups
Member of
Name
Number
Display name
Password aging
Comment
Administrator group
1
Administrator group
The 'Administrators' group i...
Users
2
Users
The 'Users' group is initially ...
Programmer
3
Programmer
Commissioner
4
Commissioner
Operator
5
Operator
<Add new>

12.7 Working with system functions

12.7.1 Basics

12.7.1.1 System functions (Basic Panels)

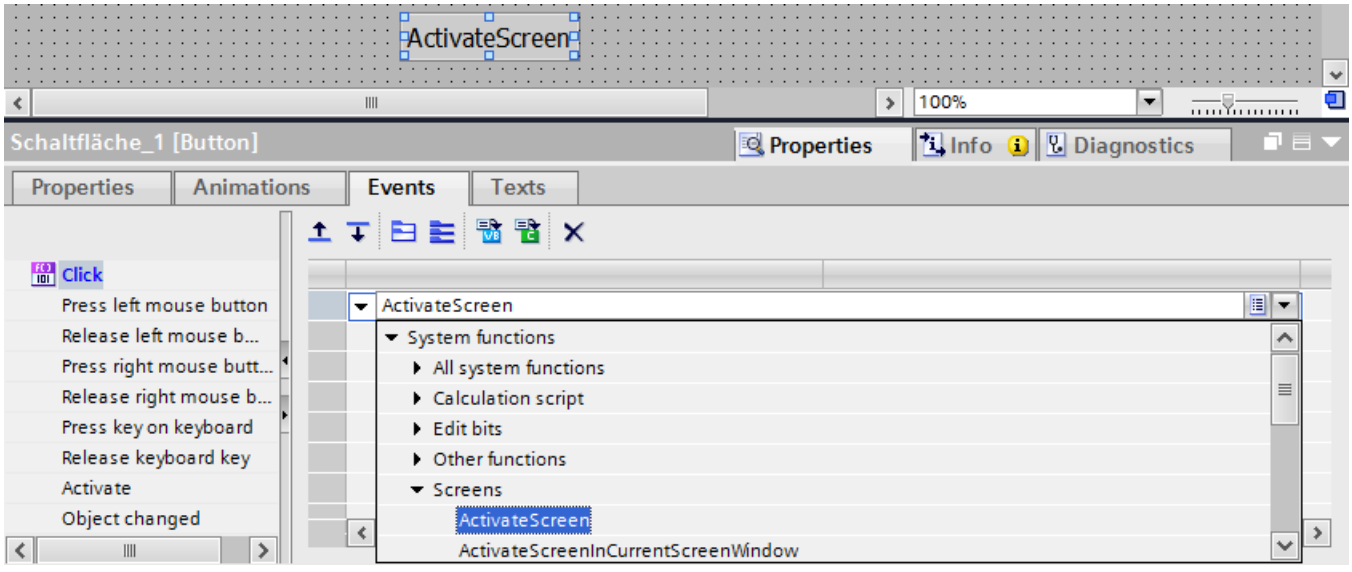
Introduction

System functions are functions supplied by WinCC. They are predefined and cannot be changed. You can use system functions to implement many tasks in Runtime even without having any programming knowledge, for example:

- Calculations, e.g. increasing a tag value by a specific or variable amount.
- Logging functions, e.g. starting a process value log.
- Settings, e.g. changing the PLC or setting a bit in the PLC.
- Alarms, e.g after a different user logs on.

Usage

You use system functions in a function list. When configuring a function list, you select the system functions from a selection list that is sorted by categories:



Each system function in WinCC is logically linked with an object and an event. As soon as the event occurs, the system function is triggered.

Language dependency

The names of the system functions are dependent on the set project language. The functionality can then be recognized immediately by the project planner.

Availability

You only configure system functions within a function list which are supported by the selected HMI device. If you use a project for several HMI devices, the system functions that are not supported by an HMI device are marked in color.

Events

The object and the selected function determine which events can be defined as triggers for executing a system function.

For example, the "Change value", "Low limit violated" and "Upper limit exceeded" events are associated with the "Tag" object. The "Loaded" and "Cleared" events are associated with the "Screen" object.

12.7.1.2 Use of system functions (Basic Panels)

Introduction

A function list is processed when a configured event occurs in runtime. The operator can trigger an event, for example, by pressing a function key on the HMI device. The system could also trigger an event if a process value falls below a specified limit, for example.

Applications

You can configure system functions on all the objects that are able to react to an event. You can use system functions directly in function lists and thereby control the sequence.

- Function list
  The system functions will be processed one line at a time in a function list. To avoid wait times, system functions in WinCC Runtime with a longer running time are executed simultaneously. For instance, a subsequent system function can already be performed even though the previous system function has not yet been completed.

An example for the configuration of a function list can be found under "AUTOHOTSPOT".

12.7.2 Working with function lists

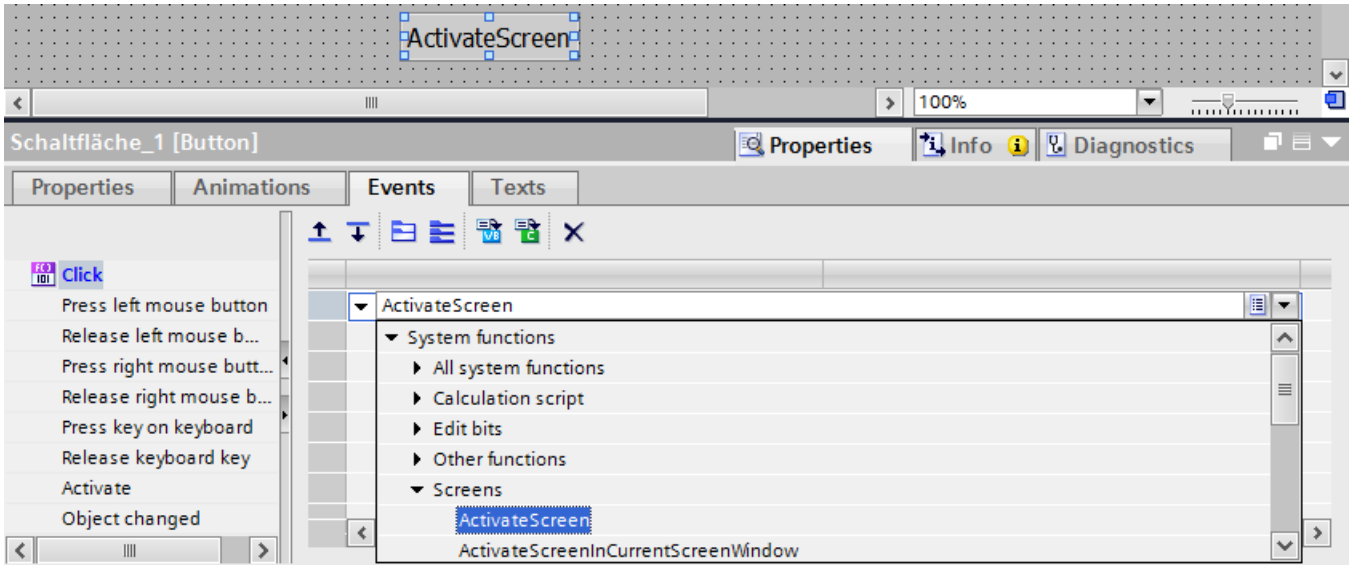
12.7.2.1 Basic of the functions list (Basic Panels)

Introduction

When the configured event occurs, several system functions can be performed with the function list.

Principle

The function list is configured for an event of an object, e.g. a screen object or a tag. The events which are available depend on the selected object and the HMI device.



Events occur only when the project is in Runtime. Events include:

- Value changes of a tag
- Pressing of a button
- Activation of Runtime

You can configure exactly one function list for each event.

Note

The choice of configurable system functions in a function list depends on the HMI device chosen.

12.7.2.2 Properties of a function list (Basic Panels)

Status information

During configuration the project data is tested in the background.

With the following causes the function list is not executed in Runtime and the incorrect entries are marked red:

- At least one system function is not completely supplied with parameters.
- At least one system function is contained which is not supported by the selected HMI device, for example, by changing the device type.

Executing system functions

System functions in a function list are executed in runtime sequentially from top to bottom. To avoid wait times, system functions with a longer running time (such as file operations) are processed simultaneously. For instance, a subsequent system function can already be performed even though the previous system function has not yet been completed.

Use a script with loops, conditional statements and abort conditions to program non-sequential and conditional procedures.

Note

Availability for specific devices

User-defined functions are not available on Basic Panels.

12.7.2.3 Configuring a function list (Basic Panels)

Introduction

You can configure a function list by selecting system functions from a drop-down list. The system functions are arranged in the drop-down list according to categories.

Requirement

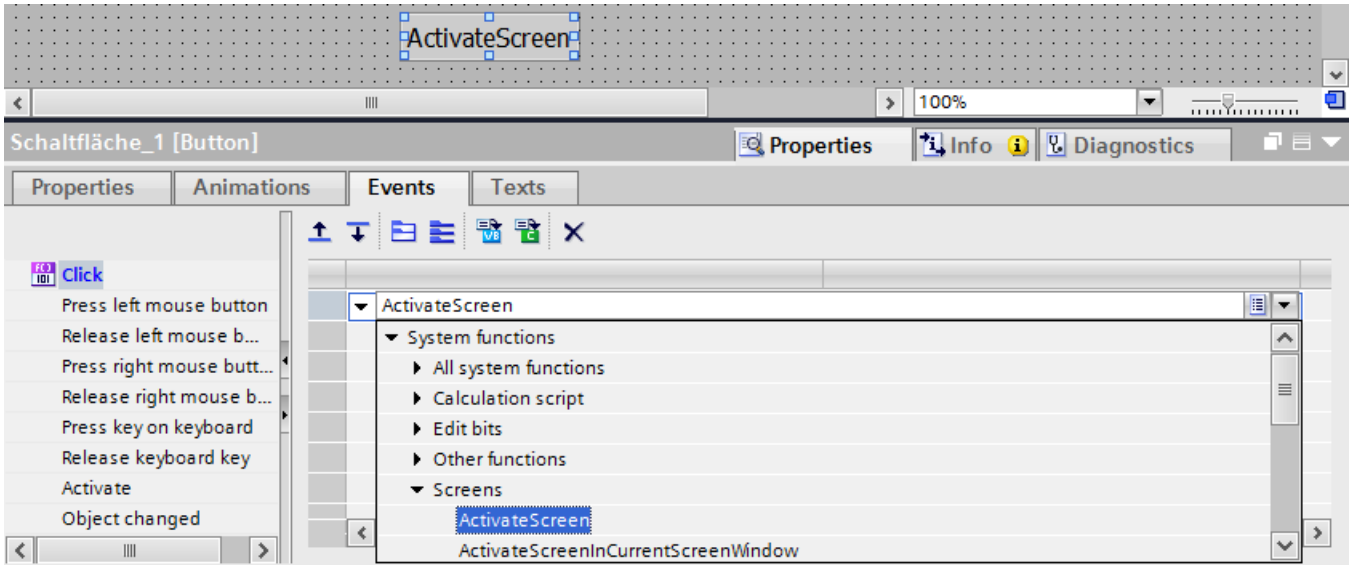
Object has at least one configurable event.

Procedure

Proceed as follows to configure a function list:

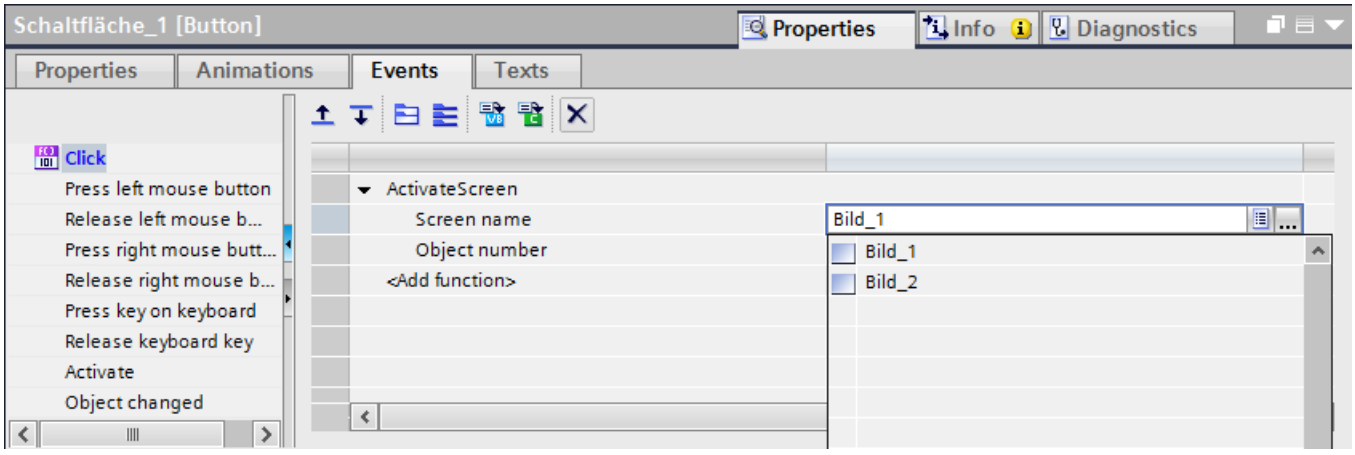
1. Open the editor in WinCC in which the object is located.
2. Select the object.
3. Click "Properties > Events" in the Inspector window. Select the event for which you want to configure the function list.

4. Select the "<No Function>" entry in the drop-down list of the Inspector window.
5. Select the desired system function.
6. You can also enter the name of the system function.



The system function is entered in the function list.

7. If the system function has parameters, specify the values for the parameters.



8. If you want to add other system functions or functions to the function list, then repeat steps four to seven.

Result

The function list is configured. In addition, to the configured event, the status of the function list is displayed in the Inspector window. When the configured event occurs in Runtime, the function list is completed from top to bottom.

12.7.2.4 Editing a function list (Basic Panels)

Introduction

A function list can be edited as follows:

- Changing the order of execution for system functions
- Removing a system function

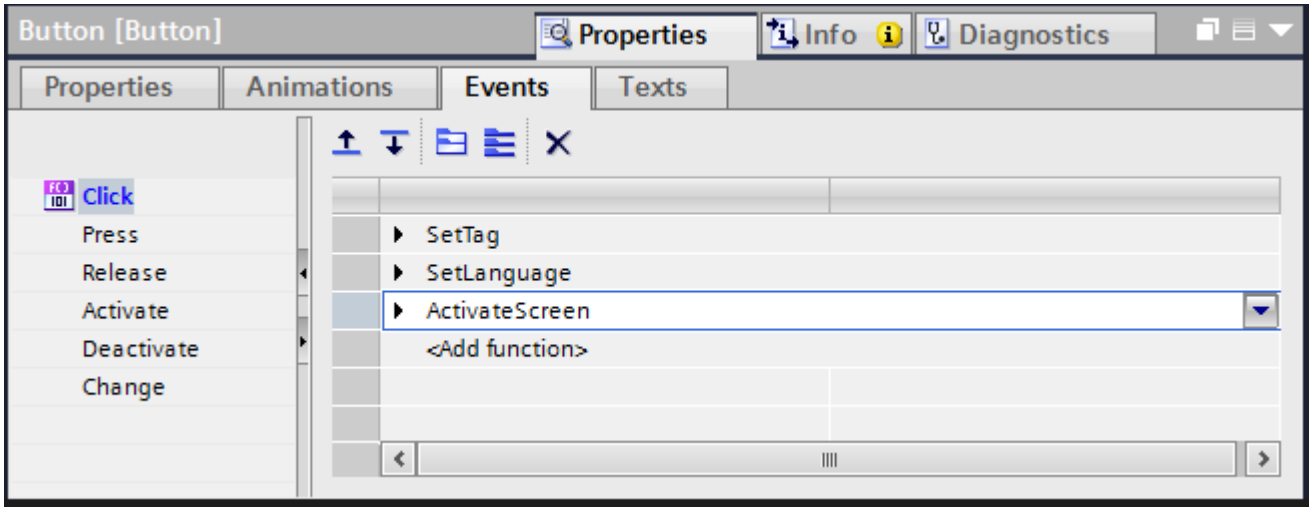
For additional information, refer to "Configuring a function list".

Requirement

The function list is configured.

Changing the order of a system function

1. Select the desired system function in the function list.
2. Then click the appropriate direction arrow in the inspector window until the system function or customized function is in the desired position.



Changing the order of several system functions

1. Hold down the <Shift> key.
2. Click the relevant system functions, working in succession.
3. Move the selection to the desired position by drag&drop.

Removing a system function

1. Select the desired system function in the function list.
2. Select "Delete" from the shortcut menu.

12.7.2.5 Executing a function list in Runtime (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Principle

A function list is executed from top to bottom in Runtime. A distinction is made between synchronous and asynchronous execution, so that no waiting periods ensue during execution. The distinction is made by the system by evaluating the different runtimes of the system functions. User-defined functions are always executed synchronously independent of the runtime. If a system function returns an error status, the execution of the function list is cancelled.

Synchronous execution

During synchronous execution, the system functions in a function list are executed one after the other. The previous system function must be finished before the next system function can be executed.

Asynchronous execution

System functions that perform file operations such as saving and reading have a longer runtime than system functions that, for example, set a tag value.

Therefore, system functions with longer runtimes are executed asynchronously. For example, while a system function is writing a recipe data record to a storage medium, the next system function is already being executed. Due to the parallel execution of system functions, waiting periods at the HMI device are avoided.

12.7.3 Example

12.7.3.1 Changing the operating mode on the HMI device with the current display (Basic Panels, Panels, Comfort Panels, RT Advanced)

Scheduled task

In this example, you use the "SetDeviceMode" system function to switch between the "online" and "offline" modes on the HMI device. You also display the current set operating mode on the HMI device.

Requirements

A screen has been created.

Settings

For this example you require a HMI-tag and a text list with the following settings:

HMI tag:

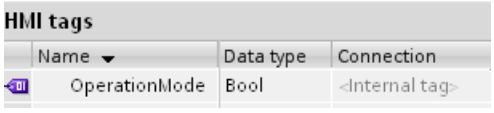
Name	PLC connection	Type
OperatingMode	No	Bool

Text list:

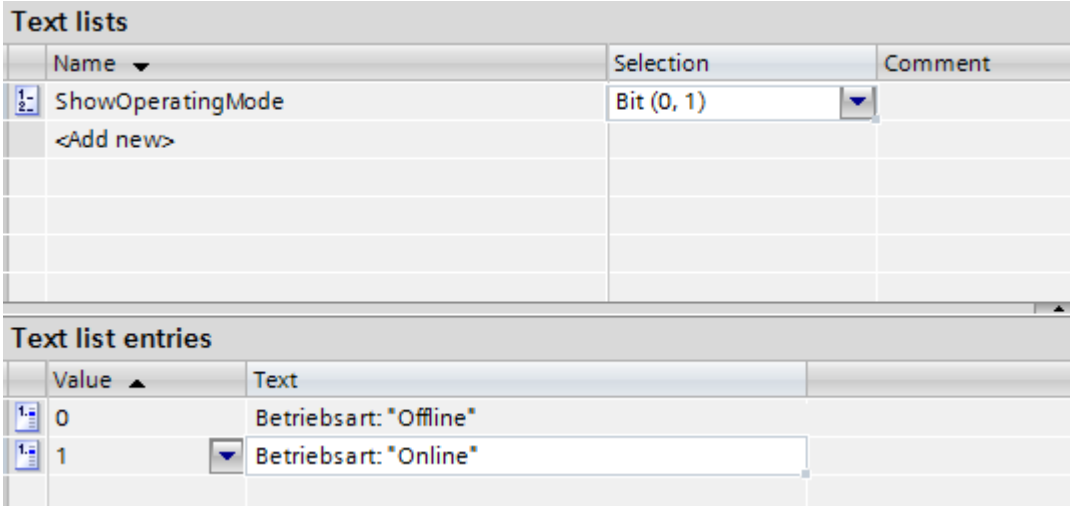
Name	Contains	Values
ShowOperatingMode	Bit (0/1)	1: Operating mode: "Online" 0: Operating mode: "Offline"

Procedure

1. Create the "OperatingMode" HMI-tag indicated above.

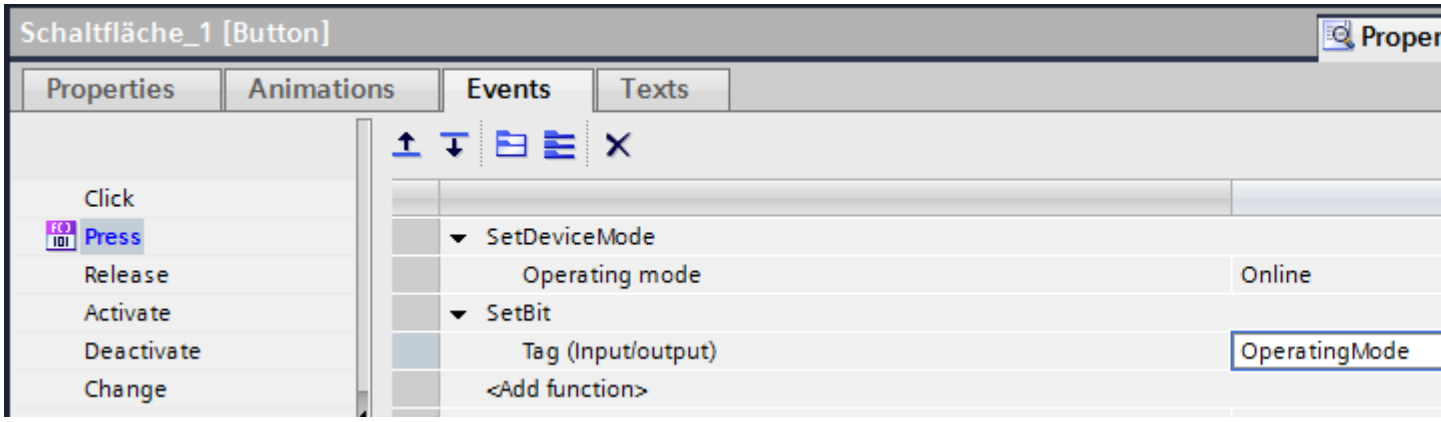


2. Create the "ShowOperatingMode" text list indicated above.

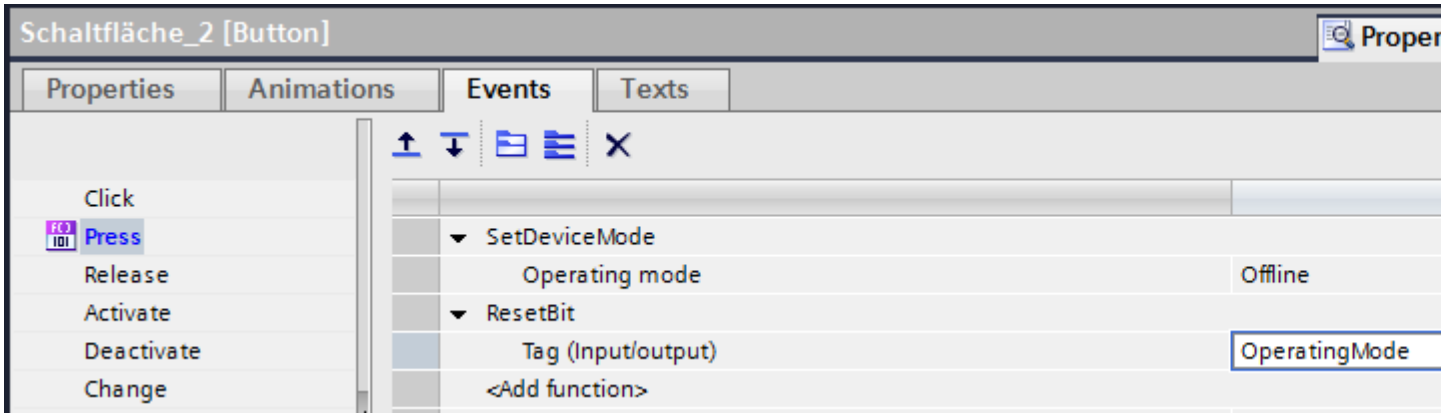


3. Open the screen and insert a button for which you configure the operating mode change to "online".
4. Click "Properties> Events" in the Inspector window. Select the "Press" event.
5. Configure the "SetDeviceMode" system function for the "Press" event. The system function is found in the selection list under "Settings".
6. For the "Mode" parameter, select the "Online" entry.
7. Configure the system function "SetBit" on the event "Press". The system function is found in the selection list under "Bit processing".

8. Select the HMI-tag "Operating mode" from the selection list for the parameter "Tag".



9. Add a button in the process screen for which you configure the operating mode change to "offline".
10. Repeat steps four to seven. For the "Mode" parameter, select the "Offline" entry. Configure the system function "ResetBit" in place of the system function "SetBit."



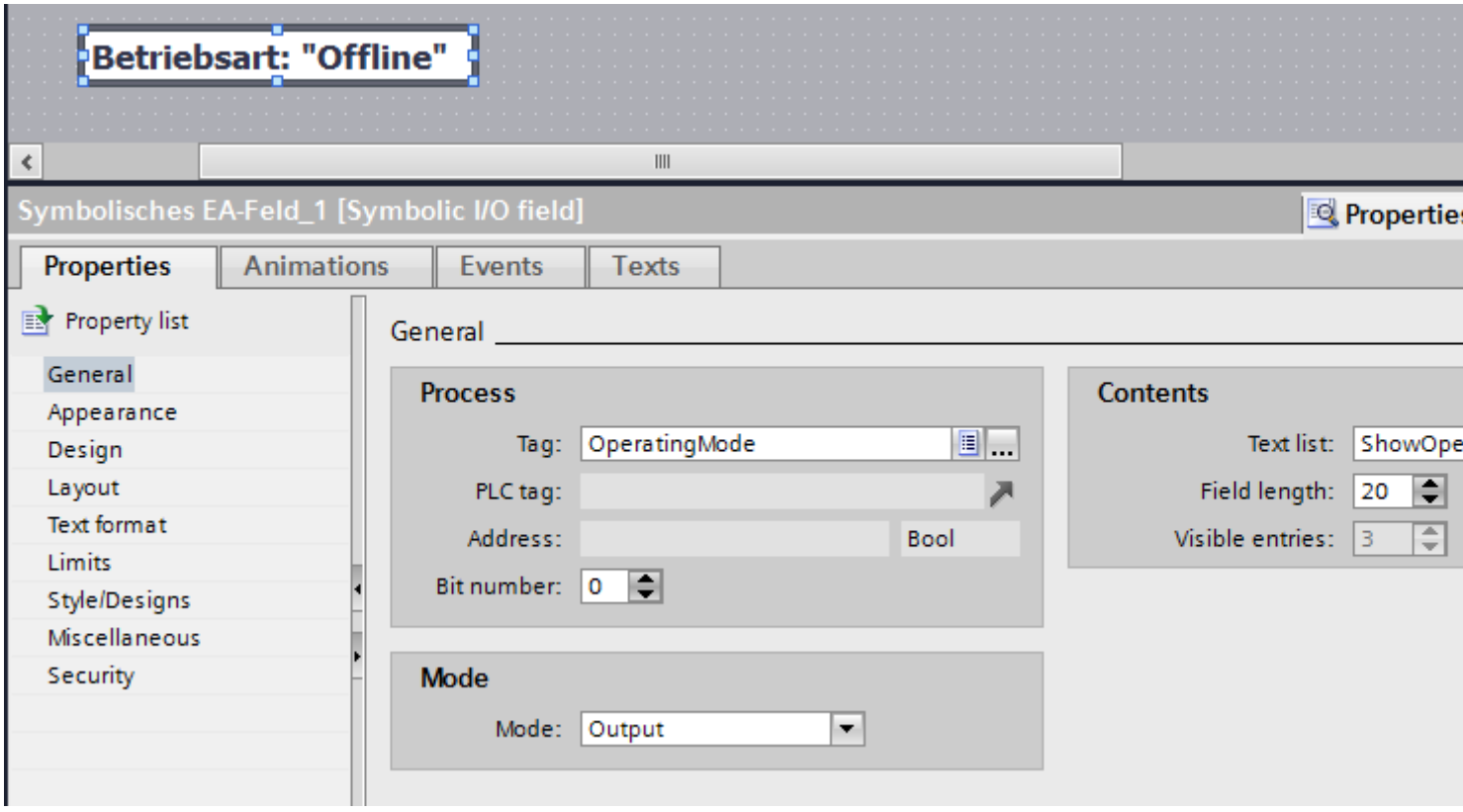
Interim result

You can toggle the operating mode of the HMI device with the two buttons in Runtime.

You want to display the current set operating mode in an output field on the HMI device.

Procedure

1. Create a "Symbolic I/O field" in the process image. Click "Properties > Properties" in the Inspector window.
2. Make the following settings in the "General" group:
   - Select "Output" as the "Mode".
   - Select the text list "Show operating mode" as "Text list".
   - Select "Operating mode" as "Tag".



Result

When you change the operating mode with the buttons, the currently set operating mode on the HMI device is always shown.

12.7.4 Reference

12.7.4.1 Function list

Device dependency of system functions

Device dependency of system functions for Basic Panels (Basic Panels)

Availability of system functions

The following table shows the availability of the system functions on the Basic Panels.

Technical data subject to change.

Overview

	KP300 Basic PN	KP400 Basic PN	KTP600 Basic PN / DP	KTP1000 Basic PN / DP	TP1500 Basic PN
User-defined functions	No	No	No	No	No
Logoff (Page 5688)	Yes	Yes	Yes	Yes	Yes
ActivateScreen (Page 5671)	Yes	Yes	Yes	Yes	Yes
ActivateScreenByNumber (Page 5672)	Yes	Yes	Yes	Yes	Yes
ActivateCleanScreen (Page 5743)	No	No	Yes	Yes	Yes
ActivatePreviousScreen (Page 5673)	Yes	Yes	Yes	Yes	Yes
UpdateTag (Page 5674)	Yes	Yes	Yes	Yes	Yes
AdjustContrast (Page 5749)	Yes	No	Yes [1)]	Yes	Yes
Logon (Page 5688)	Yes	Yes	Yes	Yes	Yes
ArchiveLogFile	No	No	No	No	No
LogTag (Page 5682)	No	No	No	No	No
EditAlarm (Page 5718)	Yes	Yes	Yes	Yes	No
ScreenObjectCursorDown (Page 5745)	Yes	Yes	Yes	Yes	Yes
ScreenObjectCursorUp (Page 5744)	Yes	Yes	Yes	Yes	Yes
ScreenObjectCursorLeft (Page 5745)	No	No	No	No	No
ScreenObjectCursorRight (Page 5746)	No	No	No	No	No
ScreenObjectPageDown (Page 5747)	Yes	Yes	Yes	Yes	Yes
ScreenObjectPageUp (Page 5746)	Yes	Yes	Yes	Yes	Yes
Encode	No	No	No	No	No
EncodeEx	No	No	No	No	No
Direct key	No	No	No	No	No
DirectKeyScreenNumber	No	No	No	No	No

	KP300 Basic PN	KP400 Basic PN	KTP600 Basic PN / DP	KTP1000 Basic PN / DP	TP1500 Basic PN
PrintScreen	No	No	No	No	No
PrintReport	No	No	No	No	No
NotifyUserAction	No	No	No	No	No
IncreaseFocusedValue (Page 5674)	Yes	Yes	Yes	Yes	No
IncreaseTag (Page 5692)	Yes	Yes	Yes	Yes	Yes
ExportDataRecords (Page 5723)	Yes	Yes	Yes	Yes	Yes
ExportDataRecordsWithChecksum	No	No	No	No	No
ExportImportUserAdministration	No	No	No	No	No
GoToHome (Page 5748)	Yes	Yes	Yes	Yes	No
GoToEnd (Page 5749)	Yes	Yes	Yes	Yes	No
SafelyRemoveHardware	No	No	No	No	No
HTMLBrowserStop	No	No	No	No	No
HTMLBrowserScrollDown	No	No	No	No	No
HTMLBrowserRefresh	No	No	No	No	No
HTMLBrowserScrollUp	No	No	No	No	No
HTMLBrowserForward	No	No	No	No	No
HTMLBrowserBack	No	No	No	No	No
HTMLBrowserZoomIn	No	No	No	No	No
HTMLBrowserZoomOut	No	No	No	No	No
HTMLBrowserScrollLeft	No	No	No	No	No
HTMLBrowserScrollRight	No	No	No	No	No
HTMLBrowserPageUp	No	No	No	No	No
HTMLBrowserPageDown	No	No	No	No	No
HTMLBrowserHome	No	No	No	No	No
ImportDataRecords (Page 5726)	Yes	Yes	Yes	Yes	Yes
ImportDataRecordsWithChecksum	No	No	No	No	No
InvertBit (Page 5698)	Yes	Yes	Yes	Yes	Yes
InvertBitInTag (Page 5700)	Yes	Yes	Yes	Yes	Yes
InvertLinearScaling (Page 5693)	Yes	Yes	Yes	Yes	Yes
CalibrateTouchScreen (Page 5750)	No	Yes	Yes	Yes	Yes
CopyLog	No	No	No	No	No
TrendViewScrollForward (Page 5758)	Yes	Yes	Yes	Yes	Yes
TrendViewScrollBack (Page 5759)	Yes	Yes	Yes	Yes	Yes
TrendViewExtend (Page 5759)	Yes	Yes	Yes	Yes	Yes
TrendViewCompress (Page 5760)	Yes	Yes	Yes	Yes	Yes
TrendViewRulerLeft (Page 5760)	Yes	Yes	Yes	Yes	Yes
TrendViewRulerRight (Page 5761)	Yes	Yes	Yes	Yes	Yes
TrendViewSetRulerMode (Page 5762)	Yes	Yes	Yes	Yes	Yes
TrendViewStartStop (Page 5762)	Yes	Yes	Yes	Yes	Yes
TrendViewBackToBeginning (Page 5763)	Yes	Yes	Yes	Yes	Yes

	KP300 Basic PN	KP400 Basic PN	KTP600 Basic PN / DP	KTP1000 Basic PN / DP	TP1500 Basic PN
LoadDataRecord (Page 5728)	Yes	Yes	Yes	Yes	Yes
GetUserName (Page 5689)	Yes	Yes	Yes	Yes	Yes
GetDataRecordFromPLC (Page 5728)	Yes	Yes	Yes	Yes	Yes
GetDataRecordName	No	No	No	No	No
GetDataRecordTagsFromPLC (Page 5730)	Yes	Yes	Yes	Yes	Yes
GetGroupNumber (Page 5689)	Yes	Yes	Yes	Yes	Yes
GetBrightness	No	No	No	No	No
GetPassword (Page 5690)	Yes	Yes	Yes	Yes	Yes
GetPLCMode	Yes	Yes	Yes	Yes	Yes
LinearScaling (Page 5694)	Yes	Yes	Yes	Yes	Yes
ClearLog	No	No	No	No	No
ClearDataRecord (Page 5731)	Yes	Yes	Yes	Yes	Yes
ClearDataRecordMemory	No	No	No	No	No
ClearAlarmBuffer (Page 5718)	Yes	Yes	Yes	Yes	Yes
ClearAlarmBufferProTool (Page 5719)	Yes	Yes	Yes	Yes	Yes
AlarmViewUpdate	No	No	No	No	No
AlarmViewLoopInAlarm	Yes	Yes	Yes	Yes	Yes
AlarmViewAcknowledgeAlarm (Page 5721)	Yes	Yes	Yes	Yes	Yes
AlarmViewShowOperatorNotes (Page 5721)	Yes	Yes	Yes	Yes	Yes
OpenAllLogs	No	No	No	No	No
OpenScreenKeyboard	No	No	No	No	No
OpenControlPanelDialog (Page 5675)	Yes [2)]	Yes [2)]	Yes [2)]	Yes [2)]	Yes [2)]
OpenCommandPrompt	No	No	No	No	No
OpenInternetExplorer	No	No	No	No	No
OpenControlPanel	No	No	No	No	No
OpenTaskManager	No	No	No	No	No
AcknowledgeAlarm (Page 5722)	Yes	Yes	Yes	Yes	No
PDFScrollDown	No	No	No	No	No
PDFScrollUp	No	No	No	No	No
PDFFitToWidth	No	No	No	No	No
PDFFitToHeight	No	No	No	No	No
PDFGoToFirstPage	No	No	No	No	No
PDFGoToLastPage	No	No	No	No	No
PDFGoToNextPage	No	No	No	No	No
PDFGoToPage	No	No	No	No	No
PDFGoToPreviousPage	No	No	No	No	No
PDFZoomIn	No	No	No	No	No
PDFZoomOut	No	No	No	No	No
PDFScrollLeft	No	No	No	No	No

	KP300 Basic PN	KP400 Basic PN	KTP600 Basic PN / DP	KTP1000 Basic PN / DP	TP1500 Basic PN
PDFScrollRight	No	No	No	No	No
PDFZoomOriginal	No	No	No	No	No
RecipeViewNewDataRecord (Page 5733)	Yes	Yes	Yes	Yes	Yes
RecipeViewGetDataRecordFromPLC (Page 5734)	Yes	Yes	Yes	Yes	Yes
RecipeViewClearDataRecord (Page 5734)	Yes	Yes	Yes	Yes	Yes
RecipeViewMenu (Page 5735)	Yes	Yes	Yes	Yes	Yes
RecipeViewOpen (Page 5735)	Yes	Yes	Yes	Yes	Yes
RecipeViewSetDataRecordToPLC (Page 5736)	Yes	Yes	Yes	Yes	Yes
RecipeViewSaveDataRecord (Page 5734)	Yes	Yes	Yes	Yes	Yes
RecipeViewSaveAsDataRecord (Page 5737)	Yes	Yes	Yes	Yes	Yes
RecipeViewSynchronizeDataRecordWith-Tags (Page 5737)	Yes	Yes	Yes	Yes	Yes
RecipeViewRenameDataRecord (Page 5738)	Yes	Yes	Yes	Yes	Yes
RecipeViewShowOperatorNotes (Page 5738)	Yes	Yes	Yes	Yes	Yes
RecipeViewBack (Page 5738)	Yes	Yes	Yes	Yes	Yes
ResetBit (Page 5701)	Yes	Yes	Yes	Yes	Yes
ResetBitInTag (Page 5703)	Yes	Yes	Yes	Yes	Yes
PressButton (Page 5676)	Yes	Yes	Yes	Yes	No
ReleaseButton (Page 5677)	Yes	Yes	Yes	Yes	No
ShiftAndMask (Page 5705)	Yes	Yes	Yes	Yes	Yes
CloseAllLogs	No	No	No	No	No
SetDataRecordToPLC (Page 5739)	Yes	Yes	Yes	Yes	Yes
SetDataRecordTagsToPLC (Page 5740)	Yes	Yes	Yes	Yes	Yes
PageDown (Page 5748)	Yes	Yes	Yes	Yes	No
PageUp (Page 5747)	Yes	Yes	Yes	Yes	No
SendEMail	No	No	No	No	No
SetAcousticSignal	No	No	No	No	No
SetDisplayMode	No	No	No	No	No
SetDeviceMode (Page 5712)	Yes	Yes	Yes	Yes	Yes
SetBit (Page 5706)	Yes	Yes	Yes	Yes	Yes
SetBitInTag (Page 5708)	Yes	Yes	Yes	Yes	Yes
SetBitWhileKeyPressed (Page 5710)	Yes	Yes	Yes	Yes	Yes
SetBacklightColor (Page 5752)	Yes	Yes	No	No	No
SetBrightness (Page 5751)	No	Yes	Yes	No	No
SetScreenKeyboardMode	No	No	No	No	No
SetAlarmReportMode	No	No	No	No	No

	KP300 Basic PN	KP400 Basic PN	KTP600 Basic PN / DP	KTP1000 Basic PN / DP	TP1500 Basic PN
SetRecipeTags (Page 5741)	Yes	Yes	Yes	Yes	Yes
SetDaylightSavingTime	No	No	No	No	No
SetLanguage (Page 5714)	Yes	Yes	Yes	Yes	Yes
SetTag (Page 5695)	Yes	Yes	Yes	Yes	Yes
SetConnectionMode (Page 5715)	Yes	Yes	Yes	Yes	Yes
SetWebAccess	No	No	No	No	No
BackupRAMFileSystem	No	No	No	No	No
SimulateSystemKey (Page 5680)	Yes	Yes	Yes	Yes	No
SimulateTag (Page 5678)	Yes	Yes	Yes	Yes	Yes
SmartClientViewRefresh	No	No	No	No	No
SmartClientViewReadOnlyOff	No	No	No	No	No
SmartClientViewReadOnlyOn	No	No	No	No	No
SmartClientViewDisconnect	No	No	No	No	No
SmartClientViewConnect	No	No	No	No	No
SmartClientViewLeave	No	No	No	No	No
SaveDataRecord (Page 5742)	Yes	Yes	Yes	Yes	Yes
StartLogging	No	No	No	No	No
StartNextLog	No	No	No	No	No
StartProgram	No	No	No	No	No
StatusForceGetValues	No	No	No	No	No
StatusForceSetValues	No	No	No	No	No
ControlSmartServer	No	No	No	No	No
ControlWebServer	No	No	No	No	No
StopLogging	No	No	No	No	No
StopRuntime (Page 5679)	Yes	Yes	Yes	Yes	Yes
SystemDiagnosticViewUpdatePLCBuffer	Yes	Yes	Yes	Yes	Yes
ActivateSystemDiagnosticsView	No	No	No	No	No
SystemDiagnosticViewDetailView	Yes	Yes	Yes	Yes	Yes
SystemDiagnosticsViewDiagnosticBuffer	No	No	No	No	No
SystemDiagnosticViewDeviceView	No	No	No	No	No
SystemDiagnosticsViewBack	Yes	Yes	Yes	Yes	Yes
LookupText	No	No	No	No	No
ResetTagToHandWheel	No	No	No	No	No
SetTagToHandWheel	No	No	No	No	No
TraceUserChange (Page 5691)	Yes	Yes	Yes	Yes	Yes
DecreaseFocusedValue (Page 5680)	Yes	Yes	Yes	Yes	No
DecreaseTag (Page 5697)	Yes	Yes	Yes	Yes	Yes
ChangeConnection (Page 5716)	Yes	Yes	Yes	Yes	Yes
ChangeConnectionEIP	No	No	No	No	No
WinACMPUpdateStartupCharacteristics	No	No	No	No	No

	KP300 Basic PN	KP400 Basic PN	KTP600 Basic PN / DP	KTP1000 Basic PN / DP	TP1500 Basic PN
WinACMPUpdateKeySwitchSetting	No	No	No	No	No
WinACMPUpdateBUSF1LED	No	No	No	No	No
WinACMPUpdateBUSF2LED	No	No	No	No	No
WinACMPUpdateAverageExecTime	No	No	No	No	No
WinACMPUpdateAverageCycleTime	No	No	No	No	No
WinACMPUpdateEXTFLED	No	No	No	No	No
WinACMPUpdateHMIEnableTime	No	No	No	No	No
WinACMPUpdateINTFLED	No	No	No	No	No
WinACMPUpdateLastCycleTime	No	No	No	No	No
WinACMPUpdateMaximumCycleTime	No	No	No	No	No
WinACMPUpdateMinimumCycleTime	No	No	No	No	No
WinACMPUpdatePowerLED	No	No	No	No	No
WinACMPUpdateSleepTime	No	No	No	No	No
WinACMPUpdateRUNLED	No	No	No	No	No
WinACMPUpdateSTOPLED	No	No	No	No	No
WinACMPArchive	No	No	No	No	No
WinACMPGetStartMode	No	No	No	No	No
WinACMPGetVersion	No	No	No	No	No
WinACMPClearCycleTimeBuffer	No	No	No	No	No
WinACMPSetStartAtBoot	No	No	No	No	No
WinACMPSetKeySwitch	No	No	No	No	No
WinACMPSetHMIExecutionTime	No	No	No	No	No
WinACMPSetResetMethod	No	No	No	No	No
WinACMPSetSleeptime	No	No	No	No	No
WinACMPSetStartMode	No	No	No	No	No
WinACMPStartHistogram	No	No	No	No	No
WinACMPControl	No	No	No	No	No
WinACMPStopHistogram	No	No	No	No	No
WinACMPRestore	No	No	No	No	No
ShowLogonDialog (Page 5691)	Yes	Yes	Yes	Yes	Yes
ShowOperatorNotes (Page 5681)	Yes	Yes	Yes	Yes	Yes
ShowAlarmWindow (Page 5723)	Yes	Yes	Yes	Yes	Yes
ShowPopupScreen	No	No	No	No	No
ShowSlideInScreen	No	No	No	No	No
ShowSoftwareVersion	No	No	No	No	No
ShowSystemDiagnosticsWindow	No	No	No	No	No
ShowSystemEvent	No	No	No	No	No

1) For KTP600 Basic mono PN only
2) Only for editing the IP settings

Device dependency of system functions for 2nd generation Basic Panels (Basic Panels)

Availability of system functions

The following table shows the availability of the system functions on the Basic Panels.

Technical data subject to change.

Overview

	KTP400 Basic PN	KTP700 Basic PN / DP	KTP900 Basic PN	KTP1200 Basic PN / DP
User-defined functions	No	No	No	No
Logoff (Page 5688)	Yes	Yes	Yes	Yes
ActivateScreen (Page 5671)	Yes	Yes	Yes	Yes
ActivateScreenByNumber (Page 5672)	Yes	Yes	Yes	Yes
ActivateCleanScreen (Page 5743)	Yes	Yes	Yes	Yes
ActivatePreviousScreen (Page 5673)	Yes	Yes	Yes	Yes
UpdateTag (Page 5674)	Yes	Yes	Yes	Yes
AdjustContrast (Page 5749)	No	No	No	No
Logon (Page 5688)	Yes	Yes	Yes	Yes
ArchiveLogFile	No	No	No	No
LogTag (Page 5682)	Yes	Yes	Yes	Yes
EditAlarm (Page 5718)	Yes	Yes	Yes	Yes
ScreenObjectCursorDown (Page 5745)	Yes	Yes	Yes	Yes
ScreenObjectCursorUp (Page 5744)	Yes	Yes	Yes	Yes
ScreenObjectCursorLeft (Page 5745)	Yes	Yes	Yes	Yes
ScreenObjectCursorRight (Page 5746)	Yes	Yes	Yes	Yes
ScreenObjectPageDown (Page 5747)	Yes	Yes	Yes	Yes
ScreenObjectPageUp (Page 5746)	Yes	Yes	Yes	Yes
Encode	No	No	No	No
EncodeEx	No	No	No	No
Direct key	No	No	No	No
DirectKeyScreenNumber	No	No	No	No
PrintScreen	No	No	No	No
PrintReport	No	No	No	No
NotifyUserAction	No	No	No	No
IncreaseFocusedValue (Page 5674)	Yes	Yes	Yes	Yes
IncreaseTag (Page 5692)	Yes	Yes	Yes	Yes

	KTP400 Basic PN	KTP700 Basic PN / DP	KTP900 Basic PN	KTP1200 Basic PN / DP
ExportDataRecords (Page 5723)	Yes	Yes	Yes	Yes
ExportDataRecordsWithChecksum	No	No	No	No
ExportImportUserAdministration	No	No	No	No
GoToHome (Page 5748)	Yes	Yes	Yes	Yes
GoToEnd (Page 5749)	Yes	Yes	Yes	Yes
SafelyRemoveHardware (Page 5675)	Yes	Yes	Yes	Yes
HTMLBrowserStop (Page 5753)	Yes	Yes	Yes	Yes
HTMLBrowserScrollDown (Page 5753)	Yes	Yes	Yes	Yes
HTMLBrowserRefresh (Page 5754)	Yes	Yes	Yes	Yes
HTMLBrowserScrollUp (Page 5754)	Yes	Yes	Yes	Yes
HTMLBrowserZoomIn (Page 5754)	Yes	Yes	Yes	Yes
HTMLBrowserZoomOut (Page 5755)	Yes	Yes	Yes	Yes
HTMLBrowserScrollLeft (Page 5755)	Yes	Yes	Yes	Yes
HTMLBrowserScrollRight (Page 5756)	Yes	Yes	Yes	Yes
HTMLBrowserPageUp (Page 5757)	Yes	Yes	Yes	Yes
HTMLBrowserPageDown (Page 5756)	Yes	Yes	Yes	Yes
HTMLBrowserHome (Page 5757)	Yes	Yes	Yes	Yes
HTMLBrowserForward (Page 5757)	Yes	Yes	Yes	Yes
HTMLBrowserBack (Page 5758)	Yes	Yes	Yes	Yes
ImportDataRecords (Page 5726)	Yes	Yes	Yes	Yes
ImportDataRecordsWithChecksum	No	No	No	No
InvertBit (Page 5698)	Yes	Yes	Yes	Yes
InvertBitInTag (Page 5700)	Yes	Yes	Yes	Yes
InvertLinearScaling (Page 5693)	Yes	Yes	Yes	Yes
CalibrateTouchScreen (Page 5750)	Yes	Yes	Yes	Yes
CopyLog (Page 5757)	No	No	No	No
TrendViewScrollForward (Page 5758)	Yes	Yes	Yes	Yes
TrendViewScrollBack (Page 5759)	Yes	Yes	Yes	Yes
TrendViewExtend (Page 5759)	Yes	Yes	Yes	Yes
TrendViewCompress (Page 5760)	Yes	Yes	Yes	Yes
TrendViewRulerLeft (Page 5760)	Yes	Yes	Yes	Yes
TrendViewRulerRight (Page 5761)	Yes	Yes	Yes	Yes
TrendViewSetRulerMode (Page 5762)	Yes	Yes	Yes	Yes
TrendViewStartStop (Page 5762)	Yes	Yes	Yes	Yes
TrendViewBackToBeginning (Page 5763)	Yes	Yes	Yes	Yes
LoadDataRecord (Page 5728)	Yes	Yes	Yes	Yes
GetUserName (Page 5689)	Yes	Yes	Yes	Yes
GetDataRecordFromPLC (Page 5728)	Yes	Yes	Yes	Yes
GetDataRecordName	No	No	No	No
GetDataRecordTagsFromPLC (Page 5730)	Yes	Yes	Yes	Yes

	KTP400 Basic PN	KTP700 Basic PN / DP	KTP900 Basic PN	KTP1200 Basic PN / DP
GetGroupNumber (Page 5689)	Yes	Yes	Yes	Yes
GetBrightness	No	No	No	No
GetPassword (Page 5690)	No [1]	No [1]	No [1]	No [1]
AUTOHOTSPOT	Yes	Yes	Yes	Yes
LinearScaling (Page 5694)	Yes	Yes	Yes	Yes
ClearLog (Page 5683)	Yes	Yes	Yes	Yes
ClearDataRecord (Page 5731)	Yes	Yes	Yes	Yes
ClearDataRecordMemory	No	No	No	No
ClearAlarmBuffer (Page 5718)	Yes	Yes	Yes	Yes
ClearAlarmBufferProTool (Page 5719)	Yes	Yes	Yes	Yes
AlarmViewUpdate	No	No	No	No
AlarmViewLoopInAlarm	Yes	Yes	Yes	Yes
AlarmViewAcknowledgeAlarm (Page 5721)	Yes	Yes	Yes	Yes
AlarmViewShowOperatorNotes (Page 5721)	Yes	Yes	Yes	Yes
OpenAllLogs (Page 5684)	Yes	Yes	Yes	Yes
OpenScreenKeyboard	No	No	No	No
OpenControlPanelDialog (Page 5675)	Yes [2]	Yes [2]	Yes [2]	Yes [2]
OpenCommandPrompt	No	No	No	No
OpenInternetExplorer	No	No	No	No
OpenControlPanel	No	No	No	No
OpenTaskManager	No	No	No	No
PDFScrollDown	No	No	No	No
PDFFitToWidth	No	No	No	No
PDFFitToHeight	No	No	No	No
PDFScrollUp	No	No	No	No
PDFGoToFirstPage	No	No	No	No
PDFGoToLastPage	No	No	No	No
PDFGoToNextPage	No	No	No	No
PDFGoToPreviousPage	No	No	No	No
PDFGoToPage	No	No	No	No
PDFZoomIn	No	No	No	No
PDFZoomOut	No	No	No	No
PDFScrollLeft	No	No	No	No
PDFScrollRight	No	No	No	No
PDFZoomOriginal	No	No	No	No
AcknowledgeAlarm (Page 5722)	Yes	Yes	Yes	Yes
RecipeViewNewDataRecord (Page 5733)	Yes	Yes	Yes	Yes
RecipeViewGetDataRecordFromPLC (Page 5734)	Yes	Yes	Yes	Yes
RecipeViewClearDataRecord (Page 5734)	Yes	Yes	Yes	Yes
RecipeViewMenu (Page 5735)	Yes	Yes	Yes	Yes

	KTP400 Basic PN	KTP700 Basic PN / DP	KTP900 Basic PN	KTP1200 Basic PN / DP
RecipeViewOpen (Page 5735)	Yes	Yes	Yes	Yes
RecipeViewSetDataRecordToPLC (Page 5736)	Yes	Yes	Yes	Yes
RecipeViewSaveDataRecord (Page 5734)	Yes	Yes	Yes	Yes
RecipeViewSaveAsDataRecord (Page 5737)	Yes	Yes	Yes	Yes
RecipeViewSynchronizeDataRecordWithTags (Page 5737)	Yes	Yes	Yes	Yes
RecipeViewRenameDataRecord (Page 5738)	Yes	Yes	Yes	Yes
RecipeViewShowOperatorNotes (Page 5738)	Yes	Yes	Yes	Yes
RecipeViewBack (Page 5738)	Yes	Yes	Yes	Yes
ResetBit (Page 5701)	Yes	Yes	Yes	Yes
ResetBitInTag (Page 5703)	Yes	Yes	Yes	Yes
PressButton (Page 5676)	Yes	Yes	Yes	Yes
ReleaseButton (Page 5677)	Yes	Yes	Yes	Yes
ShiftAndMask (Page 5705)	Yes	Yes	Yes	Yes
CloseAllLogs (Page 5685)	Yes	Yes	Yes	Yes
SetDataRecordToPLC (Page 5739)	Yes	Yes	Yes	Yes
SetDataRecordTagsToPLC (Page 5740)	Yes	Yes	Yes	Yes
PageDown (Page 5748)	Yes	Yes	Yes	Yes
PageUp (Page 5747)	Yes	Yes	Yes	Yes
SendEMail	No	No	No	No
SetAcousticSignal	No	No	No	No
SetDisplayMode	No	No	No	No
SetDeviceMode (Page 5712)	Yes	Yes	Yes	Yes
SetBit (Page 5706)	Yes	Yes	Yes	Yes
SetBitInTag (Page 5708)	Yes	Yes	Yes	Yes
SetBitWhileKeyPressed (Page 5710)	Yes	Yes	Yes	Yes
SetBacklightColor	No	No	No	No
SetBrightness (Page 5751)	Yes	Yes	Yes	Yes
SetScreenKeyboardMode	No	No	No	No
SetAlarmReportMode	No	No	No	No
SetRecipeTags (Page 5741)	Yes	Yes	Yes	Yes
SetDaylightSavingTime	No	No	No	No
SetLanguage (Page 5714)	Yes	Yes	Yes	Yes
SetTag (Page 5695)	Yes	Yes	Yes	Yes
SetConnectionMode (Page 5715)	Yes	Yes	Yes	Yes
SetWebAccess	No	No	No	No
BackupRAMFileSystem	No	No	No	No
SimulateSystemKey (Page 5680)	Yes	Yes	Yes	Yes
SimulateTag (Page 5678)	Yes	Yes	Yes	Yes
SmartClientViewRefresh	No	No	No	No
SmartClientViewReadOnlyOff	No	No	No	No

	KTP400 Basic PN	KTP700 Basic PN / DP	KTP900 Basic PN	KTP1200 Basic PN / DP
SmartClientViewReadOnlyOn	No	No	No	No
SmartClientViewDisconnect	No	No	No	No
SmartClientViewConnect	No	No	No	No
SmartClientViewLeave	No	No	No	No
SaveDataRecord (Page 5742)	Yes	Yes	Yes	Yes
StartLogging (Page 5686)	Yes	Yes	Yes	Yes
StartNextLog	No	No	No	No
StartProgram	No	No	No	No
StatusForceGetValues	No	No	No	No
StatusForceSetValues	No	No	No	No
ControlSmartServer	No	No	No	No
ControlWebServer	No	No	No	No
StopLogging (Page 5686)	Yes	Yes	Yes	Yes
StopRuntime (Page 5679)	Yes	Yes	Yes	Yes
ActivateSystemDiagnosticsView	No	No	No	No
SystemDiagnosticViewUpdatePLCBuffer	Yes	Yes	Yes	Yes
SystemDiagnosticsViewDetailView	Yes	Yes	Yes	Yes
SystemDiagnosticsViewDiagnosticBuffer	No	No	No	No
SystemDiagnosticViewDeviceView	No	No	No	No
SystemDiagnosticsViewBack	Yes	Yes	Yes	Yes
LookupText	No	No	No	No
ResetTagToHandWheel	No	No	No	No
SetTagToHandWheel	No	No	No	No
TraceUserChange (Page 5691)	Yes	Yes	Yes	Yes
DecreaseFocusedValue (Page 5680)	Yes	Yes	Yes	Yes
DecreaseTag (Page 5697)	Yes	Yes	Yes	Yes
ChangeConnection (Page 5716)	Yes	Yes	Yes	Yes
ChangeConnectionEIP	No	No	No	No
WinACMPUpdateStartupCharacteristics	No	No	No	No
WinACMPUpdateKeySwitchSetting	No	No	No	No
WinACMPUpdateBUSF1LED	No	No	No	No
WinACMPUpdateBUSF2LED	No	No	No	No
WinACMPUpdateAverageExecTime	No	No	No	No
WinACMPUpdateAverageCycleTime	No	No	No	No
WinACMPUpdateEXTFLED	No	No	No	No
WinACMPUpdateHMIEnableTime	No	No	No	No
WinACMPUpdateINTFLED	No	No	No	No
WinACMPUpdateLastCycleTime	No	No	No	No
WinACMPUpdateMaximumCycleTime	No	No	No	No
WinACMPUpdateMinimumCycleTime	No	No	No	No

	KTP400 Basic PN	KTP700 Basic PN / DP	KTP900 Basic PN	KTP1200 Basic PN / DP
WinACMPUpdatePowerLED	No	No	No	No
WinACMPUpdateSleepTime	No	No	No	No
WinACMPUpdateRUNLED	No	No	No	No
WinACMPUpdateSTOPLED	No	No	No	No
WinACMPArchive	No	No	No	No
WinACMPGetStartMode	No	No	No	No
WinACMPGetVersion	No	No	No	No
WinACMPClearCycleTimeBuffer	No	No	No	No
WinACMPSetStartAtBoot	No	No	No	No
WinACMPSetKeySwitch	No	No	No	No
WinACMPSetHMIExecutionTime	No	No	No	No
WinACMPSetResetMethod	No	No	No	No
WinACMPSetSleeptime	No	No	No	No
WinACMPSetStartMode	No	No	No	No
WinACMPStartHistogram	No	No	No	No
WinACMPControl	No	No	No	No
WinACMPStopHistogram	No	No	No	No
WinACMPRestore	No	No	No	No
ShowLogonDialog (Page 5691)	Yes	Yes	Yes	Yes
ShowOperatorNotes (Page 5681)	Yes	Yes	Yes	Yes
ShowAlarmWindow (Page 5723)	Yes	Yes	Yes	Yes
ShowPopupScreen	No	No	No	No
ShowSlideInScreen	No	No	No	No
ShowSoftwareVersion	No	No	No	No
ShowSystemDiagnosticsWindow	No	No	No	No
ShowSystemEvent	No	No	No	No

[1] Not available for devices as of device version V14.0

[2] Only for editing the IP settings

See also

AlarmViewLoopInAlarm (Page 5720)

SetBacklightColor (Page 5752)

System functions

Other functions

ActivateScreen (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Performs a screen change to the given screen.

Use the "ActivateScreenByNumber" system function to change from the root screen to the permanent area or vice versa.

Use in the function list

ActivateScreen (Screen name, Object number)

Use in user-defined functions

ActivateScreen (Screen_name, Object_number)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Screen name

Name of the screen to which you change.

Object number

The operator control element which receives the focus in the given screen after the screen change. The number of the operator control element is to be determined using the tabulator sequence during configuration.

When "0" is specified:

- If the focus is in the permanent area when the system function is called up, the permanent area maintains the focus.
- If the focus is in the root screen when the system function is called up, the first operator control element in the given screen receives the focus.

Note

If the "Reach margin" event is assigned to the "ActivateScreen" system function, only the value "0" is valid for the "Object number" parameter. The active object is not defined by the object number, but rather by the X position it had prior to the screen change.

Example

The following program code activates "Screen_2" with the ActivateScreen function, for example, when you click a button.

```
Sub ActivateScreen_2()

'User-defined code
'' i. e. when pressing a button

ActivateScreen "Screen_2",0
```

See also

ActivateScreenByNumber (Page 5672)

ActivateScreenByNumber (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Performs a screen change to a screen depending on a tag value.

The screen is identified by its screen number.

Use in the function list

ActivateScreenByNumber (Screen number, Object number)

Use in user-defined functions

ActivateScreenByNumber (Screen_number, Object_number)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Screen number

Tag which contains the screen number of the destination screen.

When a change from the root screen to the permanent area is desired, specify "0" or "-1":

0 = Change from root screen to permanent area.

-1 = Change from permanent area to root screen

Object number

The number of the screen object which receives the focus in the given screen after the screen change. The number of the operator control element is to be determined using the tabulator sequence during configuration.

When "0" is specified:

- If the focus is in the permanent area when the system function is called up, the permanent area maintains the focus.
- If the focus is in the root screen when the system function is called up, the first operator control element in the given screen receives the focus.

See also

ActivateScreen (Page 5671)

ActivatePreviousScreen (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Performs a screen change to the screen which was activated before the current screen. The screen change is not performed if no screen was activated beforehand.

The last 10 screens that were called up are saved. A system alarm is output when you change to a screen which is no longer saved.

Note

If you want to use the system function, the screen to which you change has to be used in the navigation structure.

Use in the function list

ActivatePreviousScreen

Use in user-defined functions

ActivatePreviousScreen

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

UpdateTag (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Reads the current value of the tag with the specified Update ID from the PLC.

Use in the function list

UpdateTag (Update ID)

Use in user-defined functions

-

Parameters

Update ID

Update ID assigned to the tag that will be updated.

See also

Logoff (Page 5688)

IncreaseFocusedValue (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Adds the given value to the value of the tag which is connected to the input field (drop-down list, graphic selection list, slider bar) which has the current focus.

This system function can only be used for function keys.

Use in the function list

IncreaseFocusedValue (Value)

Use in user-defined functions

-

Parameters

Value

The value which is added to the tag value.

SafelyRemoveHardware (Basic Panels, Panels, Comfort Panels)

Description

Checks whether there is read or write access to the external storage medium. If there is no access, the external storage medium can be removed without data loss.

Use in the function list

SafelyRemoveHardware(Path, Result)

Use in user-defined functions

SafelyRemoveHardware(Path, Result)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Result

The tag in which the result is entered.

TRUE : The storage medium can be removed safely. A corresponding system alarm will be output.

FALSE : The storage medium cannot be removed safely. A corresponding system alarm will be output.

Path

Path of storage medium, e.g. \Storage Card USB\

See also

Device dependency of system functions for 2nd generation Basic Panels (Page 5665)

OpenControlPanelDialog (Panels, Comfort Panels, RT Advanced)

Description

Opens a dialog that you can use to edit selected Control Panel settings.

This system function lets you set the following on the HMI device:

- Properties and value of the IP address
- User identification on the network
- WinCC Internet settings

Note

Project security

The "OpenControlPanelDialog" system function is used to bypass the SecureMode on the HMI device. Take the corresponding precautions to ensure the security of your project.

Use in the function list

OpenControlPanelDialog (Dialog)

Use in user-defined functions

-

Parameters

Dialog

Sets the Control Panel dialog to be opened.

- PROFINET_X1: Setting of the IP address and Ethernet parameters
- PROFINET_X3: Setting of the IP address and Ethernet parameters; only for Comfort Panel KP 1500, TP 1500; TP1900, TP2200
- WinCC Internet settings: Setting for Web Server, e-mail notification, provided the HMI device supports these functions
- Network ID: Setting for identification on the network, provided the HMI device supports these functions

See also

Device dependency of system functions for 2nd generation Basic Panels (Page 5665)

PressButton (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

You configure the system function to the function keys of an HMI device. The system function "PressButton" triggers the function that is assigned to the event "PressKey" at the specified screen object.

Use this system function when you want to operate a button in a screen with a function key of the HMI device, for example.

Note

The "PressButton" and "ReleaseButton" system functions must always be configured together. If you configure the "PressButton" system function on the "Press key" event for a function key, the "ReleaseButton" system function is configured on the "Release" event for the same function key.

Use in the function list

PressButton (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the screen object on which the event is triggered.

ReleaseButton (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

The system function can only be configured on the function keys of an HMI device and triggers the "Release button" event at the specified screen object.

Use this system function when you want to operate a button in a screen with a function key of the HMI device, for example.

Note

The "PressButton" and "ReleaseButton" system functions must always be configured together. If you configure the "PressButton" system function on the "Press key" event for a function key, then the "ReleaseButton" system function is configured on the "Release key" event for the same function key.

Use in the function list

ReleaseButton (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the screen object on which the event is triggered.

SimulateTag (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Simulates the behavior of tags and dynamic objects such as text lists, without having the HMI device connected to a PLC. You can, for example, configure the system function to the "Loaded" event of a screen.

This system function is used, for example, to demonstrate the functionality of a project.

Only tags of the data type Integer can be used for simulation. Tags of the data types Integer and Double Integer, however, can be used with OP 73, OP 77A, TP 177A.

Note

If you use the system function "SimulateTag" with a short cycle time on a Basic Panel, the HMI device may be overloaded.

Use in the function list

SimulateTag (Tag, Cycle, Maximum value, Minimum value, Value)

Use in user-defined functions

-

Parameter

Tag

The tag whose value is changed.

Cycle

The factor by which the basic cycle of 200 milliseconds is multiplied. The cycle defines when the tag value is changed by the specified value. Possible cycles between 1 and 32767.

Maximum value

The maximum value that the tag can assume during simulation. The maximum value must be greater than the minimum value but less than / equal to 32767.

Minimum value

The minimum value that the tag can assume during simulation. The minimum value must be lower than the maximum value but greater than / equal to -32768.

Value

The value by which the tag value is changed during each cycle. Possible values between -32768 and 32767.

- A positive value increases the tag value. When the maximum value is reached, the tag value is set to the minimum value after the next update cycle.
- A negative value reduces the tag value. When the minimum value is reached, the tag value is set to the maximum value after the next update cycle.

StopRuntime (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Exits the runtime software and thereby the project running on the HMI device.

Use in the function list

StopRuntime (Mode)

Use in user-defined functions

StopRuntime (Mode)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Mode

Determines whether the operating system is shut down after exiting runtime.

0 (hmiStopRuntime) = Runtime: Operating system is not shut down

1 (hmiStopRuntimeAndOperatingSystem) = Runtime and operating system: The operating system is shut down (not possible with WinCE)

Example

The following program code shuts down Runtime and the operating system.

```
{

//Stop runtime and shutdown
StopRuntime (hmiStopRuntimeAndOperationSystem);

}
```

The saved return value can be processed in the following code.

DecreaseFocusedValue (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Subtracts the specified value from the value of the tag which is connected to the screen object and currently has the focus.

This system function can only be used for function keys.

Use in the function list

DecreaseFocusedValue (Value)

Use in user-defined functions

-

Parameters

Value

The value which is subtracted from the tag value.

SimulateSystemKey (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Simulates the behavior of a system key. Use this system function if a system key, such as the "ACK" key, "Input" key or the number pad is not available on the HMI device.

Use in the function list

SimulateSystemKey (System key)

Use in user-defined functions

-

Parameters

System key

System key whose behavior is to be simulated.

System key "+/-"

With the SimulateSystemKey system function, the system key "+/-" is only supported for the following HMI devices:

- KP300 Basic
- KP400 Basic
- KTP400 Basic mono PN
- KTP400 Basic color PN
- KTP400 Basic color PN Portrait
- KTP600 Basic mono PN
- KTP600 Basic color PN
- KTP600 Basic color DP
- KTP700 Basic PN / DP
- KTP900 Basic PN
- KTP1000 Basic PN
- KTP1000 Basic DP
- KTP1200 Basic PN / DP
- Comfort Panels

Use the system keys "+" and "-" separately for all other HMI devices.

ShowOperatorNotes (Basic Panels, Panels, Comfort Panels, RT Advanced)

Application

Displays the tooltip configured for the selected object.

If the system function is configured on a function key, the tooltip for the screen object that currently has the focus is displayed. If a tooltip is configured for the screen itself, you can switch to this text by pressing <Enter> or by double-clicking on the help window.

If the system function is configured on a button, only the tooltip for the current screen is displayed. If a tooltip is configured on the button itself, initially only the tooltip for the button is

displayed. You can press <Enter> or double-click on the help window to switch to the tooltip for the current screen.

Note

No other screen object can be used while the help window is open. To use the screen object, close the help window.

Closing the help window

You can close the help window in the following ways:

Using the keys:

- By pressing the <HELP> key again
- By pressing the <ESC> key

Using the touch screen:

- By pressing the button

Use in the function list

ShowOperatorNotes (Layout)

Use in user-defined functions

ShowOperatorNotes (Display_mode)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Display mode

Determines whether the configured tooltip is hidden or shown:

0 (hmiOff) = Off: Configured tooltip is hidden

1 (hmiOn) = On: Configured tooltip is shown

-1 (hmiToggle) = Toggle: Toggles between the two modes

Logs

LogTag (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Saves the value of the given tags in the given data log.

This system function is used to archive a process value at a certain point in time.

Use in the function list

LogTag (Tag)

Use in user-defined functions

-

Parameters

Tag

The tag whose value is logged. The tag is stored in the log which is configured for the specified tag.

ClearLog (Panels, Comfort Panels, RT Advanced)

Description

Deletes all data records in the given log.

Use in the function list

ClearLog (Log type, Log)

Use in user-defined functions

ClearLog (Log_type, Log)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Log type

Determines the type of log:

0 (hmiTagArchive) = Data log

1 (hmiAlarmArchive) = Alarm log

2 (hmiAuditTrailArchive) = Audit trail log Available for GMP-compliant projects Additional information is available under "Activating GMP-compliant configuration".

Log

Name of the log from which all entries are deleted.

See also

Device dependency of system functions for 2nd generation Basic Panels (Page 5665)

OpenAllLogs (Panels, Comfort Panels, RT Advanced)

Description

Reestablishes the connection between WinCC and the logs. Logging can be continued.

Note

Run the "StartLogging" system function in order to restart logging.

Use in the function list

OpenAllLogs

Use in user-defined functions

OpenAllLogs

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

Application example

You are in runtime and want to change the data medium on which the process values are logged.

Notes on configuring

Configure the "StopLogging" and "CloseAllLogs" system functions on the "Close Archive" button.

Configure the "OpenAllLogs" and "StartLogging" system functions on the "Open Archive" button.

As parameter transfer the respective name of the log that is to be stopped and started.

Procedure on HMI device

When the button "Close Archive" is pressed, the specified log is stopped and all open logs are closed. The data medium can be changed. You open all logs with the "Open Archive" button. Logging will be continued in the specified log.

See also

Device dependency of system functions for 2nd generation Basic Panels (Page 5665)

CloseAllLogs (Panels, Comfort Panels, RT Advanced)

Description

Disconnects the connection between WinCC and all logs.

Use in the function list

CloseAllLogs

Use in user-defined functions

CloseAllLogs

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

Application example

You are in runtime and want to change the data medium on which the process values are logged.

Notes on configuring

Configure the "CloseAllLogs" system function on the "Close Archive" button.

Configure the "OpenAllLogs" and "StartLogging" system functions on the "Open Archive" button.

As parameter transfer the respective name of the log that is to be stopped and started.

Procedure on HMI device

When you press the "Close Archive" button, the opened logs are closed. The data medium can be changed. The logging continues during the change of the data medium. The process values to be logged are cached. You open all logs with the "Open Archive" button. The logging process will continue in the specified log. The buffered process values are later added to the log.

See also

Device dependency of system functions for 2nd generation Basic Panels (Page 5665)

StartLogging (Panels, Comfort Panels, RT Advanced)

Description

Starts the logging of data or alarms in the specified log. The function can also be applied to audit trails.

You can interrupt logging at runtime using the "StopLogging" system function.

Use in the function list

StartLogging (Log type, Log)

Use in user-defined functions

StartLogging (Log_type, Log)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Log type

Determines the type of log:

0 (hmiTagArchive) = Data log

1 (hmiAlarmArchive) = Alarm log

2 (hmiAudittrailArchive) = Audit trail

Log

Name of the log which is started.

See also

Device dependency of system functions for 2nd generation Basic Panels (Page 5665)

StopLogging (Panels, Comfort Panels, RT Advanced)

Description

Stops the logging of process values or alarms in the specified log. The function can also be applied to audit trails.

The "StartLogging" system function is used to resume logging at runtime.

Note

When logging is stopped, a connection between WinCC and the log files or log database still exists. Use the "CloseAllLogs" system function to disconnect this connection.

Use in the function list

StopLogging (Log type, Log)

Use in user-defined functions

StopLogging (Log_type, Log)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Log type

Determines the type of log:

0 (hmiTagArchive) = Data log

1 (hmiAlarmArchive) = Alarm log

2 (hmiAudittrailArchive) = Audit trail

Log

Name of the log that is stopped.

Application example

You are in runtime and want to change the data medium on which the process values are logged.

Notes on configuring

Configure the "StopLogging" and "CloseAllLogs" system functions on the "Close Archive" button.

Configure the "OpenAllLogs" and "StartLogging" system functions on the "Open Archive" button.

As parameter transfer the respective name of the log that is to be stopped and started.

Procedure on HMI device

When the button "Close Archive" is pressed, the specified log is stopped and all open logs are closed. The data medium can be changed. The "Open Archive" button opens all logs and continues logging in the specified log.

See also

Device dependency of system functions for 2nd generation Basic Panels (Page 5665)

User administration

Logoff

Description

Logs off the current user on the HMI device.

Use in the function list

Logoff

Use in user-defined functions

Logoff

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

See also

Logon (Page 5688)

Logon (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Logs on the current user on the HMI device.

Use in the function list

Logon (Password, User name)

Use in user-defined functions

Logon (Password, User_name)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Password

The tag from which the password for the user logging on is read.

If the user is logged on, the password in the tag is deleted.

User name

The tag from which the user name for the user logging on is read.

See also

Logoff (Page 5688)

GetUserName (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Writes the user name of the user currently logged on to the HMI device in the given tag.

If the given tag has a control connection, the user name is also available in the PLC connection. This system function makes it possible, for example, to implement a user-dependent release of certain functionalities.

Use in the function list

GetUserName (Tag)

Use in user-defined functions

GetUserName (Tag)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Tag

The tag to which the user name is written.

GetGroupNumber (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Reads the number of the group to which the user logged on to the HMI device belongs, and writes it to the given tag.

Use in the function list

GetGroupNumber (Tag)

Use in user-defined functions

GetGroupNumber (Tag)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Tag

The tag to which the number of the group is written.

GetPassword (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Writes the password of the user currently logged on to the HMI device in the given tag.

Note

Make sure that the value of the given tag is not displayed in another place in the project.

Note

The passwords of SIMATIC Logon users cannot be read.

Note

The "GetPassword" system function is not available for devices as of device version 14.0.0.0.

If this system feature is used in a project older than version V14, it is removed after migration to V14. The event linked to the system function is thereby invalidated. The "GetPassword" system function appears to be faulty in the configuration. A warning is generated during compiling. The "GetPassword" system function is no longer included in the compiled Runtime project.

Use in the function list

GetPassword (Tag)

Use in user-defined functions

GetPassword (Tag)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Tag

The tag to which the password is written.

TraceUserChange (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Outputs a system event that shows which user is currently logged in on the HMI device.

This system function can only be used in the Scheduler.

Use in the function list

TraceUserChange

Use in user-defined functions

-

Parameters

--

ShowLogonDialog (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Opens a dialog on the HMI device with which the user can log on to the HMI device.

Use in the function list

ShowLogonDialog

Use in user-defined functions

-

Parameters

--

Calculation

IncreaseTag (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Adds the given value to the value of the tags.

X = X + a

Note

The system function uses the same tag as input and output values. When this system function is used to convert a value, help tags must be used. You can use the "SetTag" system function to assign the tag value to the auxiliary tags.

If you configure the system function on the events of an alarm and the tag is not being used in the current screen, it is not ensured that the actual value of the tags is being used in the PLC. You can improve the situation by setting the "Cyclic continuous" acquisition mode.

Use in the function list

IncreaseTag (Tag, Value)

Use in user-defined functions

IncreaseTag (Tag, Value)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Tag

The tag to which the given value is added.

Value

The value that is added.

See also

SetTag (Page 5695)

InvertLinearScaling (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Assigns a value to the tag X, which is calculated from the value of the given tag Y using the linear function X = (Y - b) / a.

The tags X and Y must not be identical. This system function is the inverse of the "LinearScaling" system function.

Note

The tags X and Y must not be identical. If a tag is to be converted into itself, a auxiliary tag must be used.

The "SetTag" system function can be used to assign the value of the tags to be converted to the auxiliary tags.

Use in the function list

InvertLinearScaling (X, Y, b, a)

Use in user-defined functions

InvertLinearScaling (X, Y, b, a)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

X

The tag which is assigned the value calculated from the linear equation.

Y

The tag that contains the value used for calculation.

b

The value which is subtracted.

a

The value through which is divided.

Example

The following program code assigns a value to the varX tag by means of the InverseLinearScaling function.

```
{
BYTE varX;
BYTE Yvalue = 10;
BYTE bvalue = 3;
BYTE avalue = 4;

//Inverse linear scaling
 InverseLinearScaling (varX, Yvalue, bvalue, avalue);

printf ("varX = %d\r\n, varX);
...
}
```

The saved return value can be processed in the following code.

See also

LinearScaling (Page 5694)

LinearScaling (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Assigns a value to the tag Y, which is calculated from the value of the given tag X using the linear function Y= (a *X) + b.

The inverse of this function is the "InvertLinearScaling" system function.

Note

The tags X and Y must not be identical. If a tag is to be converted into itself, a auxiliary tag must be used.

The "SetTag" system function can be used to assign the value of the tags to be converted to the auxiliary tags.

Use in the function list

LinearScaling (Y, a, X, b)

Use in user-defined functions

LinearScaling (Y, a, X, b)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Y

The tag which is assigned the value calculated from the linear equation.

a

The value with which is multiplied.

X

The tag that contains the value used for calculation.

b

The value that is added.

Example

The following program code uses the LinearScaling function to assign a value to the Yvar tag.

```
{
BYTE Yvar;
BYTE Xvalue = 10;
BYTE bvalue = 3;
BYTE avalue = 4;

// linear scaling
 LinearScaling ( Yvar, avalue, Xvalue, bvalue);

printf ("Yvar = %d\r\n, Yvar);
...
}
```

The saved return value can be processed in the following code.

See also

InvertLinearScaling (Page 5693)

SetTag (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Assigns a new value to the given tag.

Note

This system function can be used to assign strings and numbers, depending on the type of tag.

Use in the function list

SetTag (Tag, Value)

Use in user-defined functions

SetTag (Tag, Value)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Tag

The tag to which the given value is assigned.

Value

The value which the given tag is assigned.

Note

The "SetTag" system function is only executed after a connection has been established.

Example

The following program code uses the SetTag function to set the value of the gs_tag_bit tag to TRUE and saves the return value to the ok tag.

```
{
BOOL ok;
BOOL bvalue;

//Set the tag to true
ok = SetTag("gs_tag_bit", TRUE);
//error handling
if(ok)
{
  // succeeded
  printf ( "Function has run through.\r\n" );
bvalue = GetTagBit("gs_tag_bit");
  printf ("Value of gs_tag_bit: %d\r\n", bvalue);
}
else
{
  // failed
  printf ( "Error - function failed." );
}
...
}
```

The saved return value can be processed in the following code.

See also

IncreaseTag (Page 5692)

DecreaseTag (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Subtracts the given value from the tag value.

X = X - a

Note

The system function uses the same tag as input and output values. When this system function is used to convert a value, auxiliary tags must be used. The auxiliary tags are assigned to the tag value with the "SetTag" system function.

If you configure the system function on the events of an alarm and the tag is not being used in the current screen, it is not ensured that the actual value of the tags is being used in the PLC. You can improve the situation by setting the "Cyclic continuous" acquisition mode.

Use in the function list

DecreaseTag (Tag, Value)

Use in user-defined functions

DecreaseTag (Tag, Value)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Tag

The tag from which the given value is subtracted.

Value

The value which is subtracted.

Bit processing

InvertBit (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Inverts the value of the given tag of the "Bool" type:

- If the tag has the value of 1 (TRUE), it will be set to 0 (FALSE).
- If the tag has the value of 0 (FALSE), it will be set to 1 (TRUE).

Use in the function list

InvertBit (Tag)

Use in user-defined functions

InvertBit (Tag)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Tag

The tag whose bit is set.

Example

The following program code inverts the value of the Boolean tag bStatus and outputs the result along with the original bSaved value.

```
'Programming language: VB
Dim myTag
Dim myOutputField
Dim bValue, bSaved, strResult

Set myTag = SmartTags("bStatus")
Set myOutputField=HMIRuntime.Screens("MyScreen").ScreenItems("objTextField")

'Get current value
bValue=myTag.Value

'Save current value
bSaved=bValue

'Invert Bit
InvertBit(myTag)
bValue=myTag.Value

'Output result old and new value:
strResult="Old Value: "&bSaved &Chr(13)&"New Value: "&bValue
myOutputField.Text=strResult
```

```
//Programming language: C
{
BOOL bStatus = 0;
BOOL bSaved = bStatus;

//Invert variable
invertBit(bStatus);

//print current and saved value
printf ("Current value: %d\r\n, Saved value: %d\r\n",bStatus, bSaved);
  ...
}
```

See also

InvertBitInTag (Page 5700)

InvertBitInTag (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Inverts a bit in the given tag:

- If the bit in the tag has the value of 1 (TRUE), it will be set to 0 (FALSE).
- If the bit in the tag has the value of 0 (FALSE), it will be set to 1 (TRUE).

After changing the given bit, the system function transfers the entire tag back to the PLC. It is not checked whether other bits in the tags have changed in the meantime. Operator and PLC have read-only access to the indicated tag until it is transferred back to the PLC.

Note

If the PLC supports BOOL tags, do not use this system function. Use the "InvertBit" system function instead.

Use in the function list

InvertBitInTag (Tag, Bit)

Use in user-defined functions

InvertBitInTag (Tag, Bit)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Tag

The tag in which the given bit is set.

Bit

The number of the bit that is set.

When this system function is used in a user-defined function, the bits in a tag are counted from right to left. The counting begins with 0.

Example

The following program code inverts a bit at the specified bitposition in the bStatusWord tag and outputs the result along with the original bSaved value.

```
'Programming language: VB
Dim myTag
Dim myOutputField
Dim bValue, bSaved, bitposition, strResult

Set myTag = SmartTags("bStatusWord")
Set myOutputField=HMIRuntime.Screens("MyScreen").ScreenItems("objTextField")

'Get current value
bValue=myTag.Value

'Save current value
bSaved=bValue

'Invert Bit in position
bitposition=2
InvertBit(myTag, bitposition)
bValue=myTag.Value

'Output result old and new value:
strResult="Old Value: "&bSaved &Chr(13)&"New Value: "&bValue
myOutputField.Text=strResult
```

```
//Programming language: C
{
BYTE bStatusWord;
BYTE bsaved = bStatusWord;
BYTE bitposition = 2;

//Invert bit in bitposition
InvertBitInTag (bStatusWord, bitposition);
//print current and saved value
printf ("Current value: %d\r\n, Saved value: %d\r\n",bStatusWord, bsaved);
  ...
}
```

See also

InvertBit (Page 5698)

ResetBit (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Sets the value of a "Bool" type tag to 0 (FALSE).

Use in the function list

ResetBit (Tag)

Use in user-defined functions

ResetBit (Tag)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Tag

The BOOL type tag which is set to 0 (FALSE).

Example

The following program code sets the value of the Boolean tag bStatus to 0 by means of the ResetBit function and outputs the result along with the original bSaved value.

```
'Programming language: VB
Dim myTag
Dim myOutputField
Dim bValue, bSaved, strResult

Set myTag = SmartTags("bStatus")
Set myOutputField=HMIRuntime.Screens("MyScreen").ScreenItems("objTextField")

'Set value
bValue=1
myTag.Value=bValue

'Save current value
bSaved=bValue

'Reset Bit
ResetBit(myTag)
bValue=myTag.Value

'Output result old and new value:
strResult="Old Value: "&bSaved &Chr(13)&"New Value: "&bValue
myOutputField.Text=strResult
```

```
//Programming language: C
{
BOOL bStatus = 1;
BOOL bSaved = bStatus;

//Reset bit
  ResetBit (bStatus);

//print current and saved value
printf ("Current value: %d\r\n, Saved value: %d\r\n",bStatus, bSaved);
...
}
```

See also

ResetBitInTag (Page 5703)

ResetBitInTag (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Sets a bit in the specified tag to 0 (FALSE).

After changing the given bit, the system function transfers the entire tag back to the PLC. It is not checked whether other bits in the tags have changed in the meantime. Operator and PLC have read-only access to the indicated tag until it is transferred back to the PLC.

Note

If the PLC supports BOOL tags, do not use this system function. Use the "ResetBit" system function instead.

Use in the function list

ResetBitInTag (Tag, Bit)

Use in user-defined functions

ResetBitInTag (Tag, Bit)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Tag

The tag in which a bit is set to 0 (FALSE).

Bit

The number of the bit that is set to 0 (FALSE).

When this system function is used in a user-defined function, the bits in the specified tag will be counted from right to left independent of the PLC used. The counting begins with 0.

Example

The following program code sets a bit to 0 at the specified bitposition in the bStatusWord tag and outputs the result along with the original bSaved value.

```
'Programming language: VB
Dim myTag
Dim myOutputField
Dim bValue, bSaved, bitposition, strResult

Set myTag = SmartTags("bStatusWord")
Set myOutputField=HMIRuntime.Screens("MyScreen").ScreenItems("objTextField")

'Save current value
bValue=myTag.Value
bSaved=bValue

'Reset Bit
bitposition=2
ResetBit(myTag, bitposition)
bValue=myTag.Value

'Output result old and new value:
strResult="Old Value: "&bSaved &Chr(13)&"New Value: "&bValue
myOutputField.Text=strResult
```

```
//Programming language: C
{
BYTE bStatusWord;
BYTE bSaved = bStatusWord;
BYTE bitposition = 2;

//Reset bit in bitposition
ResetBitInTag (bStatusWord, bitposition);

//print current and saved value
printf ("Current value: %d\r\n, Saved value: %d\r\n",bStatusWord, bSaved);
  ...
}
```

See also

ResetBit (Page 5701)

ShiftAndMask (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

This system function converts the input bit pattern of the source tags into an output bit pattern of the target tags. This involves bit shifting and masking.

Note

If the source and target tag have a different number of bits, using the system function in the target tag can result in a violation of the value range.

Use in the function list

ShiftAndMask (Source tag, Target tag, Bits to shift, Bits to mask)

Use in user-defined functions

ShiftAndMask (Source_tag, Target_tag, Bits_to_shift, Bits_to_mask)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Source tag

The tag includes the input bit pattern. Integer-type tags, e.g. "Byte", "Char", "Int", "UInt", "Long" and "ULong" are permitted.

Example: The actual value 72 is set at the 16-bit integer source tag: 0000000001001000.

Target tag

The output bit pattern is saved in the tag. Integer type tags, e.g. "Byte", "Char", "Int", "UInt", "Long" and "ULong" are permitted.

Example: The shifted input bit pattern is multiplied by the bit mask, with bit-by-bit logical AND operation: 0000000000001001. The resultant decimal value "8" is saved to the target tag.

Please note the following:

- The source and target tags have the same number of bits.
- The number of bits to shift is less than the number of bits in the source tag and target tag.
- Bits to mask does not have more bits than the source tag and the target tag.

Bits to shift

Number of bits by which the input bit pattern is shifted right. A negative value shifts the input bit pattern to the left.

Example: "Bits to shift" has the value "+3". The input bit pattern is shifted right by three bits when the system function is called: 0000000000001001.

Bits to the left are padded with "0". Three bits are truncated on the right. The new decimal value is "9".

Note

The left bit is "1" in a source tag of the data type with negative signed integer. This sign bit is padded with "0" when the bits are shifted right. The sign changes to "+".

Bits to mask

An integer serves as bit mask. The bit pattern is used to multiply the shifted input bit pattern. Example: Integer "2478" with the bit pattern "0000100110101110".

You can enter the bit mask in three different ways:

- Hexadecimal: First enter the prefix "0h" or "0H", followed by an optional space for better readability. Then group the bit pattern in blocks of four (0000)(1001)(1010)(1110) and set each block in hexadecimal code: (0)(9)(A)(E). Only the characters 0-9, A-F, a-f are allowed: "0h 09AE".
- Binary: First enter the prefix "0b" or "0B", followed by an optional space for better readability. Then group the binary bit pattern into blocks of four 0000 1001 1010 1110 with spaces in between as a check. Only the characters "0" or "1" are allowed: "0b 0000 1001 1010 1110".
- Decimal: Enter the value "2478" directly, without a prefix.

Note

If you change the device version of the target HMI device after configuration (e.g."13.1.0" to "13.0.0" or vice versa), you must check and test the parameters of this system function.

You can use the "Char" and "Word" data types for the "Source tag" and "Target tag" parameters as of device version 13.1.0. In the device versions before 13.1.0, these parameters must be assigned other data types:

- Use "SInt" instead of "Char"
- Use "Int" instead of "Word"

Otherwise, there may be unwanted effects, for example incorrect or unexpected behavior of the configured system functions.

SetBit (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Sets the value of a "Bool" type tag to 1 (TRUE).

Use in the function list

SetBit (Tag)

Use in user-defined functions

SetBit (Tag)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Tag

The BOOL type tag which is set to 1 (TRUE).

Example

The following program code sets the value of the Boolean tag bStatus to 1 by means of the SetBit function and outputs the result along with the original bSaved value.

```
'Programming language: VB
Dim myTag
Dim myOutputField
Dim bValue, bSaved, strResult

Set myTag = SmartTags("bStatus")
Set myOutputField=HMIRuntime.Screens("MyScreen").ScreenItems("objTextField")

'Set value
bValue=0
myTag.Value=bValue

'Save current value
bSaved=bValue

'Set Bit
SetBit(myTag)
bValue=myTag.Value

'Output result old and new value:
strResult="Old Value: "&bSaved &Chr(13)&"New Value: "&bValue
myOutputField.Text=strResult
```

```
//Programming language: C
{
BOOL bStatus = 0;
BOOL bSaved = bStatus;

//Set bit
  SetBit (bStatus);

//print current and saved value
printf ("Current value: %d\r\n, Saved value: %d\r\n",bStatus, bSaved);
...
}
```

SetBitInTag (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Sets a bit in the given tag to 1 (TRUE).

After changing the given bit, the system function transfers the entire tag back to the PLC. It is not checked whether other bits in the tags have changed in the meantime. Operator and PLC have read-only access to the indicated tag until it is transferred back to the PLC.

Note

If the PLC supports BOOL tags, do not use this system function. Use the "SetBit" system function instead.

Use in the function list

SetBitInTag (Tag, Bit)

Use in user-defined functions

SetBitInTag(Tag, Bit)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Tag

The tag in which a bit is set to 1 (TRUE).

Bit

The number of the bit that is set to 1 (TRUE).

When this system function is used in a user-defined function, the bits in the specified tag will be counted from right to left independent of the PLC used. The counting begins with 0.

Note

The guaranteed update of the tags used with actual process values is absolutely conditional in terms of reliable functionality. You should therefore configure the tag in an I/O field or assign the system function to a screen object, such as a button.

If you have configured a short event such as the activation of an alarm for the system function you can only access the actual process values by setting the tag for continuous reading.

Example

The following program code sets a bit to 1 at the specified bitposition in the bStatusWord tag and outputs the result along with the original bSaved value.

```
'Programming language: VB
Dim myTag
Dim myOutputField
Dim bValue, bSaved, bitposition, strResult

Set myTag = SmartTags("bStatusWord")
Set myOutputField=HMIRuntime.Screens("MyScreen").ScreenItems("objTextField")

'Save current value
bValue=myTag.Value
bSaved=bValue

'Set Bit in tag
bitposition=2
SetBitInTag(myTag, bitposition)
bValue=myTag.Value

'Output result old and new value:
strResult="Old Value: "&bSaved &Chr(13)&"New Value: "&bValue
myOutputField.Text=strResult
```

```
//Programming language: C
{
BYTE bStatusWord;
BYTE bSaved = bStatusWord;
BYTE bitposition = 2;

//Reset bit in bitposition
SetBitInTag (bStatusWord, bitposition);

//print current and saved value
printf ("Current value: %d\r\n, Saved value: %d\r\n",bStatusWord, bSaved);
  ...
}
```

See also

SetBit (Page 5706)

SetBitWhileKeyPressed (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Sets the bit of the given tag to 1 (TRUE) as long as the user keeps the configured key pressed.

After changing the given bit, the system function transfers the entire tag back to the PLC. It is not checked whether other bits in the tags have changed in the meantime. Operator and PLC have read-only access to the indicated tag until it is transferred back to the PLC. You should only access tags of the BOOL type with this system function to avoid problems with overlapping access to the same tag.

Note

All functions on the event "Release" are performed immediately by means of a screen change configured for a key, even if the key is kept pressed.

If the "SetBitWhileKeyPressed" system function is configured for a function key, the bit will be reset immediately following a screen change. This action is necessary since the key assignments change after the screen change.

If the PLC supports BOOL tags, do not use this system function. Use the "SetBit" system function instead.

Use in the function list

SetBitWhileKeyPressed (Tag, Bit)

Use in user-defined functions

-

Parameters

Tag

The tag in which a bit is temporarily set to 1 (TRUE). Use only tags of the type BOOL, as far as allowed by the PLC.

Bit

The number of the bit that is temporarily set to 1 (TRUE).

Note

The guaranteed update of the tags used with actual process values is absolutely conditional in terms of reliable functionality. You should therefore configure the tag in an IO field, or assign the function to a screen element such as a button.

If you configured a short event such as the activation of an alarm for the function you can only access the actual process values by setting the tag for continuous reading.

See also

SetBit (Page 5706)

Settings

GetPLCMode (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Evaluates the current state of the connected PLC.

The system function "GetPLCMode" can only be configured for the following PLCs:

- SIMATIC S7-1200
- SIMATIC S7-1500

Use in the function list

GetPLCMode (Connection, Mode)

Use in user-defined functions

-

Parameters

Connection

Connection of PLC and HMI device.

Mode

Evaluates the state of the connected PLC.
8: RUN = The PLC program is being executed.

4: STOP = The PLC program has been interrupted.
Select a suitable user variable for the evaluation.

SetDeviceMode (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Toggles the operating mode on the HMI device. The following types of operation are possible: "Online", "Offline" and "Loading".

Use in the function list

SetDeviceMode (Operating mode)

Use in user-defined functions

SetDeviceMode (Operating_mode)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Operating mode

Determines the operating mode of the HMI device:

0 (hmiOnline) = online: The connection to the PLC is established. The configured connection status is always set in this process. The states that were last used in Runtime are not considered.

1 (hmiOffline) = offline: The connection to the PLC is disconnected.

2 (hmiTransfer) = Load: A project can be transferred from the configuration computer to the HMI device.

Note

If you use a PC as an HMI device, the runtime software will be exited when you switch operating mode after "Load".

See also

SetConnectionMode (Page 5715)

SetPLCDateTime (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Changes the data and the time of the linked PLC

The system function "SetPLCDateTime" can only be configured for the following PLCs:

- SIMATIC S7-1200
- SIMATIC S7-1500

Use in the function list

SetPLCDateTime (connection, time)

Use in user-defined functions

-

Parameters

Connection

Connection of PLC and HMI device.

Time

Transfers the date and the time of the HMI device to the PLC. The PLC applies the date and the time of the HMI device.

SetPLCMode (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Switches the operating mode of the PLC to one of the following states:

- RUN
- STOP

The system function "SetPLCMode" can only be configured for the following PLCs:

- SIMATIC S7-1200
- SIMATIC S7-1500

Use in the function list

SetPLCMode (connection, mode)

Use in user-defined functions

-

Parameters

Connection

Connection of PLC and HMI device.

Mode

Specifies the operating mode of the PLC:

8: RUN = The controller is set to the RUN state. The control program is executed.

4: STOP = The controller is set to the STOP state. The PLC program is interrupted.

SetLanguage (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Toggles the language on the HMI device. All configured text and system events are displayed on the HMI device in the newly set language.

Use in the function list

SetLanguage (Language)

Use in user-defined functions

SetLanguage (Language)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Language

Determines which language is set on the HMI device. The following specifications are possible:

- -1 (hmiToggle) = Toggle: Changes to the next language. The sequence is determined during configuration in the "Project languages" editor.
- Number you have defined under "Languages and fonts" in the "Runtime Settings" editor. Changes to the language with the given number.

- Language you have defined under "Languages and fonts" in the "Runtime Settings" editor.
- Language abbreviation in accordance with the VBScript5 reference: This changes to the language corresponding to the specified language code, e.g. "de-DE" for German (Germany) or "en-US" for English (United States).
  An overview of the language abbreviations is available in the basic information of VBScript under the topic "Area diagram-ID (LCID) Diagram".

SetConnectionMode (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Connects or disconnects the given connection.

Note

A connection to the PLC cannot be established until the "Online" operating mode has been set on the HMI device. Use the "SetDeviceMode" system function for this purpose.

If the connection is loaded in "Offline" mode, the connection is closed again each time the mode switches to "Offline". To re-establish the connection after switching to "Online" mode, set the connection again to "Online" mode.

Use in the function list

SetConnectionMode (Mode, Connection)

Use in user-defined functions

SetConnectionMode (Mode, Connection)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Mode

Determines whether a connection to the PLC is established or disconnected:

0 (hmiOnline) = online: Connection is established.

1 (hmiOffline) = offline: Connection is disconnected.

Connection

The PLC to which the HMI device is connected. You specify the name of the PLC in the connection editor.

Multiple use of the system function in a user-defined function

If you use the "SetConnectionMode" system function for different connections, it may be possible that not all system functions are executed correctly. Proceed as follows to prevent this situation:

1. Create a "BOOL" type tag with the start value "0".
2. Configure the "SetConnectionMode" system function on the "Value change" event of the HMI tags. If you want to disconnect three connections, for example, you must configure the system function three times.
3. In the user-defined function, apply the "InvertBit" system function to the HMI tag.

Application example

Two typical application examples for this system function are as follows:

- Test
  As long as no PLC is connected to the HMI device, no error messages will be output during the test on the HMI device. If the HMI device is connected to a PLC, the connection to the PLC can be established by pressing a key.
- Commissioning
  Several PLCs are to be configured for a system. At first, all PLCs except one are configured "Offline". After commissioning of the first PLC, the connection to each of the other PLCs is established by pressing a key. In this way, the other PLCs are started up one after another.

See also

SetDeviceMode (Page 5712)

ChangeConnection (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Disconnects the connection to the PLC currently in use and establishes a connection to a PLC with another address. The newly connected PLC must belong to the same device class (S7-300, S7-400, etc).

Note

When changing to another address, ensure that the address is not already being used by another HMI device.

The follows address types are supported:

- IP address
- MPI address

The follows PLC types are supported:

- SIMATIC S7 300/400
- SIMATIC S7 1200 (device version up to V11)
- SIMATIC S7-NC
- SIMOTION

Use in the function list

ChangeConnection (Connection, Address, Slot, Rack)

Use in user-defined functions

ChangeConnection (Connection, Address, Slot, Rack)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Connection

Name of the connection that is disconnected. The name is set during configuration, for example, in the "Connections" editor.

Address

MPI/PROFIBUS or IP address of the PLC to which the connection will be established.

Note

Set the address by means of a tag. The objects list displays the tags of all data types. Select only tags of the following data types:

- Ethernet connection: "String" data type
- MPI connection: "Int" data type

Slot

Slot of the PLC to which the connection will be established.

Rack

Rack of the PLC to which the connection will be established.

Application example

You want to operate one HMI device on several machines. Configure a controller in the project. When changing the PLC, the connection to the PLC in use is disconnected. Then the connection to the new PLC with other address parameters is reestablished. All tags in this connection are now updated from the new controller.

The PLC which you have indicated when creating the project will be used as default.

1. Enter the name and address of the PLC in the "Connections" editor.
2. Configure a button in the process screen.
3. Configure the "ChangeConnection" system function on the "Press" event.
4. Enter the name of the connection and address of the PLC as parameters.

Alarms

EditAlarm (Panels, Comfort Panels, RT Advanced)

Description

Triggers the "Edit" event for the selected alarm or, with multiple selection, the last selected alarm.

If the alarms to be edited have not yet been acknowledged, the acknowledgment takes place automatically when this system function is called up.

This system function can only be used for function keys.

Use in the function list

EditAlarm

Use in user-defined functions

EditAlarm

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

ClearAlarmBuffer (Panels, Comfort Panels, RT Advanced)

Description

Deletes alarms from the alarm buffer on the HMI device.

Note

Alarms which have not yet been acknowledged are also deleted.

Use in the function list

ClearAlarmBuffer (Alarm class number)

Use in user-defined functions

ClearAlarmBuffer (Alarm_class_number)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Alarm class number

Determines which alarms are to be deleted from the alarm buffer:

0 (hmiAll) = All alarms

1 (hmiAlarms) = Alarms of alarm class "Error"

2 (hmiEvents) = Alarms of alarm class "Warning"

3 (hmiSystem) = Alarms of alarm class "System event"

4 (hmiS7Diagnosis) = Alarms of alarm class "S7 diagnostic alarm"

Note

Device dependency

Alarms of alarm class "Diagnosis Events" are not available on Basic Panels.

ClearAlarmBufferProTool (Panels, Comfort Panels, RT Advanced)

Description

The system function exists to ensure compatibility.

It has the same functionality as the "ClearAlarmBuffer" system function, but uses the old ProTool numbering.

Use in the function list

ClearAlarmBufferProTool (Alarm class number)

Use in user-defined functions

ClearAlarmBufferProtoolLegacy (Alarm_class_number)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Alarm class number

Alarm class number whose messages are to be deleted:

-1 (hmiAllProtoolLegacy) = All alarms/events

0 (hmiAlarmsProtoolLegacy) = Alarms of alarm class "Errors"

1 (hmiEventsProtoolLegacy) = Alarms of alarm class "Warnings"

2 (hmiSystemProtoolLegacy) = Alarms of alarm class "System"

3 (hmiS7DiagnosisProtoolLegacy) = Alarms of alarm class "Diagnosis Events"

Note

Device dependency

Alarms of alarm class "Diagnosis Events" are not available on Basic Panels.

See also

ClearAlarmBuffer (Page 5718)

AlarmViewLoopInAlarm (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Triggers the "Loop-in-Alarm" event for all alarms selected in the given alarm view.

This system function is used when the integrated button of the ActiveX control should not be used.

You can configure a system function for the "Loop-in-Alarm" event in turn. For example, it is possible to change to the process screen in which the alarm appeared.

Note

If the alarms to be edited have not yet been acknowledged, the acknowledgment takes place automatically when this system function is called up.

Use in the function list

AlarmViewLoopInAlarm (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the alarm view in which the event is triggered.

Note

The HMI devices listed below do not support this system function for the "screen" object: OP 73, OP 77A, TP 177A.

AlarmViewAcknowledgeAlarm (Panels, Comfort Panels, RT Advanced)

Description

Acknowledges the alarms selected in the given alarm view.

This system function is used when the integrated button of the ActiveX control should not be used.

Use in the function list

AlarmViewAcknowledgeAlarm (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the alarm view in which the event is triggered.

Note

The HMI devices listed below do not support this system function for the "screen" object: OP 73, OP 77A, TP 177A.

AlarmViewShowOperatorNotes (Panels, Comfort Panels, RT Advanced)

Description

Displays the configured tooltip of the alarm selected in the given alarm view.

Use in the function list

AlarmViewShowOperatorNotes (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the alarm view in which the event is triggered.

Note

The HMI devices listed below do not support this system function for the "screen" object: OP 73, OP 77A, TP 177A.

AcknowledgeAlarm (Panels, Comfort Panels, RT Advanced)

Description

Acknowledges all selected alarms.

This system function is used when the HMI device does not have an ACK key or when the integrated key of the alarm view should not be used.

This system function can only be used for function keys.

Use in the function list

AcknowledgeAlarm

Use in user-defined functions

AcknowledgeAlarm

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

ShowAlarmWindow (Panels, Comfort Panels, RT Advanced)

Description

Hides or shows the alarm window on the HMI device.

Use in the function list

ShowAlarmWindow (Object name, Display mode)

Use in user-defined functions

ShowAlarmWindow (Object_name, Display_mode)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Object name

Name of the alarm view which is hidden or shown.

Display mode

Determines whether the alarm window is hidden or shown:

0 (hmiOff) = Off: Alarm view is hidden

1 (hmiOn) = On: Alarm view is shown

-1 (hmiToggle) = Toggle: Toggles between the two modes

Recipes

ExportDataRecords (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Exports one or all data records of a recipe to a CSV file.

One file is created per recipe.

Use in the function list

ExportDataRecords (Recipe number/name, Data record number/name, File name, Overwrite, Output status message, Processing status)

Use in user-defined functions

ExportDataRecords (Recipe_number/name,Data_record number/name, File_name, Overwrite, Output_status_message, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Recipe number/name

Number or name of the recipe from which the data records are exported. Specify "0" if recipe data records are to be exported from all available recipes.

Data record number/name

Number or name of the recipe data record to be exported. Specify "0" if all recipe data records are to be exported.

File name

Name of the CSV file to which the recipe data records are exported. Enter the storage location and the file extension (*.csv), for example "C:\TEMP\Orange.csv", for Basic Panels "\USB_X60.1\Orange.csv".

Note

Storage of the CSV file

Does not apply for Basic Panels:

- If a storage card is used as storage location, specify the storage location as follows: "\StorageCard\<FileName>".
- If you only enter one file name and no path, the file will be saved in a system directory, for example, "C:\Documents and Settings\[User]".
- When only one path for the export is specified, the file name is automatically created from the respective recipe name. It is a requirement that the folder "D:\Temp\", for example, has been created. If the folder "D:\Temp" has not been created, the folder name will be used as the prefix of the file name, Temp_Recipe names.csv.

Overwrite

Determines whether an existing CSV file with the same name is overwritten:

0 (hmiOverwriteForbidden) = No: CSV file will not be overwritten. The export process will not be carried out.

1 (hmiOverwriteAlways) = Yes: CSV file is overwritten without a prompt for confirmation.

2 (hmiOverwriteWithPrompting) = With confirmation: CSV file is not overwritten until confirmed.

Output status message

Determines whether a status message is output after the export:

0 (hmiOff) = Off: Status message is not output.

1 (hmiOn) = On: Status message is output.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

Export format of the recipe data records

If ".csv" is chosen as a file extension for the export file, then only valid characters from the ANSI character set will be supported. This also applies to separators in decimal numbers and list elements. The separators used are defined in the country/regional settings of the operating system of the exporting computer.

Does not apply for Basic Panels:

You may also set "Unicode text" (".txt") format for the export file. This file format supports the WinCC and WinCC Runtime character set. Again, the separators used are specified in the country/regional settings of the operating system of the exporting computer. This file format always uses tab separators in list elements.

The corresponding file import function also supports the ".csv" and ".txt" (Unicode) file formats.

Application example

You want to export all data records using a key.

Notes on configuring

Configure the "ExportDataRecords" system function on the "Press" event for the desired key. Transfer the following parameters:

- Recipe number/name = 1
- Data record number/name = 0
- File name = c:\temp\orange.csv, for Basic Panels "\USB_X60.1\orange.csv"
- Overwrite = 1
- Output status message = 1

Tags can also be specified in place of the constants. Depending on the configuration, the operator can enter the desired values in the I/O field or read from the PLC. In this way, the operator can determine which recipe data records are exported.

Procedure on HMI device

As soon as the key is activated, the system function is triggered. The constants or tags are evaluated. All data records of Recipe 1 are exported to the orange.csv file. If the file already exists, it will be overwritten.

After exporting the data records, a system alarm is output.

ImportDataRecords (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Imports one or all data records of a recipe from a CSV file.

When a path is specified, all records of the given directory are imported.

Use in the function list

ImportDataRecords (File name, Data record number/name, Overwrite, Output status message, Processing status)

Use in user-defined functions

ImportDataRecords (File_name, Data_record number/name, Overwrite, Output_status_message, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

File name

Name of the CSV file from which the recipe data records are imported. Enter the storage location and the file extension (*.csv), for example, "C:\TEMP\Orange.csv", for Basic Panels "\USB_X60.1\Orange.csv".

Note

Does not apply for Basic Panels: If a storage card is used as file location, specify the file location as follows: "\StorageCard\<FileName".

Data record number/name

Number or name of the recipe data record to be imported. Specify "0" if all recipe data records are to be imported.

Overwrite

Determines whether existing recipe data records are to be overwritten:

0 (hmiOverwriteForbidden) = No: Recipe data records are not overwritten. The import process will not be carried out.

1 (hmiOverwriteAlways) = Yes: Recipe data records are overwritten without prompting.

2 (hmiOverwriteWithPrompting) = With confirmation: Recipe data records are not overwritten until confirmed.

Output status message

Determines whether a status message is output after the import:

0 (hmiOff) = Off: Status message is not output.

1 (hmiOn) = On: Status message is output.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

Configurable objects

Object	Event
Tag	Value change Upper limit exceeded Lower limit violated
Function key (global)	Release Press
Function key (local)	Release Press
Screen	Loaded Cleared
Screen object	Press Release Click Change (or toggle for switch) Switch on Switch off Enable Disable
Scheduler	Time expired

LoadDataRecord (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Loads the given recipe data record from the memory medium of the HMI device in the recipe tags. This system function is used, for example, to display a recipe data record in the recipe screen.

Activate the "Synchronize recipe view and recipe tags" option in the synchronization settings of the recipe. If this option is deactivated, the system function has no effect.

Use in the function list

LoadDataRecord (Recipe number/name, Data record number/name, Processing status)

Use in user-defined functions

LoadDataRecord (Recipe_number/name, Data_record_number/name, Confirmation, Output_status_message, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Recipe number/name

Number or name of the recipe from which a recipe data record is loaded.

Data record number/name

Number or name of the recipe data record to be loaded.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

GetDataRecordFromPLC (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Transfers the selected recipe data record from the PLC to the storage medium of the HMI device.

Use in the function list

GetDataRecordFromPLC (Recipe number/name, Data record number/name, Overwrite, Output status message, Processing status)

Use in user-defined functions

GetDataRecordFromPLC (Recipe_number/name, Data_record_number/name, Overwrite, Output_status_message, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Recipe number/name

Number or name of the recipe from which recipe data records are transferred.

Data record number/name

Number or name of the recipe data record which is transferred from the PLC to the data medium of the HMI device.

Overwrite

Determines whether an existing recipe data record with the same name is overwritten:

0 (hmiOverwriteForbidden) = No: Recipe data record is not overwritten. The transfer process will not be carried out.

1 (hmiOverwriteAlways) = Yes: Recipe data record is overwritten without prompting.

2 (hmiOverwriteWithPrompting) = With confirmation: Recipe data record is not overwritten until confirmed.

Output status message

Determines whether a status message is output after the transfer:

0 (hmiOff) = Off: Status message is not output.

1 (hmiOn) = On: Status message is output.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

Application example

You want to transfer a data record from the PLC to the data medium of the HMI device using a key.

Notes on configuring

Configure the "GetDataRecordFromPLC" system function on the "Press" event for the desired key. Transfer the following parameters:

Recipe number/name = 1

Data record number/name = 1

Overwrite = 1

Output status message = 1

Tags can also be specified in place of the constants. Depending on the configuration, the operator can enter the desired values in the I/O field or read from the PLC. In this way, the operator can determine which recipe data record is transferred from the PLC.

Procedure on HMI device

As soon as the key is activated, the system function is triggered. The constants or tags are evaluated and the first data record of Recipe 1 is transferred from the PLC to the data medium of the HMI device. If the recipe data record already exists, it will be overwritten.

A system message is output after the transfer.

GetDataRecordTagsFromPLC (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Transfers the values of the recipe data record loaded in the PLC to the corresponding recipe tags.

This system function is used, for example, during teach-in mode on a machine.

Use in the function list

GetDataRecordTagsFromPLC (Recipe number/name, Processing status)

Use in user-defined functions

GetDataRecordTagsFromPLC (Recipe_number/name, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Recipe number/name

Number or name of the recipe data record whose values are written from the PLC to the tags.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

ClearDataRecord (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Deletes a recipe data record.

Several data records can be deleted from one or more recipes.

Use in the function list

ClearDataRecord (Recipe number/name, Data record number/name, Confirmation, Output status message, Processing status)

Use in user-defined functions

ClearDataRecord

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Recipe number/name

Number or name of the recipe from which recipe data records are deleted. "0" is specified if you want to delete recipe data records from all available recipes.

Data record number/name

Number or name of the recipe data record to be deleted. "0" is specified if you want to delete all recipe data records.

Confirmation

Determines whether the operator should confirm the deletion:

0 (hmiOff) = Off: Deletion is begun without confirmation.

1 (hmiOn) = On: The starting of the deletion process must be confirmed.

Output status message

Determines whether a status message is output after the deletion:

0 (hmiOff) = Off: Status message is not output.

1 (hmiOn) = On: Status message is output.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

ClearDataRecordMemory (Panels, Comfort Panels, RT Advanced)

Description

Deletes all recipes data records from the specified storage medium.

Use in the function list

ClearDataRecordMemory (Storage location, After confirmation, Output status message, Processing status)

Use in user-defined functions

ClearDataRecordMemory (Storage_location, Confirmation, Output_status_message, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Storage location

Determines the storage location:

0 (hmiFlashMemory) = Flash memory: Internal flash memory of the HMI device

1 (hmiStorageCard) = Storage card

2 (hmiStorageCard2) = Storage card

3 (hmiStorageCard3) = Storage card MultiMediaCard

4 (hmiStorageCard4) = Storage card

With confirmation

Determines whether the operator should confirm the deletion:

0 (hmiOff) = Off: Deletion is begun without confirmation.

1 (hmiOn) = On: The starting of the deletion process must be confirmed.

Output status message

Determines whether a status message is output after the deletion:

0 (hmiOff) = Off: Status message is not output.

1 (hmiOn) = On: Status message is output.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

See also

Device dependency of system functions for 2nd generation Basic Panels (Page 5665)

RecipeViewNewDataRecord (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Creates a new data record in the given recipe view.

Use in the function list

RecipeViewNewDataRecord (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the new recipe data record is created.

RecipeViewGetDataRecordFromPLC (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Transfers the data record that is currently loaded in the PLC to the HMI device and displays it in the recipe view.

Use in the function list

RecipeViewGetDataRecordFromPLC (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the recipe data record from the PLC is displayed.

RecipeViewSaveDataRecord (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Saves the recipe data record which is currently displayed in the recipe view.

Use in the function list

RecipeViewSaveDataRecord (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the recipe data record is saved.

RecipeViewClearDataRecord (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Deletes the data record which is currently displayed in the recipe view.

Use in the function list

RecipeViewClearDataRecord (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the displayed recipe data record is deleted.

RecipeViewMenu (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Opens the menu of the specified simple recipe view.

Only use this system function at a simple recipe view.

Use in the function list

RecipeViewMenu (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the menu is to be opened.

RecipeViewOpen (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Displays the data record values in the given recipe view or changes to the next selection field. The system function has no effect if the selection field for the recipe data record values is displayed on the HMI device.

Operation sequence of the selection lists in runtime:

- Recipe name
- Data record name
- RecipeDataRecordValues

This system function is used when a simple recipe view has been configured. In the simple recipe view, only one selection list is displayed at a time on the HMI device. Use the "RecipeViewBack" system function to display the previous selection list.

Use in the function list

RecipeViewOpen (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the command is triggered.

See also

RecipeViewBack (Page 5738)

RecipeViewSetDataRecordToPLC (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Transfers the recipe data record which is currently displayed in the recipe view to the PLC.

Use in the function list

RecipeViewSetDataRecordToPLC (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view from which the recipe data record is transferred to the connected PLC.

RecipeViewSaveAsDataRecord (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Saves the data record currently being displayed in the recipe view under a new name.

Use in the function list

RecipeViewSaveAsDataRecord (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Saves the data record currently being displayed in the recipe view under a new name and/or new number.

RecipeViewSynchronizeDataRecordWithTags (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Synchronizes the values of the data record which is currently displayed in the recipe view with their recipe tags. Only use this system function at an advanced recipe view.

During synchronization, all values of the data record are written to their recipe tags.

Use in the function list

RecipeViewSynchronizeDataRecordWithTags (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the values are synchronized with their tags.

Note

The OP 77A and TP 177A HMI devices do not support this function for the screen object.

RecipeViewRenameDataRecord (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Renames the selected data record in the given recipe view.

Use in the function list

RecipeViewRenameDataRecord (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the recipe data record is renamed.

RecipeViewShowOperatorNotes (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Displays the configured tooltip of the specified recipe view.

Use in the function list

RecipeViewShowOperatorNotes (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view whose configured tooltip is displayed.

RecipeViewBack (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Returns to the previous selection list in the given recipe view.

The system function has no effect if the selection field for the recipe is displayed on the HMI device. Operation sequence of the selection lists in runtime:

- Recipe name
- Data record name
- RecipeDataRecordValues

This system function is used when a simple recipe view has been configured. In the simple recipe view, only one selection list is displayed at a time on the HMI device. Use the "RecipeViewOpen" system function to display the recipe data record values or the next selection field.

Use in the function list

RecipeViewBack (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the recipe view in which the command is triggered.

SetDataRecordToPLC (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Transfers the given recipe data record directly from the data medium of the HMI device to the PLC with which the HMI device is connected.

Note

The values of the recipe data record don't need to be displayed on the HMI device.

Use in the function list

SetDataRecordToPLC (Recipe number/name, Data record number/name, Output status message, Processing status)

Use in user-defined functions

SetDataRecordToPLC (Recipe_number/name, Data_record_number/name, Output_status_message, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Recipe number/name

Number or name of the recipe from which recipe data record is transferred to the PLC.

Data record number/name

Number or name of the recipe data record to be transferred to the PLC.

Output status message

Determines whether a status message is output after the transfer:

0 (hmiOff) = Off: Status message is not output.

1 (hmiOn) = On: Status message is output.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

SetDataRecordTagsToPLC (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Transfers the values of the recipe tags to the PLC. The recipe tags contain the values of the data record which is displayed on the HMI device.

Use in the function list

SetDataRecordTagsToPLC (Recipe number/name, Processing status)

Use in user-defined functions

SetDataRecordTagsToPLC (Recipe_ number/name, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Recipe number/name

Number or name of the recipe from which recipe data record is transferred to the PLC.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

SetRecipeTags (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Changes the status of the recipe tags from "Online" to "Offline" and vice versa.

This system function is used, for example, when recipe data record values are fine tuned when starting up a machine.

Use in the function list

SetRecipeTags (Recipe number/name, Status, Output status message, Processing status)

Use in user-defined functions

SetRecipeTags (Recipe_number/name, Status, Output_status_message, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Recipe number/name

Number or name of the recipe in which the recipe data record is saved.

Status

Determines the status of the recipe tags:

0 (hmiOnline) = Online: Value changes of the recipe tags are transferred immediately to the PLC connected to the HMI device.

1 (hmiOffline) = Offline: Value changes to the recipe tags are only transferred to the PLC connected to the HMI device when, for example, the "SetDataRecordTagsToPLC" system function is executed.

Output status message

Determines whether a status message is output after saving:

0 (hmiOff) = Off: Status message is not output.

1 (hmiOn) = On: Status message is output.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

SaveDataRecord (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Saves the current values of the recipe tags as data record to the memory medium of the HMI device.

This system function is used, for example, to save a recipe data record in the recipe screen.

Use in the function list

SaveDataRecord (Recipe number/name, Data record number/name, Overwrite, Output status message, Processing status)

Use in user-defined functions

SaveDataRecord (Recipe_number/name, Data_record_number/name, Overwrite, Output_status_message, Processing_status)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameters

Recipe number/name

Number or name of the recipe to which a recipe data record is saved.

Data record number/name

Number or name of the recipe data record to be saved. A new data record will be created if no record of this name or number was found in the recipe, independent of the value at the "Overwrite" parameter.

Overwrite

Specifies whether an existing data record is overwritten:

0 (hmiOverwriteForbidden) = No: The recipe data record is not overwritten, the data record is not saved.

1 (hmiOverwriteAlways) = Yes: The recipe data record is overwritten without a prompt for confirmation.

2 (hmiOverwriteWithConfirmation) = With confirmation: The recipe data record is overwritten only with confirmation by the user.

Output status message

Determines whether a status message is output after saving:

0 (hmiOff) = Off: Status message is not output.

1 (hmiOn) = On: Status message is output.

Processing status

Returns the processing status of the system function. You can use the return value, for example, to delay execution of other system functions until this system function has been successfully completed:

2 = System function is being performed.

4 = System function was successfully completed.

12 = System function was not performed because an error has occurred.

System

ActivateCleanScreen (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Activates the clean screen on the HMI device. The display of the HMI device is disabled for the given time period.

When the display of the HMI device is deactivated, it can be cleaned without triggering touch functions by mistake.

Use in the function list

ActivateCleanScreen (Time period)

Use in user-defined functions

--

Parameters

Time period

Time period for which the display is disabled. The time remaining is displayed as a progress bar.

Value range in seconds from 10 through 300.

Note

The system function ActivateCleanScreen cannot be simulated.

See also

Logoff (Page 5688)

Keyboard

Keyboard operation for screen objects

ScreenObjectCursorUp (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Results in a line-by-line, upward cursor movement in the table of the specified screen object.

The system function can be used for the following screen objects:

- User view
- Alarm view
- Recipe view
- NC subprogram display
- PLC code view

Use in the function list

ScreenObjectCursorUp (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the screen object in which the key function is triggered.

ScreenObjectCursorDown (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Results in a line-by-line, downward cursor movement in the table of the specified screen object.

The system function can be used for the following screen objects:

- User view
- Alarm view
- Recipe view
- NC subprogram display
- PLC code view

Use in the function list

ScreenObjectCursorDown (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the screen object in which the key function is triggered.

ScreenObjectCursorLeft

Description

Results in a line-by-line, cursor movement to the left in the specified screen object.

Use in the function list

ScreenObjectCursorLeft (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the screen object in which the command is executed.

ScreenObjectCursorRight

Description

Results in a line-by-line, cursor movement to the right in the specified screen object.

Use in the function list

ScreenObjectCursorLeft (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the screen object in which the command is executed.

ScreenObjectPageUp (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Results in a page-by-page, upward cursor movement in the table of the specified screen object.

The system function can be used for the following screen objects:

- User view
- Alarm view
- Recipe view
- NC subprogram display

Use in the function list

ScreenObjectPageUp (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the screen object in which the key function is triggered.

ScreenObjectPageDown (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Results in a page-by-page, downward cursor movement in the table of the specified screen object.

The system function can be used for the following screen objects:

- User view
- Alarm view
- Recipe view
- NC subprogram display

Use in the function list

ScreenObjectPageDown (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the screen object in which the key function is triggered.

PageUp (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Executes the key function <PageUp> on the HMI device.

This system function can only be used for function keys and tasks with a time trigger.

Use in the function list

PageUp

Use in user-defined functions

PageUp

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

-

PageDown (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Executes the key function <Pagedown> on the HMI device.

This system function can only be used for function keys.

Use in the function list

PageDown

Use in user-defined functions

PageDown

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

-

GoToHome (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Executes the key function <Home> on the HMI device.

This system function is used when the HMI device does not have this functionality by default.

The system function can only be used for the following function keys.

Use in the function list

GoToHome

Use in user-defined functions

GoToHome

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

GoToEnd (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Executes the key function <End> on the HMI device.

This system function is used when the HMI device does not have this functionality by default.

The system function can only be used for the following function keys.

Use in the function list

GoToEnd

Use in user-defined functions

GoToEnd

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

Functions of the display

AdjustContrast (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Changes the contrast of the display one level on the HMI device.

Use in the function list

AdjustContrast (Adjust)

Use in user-defined functions

-

Parameters

Adjust

Specifies how the contrast is changed:

0 (hmiDecrease) = Decrease: Decreases the contrast one level.

1 (hmiIncrease) = Increase: Increases the contrast one level.

Application example

Objective

One button each for increasing and decreasing the screen contrast is desired.

Notes on configuring

Configure two buttons and configure the "AdjustContrast" system function on the "Press" event. The parameters "Increase" and "Decrease" are allocated.

Procedure on HMI device

When one of the two buttons is pressed in runtime, the contrast is increased or decreased one level.

See also

Logoff (Page 5688)

CalibrateTouchScreen (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Calls a program for calibrating the touch screen.

During the calibration process, there is a prompt to touch five positions on the screen display. Touch the screen display within 30 seconds, to confirm the calibration process. If the calibration is not completed within this time span, the calibration settings are discarded. The user prompt is in English.

Use this system function the first time you start the HMI device.

Use in the function list

CalibrateTouchScreen

Use in user-defined functions

CalibrateTouchScreen

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

--

Note

The CalibrateTouchScreen system function cannot be simulated.

SetBrightness (Panels, Comfort Panels, RT Advanced)

Description

Determines the brightness of the display.

Note

The configuration that is set in the Control Panel / Start Center will be reestablished when you restart the HMI device.

For Basic Panels 2nd Generation, Mobile Panels and Comfort Panels:

The value for the system function "SetBrightness" can be set between 0% and 100%. The set value is transferred to the HMI device. The brightness settings on the HMI device can be viewed and edited in "Start Center > Settings > Display". The HMI devices support a brightness setting between 10% and 100%.

If the system function "SetBrightness" is assigned a value of 0%, the display of the HMI device is switched off by default in Runtime. If the operator touches the display, the display switches to the previous brightness setting.

If the system function "SetBrightness" is assigned a value between 1% and 10% and the operator opens the display settings in the Start Center, brightness is reset to 10%.

Use in the function list

SetBrightness (value)

Use in user-defined functions

SetBrightness (Value)

Can be used if the configured device supports user-defined functions. For additional information, refer to "Device dependency".

Parameter

Value

New value for the brightness.

SetBacklightColor (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Defines the background lighting of the button.

Note

The configuration that was set at Switch off is reestablished when restarting the HMI device.

Use in the function list

SetBacklightColor (Value)

Use in user-defined functions

-

Parameters

Value

Defines the background lighting of the button:

0 (hmiWhite) = White: No color

1 (hmiGreen) = Green: Green color

2 (hmiYellow) = Yellow: Yellow color

3 (hmiRed) = Red: Red color

Functions of the HTML browser

HTMLBrowserStop (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Performs the function "Stop" the HTML browser.

Use in the function list

HTMLBrowserStop (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is performed.

HTMLBrowserScrollDown (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Scrolls down in the HTML browser.

Use in the function list

HTMLBrowserScrollDown (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserRefresh (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Performs the function "Refresh" of the HTML browser.

Use in the function list

HTMLBrowserRefresh (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is performed.

HTMLBrowserScrollUp (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Scrolls up in the HTML browser.

Use in the function list

HTMLBrowserScrollUp (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserZoomIn (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Zooms in one zoom level in the HTML browser display.

Use in the function list

HTMLBrowserZooomIn (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserZoomOut (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Zooms out one zoom level in the HTML browser display.

Use in the function list

HTMLBrowserZoomOut (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserScrollLeft (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Scrolls left in the HTML browser.

Use in the function list

HTMLBrowserScrollLeft (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserScrollRight (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Scrolls right in the HTML browser.

Use in the function list

HTMLBrowserScrollRight (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserPageDown (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Results in a page-by-page, downward cursor movement in the HTML browser.

Use in the function list

HTMLBrowserPageDown (screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserPageUp (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Results in a page-by-page, upward cursor movement in the HTML browser.

Use in the function list

HTMLBrowserPageUp (screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserHome (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Switches to the start page stored for the HTML browser.

Use in the function list

HTMLBrowserHome (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is executed.

HTMLBrowserForward (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Performs the function "Forward" of the HTML browser.

Use in the function list

HTMLBrowserForward (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is performed.

HTMLBrowserBack (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Performs the "Back" function of the HTML browser.

Use in the function list

HTMLBrowserBack (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Object name of the HTML browser in which the command is performed.

Functions of the trend view

TrendViewScrollForward (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Scrolls forward one display width in the Trend view.

Use in the function list

TrendViewScrollForward (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which is scrolled forward.

TrendViewScrollBack (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Scrolls back one display width in the trend view.

Use in the function list

TrendViewScrollBack (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which is scrolled back.

TrendViewExtend (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Reduces the time period which is displayed in the trend view.

Use in the function list

TrendViewExtend (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which the displayed time period is reduced.

TrendViewCompress (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Increases the time period which is displayed in the trend view.

Use in the function list

TrendViewCompress (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which the displayed time period is increased.

TrendViewRulerLeft (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Moves the read-line backwards (to the left) in the trend view.

Note

In order to be able to move the read-line, the read-line must have been switched on. This is done using the "TrendViewSetRulerMode" system function.

Use in the function list

TrendViewRulerLeft (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which the read-line is moved backwards.

See also

TrendViewSetRulerMode (Page 5762)

TrendViewRulerRight (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Moves the read-line forwards (to the right) in the trend view.

Note

In order to be able to move the read-line, the read-line must have been switched on. This is done using the "TrendViewSetRulerMode" system function.

Use in the function list

TrendViewRulerRight (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which the read-line is moved forward.

See also

TrendViewSetRulerMode (Page 5762)

TrendViewSetRulerMode (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Hides or shows the read-line in the trend view. The read-line displays the Y value belonging to the X value.

Note

To ensure that the ruler is displayed, you have to activate the setting "Show ruler" in the trend view properties.

Use in the function list

TrendViewSetRulerMode (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which the read-line is hidden or shown.

See also

TrendViewRulerLeft (Page 5760)

TrendViewRulerRight (Page 5761)

TrendViewStartStop (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Stops the trend recording or continues the trend recording in the trend view.

Use in the function list

TrendViewStartStop (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which the recording of the trend is started or stopped.

TrendViewBackToBeginning (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Scrolls back to the beginning of the display range in the trend view.

Use in the function list

TrendViewBackToBeginning (Screen object)

Use in user-defined functions

-

Parameters

Screen object

Name of the trend view in which you scroll to the beginning of the display range.

12.7.4.2 Events

Overview

Editors

Introduction

The following table shows which events occur in which editor.

Technical data subject to change.

Icon	Editor
	Screens
	HMI alarms
	HMI tags
	Scheduler

1	Cleared (Page 5769)	X	--	--	--
2	Activate (Page 5769)	--	--	--	--
3	Change (Page 5769)	--	--	--	--
4	Loaded (Page 5770)	X	--	--	--
5	Execute (Page 5770)	--	--	--	X
6	Selection changed (Page 5770)	--	--	--	--
7	On exceeding (Page 5770)	--	--	X	--
8	On falling below (Page 5771)	--	--	X	--
9	When dialog is opened (Page 5771)	--	--	--	X
10	When dialog is closed (Page 5771)	--	--	--	X
11	User change (Page 5771)	--	--	--	X
12	Screen change (Page 5772)	--	--	--	X
13	Deactivate (Page 5772)	--	--	--	X
14	Double-click (Page 5772)	--	--	--	--
15	Press (Page 5773)	--	--	--	--
16	On Finish Input (Page 5773)	--	--	--	--
17	Press ESC twice (Page 5773)	--	--	--	--
18	Outgoing (Page 5774)	--	X	--	--
19	Incoming (Page 5774)	--	X	--	--
20	Click (Page 5774)	--	--	--	--
21	Click when flashing (Page 5775)	--	--	--	--
22	Loop-in alarm (Page 5775)	--	X	--	--
23	Release (Page 5775)	--	--	--	--
24	Alarm buffer overflow (Page 5776)	--	--	--	X
25	Acknowledge (Page 5776)	--	X	--	--
26	Reach margin (Page 5776)	--	--	--	--
27	Runtime Stop (Page 5777)	--	--	--	X
28	Press key (Page 5777)	--	--	--	--
29	Release key (Page 5777)	--	--	--	--
30	Overflow (Page 5778)	--	--	--	--
31	Switch OFF (Page 5778)	--	--	--	--
32	Switch ON (Page 5778)	--	--	--	--
33	Low free storage space (Page 5778)	--	--	--	--
34	Free space critically low (Page 5779)	--	--	--	--
35	Value change (Page 5779)	--	--	X	--
36	Time expired (Page 5779)	--	--	--	--

Basic objects

Introduction

The following table shows which events occur on which objects.

Technical data subject to change.

Icon	Object
	Line
	Ellipse
	Circle
	Rectangle
	Text field
	Graphic view

						A	
1	Cleared (Page 5769)	--	--	--	--	--	--
2	Activate (Page 5769)	--	--	--	--	--	--
3	Change (Page 5769)	--	--	--	--	--	--
4	Loaded (Page 5770)	--	--	--	--	--	--
5	Execute (Page 5770)	--	--	--	--	--	--
6	Selection changed (Page 5770)	--	--	--	--	--	--
7	On exceeding (Page 5770)	--	--	--	--	--	--
8	On falling below (Page 5771)	--	--	--	--	--	--
9	When dialog is opened (Page 5771)	--	--	--	--	--	--
10	When dialog is closed (Page 5771)	--	--	--	--	--	--
11	User change (Page 5771)	--	--	--	--	--	--
12	Screen change (Page 5772)	--	--	--	--	--	--
13	Deactivate (Page 5772)	--	--	--	--	--	--
14	Double-click (Page 5772)	--	--	--	--	--	--
15	Press (Page 5773)	--	--	--	--	--	--
16	On Finish Input (Page 5773)	--	--	--	--	--	--
17	Press ESC twice (Page 5773)	--	--	--	--	--	--
18	Outgoing (Page 5774)	--	--	--	--	--	--
19	Incoming (Page 5774)	--	--	--	--	--	--
20	Click (Page 5774)	--	--	--	--	--	--
21	Click when flashing (Page 5775)	--	--	--	--	--	--
22	Loop-in alarm (Page 5775)	--	--	--	--	--	--
23	Release (Page 5775)	--	--	--	--	--	--
24	Alarm buffer overflow (Page 5776)	--	--	--	--	--	--
25	Acknowledge (Page 5776)	--	--	--	--	--	--
26	Reach margin (Page 5776)	--	--	--	--	--	--

27	Runtime Stop (Page 5777)	--	--	--	--	--	--
28	Press key (Page 5777)	--	--	--	--	--	--
29	Release key (Page 5777)	--	--	--	--	--	--
30	Overflow (Page 5778)	--	--	--	--	--	--
31	Switch OFF (Page 5778)	--	--	--	--	--	--
32	Switch ON (Page 5778)	--	--	--	--	--	--
33	Low free storage space (Page 5778)	--	--	--	--	--	--
34	Free space critically low (Page 5779)	--	--	--	--	--	--
35	Value change (Page 5779)	--	--	--	--	--	--
36	Time expired (Page 5779)	--	--	--	--	--	--

Elements

Introduction

The following table shows which events occur on which objects.

Technical data subject to change.

Icon	Object
	IO field
	Button
	Symbolic IO field
	Graphic IO field
	Date/time field
	Bar
	Switch

1	Cleared (Page 5769)	--	--	--	--	--	--	--
2	Activate (Page 5769)	X	X	X	X	--	--	X
3	Change (Page 5769)	--	X	X	--	--	--	X
4	Loaded (Page 5770)	--	--	--	--	--	--	--
5	Execute (Page 5770)	--	--	--	--	--	--	--
6	Selection changed (Page 5770)	--	--	--	--	--	--	--
7	On exceeding (Page 5770)	--	--	--	--	--	--	--
8	On falling below (Page 5771)	--	--	--	--	--	--	--
9	When dialog is opened (Page 5771)	--	--	--	--	--	--	--
10	When dialog is closed (Page 5771)	--	--	--	--	--	--	--
11	User change (Page 5771)	--	--	--	--	--	--	--
12	Screen change (Page 5772)	--	--	--	--	--	--	--
13	Deactivate (Page 5772)	X	X	X	X	--	--	X
14	Double-click (Page 5772)	--	--	--	--	--	--	--

15	Press (Page 5773)	--	X	--	--	--	--	--
16	On Finish Input (Page 5773)	--	--	--	--	--	--	--
17	Press ESC twice (Page 5773)	--	--	--	--	--	--	--
18	Outgoing (Page 5774)	--	--	--	--	--	--	--
19	Incoming (Page 5774)	--	--	--	--	--	--	--
20	Click (Page 5774)	--	X	--	--	--	--	--
21	Click when flashing (Page 5775)	--	--	--	--	--	--	--
22	Loop-in alarm (Page 5775)	--	--	--	--	--	--	--
23	Release (Page 5775)	--	X	--	--	--	--	--
24	Alarm buffer overflow (Page 5776)	--	--	--	--	--	--	--
25	Acknowledge (Page 5776)	--	--	--	--	--	--	--
26	Reach margin (Page 5776)	--	--	--	--	--	--	--
27	Runtime Stop (Page 5777)	--	--	--	--	--	--	--
28	Press key (Page 5777)	--	--	--	--	--	--	--
29	Release key (Page 5777)	--	--	--	--	--	--	--
30	Overflow (Page 5778)	--	--	--	--	--	--	--
31	Switch OFF (Page 5778)	--	--	--	--	--	--	X
32	Switch ON (Page 5778)	--	--	--	--	--	--	X
33	Low free storage space (Page 5778)	--	--	--	--	--	--	--
34	Free space critically low (Page 5779)	--	--	--	--	--	--	--
35	Value change (Page 5779)	--	--	--	--	--	--	--
36	Time expired (Page 5779)	--	--	--	--	--	--	--

Controls

Introduction

The following table shows which events occur on which objects.

Technical data subject to change.

	Object
	Alarm view/alarm window
	Alarm indicator
	Trend view
	User view
	Recipe view
	Auxiliary indicator

1	Cleared (Page 5769)	--	--	--	--	--	--
2	Activate (Page 5769)	X	--	X	X	--	--

							
3	Change (Page 5769)	--	--	--	--	--	--
4	Loaded (Page 5770)	--	--	--	--	--	--
5	Execute (Page 5770)	--	--	--	--	--	--
6	Selection changed (Page 5770)	--	--	--	--	--	--
7	On exceeding (Page 5770)	--	--	--	--	--	--
8	On falling below (Page 5771)	--	--	--	--	--	--
9	When dialog is opened (Page 5771)	--	--	--	--	--	--
10	When dialog is closed (Page 5771)	--	--	--	--	--	--
11	User change (Page 5771)	--	--	--	--	--	--
12	Screen change (Page 5772)	--	--	--	--	--	--
13	Deactivate (Page 5772)	X	--	X	X	--	--
14	Double-click (Page 5772)	--	--	--	--	--	--
15	Press (Page 5773)	--	--	--	--	--	--
16	On Finish Input (Page 5773)	--	--	--	--	--	--
17	Press ESC twice (Page 5773)	--	--	--	--	--	--
18	Outgoing (Page 5774)	--	--	--	--	--	--
19	Incoming (Page 5774)						
20	Click (Page 5774)	--	X	--	--	--	--
21	Click when flashing (Page 5775)	--	X	--	--	--	--
22	Loop-in alarm (Page 5775)	--	--	--	--	--	--
23	Release (Page 5775)	--	--	--	--	--	--
24	Alarm buffer overflow (Page 5776)	--	--	--	--	--	--
25	Acknowledge (Page 5776)	--	--	--	--	--	--
26	Reach margin (Page 5776)	--	--	--	--	--	--
27	Runtime Stop (Page 5777)	--	--	--	--	--	--
28	Press key (Page 5777)	--	--	--	--	--	--
29	Release key (Page 5777)	--	--	--	--	--	--
30	Overflow (Page 5778)	--	--	--	--	--	--
31	Switch OFF (Page 5778)	--	--	--	--	--	--
32	Switch ON (Page 5778)	--	--	--	--	--	--
33	Low free storage space (Page 5778)	--	--	--	--	--	--
34	Free space critically low (Page 5779)	--	--	--	--	--	--
35	Value change (Page 5779)	--	--	--	--	--	--
36	Time expired (Page 5779)	--	--	--	--	--	--

Events

Cleared (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the active screen on the HMI device is cleared.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Activate (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the user selects a display or operating object using the configured tab sequence.

Note

Note that the availability of the event depends on the HMI device and object type.

Note

If the user e.g. clicks a button with the mouse, the "Click" event is triggered. Users wishing to trigger the "Enable" event must select the button using the tab key.

The "Enable" event is only used to detect whether an object was selected. The event does not trigger a password prompt.

For this reason, do not use the "Enable" event if you want to configure access protection on the function call of the object.

Change (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs if the status of a display and operator control object changes.

The status of an object changes if, for example, the user presses the key.

Note

Note that the availability of the event depends on the HMI device and object type.

Loaded (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when all configured display and operating objects are loaded in the active screen after a screen change.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Note

Enable a screen change to ensure that the connection with the control is established after switch-on.

Execute (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the scheduled task has been executed.

Selection changed (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the user changes the selection.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

On exceeding (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the high limit of a tag is exceeded.

Note

Note that the availability of the event depends on the HMI device and object type.

On falling below (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the low limit of a tag is undershot.

Note

Note that the availability of the event depends on the HMI device and object type.

When dialog is opened (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

The event is triggered when a modal dialog opens.

Note

Please note that the availability of the event depends on the HMI device and object type.

When dialog is closed (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

The event is triggered when a modal dialog closes.

Note

Please note that the availability of the event depends on the HMI device and object type.

User change (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when a user logs off at an HMI device or another user logs on at the HMI device.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Screen change (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when all configured display and operating objects are loaded in the screen after a screen change.

Use the "Loaded" event if you want to perform other system functions during a screen change to a certain screen.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Deactivate (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the user takes the focus from a display and operating object.

A screen object can be disabled using the configured tab order or by performing another action with the mouse.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Note

System functions or user-defined functions on the "Deactivate" event of a screen are not executed when a screen is being closed.

The "Deactivate" event is only used to detect whether an object was deselected. The event does not trigger a password prompt.

For this reason, do not use the "Deactivate" event if you want to configure access protection on the function call of the object.

Double-click (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the user double-clicks on an object from the symbol library.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Press (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the user clicks on a button with the left mouse button, presses <RETURN> or <SPACE>.

Also occurs when the user right-clicks on an object of the symbol library.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

On Finish Input (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Triggered when you confirm input at an I/O field by pressing ENTER, or by mouse click, or by touch screen operation.

The "On Finish Input" event is also started if the value of a tag does not change, for example, if a value is exceeded, or if a user cancels the dialog to acknowledge a tag (Audit option package) that has to be acknowledged.

The event is not triggered, on the other hand, by user logon or by input fields configured with an authorization.

Note

Note that the availability of the event depends on the HMI device and object type.

Press ESC twice (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the user presses the <ESC> key twice at the HMI device.

Note

Note that the availability of the event depends on the HMI device and object type.

Outgoing (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when an alarm is deactivated.

Note

Please note that the availability of the event depends on the HMI device and object type.

Incoming (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when an alarm is triggered and displayed in the alarm view.

Note

Please note that the availability of the event depends on the HMI device and object type.

Click (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs if the user clicks a display and operating object with the mouse or touches the touch display with a finger.

In case you click the incorrect object, prevent processing of configured function list as follows:

- Move the mouse pointer away from the object while keeping the mouse button pressed. Release the mouse button as soon as the mouse pointer leaves the object. The function list will then not be processed.
- On touch displays, the display must be touched with the finger until a reaction occurs, e.g., a screen change.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Click when flashing

Description

Occurs when the user clicks a flashing alarm indicator with the mouse or touches it with a finger.

Note

Note that the availability of the event depends on the HMI device and object type.

Loop-in alarm (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs as soon as the user selects an alarm in the alarm view and then clicks on the "Loop-In-Alarm" button or double clicks on the alarm.

For the "Loop-In-Alarm" event, you configure system functions, such as a change to the screen in which the alarm occurred.

You cannot configure local scripts for the "Loop-In-Alarm" event in Runtime Professional.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Release (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the user releases a button.

This even does not occur, as long as the button remains pressed.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Alarm buffer overflow (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the configured size of the alarm buffer is reached.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Acknowledge (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the user acknowledges an alarm.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Reach margin (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the user reaches the beginning or the end of a scrollable area.

Note

Note that the availability of the event depends on the HMI device and object type.

Note

A user-defined function must not be configured for the "Boundary reached" event.

Configurable objects

The event can only be configured on the <Up> and <Down> keys, or on the keys on which you have configured the "ScreenObjectPageUp" or "ScreenObjectPageDown" system functions.

Runtime Stop (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Description

Occurs when the user exits the Runtime software on the HMI device.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Note

A user-defined function must not be configured for the "Runtime stop" event.

Press key (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the user presses a function key.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Release key (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the user releases a function key.

Note

Note that the availability of the event depends on the HMI device and object type.

Overflow (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the configured size of the log is reached. You use the log type "Trigger event".

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

Switch OFF (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the user moves the display and operating object "Switch" to the OFF position.

Note

Note that the availability of the event depends on the HMI device and object type.

Switch ON (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the user moves the display and operating object "Switch" to the ON position.

Note

Note that the availability of the event depends on the HMI device and object type.

Low free storage space (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

This event is triggered if the storage space available on the medium to which the Audit Trail is less than the configured minimum.

Free space critically low (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

This event is triggered if the storage medium to which an Audit Trail is saved provides insufficient storage space due to hardware restrictions.

Value change (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the value of a tag or the value of an array element changes.

The value change of a tag is triggered by the PLC or by the user, e.g. when a new value is entered. No event is output if a system function changes the value.

Note

Please note that the availability of the event depends on the HMI device and object type.

Time expired (Basic Panels, Panels, Comfort Panels, RT Advanced)

Description

Occurs when the time configured in the scheduler expires.

Note

Please note that the availability of the event is dependent upon the HMI device and object type.

12.8 Planning tasks

12.8.1 Basics

12.8.1.1 Field of application of the Scheduler (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Definition

You can use the Scheduler to configure tasks to run independent of the screen in the background. You create tasks by linking system functions or scripts to a trigger. The linked functions will be called when the triggering event occurs.

Example of an application

The Scheduler is used to execute event-controlled tasks automatically. For example, you use a task to automate the following:

- Regular swap out of log data
- Printout of an alarm report when an alarm buffer overflow occurs
- Printout of a report at shift end
- Monitoring a tag
- Monitoring of a user change

Note

The availability of the listed examples is determined by the HMI device.

See also

Working with tasks and triggers (Page 5781)

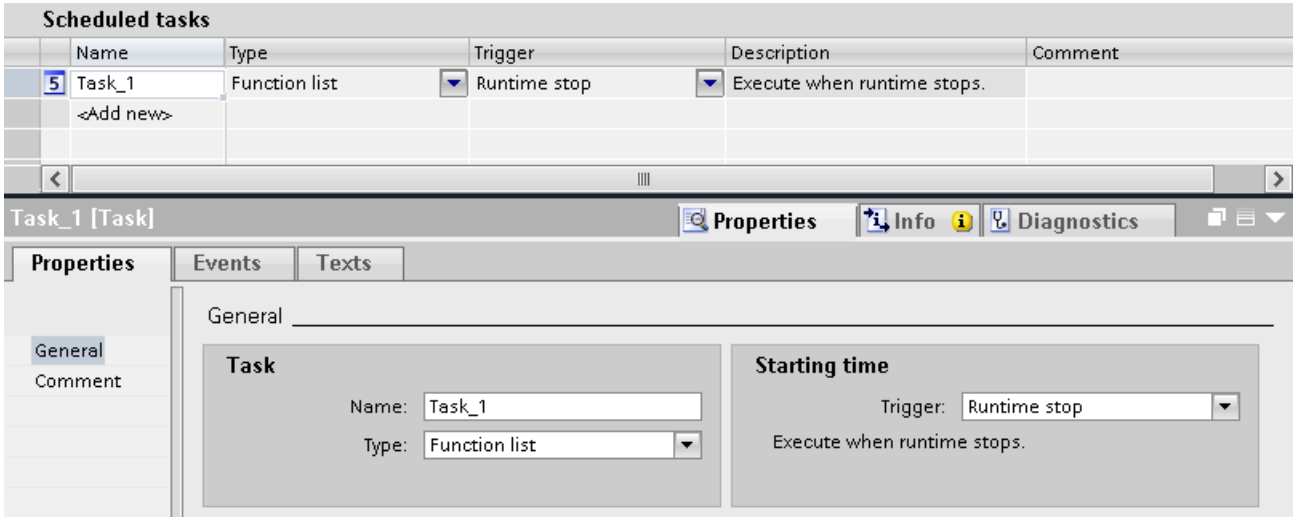
Example: Update user following change of user (Page 5786)

Work area of the "Scheduler" editor (Page 5781)

12.8.1.2 Working with tasks and triggers (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

A task consists of a trigger and a task type.



Starting a task

Controlled by a trigger, the Scheduler starts the task linked to the trigger.

See also

Field of application of the Scheduler (Page 5780)

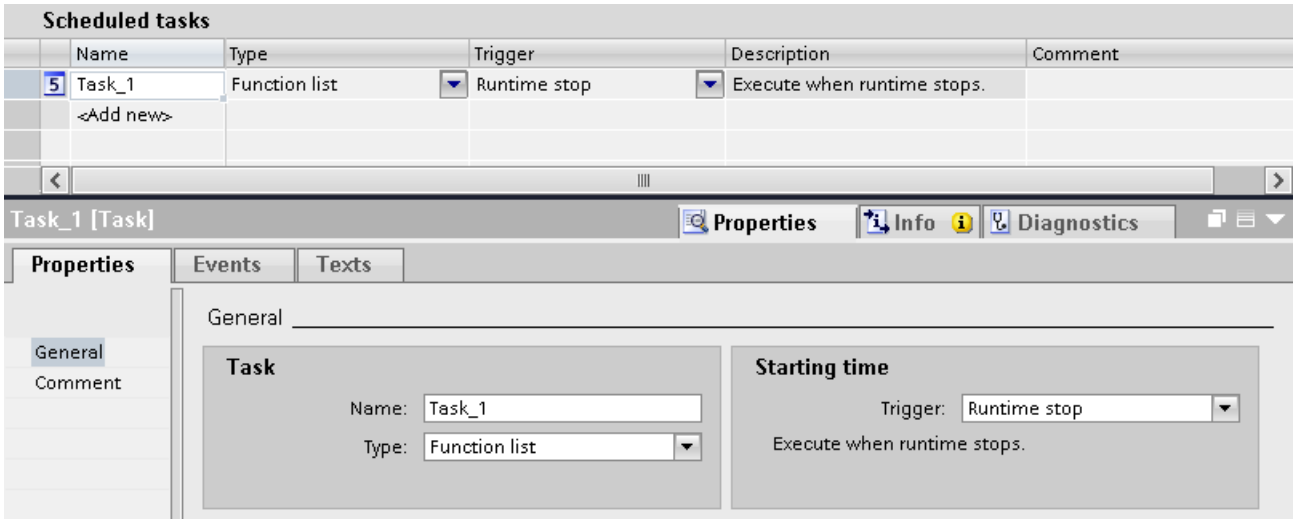
12.8.1.3 Work area of the "Scheduler" editor (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Double-click on "Scheduler" to open it in the project view. The work area shows the scheduled tasks, which consist of the trigger and the task type, for example, the function list.

Structure

The work area consists of the table of jobs.



The table of tasks shows specified tasks with their properties, such as triggers. You select a task type and a trigger. You assign a name and a comment to the task. The description provides a written summary of the task including the timing for the task.

Inspector window

The "Properties" tab of the Inspector window is split into two parts.

The "Job" area lists the name of the job and the job type. The "Starting time" area shows the trigger. The area is different depending on the trigger selected.

In the "Events" tab use the function list with system functions that will be executed in the task.

Note

You can obtain more detailed information about the elements of the user interface using the tooltips. To do so, move the mouse pointer to the relevant object or press <F1> if the object has already been selected.

See also

Field of application of the Scheduler (Page 5780)

Planning tasks with event triggers (Page 5784)

Triggers (Page 5783)

Function list (Page 5783)

12.8.1.4 Function list (Basic Panels)

Function list

A trigger starts the function list. The function list is executed line by line. Each line contains a system function. You can configure exactly one function list for each task.

Note

The choice of configurable system functions in a function list depends on the selected trigger and the HMI device.

See also

Work area of the "Scheduler" editor (Page 5781)

12.8.1.5 Triggers (Basic Panels)

Introduction

A trigger is linked to a task and forms the triggering event which will call this task. The task is executed when the trigger occurs.

Event trigger

When a task is linked to a system event, the task will be triggered by the event. System events include, for example, Runtime stop, screen change, user change, etc.

Each system event can only be configured once for each HMI device.

Deactivating job

If you do not need a certain job temporarily, deactivate the job in the Engineering System. You also use the trigger "Deactivated" to make a previously configured system event available once again.

Example: Task "A" is planned with the system event "Shutdown". This system event is then no longer available for another task "B". Select "Disabled" as the trigger for task "A" to make the "Runtime stop" system event available again.

Note

The available triggers depend on the HMI device.

See also

Work area of the "Scheduler" editor (Page 5781)

12.8.1.6 Planning tasks with event triggers (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You plan a task that generates a screen change when the user changes.

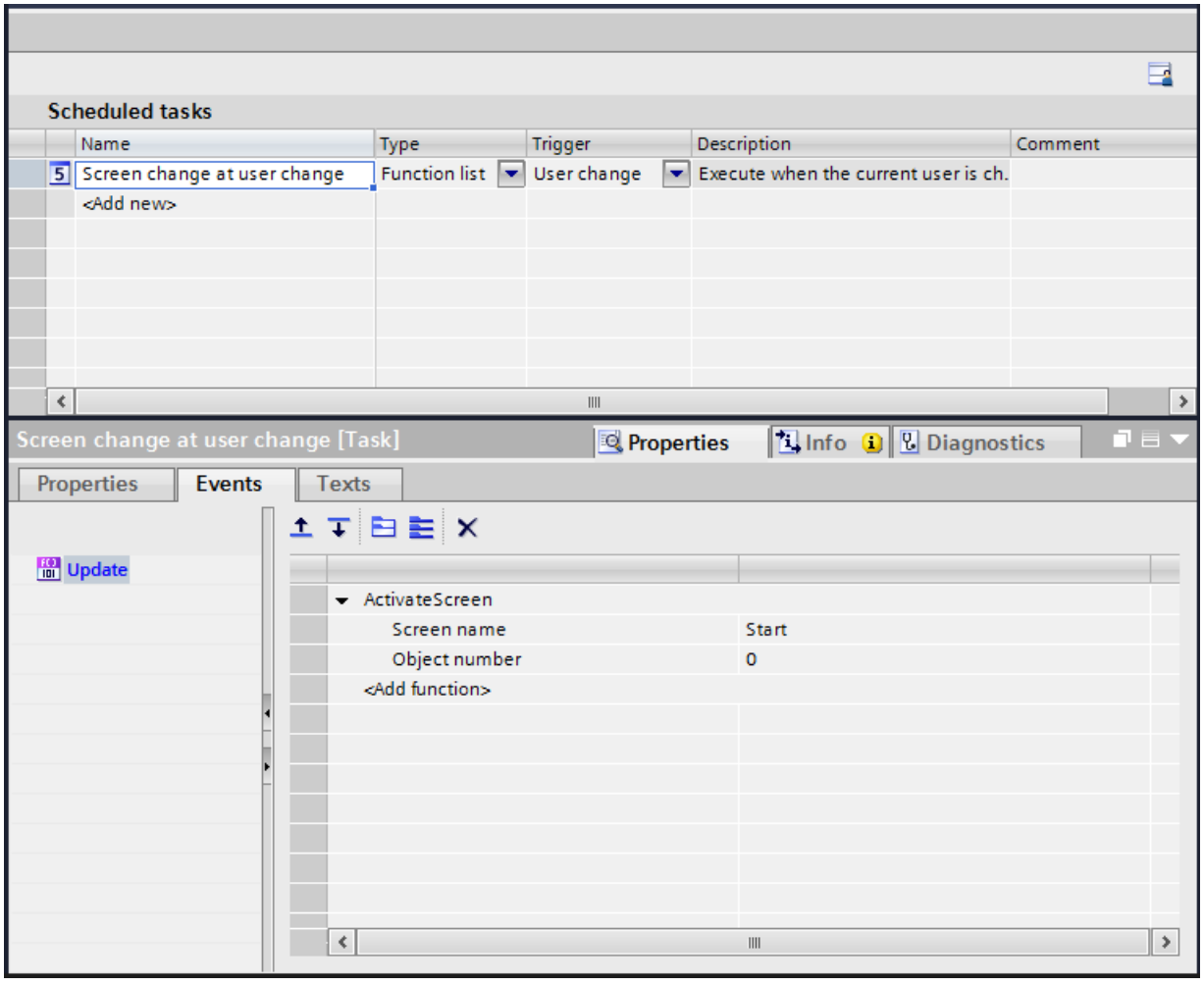
Requirements

- The "Scheduler" work area is open.
- You have created the "Start" screen.

Procedure

1. Click "Add..." in the table of the task area.
2. Enter "Screen change at user change" as the "Name."
3. Select "User change" as the "Trigger."
4. In the Inspector window, select "Properties > Events".

5. Select the "Screen/ActivateScreen" system function in the function list.
6. Select the "Start" screen in the screen name field.



Result

The task is executed with the "User change" event. When a new user logs on successfully, the "Start" screen is called up.

See also

Work area of the "Scheduler" editor (Page 5781)

12.8.2 Examples

12.8.2.1 Example: Update user following change of user (Basic Panels, Panels, Comfort Panels, RT Advanced)

Task

Configure an I/O field which displays the logged on user. Configure a task which updates the I/O field when the logged on user changes.

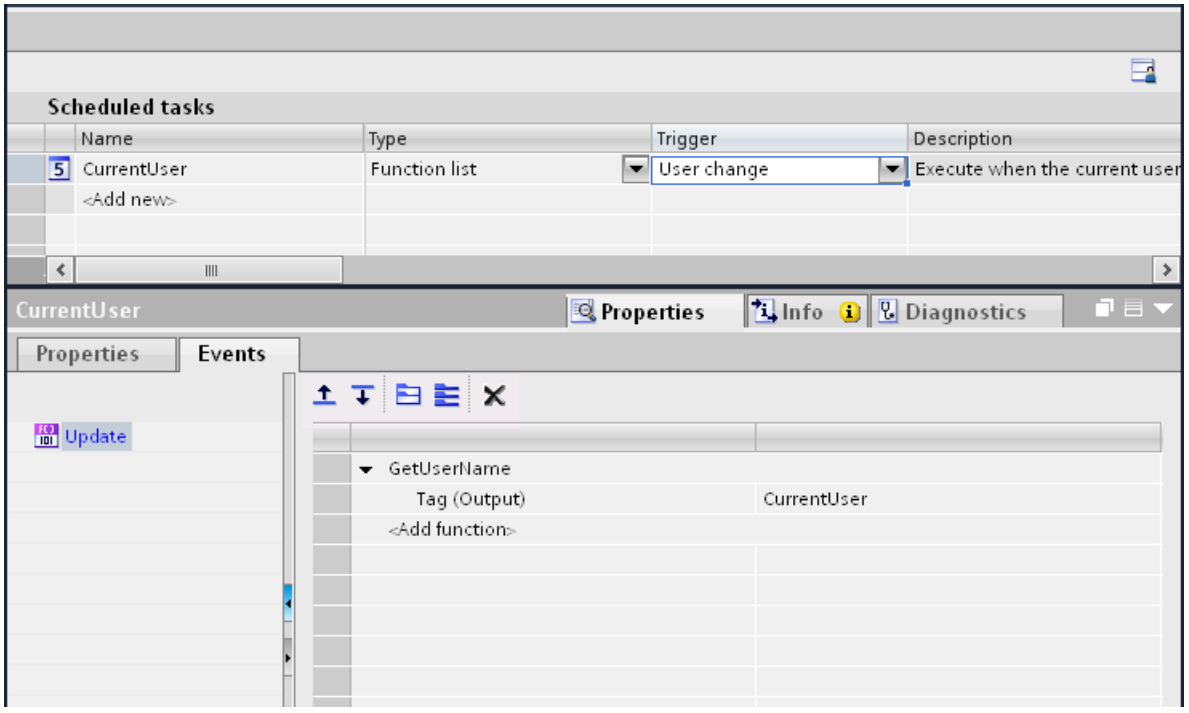
Requirements

- A "CurrentUser" tag of the "String" type is created.
- A screen has been created and opened.
- An I/O field is created in the screen.

Procedure

1. Click on the "I/O field" object.
2. In the Inspector window, select "Properties > Properties > General":
   - Select "String" as the "Display format."
   - Select "CurrentUser" as the "Variable."
   - Select "Output" as the mode.
3. Change to the work area of the Scheduler.
4. Click "Add..." in the table of the task area.
5. Enter "CurrentUser" as the "Name".
6. Select "User change" as the "Trigger."
7. In the Inspector window, select "Properties > Events".

8. Select the system function "ReadUserName" from the "User Management" group in the function list.
9. Select "CurrentUser" as the "Variable."



Result

When a new user logs on successfully, the "ReadUserName" function is called up. The "CurrentUser" tag is updated and displayed in the I/O field of the newly logged on user.

If a user does not log on successfully, the logged on user is logged off. The I/O field continues to display the user previously logged on until a new user logs on successfully.

See also

Field of application of the Scheduler (Page 5780)

12.9 Using diagnostics functions

12.9.1 Configuring system diagnostics

12.9.1.1 System diagnostics basics (Basic Panels)

Introduction

Using system diagnostics, you can display the messages from the diagnostic buffer of all integrated connections.

System diagnostics view

The system diagnostics display is an operating and display object that you use in a screen.

You navigate directly to the cause of the error and the associated connection. You have access to all integrated connections that you have configured in the "Devices & networks" editor.

12.9.1.2 System diagnostics views (Basic Panels)

Introduction

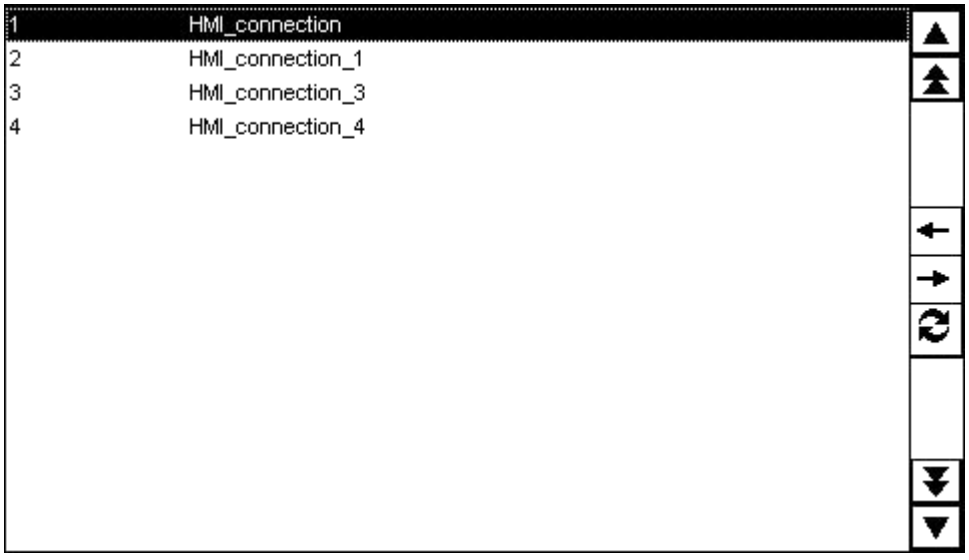
Three different views are available in the simple system diagnostic view.

- Device view
- Diagnostic buffer view
- Detail view

Device view

The device view is only displayed if more than one integrated connection has been configured.

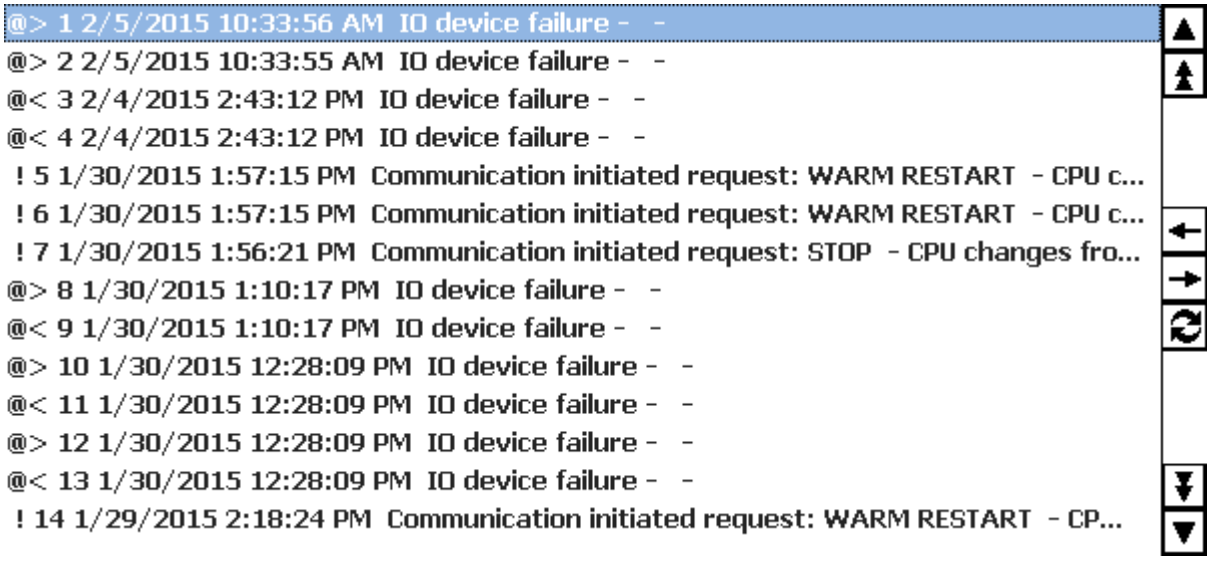
The device view shows all available connections in a table. Double-clicking a connection opens the diagnostic buffer view.



Diagnostic buffer view

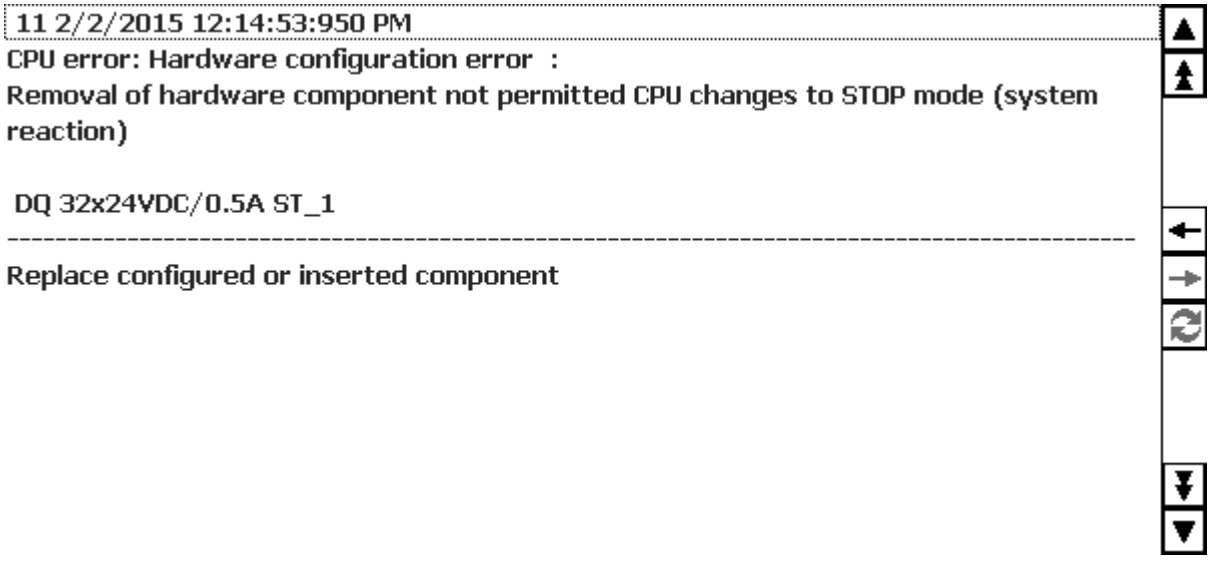
The diagnostic buffer view shows the current data from the diagnostic buffer.

To update the diagnostic buffer view, you click .



Detail view

The detail view gives detailed information about the selected connection and any pending errors. Check whether the data is correct in the detail view.



Note

Contents of the detail view

The contents of the detail view are only available for integrated connections with S7-1200 and S7-1500 PLCs.

Navigation buttons

Button	Key	Function
	Enter key	In the device view: Opens the diagnostic buffer view of the selected device. In the diagnostic buffer view: Opens the detail view.
	Esc key	In the diagnostic buffer view: Opens the device view. In the detail view: Opens the diagnostic buffer view.
	Configured function key, e.g. F1.	Updates the diagnostic buffer view.

See also

Configuring the system diagnostic view (Page 5791)

12.9.1.3 Configuring the system diagnostic view (Basic Panels)

Introduction

For an overview of all integrated connections, you insert a system diagnostics view in your project.

Requirements

- A PLC has been created.
- A Basic Panel has been created.
- An integrated connection has been created in the "Devices & Networks" editor.
- You have created a screen.
- The Inspector window is open.

Procedure

1. Double-click the "System diagnostics view" object in the "Tools" task card. The object is added to the screen.
2. In the Inspector window, select "Properties > Layout".
3. Enter a number under "Lines per entry", i.e. 5.

Result

The system diagnostics view has been added to the screen.

To obtain the latest alarms, update the diagnostic buffer.

See also

System diagnostics views (Page 5788)

12.10 Communicating with controllers

12.10.1 WinCC Basic V14 SP1 - Communication

Communicating with controllers

You can find a detailed description of communicating with controllers in the "WinCC Basic V14 SP1 - Communication" manual.

12.11 Using global functions

12.11.1 Working with libraries

12.11.1.1 Basics on libraries

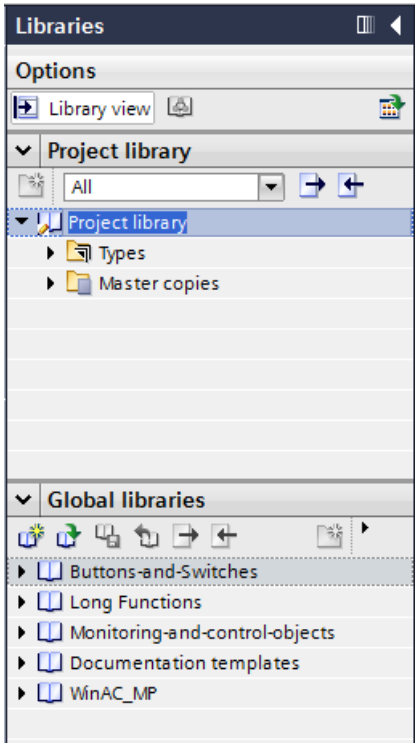
Introduction

Store all objects you need frequently in the libraries. An object that is stored in the library only has to be configured once. It can then be used repeatedly as often as required. Library objects extend the number of available screen objects and increase the effectiveness during configuration through the multiple usage of ready-to-use objects.

Your WinCC software package is supplied with comprehensive libraries that contain, for example, "Motor" or "Valve" objects. You may, however, define your own library objects.

You manage libraries in the "Libraries" task card or library view. The following libraries are available:

- Project library
- Global libraries



Note

There is a symbol library in the "Tools" task card in the "Graphics" palette.

Project library

There is one library for each project. Objects of the project library are stored alongside with the project data and are available only for the project in which the library was created. If the project is moved to another PC, any project library created in it is also moved.

To use the library object of the project library in other objects, move or copy the object into a global library.

Global libraries

A global library is saved independently of the project data in its own file with the extension *.al14.

A project can access several global libraries. A global library may be used concurrently in several projects.

When a library object is changed by a project, this library will be changed in all projects in which these libraries are open.

Library objects

A library can contain all WinCC objects. Examples:

- Complete HMI device
- Screens
- Display and control objects including tags and functions
- Graphics
- Tags
- Alarms
- Text and graphics lists
- Faceplates
- User data types
- Styles and style sheets

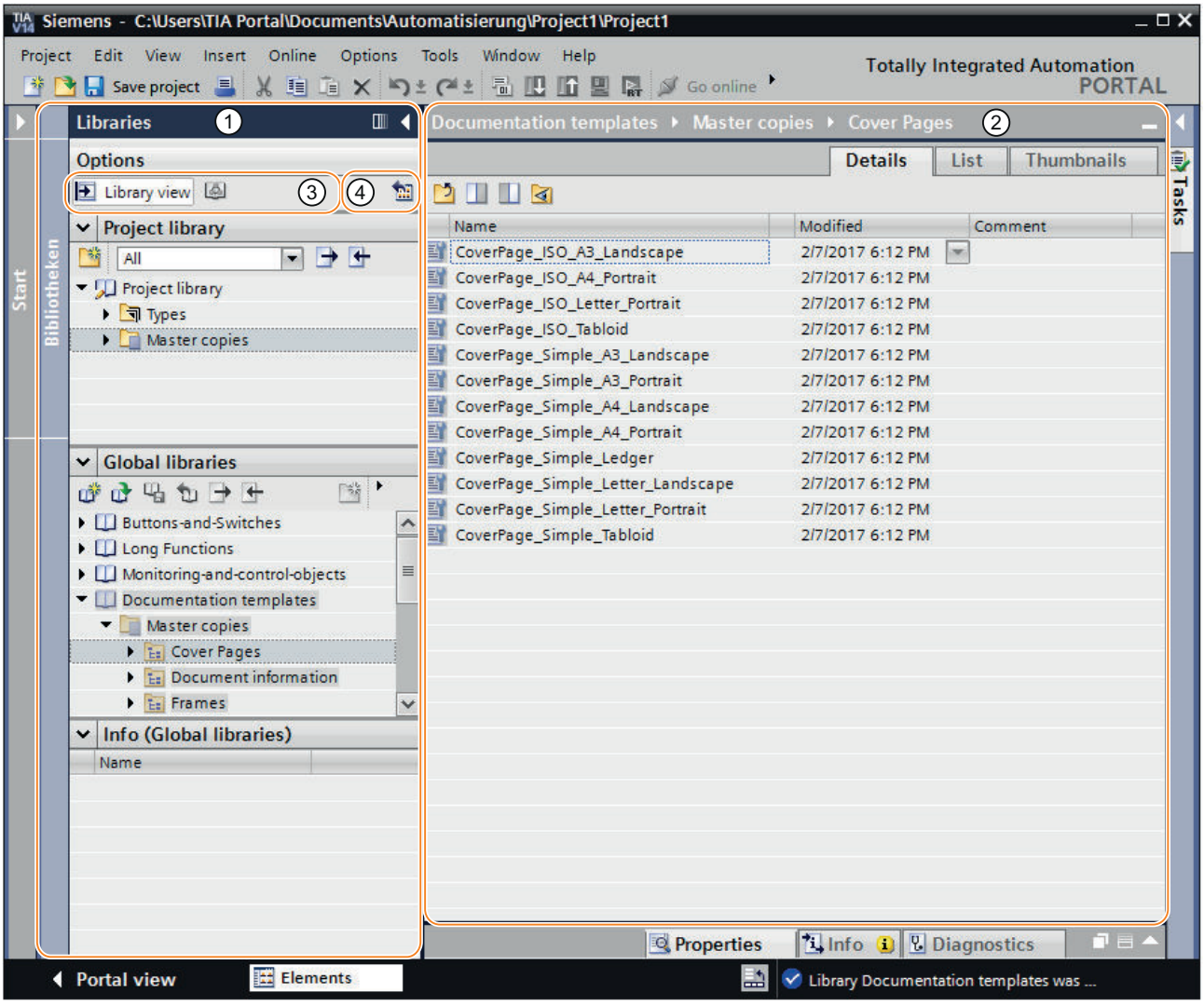
12.11.1.2 Overview of the library view

Function of the library view

The library view combines the functionality of the "Libraries" task card and the overview window. In the library view, the elements of a library are displayed in various views. In the details view, for example, you see additional properties of the individual elements. You can also edit and version the types in the library view.

Layout of the library view

The following figure shows the components of the library view:



1 Library tree

2 Library overview

3 "Library view" button

4 "Open or close library overview" button

Library tree

The library tree is similar to the "Libraries" task card, apart from a few minor differences. In contrast to the task card, there is no "Elements" palette, because the elements are displayed in the library overview. In addition, you can close the library view in the library tree, or open and close the library overview.

Library overview

The library overview corresponds to the overview window and displays the elements of the currently selected object in the library tree. You can display the elements in three different views. You can also perform the following actions in the library overview, for example:

- Copying elements
- Moving elements
- Versioning types
- Editing faceplates and HMI user data types
- Editing type instances

12.11.1.3 Master copies and types

Introduction

Both the "Project library" and the "Global library" contain the two folders "Master copies" and "Types". You can create or use the library objects either as a master copy or a type.

Master copies

Use master copies to create independent copies of the library object.

Types

Create instances of objects of the "Types" folder and use these in your project. The instances are bound to their respective type.

Administration of the library objects

You can copy and move library objects to another library. You can only copy master copies to the "Master copies" folder or any sub-folder of "Master copies". You can also only insert types in the "Types" folder or any sub-folder of "Types".

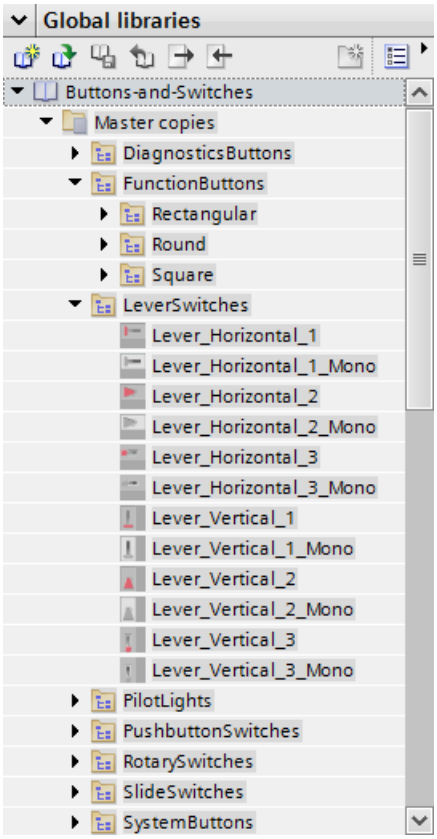
12.11.1.4 Libraries in WinCC

Introduction

The WinCC software package includes extensive libraries. Sorted by topic into folders, they contain preassembled graphic objects which you can use in screens for operation and monitoring of your plant.

Global library "Buttons and Switches"

The libraries "Buttons and Switches" offer a wide selection of buttons and switches.



The folders divide switches and buttons into categories. The "DiagnosticsButtons" folder contains the object "System diagnostics indicator", for example. You use the "System diagnostics indicator" object for system diagnostics in your plant.

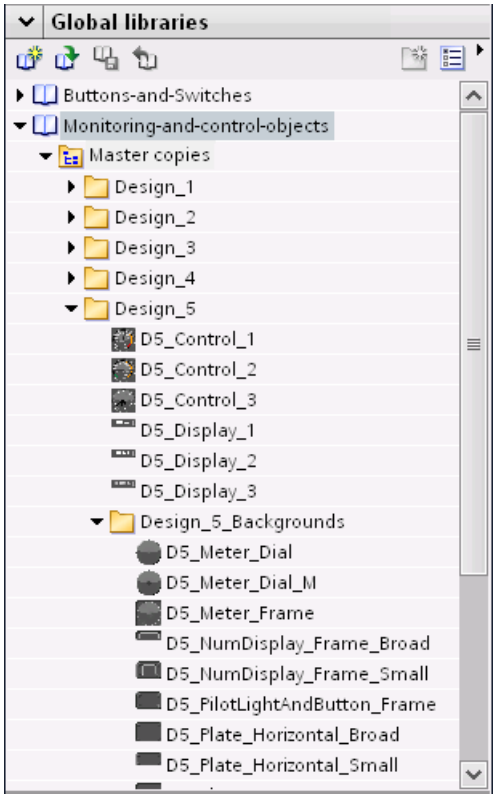
Note

You can only use the objects in the "DiagnosticsButtons" folder on Comfort Panels.

You cannot use objects which have "Switch" in the object name or in the associated folder name in Runtime Professional.

Global library "Monitoring and Control objects"

The "Monitoring and Control objects" library offers more complex display and operating objects in several designs and corresponding control lights, buttons and switches.



In addition, graphics views for the designs are stored in the "Design_Backgrounds" folder; they can be used as object backgrounds for the custom extension of the scope of the library.

Note

You cannot use objects which have "Switch" in the object name in Runtime Professional. The same applies for the object "D5_Display_3" with the date/time field it contains.

12.11.1.5 Managing libraries

Overview of the library management

Function of the library management

Master copies and types with dependencies to other library elements are subject to some functional restrictions. They can, for example, not be deleted as long as dependencies still exist. This prevents other library elements from becoming useless. The library management is used to identify the dependencies and to create an overview of the work progress.

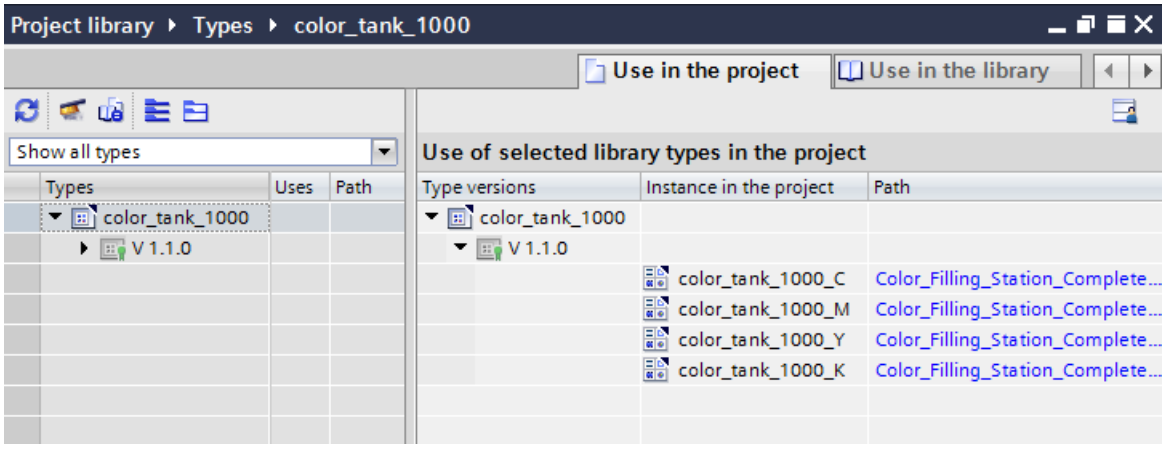
The library management offers the following functions:

- Display of the correlations of types and master copies
  If a type is referenced in other types or master copies, the correlations are displayed in the library management. You will also be able to see which library elements reference a type or a master copy.
- Display of points of use of types in the project
- Display all types that include a version with the "In test" or "In progress" status

Layout of the library management

The figure shows the library management, consisting of the following components:

- Toolbar of the library management
- "Types" area
- "Use" area



Toolbar of the library management

You can perform the following tasks in the toolbar of the library management:

- Update view
  If the project was changed, you can update the view of the library management.
- Clean up library
  You can clean up the project library and global libraries. By cleaning up a library, you delete all types and type versions that are not linked to an instance in the project.
- Harmonize project
  By harmonizing a project, you adapt the names and the path structures of type uses in the project to the corresponding names and path structures of the types within a library.

"Types" area

The "Types" area displays the contents of the folder that you have selected in the library view. You can open or close all types by using the buttons in the toolbar of the "Types" area. You can also use the "Filter" drop-down list to filter the view and display all types with the "In test" or "In progress" status only. For each type, the types that it references are displayed.

"Uses" area

The "Uses" area gives you an overview of the points of use of the selected types and master copies. The "Uses" area is divided into two tabs:

- "Uses in the project" tab
  The "Uses in the project" tab is used to show the instances of type versions and their respective point of use in the project. When you select an instance, you can show the cross references of the instance in the project in the Inspector window.
- "Uses in the library" tab
  The "Uses in the library" tab is used to show all points within the library at which a type or a master copy is used.

Opening library management

Procedure

To open the library management, follow these steps:

1. Open the library view.
2. Select a type or any folder that contains types.
3. Select the "Library management" command from the shortcut menu.

Result

The library management opens and the types are displayed with their versions.

Filtering types in the library management

Introduction

A filter function in the library management enables you to limit the displayed types. The following filters are available:

- Display all types that include a version with "in progress" status
- Display all types that have no instances in the project
- Display all types that have more than one version
- Display all released types

Requirement

At least one type has been created.

Filtering according to all types with pending changes

1. Select the "Types" folder in the project library.
2. Select the "Library management" command in the shortcut menu of the "Types" folder.
   The library management opens.
3. Select "Types with pending changes" from the "Filter" drop-down list.
   The "Types" area only displays types that have the "in progress" status.

Filtering for all types that have no instances in the project

1. Select the "Types" folder in the project library.
2. Select the "Library management" command in the shortcut menu of the "Types" folder.
   The library management opens.
3. Select "Types not used in the project" from the "Filter" drop-down list.
   The "Types" area only displays types that have no instances in the project.

Filtering for types with multiple versions

1. Select the "Types" folder in the project library.
2. Select the "Library management" command in the shortcut menu of the "Types" folder.
   The library management opens.
3. Select "Types with multiple versions" from the "Filter" drop-down list.
   The "Types" area only displays types that have more than one version.

Filtering for all released types

1. Select the "Types" folder in the project library.
2. Select the "Library management" command in the shortcut menu of the "Types" folder.
   The library management opens.
3. Select "Released types" from the "Filter" drop-down list.
   The "Types" area only displays types that have released versions.

Filtering according by types with new versions for upgrading other types

1. Select the "Types" folder in the project library.
2. Select the "Library management" command in the shortcut menu of the "Types" folder.
   The library management opens.
3. Select "Types with new versions for upgrading other types" from the "Filter" drop-down list.
   The "Types" area only displays types that can be potentially used as a source for upgrading other types.

Creating a global library

Introduction

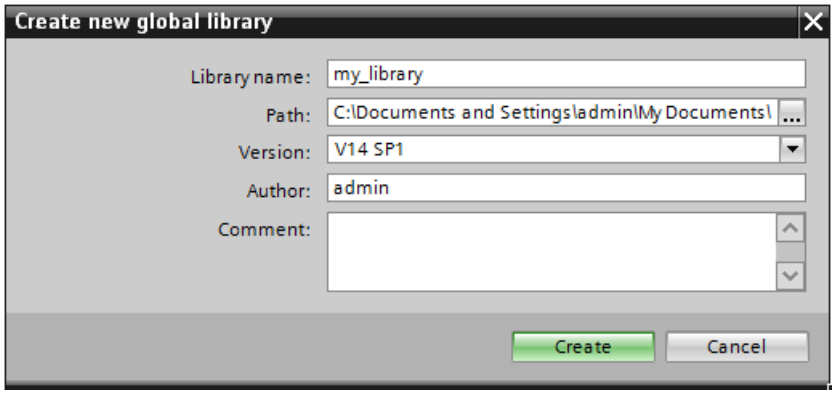
In the libraries you store the configured objects that you want to use several times in your configuration. To use objects in several projects, create a global library.

Requirement

- A project is open.
- The "Libraries" task card is displayed or the "Library view" is open.

Procedure

1. Click the icon under "Global library".
   The "Create new global library" dialog opens.



2. Enter a name.

3. Select the path where the new library is to be stored.
4. Click "Create".

Result

The new library is shown in the "Global libraries" palette. The global library contains the "Types", "Master copies" and "Common data" folders. Under the "Common data" you can find reports for the global library.

A folder with the name of the global library is created in the file system at the storage location of the global library. This actual library file is given the file name extension ".al12".

Saving a global library

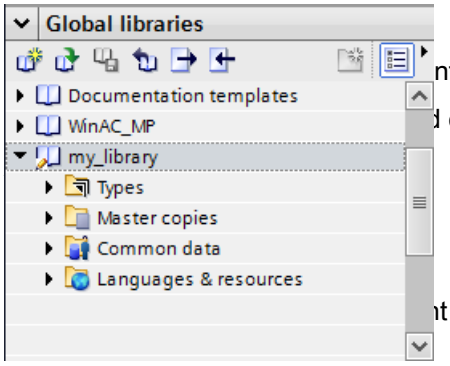
Introduction

A global library is stored in a separate file on your hard disk drive. The file contains the objects of the global library including the referenced objects. E.g. the reference of a tag which was configured on an I/O field is also saved in the library.

WinCC prompts you to save the global libraries when you close WinCC or your project without saving. You also can store the global library during configuration, without storing the entire project.

Requirement

- ntains at least one library.
- or the library view is open.
-



Procedure

1. t to save.
2. Click the icon in the "Global library" palette.

You can alternatively select the "Save Library" command in the shortcut menu.

If you want to store the global library in a different folder, select "Save as" in the shortcut menu. Select the path in which you want to store the new library and enter a file name.

Result

The global libraries are saved under their current file name or a file name you have specified.

Opening a global library

Introduction

In WinCC, the global libraries are stored in separate files. You can use a global library in every project.

Requirement

- You have saved a global library.
- A project is open.
- The "Libraries" task card is displayed or the library view is open.

Procedure

1. Click the  icon in the "Global library" palette.
   The "Open global library" dialog box is displayed.
2. Select the path in which the library is stored.
3. Click "Open".

Note

To have the access to a global library from multiple projects, open the global library read-only. As soon as a global library is not opened read-only, access from other projects is blocked.

Result

WinCC displays the opened global library in the "Global library" palette.

Displaying logs of global libraries

Logs listing all changes made to the global library are created when global libraries are updated. The logs are stored together with the global library and are always available once you have opened the global library.

Procedure

To open the logs of a global library, follow these steps:

1. Open the global library in the "Libraries" task card or in the library view.
2. Open "Common data > Logs" in the lower-level folder.
3. Double-click the required log.
   The log opens in the work area.

Updating a project with the contents of a library

Introduction

After you have edited several types in the project library, update all instances in the project to the most recent version of the types from the project library.

Requirement

The "Libraries" task card or the library view is open.

Procedure

1. Select the project library.
2. Select "Update > Project" from the shortcut menu. A dialog opens.
3. Select either the entire project or individual devices for the update.
4. Select the "Delete all unused versions of affected types" check box to delete all older versions of the updated types from the project library.
5. Click "OK" to confirm.

Result

All instances of the types are updated in the project to the most recent version of the selected types in the project library.

You can find a log of the update process in the project tree under "Common data".

Updating a library with the contents of another library

The following options are available for updating libraries:

- Updating a global library with types from another global library
- Updating the project library with types from a global library

Each of the following elements can be selected as source for the update:

- An entire library
- Individual folders within a library
- Individual types

Requirement

To update a global library, open the library with write rights.

Requirement

The "Libraries" task card or the library view is open.

Procedure

To update a library with the contents of a different library, follow these steps:

1. Select the entire library or a folder within the library or individual types.
2. Right-click the source and select the "Update > Library" command from the shortcut menu. The "Update library" dialog box opens.
3. Select the type of library you want to update:
   - Select "Update the project library" to update the project library with types from a global library.
   - Select "Update a global library" if you want to update a global library.
4. Optional: Select the global library you want to update from the drop-down list.
5. Select the "Delete all unused versions of affected types" check box to delete all older versions of the updated types from the project library.
6. Click "OK" to confirm.

Result

- Types not yet available in the target library are supplemented there with all their versions. More recent versions are added to the types that already exist in the target library.
  If a more recent version of a type already exists in the target library, the latest version is nevertheless copied and automatically assigned a newer version number.
- A log listing all performed changes to the target library is created for the update process.
  If you have updated the project library, you can find the log in the project tree under "Common data > Logs".
  If you have updated a global library, you can find the log in the "Common data > Logs" folder in the level below the global library.

Upgrading types of a library

Description

Your project contains objects which use or reference other objects, e.g. user data types. You have created a project library and stored the library types used in the project library. You have continued to edit the types outside the project. Reintegrate the new versions of the edited types into the project library.

With the help of the “Upgrading of types” function, you can recognize all the referenced types which are dependent on a changed type and automatically upgrade them in the entire project. The upgraded types are set to the status "in progress" or "in test" in the project library. You can thus see at a glance which types were changed in order to test and release them.

Requirement

- The project library has been updated with the types of the global library.
- The new type versions have been integrated into the existing project library.

Procedure

1. Open the library management.
2. In the "Types" area, select "Types with new versions for upgrading other types" from the drop-down list.
   All types for which a new type version is available are displayed in the project library.
3. Click the "Types" folder and select "Display upgradable types" from the shortcut menu.
   All referenced types are displayed in the "Upgrading of types" tab.
4. Select one of the options under "Upgrading of types":
   - "With test environment only" in order to display all types that reference an instance in the test environment.
   - "Compatible with selected test environment" in order to display all types that can reference an instance in the test environment. The instance was not yet created.
5. Select a test environment.
   All of the types that must be upgraded are displayed.

   Note

   If one of the types to be upgraded has the "Error" status, the types cannot be upgraded. First correct the error and restart the procedure.

6. Click "Start".
   The "Upgrading of types" dialog opens after successful upgrade.
7. Check the upgraded types and then release them.

Result

All types in your project were upgraded and reference the current type versions.

See also

Updating a project with the contents of a library (Page 5806)

Updating a library with the contents of another library (Page 5806)

Exporting and importing library texts

Introduction

You can export the texts of the library objects to an .xlsx file to edit them in MS Excel, for example, or export them for compilation.

You export and import texts of the following objects in the library:

- Individual library types and master copies
- Multiple library types and master copies
- All library objects of the project library or a global library

After editing or external compilation, you import the texts into the TIA Portal.

When the texts are imported, all texts from the import file are imported for the entire library, even if you have only selected one library object. The target languages of the import file must be activated in the project.

During import to a master copy, the texts of the template are overwritten in the library with the new texts from the import file. During import of the texts to a library type, the latest version is overwritten in the library with the new texts from the import file. If a version of a type has not been released yet in a project library, no texts can be imported for the entire project library.

Note

Restrictions for text import

The texts which belong to the following library objects cannot be imported:

- Type instances which are contained in a master copy
- Library types whose versions have not yet been released and which have the status "in progress" or "in test"
- Write-protected global library

Defining the source language and target language

You define the source language and target language for export of the texts in the Export dialog. The selection of available languages depends on the project languages defined.

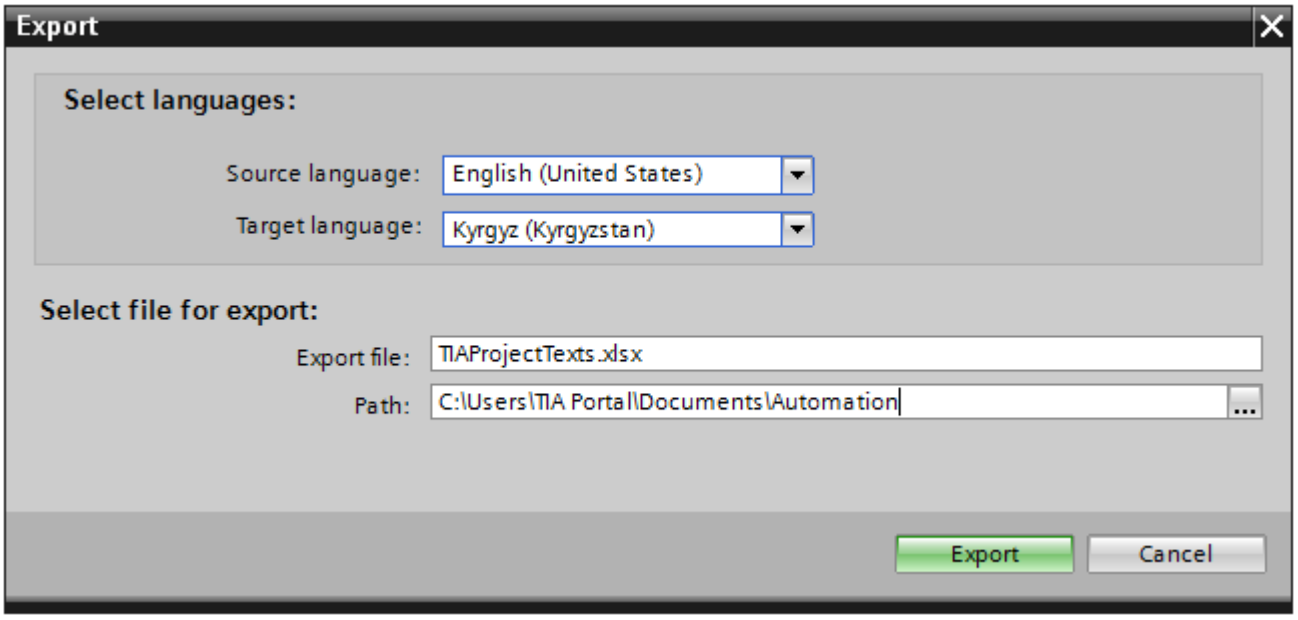
With the global library, the selection of available source and target language depends on the languages defined by the creator of the library. To see the available languages of the global library, double-click the entry "Library languages" in the project folder "Language and Resources" of the library in question.

Exporting texts

To export the texts of a single or several library objects, follow these steps:

1. Open the library.
2. Select the desired library object in the library.
3. Select the command "Export library texts" in the shortcut menu of the object.
   Alternatively, click the  "Export library texts" button in the toolbar.
   The "Export" dialog box opens.
4. Select the source language and the target language for export in the dialog.

5. Enter the name and the path for the export file.



6. Click "Export".
   After successful export, the export file is stored under the specified path.

Importing texts

To import the texts after editing or compilation into the TIA Portal again, follow these steps:

1. Open the project library or the global library.
2. Select the command "Import library texts" in the shortcut menu of the object.
   Alternatively, click the "Import library texts" button in the toolbar.
   The "Import" dialog box opens.
3. Select the path and the name of the import file from the "Select file for import" field.
   Activate the "Import source language" check box if you have made changes to the source language in the export file and would like to overwrite the entries in the project with the changes.
4. Click on "Import".

12.11.1.6 Managing objects in a library

Displaying library objects

Introduction

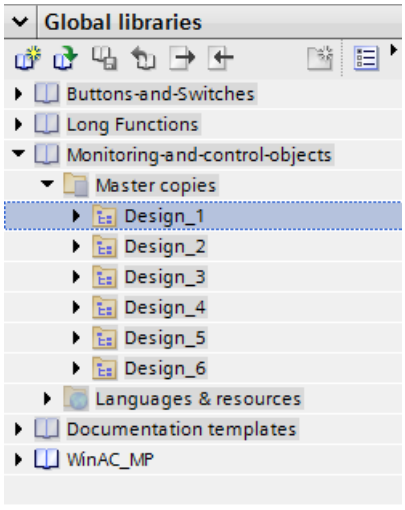
The libraries are displayed as file folders in the corresponding palette. The elements contained in the library are displayed in the file folder and in the "Elements" palette.

Requirement

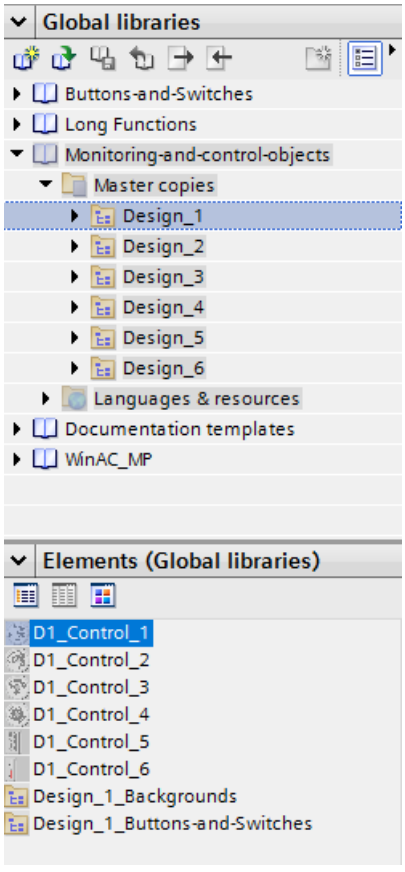
- At least one library object has been created in a library.
- The "Libraries" task card is opened.

Procedure

1. Select the library in the corresponding palette whose library objects you want to display.



2. Click .
The contained library objects are displayed in the "Elements" palette.



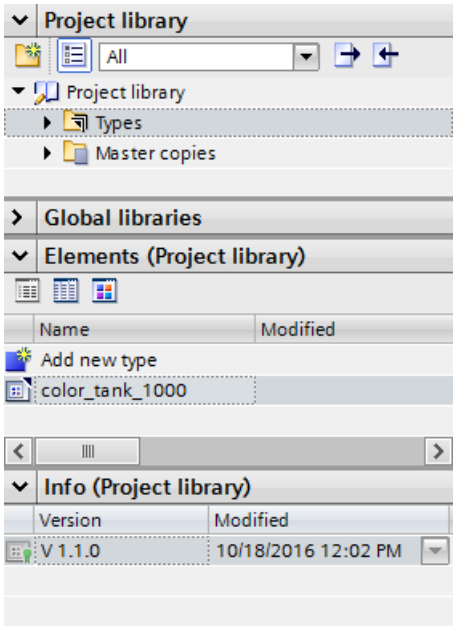
3. Click one of the following icons:

Icon	Description
	Element view in detailed mode
	Element view in list mode
	Element view in overview mode with icons

When several objects are assigned to the library with a multiple selection, only one of the objects is shown in the "Elements" palette. The individual components of this element are displayed in the "Parts" palette.

Show parts of the library objects

1. Select the library in the corresponding palette from which you want to view the components of an element.
2. Click .
3. The contained library objects are displayed in the "Elements" palette.
4. Select the element.
   The "Parts" palette shows the objects of which the element consists.



Result

The library objects are displayed in accordance with the configuration. The components of the faceplates are displayed.

Storing an object in a library

Introduction

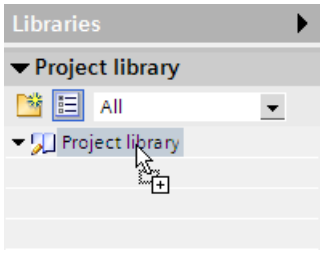
You can store all of WinCC objects, such as screens, tags, graphic objects or alarms in your libraries. You can use drag-and-drop to move the corresponding object from the work area, project window or detail view to the library. In a library you have divided into categories, you can directly add objects to a specific category.

Requirement

- The "Screens" editor is open.
- A screen object has been created in the work area of the screen.
- The created libraries are displayed.

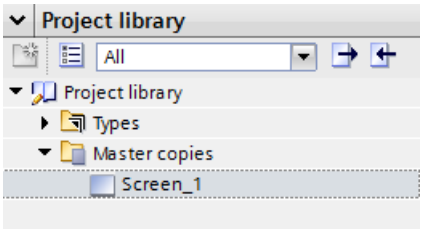
Procedure

1. Select the object in the work area of the "Screens" editor.
2. Drag-and-drop the object from the work area to the desired library.
   The mouse pointer is transformed into a crosshair with an appended object icon.



Result

The object is saved to the library for further use in multiple instances of your configuration.



Inserting a library object

Introduction

The system always assigns the inserted library object a name which consists of the name of the object type and of a consecutive number.

If the inserted object already exists, you have the option of replacing the object or of saving it under a new name.

You cannot insert library objects that are not supported by the HMI device.

Note

If you insert a screen with interconnected template from the library, the template will also be inserted. Any existing matching template is not used.

Requirement

- The "Libraries" task card is opened.
- The editor in which you want to insert the library object is open.

Procedure

1. Select a library object from the library.
2. Drag-and-drop the library object to the position in the work area where you want to insert the object.
   The library object is inserted.

Result

If the object was contained in the "Copy templates" folder, you have inserted an independent copy of the library object in the editor.

If the object was contained in the "Types" folder, you have inserted an instance of the library object in the editor.

12.11.1.7 Using types and their versions

State of type versions

Introduction

Depending on the point of use, the version of a type has different statuses.

Released version

The "released version" status is available for all types, regardless of the point of use.

If you want to edit a released version, you must first create a new test version or an "in progress" version.

Released type versions of scripts and screens can be opened and viewed at their instance.

"In progress" version

The "in progress" status is possible for the following types:

- Faceplates
- User data types (HMI UDT)
- Styles
- Style sheets

When you create a new type or a new version of a released type, the type is assigned the status "in progress".

Types with the "in progress" status can be edited in the library view without the need for a reference to an instance in the project. Upon release, the compatibility of the type is tested by a consistency check.

"In test" version

Only versions of scripts and HMI screens have the "In test" status.

When you create a new version of a type, the type is assigned the status "In test".

A version with "In test" status is linked to an instance in the project. You can set only one version to "In test" for each type at a given time.

A version in testing may only be linked to a single instance in the project. Therefore, it is not possible to copy an instance to the clipboard, to duplicate it or to create an additional type from the instance as long as it has "In test" status.

Generating a script as a type

Requirement

- A project is open.
- An HMI device has been created and opened.
- The project tree is open.
- The "Libraries" task card is opened.

Procedure

1. Open the "Scripts" editor in the project tree.
2. Create a new script.
3. Select the script in the project tree.
4. Drag-and-drop the script into a library in the "Libraries" task card. A dialog opens.
5. Enter a name.
6. Enter a comment.

Result

You have created a type version of a script in the library. The created type is stored as a released version in the library. An instance of the type is used in the project.

To change the script create a new version of the script.

Generating a screen as a type

Requirement

A project is open.

An HMI device has been created and opened.

The project tree is open.

The "Libraries" task card is opened.

Procedure

1. Open the "Screens" editor in the project tree.
2. Create a new screen.
3. Select the screen in the project tree.
4. Drag-and-drop the screen into a library in the "Libraries" task card. A dialog opens.
5. Enter a name.
6. Enter a comment.

Result

You have created a type of a screen in the library.

The created type is stored as a released version in the library. An instance of the type is used in the project.

To change the screen create a new version of the screen.

Generating a style as a type

Introduction

To define a new style, add a new type in the project library.

Adding a new style

1. Open the "Libraries" task card.
2. Select the "Add new type" command under "Types" in the shortcut menu of the project library. A dialog opens.
3. Select "HMI style".

Result

The new style is created and displayed under its selected name in the project library. The HMI style type is assigned the status "in progress" and the version 0.0.1.

Creating a style sheet as a type

Introduction

To define a new style sheet, add a new type in the project library.

Adding a new style sheet

1. Open the "Libraries" task card.
2. Select the "Add new type" command under "Types" in the shortcut menu of the project library. A dialog opens.
3. Select "HMI style sheet".
4. Assign a meaningful name to the style sheet.
5. Select the style sheet category from the list.

Result

The new style sheet is created and displayed under its selected name in the project library.

The HMI style sheet type is assigned the status "in progress" and the version 0.0.1.

Create a new version of a type

Principle

If you create a new version of a type, the point of use of the type determines the status of the newly created version.

Requirement

The "Libraries" task card is opened.

A type has been created and released.

Procedure

1. Select the released type.
2. Select "Edit type" in the shortcut menu.

Result for types of faceplates, user data, and styles

A new version of the type is created.

The version has the status "in progress". The library view opens.

Result for types of scripts and screens

A dialog opens.

After you have selected the settings in the dialog, the version is set to the status "In test". The instance used in the project is set to the status "In test". The library view opens.

12.11.2 Importing and exporting project data

12.11.2.1 Importing and exporting project data

Introduction

WinCC gives you the option of exchanging project data between the projects or copying them to external applications.

Exporting and importing between projects

You can export the following project data from a project and import them into another project.

- Recipe data records
- Alarms
- Tags
- Text lists
- Project texts

Exporting and importing reduces the workload. Instead of creating new data records, you use data already created in previous projects.

Editing the export file

The following file formats are available for export and import depending on the editor:

- *.xlsx for alarms, tags, project texts and text lists
- *.csv for recipe data records

You can edit the import file in Excel, for example.

XLSX file format

XLSX format is a file format for Excel tables based on the Open XML format. XLSX files are optimized for Microsoft Excel 2007.

You can sort the columns as required in the XLSX file.

CSV file format

CSV stands for Comma Separated Value. In this format, the columns of the table that contain the names and the value of the entry are separated by semicolons. Each table row terminates with a line break. You can also open the CSV file for editing in Excel.

Importing project data

When project data is imported, the objects in the project are created.

The syntax of the import file is checked during import. The accuracy of the values imported and dependencies between the imported values are not checked.

Any errors found in the imported data are reported when the project is compiled.

Copying in Excel format

In all spreadsheet editors you can copy the content in Excel format into the buffer of your PC. Then you insert the project data in Excel format directly into any application outside the TIA Portal. To this purpose you use the corresponding command in the shortcut menu of the work area:

- If you select the command in the shortcut menu of the line header, the complete line is copied into the buffer.
- If you select the command in the shortcut menu of a cell, only the cell content is copied into the buffer.
- If you select several lines and select the command, all marked data are always copied into the buffer.

This data exchange is only possible as an export.

12.11.2.2 Importing and exporting recipes

Exporting recipes (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

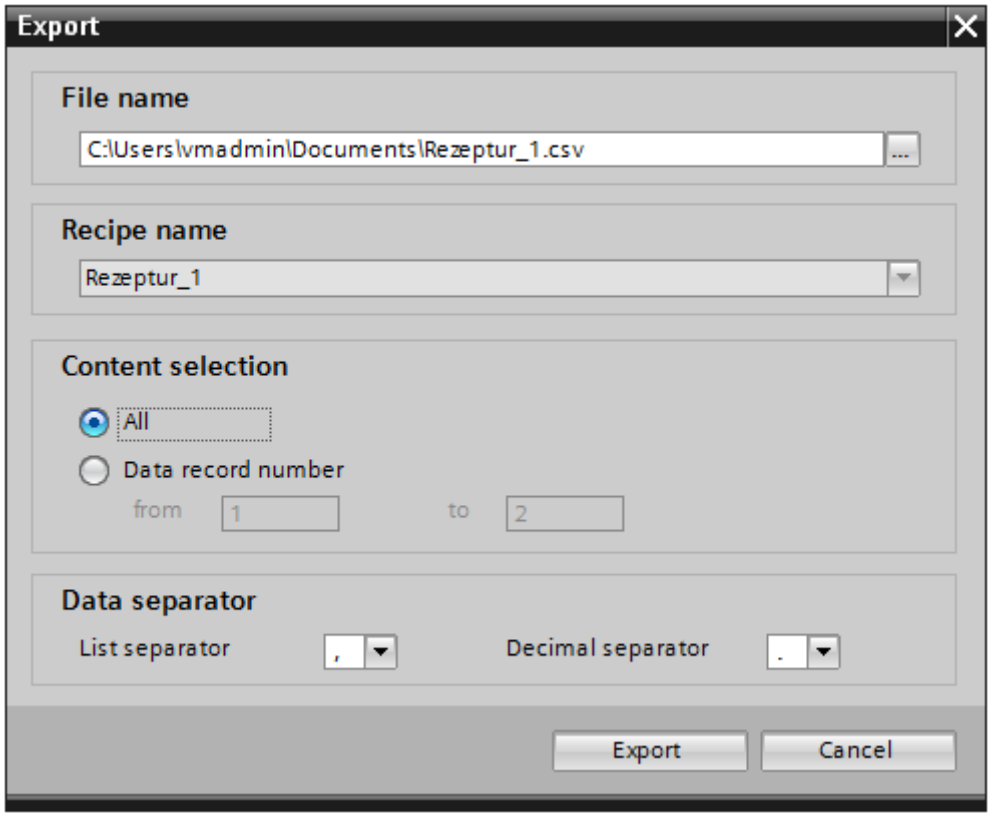
WinCC features an export function for exporting data records from recipes.

Requirements

- The WinCC project for the export is open.
- Recipes have been created in a project.
- The "Recipes" editor is open.

Exporting recipes

1. In the "Recipes" editor, select the recipe with the data records you want to export.
2. Click .
   The "Export" dialog box opens.



   The selected recipe is shown under "Recipe selection".
3. Under "Content selection", specify if all or only selected data records are to be exported.
4. Under "File selection", specify the file in which the recipe data is to be stored.
5. Specify the list separator and decimal separator under "Data separation".
6. Click "Export."
   The export will start.

Result

The exported data has been written to a CSV file. The CSV file will be stored in the specified directory.

Importing recipes (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Recipes are identified by their name. The recipe name must therefore be unique. Open the import file in a simple text editor to check that it has the correct data structure.

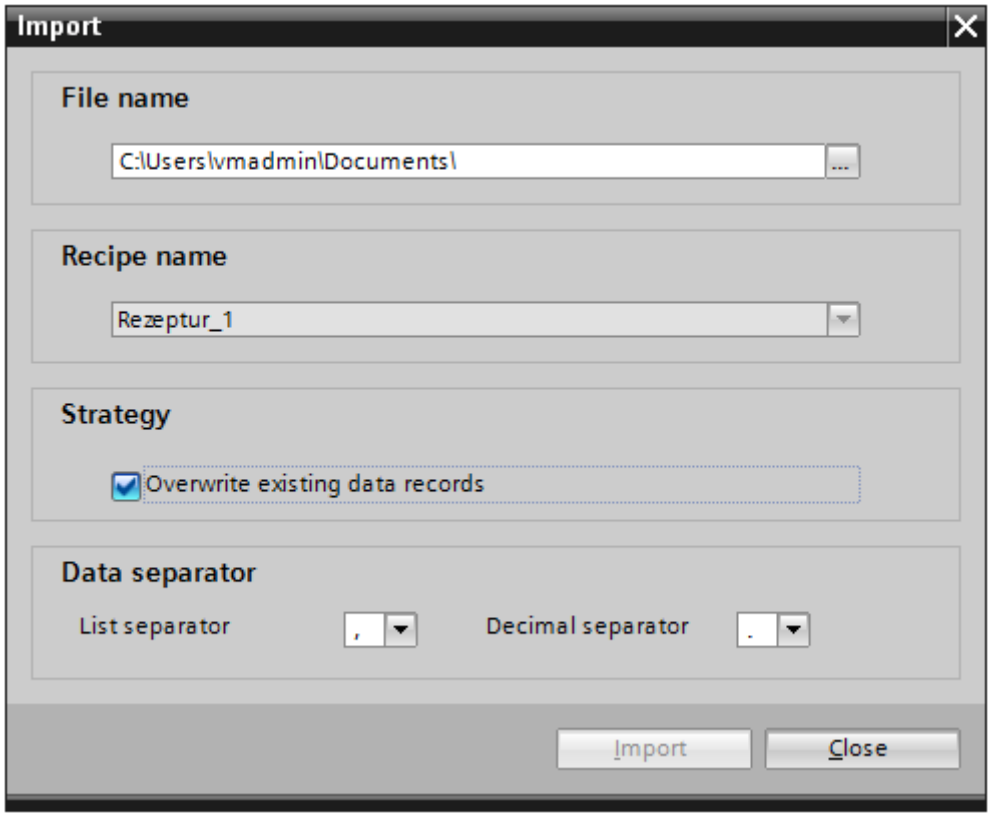
Specify whether or not existing data records should be overwritten by records with the same name during the import.

Requirements

- A CSV file containing at least one recipe has been created.
- The WinCC project for the import is open.
- The "Recipes" editor is open with at least one recipe.

Importing a recipe

1. In the "Recipes" editor, select the recipe with the data records you want to import.
2. Click .
   The "Import" dialog box opens.



   The selected recipe is shown under "Recipe selection".
3. Select the file you want to import under "File selection".
4. Under "Strategy", specify if existing data records should be overwritten by records of the same name.
5. Under "Data separation", select the list separator and the decimal separator to use in the CSV file.
6. Click "Import".
   The import will start.

Result

The data records are created in the selected recipe. Depending on the setting for "Strategy", existing data records are overwritten by records with the same name from the CSV file.

Existing data records with the same name will also be imported from the CSV file if you deactivate the "Overwrite existing data records" option.

Format of recipe data

Introduction

This section describes the required format of the file for the import of recipes. The file containing the data of the recipes must be available in "*.csv" format. :

Structure of recipe data

The structure of the import file is fixed. The following example shows the structure of a recipe containing two recipe elements, each with two data records:

```
List separator=<List separator>Decimal symbol=<Decimal
separator><List separator><Line break>
<Name of the recipe><List separator><List separator><Line break>
LANGID_<ID of the language><List separator>
<Display name, recipe element 1><List separator>
<Display name, recipe element 2><Line break>
<Number recipe><List separator>
<Recipe data record number 1><List separator>
<Recipe data record number 2><Line break>
<Tag recipe element 1><List separator>
<Recipe data record 1 value 1><List separator>
<Recipe data record 2 value 1><Line break>
<Tag recipe element 2><List separator>
<Recipe data record 1 value 2><List separator>
<Recipe data record 2 value 2><Line break>
```

ID of the language

Use the "Windows language ID" in decimal notation, e.g. "1033" for English. Additional information is available in the documentation for the Windows operating system.

12.11.2.3 Importing and exporting alarms

Exporting alarms (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

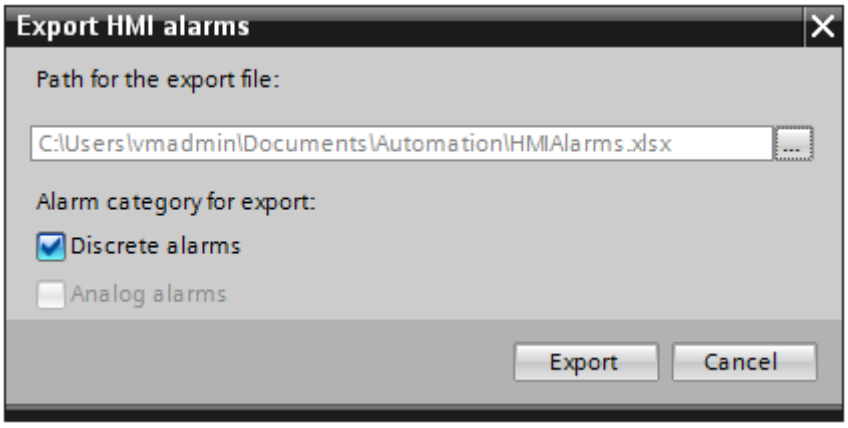
WinCC has an export function for alarms.

Requirements

- The WinCC project for the export is open.
- Alarms have been created in the project.
- The "HMI alarms" editor is open.

Exporting alarms

1. Click the button in "Discrete alarms" or "Analog alarms".
   The "Export" dialog box opens.



2. Click the "..." button and specify in which file the data are saved.
3. Specify whether you want to export "Disrete alarms" or "Analog alarms".
4. Click "Export". The export will start.

Result

The exported data has been written to an xlsx file. The xlsx file will be stored in the specified folder.

Importing alarms (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

Alarms are identified by their alarm number. The alarm numbers must be unique in the analog and discrete alarm types. Alarms with redundant alarm numbers will be overwritten. An alarm without an existing alarm number is created.

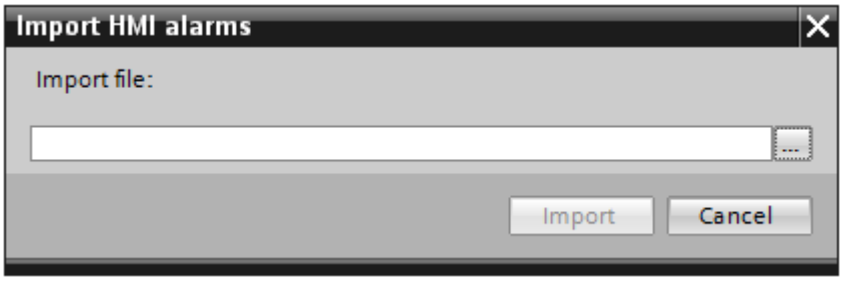
Any empty list entries for existing alarms contained in an xlsx file will be ignored for the purposes of the import. The entries of the existing alarms remain active and will not be replaced by empty ones.

Requirements

- An xlsx file with alarms has been created.
- The structure of the xlsx file meets the requirements.
- The WinCC project for the import is open.
- The "HMI alarms" editor is open.

Importing alarms

1. Click the button in "Discrete alarms" or "Analog alarms". The "Import" dialog box opens.



2. Click the "..." button and select the file that you want to import.
3. Click on the "Import" button. The import will start. A progress bar indicates the progress of the import operation.

Result

The corresponding alarms including alarm texts are created in WinCC on the basis of the import data. Alarms relating to the import operation are displayed in the output window. A log file is saved in the source directory of the import files. The log file has the same name as the respective import file but with the "*.xml" extension.

Check when importing the data whether there are any links to objects, for example dynamic parameters such as tags. These parameters must be available in all languages to be imported.

- If an object with the same name already exists, the existing object is used.
- If no object of the same name yet exists, create an object with the relevant name or create a new link.

Note

The syntax of the import file is checked during xlsx file import. The meaning of the properties or dependencies between the properties is not checked. It is possible to assign an alarm a trigger tag of the incorrect type, such as String. An error will be reported during compilation.

Format of the analog alarm data (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

This chapter describes the required format of the file for the import of analog alarms. The file containing the analog alarm data must be in "*.xlsx" format.

Structure of the alarm data

The import file in Microsoft Excel consists of a number of worksheets:

- Analog alarms(Analog alarms)
- Limits (Limits)

Each alarm is assigned a separate row in the import file. The import file with the analog alarms must be formatted as follows:

Example of the worksheet "Analog alarms"

	A	B	C	D	E
1	ID	Name	Event text [en-US], Alarm text	FieldInfo [Alarm text]	Class
2	1	Analog_alarm_1	AA1 Error-AC with maximum text length: abcdefghijklmnopqrstuvwxyz		Errors
3	2	Analog_alarm_2	AA2 Warning-AC this text should be bold		Warnings
4	3	Analog_alarm_3	AA3 SDm-AC this text should be italic		SDm
5	4	Analog_alarm_4	AA4 SDo-AC <u>this text should be underlined</u>		SDo
6	5	Analog_alarm_5	AA5 SystemAcknowledgement-AC <blink>this text should be flashing<		System_Ackn
7	25	Analog_alarm_25	Internal AA23 switchDT: Deadband mode in case of violation - value HL		AADT-Interna
8	26	Analog_alarm_26	Internal AA23 switchDT: Deadband mode in case of violation - percent		AADT-Interna
9	31	Analog_alarm_31	Internal AA23 switchDT: Low limit violation static		AADT-Interna
10	32	Analog_alarm_32	Internal AA23 switchDT: High limit violation static		AADT-Interna
11	33	Analog_alarm_33	Internal AA23 switchDT: Low limit violation dynamic		AADT-Interna
12	34	Analog_alarm_34	Internal AA23 switchDT: High limit violation dynamic		AADT-Interna
13	35	Analog_alarm_35	Internal AA23 switchDT: delay 3 seconds High limit violation		AADT-Interna
14	42	Analog_alarm_40	Internal AA23 switchDT: delay 3 seconds Low limit violation LLV		AADT-Interna
15	23	Analog_alarm_41	AA23 DT in event text: <field ref="0" /> Bool, <field ref="1" /> Byte, <fi	<ref id = 0; type = AlarmTag; T	ACAT
16	24	Analog_alarm_42	AA24 DT in event text: <field ref="0" /> Timer, <field ref="1" /> Counte	<ref id = 0; type = AlarmTag; T	ACAT

Table 12-15 Meaning of the entries

List entry	Meaning
ID	The alarm number is used to reference an alarm. The alarm number is unique. Alarms with identical alarm numbers are overwritten during import. An alarm without an existing alarm number is created.
Name	Name of the analog alarm
Event text [de-DE], Alarm text	Displays the alarm text. The field designation contains a language ID. Alarm texts must be assigned a language ID for import. An expression with a reference ID will be added to the alarm text if the text has a dynamic parameter. Example: text <field ref="0" />. Use the ID to assign dynamic parameters to alarm texts.
FeldInfo	Specifies whether the alarm text contains dynamic parameters. The settings are separated by a semicolon ";". Example of dynamic parameters: Tag: <ref id = 0; type = AlarmTag; Tag = Tag1; DisplayType = Decimal; Length = 5;> Text list: <ref id = 1; type = CommonTextList; TextList = Textlist1; Tag = tag 2; Length = 5;>

List entry	Meaning
Class	The class of an alarm determines whether or not the alarm must be acknowledged. It can also be used to determine how the alarm appears when it is displayed on the HMI device. The alarm class also determines whether and where the corresponding alarm will be logged.
Group	Indicates the allocation to an alarm group. If an alarm belongs to a group with other alarms, it can be acknowledged together with these alarms of the same group in a single operation.
Trigger tag	Specifies the tag monitored for limit value violation.
Delay time value	Specifies the delay time. The alarm is not triggered until the duration of the limit value violation equals the specified delay time.
Delay time unit	Specifies the time unit for the delay.
Report	Enables reporting of the specific alarm on a printer. True or "1" = Reporting enabled. False or "0" = Reporting disabled. Reporting must also be globally enabled in the project.
Info text [de-DE], Info text	The tooltip is an optional property of an alarm. Tooltips can contain additional information about the alarm. A tooltip will be displayed in a separate window on the HMI device when the operator presses the <HELP> key. The field designation contains a language ID.

Example of the worksheet "Limits"

	A	B	C	D	E	F	G
1	Alarm ID	Limit type	Limit value	Limit mode	Deadband mo	Deadband valu	Deadband in percent
2	1	Constant	0	Upper limit	Off	0	False
3	2	Constant	1	Upper limit	Off	0	False
4	3	Constant	2	Upper limit	Off	0	False
5	4	Constant	3	Upper limit	Off	0	False
6	5	Constant	4	Upper limit	Off	0	False
7	25	Constant	50	Upper limit	On both	5	False
8	26	Constant	50	Upper limit	On both	10	True
9	31	Constant	50	Lower limit	Off	0	False
10	32	Constant	50	Upper limit	Off	0	False
11	33	Tag	AASDTdyn	Lower limit	Off	0	False
12	34	Tag	AASDT1dyn	Upper limit	Off	0	False
13	35	Constant	50	Upper limit	Off	0	False
14	36	Constant	50	Lower limit	On both	5	False

Table 12-16 Meaning of the entries

List entry	Meaning
Alarm ID	Alarm number The alarm number is used to reference an alarm. The alarm number is unique. Alarms with identical alarm numbers are overwritten during import. An alarm without an existing alarm number is created.
Limit mode	Trigger mode Indicates the method used for monitoring the limit value.
Limit type	Specifies the limit that will be monitored. Both a tag and a constant can be used as limit value.
Limit value	Limit value Indicates the tag or constant monitored for limit violation.
Deadband mode	Hysteresis mode Specifies whether and in which cases hysteresis will be used. For "Outgoing" For "Incoming" For "Incoming" and "Outgoing"
Deadband in percent	0 = The value specified for "Hysteresis" is considered to be absolute. 1 = The value specified for "Hysteresis" is referred to as a percentage of the limit value.
Deadband mode	Hysteresis Specifies a constant as a value of the hysteresis.

Note

"No value" in the table

Entries in the table which have the value "No value" delete the corresponding values in an existing alarm of the same name.

Format of the discrete alarm data (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

This chapter describes the required format of the file for the import of discrete alarms. The file containing the discrete alarm data must be in "*.xlsx" format.

Structure of the alarm data

The import file in Microsoft Excel consists of the worksheets "Discrete alarms" (discrete alarms). Each alarm is assigned a separate row in the import file. Structure of the import file containing the discrete alarms:

Example of the worksheet "Discrete alarms"

	A	B	C	D	E	F
1	ID	Name	Event text [en-US], Alarm text	FieldInfo [Alarm text]	Class	Trigger t
2	1	Discrete_alarm_1	DA1 Error-AC with maximum text length: abcdefghijklmnopqrstuvwxyz äöü\na		Errors	HMI_AC
3	2	Discrete_alarm_2	DA2 Warning-AC this text should be bold		Warnings	HMI_AC
4	3	Discrete_alarm_3	DA3 SDm-AC this text should be italic		SDm	HMI_AC
5	4	Discrete_alarm_4	DA4 SDo-AC <u>this text should be underlined</u>		SDo	HMI_AC
6	5	Discrete_alarm_5	DA5 SystemAcknowledgement-AC <blink>this text should be flashing</blink		System_Ackn	HMI_AC
7	6	Discrete_alarm_6	DA6 SystemNoAcknowledgement-AC mixed test: Bold, Italic, <l		System_No_A	HMI_AC
8	7	Discrete_alarm_7	DA7 DT in event text: <field ref="0" /> Integer, <field ref="1" /> Real, <field re	<ref id = 0; type = AlarmTag; Tag = Pl	ACAT	HMI_Tri
9	8	Discrete_alarm_8	DA8 DT in event text: <field ref="0" /> S5Time, <field ref="1" /> Timer, <field r	<ref id = 0; type = AlarmTag; Tag = Pl	ACAT	HMI_Tri
10	11	Discrete_alarm_9	DA11 DT in event text: <field ref="0" /> Int, <field ref="1" /> Real, <field ref="2	<ref id = 0; type = AlarmTag; Tag = In	ACAT	HMI_Tri
11	12	Discrete_alarm_10	DA12 DT in event text: <field ref="0" /> UDInt, <field ref="1" /> UInt;	<ref id = 0; type = AlarmTag; Tag = U	ACAT	HMI_Tri
12	13	Discrete_alarm_11	DA13 Textformat: Integer: <field ref="0" /> decimal, <field ref="1" /> binary, <	<ref id = 0; type = AlarmTag; Tag = H	ACAT	HMI_TF

Table 12-17 Meaning of the entries

List entry	Meaning
ID	The alarm number is used to reference an alarm. The alarm number is unique. Alarms with identical alarm numbers are overwritten during import. An alarm without an existing alarm number is created.
Name	Name of the analog alarm
Event text [de-DE], Alarm text	Displays the alarm text. The field designation contains a language ID. For import, a language ID must be assigned to alarm text. An expression with a reference ID will be added to the alarm text if the text has a dynamic parameter. Example: text <field ref="0" />. Use the ID to assign dynamic parameters to alarm texts.
FeldInfo	Specifies whether the alarm text contains dynamic parameters. The settings are separated by a semicolon ";". Example of dynamic parameters: Tag: <ref id = 0; type = AlarmTag; Tag = Tag1; DisplayType = Decimal; Length = 5;> Text list: <ref id = 1; type = CommonTextList; TextList = Textlist1; Tag = tag 2; Length = 5;>
Class	The class of an alarm determines whether or not the alarm must be acknowledged. It can also be used to determine how the alarm appears when it is displayed on the HMI device. The alarm class also determines whether and where the corresponding alarm will be logged.
Group	Indicates the allocation to an alarm group. If an alarm belongs to a group with other alarms, it can be acknowledged together with these alarms of the same group in a single operation.
Trigger tag	Specifies the tag containing the bit that triggers the alarm.
Trigger bit	Specifies the number of the bit that triggers the alarm.

List entry	Meaning
Acknowledge tag	Specifies the tag containing the bit that is set by the operator upon acknowledgment. Only available if the selected alarm class requires alarm acknowledgment.
Acknowledgment bit	Specifies the number of the bit that is set when the operator acknowledges the alarm.
PLC acknowledgement tag	Specifies the tag containing the bit that acknowledges the alarm of the control program. Only available if the selected alarm class requires alarm acknowledgment.
PLC acknowledgment bit	Specifies the number of the bit that acknowledges the alarm of the control program.
Delay time value	Specifies the delay time. The alarm is not triggered until the duration of the limit value violation equals the specified delay time.
Delay time unit	Specifies the time unit for the delay.
Report	Enables reporting of the specific alarm on a printer. True or "1" = Reporting enabled. False or "0" = Reporting disabled. Reporting must also be globally enabled in the project.
Info text [de-DE], Info text	The tooltip is an optional property of an alarm. Tooltips can contain additional information about the alarm. A tooltip will be displayed in a separate window on the HMI device when the operator presses the <HELP> key. The field designation contains a language ID.

Note

"No value" in the table

Entries in the table which have the value "No value" delete the corresponding values in an existing alarm of the same name.

12.11.2.4 Importing and exporting tags

Exporting tags (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

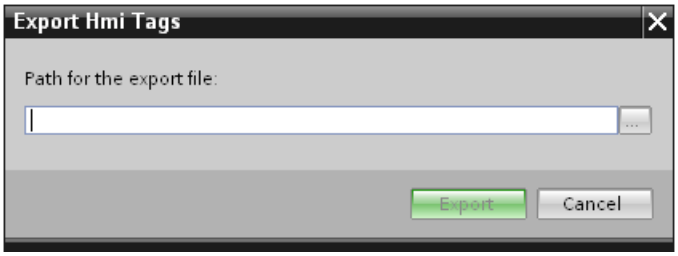
WinCC has an export function for tags.

Requirements

- The WinCC project for the export is open.
- Tags have been created in the project.
- The "HMI tags" editor is open.

Exporting tags

1. Click on the button in the "HMI Tags" tab.
   The "Export" dialog box opens.



2. Click the "..." button and specify in which file the data are saved.
3. Click "Export". The export will start.

Note

The version number of the xlsx file with the exported tags depends on the TIA Portal version. If the xlsx file was exported from a project in WinCC V13 SP1, it has the version number 1.2. If the xlsx file was exported from a project prior to WinCC V13 SP1, it has the version number 1.1.

Note

It is not possible to export HMI tags of the data type "UDT" which contain structured elements via Excel for subsequent editing.

After export, only the higher-level HMI tag appears in Excel. Its lower-level elements cannot be edited.

Result

The exported data has been written to an xlsx file. The xlsx file will be stored in the specified folder.

Importing tags (WinCC V13 SP1 or higher) (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

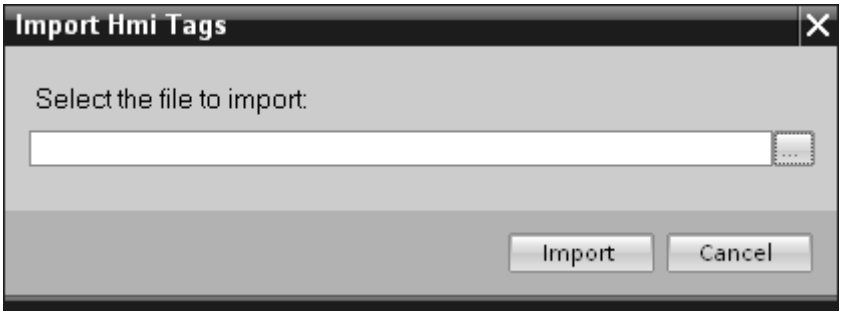
Tags are identified by the tag name. An existing tag is overwritten with the data from the xlsx file if the tag name already exists in the project. A new tag is created if the tag does not yet exist.

Requirements

- An xlsx file with tags has been created.
- The structure of the xlsx file meets the requirements.
- The WinCC project for the import is open.

Importing tags

1. Click "HMI tags" in the project navigation.
2. Double-click "Show all tags". The "HMI tags" editor opens.
3. Click the button. The "Import" dialog box opens.



4. Click the "..." button and select the file that you want to import.
5. Click on the "Import" button. The import will start.

Result

The relevant tags have been created in WinCC. Alarms relating to the import operation are displayed in the output window. A log file is saved in the source directory of the import files. The log file has the same name as the respective import file but with the "*.xml" extension.

Check when importing the data whether there are any links to objects, for example, dynamic parameters such as tags.

- If an object with the same name already exists, the existing object is used.
- If no object of the same name yet exists, create an object with the relevant name or create a new link.

Note

The syntax of the import file is checked during xlsx file import. The meaning of the properties or dependencies between the properties is not checked. It is possible to assign a tag a trigger tag of the wrong type, for example, string. An error will be reported during compilation.

Importing array elements and structure elements

Basics

Array tags and structure tags are always imported in full with all elements. The elements of the array tags and structure tags are not filled further during import.

A new tag is created if the name of the structured data type corresponds to the name of an array element or structure element in the import file.

Example:

The import file contains an array tag called "Otto" with 10 array elements. The array elements are then called Otto[1], Otto[2], for example.

If the import file contains a tag named "Otto[1]", the first element of the array tag will not be filled. Instead, a new tag will be created in the Engineering System.

Format of the tag data (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

This section describes the format required for the file with tag data used for imports. The tag data file must be in "*.xlsx" format.

Tag data structure

The import file in Microsoft Excel consists of a number of worksheets:

- HMI Tags (HMI tags)
- Multiplexing (multiplex tags)

Each tag is on a separate line in the import file. The import file with the tag data must have the following format:

Example of the worksheet "HMI Tags"

	A	B	C	D	E	F	G	H
1	Name	Path	Connection	PLC tag	DataType	Length	Address	Access
2	HMI_Int	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
3	Mux_Tag_1	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
4	Mux_Tag_2	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
5	Mux_11	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
6	Mux_21	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
7	Mux_13	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
8	Mux_12	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
9	Mux_23	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
10	Mux_22	Internal Tags	<No Value>	<No Value>	Int	2	<No Value>	<No Va
11	Mux_Tag_1_Index	Internal Tags	<No Value>	<No Value>	UInt	2	<No Value>	<No Va
12	Mux_Tag_12	Internal Tags	<No Value>	<No Value>	USInt	1	<No Value>	<No Va
13	Mux_Tag_11	Internal Tags	<No Value>	<No Value>	USInt	1	<No Value>	<No Va
14	Mux_Tag_13	Internal Tags	<No Value>	<No Value>	USInt	1	<No Value>	<No Va
15	HMI_UDInt	Internal Tags	<No Value>	<No Value>	UDInt	4	<No Value>	<No Va
16	Gauge_Process	Default tag table	<No Value>	<No Value>	Int	2	<No Value>	<No Va
17	Button_Tag_4	Default tag table	<No Value>	<No Value>	Int	2	<No Value>	<No Va
18	HMI_USInt	Internal Tags	<No Value>	<No Value>	USInt	1	<No Value>	<No Va
19	Data_block_2_PLC_DateTime_2	Default tag table	HMI_connection_1	Data_block_2.PLC_[	Date_And_Time	8	%DB28.DBX598.(	<absol

Table 12-18 Meaning of the entries

List entry	Meaning
Name	Indicates the configured name of an HMI tag.
Path	Specifies which folders in the project tree contain the tag. The folder structure is represented by "\" : "FolderName1\FolderName2\TagName".
PLC Tag	Indicates whether the tag is linked to a PLC tag.
Connection	Indicates the name of the connection to the PLC.
Data type	Specifies the data type of a tag. The data types allowed depend on the communication driver being used. See the "Communication" section of the documentation for additional information on the data types permitted for the various communication drivers.
Length	Specifies the length of the tag. This entry is only useful for data types with a dynamic length such as strings; it is left empty for all other data types.
Address	Specifies the tag address in the PLC. The tag address must exactly match the one used in WinCC, for example, "%DB1.DBW0". The tag address is empty for internal tags.
Multiplexing	Specifies whether multiplexing is used.
Index tag	Shows the name of the index tag for multiplexing. In Runtime, the system first reads the value of the index tag. It then accesses the tag in the corresponding place in the tag list.
StartValue	Specifies the start value of a tag.
ID tag	The update ID updates the value of a tag with the aid of a function or a PLC job. The update ID must be unique within an HMI device.
Coding	Shows the coding method.

List entry	Meaning
DiplayName [de_DE]	Shows the display name of an HMI tag. The field designation contains a language ID. The field designation contains a language ID. Display names must be assigned a language ID for import. Texts are imported to the corresponding project language.
Acquisition mode	Specifies the tag acquisition mode.
Acquisition cycle	Specifies the tag acquisition cycle. The acquisition cycle must correspond exactly to the one used in WinCC. The value is not language-dependent and should therefore be the same in every language. The default value is "1 s". The acquisition cycle is undefined if the tag acquisition mode is "on demand". User-defined acquisition cycles must be created beforehand as the file will otherwise not be imported.
High High Limit type	Indicates whether the limit value "High high" is monitored by a constant, a tag or not at all.
High High Limit	Displays the limit value "High High".
High Limit type	Indicates whether the limit value "High" is monitored by a constant, a tag or not at all.
High Limit	Displays the limit value "High".
Low Limit type	Indicates whether the limit value "Low" is monitored by a constant, a tag or not at all.
Low Limit	Displays the limit value "Low".
Low Low Limit type	Indicates whether the limit value "Low Low" is monitored by a constant, a tag or not at all.
Low Low Limit	Displays the limit value "Low Low".
Linear scaling	Indicates whether linear scaling is enabled. This entry can only be used for external tags.
End value PLC	Specifies the end value of the PLC tag.
Start value PLC	Specifies the start value of the PLC tag.
End value HMI	Specifies the end value of the HMI tag.
Start value HMI	Specifies the start value of the HMI tag.

Example of the worksheet "Multiplexing"

	A	B	C
1	HMI Tag name	Multiplex Tag	Index
2	Mux_Tag_1	Mux_11	0
3	Mux_Tag_1	Mux_12	1
4	Mux_Tag_1	Mux_13	2
5	Mux_Tag_2	Mux_21	0
6	Mux_Tag_2	Mux_22	1
7	Mux_Tag_2	Mux_23	2
8	Mux_Tag_12	HMI_Array_Mux2	-1
9	Mux_Tag_11	HMI_Array_Mux1	-1
10	Mux_Tag_13	HMI_Array_Mux3	-1

Table 12-19 Meaning of the entries

List entry	Meaning
Name	Indicates the configured name of an HMI tag which uses indirect addressing. The HMI tag must be available in the "HMI Tags" worksheet.
Index	Shows the value which governs which tag is selected.
Multiplex Tag	Displays the tag from the tag list corresponding to the index value.

Note

"No value" in the table

Entries in the table which have the value "No value" delete the corresponding values in an existing tag of the same name.

12.11.2.5 Importing and exporting text lists

Exporting text lists (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

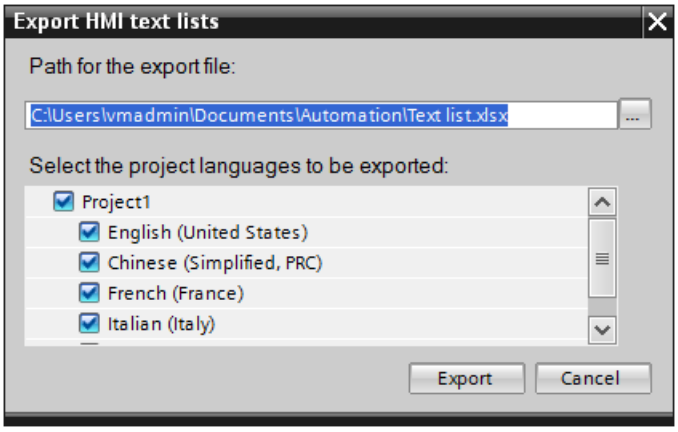
WinCC has an export function for text lists.

Requirements

- The WinCC project for the export is open.
- Text lists have been created in the project.
- The "Text & graphics lists" editor is open.

Exporting text lists

1. Click on the button in the "TextLists" tab.
   The "Export" dialog box opens.



2. Click the "..." button and specify in which file the data are saved.
3. By default, texts are exported in all defined project languages.
   If you do not want to export specific languages, disable the unnecessary languages in the dialog.
4. Click "Export". The export will start.

Result

The exported data has been written to an xlsx file. The xlsx file will be stored in the specified folder.

Importing text lists (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

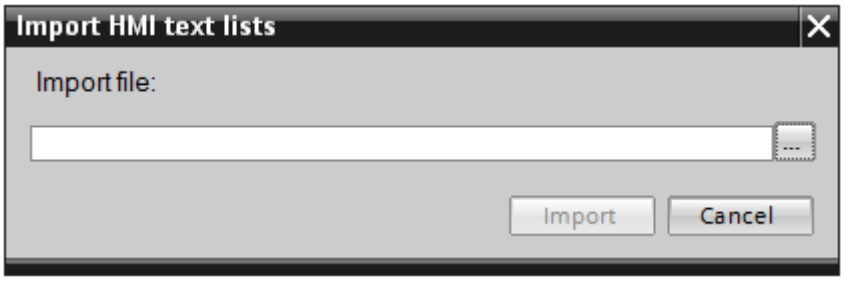
You then import text lists from an xlsx file to WinCC.

Requirements

- An xlsx file with text lists has been created.
- The structure of the xlsx file meets the requirements.
- The WinCC project for the import is open.
- The "Text & graphics lists" editor is open.

Importing text lists

1. Click on the button in the "Text lists" tab.
   The "Import" dialog box opens.



2. Select the file you want to import under "File selection".
3. Click on the "Import" button. The import will start.

Result

You have now imported the text lists. The relevant text lists have been created in WinCC. Alarms relating to the import operation are displayed in the output window. A log file is saved in the source directory of the import files. The log file has the same name as the respective import file but with the "*.xml" extension.

Check when importing the data whether there are any links to objects, for example, dynamic parameters such as tags.

- If an object with the same name already exists, the existing object is used.
- If no object of the same name yet exists, create an object with the relevant name or create a new link.

Format of text list data (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

This section describes the format required for the file with the text lists used for imports. The text list data file must be in "*.xlsx" format.

Tag data structure

The import file in Microsoft Excel consists of two worksheets:

- TextList (Text lists)
- TextListEntry (Text list entry)

Each text list is assigned a separate line in the import file. The import file containing the data must be structured as follows:

Example of the worksheet "TextList"

	A	B	C
1	Name	ListRange	Comment
2	TLValue/Range	Decimal	
3	TLBit	Bit	
4	TLBitnumber	Binary	
5	TLall_1	Decimal	
6	TLall_2	Decimal	

Table 12-20 Meaning of the entries

List entry	Meaning
Name	Shows the name of the text list.
ListRange	Shows the text list range: Number = Bit number (0-31) Range = value/range (...-...) Bit = Bit (0;1)
Comment	Any comments on the text list. You can use up to 500 characters

Example of the worksheet "TextListEntry"

	A	B	C	D	E	F
1	Name	Parent	DefaultEntry	Value	Text	FieldInfos
2	Text_list_entry_1	TLValue/Range	TRUE	0 - 1	Default entry TLValue/Range ->	
3	Text_list_entry_2	TLValue/Range		2 - 3	TLValue/Range = 2-3 ->	
4	Text_list_entry_3	TLValue/Range		1	TLValue Range - single value = 1 ->	
5	Text_list_entry_1	TLBit		0	TLBit = 0 ->	
6	Text_list_entry_2	TLBit		1	TLBit = 1 ->	
7	Text_list_entry_1	TLBitnumber	TRUE	0	Default entry TLBitnumber; ->	
8	Text_list_entry_2	TLBitnumber		0	TLBitnumber - Bitnumber 0 is set ->	
9	Text_list_entry_3	TLBitnumber		1	TLBitnumber - Bitnumber 1 is set ->	
10	Text_list_entry_4	TLBitnumber		2	TLBitnumber - Bitnumber 2 is set ->	
11	Text_list_entry_5	TLBitnumber		3	TLBitnumber - Bitnumber 3 is set ->	
12	Text_list_entry_1	TL1	TRUE	0 - 1	Default entry TL1	
13	Text_list_entry_2	TL1		1 - 3	TL1 Value between 1 - 3 ->	
14	Text_list_entry_3	TL1		4 - 6	TL1 Value between 4 - 6 ->	
15	Text_list_entry_4	TL1		7	TL1 Single value = 7 ->	
16	Text_list_entry_1	TL2	TRUE	0 - 1	<field ref="0" /> ->	<ref id = 0; type = CommonTextList; TextList = TL1; Tag = HMI_TL1control; Length = 1
17	Text_list_entry_2	TL2		1	TL2 Single value = 1 ->	
18	Text_list_entry_3	TL2		2 - 3	TL2 Range between 2 - 3 ->	
19	Text_list_entry_1	TLMultilined	TRUE	0 - 1	Default entry TLMultilined; last row	
20	Text_list_entry_2	TLMultilined		0 - 3	TLMultilined Value between 0-3\nwith test of"\n"	
21	Text_list_entry_1	TLall_1	TRUE	0 - 1	Default entry TLall_1	
22	Text_list_entry_1	TLall_2	TRUE	0 - 1	Default entry TLall_2	

Table 12-21 Meaning of the entries

List entry	Meaning
Name	Shows the name of the text list entry.
Parent	Specifies the name of the corresponding text list.
DefaultEntry	Indicates whether the text list entry is a default entry. The default entry is always displayed when the tag has an undefined value.
Value	Specifies the tag integer values or value ranges which are assigned to the text entries in the text list.
Text	Shows the text list entry. The field designation contains a language ID. Text list entries must be assigned a language ID for import. An expression with a reference ID will be added to the text if the text list entry has a dynamic parameter. Example: text <field ref="0" />. Use the ID to assign the dynamic parameter to a text list entry.
FeldInfo	Specifies whether the text list contains dynamic parameters. The settings are separated by a semicolon ";". Example of dynamic parameters: Tag: <ref id = 0; type = CommonTagDisplayFormat; Tag = tag 1; DisplayType = Decimal; DisplayFormat = 9;> Text list: <ref id = 1; type = CommonTextList; TextList = Textliste_1; Tag = tag 2; Length = 5;> PLC tag: <ref id = 0; type = CommonControlTagDisplayFormat; DisplayType = Decimal; DisplayFormat = 9;>

12.11.2.6 Importing and exporting project texts

Exporting project texts

Project texts are exported for translation. Texts are exported to Office Open XML files ending in ".xlsx". These files can be edited in Microsoft Excel, for example.

You can exchange the file with the translators and import it back to the project as soon as it has been translated.

Requirements

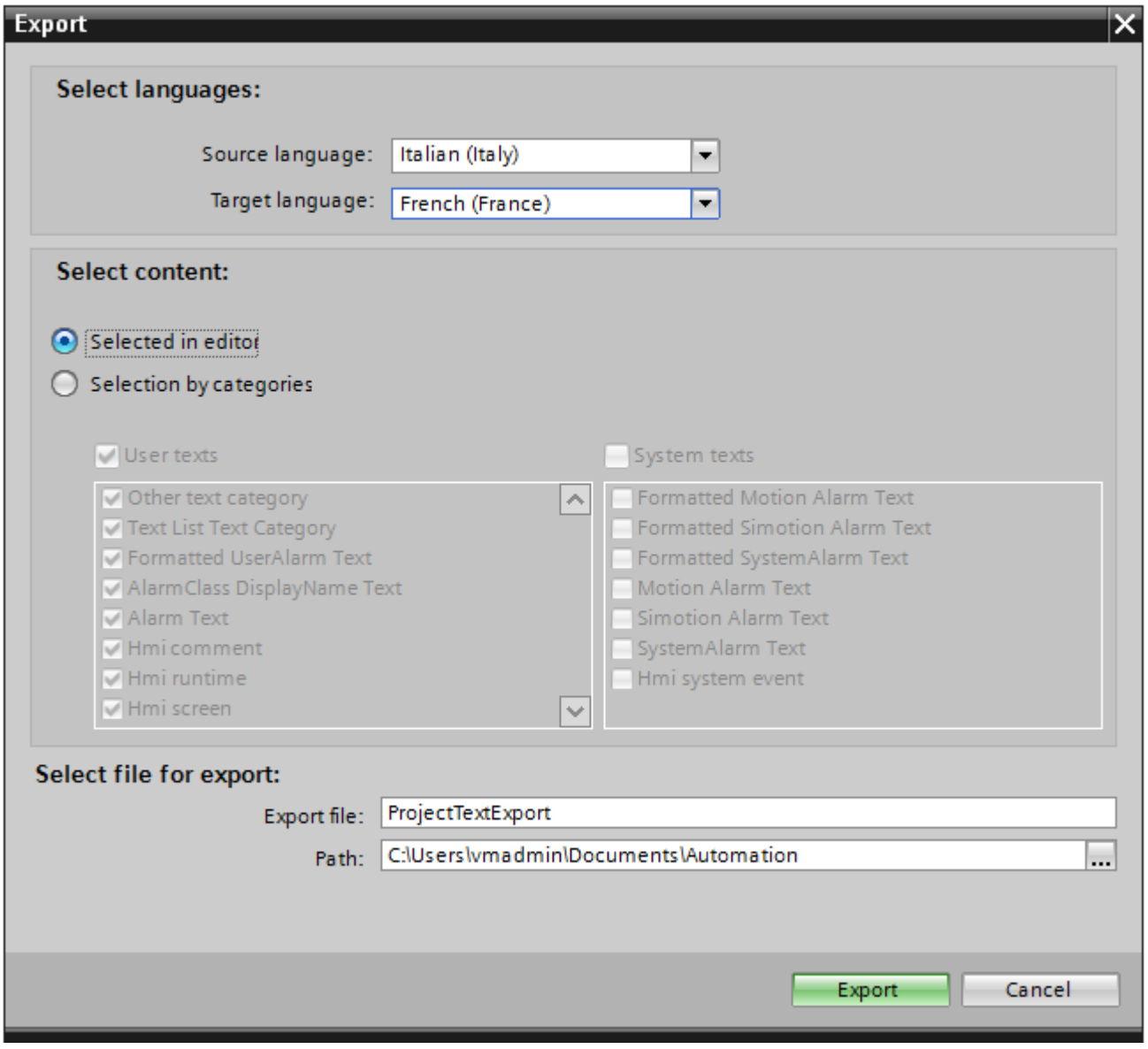
- At least two languages have been enabled in the "Project languages" editor, for example, Italian and French.

Exporting project texts

To export individual project texts, proceed as follows:

1. Click on the arrow to the left of "Languages & resources" in the project tree.
   The child elements are displayed.
2. Double-click on "Project texts". The "Project texts" editor will open.
3. Select the texts you want to export.

4. Click on the button. The "Export" dialog opens.



5. From the "Source language" drop-down list, select the language from which you wish to translate, for example Italian.
6. From the "Target language" drop-down list, select the language into which the texts are to be translated, for example, French.
7. Enter a file path and a file name for the export file in the "Export file" input field.
8. Click "Export".

Result

The texts selected in the "Project texts" editor are written to an xlsx file. The xlsx file will be stored in the specified folder.

You can alternatively select and export all project texts from categories. Select "User texts" or "System texts" in the "Export" dialog in line with the type of texts you wish to export. In this case, export can additionally be limited by categories.

Note

Project texts in library objects cannot be exported.

Importing project texts

Edit the xlsx file or send it to a translator. Import the texts once they have been translated. The foreign languages will be imported to the relevant object in the project.

Note

In WinCC, you only import the previously exported project texts into the same project. Importing into a different project is not supported.

Requirements

- At least two languages have been enabled in the "Project languages" editor, for example, Italian and French.

Importing project texts

To import a project text file, proceed as follows:

1. Click on the arrow to the left of "Languages & resources" in the project tree.
   The lower-level elements will be displayed.
2. Double-click on "Project texts". The "Project texts" editor will open.
3. Click on the  button. The "Import" dialog opens.
4. Select the path and file name of the import file from the "Import file" field.
5. Activate the "Import source language" check box if you have made changes to the source language in the export file and would like to overwrite the entries in the project with the changes.
6. Click on "Import".

Result

You have imported the project texts.

12.11.3 Using cross-references

12.11.3.1 General information about cross references (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The cross-reference list provides an overview of the use of objects and devices within the project and project library. The cross-reference list can be used to show the relationships and the dependencies of objects among themselves.

Uses of cross-reference list

The cross-reference list offers you the following advantages:

- When creating and changing a program, you retain an overview of the devices, objects, tags, alarms, scripts, etc. that you have used.
- From the cross-references, you can jump directly to the object location of use.
- You can see whether the object uses other objects or is used by itself.
- You can learn the following when debugging:
  - Which objects are used in which screen and on which devices.
  - The alarms and recipes shown in a specific display.
  - The tags used in a specific alarm or object.
- As part of the project documentation, the cross-references provide a comprehensive overview of all object, alarms, recipes, tags and screens.

For additional information on working with cross-reference lists, please refer to "Displaying cross-reference lists".

12.11.3.2 Special considerations when working with the cross-reference list

Working with cross-references in WinCC

In WinCC, there are a number of special features that you must take into account when working with cross-references.

For additional information on working with cross-reference lists, please refer to "Displaying cross-reference lists".

"Change cross-reference" dialog

You can change the references of tags in the properties of screen objects using the "Change object references" dialog.

Adding a new object to the cross-reference list

The auto-complete function only offers HMI tags for selection. Array elements of an HMI tag are not displayed in auto-complete. You can only add array elements by entering the name of an array element.

Displaying cross-references in the Inspector window

If you select an object in the screen and an HMI tag is being used as the process tag at the object, the object and the linked HMI tag are displayed in the cross-references. All locations of use of the object and the HMI tags are also listed.

If the HMI tag is interconnected with a PLC tag or a DB tag, the locations of use of the interconnected PLC tag or DB tag are displayed.

Shortcut menu in the cross-reference list

The shortcut menu is object-based. The shortcut menu is available for a line when the "Object" column actually contains an object.

12.11.3.3 Rewiring tags in the screens (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You can change the references of tags in the properties of screen objects using the "Change object references" dialog.

This function lets you to replace a large number of tags in various screen objects. Replacing references is useful, for example, if you have created a screen by copying and pasting and need to adapt its references.

You search for and replace individual or multiple characters in the tag names.

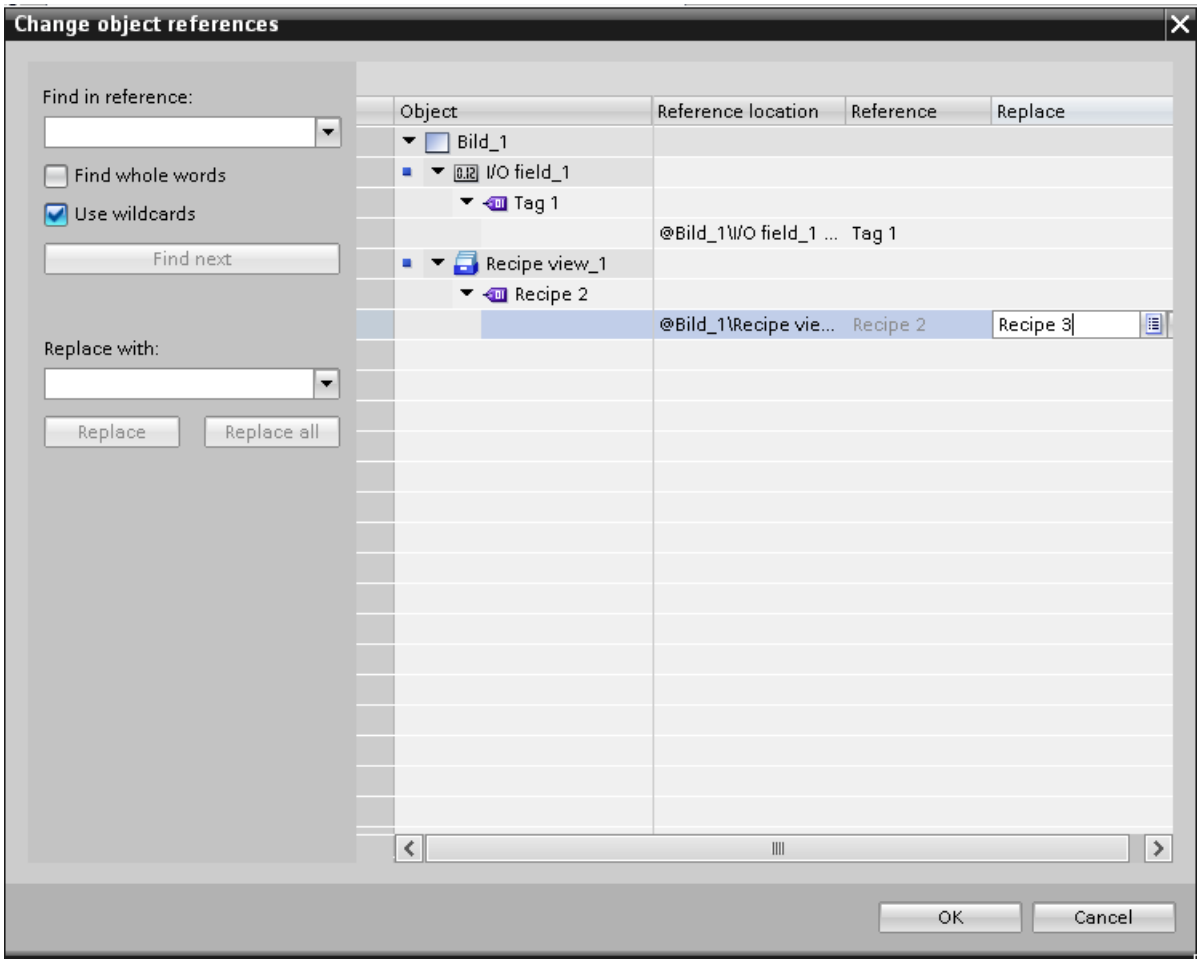
You can open the "Change object references" dialog with the command in the shortcut menu. You can open the dialog for one or more screen objects or a screen.

The "Change object references" dialog is available for the following screens.

- Overview screen (permanent area)
- Templates
- Pop-up screens
- Slide-in screens
- Global screen

How to replace a tag

1. Select an object in the screen that contains a tag that you want to change.
2. Select the "Change object references" command in the shortcut menu or in the "Edit" menu.
   The "Change object references" dialog opens. You can see the selected screen objects and the tags used in the "Object" column.



3. Enter the name or part of the name you want to find in the "Find in reference" input box.
4. Start the search with the "Find next" button. If a reference is found, it is highlighted in the "Reference" column.
   Another click on the "Find next" button continues the search process until the list has been completely searched.
   After this, the search process starts again from the top of the list.
5. Enter the name or part of the name you want to replace for the found name in the "Replace with" input box.
6. Click the "Replace" button to replace the selected tag.
   All references found are replaced by clicking the "Replace all" button. In this way, you can replace a large number of references with another
   reference.

7. To select a tag from the list of all tags created in the project, click on the button in the "Replace" column.
8. To select a tag from the object list or to create a new tag, click on the button in the "Replace" column.
9. Confirm your entry with "OK".

The rewiring of the tags is applied in the project.

Alternatively, you can enter a tag to replace the existing tag directly in the "Replace" column. You are supported by the auto-complete function in this process.

Note

When you search for a tag in the "Replace" column, you are supported by the auto-completion. After entering the first letter of the tag name in the "Replace", the auto-completion list contains the tags whose names contain the letters. Depending on the context, the auto-completion list only contains the tags that are permitted for the intended purpose.

Note

The tags shown in the column are displayed as read-only if the reference object is read-only, for example, a tag in a screen that has been stored in the library.

Note

It is possible to manually create a tag that does not yet exist. You create this tag at a later time.

Additional options for searching

You can narrow down your search by enabling the following options:

- Use wildcards
  Enter * for any number of characters. Example: You want to search for all tags starting with "HMI". Enter "HMI*" in the search key box.
  Enter ? for an individual character.
- Find whole words
  To search only for full names, enable the "Find whole words" option.

Replacing arrays and user-defined tags

Tags with names containing the special characters . [ ] are shown in quotation marks.

Array type tags and tags of user-defined data types (UDT) contain special characters to separate tags from their elements. However, they are not shown in quotation marks. To replace an array or a UDT with an HMI tag with special characters, enter the name of the HMI tag in quotation marks in the "Replace" input box. If you type the name without quotation marks, a reference to a non-existent array or UDT is created instead of a reference to the desired HMI tag with special characters.

Correcting entries

If an incorrect entry is made in the "Replace with" input box, the existing references can be replaced by references to non-existent tags.

If you detect an incorrect entry before confirming with "OK", you can correct the error. To do this, select the "Replace" column and press the "Remove" button. Any incorrect entries in the "Reference" column are deleted.

If you confirm the dialog with "OK", the affected properties of the screen objects are marked in red. Non-existent tags are highlighted in red in the "Change cross references" dialog.

If the new tag does not yet exist, you must subsequently create the tag.

12.11.3.4 Textual cross-references: Overview (Basic Panels)

Introduction

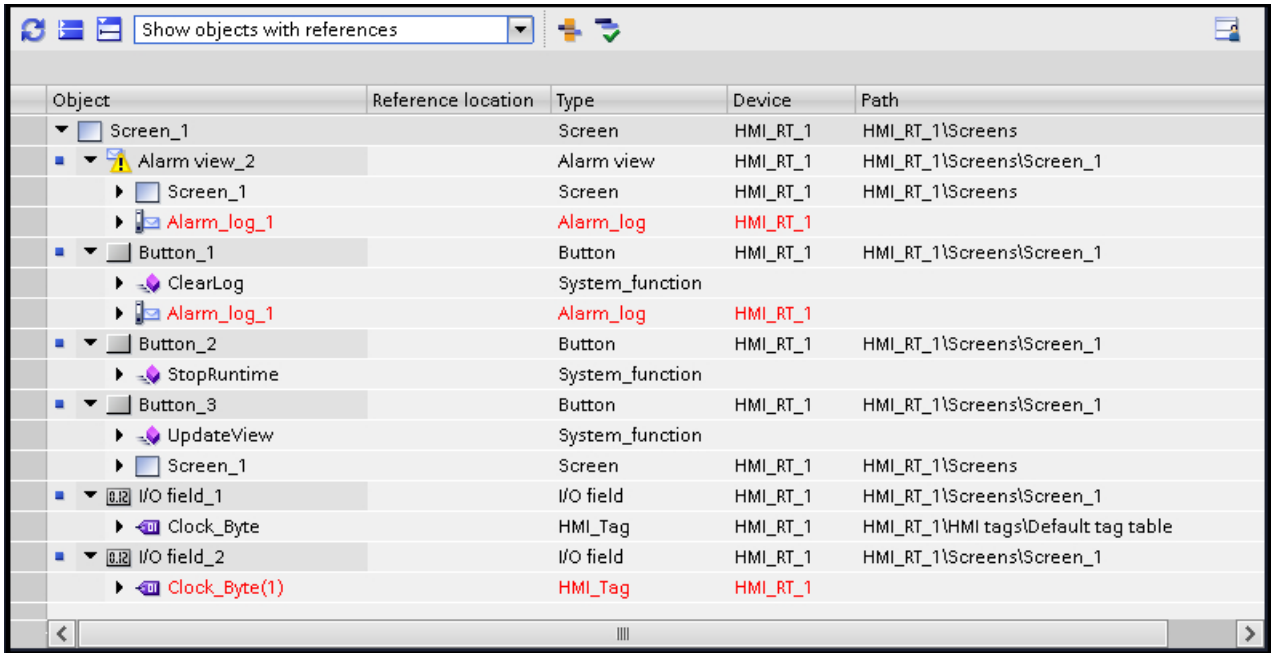
Textual cross-references are cross-references that are stored as pure text. You cannot navigate to a different object with the help of textual cross-references. Such cross-references are displayed in red in the cross-reference list.

Textual cross-references are created as follows:

- You create a textual cross-reference for a non-existent object manually by entering the object name under Property.
- The object used has already been deleted, and the connection to it is shown as textual cross-reference.

Principle

When references and child objects of a source object are deleted, they are converted into textual cross-references. A textual cross-reference is not stored when the source object itself is deleted.



Note

Almost all objects, with the exception of data logs, support textual cross-references.

Note

Textual cross-references cannot be deleted.

You can also create a textual cross-reference manually to establish a connection to a non-existent object.

12.11.3.5 Invalid cross-references (Basic Panels)

Introduction

A cross-reference can become invalid as a result of certain changes to the configuration. Invalid cross-references are shown grayed out in the cross-reference list and in the Inspector window. A question mark is displayed beside the cross-reference text. Clicking on the question mark opens the online help.

Invalid cross-references do not have any negative implications for your project.

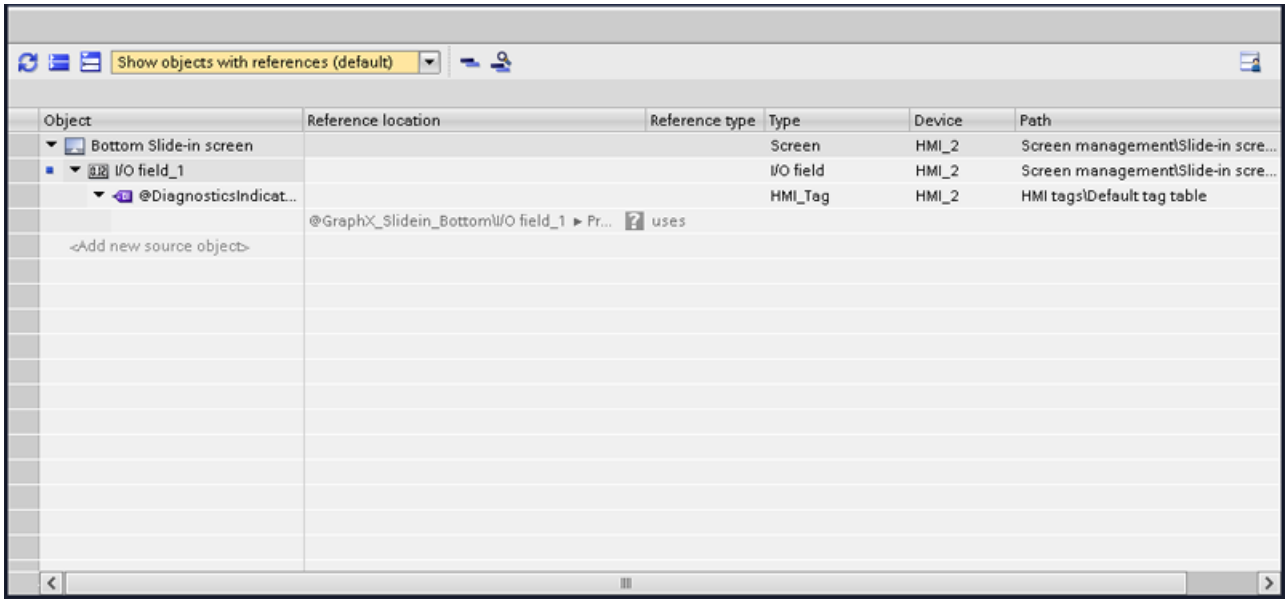
Basics

A cross-reference can become invalid for the following reasons:

- A change of device type
- Copying a cross-reference from another HMI device to an HMI device that does not support the object in question or object property.
- A change to an object property or the assignment of invalid values

The following objects can be invalid in the cross-reference list:

- Source and target object
- Subordinate objects and their higher-level objects
- Location of use



If an object or location of use is invalid, you cannot navigate to that object or location of use. You cannot change the properties of an invalid location of use. For invalid objects, you can access the commands "Delete" and "Properties" over the shortcut menu. No other shortcut menu commands are available. No shortcut menu is available for invalid locations of use.

Note

Textual cross-reference invalid

An object can be an invalid object and a textual cross-reference in the cross-reference list at the same time. In such a case, the invalid cross-reference is shown in red like a textual cross-reference.

12.11.4 Managing languages

12.11.4.1 Languages in WinCC

User interface language and project languages

A distinction is drawn between two different language levels in WinCC:

- User interface language
  During configuration, the text in the WinCC menus and dialogs is displayed in the user interface language. The user interface language also affects the labeling of operating elements, the parameters of the system functions, the online help, etc.
- Project languages
  Project languages are all languages in which a project will later be used. Project languages are used to create a project in multiple languages.

The two language levels are completely independent of one another. For example, you can create English projects at any time using a German user interface and vice versa.

Project languages

The following languages are differentiated within the project languages:

- Reference language
  The reference language is the language that you use to configure the project initially. During configuration, you select one of the project languages as the reference language. You use the reference language as a template for translations. All of the texts for the project are first created in the reference language and then translated. While you are translating the texts, you can have them displayed simultaneously in the reference language.
- Editing language
  You produce translations of the texts in the editing language.
  Once you have created your project in the reference language, you can translate the texts into the remaining project languages. Select a project language respectively as an edit language and edit the texts for the appropriate language variant. You can change the editing language at any time.

  Note

  When switching the project languages, the assignment to the keys on the keyboard also changes. For some languages (for example, Spanish), the operating system does not allow you to switch to the corresponding keyboard assignment. In this case, the keyboard assignment is switched to English.

- Runtime languages
  Runtime languages are those project languages that are transferred to the HMI device. You decide which project languages to transfer to the HMI device depending on your project requirements.
  You must provide appropriate controls so that the operator can switch between languages in runtime.

12.11.4.2 Language settings in the operating system

Introduction

The configuration PC operating system settings influence WinCC language management in the following areas:

- Selection of project languages
- Regional format of dates, times, currency, and numbers
- Displaying ASCII characters

Project language selection

A language is not available as a project language unless it is installed in the operating system.

Regional format of dates, times, currency, and numbers

WinCC specifies a fixed date and time format in the Date - Time field for the selected project language and runtime language.

In order for dates, times, and numbers to be presented correctly in the selected editing language, this language must be set in the Regional Options in the Control Panel.

Displaying ASCII characters

With text output fields, the display of ASCII characters as of 128 depends on the set language and the operating system being used.

If the same special characters are to be displayed on different PCs, the PCs must use the same operating system and regional settings.

12.11.4.3 Operating system settings for Asian languages

Settings on Western operating systems

If you want to enter Asian characters, you must activate the support for this language in the operating system.

The Input Method Editor (IME) is available in Windows for configuring Asian texts. Without this editor, you can display Asian text but not edit it. For more information on the Input Method Editor, refer to the documentation for Windows. To enter Asian characters when configuring, switch to the Asian entry method in the "Input Method Editor".

Switch the operating system to the appropriate language to have language-specific project texts, such as alarm texts, displayed in the simulator in Asian characters.

Settings on Asian operating systems

If you are configuring on an Asian operating system, you must switch to the English default input language to enter ASCII characters, for example, for object names. As the English default input language is included in the basic installation of the operating system, you do not need to install an additional input locale.

Enable language support

1. Open the system controller.
2. Select "Regional and Language Options".
3. On the "Languages" tab, activate the check box "Install files for East Asian languages".
4. Then click on "Details" under "Text Services and Input Languages". The dialog "Text Services and Input Languages" is opened.
5. On the "Settings" tab add the required default input language under the "Installed Services".
6. Select the language of the operating system in the "Language for non-Unicode programs" area in the "Advanced" tab.

12.11.4.4 Setting project languages

Selecting the user interface language

Introduction

The user interface language is used for displaying menu entries, title bars, infotexts, dialog texts and other designations in the WinCC user interface.

You can switch between the installed user interface languages during configuration. The labeling of the operating elements remains in the language you set when you added the object even if you change the user interface language.

Procedure

1. Select "Options > Settings" in the menu.
   The "Settings" dialog box is opened.
2. Select the desired user interface language under "General > General settings".

Result

WinCC will use the selected language as user interface language.

Enable project languages

Introduction

The project languages are set in the "Project languages" editor. You define which project language is to be the reference language and which the editing language.

Enable project languages

1. Click on the arrow to the left of "Languages & resources" in the project tree.
   The lower-level elements will be displayed.
2. Double-click on "Project languages".
   The possible project languages will be displayed in the working area.
3. Enable the relevant project languages.

Note

Copying multilingual objects

The copies of multilingual objects to a different project only include text objects in the project languages which are activated in the target project. Activate all project languages in the target project to include the corresponding text objects when transferring the copy.

Disabling project languages

1. Disable the languages which are not relevant for the project.

NOTICE
If you disable a project language, all text and graphic objects you have already created in this language will be deleted from the current project.

Selecting the reference language and editing language

Introduction

The project languages are set in the "Project languages" editor. You define which project language is to be the reference language and which the editing language. You can change the editing language at any time.

Requirements

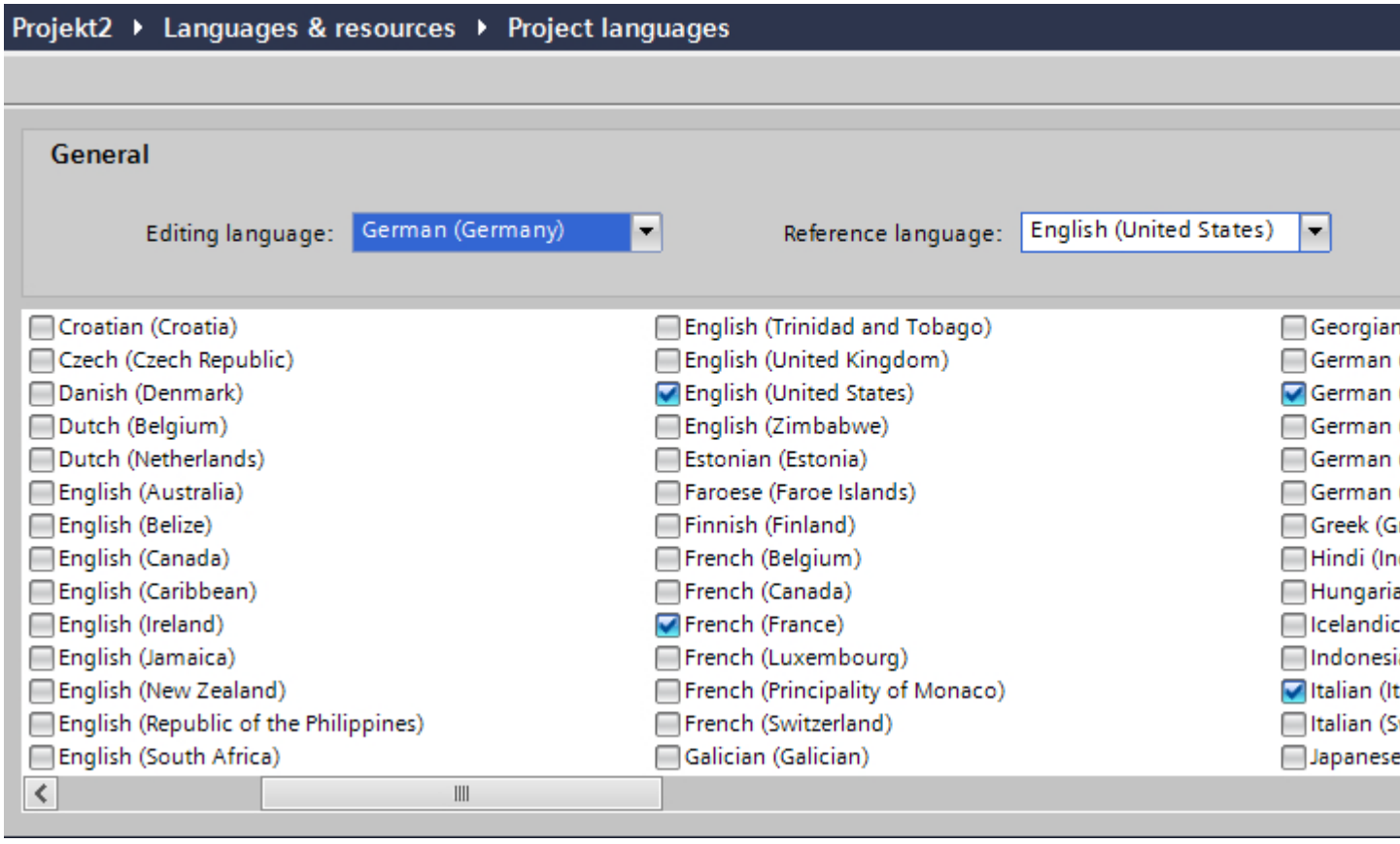
The "Project languages" editor is open.

Several project languages have been activated.

Selecting the reference language and editing language

1. Click the arrow in the drop-down list in the "General > Editing language" section.
2. Click on the required language in the drop-down list, for example, German.
3. Click on the arrow in the drop-down list in the "General > Reference language" section.
4. Click on the required language in the drop-down list, for example, English.

The language selection is displayed in the list box.



Result

You have now selected the editing and reference languages.

If you change the editing language, all future text input will be stored in the new editing language.

12.11.4.5 Creating one project in multiple languages

Working with multiple languages

Multilingual configuration in WinCC

You can configure your projects in multiple languages using WinCC. There are various reasons for creating a project in multiple languages:

- You would like to use a project in more than one country.
  You create the project in multiple languages but when the HMI device is commissioned, only the language spoken by the operators at the respective site will be transferred to the HMI device.
- The operators of a system speak a range of different languages.
  Example: An HMI device is used in China, but the service personnel understand only English.

Translating project texts

With WinCC, you can enter project texts directly in several languages in various different editors, for example, in the "Project texts" editor. WinCC also allows you to export and import your configuration for translation purposes. This is particularly advantageous if you configure projects containing a large amount of text and want to have it translated.

Language management and translation in WinCC

The following editors are used to manage languages and translate texts in WinCC:

Editor	Short description
Project languages	Selection of project languages, editing language and reference language.
Languages and fonts	Management of runtime languages and fonts used on the HMI device.
Project texts	Central management of configured texts in all project languages.
Graphics	Graphic library for managing graphics and their language-specific versions.

Project text basics

Texts in different languages in the project

Texts that are output on display devices during processing are typically entered in the language in which the automation solution is programmed. Comments and the names of objects are also entered in this language.

If operators do not understand this language, they require a translation of all operator-relevant texts into a language they understand. You can therefore translate all the texts into any language. In this way, you can ensure that anyone who is subsequently confronted with the texts in the project sees the texts in his/her language of choice.

User texts and system texts

In the interests of clarity, a distinction is drawn between user texts and system texts:

- User texts are texts created by the user.
- System texts are texts created automatically and which are a product of configuration in the project.

The project texts are managed in the project text editor. This can be found in the project tree under "Languages & Resources > Project texts".

Examples of multilingual project texts

You can, for example, manage the following types of text in more than one language:

- Display texts
- Alarm texts
- Comments in tables
- Labels of screen objects
- Text lists

Translating texts

There are two ways of translating texts.

- Translating texts directly
  You can enter the translations for the individual project languages directly in the "Project texts" editor.
- Translating texts using reference texts
  You can change the editing language for shorter texts. You can enter the new texts in the editing language while the texts of the reference language are displayed.

Translating texts directly

Translating texts

If you use several languages in your project, you can translate individual texts directly. As soon as you change the language of the software user interface, the translated texts are available in the selected language.

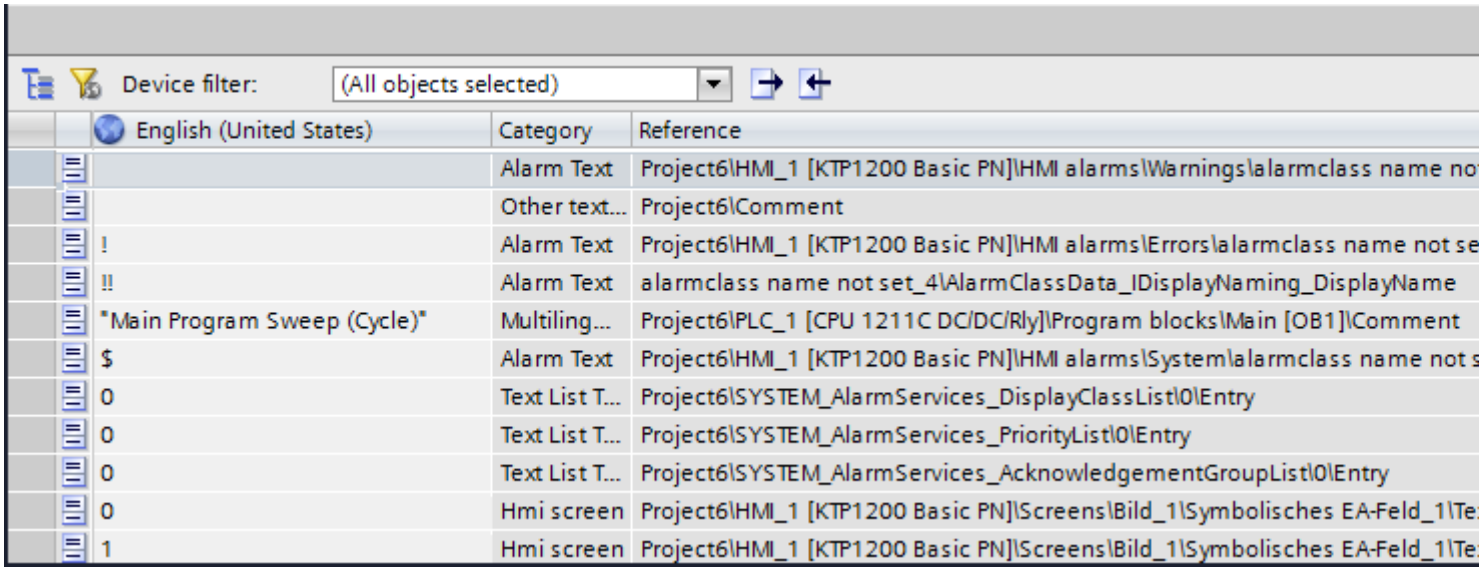
Requirements

- You are in the project view.
- A project is open.
- You have selected at least two further project languages.

Procedure

Proceed as follows to translate individual texts:

1. Click on the arrow to the left of "Languages & resources" in the project tree.
   The elements below this are displayed.
2. Double-click on "Project texts".
   A list with the texts in the project is displayed in the work area. There is a separate column for each project language.



3. To group identical texts and translate them simultaneously, click on the "" button in the toolbar.
4. To hide texts that do not have a translation, click on the button in the toolbar.
5. Click on an empty column and enter the translation.

Result

You have translated individual texts in the "Project texts" editor. The texts will then be displayed in the runtime language.

Translating texts using reference texts

Introduction

After changing the editing language, all texts are shown in input boxes in the new editing language. If there is not yet a translation available for this language, the input boxes are empty or filled with default values.

If you enter text again in an input field, this is saved in the current editing language. Following this, the texts exist in two project languages for this input field, in the previous editing language and in the current editing language. This makes it possible to create texts in several project languages.

You can display existing translations for an input box in other project languages. These serve as a comparison for text input in the current editing language and they are known as the reference language.

Requirement

There is at least one translation into a different project language for an input field.

Procedure

To display the translation of an input cell in a reference language, follow these steps:

1. Select "Tasks > Languages & resources" in the task card.
2. Select a reference language from the "Reference language" drop-down list.

Result

The reference language is preset. If you click in a text block, translations that already exist in other project languages are shown in the "Tasks > Reference text" task card.

Exporting project texts

Project texts are exported for translation. Texts are exported to Office Open XML files ending in ".xlsx". These files can be edited in Microsoft Excel, for example.

You can exchange the file with the translators and import it back to the project as soon as it has been translated.

Requirements

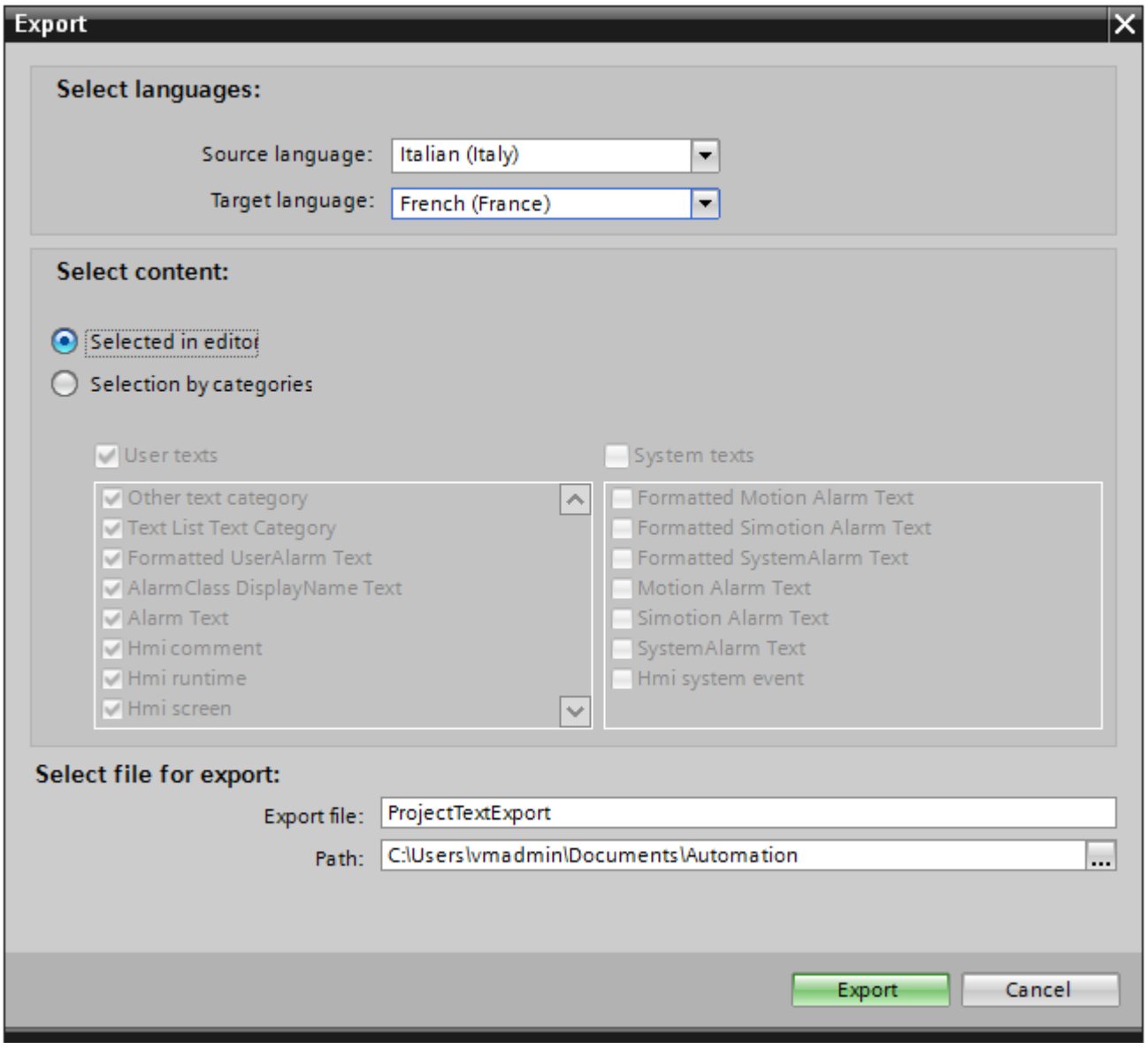
- At least two languages have been enabled in the "Project languages" editor, for example, Italian and French.

Exporting project texts

To export individual project texts, proceed as follows:

1. Click on the arrow to the left of "Languages & resources" in the project tree.
   The child elements are displayed.
2. Double-click on "Project texts". The "Project texts" editor will open.
3. Select the texts you want to export.

4. Click on the button. The "Export" dialog opens.



5. From the "Source language" drop-down list, select the language from which you wish to translate, for example Italian.
6. From the "Target language" drop-down list, select the language into which the texts are to be translated, for example, French.
7. Enter a file path and a file name for the export file in the "Export file" input field.
8. Click "Export".

Result

The texts selected in the "Project texts" editor are written to an xlsx file. The xlsx file will be stored in the specified folder.

You can alternatively select and export all project texts from categories. Select "User texts" or "System texts" in the "Export" dialog in line with the type of texts you wish to export. In this case, export can additionally be limited by categories.

Note

Project texts in library objects cannot be exported.

Importing project texts

Edit the xlsx file or send it to a translator. Import the texts once they have been translated. The foreign languages will be imported to the relevant object in the project.

Note

In WinCC, you only import the previously exported project texts into the same project. Importing into a different project is not supported.

Requirements

- At least two languages have been enabled in the "Project languages" editor, for example, Italian and French.

Importing project texts

To import a project text file, proceed as follows:

1. Click on the arrow to the left of "Languages & resources" in the project tree.
   The lower-level elements will be displayed.
2. Double-click on "Project texts". The "Project texts" editor will open.
3. Click on the  button. The "Import" dialog opens.
4. Select the path and file name of the import file from the "Import file" field.
5. Activate the "Import source language" check box if you have made changes to the source language in the export file and would like to overwrite the entries in the project with the changes.
6. Click on "Import".

Result

You have imported the project texts.

12.11.4.6 Using language-specific graphics

"Project graphics" editor

Introduction

You use the "Project Graphics" editor to manage the configured graphic objects in different language versions. Multilingual projects sometimes also require language-specific versions of the graphics, for example, if

- the graphics contain text;
- cultural aspects play a role in the graphics.

Opening the "Project graphics" editor

Double-click in the project tree on "Languages and resources > Project graphics".

Work area

The work area displays all configured graphic objects in a table. There is a separate column in the table for each project language. Each column in the table contains the versions of the graphics for one particular language.

In addition, you can specify a default graphic for each graphic to be displayed whenever a language-specific graphic for a project language does not exist.

Preview

The preview shows you how the graphics will look on various devices.

Storing an image in the graphics library

Introduction

You use the "Graphics" editor to import graphics you want to use in screens in the "Screens" editor. It also allows you to manage language-specific versions of graphics. A preview shows the graphic displays on various HMI devices.

Note

File names for language-dependent graphics

Case is relevant in the file names of language-dependent graphics. Make sure that the format is consistent for all languages.

Requirement

- The language-dependent versions of a graphic are available.
- Multiple languages have been enabled in the "Project languages" editor.
- The "Graphics" editor is open.

Inserting graphics

1. Click "Add" in the "Graphics library" table. A dialog opens.
2. Select the required graphic file.
3. Click "Open" in the dialog box.
   The graphic will be imported to the project and displayed in all cells in this row in the "Graphics" editor.
4. Click in the corresponding cell of a language for which a language-dependent version of this graphic exists.
5. Select "Add graphic" from the shortcut menu. A dialog box opens.
6. Select the desired graphic file and click "Open."
   The language-dependent version is inserted in the table in place of the reference language graphic.
7. Then, in the "Default graphic" column, import a graphic to be displayed in runtime for those languages for which there is no language-specific graphic.

You can also drag&drop a graphic from Windows Explorer to the relevant position in the "Graphics library" table.

Displaying graphics in the HMI device preview

1. Click on a graphic in the table.
2. Select the required HMI device under "Properties > Graphics settings > Device preview" in the Inspector window.
   The graphic will then be displayed as it will appear in runtime on the selected HMI device.

Result

The graphics added are available in the "Graphics" editor. The graphic assigned to the respective editing language will be displayed during editing. The default screen will be displayed in all editing languages for which no screen has been imported.

The screens assigned to the respective runtime language are displayed during runtime. The default screen is displayed in all runtime languages for which a screen has not been imported.

Note

If you disable a project language, all of the graphic objects you have already created in this language will be deleted from the current project.

Storing an external image in the graphics library

Introduction

To display graphics that have been created in an external graphics program in your screens, you will first have to store these graphics in the graphics browser of the WinCC project.

Requirement

- Multiple languages have been enabled in the "Project languages" editor.
- The "Graphics" editor is open.
- There is a graphic in the "Graphics" editor.

Creating and adding a new graphic as an OLE object

1. Click "Add" in the "Graphics library" table. A dialog box opens.
2. Navigate to the folder in which the graphic is stored.
3. Click "Open" in the dialog box.
   The graphic will be imported to the project and displayed in all cells in this row in the "Graphics" editor.
4. Click in the corresponding cell of a language for which a language-dependent version of this graphic exists.
5. Select "Insert object" from the shortcut menu. The "Insert object" dialog box opens.

   Note

   In addition, the dialog "External application running..." will open. The dialog will not close until you exit the external application.

6. Select "Insert object > Create new" and an object type in the dialog.
7. Click "OK." The associated graphic program is opened.
8. Close the graphics program once you have created the graphic.
   The graphic will be stored in the graphic programming software standard format and added to the graphic browser.

Inserting created graphics in WinCC

1. Click in the corresponding cell of a language for which a language-dependent version of this graphic exists.
2. Select "Insert object" from the shortcut menu. The "Insert object" dialog box opens.

   Note

   In addition, the dialog "External application running..." will open. The dialog will not close until you exit the external application.

3. From the "Insert object" dialog box, select "Create from file."

4. Click the "Browse" button.
5. Navigate to the created graphic and select it.

Note

To import graphics files, note the following size restrictions:

*.bmp, *.tif, *.emf, *.wmf ≤4 MB

*.jpg, *.jpeg, *.ico, *.gif "*≤1 MB

Result

The OLE objects added are available in the "Graphics" editor.

Versions of the graphics for the current editing language are displayed in the "Screens" editor. The default graphic is displayed in all editing languages for which no screen has been imported.

The graphic is displayed in runtime in the set runtime language. The default graphic is displayed in all runtime languages for which no graphic has been imported.

You can double-click OLE objects in your library to open them for editing in the corresponding graphic editor.

12.11.4.7 Languages in Runtime

Languages and fonts in Runtime

Using multiple Runtime languages

You can decide which project languages are to be used in Runtime on a particular HMI device. The number of Runtime languages that are available at one time on the HMI device depends on the device. To enable the operator to switch between languages in Runtime, you need to configure a corresponding operator control.

When Runtime starts, the project is displayed according to the most recent language setting. When Runtime starts the first time, the language with the lowest number in the "Order for language setting" is displayed.

"Language & font" Runtime setting

Configure the following under "Language & font":

- The project languages available as Runtime languages for the respective HMI device.
- The order in which the languages are switched.
- Default font for a Runtime language

The license status of the fonts is also shown in the "Download font" area. If a specific license is required for a protected font, this is indicated in the "License status" column.

Note

Post installing fonts

When a configured font on the configuration computer is missing, the project is not compiled. Re-installation is required.

After installing a font, restart the TIA Portal and completely re-compile your project. Only then can the HMI device be compiled without errors.

Note

Runtime language in controls (RT Professional)

If you have configured a control in a specific Runtime language that is not a user-interface language, it can happen that some texts, for example, system texts in the status bar, appear in the English language.

Note

Specify Runtime languages as input languages

To enter and edit data in Runtime, configure the specified Runtime languages also as input languages for the keyboard of your PC.

See also

Standardizing font for all languages (Page 5870)

Methods for language switching

Introduction

You need to configure language switching if you want to have multiple runtime languages available on the HMI device. This is necessary to enable the operator to switch between the various Runtime languages.

Methods for language switching

You can configure the following methods for language switching:

- Direct language selection
  Each language is set by means of a separate button. In this case, you create a button for each Runtime language.
- Language switching
  The operator switches the languages using a button.

Regardless of the method used, the button names must be translated into each of the languages used. You can also configure an output field that displays the current language setting.

Enabling the runtime language

Introduction

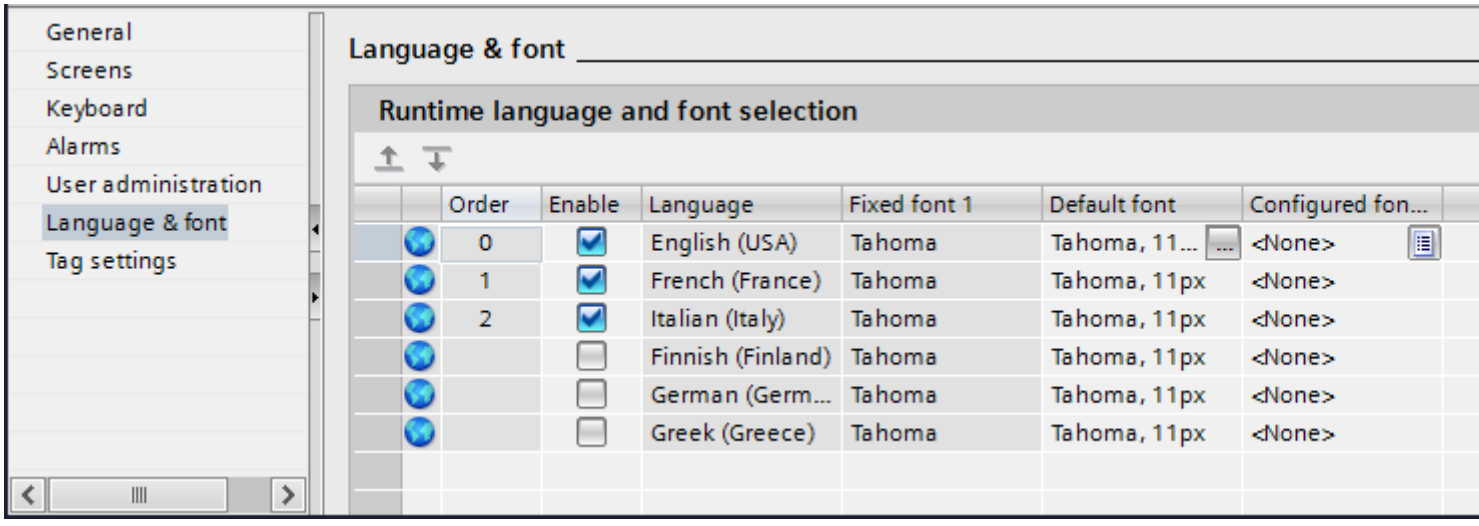
The "Language & Font" editor shows all project languages available in the project. Here you select which project languages are to be available as runtime languages on the HMI device.

Requirements

Multiple languages have been selected in the "Project languages" editor.

Procedure

1. Double-click on "Runtime settings" in the project tree.
2. Click on "Language & Font".
3. Select the following languages:
   - English
   - French
   - Italian



Result

You have now set three runtime languages. A number is automatically assigned to each language in the "Order" column. The enabled runtime languages are transferred with the compiled project to the HMI device.

If the number of languages selected exceeds the number that can be transferred to the HMI device, the table background changes color.

See also

Standardizing font for all languages (Page 5870)

Setting the runtime language order for language switching

Introduction

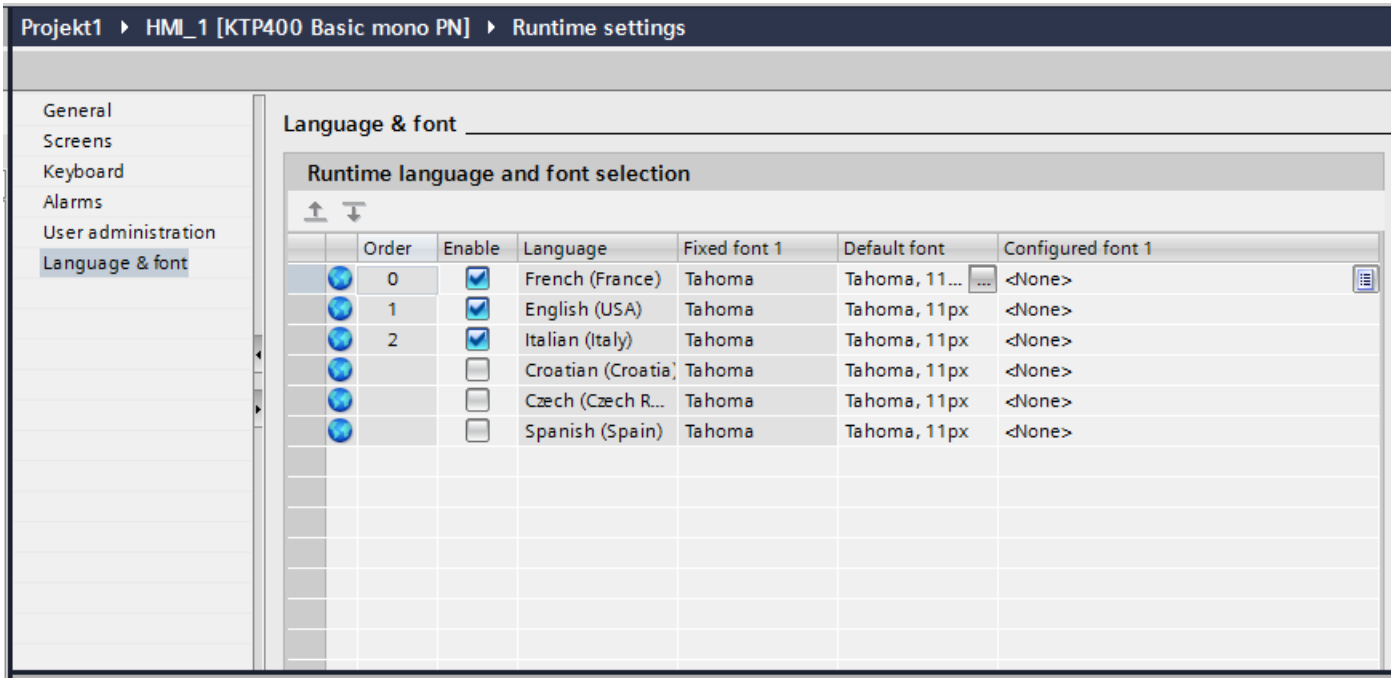
You specify the language order for Runtime language switching. The first time Runtime starts, the project is displayed in the language with the lowest number in the "Order" column.

Requirements

- Multiple languages have been enabled in the "Project languages" editor.
- The "Language & Font" editor is open and three Runtime languages have been set in the following order:
  1. English
  2. Italian
  3. French

Procedure

1. Select the runtime language "English".
2. Click the button. The runtime language "English" is moved down a place. The number will automatically be changed to "1" in the "Order" column.



Result

You have changed the order of Runtime languages. The first time runtime starts, the project will be displayed in the language with the lowest number. If the language is switched, this will happen in numerical order.

Setting the default font for a runtime language

Introduction

You can specify the font used to display the texts for each runtime language on the HMI device in the "Language & Font" editor. The default font is used in all texts, such as dialog texts, for which you cannot define a specific font.

WinCC offers only fonts supported by the HMI device.

Note

"Bold" and "Italic" fonts

"Bold" and "italic" fonts are not supported on all HMI devices in Runtime.

Requirements

- Multiple languages have been enabled in the "Project languages" editor.
- Three runtime languages have been enabled in the "Language & Font" editor.
  1. Chinese
  2. German
  3. French

Procedure

1. Double-click on "Runtime settings" in the project tree.
2. Click on "Language & Font". The table shows the runtime languages and fonts set.
3. Click in the "French" row in the "Default font" column.
4. Select the font to be used by default if a font cannot be selected for a given text.

Result

The project texts for the runtime language "French" are displayed on the HMI device in the selected font.

These fonts are transferred to the HMI device during a transfer operation.

The default font is also used for the representation of dialogs in the operating system of the HMI device. Select a smaller font as default if the full length of the dialog texts or headers is not displayed.

Standardizing font for all languages

Introduction

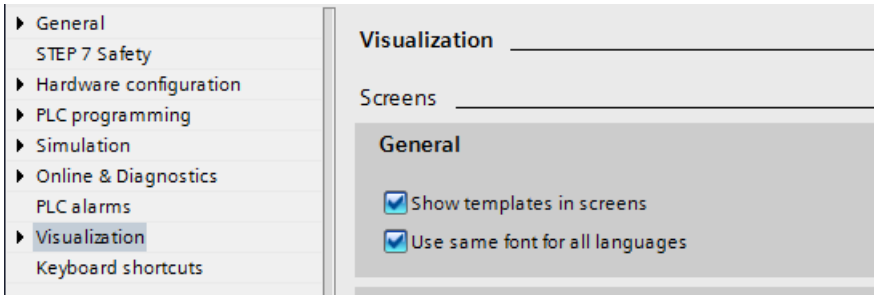
You can standardize the font for all project languages during configuration with the "Use same font for all languages" option.

Requirement

- Multiple languages have been selected in the "Project languages" editor.
- Multiple languages have been selected in the "Language & font" editor.
- The same font is defined for the selected runtime languages under "Configured font".

Procedure

1. In the "Options > Settings > Visualization > General" menu, select the "Use same font for all languages" option.



Result

You have enabled the option "Use same font for all languages". If you change the font of an object in one language during configuration, this font will be applied to all active languages.

See also

Languages and fonts in Runtime (Page 5865)

Enabling the runtime language (Page 5867)

Selecting the log language

Introduction

In the "Runtime settings > General" editor, select the language to be used for writing to logs in runtime.

Requirements

- The languages used in your project are activated in the "Project languages" editor, for example "German" and "English" .

Procedure

1. Double-click on "Runtime settings" in the project tree.
2. Click on "Language & Font".
3. Activate the runtime languages, for example, "German" and "English".
4. Specify the "order":
   - 1 German
   - 2 English

5. Click on "Runtime settings > General".
6. Select "German" for "Logs > Log language".

Result

After loading, the project will start in the runtime language "German". The logs are now written in German. During runtime, the operator switches the runtime language to English. The logs will still to be written in German.

Specific features of Asian and Eastern languages in Runtime (Panels, Multipanels, Comfort Panels, Mobile Panels, RT Advanced)

Introduction

Note the following special considerations for the operation in Runtime of projects for Asian languages.

Note

During configuration, only use the Asian fonts that your configuration computer supports.

Memory requirement for Asian character sets

The memory requirement is greater when using Asian languages. Therefore look out for corresponding error messages when compiling the project.

Font size for Asian character sets

Use at least a font size of 10 points to display the text of projects created for Asian languages in Runtime. Asian characters will become illegible if smaller font sizes are used. This also applies to the default font in the Runtime settings under "Language & font".

Text field length for Asian languages

Make allowances for an appropriate length of the text fields when working on multilingual projects with Asian languages. Field contents may be partially hidden, depending on the font and the font size.

1. Open the "Properties > Appearance" text box in the Inspector window.
2. Under "Fit to size", disable the "Auto-size" option.
3. Verify the proper display in Runtime.

12.11.4.8 Example of multilingual configuration

Example: Configuring a button for language switching

Introduction

In this example, you configure a button that can be used to toggle between multiple runtime languages during runtime.

Requirements

- You have completed the "Configuring a button in multiple languages" example.
- The "Screen_1" screen is open.
- The button on the screen has been selected.

Procedure

1. In the Inspector window, select "Properties > Events > Press".
2. Click on "Add function" in the table.
3. Select the "SetLanguage" system function and the "Switch" setting.

Result

You have assigned the button the function "SetLanguage". Pressing the button during runtime will switch the runtime language. The runtime languages are switched in the order specified by the number sequence in the "Languages and fonts" editor.

Example: Configuring a button in multiple languages

Introduction

In this example, you configure a "Sprache umschalten" button in German and "Switch language" button in English.

Requirements

- The languages "German" and "English" have been enabled in the "Project languages" editor.
- German has been set as editing and reference language.
- You have created and opened the "Screen_1" screen.
- The Inspector window is open.

Procedure

1. Drag-and-drop a button from the "Tools" task card into the screen.
   The button will be added to the screen.
2. In the Inspector window, open "Properties > Properties > General".
3. Enter the text ""Sprache umschalten" under "Text > off".
4. Press <Enter> to confirm. The button is named.
5. Open the "Tasks" task card.
6. Select "English" under "Languages & Resources > Editing language".
7. Enter the label "Switch Language" under "Properties > Properties > General > Text > Off" in the Inspector window.

Result

The button name is configured in German and English language. The button name corresponding with the current Runtime language is shown in Runtime.

Example: Configuring a button for language switching for each runtime language

Introduction

In this example, you configure a "Sprache umschalten" button in German and "Switch language" button in English.

Requirements

- The following languages have been enabled in the "Project languages" editor
  - German
  - English
  - Italian
- All languages have been set as runtime languages in the "Runtime settings > Language & Font" editor.
- You have created and opened the "Screen_1" screen.
- Three buttons have been created on the screen:
  - Button_1 labelled "Deutsch"
  - Button_2 labelled "English"
  - Button_3 labelled "Italiano"
- The Inspector window is open.

Procedure

1. Select "Button_1".
2. In the Inspector window, select "Properties > Events > Press".
3. Click on <Add function> in the table.
4. Select the "SetLanguage" system function.
5. Click on the "Switch" field.
6. Click on the  button.
7. Select "Runtime language". The field will be highlighted in red.
8. Select "German" from the drop-down list.
9. Repeat steps 1 - 8 for the other two buttons and select the corresponding runtime language.

Result

You have configured three buttons for switching language in runtime. Each button will switch to a different runtime language. For example, clicking on the "English" button during runtime will switch the runtime language to English.

See also

Example: Configuring a button for language switching (Page 5873)

12.11.5 Replacing devices

12.11.5.1 Basics (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

You can use existing configurations for new devices and optimize these configurations with little manual effort.

All data configured by you is retained in the configuration data. This means you do not need to copy individual objects of one device and paste them to another.

Principle

The following applies when you replace devices:

- Only functions supported by the new device are available. Only configuration data supported by the new device are displayed.
  This affects
  - recipes,
  - objects available on the screens,
  - available system functions,
  - available communication logs, etc.
- The number of supported objects, such as screens or tags, may be limited on the new device. If the existing objects exceed the limitations on the new device, the objects are displayed in full. The objects are, however, highlighted in color in the individual editors. An error is generated when the project data is compiled.
  Manual post processing is required when switching to a device with fewer features.
  Example: Limited number of connections
  All connections will be highlighted in color as invalid if fewer connections are supported on the new device than have been configured. Delete any excess connections.
- After an HMI device has been replaced, you should check the appearance of the configured screens. Changing the size of the display may result in changes to the position and appearance of screen objects, e.g. the recipe view and alarm view.

Note

If you replace a Panel and select a PC Station as your new device, for example, WinCC Runtime Advanced will automatically be moved below the PC Station in the project tree.

12.11.5.2 Device-specific functions

Device-specific functions (Basic Panels, Panels, Comfort Panels, RT Advanced)

Functions dependent on the device

Functions dependent on the device are implemented as follows:

- Colors
  The color is changed automatically when you switch from a device with full color display to one with a smaller color range.
  If you change the color manually and then change back again to a device with a larger range of colors, the reduced range of colors will be retained.
- Fonts
  Any configured font not available on a device will be replaced by a similar one or by the configured default font. The default font depends on the device selected.

- Character sets with different font sizes
  Avoid using too many different font sizes when configuring the following devices:
  - OP 73
  - OP 77A
  - TP 177A

  A character set is downloaded to the device for each font size. Check the Inspector window during compilation to see how much of the device memory is being used by the character sets.
- Font size
  Use small Windows fonts to display the text on devices. If you use large Windows fonts, then, depending on the size of the display, the text will not be displayed in full.
  Using font sizes of 28 pixels or more for the OP 77A and TP 177A devices will affect device performance.
  The character scope is much greater for Asian languages. The use of different font sizes therefore has serious implications on the memory requirements of all devices.
  Use the same font type for all large characters throughout the project to ensure effective and efficient configuration.
- Screens and screen objects
  If the new device supports a different resolution than the previous device when you replace a device, there are several ways to adjust the screens.
  Adjust the size of the screens to the new device in the menu under "Options > Settings > Visualization > Fit to size screen".

Key assignment when replacing devices (Basic Panels)

Introduction

The devices available each have different function keys. The functions configured for these keys will be mapped to the available function keys of the new device if the device is replaced.

Function key mapping

The function keys below the display are mapped from left to right to the new device. If the new device has fewer keys, the keys it does not have are not mapped.

Note

HMI devices without LED keys

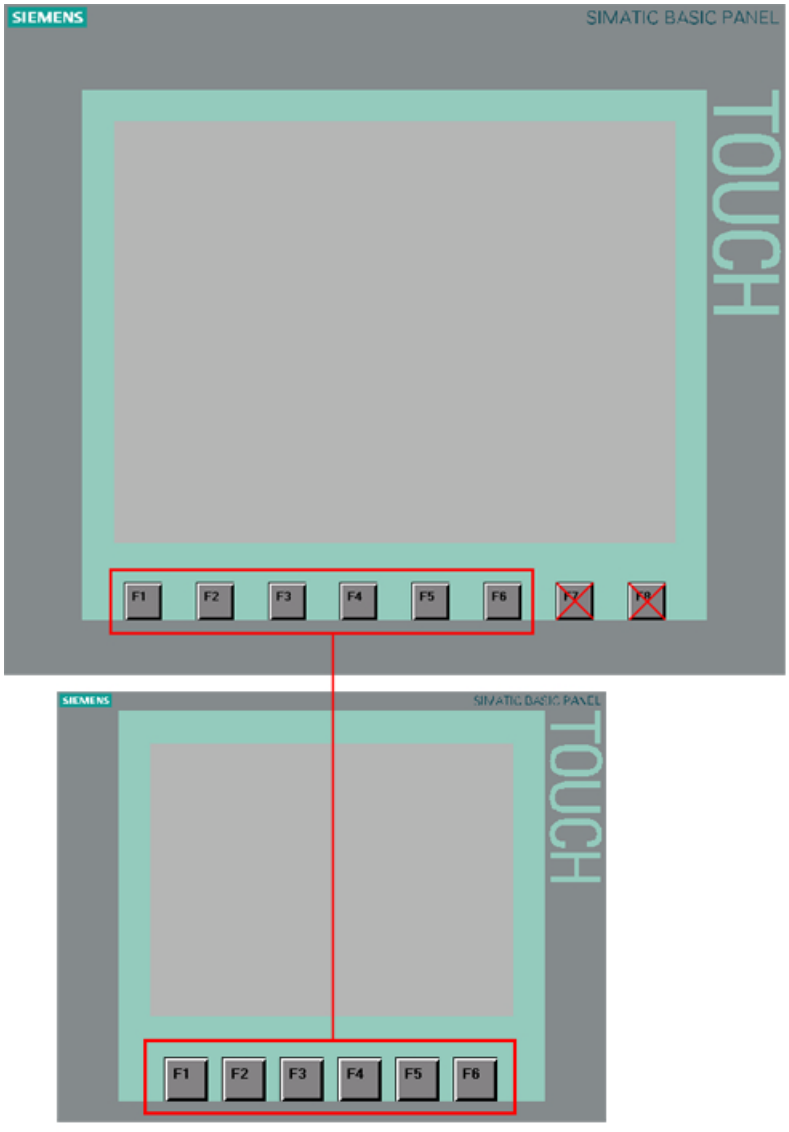
If you replace a device with configured LED keys with a device without LED keys in the Engineering System, runtime will not start after the project data have been transferred to the device.

Delete the LED key configuration before you replace the device.

Example: Replacing a KTP1000 Basic with a KTP600 Basic

You have configured a function for F2 in KTP1000 Basic. This function is triggered by F2 following replacement with a KTP600 Basic.

If you have used F7 in a KTP1000 Basic, this function will no longer be available if the panel is replaced with a KTP600 Basic.



Mapping of control keys and cursor keys

The following keys are mapped only to the same keys of the new device:

- HELP
- ESC
- ACK

- ENTER
- PAGE UP
- PAGE DOWN
- CURSOR UP
- CURSOR DOWN

12.11.5.3 Adjusting screens to the new device

Screen adjustment options (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

Select fit to size for screens before you replace a device. Fit to size is particularly important when switching devices with different display resolutions.

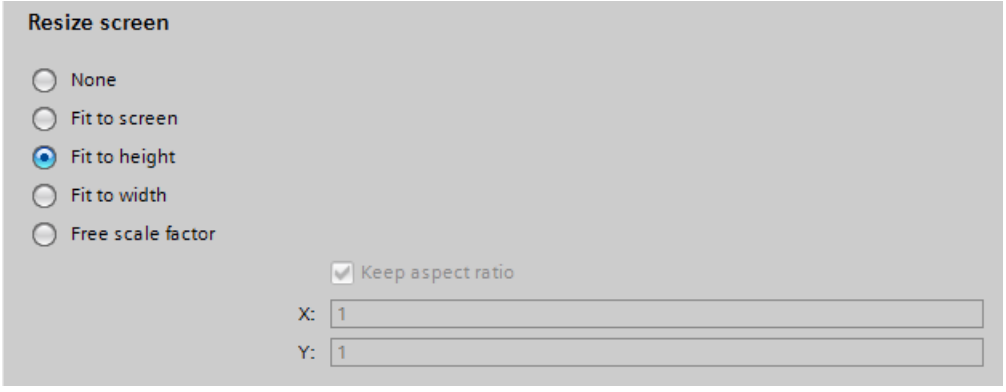
Object adjustment to content can be prevented for objects such as graphic views or text fields.

Note

The objects are distorted if you replace a device with a landscape format display with a device with a portrait format display. The difference in display format can, for example, result in object labels being cut off and content not being fitted to the object. You must therefore adjust the screens to the new device once you have replaced devices.

Screen adjustment when replacing devices

Adjust the size of the screens to the new device in the menu under "Tools > Settings > Visualization > Resize screen".



Select one of the following settings.

None (default setting)

The screens are not scaled. The objects in the screen retain their position and size. Use this setting as first test for checking of a possible exchange result because there are no rounding losses during forth and back exchange.

This option may result in objects being outside the configurable area if the display of the new device is smaller than the old one.

Fit to screen

The position and object size are adjusted to the new display size. Resizing takes place along the x-axis and the y-axis. Graphics and font size are adjusted accordingly.

Fit to height

The aspect ratio is maintained and the screens are adjusted to the height of the new device.

Use this option when you are replacing a device with display format 4:3, for example, with a device with widescreen.

Fit to width

The aspect ratio is maintained and the screens are adjusted to the width of the new device.

Use this option when you are replacing a device with widescreen, for example, with a device with display format 4:3.

Free scale factor

You select a free scale factor for screen adjustment. You can specify a factor for the x-axis and the y-axis.

Using a free scale factor of < 1 may distort the objects. Object labels may, for example, be cut off and the content may not be fitted to the object.

You must therefore adjust the screens to the new device once you have replaced devices.

Note

The aspect ratio is not adjusted for objects with a fixed aspect ratio, for example, gauge, circle. The objects are displayed on the new device with the same aspect ratio as prior to the replacement of the device.

Fit objects to contents (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

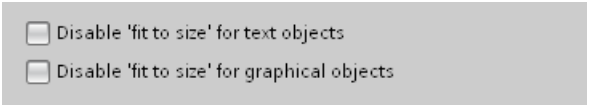
For some objects, you can specify fit to respective content in the Inspector window, for example:

- Text field: fit to text content
- I/O field: fit to text content
- Symbolic I/O field: fit to text content or to text list
- Graphic view: fit to included graphic

Fit to size for text and graphic objects

Disable automatic fit to size of the individual objects in the menu under "Options > Settings > Visualization > Resize screen and screen objects > Fit to content". This results in scaling of the objects as specified under "Options > Settings > Visualization > Resize screen and screen objects".

Select the objects which are not automatically fitted to size.



- If "Disable 'fit to size' for text objects" is activated, automatic fit to size is ignored in the text object properties.
  If you have activated "Fit screen to window height", the text field along with the other objects is scaled in accordance with the height of the new device.
- If "Disable 'fit to size' for graphic objects" is activated, automatic fit to size is ignored in the graphic object properties.
  If you have activated "Fit screen to window width", the graphics view along with the other objects is scaled in accordance with the width of the new device.

Note

The settings have no effect on screen objects whose size cannot be changed, such as alarm indicators or screen objects with a fixed aspect ratio.

"Disable 'fit to size' for text objects" and "Disable 'fit to size' for graphic objects" have no effect if:

- You have activated "Resize screen and screen objects > None".

- You have activated "Fit screen to window width and height" and the new device has the same resolution as the current device.

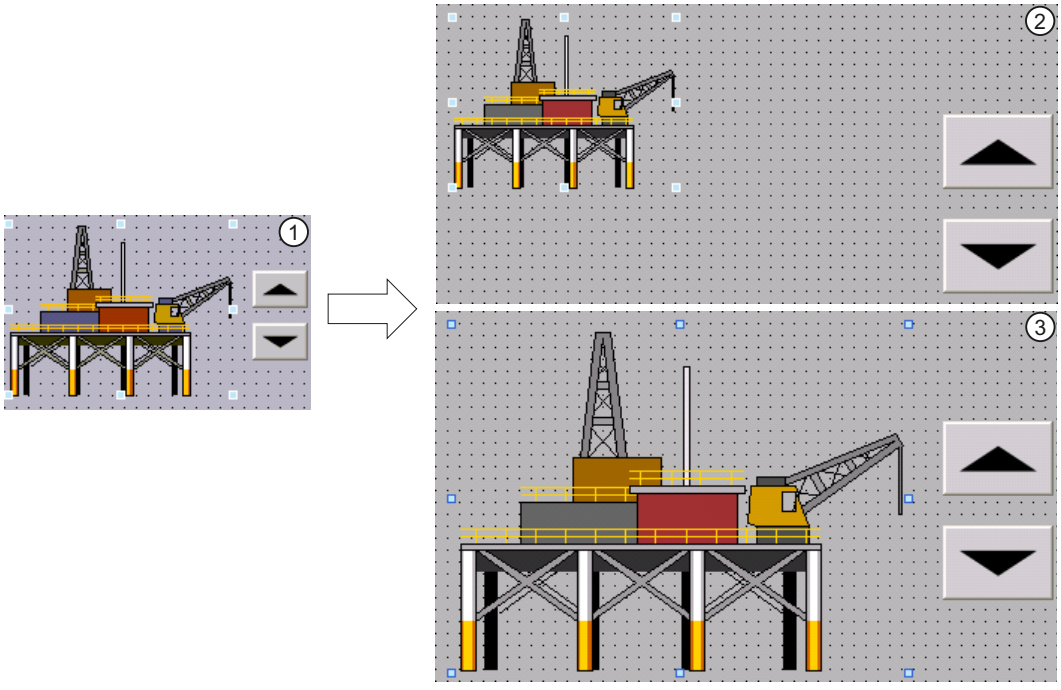
- You have activated "Fit screen to window height" and the new device has the same resolution as the current device.

- You have activated "Fit screen to window width" and the new device has the same resolution as the current device.

- You have selected "Resize screen and screen objects > None" and position "Top left" which means an adaptation is not necessary.

Example

The figure below shows the effects of automatic sizing using a graphic object with two buttons aligned as an example:



① Initial situation:

- Two buttons are aligned on a graphic object.
- The option "Fit object size to graphic" or "Adjust object size to graphic" is activated in the object properties of the graphic object under "Display > Sizing".

② Option 1: The original properties of the graphic object are to be maintained after switching the HMI device.

- Deactivate the option "Disable 'fit to size' for graphical objects" in the settings under "Size adaptation of objects".

Effect: The graphic object retains its original size after switching the HMI device. The alignment to the buttons is lost.

③ Option 2: The graphic object is to be placed relative to the new screen resolution after switching the HMI device.

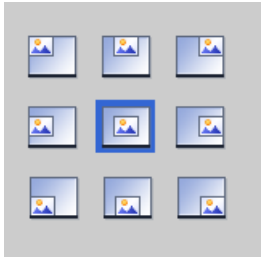
- Activate the option "Disable 'fit to size' for graphical objects" in the settings under "Size adaptation of objects".

The option "Fit graphic to object size" is activated automatically in the object properties of the graphic object. The two buttons are properly aligned on the graphic object even after switching the HMI device.

Specifying the position of screen objects (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

There are various ways to adjust the position of screen objects to the new device.



Select position

Adjust the position of the screen objects to the new device in the menu under "Options > Settings > Visualization > Fit to size screen > Select position".

Example

The following option aligns the objects with the top left edge.



The following object centers the objects in the middle of the screen.



12.11.6 Copying between devices and editors

12.11.6.1 Basics

Basics (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

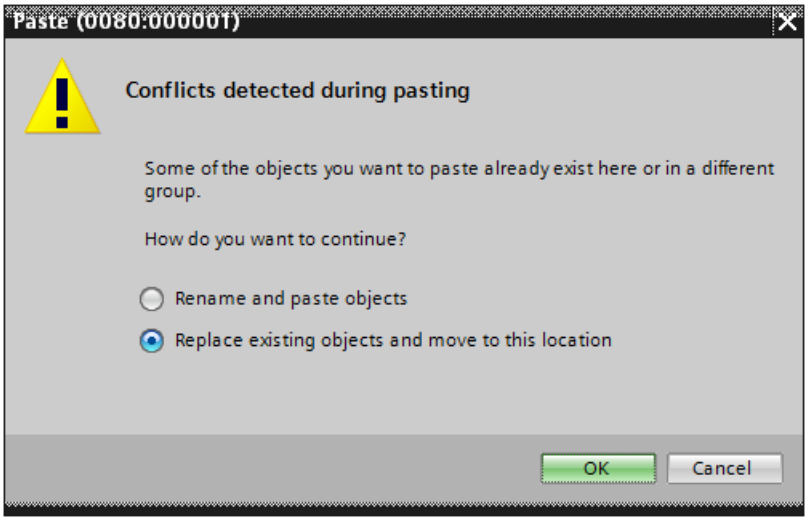
Copying and pasting within an HMI device

You can copy and paste objects, such as display objects, within an HMI device. If the object is already created in the editor, when the object name is inserted a number is automatically attached, in accordance with the following principle:

- "<Object_name>_1" is renamed to "<Object_name>_2".
- "<Object_name>_2" is renamed to "<Object_name>_3".

Copying and pasting between HMI devices

You can also copy and paste between HMI devices. If an object with the same name already exists, you have the following options:



Note

Exception to this basic rule

Copying and pasting of an alarm class that has been generated from project-wide alarm class is handled differently than with this basic rule. When the copied alarm class already exists in the target HMI device within the same project, the "Paste" command is not performed.

Copying user-defined folders

You can create user-defined folders for editors, for example, for HMI tags, screens, etc.

You can copy user-defined folders and paste them into another HMI device. The objects contained in a user-defined folder may exceed the limitations applying to the other HMI device, such as the number of supported screens. After they have been pasted, all the objects are displayed. An error is displayed when the project data is compiled.

System folders cannot be copied.

Unsupported objects and functionalities (Panels, Comfort Panels, RT Advanced)

Introduction

When an object is copied, all its properties and settings are transferred to the target HMI device.

Unsupported objects

Objects that are not supported in the target HMI device cannot be pasted.

Note

When you copy a screen containing objects which are not supported by the destination HMI device, the objects remain in the background. When you copy the screen again and the new device supports the objects, they are displayed again.

Invalid objects

The following objects become invalid once they have been pasted into the target HMI device:

- Referenced objects that do not exist in the target HMI device.
- Objects with settings that are not supported in the target HMI device.
- System functions that were configured for objects and that are not supported in the target HMI device.

Invalid objects are highlighted by a color coding. Select a supported object or create a new one. If you retain an invalid object, an error will be displayed when the project data is compiled.

Colors and fonts

Colors and fonts are supported to varying degrees by HMI devices. When unsupported colors and fonts are pasted, they are replaced by supported colors and fonts. When you paste the same object back into the source HMI device, the original settings become active again.

Alarm classes

Only some of the enabled alarm classes are enabled.

When you configure an alarm window, for example, in the global screen in Project_1, you copy the alarm window and paste it to the global screen in Project_2.

Only some of the enabled alarm classes are enabled after pasting.

This behavior applies to the following display objects:

- Alarm window
- Alarm indicator
- Alarm view

12.11.6.2 Copying and pasting

Copying screens (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You copy one or more screens from the "Screens" folder and paste them into the "Screens" folder of another device.

Type and size of the displays

In the case of HMI devices with keys, the available keys are displayed automatically in the screen. When a screen is copied between HMI devices, the keys are either displayed or hidden; functions configured for function keys are not transferred.

If there is less space for the screen in the target HMI device than in the source HMI device, you can adjust the size of and the spacing between existing objects.

Automatic fit to size for objects

1. Select "Options > Settings > Visualization > Resize screen and screen objects" in the menu.
2. Activate, for example, "Fit to height".

Copying recipes within an HMI device (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

"Recipes" Editor

You can copy recipes, recipe elements and recipe data records within each table. You copy a recipe element to another recipe.

Only WinCC Runtime Professional: You can copy a recipe view element to another recipe view. If a recipe view element of the same name already exists, a conflict dialog is displayed. You can select whether to replace or rename the recipe element. You can copy recipe elements to the first empty row of the "Recipe views" editor, "Elements" tab.

You can copy a recipe data record to another recipe, if the other recipe contains the same number of recipe elements. If the data types differ, the value will be copied to the target data record but it is assigned an error flag.

"Tags" editor

You can drag-and-drop a tag to a recipe element in the "Tag" column. The tag is linked to the recipe element. If a tag is already linked, an error message will be generated.

"Screens" editor

If you drag-and-drop a recipe to a screen, a new recipe display will be created and linked to the recipe.

Copying objects with linked objects (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

An object is linked to another object in the following situations, for example:

- You specify a tag for an alarm as a trigger tag.
  The alarm is the object. The tag is the linked object.
- You specify a connection for an external tag.
  The tag is the object. The connection is the linked object.

The object is always fully inserted during copying and pasting. Whether or not the linked object is pasted depends on the command used to insert it.

Simple pasting

The linked object is not copied. The linked object is transferred and handled as follows in the target HMI device:

- If an object with the same name exists, the existing object with its settings is used.
- If no object with the same name exists, the name of the object will be displayed. The object becomes invalid.

For some objects, linked objects are pasted automatically during simple pasting.

Linked objects copied automatically (Basic Panels)

Copying linked objects

The following table shows the objects for which linked objects are pasted automatically in simple pasting.

Object	Linked object
Screen	Template
Symbolic I/O field	Text list
Graphic I/O field	Graphics list
Graphic view	Graphic
Tag	Alarm
	Cycle
Recipe element	Text list
Scheduler	Triggers

Drag & drop from the details view (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You can improve configuration efficiency with just a few simple measures. Below are a few examples of efficient configuration.

Pasting objects to a screen from the details view

You can drag objects in the details view from various different editors to other editors.

Pasting a symbolic I/O field

1. Open a screen.
2. Click on the "Text and graphics lists" editor in the project tree. All existing text and graphics lists will be shown in the details view.
3. Click on a text list, for example, "Textlist1" in the Details view.
4. Drag-and-drop a text list from the Details view to a screen. A symbolic I/O field has been created and connected to the text list "Textlist1".

Pasting a graphic I/O field

1. Open a screen.
2. Click on the "Text and graphics lists" editor in the project tree. All existing text and graphics lists will be shown in the details view.

3. Click on a graphics list in the Details view, for example "Graficlist1".
4. Drag-and-drop a graphics list from the Details view to a screen. A graphic I/O field has been created and connected to the graphics list "Graficlist1".

Pasting an I/O field

1. Open a screen.
2. Click on the "HMI tags" editor in the project tree. All existing HMI tags will be shown in the Details view.
3. Click on an HMI tag in the Details view, for example "Tag1".
4. Drag-and-drop the HMI tag from the Details view to a screen. An I/O field has been created and connected to the HMI tag "Tag1".

12.11.6.3 Copying between different RT and ES versions (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

You can copy and paste project data such as screens, objects or tags between projects with different WinCC versions.

All configurations that are supported in the target version are retained when you copy data between projects of different WinCC versions. Configurations that are not supported by the target version are marked as invalid with a color code.

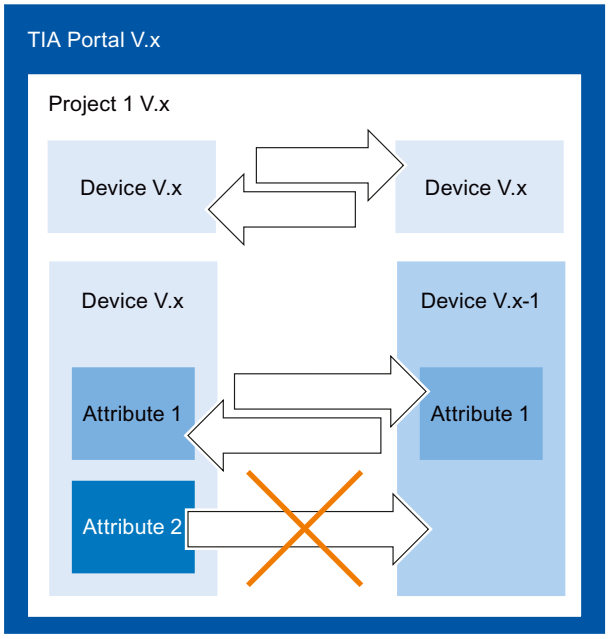
Copy of control between HMI devices from different device versions

WinCC supports all configurations of a previous WinCC version.

The following applies when copying within different ES versions:

- All the configurations that are also supported in the respective RT version are retained.
- Default settings are defined for configurations that are only supported in the WinCC version of the target project.

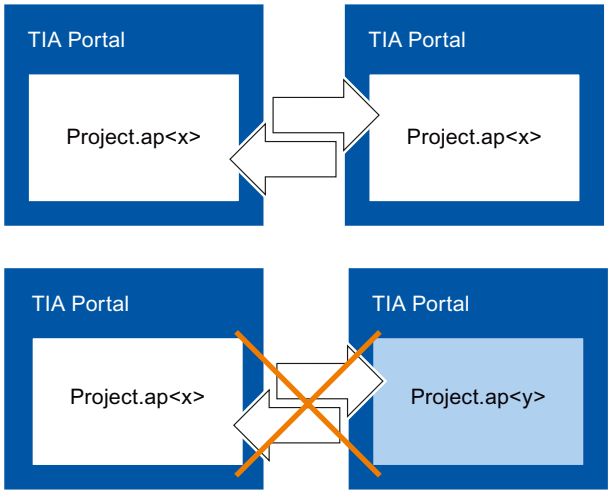
- Configurations that are not supported by the respective RT version are marked by color as invalid or are not displayed.
  All properties and settings originally defined in the source HMI device are reactivated when you copy an object back to the source HMI device unchanged.



- The HMI device must be valid for the current Runtime version.

Copying between different ES versions

To copy between two TIA projects, open a second instance of the TIA Portal. You can only copy between projects that have the same ES version. The ES version of a project is indicated by the file extension *.ap<version_number>.



12.11.7 Using WinCC version compatibility

12.11.7.1 Basics on version compatibility (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

Edit existing projects as follows with WinCC:

- You edit, compile and download existing projects with the range of functions of the previous version of WinCC. You can continue to edit these projects afterwards with the previous version of WinCC.
- You upgrade existing projects and use the functions of the current WinCC version.

Note

WinCC functions

While editing a project of a previous WinCC version, you can only access the functions and HMI devices of this previous WinCC version.

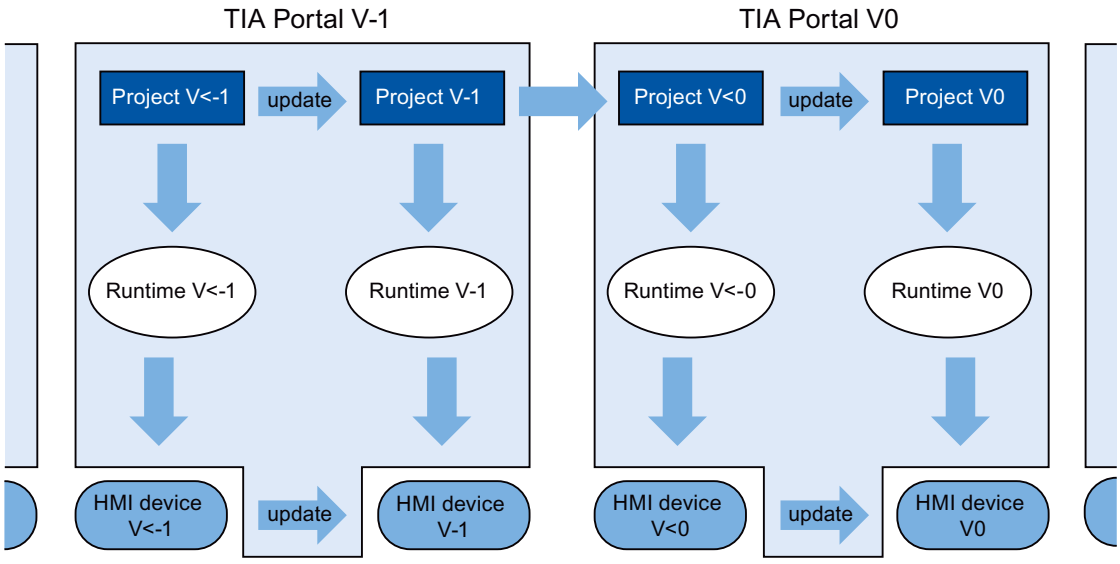
Versions in WinCC

In WinCC, you work with different version types:

- WinCC version
  The WinCC version installed on the configuration PC, for example, WinCC V12.
- Project version
  Projects are created using the WinCC version that is installed on the configuration PC. While you are editing a WinCC project from a previous version in the current WinCC version, the version ID is displayed behind the project name in the project tree.
- Runtime version
  You can configure HMI devices with different runtime versions in WinCC. You specify the runtime version once for an HMI device. The device version must match the Runtime version.
- Device versions
  Depending on the used HMI device, the image is a combination of the operating system and / or runtime software. For each HMI device, WinCC provides various images which can be loaded onto the HMI device, if necessary, according to the configuration. The device version corresponds to a specific image. The device version must match the configuration.

Compatibility of WinCC versions, Runtime versions and device versions

The figure below shows the interaction of the versions in the TIA Portal:



Creating projects

After you create a new project in WinCC, open and edit it in the WinCC version in which you created it.

Saving

To save a project of a predecessor WinCC version in this version again, save it in the usual way. If you manually upgrade the project to your WinCC version, please note that you will no longer be able to open the project in the previous WinCC version.

To save a project of the previous WinCC version in the current version, upgrade the project to your WinCC version. You can then no longer edit the project with a previous version of WinCC.

Compiling, simulating and loading

If you are using a project of a previous WinCC version, you can use your current version to generate Runtime data for this previous version. This also allows you to load HMI devices that are no longer compatible with your WinCC version.

Copying within projects with different WinCC versions

If objects and configurations are also available in the target version, copy these as required via the clipboard or using drag-and-drop.

Opening, editing and saving projects of a previous WinCC version

You can open and edit projects of previous WinCC versions as required. In doing so, you only utilize the functions of the previous WinCC version. Once you have completed editing, you can once again save and edit the project in the previous WinCC version.

Compiling, downloading and simulating projects of a previous WinCC version

You can compile, download and simulate projects of previous WinCC versions as required. Your current WinCC version will provide the Runtimes and device versions for the corresponding WinCC version.

12.11.7.2 Editing projects of a previous WinCC version (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

WinCC provides the option of editing projects of a previous WinCC version. While editing a project of a previous WinCC version, you can only access the functions of this version. In order to use the functions of your current WinCC version for this project, upgrade the project to your WinCC version.

Note

If you upgrade a project to your WinCC version, please note that you will no longer be able to open and edit the project in the previous WinCC version.

Requirement

- A project of a previous WinCC version has been created.
- The current WinCC version is installed on the configuration PC.

Procedure

Proceed as follows to edit a project of a previous WinCC version:

1. Open the project.
2. Edit the project using the functions of the previous WinCC version.
3. Save the project.
4. Compile the project.
5. Download and simulate the project.
6. You can open the project in the previous WinCC version for further editing, if required.

Result

The modified project data can be edited further on a different configuration PC with the previous WinCC version for further processing. The Runtime project was generated and downloaded in the corresponding Runtime version.

12.11.7.3 Upgrading projects (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

If the project version is older than the WinCC version, the version ID is displayed in the project navigation. Your WinCC version also contains the previous version that you can use to edit projects as required. In order to use the functions and options of your WinCC version in a project, upgrade the project to your WinCC version. Then change your device or Runtime version(s) to match the new project version.

Note

WinCC version compatibility

If you upgrade a project to your WinCC version, please note that you will no longer be able to edit the project with the previous version.

Requirements

- The project version is a predecessor of your WinCC version.
- You have write access to your project drive.
- The project drive provides sufficient storage capacity for another project of this size.

Procedure

Proceed as follows to upgrade a project to your WinCC version:

1. Select the project in the project navigation.
2. Select the "Upgrade project" command from the shortcut menu of the project.
   A dialog opens.
3. Click "Confirm".
   The project closes and the progress bar is displayed.
   An alarm is output when the project has been upgraded.
4. Then compile each device in the project with the command "Edit > Compile".

Note

Sequence of compilation

Compile the controllers first, and then the HMI devices in your project. In this way, you ensure that the controller data required to compile the HMI devices is available.

Result

- The project has been saved on the project drive in the previous WinCC version and with the corresponding file extension.
- The project is displayed on the project drive in the current WinCC version and with the corresponding file extension.
- The project is displayed without a WinCC version ID in the project navigation.

To make use of the functions of the new WinCC version, change your device or Runtime version(s) in accordance with the new project version in the next step.

12.11.7.4 Upgrading a global library (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

In order to process objects of a global library in a project, the global library must be available in the product version of the project. You can upgrade each global library from an older product version to the current product version. You are prompted accordingly when you open the global library.

Requirements

- The version of the global library is a predecessor of your WinCC version.
- You have write access to your project drive.
- All types in the library have been released.

Note

Upgrading a user library

If you want to use a user library from an earlier version of WinCC, you must upgrade it. Make sure that all types of the library have been released. The library being upgraded cannot contain a type with the "in progress" status.

Procedure

Proceed as follows to upgrade a global library from TIA Portal V12.x or lower:

1. Open the global library.
   The "Upgrade global library" dialog box opens.
2. Click "OK".

A copy of the global library is created and upgraded. The copy of the global library receives the name extension "_V13". The global library opens.

Result

The global library is stored with the appropriate file extension.

12.11.7.5 Changing between device versions (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Selection of the device version

When you configure a new HMI device, WinCC automatically selects the latest version of the device.

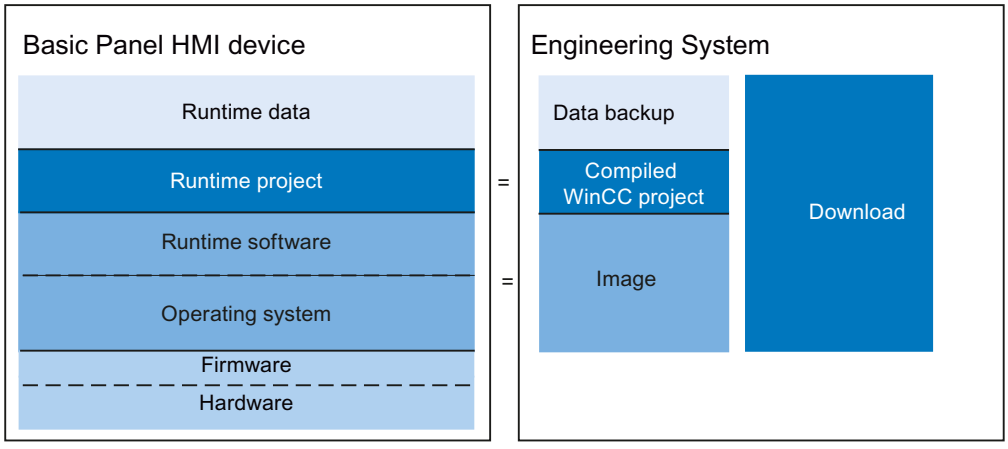
If you want to use a device version other than the one set in WinCC, transfer an image to the HMI device. WinCC provides the images required for the supported HMI devices.

Information on the device versions used in WinCC is available in the FAQs on the Internet, entry ID 21742389.

NOTICE
Changing the device version deletes all data on the HMI device. Data is deleted on the target system if you change the device version. For this reason, you should save existing Runtime data before changing the device version.

HMI device configuration

The following figure shows the software components of an HMI device:



12.11.7.6 Changing the device version (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

Depending on the required Runtime version, select the suitable device version for your configuration.

Note

Selection of device versions

The selection of available devices versions depends on the version of the project.

Requirements

- A project has been created and opened.
- The project contains an HMI device.

Procedure

To change the device version, follow these steps:

1. Double-click on "Devices & Networks" in the project tree.
   The editor opens.
2. Select the required HMI device from the device view.
3. Select "Change Device/Version" in the device shortcut menu of the HMI device.
   A dialog opens.
4. Select the required HMI device.
5. Select the required device version under "Version".
6. Confirm your selection with "OK".

Result

You have changed the device version in the WinCC project.

NOTICE
Changing the device version deletes all data on the HMI device. All data is deleted from the HMI device when you change the device version and compile/ download the project. You should therefore backup your Runtime data prior to the download.

12.11.8 Viewing memory card data

12.11.8.1 Basics (Basic Panels, Panels, Comfort Panels)

Introduction

WinCC provides you with the possibility of viewing data stored on your memory card. The function supports the use of memory cards of the HMI device and of the CPU.

You have the following options:

AUTOHOTSPOT

AUTOHOTSPOT

AUTOHOTSPOT

AUTOHOTSPOT

AUTOHOTSPOT

12.11.8.2 Working with backups

Viewing a backup (Basic Panels)

Introduction

The backup of a Basic Panel that is stored on a memory card can also be viewed in the TIA Portal.

Requirements

- WinCC is installed.
- A memory card with a backup is available.
- The card reader is connected to the configuration PC.
- The project view is open.

Backup on the memory card in the card reader

1. Insert the memory card into the card reader.
2. Open "SIMATIC Card Reader" in the project navigation.
3. Select the card reader drive.
   The "Online Card Data" folder is displayed.
4. Open the "Online Card Data" folder
5. Click the backup to open the shortcut menu.
6. Select "Properties".

Backup on the memory card of the PLC

Proceed as follows if the backup is stored on the memory card of the PLC:

1. Connect the PLC with the configuration PC.
2. Click on the PLC in the project navigation.
3. Select "Connect online" from the shortcut menu.
   A connection to the PLC is established.
   Once the PLC is connected, the "Online Card Data" folder is displayed.
4. Open the "Online Card Data" folder.

   Note

   Accessing a password-protected PLC

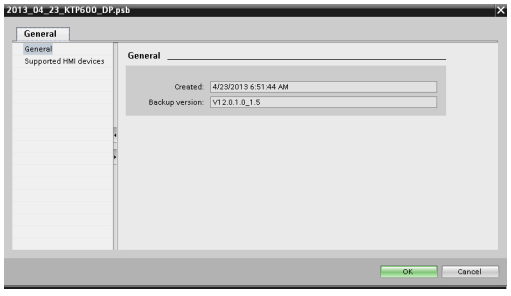
   When you attempt to access a PLC that is protected by a password, you will be prompted to enter the password.

   You need at least read access rights in order to view the data that is stored on the memory card.

5. Click the backup to open the shortcut menu.
6. Select "Properties".

Result

The backup properties are displayed in a separate dialog.



Renaming and deleting backups (Basic Panels)

Introduction

You can rename and delete backups from a memory card in the project navigation of the TIA Portal.

Requirements

- WinCC is installed.
- The card reader is connected to the configuration PC.
  The PLC is connected online with the configuration PC.

- A memory card with a backup is available.
- The project view is open.
- The backup is displayed in the project navigation.

Note

Accessing a password-protected PLC

When you attempt to access a PLC that is protected by a password, you will be prompted to enter the password.

You need write access rights to rename or delete memory card data.

Procedure

1. Click on the backup in the project navigation.
2. Open the shortcut menu.
3. Select "Rename" to rename the file.
4. Enter a new name.
5. Select "Delete" to delete the file.

Result

The backup file is now renamed or deleted.

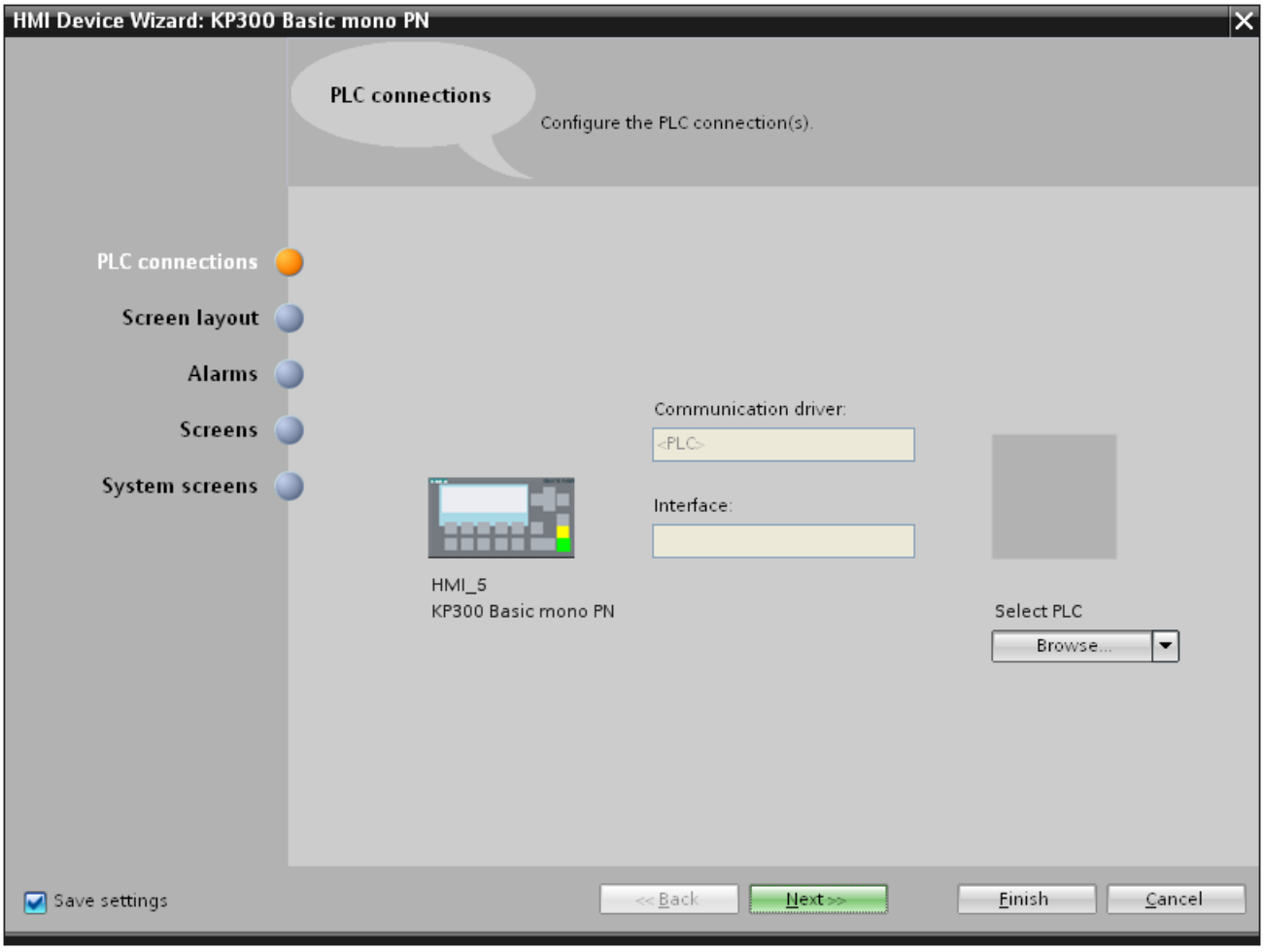
12.11.9 HMI device wizard basics (Basic Panels, Panels, Comfort Panels)

Introduction

The HMI device wizard will automatically start when you create a new HMI device in your project.

HMI device wizard

The HMI device wizard will guide you through each dialog step by step and help you set up a device. You use the HMI device wizard to specify the basic settings for your HMI device, such as screen layout and the connection to your PLC.



Note

When you create a device with a color display using the HMI device wizard, the graphics of the navigation buttons may be displayed in black and white. This error only occurs, however, if the new device is created with the same name as a device with a monochrome display which has been deleted in the meantime.

You can avoid this error by always deleting the associated graphics in the Graphics collection whenever you delete a device from the project.

Note

Multiple languages in the HMI device wizard

If you create multilingual screen navigation with the HMI device wizard, WinCC uses the user interface language for user-defined labels in all project languages.

If the character set of the user interface language is not supported in a project language, these names will not appear in this project language, for example, with a Chinese interface language and English project language.

Note

Interface type of the connection

If possible, the HMI device wizard makes a connection between the HMI device and controller with the default interface type of the HMI device.

After configuration with the HMI device wizard, check the connection in the "Devices & Networks" editor.

12.11.10 Managing colors centrally

12.11.10.1 Basic principles for color management

Introduction

WinCC provides the option of changing the colors used in a project centrally. The "Change color reference" dialog box contains a hierarchical overview of all color-relevant properties contained in the selected object. In the display, you can navigate within the display and operating objects that are shown and thus get an overview of all colors used. Using search and filter you can specify a color selection and replace it with other colors, if necessary.

Supported objects

You have access to all colors used and configured in the project with the "Change color reference" dialog. Excluded from this are colors that are used as follows:

- In types and instances from a library
- In faceplates
- In scripts
- In designs
- In screens with write protection

12.11.10.2 Finding and replacing colors

Introduction

The scope of the objects displayed in the "Change color references" dialog depends on the location at which the dialog is called.

- If you select an HMI device and call the dialog, all color references used on the HMI device are displayed.
- If you select a display object within a screen and call the dialog, only those color references that are included in the display object are displayed.

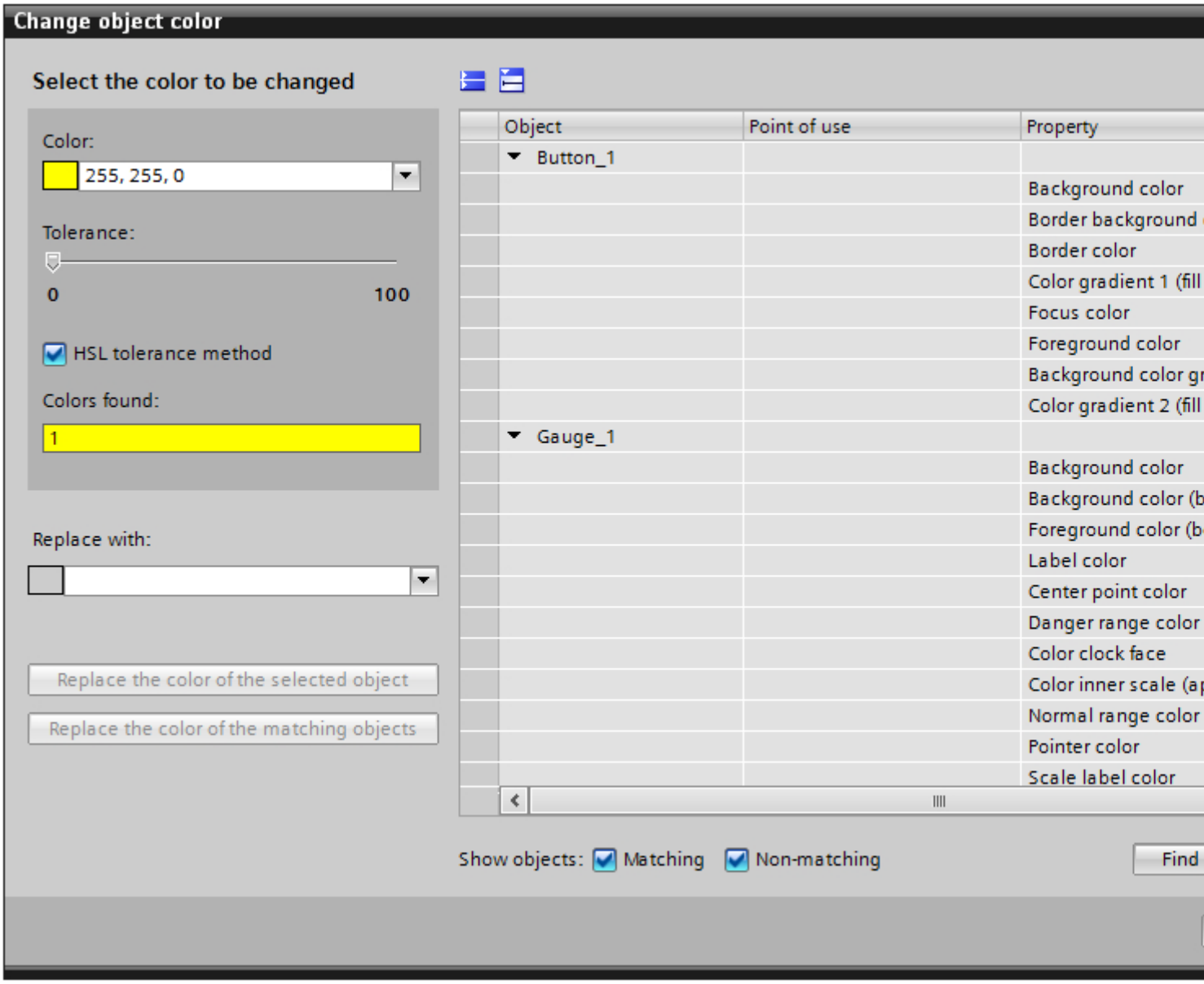
Requirement

- You have created a project.
- Screens have been created.

Procedure

1. Select the object that contains the required color references.
2. Select "Change object color" in the shortcut menu.
   A dialog opens.

3. Select the color that you want to change.
   - Click the color field in the search box.
     The project color selection opens.
   - To select a standard color or a user-defined color, click "More colors".
   - To use a color directly from the selected object, drag a color field from the overview table to the search box.
   - To select similar colors as well, set a tolerance.



4. Filter the displayed table.
5. In the table column "Replace with" select the new color for the individual properties.
6. Click "OK".

Result

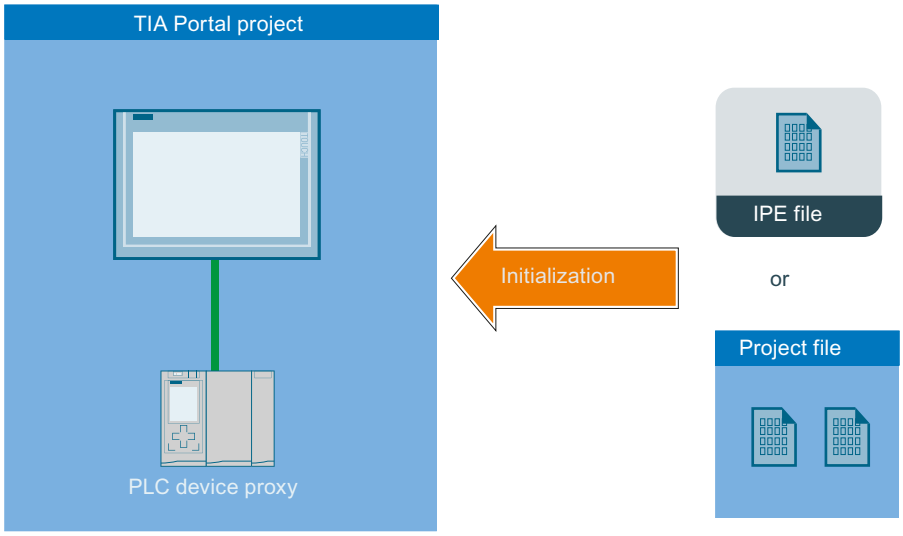
The new color references are configured in the selected object.

12.11.11 Exchanging controller data from other projects

Inter-project engineering

You can use the IPE (Inter Project Engineering) function to read in controller data from other TIA Portal projects and use it for configuring.

IPE enables you to transfer and merge the HMI configuration and PLC programming communication into various TIA Portal projects.



You can find more information and notes on configuration under: Basics of Inter Project Engineering (IPE)

See also

Using controller data from other projects in an HMI device (Page 6655)

12.12 Compiling and loading

12.12.1 Establishing a connection to the HMI device

Introduction

To download a WinCC project to an HMI device, a properly configured connection must be set up between the configuration PC and HMI device. The connection cannot be set up, the download is cancelled.

Setting up a connection between the configuration PC and HMI device

1. Check the cable connection between the HMI device and configuration PC.
2. Open the "Devices & Networks" editor in WinCC and start the network view.
3. Select the subnet in the network view and check the settings for the subnet.
4. Select the interface of the HMI device in the network view or device view and check the connection parameters in the Inspector window.
5. Switch on the HMI device and press the "Control Panel" button in the loader.
   The Control Panel opens.
6. Press "Transfer" twice in the Control Panel.
   The "Transfer Settings" dialog box opens.
7. Check the settings and then press "Advanced".
   The [Protocol*] Settings" dialog opens.
   *: The title of the dialog depends on the protocol used, for example, "PROFIBUS Settings".
8. Check the advanced settings and close the dialog with "OK".

Important settings

Check the connection settings and in particular the following parameters:

- Network and station addresses
- Selected transmission rate
- Master on the bus; as a general rule, only one master is permitted.

Note

Connections over Ethernet

For Ethernet-based communication, check the required network settings. You can find additional information under "Readme > WinCC > Security information".

If using a configurable adapter for the connection, check the adapter settings, for example, transmission rate, master on the bus.

12.12.2 Runtime settings

12.12.2.1 Runtime settings

Introduction

To edit the Runtime settings for your HMI device, select "Runtime settings" under your HMI device in the project tree.

Note
Device dependency

The options available for selection in the Runtime settings depend on the HMI device.

Overview

You configure options from different areas of your project in the runtime settings:

- Display in Runtime
  - Start screen
  - Default template
  - Standard style of the project and style of the HMI device
  - Screen resolution
  - Full-screen mode
  - Graphic browser
  - Bit selection for color and flashing
  - Bit selection for text and graphic lists
- Project ID for compatibility check with the controller
- Logging language
- Remote control via Sm@rt server in the "Services" area
  You can find additional information on this in the WinCC online help under "Options > Sm@rt Options"
- Operating in Runtime
  - Task switching
  - Load additional information for debugging scripts: Object names and comments in scripts
  - Show limits as a tooltip
  - On-screen keyboard
  - Disable function keys
  - Release buttons when you leave the button
  - Keyboard layout and keyboard shortcuts

- GMP-compliant configuration
  You can find additional information on this in the WinCC online help under "Options > WinCC Audit".
- Alarms
  - Buffer overflow
  - Text of the acknowledgment group
  - Log
  - Colors of the alarm classes
  - Configuration of the system events
- User administration
  - Configuration of the password and logoff time
  - Limit for logon attempts
  - Hierarchy levels
  - Configuration of SIMATIC Logon
- Language and font
  - Available Runtime languages
  - Order of language switching
  - Default font
  - Download font with valid license
- Configuration of the OPC Unified Architecture Server
- Settings for synchronization of the controller and HMI tags

See also

Compatibility check with the controller (Page 5911)
Configuring display in Runtime (Page 5909)
Configuring operation in Runtime (Page 5910)
Creating alarm classes (Page 5494)
Settings for the user administration (Page 5617)
Settings for synchronization (Page 5440)
Log basics (Page 5561)
Configuring alarm buffer overflow (Advanced) (Page 5521)
Editing system events (Basic, Advanced) (Page 5502)
Managing languages (Page 5851)

12.12.2.2 Configuring display in Runtime (Comfort Panels)

Introduction

In the Runtime settings you specify how screens, alarms and language are displayed.

To edit the Runtime settings for your HMI device, select "Runtime settings" under your HMI device in the project tree. The options available for selection depend on the HMI device.

Display on the target system

In WinCC, configure the visual representation of the generated project in Runtime:

- Select the full-screen or window mode for the project. The window has a smaller size than the screen. In full-screen mode, the project is zoomed to the full screen. There will be no window and operator controls for this view.
  Click "General" in the "Settings" editor to activate the full-screen display at startup. Activate the "Full-screen mode" check box in the "Screen" field.

  Note

  Truncated view

  If the HMI screen does not match the configured size (in pixels), the project only appears on part of the screen in full-screen mode.

- Under "General > Screen", you specify whether the HMI device uses the default style of the project or another pre-defined style.

Screen settings

In the Runtime settings, specify the start screen and default template. You can find additional information in the online help for WinCC under "Working with screens".

Language settings

- Logging language
  Under "General > Logs", you specify the language in which you want to write the logs. The selected logging language always remains the same, even if the user switches the language in Runtime.
  If you select "Startup language", the logging language remains the same until Runtime is stopped, even if the user has switched the language. However, the next time Runtime is started, the last configured Runtime language is used as the logging language.
- Runtime language
  Under "Language & font", you specify all Runtime languages and their display. Text entries may be truncated if you selected an unsuitable font size or style. This setting can, for example, affect the following texts:
  - Tooltips
  - Long alarm text
  - Text in dialogs

Setting for specific screen objects

- Specify bit selection
  Under "Screens > Bit selection", you specify whether bit selection will be used on this HMI device for the following:
  - Text and graphics (text lists and graphic lists)
  - Color and flashing (appearance analysis)
    When you enable an option, the selection that is configured in the set bit with the least significance is displayed:
    When you disable an option and several bits are set, the selection that has been configured only for the set bit is displayed.
- Tooltips for I/O fields
  Under "Screens > Display", you specify whether configured limit values are displayed as tooltips in Runtime when values are entered in I/O fields.
- Colored objects
  You specify the color depth of your HMI device under "General > Screens > Color depth". As needed, you assign screen objects a colored appearance. The range of colors is determined by the color depth supported on your selected HMI device.

12.12.2.3 Configuring operation in Runtime

Introduction

In the Runtime settings you specify how screens are operated, and how you use keys for operation in Runtime.

To edit the Runtime settings for your HMI device, select "Runtime settings" under your HMI device in the project tree. The options available for selection depend on the HMI device.

Locking task switching

Depending on the HMI device, you can lock task switching on the HMI device. This will prevent the operator from opening other applications in Runtime.

In the Runtime settings of the HMI device, enable the "Lock task switching" option under "General > Screen".

Note

Stop Runtime

If you lock program switching, always configure in your project the system function "StopRuntime" for an object, e.g. a softkey or button.

Settings for data transfer to the HMI device

- Transferring names of the screen objects in scripts
  If you enable the "Load names" option under "General > Screen", the object names of the screen objects are transferred instead of coded addressing information. You need the object names, if you want to address the screen objects in a script via the object names. When you test a script in the debugger, the code becomes more comprehensible by displaying the object names.
- Displaying comments in scripts
  Under "Screens > Display", you specify whether the comments in scripts are also transferred to the HMI device.
  If you do not transfer the comments, you save storage space on your HMI device.
  If you transfer the comments, the script is more easily understood during testing in the debugger.

Operation using function keys

To symbolize the function of a function key for keyboard devices, you configure a graphic next to the function key in the display. Under "Screens > Function keys > User-defined pictogram size", you specify whether you want to use a graphic size that deviates from the standard size.

Operation with keyboard

Under "Keyboard > General > Use screen keyboard", you specify whether the screen keyboard is available on your HMI device.

Under "Keyboard > General > Release button on exit", you specify whether the "Release" event is triggered when the user exits a button without releasing it.

Under "Keyboard > Disable function keys in dialogs", you specify for keyboard HMI devices whether the function keys are disabled for the duration of the displayed dialog. Large dialogs can hide explanatory areas or graphics at function keys or that border function keys. Functions keys can be unintentionally pressed due to this.

12.12.2.4 Compatibility check with the controller (Panels, Multipanels, Comfort Panels, Mobile Panels, RT Advanced)

Introduction

The "Project ID" area pointer is used to configure the project ID for checking the consistency of the project with the PLC. The area pointer reads the project version from the PLC. You can find additional information in the online help for WinCC under "Communicating with controllers".

Compatibility check with the controller

You select the project ID in the Runtime settings of the HMI device under "General > Identification > Project ID".

At startup, Runtime checks whether the project ID of the project matches the PLC project version. Runtime will only start if the two values match.

An area pointer "Project ID" is created for connections whose address you specify in the PLC. The value that you store at this address will become the project ID.

The project ID is not available on all HMI devices.

12.12.2.5 Password protection in runtime

Password protection in runtime (Basic Panels)

Overview

Connection establishment in the TIA Portal can be password-protected. The configured passwords are saved in encrypted form. When transferred to the target system, the passwords are also protected from unauthorized access and cannot be read. The passwords are securely decrypted on the HMI devices in runtime.

Password protection can be set up for the following connections:

- SIMATIC S7-1200
- SIMATIC S7-1500
- SIMATIC S7-1500 software controller (WinAC)
- PC Station 2.0 (station manager)

Note

The passwords become invalid if you switch to an older version.

Downloading certificates

The certificate for decrypting the passwords is saved in the certificate store of the HMI device. A system alarm is output if the certificate is not available.

You can restore the certificate by downloading the configuration again.

See also

Configuring a password for transport (Page 5912)

Configuring a password for transport (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

To protect the private password key during transport, you can configure a password for transport under "Runtime settings".

Creating a password

1. Open the "Runtime settings > General" editor in the Project window.
2. Enter the password in the "Enter password" box in the "Transport password" area.
3. Confirm the password in the "Confirm password" box.

Note

If there was no previous password, the "Old password" box is disabled.

Result

You have now configured the password for transport. If a password for transport has been configured and you transfer the project to the HMI device, a dialog with a password prompt will appear once in runtime.

The dialog with the password prompt also appears if the password has been changed. If the password entered is correct, the certificate is saved in the certificate store of the HMI device.

See also

Password protection in runtime (Page 5912)

Resetting the password for transport (Page 5913)

Resetting the password for transport (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

If you have moved the transport password, for example, you can create a new security certificate. To do so, reset the transport password.

Resetting the password

1. Click the "Reset password" button.
   The password is reset. A new certificate without a password is generated.

   Note

   Other passwords, for example the password for the PLC, will become invalid. The next time you compile, an error message will appear telling you that these passwords are no longer valid.

2. Enter the new password and confirm it in the "Confirm password" field.

   Note

   The certificate is not password-protected until you enter a new password for transport.

See also

Configuring a password for transport (Page 5912)

12.12.3 Compiling and loading projects

12.12.3.1 Overview of compiling and loading projects

Overview

The project is compiled in the background even as you are configuring it in WinCC. This reduces the time for final compilation. When you start compilation, you create a file that can be run on the corresponding HMI device.

If an error occurs during compilation, WinCC provides support in locating and correcting it.

Once you have corrected any problems, you download the compiled project to the HMI devices on which the project is to run.

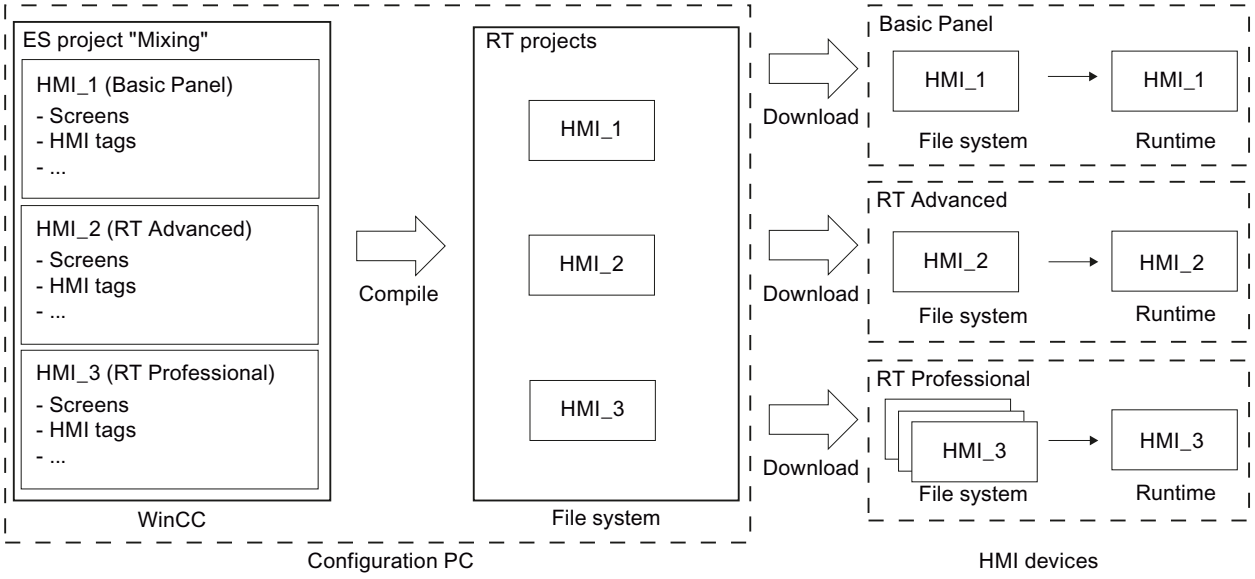
If you are using HMI tags in your project that are connected to PLC tags, you should also compile all modified S7 blocks with the command "Compile > Software" in the shortcut menu before you compile the HMI device.

Definition of terms

The term "project" has two different meanings in the contexts of compilation and loading. "Project" is the WinCC project on the configuration PC. "Project" is also the Runtime project you create by compiling the configuration data of an HMI device and download to the HMI device.

- WinCC project: contains the configuration data of one or more HMI devices
- Runtime project: contains the compiled configuration data of an HMI device

The figure below illustrates the link between WinCC projects and Runtime projects using the example of the "Compile and load" process:



Runtime version

The Runtime version depends on the image of the configured HMI device. The Runtime version of the compiled project is displayed under "Info" in the Inspector window.

12.12.3.2 Compiling a project (Basic Panels, Panels, Comfort Panels, RT Advanced, RT Professional)

Introduction

The changes made to the project are compiled in the background even as you are configuring a project in WinCC. Projects are compiled automatically when you load them. This ensures that the latest version of the project is loaded at all times.

WinCC checks consistency of the project during compilation. The error locations in the project are listed in the Inspector window. You can jump directly to the source of the error from the entry in the Inspector window. Check and correct errors found.

Scope of the compilation

Configuration data is compiled in the background as soon as you start configuring an HMI device. If you compile a project manually, only the changes in the configuration made since the last compilation process are compiled in the background.

You can start complete project compilation manually at any time; this may, for example, be done to test the consistency of the configured data.

Requirement

- A project is open.

Procedure

Proceed as follows to compile a project:

1. If you want to compile several HMI devices at the same time, select all the relevant HMI devices with multiple selection in the project tree.
2. Compile the project:
   - To only compile changes in the project, select the "Compile > Software (only changes)" command from the shortcut menu of the HMI device.
   - To compile all project data, select the "Compile > Software (compile all)" command from the shortcut menu.

Result

The configuration data of all selected HMI devices is compiled. Any errors that occur during compilation are shown in the Inspector window.

12.12.3.3 Loading projects

Overview for loading of projects

Overview

Delta data of the project is automatically compiled before you download it to one or several HMI devices. This always ensures that the latest version of the project is transferred.

Loading a project to an HMI device

The following steps are completed prior to downloading:

1. The download settings are verified. The "Extended loading" dialog box is opened automatically during the initial download of a project to an HMI device. You use this dialog to define the protocol and interface or destination path for the project in accordance with the HMI device Runtime used.
   You can open the "Extended download" dialog at any time with the menu command "Online > Extended download to device...".
   The "Load preview" dialog opens.
2. The project is compiled. Warnings and errors during compilation are displayed in the Inspector window and in the "Load preview" dialog.
3. The "Load preview" dialog shows you the following information for each HMI device:
   - The individual steps for loading
   - If the image of the target HMI device does not match the image from the configuration, a prompt is displayed asking whether you want to now change the image.

NOTICE
Changing the image deletes all data from the HMI device. Data is deleted on the target system if you change the image. For this reason, first back up the following data, if necessary: • User administration • Recipes

   - Presettings that take effect at loading. You can change the default settings for this download process, if necessary.
   - Occurring warnings (optional).
     You can download a project while ignoring the "warnings". The functionality may be restricted in runtime.
   - Occurring errors (optional).
     You cannot load the project.
     WinCC will open the invalid configuration in the corresponding editor if you double-click the error message in the Inspector window. Correct the errors and reload the project.

If you are using HMI tags in your project that are connected to PLC tags, you should also compile the user program before you compile the HMI device.

Loading with S7 routing

Configure the S7 routing settings in the "Devices & Networks" editor in the relevant controller. The settings depend on the device configured.

S7 routing supports the following protocols:

- MPI/PROFIBUS
- Ethernet

Loading a project without a connected HMI device

If you cannot establish a direct connection from the configuration PC to the HMI device, copy the compiled project to an external storage medium, such as a USB stick, and then load it onto the HMI device.

This function is available in connection with a HMI device image that is compatible with TIA Portal version V14 or higher.

Generate the required project data in WinCC by configuring the HMI device and then dragging the HMI device folder (such as "HMI_1 [<*Device type*>]") to an external storage medium under "Card Reader/USB memory".

Transferring Runtime add-ons

Projects can contain Runtime add-ons in the form of controls or CSP (Communication Support Packages). These Runtime add-ons are automatically transferred with the project.

See also

Loading project from external storage medium (Page 5921)

Loading a project

Introduction

Before a project can run on an HMI device, you must first load it to the HMI device. During loading, you must most importantly specify whether existing data on the HMI device such as "user administration" and "recipe data" is to be overwritten.

If the HMI device supports PROFINET, the name of the HMI device entered in the project tree is used as the device name for the PROFINET communication. The name is written to the HMI device during download. If a device name for the PROFINET communication has already been entered in the HMI device, it will be overwritten.

As a general rule, only one project can be active in Runtime on an HMI device. An HMI device is generally configured to exit Runtime automatically when loading is started. If this is not the case, you will have to exit Runtime manually on the HMI device.

If the image of the target HMI device does not match the image from the configuration, a prompt is displayed asking whether you want to now change the image.

NOTICE
Changing the image deletes all data from the HMI device. Data is deleted on the target system if you change the image. For this reason, first back up the following data, if necessary: • User administration • Recipes

Controlling the transfer behavior on the HMI device

As a general rule, only one project can be active in Runtime on an HMI device. An HMI device is generally configured to exit Runtime automatically when loading is started. If this is not the case, you will have to exit Runtime manually on the HMI device.

You define how the HMI device reacts when the project is loaded in the "Start Center" under "Settings" on the HMI device:

Transfer mode	Effect
Off	The project cannot be loaded to the HMI device.
Manually	The project can only be loaded to the HMI device if the following requirements are met: • Runtime is not running • The HMI device is in "Transfer" mode.
Automatic	The project can always be loaded to the HMI device. If a transfer is started on the configuration PC and a project is in runtime on the HMI device, the running project is automatically closed. For Mobile Panels, this transfer mode is disabled for security reasons.

Note

Closing Runtime automatically

After the commissioning phase, disable the automatic transfer function to prevent the HMI device from switching inadvertently to transfer mode.

Transfer mode can trigger unwanted responses in the plant.

In order to restrict access to the transfer settings and thus avoid unauthorized changes, enter a password in the "Start Center".

Please refer to the documentation for the HMI device used for more detailed information on transfer settings.

Requirements

- You have created an HMI device in the project.
- The HMI device is connected to the configuration PC.
- The "Start Center" has been started on the HMI device.
- The protocol by which the project is loaded is set on the HMI device in the "Start Center" under "Settings".
- Transfer mode is set as "Automatically" or "Manually" in the HMI device.

Procedure

Proceed as follows to load a project:

1. To download a project simultaneously to several HMI devices, select the HMI devices by means of multiple selection in the project tree.
2. Select the "Download to device > Software" command from the shortcut menu of an HMI device.
3. If the "Extended loading" dialog is open, configure the "Settings for loading". Make sure that the "Settings for loading" correspond to the "Transfer settings in the HMI device".
   - Select the protocol used, for example, Ethernet or HTTP.
   - Configure the relevant interface parameters on the configuration PC.
   - Make any interface-specific or protocol-specific settings required in the HMI device.
   - Click "Download".

   You can open the "Extended download" dialog at any time with the menu command "Online > Extended download to device...".
   The "Load preview" dialog opens. The project is compiled at the same time. The result is displayed in the "Load preview" dialog.
4. Check the displayed presettings and change them as necessary.
5. Click "Download".

Result

The project is loaded to all selected HMI devices. Any existing project is replaced. The data for user administration and / or recipes is replaced in accordance with the settings in the "Load preview" dialog.

During the download, you can keep track of the files that are transferred.

If errors or warnings occur during the download, corresponding alarms are displayed under "Info > Load" in the Inspector window.

On completion of the successful download of the project, you can execute it on the HMI device.

Note

If the transfer is interrupted, WinCC automatically ensures that no data is lost and that existing data is deleted on the HMI device only after complete transmission.

See also

Backing up and restoring data of the HMI device (Page 5952)

Updating the HMI device operating system (Page 5954)

Error messages during loading of projects (Page 5956)

Adapting the project for another HMI device (Page 5957)

Establishing a connection to the HMI device (Page 5958)

Loading project from external storage medium (Basic Panels)

Introduction

If you cannot establish a direct connection from the configuration PC to the HMI device, copy the compiled project to an external storage medium, such as a USB stick, and load it from the USB stick onto the HMI device.

This function is available in connection with a HMI device image that is compatible with TIA Portal version V14 or higher.

This function is available for Basic Panels 2nd Generation.

NOTICE
Data loss If you load a project with the option "Upgrade or downgrade Firmware", the operating system of the HMI device is updated. Existing data on the HMI device including the HMI device password is deleted. Settings in the Start Center are retained, license keys are stored on the external storage medium before the operating system is updated. If required, back up the data before loading.

You create the required project data in WinCC by configuring the HMI device and then dragging and dropping or copying and pasting the folder of the HMI device (e.g. "HMI_1 [<device type>]") to an external storage medium under "Card Reader/USB memory".

Requirements

- You have started the Start Center on the HMI device.
- The storage medium with the saved project is inserted in the HMI device.

Procedure

1. Click "Settings".
2. Open the "Service & Commissioning" dialog.
3. Confirm with the "Project Download" button.
   With "Project Download", you have the option of downloading a single project from an external storage medium to the HMI device.
4. Select the storage medium on which the project data is saved.
5. Select the project file.

Note

If no storage medium is inserted in the HMI device, the "Projects" list is empty.

6. Select the options for loading the project:
   - "User administration:"
     Overwrite user administration on the HMI device with the user administration of the project
   - "Recipe data:"
     Overwrite HMI device recipes with project recipes
   - "Upgrade or downgrade Firmware:"
     Update firmware on the HMI device if the firmware version of the HMI device is not incompatible with the firmware version of the project

   Note

   You can only change the settings for overwriting the user management or recipes if they were not previously interlocked in the configuration.

7. Confirm with the "Accept" button.

Result

After the loading process, the new project is started on the HMI device.

See also

Overview for loading of projects (Page 5916)

12.12.3.4 Compiling and loading with multiuser engineering (Basic Panels)

Compiling and loading with multiuser engineering (overview) (Basic Panels)

Introduction

When using multiuser engineering for your projects, you should take into account the response when compiling the Runtime projects and the response when downloading them to HMI devices.

You can compile and download to an HMI device in both the server project view and in the local session.

You can find more information about the topic of "multiuser engineering" under "Using Multiuser Engineering".

Basics

For HMI devices and RT Advanced, the following scenarios are possible in multiuser engineering:

- Compiling in the server project view
- Compiling in the local session

- Loading from the server project view
- Loading from the local session

Note

Complete download from the server project view or local session is no different to complete download in a single-user project. With a complete download, the current Runtime project is loaded from the currently active view to an HMI device.

Note

Compiling and downloading in a local session is no different from compiling and downloading in a single-user project.

In principle, you can execute all commands for compiling and loading in multiuser engineering projects:

- "Software (rebuild all)"
- "Compile > Software (only changes)"
- Software (all)

The term "project"

The term "project" has two different meanings in the contexts of compilation and loading. "Project" is the WinCC project on the configuration PC. "Project" is also the Runtime project you create by compiling the configuration data of an HMI device and download to the HMI device.

- WinCC project: contains the configuration data of one or more HMI devices
- Runtime project: contains the compiled configuration data of one HMI device

Rules

The following basic rules apply to compiling and downloading in multiuser engineering:

- The Runtime project that has been compiled in a local session always remains local and is not uploaded to the multiuser server. It cannot be saved in the multiuser server project.
- Only Runtime projects compiled in the server project view can be saved in the multiuser server project.

See also

Compiling in the local session (Page 5924)

Compiling in the server project view (Page 5924)

Compiling in the server project view (Basic Panels)

Basics

Compiling and downloading in the server project view is no different from compiling and downloading in a single-user project.

During the compiling of a project in the server project view, the multiuser server project is blocked. Other users cannot make changes to this server project during this time. The Runtime project compiled in the server project view is stored along with the engineering project in the central multiuser server. Blocking the multiuser server project ensures that the configuration data and the Runtime project remain in sync.

Note

When you compile and save in the server project view, other users obtain the Runtime project you have updated along with the engineering project when they "refresh" their local session. Other users do not have to recompile the changes you have made after an update.

Example: Compiling during check-in

You make changes to a tag in a local session. All prior changes have been compiled in the associated server project.

If there are no compilation errors, both projects - the modified engineering project (with the modified tags) and the compiled Runtime project - are saved in the central multiuser server project with the "Save changes" command.

If you skip compiling during the check-in, the project contains the changes that have been saved on the server.

The next user who creates a local session from the server project or updates an existing local session must compile your two changes in addition to his or her own changes.

Note

Working on a shared project through multiple local sessions increases the probability of error. It is therefore recommended to compile the project at check-in and eliminate any errors that are reported during compiling. In this way, you provide the next user with a project free of errors.

See also

Compiling and loading with multiuser engineering (overview) (Page 5922)

Compiling in the local session (Basic Panels)

Basics

Compiling and downloading projects in the local session is no different from compiling and downloading in a single-user project.

Since the local session is a copy of the server project, the first compilation status of the local session is identical to that of the server project. If the server project contains contents that are not compiled or error messages occurred during the compiling, they are transferred to the local session.

Note

It is recommended to compile the project at check-in and eliminate any errors that are reported during compiling. In this way, you provide the next user with a project free of errors and avoid spreading errors.

Updating in the local session

If you update a project in the local session, the local session - including the compilation status - is completely replaced by the content of the server project. Only the changes marked for check-in are retained in the updated local session and generate additional compiling steps in the local session.

Example: Updating the local session

You make changes to a tag in a local session. All prior changes have been compiled in the associated server project.

You update the content of the local session by clicking the "Update" button. After the update, the local session obtains the compilation status of the server project. There are also compiling tasks for the acquisition of the modified tags.

See also

Compiling and loading with multiuser engineering (overview) (Page 5922)

12.12.3.5 Runtime start

Starting Runtime on the HMI device (Basic Panels)

Introduction

On completion of the project download to the HMI device, you can start the project in Runtime. The project is saved in the HMI device to a file with the following extension:

- Basic Panels as well as OP 73, OP 77A and TP 177A: "*.srt"

The project settings defined in the "Runtime settings" of the HMI device are activated when the project is started in Runtime.

The programs that you can use to start projects on the HMI device are available in the Runtime installation folder.

Note

Closing Runtime automatically

If automatic transfer is activated on the HMI device and a transfer is started on the configuration PC, the running project is automatically terminated.

The HMI device then automatically switches to "Transfer" operating mode.

After the commissioning phase, disable the automatic transfer function to prevent the HMI device from switching inadvertently to transfer mode.

Transfer mode can trigger unwanted responses in the plant.

In order to restrict access to the transfer settings and thus avoid unauthorized changes, enter a password in the "Start Center".

Requirements

- WinCC Runtime is installed on the HMI device.
- The project was downloaded to the HMI device.
- The "Start Center" has been started.

Procedure

The project is stored on a panel in a folder you specify in the HMI device transfer settings. The "Start Center" application is started on a panel. The project loaded is started automatically after expiration of the configured delay.

If the project does not start automatically:

1. Click on "Start" in the "Start Center" to start the loaded project.

Refer to the documentation for the HMI device for additional information on startup of projects.

12.12.4 Simulating projects

12.12.4.1 Simulation basics

Introduction

You can use the simulator to test the performance of your configuration on the configuration PC. This allows you to quickly locate any logical configuration errors before productive operation.

You can start the simulator as follows:

- In the shortcut menu of the HMI device, or in a screen: "Start simulation"
- Menu command "Online > Simulation > [Start|With tag simulator|With script debugger]"
- Under "Visualization > Simulate device" in the portal view.

Requirements

The simulation/runtime component is installed on the configuration PC.

Field of application

Use the simulator to test various functions of the operator control and monitoring system, such as:

- Checking limit levels and alarm outputs
- Consistency of interrupts
- Configured interrupt simulation
- Configured warnings
- Configured error messages
- Check of status displays
- Interconnection and screen layout

See also

Simulating a project (Page 5928)

12.12.4.2 Simulating a project (Panels, Comfort Panels, RT Advanced)

Introduction

You simulate your project with one of the following two methods:

- Without a connected PLC
  You change the value of area pointers and tags in a tag simulator that is read for the simulation of WinCC Runtime.
- With a connected PLC without a running process
  You simulate your project by running it directly in Runtime. The tags and area pointers become active. This allows you to create an authentic simulation of your configured HMI device in Runtime.

Note

Simulation restrictions

You cannot simulate the following system functions:

- CalibrateTouchScreen

You cannot simulate the Media Player. A static screen appears in the simulation window instead of the Media Player.

File access via scripts is not possible for HMI devices with Windows CE.

Requirement

- Simulation without a connected PLC: Tags have been created
- Simulation with a connected PLC but no active process: A project with tags and area pointers has been created

Procedure

To simulate a project using the tag simulator, follow these steps:

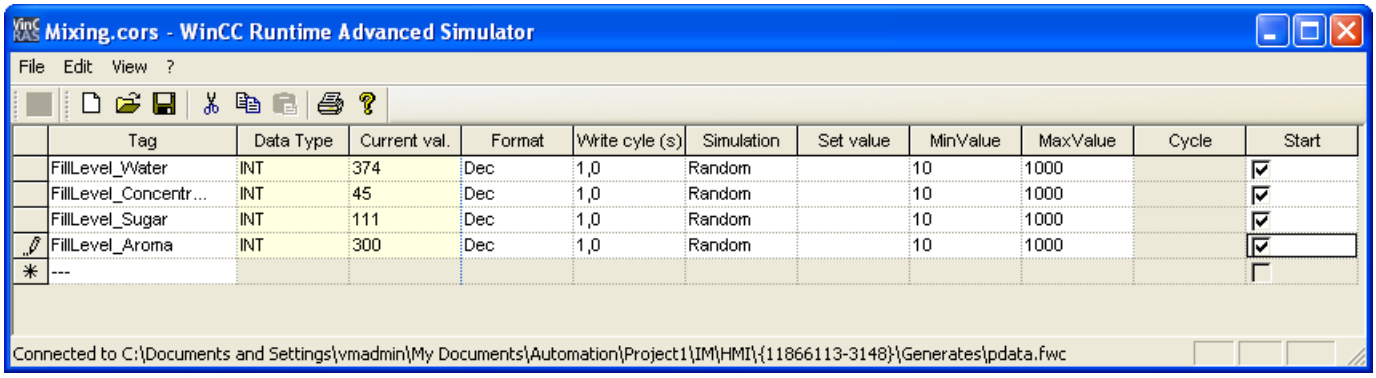
1. Open the project on the configuration PC.
2. Select the "Online > Simulation > With tag simulator" menu command.
   For initial project simulation, the simulator is started with a new, empty table. The project is opened simultaneously in Runtime.
   Toggle between the tag simulator and Runtime using the <Alt +Tab> key combination.
3. To simulate a process value, select the corresponding "tag" from the tag simulator.
   The table lists all configured tags. You can simulate up to 300 tags simultaneously.
4. Select the simulation mode in the "Simulation" column.
5. Change the value of tags and area pointers in the respective columns.
6. Activate the "Start" check box to start the simulation for this tag.
7. To save the simulation, select the menu command "File > Save" and enter a descriptive name, for example, "Mixing".
   The file name is assigned the extension "*.cors".

Result

The process values are simulated in Runtime. The tag values are created at random, or incremented, depending on the simulation mode.

To specify tag values, change the simulation mode to "<Display>" and enter a value at "Set value".

The following figure shows a tag simulator with four tags whose values can be determined at random in a range of values from 10 to 1000:



Managing simulation data

If you have saved data from a previous simulation, you can open the file at a later point in time and simulate your project again. The tags and area pointers listed in the tag simulator must still be available in the project.

Proceed as follows to open a simulation file:

1. Select the menu command "Online > Simulate Runtime > With tag simulator".
2. Select the menu command "File > Open" in the tag simulator.
3. Select the corresponding simulation file and click "Open".
   The simulator loads the stored data.

Enabling and disabling tags

Start and stop the simulation for each tag separately in order to facilitate the transition from offline to online engineering. Activate "Start" in the corresponding row.

If a tag is activated, the simulation values are calculated and transferred to the WinCC simulator.

Deleting a tag

To delete a tag from the tag simulator, follow these steps:

1. Select the cell that contains the tag name.
2. Select the "Edit > Cut" menu command.
   The tag is removed from the table.

See also

Simulation basics (Page 5926)

Working with the tag simulator (Page 5930)

12.12.4.3 Working with the tag simulator (Basic Panels, Panels, Comfort Panels, RT Advanced)

About the tag simulator

The tag simulator has the following columns:

Column	Description
Tag	Specifies the tags for the simulation.
Data type	Shows the data type of the selected tag.
Current value	Shows the simulated value of the defined tags.
Format	Specifies the selected format in which the tag values are simulated: • Decimal (1, 2, 3, 4, ...) • Hexadecimal (03CE, 01F3, ...) • Binary (0 and 1)
Write cycle	Specifies the selected time interval at which the current tag values are simulated. If you enter "2", for example, the current value of the tag will be shown every 2 seconds.
Simulation	Shows the method by which the tag values are processed during simulation.
Set value	Sets the selected value for the respective tag. The simulation start with the specified value.
minValue maxValue	Specifies the value range of the tag. You set a minimum and maximum value for this range. The default values are -32768 for the minimum and 32767 for the maximum.
Period	Contains the period during which the value of the tag is repeated for the "Increment" and "Decrement" simulation modes.
Start	Starts simulation of the tag based on the previously entered information.

Simulation modes

The simulator has six different simulation modes. The configured tags are supplied with nearly realistic values during the simulation.

Simulation mode	Description
Sinusoidal	Changes the tag value to form a sinusoidal curve. The value is visualized as a periodic, non-linear function.
Random	Provides randomly generated values. The tag value is changed by means of a random function.
Increment	Increases the value of the tag continuously up to a specified maximum value. Begins again at the minimum after the maximum has been reached. The value trend corresponds to a positive saw-tooth curve.
Decrement	Reduces the value of the tag continuously down to a specified minimum value. Begins again at the maximum after the minimum has been reached. The value curve corresponds to a negative saw-tooth curve.

Simulation mode	Description
Shift bit	Shifts a set bit continuously by one position. The previous position is always reset. This lets you test the alarms of an HMI device, for example.
<Display>	The current tag value is displayed statically.

Example: Simulate tags with the "Shift bit" simulation mode

Proceed as follows to simulate tags with the "Shift bit" simulation mode:

1. Open the project you want to simulate.
2. Select the menu command "Online > Simulate Runtime > With tag simulator". The tag simulator opens.
3. In the "Tag" column, select a tag from your project.
4. Select "Bin" in the "Format" column.
5. Enter the value "1" in the "Write cycle" column.
6. Select the "Shift bit" simulation mode in the "Simulation" column.
7. Enter the value "1" in the "Set value" column.
8. Enable the tag with the "Start" check box.

Result

The simulator tests the selected tag bit-by-bit as follows:

Simulation values	Byte for alarms
Set start value	00000001
1. Simulation value	00000010
2. Simulation value	00000100
3. Simulation value	00001000
....	...

In Runtime you see if the desired alarm is output at a given value.

See also

Simulating a project (Page 5928)

12.12.4.4 Simulation restrictions

Alarms with dynamic parameters

If you use tags or text lists as external tags for alarms, the dynamic parameters of the alarms are not displayed.

Only internal tags are enabled for the simulation of alarms in the tag simulator .

Use PLCSim to simulate dynamic parameters.

Comfort Panels

If you want to use PLCSim with version V14 or higher for simulation with a Comfort Panel, the device version of the Comfort Panel must be 14.0.0.0 or higher.

12.12.5 Reducing the project size (Basic Panels, Panels, Comfort Panels, RT Advanced)

Introduction

When loading a large-scale project to an HMI device, the memory of the HMI device may be insufficient for the project. There are several ways to reduce the size of your project.

Options for reducing the project size

There are several ways to reduce the size of the project and save space:

- Reduce the number of available Runtime languages
  Check whether all selected Runtime languages are actually needed. You can disable the languages that you do not need under "Runtime settings > Language & Font > Runtime language and font selection".
- Do not use help texts for S7 diagnostic alarms
  To reduce the size of the project, you can disable the download of help texts for the S7 diagnostic alarms. In order to avoid downloading the help texts to the HMI device, disable the option "Download S7 diagnostics help texts" under "Runtime settings > Alarms > General".
- Rebuild all software
  In order to optimize the project data and to clean up obsolete changes, compile the entire project using the "Compile > Software (rebuild all)" command from the shortcut menu of the HMI device.
- Harmonize the presentation using styles (as of WinCC V13)
  It is recommended to harmonize screen objects using styles. Standardize the appearance of screen objects in a project to optimize project data. Use the specified preset or customized style for the configuration of the screen objects throughout the project.
- Enable automatic update of PLC alarms (for S7-1500 controllers and V14 HMI devices)
  To save space, you can specify that the PLC alarm texts are only to be loaded in Runtime when needed. To do this, enable the "Automatic update" option under "Runtime settings > Alarms > Controller alarms". Make sure that automatic update of alarms is also enabled in the corresponding controller.
  The "Automatic update" option is not available on Basic Panels.
  The amount of space that can be saved depends on the number of PLC alarms and the number of Runtime languages.

- Reduce the number of fonts loaded
  Check whether the number of downloaded user-defined fonts can be reduced. If necessary, configure only the standard fonts for the required Runtime languages under "Runtime settings > Language & font > Runtime language and font selection".
  To save space, use fewer font groups for the configuration.
- Reduce the size of the graphics
  Check the size of the graphics that you use in the project. If necessary, reduce the size of the graphics by reducing the resolution or choose a higher compression format without noticeable loss of quality for the project graphics.
  To keep the project size small for Basic HMI devices, harmonize the sizes of graphics used in the project.
  Select appropriate graphic formats for your screens: For example, use PNG images for drawings that are not vector graphics and JPEG images for photos.

12.12.6 Basics of operating in Runtime

12.12.6.1 Overview (Basic Panels)

Overview of operator control of a project

Depending on the HMI device, the following options are available for the input with Basic HMI devices.

- Touch screen
- Function keys
- Mouse and keyboard

Use the touch screen and function keys or mouse and keyboard to operate the Control Panel / Start Center or the project running on your HMI device.

Note

Incorrect operation

A project can contain certain operations that require in-depth knowledge about the specific plant on the part of the operator.
Make sure that your plant is only operated by qualified personnel.

Operating options for an HMI device

Depending on your HMI device, operate your plant as follows:

- Touch screen operation
  The display of the device is touch-sensitive. You operate the operating elements in the display with your finger or a stylus.
- Operation with control keys and function keys
  Control keys and function keys are integrated into the housing of the device.
  - Control keys have a specified function, for example, navigation or the acknowledgment of alarms.
  - Function keys are freely user-assignable and their function is therefore project-specific.
- Mouse and keyboard operation
  Mouse and keyboard are connected via an integrated USB port. You operate the operating objects with mouse and keyboard.

Individually configured operation

The configuration engineer has various options available for setting up operation.

Examples of actions whose execution is always determined on a project-specific basis:

- Screen change
- Reporting
- Changing Runtime language

There are no specific operating elements to execute certain functions. The configuration engineer specifies the project-specific execution. A screen change can be triggered with a button or a function key, for example.

Information on project-specific operations can be found in the system documentation.

12.12.6.2 Operation with the touch screen

Overview of operation with the touch screen

Use the touch screen to operate the HMI device of the project that is running on your HMI device.

Operating the touch screen

NOTICE
Damage to the touch screen Do not touch the touch screen with sharp or pointed objects. Avoid tapping the touch screen with hard objects, and avoid constantly using motion control. Both can significantly reduce the useful life or even cause the failure of the touch screen. **Triggering unintended actions** You can trigger unintended actions if you touch several operating elements at the same time. Always touch only one operating element on the screen. Operating elements are touch-sensitive symbols on the screen of the HMI device.

Special features of operation using the touch screen

Operation with the touch screen is characterized by the following special features:

- Enable
  To enable an operator control, touch the touch screen with your fingers or with a stylus. To generate a double-click, touch the operator control twice in rapid succession.
- Value input
  You enter numbers and letters on the touch screen with a screen keyboard.
- Careful operation
  If you touch multiple operator controls at the same time, you may trigger unintentional actions.

Value input using the screen keyboard

The screen keyboard is displayed as soon as you operate a screen object that requires an input. Depending on the HMI device and the configured operating element, the system displays different screen keyboards for entering numerical or alphanumerical values. The screen keyboard is hidden again when input is complete.

File browser (Windows 8 or higher)

You can only operate the file browser dialog with a mouse, keyboard or on-screen keyboard (without using the touch function) on a Windows 8 or higher PC with touch screen. Use the file browser dialog of the Windows operating system with the help of a script on a touch screen PC with Windows 8 or higher.

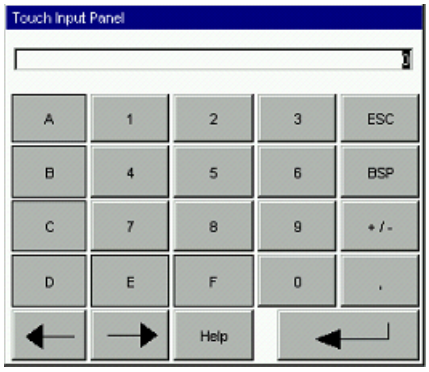
See also

Screen keyboard (Page 5936)

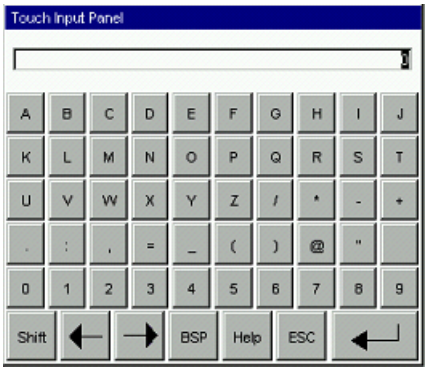
Screen keyboard

Layout

The figure below shows the basic layout of a screen keyboard on a TP1500 Basic.



Numbers



Letters

Operator controls

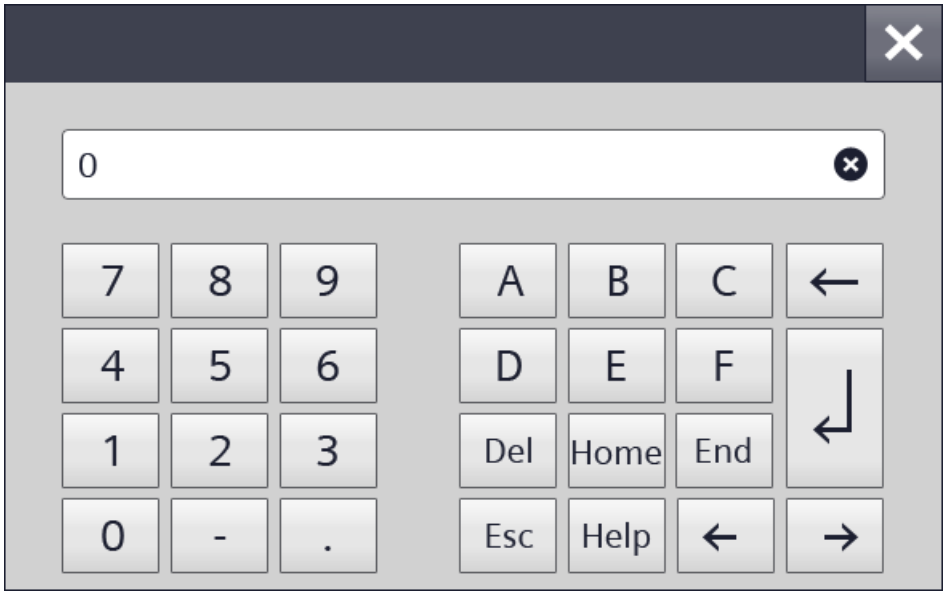
The following keys are available on the screen keyboard of all HMI devices:

Button	Name	Function
	Cursor left	Navigates to the left
	Cursor right	Navigates to the right
	Backspace	Deletes a character
	Escape	Cancels the input
	Enter	Confirms the input
	Help	Displays the infotext This key is only displayed when an infotext has been configured for the operator control.

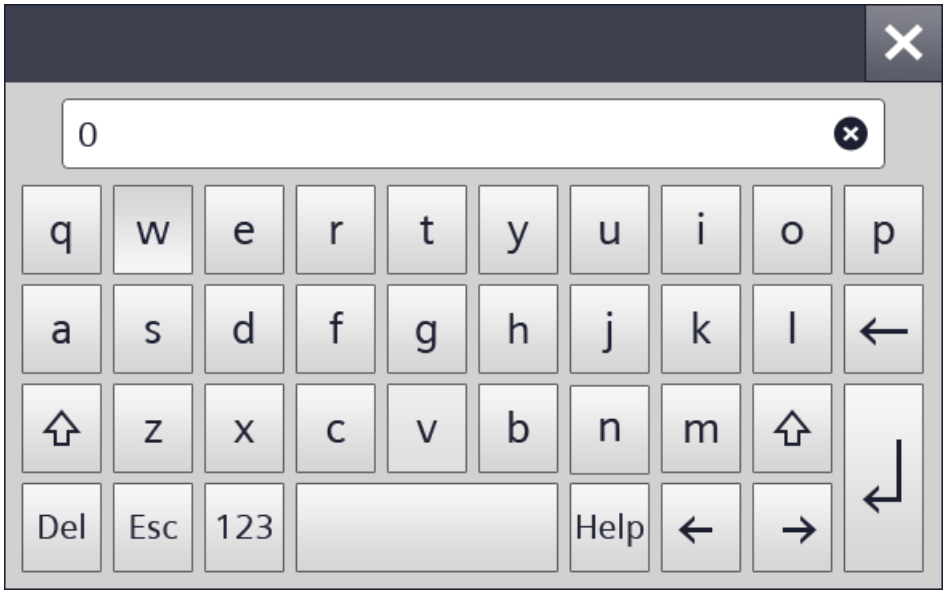
Screen keyboard for 2nd generation Basic Panels

Layout

The figure below shows the basic layout of a screen keyboard on a Basic Panel 2nd Generation.



Numbers



Letters

Operator controls

The following keys are available on the screen keyboard of all HMI devices:

Button	Name	Function
	Cursor left	Navigates to the left
	Cursor right	Navigates to the right
	Backspace	Deletes a character
	Escape	Cancels the input
	Enter	Confirms the input
	Help	Displays the infotext This key is only displayed when an infotext has been configured for the operator control.

12.12.6.3 Operation with keys

Overview of operation with keys

Introduction

Use the keys of the HMI device to operate the Control Panel / Start Center of your HMI device or the project that is running on your device. Depending on the device, control keys and function keys are available.

For more detailed information, refer to the operating instructions for your HMI device.

See also

Control keys and shortcuts (Page 5939)

Function Keys (Page 5941)

Control keys and shortcuts

Introduction

The following tables list the control keys available to operate the project.

Note

The availability of control keys is determined by the HMI device used.

You trigger functions on keyboard HMI devices either with a key or a shortcut. With shortcuts, you keep the first key pressed. Then you press the second key.

Navigating in the display

Key or shortcut		Function	Description
TAB		Tabulator	Selects the next operating element in the tab sequence
SHIFT	TAB	Tabulator	Selects the previous operating element in the tab sequence
◀ ▶ ▼ ▲		Cursor keys	Selects the next operating element to the left, to the right, above or below the current screen object Navigating in the operating element

Operation of operating elements

Key or shortcut		Function	Description
ENTER		ENTER key	• Controls buttons. • Applies and ends an entry • Opens a selection list • Toggles between character mode and standard mode within an input field A single character is selected in character mode. In this mode, you can scroll within the character set using the cursor keys.
SHIFT	▶ ◀ ▲ ▼	Positioning cursor	Positioning the cursor within an operating element, for example, in an I/O field
←		Delete characters	Deletes the character to the left of the current cursor position
INS DEL		Delete characters	Deletes the next character to the right of the current cursor position
ESC		Cancel	• Deletes the input characters of a value and restores the original value • Closes the active dialog.
F N	HOME	Scroll to start	Scrolls to the start page of a list
HOME		Scrolling back	Scrolls the list back by one page
F N	END	Scroll to end	Scrolls to the end of a list.
END		Scrolling up	Scrolls the list up by one page
ALT	▼	Open selection list	Opens a selection list
CTRL	ENTER	Accept value	Accepts the value selected in the selection list without closing the list

Enter shortcut

Key	Function	Purpose
A-Z	Toggle (numbers/letters)	Toggles the assignment from numbers to letters • No LED is lit: The number assignment is enabled. Pressing the key once switches to letter assignment. • The right or left LED is lit: The left or right letter assignment is enabled. Each time the key is pressed, the system toggles between the left letter assignment, the right letter assignment and the number assignment.
SHIFT	Shift (upper/lower case)	Used in shortcuts, for example, for switching to uppercase letters
F_N	Toggle to additional keyboard layout	Some of the keys contain a blue special character in their bottom left corner, for example, the percent sign "%". To input these characters, press the relevant key in combination with the special character key shown on the left.
CTRL	General control function	Used in shortcuts, for example, for navigating trend views
ALT	General control function	Used in shortcuts, for example for the "Watch table" screen object

Acknowledge alarms

Key	Function	Purpose
ACK	Acknowledge	Acknowledges the currently displayed alarm or all alarms of an alarm group The LED is lit as long as an unacknowledged alarm is active.

Displaying infotext

Key	Function	Description
HELP	Displaying infotext	For the selected object, opens a window with the configured infotext, for example, for an alarm or an I/O field The LED is lit if an infotext is available for the selected object.

Key or shortcut

Function Keys

The assignment of the function keys (F1, F2, F3, etc.) is defined during configuration.

Function keys with global function assignment

A globally assigned function key always triggers the same action on the HMI device or in the PLC regardless of the screen displayed. The activation of a screen or the closing an alarm window, for example, is such an action.

Function keys with local function assignment

A function key with local function assignment is screen-specific and is therefore only effective within the active screen.

The function of a locally assigned function key can vary from screen to screen.

Within a screen, a function key has only one function assignment, either a global or local one. The project planner specifies which assignment has priority.

Operating function keys

Note

Operating function key after screen change

If you press a function key after a screen change, the associated function may be triggered in the new screen before the new screen is fully displayed.

12.12.6.4 Navigating in the display (BS)

Introduction

You navigate on the display of your HMI device as follows:

- between configured screen objects
- within screen objects
  When you select a complex screen object, the cursor focus switches to the screen object and follows the tab sequence there.
- in tables of screen objects

Procedure

- To navigate in the specified tab sequence, press the <TAB> key.
- To navigate freely between the operator controls, press the cursor keys.

Depending on the configuration of your HMI device, you can also use function keys or shortcuts for navigation.

When you operate your HMI device with the touch screen or with the mouse, you implicitly navigate by triggering a desired action. For this purpose, touch or click the operator control.

Result

The operator controls receive the cursor focus according to the selected sequence. You can trigger an action on the selected operator control.

For more detailed information, refer to the operating instructions for your HMI device.

12.12.6.5 Triggering an action

Introduction

Triggering an action at an operator control can mean the following:

- A command is executed.
  Example: Click a button to trigger a script or to execute a predefined function.
- An object is enabled.
  Example: To enter a value, select a table cell with the <Enter> key.

Requirement

- You have navigated to the operator control on which you want to trigger the action.
- The operator control has the cursor focus.

Procedure

- Press <Enter>.
  Or
- Touch the operator control on the touch screen once or twice in rapid succession.
  Or
- Click or double-click the operator control with the mouse.

Result

The following results are possible:

- The requested command is executed.
- The screen keyboard is opened and/or the cursor blinks in the input area of the operator control.
- The element is selected and can be moved.

For more detailed information, refer to the operating instructions for your HMI device.

12.12.6.6 Entering a value

Introduction

Depending on the input format, you enter numerical or alphanumerical values in an input field.

You enter these values depending on the existing hardware using the screen keyboard, the control keys of the HMI device or an external keyboard.

Requirement

- The object is an input field or table field.
- The operator control is enabled.

Entering a value

1. Enter the desired value.
2. To confirm the value and exit the field, press the <Enter> key.
3. To discard the value and exit the field, press the <Esc> key.

Result

A value is entered or discarded. You navigate as needed to the next operator control.

For more detailed information, refer to the operating instructions for your HMI device.

12.12.6.7 Moving operator controls

Introduction

In Runtime, you can move the movable operator controls of a screen object with the mouse or using the touch screen, for example, a slider or a scroll bar. Operation with the keyboard is described below.

Requirement

- A movable operator control is selected.

Procedure

- To move the operator control, proceed as follows depending on the operating element:
  - Standard for touch screen: Press the cursor keys.
  - Standard for keyboard devices: Press <SHIFT> and the cursor keys.
  - Switches: Press <ENTER>
  - Slider: Press <PgUp> or <PgDn>

1. To finish the movement, navigate to another screen object or operator control.

Slider procedure

1. To move the operator control, press the cursor keys.
2. To finish the movement, navigate to another screen object or operator control.

Result

The position of the movable operator control and the display in the screen object have changed.

For more detailed information, refer to the operating instructions for your HMI device.

12.12.6.8 Displaying infotext

Introduction

Depending on the configuration, additional information and operating instructions are available as infotext. The infotext is assigned to an operating element, an alarm or to the open screen. The infotext of an I/O field may contain, for example, information about the value to be entered.

As an alternative to the <Help> key of the HMI device, use the <Help> key of the screen keyboard for input objects.

Requirement

- An infotext is configured for the operating element, the screen or an alarm.

Calling the infotext

1. Enable the desired operating element.
2. Press the <Help> key of the HMI device.
   The infotext for the operating element is displayed.

If you are operating your input object with the touch screen, the screen keyboard opens. If the <Help> key appears, an infotext is configured for the operating element or the current screen.

If there is no infotext for the selected screen object, the infotext for the current screen is displayed, if it has been configured.

Use the scroll bar for long infotexts.

Depending on your configuration, infotext can also be retrieved by means of a configured operating element.

Switching between infotexts

- To switch between the infotexts of the operating elements and the screen, enable the infotext window.

Hiding infotext

- To hide the infotext, press the <Esc> key or press the <Help> key again.

12.12.6.9 Changing Runtime language

Introduction

The HMI device supports multilingual projects. A corresponding operating element which lets you change the language setting on the HMI device in Runtime has been configured.

The project always starts with the language set in the previous session.

Requirement

- The required language for the project is available on the HMI device.
- The language switching function is linked to an operating element, for example, to a button.

Selecting a language

You can change project languages at any time. Language-specific objects are immediately displayed on the screen in the new language when you switch languages.

You can switch the language in Runtime in one of the following ways:

- Use a configured operating element to switch from one language to the next in a list.
- Use a configured operating element to directly set the required language.

12.12.7 Entering bar codes with optical handheld readers

Introduction

With the optical handheld readers you can identify components, machines and other objects optically and enter them directly into multiple operating elements on your HMI device.

Optical handheld reading devices read two dimensional data matrix codes, one dimensional barcodes and postal bar codes.

The following optical handheld readers are supported:

- SIMATIC MV320
- SIMATIC MV340

You can find templates for the settings and instructions on configuration in the manual for your optical handheld reader.

Note

The optical handheld reader is connected to the USB port of the HMI device. Only one device at a time can use the USB port. This is why it is not possible to use a USB keyboard and an optical handheld reader or two optical readers at the same time.

Note

When you connect the optical reader to a configuration PC, ensure that the USB port provides sufficient power for the reader.

Procedure

You read the marking of an object using the connected optical handheld reader. The value is entered in the object on the HMI device which has the focus during runtime.

After it has been read in, confirm the value with the Enter key or with the "Suffix - Enter" that you have previously configured in the settings of your optical handheld reader.

Basic Panels, Comfort Panels, Mobile Panels, RT Advanced

The following objects support input with optical handheld readers:

Object	Preconditions for input
I/O field Date/time field	The corresponding data type is selected. The object and the tag length are configured accordingly. The operator element has the cursor focus.
Recipe view	The recipe data record has the cursor focus.
User view	The logon dialog is open. The user name or password has the cursor focus.
HTML browser	The operator element has the cursor focus.
Runtime dialogs which support keyboard entry	The dialog is open and the corresponding input field has the cursor focus.

Comfort Panels, Mobile Panels, RT Advanced

The following objects support input with optical handheld readers:

Object	Preconditions for input
Sm@rtClient view	Write access is configured for Sm@tClient. The operator element has the cursor focus.
File browser	The field "File path" has the cursor focus.
Watch table	The input field "Control value" has the cursor focus.

Object	Preconditions for input
PDF view	The search field has the cursor focus.
SINUMERIK NC operator display	The operator element has the cursor focus.

Result

The code is read and entered into the corresponding input field.

12.12.8 Security on the HMI device

12.12.8.1 Signature basics (Basic Panels, Panels, Comfort Panels, RT Advanced)

Overview

The HMI device software and the runtime and device options are protected from manipulation by signatures. The device options available over ProSave are also signed.

HMI device signature

The validity of the HMI device signature is checked when the HMI device software is loaded. If the HMI device signature is invalid, loading of the HMI device software is canceled and an error message is output.

Signature for runtime and device options

If the signature check finds that the signature is invalid, an error message to this effect is output. An error message indicating an invalid signature is also output in ProSave.

Note

To allow loading of the HMI device software or the options without a signature (for versions prior to V14), you will need to switch off the signature check.

For details of how to do this, see Switching off the signature check (Page 5949).

See also

Switching off the signature check (Page 5949)

12.12.8.2 Switching off the signature check (Basic Panels, Panels, Comfort Panels, RT Advanced)

Procedure

1. Press "Transfer" twice in the control panel of the HMI device.
   The "Transfer Settings" dialog box opens.
   The option "Validate signatures" is enabled by default.

   Note

   If you wish to load an older WinCC project to an HMI device, you will need to check the settings for the signature check.

2. To allow the HMI device software to be loaded without a signature or to allow the download of a trustworthy option from a third-party provider, disable the "Validate signatures" option.

See also

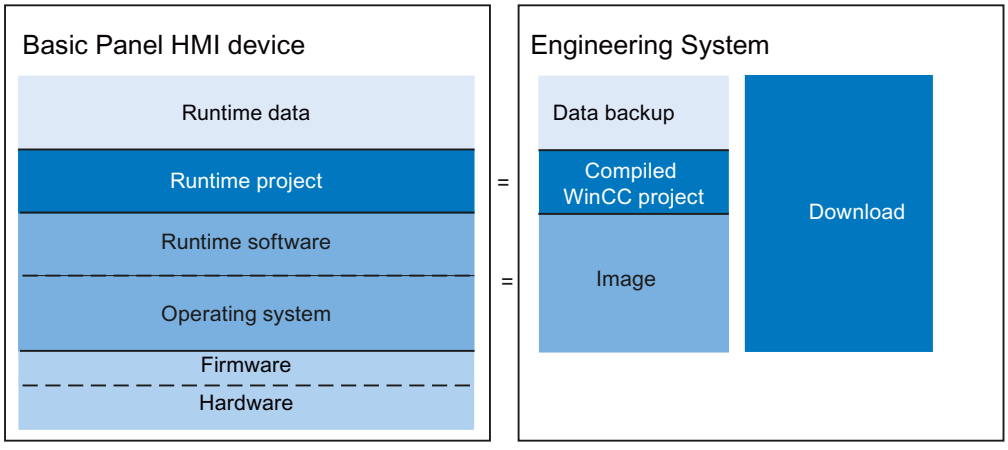
Signature basics (Page 5948)

12.12.9 Servicing the HMI device

12.12.9.1 Overview of HMI device maintenance (Basic Panels)

Structure

The following figure shows the software components of an HMI device and their relation to the engineering system.



Runtime data

The Runtime data is generated during operation of the plant and stored on the HMI device. This data includes, for example, recipes and data for the user administration. This data is overwritten during loading. If required, save this data before loading a Runtime project.

Runtime project

The Runtime project contains the compiled configuration data for an HMI device. You download the Runtime project from WinCC to the HMI device.

Runtime software and operating system

Together, the Runtime software and operating system of an HMI device form the image. Various images are available for the HMI device. All images of an HMI device are available in WinCC. Depending on the configuration, download the appropriate image along with the Runtime project to the HMI device as required.

Firmware and hardware

The HMI device is delivered with preconfigured firmware and hardware.

12.12.9.2 ProSave

Introduction

The "ProSave" service tool is installed by default when WinCC is installed. The ProSave functions are accessed in WinCC with the menu "Online > HMI Device maintenance".

Functional scope

ProSave provides all of the functions needed to transfer files to the HMI device.

- Data backup and restoration of backed-up data
- Operating system update
- Communication settings

See also

Backup of HMI data (Page 5951)

Updating the operating system (Page 5953)

Overview of HMI device maintenance (Basic Panels) (Page 5949)

12.12.9.3 Backup of HMI data

Introduction

Regular backups of HMI device data keep downtimes to a minimum, for example, when you replace a device. You simply transfer the backup data to the HMI device, restoring the original status.

Data backup with WinCC

If an HMI device is connected to the configuration PC, you can back up and restore HMI device data from the configuration PC using WinCC.

Scope of data backup

Which data is backed up and restored depends on the type of HMI device:

- Complete backup.
  Depending on the HMI device: Runtime, firmware, operating system, configuration, recipes, user administration, settings
- Recipes only
- User administration only
- Recipes as CSV file

A backup file with the extension *.psb is generated when you backup the data of an HMI device.

As a general rule, you can backup the data to any storage medium. If the HMI device is networked, you can also backup the data to a server.

Note

Scope of data backup

Data backup secures the contents of the flash memory. Note the following when performing a complete data backup and restore operation for HMI devices:

- In the event of a full restore of all data, all data previously on the device, including the operating system, is deleted irrevocably.
- If the data restoration was interrupted, execute the command "Reset to factory settings". Restart data restoration.

See also

Backing up and restoring data of the HMI device (Page 5952)

ProSave (Page 5950)

Overview of HMI device maintenance (Basic Panels) (Page 5949)

12.12.9.4 Backing up and restoring data of the HMI device

Note

Use the restore function for project data only on operating devices which were configured using the same configuration software.

Requirement

- The HMI device is connected to the configuration PC
- The HMI device is selected in the project tree.
- If a server is used for data backup: The configuration PC has access to the server

Backup of the data of the HMI device

Proceed as follows to backup the data of the HMI device:

1. Select the "Backup" command from the "Online > HMI Device maintenance" menu.
   The "Create backup" dialog box opens.
2. Select the type of the PG/PC interface and the target device, and click "Create".
   The "SIMATIC ProSave" dialog box opens.
3. Select the data to backup for the HMI device under "Data type".
4. Enter the name of the backup file under "Save as".
5. Click "Start Backup".

This starts the data backup. The backup operation takes some time, depending on the connection selected.

Restoring the data of the HMI device

Proceed as follows to restore the data of the HMI device:

1. Select the "Restore" command from the "Online > HMI Device maintenance" menu.
   The "Restore backup" dialog box opens.
2. Select the type of the PG/PC interface and the target device, and click "Load".
   The "SIMATIC ProSave" dialog box opens.
3. Enter the name of the backup file under "Save as".
   Information about the selected backup file is displayed under "File information".
4. Click "Start Restore".

This starts the restoration. This operation takes some time, depending on the connection selected.

Backup/Restore from the "Backup/Restore" dialog in the Start Center of the HMI device

The "Backup/Restore" function is enabled for MMC, SD memory cards and USB mass storage devices. The storage media supported depend on the HMI device.

Note

Connections between Basic Panels and controllers

If you use the "Backup/Restore" function, a maximum of two connections from Basic Panels to the following controllers are possible at any given time:

- SIMATIC S7-1200
- SIMATIC S7-1500

For more information on this topic, refer to the device manual of the employed HMI device.

See also

Backup of HMI data (Page 5951)

Overview of HMI device maintenance (Basic Panels) (Page 5949)

12.12.9.5 Updating the operating system

Introduction

If the image of an HMI device has a version that does not match the configuration, it is automatically updated, after prompting, during the loading of the project. Loading will then continue. If you decline to update the image, loading is canceled. Updating the image also updates the operating system of the HMI device.

Updating the image

Connect the HMI device to the configuration PC to update an image. If possible, you should use the interface with the highest bandwidth for this connection, such as Ethernet. Updating the image via a serial connection can take up to an hour.

"Reset to factory settings"

If the operating system on the HMI device is no longer operational, update the operating system and reset the HMI device to the factory settings.

Note

You will require the following to reset to the factory settings via Ethernet:

- the MAC address of the HMI device
- the available IP address
- the programming device/PC interface of the configuration PC set to Ethernet TCP/IP

The programming device/PC interface is configured using the control panel of the configuration PC. Select" "S7ONLINE (STEP7) -> TCP/IP" in the "Access point of the application" field.

See also

Updating the HMI device operating system (Page 5954)

ProSave (Page 5950)

Overview of HMI device maintenance (Basic Panels) (Page 5949)

12.12.9.6 Updating the HMI device operating system

When you update the operating system, the Runtime software on the HMI device is also updated and the device version is changed.

NOTICE
An operating system update deletes all the data from the HMI device The operating system update deletes all data from the target system. For this reason, you should back up the following data beforehand: • User administration • Recipes

Requirement

- The HMI device is connected to the configuration PC or the PC with ProSave.
- The HMI device is selected in the project navigation.

Updating the operating system

Proceed as follows to update the operating system:

1. Select the "Update operating system" command from the "Online > HMI device maintenance" menu.
2. Select the "Update operating system" command from the menu under "Online > HMI Device maintenance" on the configuration PC in WinCC.
   The "SIMATIC ProSave [OS-Update]" dialog opens. The path to the image is preset.
3. If required, you can select a different path for the image that you want to transfer to the HMI device.
4. Click "Update OS".

This starts the update. The update operation can take time, depending on the connection selected.

Resetting the HMI device to factory settings

To reset the HMI device to factory settings, proceed as follows:

1. Switch off power to the HMI device.
2. Select the "Update operating system" command from the menu under "Online > HMI Device maintenance" on the configuration PC in WinCC.
   The "SIMATIC ProSave [OS-Update]" dialog opens. The path to the image is preset.
3. If required, you can select a different path for the image that you want to transfer to the HMI device.
4. Activate "Reset to factory settings".
5. Click "Update OS".
6. To start "Reset to factory settings", switch on power to the HMI device again.

This operation can take time.

Result

The operating system of the HMI device is updated to the latest version.

See also

Updating the operating system (Page 5953)

Overview of HMI device maintenance (Basic Panels) (Page 5949)

12.12.10 Reference

12.12.10.1 Error messages during loading of projects

Possible problems during the download

When a project is being downloaded to the HMI device, status messages regarding the download progress are displayed in the output window.

Usually, problems arising during the download of the project to the HMI device are caused by one of the following errors:

- Incorrect download settings on the HMI device
- Incorrect HMI device type in the project
- The HMI device is not connected to the configuration PC.

The most common download failures and possible causes and remedies are listed below.

The serial download is cancelled

Possible remedy: Select a lower baud rate.

The download is cancelled due to a compatibility conflict

Possible cause	Remedy
The configuration PC is connected to the wrong device, e.g. a controller.	Check the cabling. Correct the communication parameters.

Project download fails

Possible cause	Remedy
Connection to the HMI device cannot be established (alarm in the output window)	Check the physical connection between the configuration PC and the HMI device. Check whether the HMI device is in transfer mode. Exception: Remote control
The default communication driver is not listed in the Windows Device Manager.	Check the device status of the COM connection in the properties window of the Device Manager.

Download over MPI/DP interface fails

Possible cause	Remedy
"Configured mode" is set on the CP, for example, if you are using the SIMATIC NET CD.	Set the CP to "PG mode" using the "Set PC station" application. Check the "baud rate" and "MPI address" network parameters. Download the project from WinCC to the CP. Set the CP back to "configured mode".
On the programming device/PC panel, the "S7ONLINE" access point is not set to a hardware device such as CP5611 (MPI). The cause may be the installation of "SIMATIC NET CD 7/2001".	Set the access point "S7ONLINE" on the selected device using the "PG/PC Panel" or "Set PC station" application. Check the "baud rate" and "MPI address" network parameters. Download the project from WinCC to the HMI device. Restore the "S7ONLINE" access point to the original device.

The configuration is too complex

Possible cause	Remedy
The configuration contains too many different objects or options for the HMI device selected.	Remove all objects of a type, e.g. all graphic views.

12.12.10.2 Adapting the project for another HMI device

Introduction

When you download a WinCC project to an HMI device, WinCC checks whether this is compatible with the HMI device type used in the project. If the types of HMI device do not match, you will see a message before the download starts.

The download is aborted.

Adapting the project for the HMI device

You need to adapt the project accordingly to be able to download the project to the connected HMI device.

- Add a new HMI device in the project tree. Select the correct type of HMI device from the HMI device selection.
- Copy the configured components from the previous to the new HMI device.
  Copy large amounts of components directly in the project navigation and details view.
  For example, copy the "Screens" folder to the screens folder of the new HMI device with the help of the shortcut menu.
- Use the detail view to copy entries in the project tree for which the "Copy" command is not available in the shortcut menu.

- Select the "Recipes" entry in the project tree, for example. The recipes are displayed in the detail view.
- Select the recipes in the detail view and drag them to the "Recipes" entry of the new HMI device. The recipes are copied. You can also select multiple objects in the detail view.
- Configure the components that cannot be copied, e.g. connections, area pointers, and alarms.
- Save the project at various points in time.
- Compile the full project.
- When the compilation is successfully completed, download the project to the HMI device.

Linking references

References to linked objects are included in the copying. The references are linked again once the linked objects are copied.

Example:

You copy a screen in which objects are linked to tags. The tag names are entered at the individual objects after the screen is added to the new HMI device. The tag names are marked in red because the references are open. When you then copy the tags and insert them into the new HMI device, the open references are closed. The red marking for the tag names disappears.

For complete references to connected objects in the PLC, you first need to configure a connection to the PLC.

Using the information area

When you compile the project for the HMI device, errors and warnings are displayed in the "Info" tab of the Inspector window. You can use the shortcut menu command "Go to" to go directly to the location where the error or warning can be corrected.

Work through the list of errors and warnings from top to bottom.

When the compilation is successfully completed, download the project to the HMI device.

12.12.10.3 Establishing a connection to the HMI device

Introduction

To download a WinCC project to an HMI device, a properly configured connection must be set up between the configuration PC and HMI device. The connection cannot be set up, the download is cancelled.

Setting up a connection between the configuration PC and HMI device

1. Check the cable connection between the HMI device and configuration PC.
2. Open the "Devices & Networks" editor in WinCC and start the network view.

3. Select the subnet in the network view and check the settings for the subnet.
4. Select the interface of the HMI device in the network view or device view and check the connection parameters in the Inspector window.
5. Switch on the HMI device and press the "Control Panel" button in the loader.
   The Control Panel opens.
6. Press "Transfer" twice in the Control Panel.
   The "Transfer Settings" dialog box opens.
7. Check the settings and then press "Advanced".
   The [Protocol*] Settings" dialog opens.
   *: The title of the dialog depends on the protocol used, for example, "PROFIBUS Settings".
8. Check the advanced settings and close the dialog with "OK".

Important settings

Check the connection settings and in particular the following parameters:

- Network and station addresses
- Selected transmission rate
- Master on the bus; as a general rule, only one master is permitted.

Note

Connections over Ethernet

For Ethernet-based communication, check the required network settings. You can find additional information under "Readme > WinCC > Security information".

If using a configurable adapter for the connection, check the adapter settings, for example, transmission rate, master on the bus.

12.13 Performance features

12.13.1 General technical data

12.13.1.1 Permitted characters

Introduction

The following table shows the restrictions that must be observed when allocating names and passwords.

Permitted characters

Names / passwords	Restriction
Computer names	The name must start with a letter.
Names of objects	Do not use the characters ? " / \ * < > %
Names and passwords in user view	Do not use the characters ? " / \ § & $ %
Recipe data	Do not use the characters , ;
Names of alarm logs	In the storage location file - RDB, file - CSV (ASCII) and file -TXT (Unicode) not the characters \ / * ? : " < > |
	For the storage location database, use only the characters a-z A-Z 0-9 _ @ # $ The characters _ @ # $ may not be used as the first character of a name
Name of HMI tags	HMI tag names may not start with the @ character
Names of screens	Do not use the special characters ? " / \ * < > Avoid the character . in screens for Runtime Professional:
Tag names in faceplates	Do not use the characters . @
Property names of the faceplates	Do not use any UNICODE characters (for example, Chinese characters) in the names. The name must start with a letter. The name may not contain more than 255 characters.
Names of AuditTrails	Do not use any special characters
Names of customized user data types as library objects (RT Professional)	Do not use the special characters : ? " \\ \ * $ % " ' The name may not contain more than 128 characters.
Names of C scripts and VB scripts as library objects	The name must start with an ASCII letter, followed by an alphanumeric ASCII character or underscore. The name may not contain more than 128 characters.
Names of connections	Do not use spaces or special characters
Name of VB functions	Do not use spaces, special characters or VB keywords
Name of Windows user groups and Windows users when using SIMATIC Logon	Do not use the special characters / \
Communication / OPC: names used	Do not use either spaces or the characters . : ? " ' \ * %

Note

Device dependency - quotes in project name

WinCC Professional projects whose names are in quotes have not been compiled.

When you create an HMI device for WinCC RT Professional, do not use quotation marks within the project name.

12.13.1.2 Recommended printers

Recommended printers

The current list of printers recommended for use with the HMI devices is available on the Internet at:

Link to the current printer list (http://support.automation.siemens.com/WW/llisapi.dll?aktprim=0&lang=en&referer=%2fWW%2f&func=cslib.csinfo&siteid=csius&caller=view&extranet=standard&viewreg=WW&nodeid0=10805558&objaction=csopen)

Note

All HMI devices except for a PC and Panel PC support only one printer at their USB port, even if several ports are available.

See also

Printer list (http://support.automation.siemens.com/WW/llisapi.dll?aktprim=0&lang=en&referer=%2fWW%2f&func=cslib.csinfo&siteid=csius&caller=view&extranet=standard&viewreg=WW&nodeid0=10805558&objaction=csopen)

12.13.1.3 Memory requirements of recipes for Basic Panels

Introduction

The following calculation of memory requirements of recipes is valid for Basic Panels, OP 77A, and TP 177A devices.

Restrictions

The HMI device provides 39 KB of memory space for recipes. This memory space may not be exceeded. The total memory space for recipes is calculated as follows: Total of all recipes + recipe with highest memory requirement.

Each recipe may not exceed a maximum memory space of 19 KB.

Calculation of memory requirements

The memory space requirement of each recipe (in KB) is calculated based on the three addends D1 + D2 + D3.

Valid is::

- D1 = number of data records x M
  Rule for M (size of a data record):
  M = 1 x number of elements of a byte + 2 x number of elements of 2 bytes + 4 x number of elements of 4 bytes + 8 x number of elements of 8 bytes + K
  Rule for K (size of the string elements):
  K = number of string elements x (string length + 1) x 2

- D2 - data record size
  D2 = 4 + number of languages x 8 + number of languages x (4 + 4 x number of data records + (length of the data record name + 1) x 2 x number of data records) + 8 + 8 x number of data records
  Or rewritten:
  D2 = 12 + 8 x number of data records + number of languages x (12 + number of data records x (4 + (length of the data record name +1) x 2))

- D3 - shared memory
  D3 = 14 + number of elements

Note

Arrays and single elements can be calculated as described above.

12.13.2 Performance features of the devices

12.13.2.1 Basic Panel (Basic Panels)

Basic Panel

The following table helps you assess whether your project meets the performance features of the HMI device.

The specified maximum values are not additive. It cannot be guaranteed that configurations running on the devices at the full system limits will be functional.

In addition to the specified limits, allowances must be made for restrictions imposed by configuration memory resources.

Tags

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of tags in the project	250	500	250 (mono) 500 (color)	500	500	500
Number of PowerTags	--	--	--	--	--	--
Number of elements per array	100	100	100	100	100	100
Number of local tags	--	--	--	--	--	--
Number of structures	--	--	--	--	--	--
Number of structure elements	--	--	--	--	--	--

Alarms

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of alarm classes	32	32	32	32	32	32
Number of discrete alarms	200	200	200	200	200	200
Number of analog alarms	15	15	15	15	15	15
Length of an alarm in characters	80	80	80	80	80	80
Number of process values per alarm	8	8	8	8	8	8
Size of the alarm buffer	256	256	256	256	256	256
Number of queued alarm events	64	64	64	64	64	64

Screens

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of screens	50	50	50	50	50	50
Number of fields per screen	30	30	30	30	30	30
Number of tags per screen	30	30	30	30	30	30
Number of complex objects per screen [1)]	5	5	5	5	5	5
Number of array elements per screen [2)]	100	100	100	100	100	100

1) Complex objects include: Bars, sliders, symbol library, clock, and all objects from the Controls area.

2) Array elements contained in recipes are included in the count.

Recipes

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of recipes	5	5	5	5	5	5
Number of elements per recipe[1)]	20	20	20	20	20	20
User data length in bytes per data record	--	--	--	--	--	--
Number of data records per recipe	20	20	20	20	20	20
Reserved memory for data records in the internal Flash	40 KB	40 KB	40 KB	40 KB	40 KB	40 KB

1) If arrays are used, each array element counts as one recipe element

Logs

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of logs	--	--	--	--	--	--
Number of entries per log (including all log segments) [1)]	--	--	--	--	--	--
Number of log segments	--	--	--	--	--	--
Cyclic trigger for tag logging	--	--	--	--	--	--
Number of tags that can be logged per log	--	--	--	--	--	--

1) The number of entries for the "segmented circular log" logging method is the total number for all sequential logs. The product of the number of sequential logs and the number of data records per sequential log may not exceed the system limit

Trends

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of trends	25	25	25	25	25	25

Text lists and graphics lists

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of graphics lists	100	100	100	100	100	100
Number of text lists	150	150	150	150	150	150
Number of entries per text or graphics list	30	30	30	30	30	30
Number of graphic objects	500	500	500	500	500	500
Number of text elements	500	500	500	500	500	500

Scripts

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of scripts	--	--	--	--	--	--

Communication

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of connections	4	4	4	4	4	4
Number of connections based on "SIMATIC HMI HTTP"	--	--	--	--	--	--

Help system

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of characters in a help text	500	500	500	500	500	500

Languages

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of runtime languages	5	5	5	5	5	5

Scheduler

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Time-triggered tasks [1)]	--	--	--	--	--	--

1) Event-triggered tasks are not relevant for the system limits

User administration

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Number of user groups	50	50	50	50	50	50
Number of authorizations	32	32	32	32	32	32
Number of users	50	50	50	50	50	50

Project

	KP300 Basic	KP400 Basic	KTP400 Basic	KTP600 Basic	KTP1000 Basic	TP1500 Basic
Size of the project file "*.srt"	1024 KB	1024 KB	1024 KB	1024 KB	1024 KB	1024 KB

See also

S7-1200 manual (http://support.automation.siemens.com/WW/view/en/36932465/0/en)

12.13.2.2 Basic Panel 2nd Generation (Basic Panels)

Basic Panel 2nd Generation

The following table helps you assess whether your project meets the performance features of the HMI device.

The specified maximum values are not additive. It cannot be guaranteed that configurations running on the devices at the full system limits will be functional.

In addition to the specified limits, allowances must be made for restrictions imposed by configuration memory resources.

Tags

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of tags in the project	800	800	800	800
Number of PowerTags	--	--	--	--
Number of elements per array	100	100	100	100
Number of local tags	--	--	--	--
Number of structures	--	--	--	--
Number of structure elements	--	--	--	--

Alarms

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of alarm classes	32	32	32	32
Number of discrete alarms	1000	1000	1000	1000
Number of analog alarms	25	25	25	25
Length of an alarm in characters	80	80	80	80
Number of process values per alarm	8	8	8	8
Size of the alarm buffer	256	256	256	256
Number of queued alarm events	64	64	64	64

Screens

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of screens	250	250	250	250
Number of fields per screen	100	100	100	100
Number of tags per screen	100	100	100	100
Number of complex objects per screen [1)]	This information is not relevant for Basic Panels 2nd Generation.			
Number of recipe displays per screen	10	10	10	10
Number of trend displays per screen	8	8	8	8
Number of alarm displays per screen	20	20	20	20
Number of user displays per screen	1	1	1	1
Number of system diagnostic displays per screen	5	5	5	5
Number of system functions per screen	150	150	150	150
Number of multiplex tags per screen	100	100	100	100

1) Complex objects include: Bars, sliders, symbol library, clock, and all objects from the Controls area.

Recipes

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of recipes	50	50	50	50
Number of elements per recipe[1)]	100	100	100	100
User data length in KB per data record	32	32	32	32
Number of data records per recipe	100	100	100	100
Reserved memory for data records in the internal Flash	256 KB	256 KB	256 KB	256 KB

1) If arrays are used, each array element counts as one recipe element

Logs

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of data logs	1	1	1	1
Number of alarm logs	1	1	1	1
Number of entries per log (including all log segments) [1)]	10000	10000	10000	10000
Number of log segments	400	400	400	400
Number of tags that can be logged per log	10	10	10	10
Cyclic trigger for tag logging	1 s	1 s	1 s	1 s

1) The number of entries for the "segmented circular log" logging method is the total number for all sequential logs. The product of the number of sequential logs and the number of data records per sequential log may not exceed the system limit

Trends

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of trends	25	25	25	25

Text lists and graphics lists

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of graphics lists	100	100	100	100
Number of text lists	300	300	300	300
Number of entries per text or graphics list	100	100	100	100
Number of graphic objects	1000	1000	1000	1000
Number of text elements	2500	2500	2500	2500

Scripts

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of scripts	--	--	--	--

Communication

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of connections	4	4	4	4
Number of connections based on "SIMATIC HMI HTTP"	--	--	--	--

Help system

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of characters in a help text	500	500	500	500

Languages

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of runtime languages	10	10	10	10

Scheduler

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Time-triggered tasks [1)]	--	--	--	--

1) Event-triggered tasks are not relevant for the system limits

User administration

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Number of user groups	50	50	50	50
Number of authorizations	32	32	32	32
Number of users	50	50	50	50

Project

	KTP400 Basic	KTP700 Basic	KTP900 Basic	KTP1200 Basic
Size of the project file "*.srt"	10 MB	10 MB	10 MB	10 MB

12.14 Migrating to WinCC in the TIA Portal

12.14.1 Overview of the migration to WinCC in the TIA Portal

Overview of the section "Migration to WinCC in the TIA Portal"

SIMATIC WinCC offers a number of functional changes in the TIA Portal. Some functions differ from the functions that you know from familiar environments such as WinCC V7 or WinCC flexible.

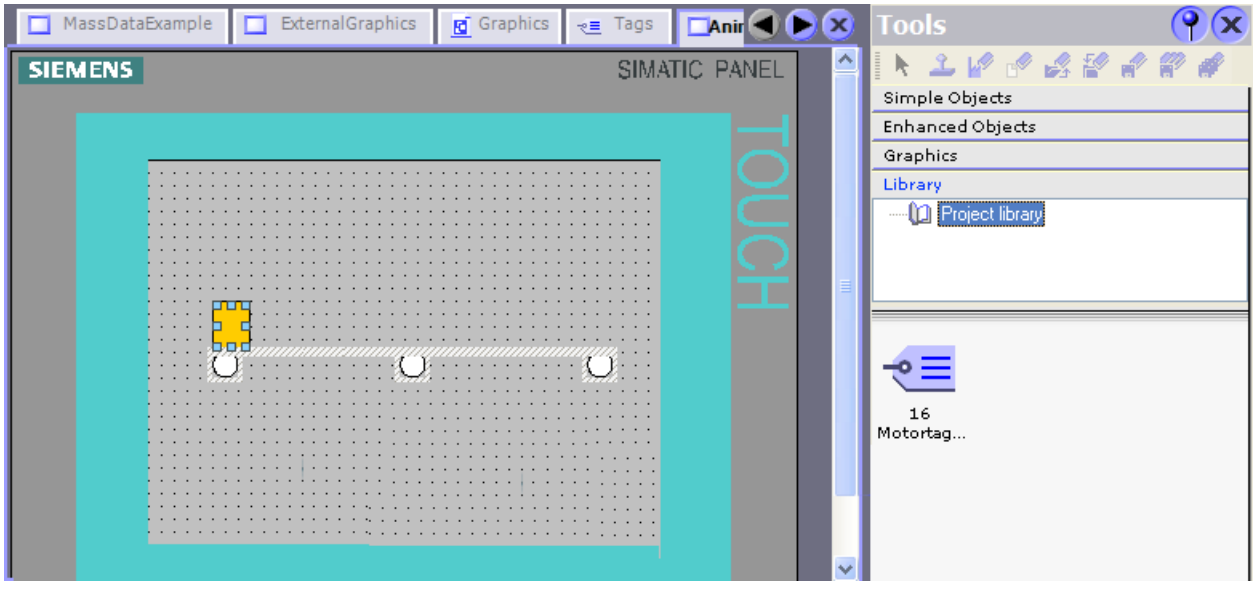
This document provides an overview of the special functions and procedures in SIMATIC WinCC in the TIA Portal.

These functions and procedures are fundamentally different from the WinCC V7 and WinCC flexible version, or have a different name.

12.14.2 Libraries

Libraries in WinCC flexible

Libraries are a collection of pre-configured screen objects. They expand the number of available screen objects and increase engineering efficiency, because library objects are always available for reuse; there is no need to reconfigure them.



WinCC flexible enables you to create two library types:

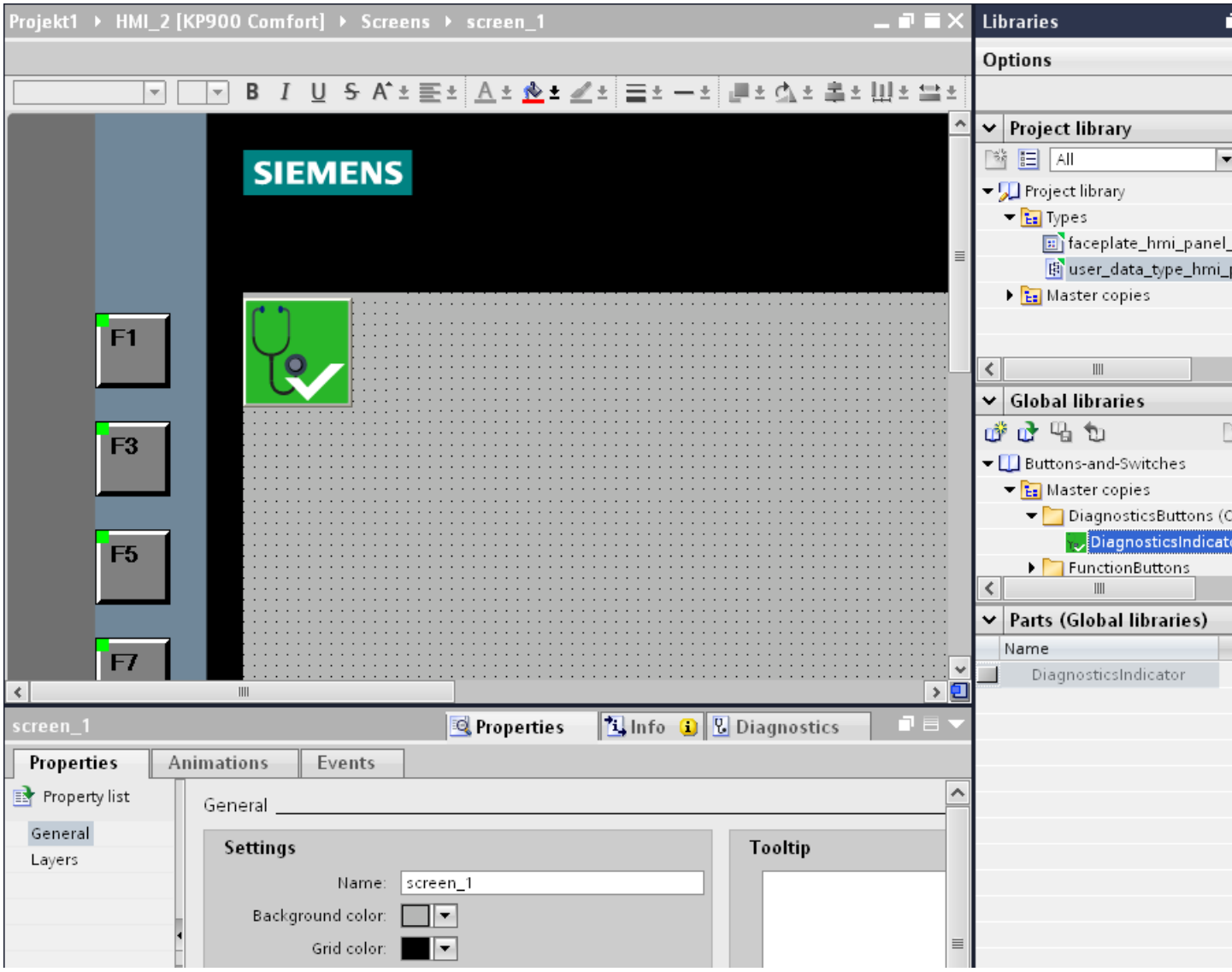
- Project library
- Global library

A library can contain all the WinCC flexible objects, such as screens, tags, graphic objects, or alarms.

How do I configure libraries with WinCC in the TIA Portal?

In WinCC, you also configure the "Project library" and the "Global library".

You can no longer store any system functions in libraries, as was the case in WinCC flexible.



Both the "Project library" and the "Global library" contain the two folders, "Copy templates" and "Types". You can create or use the library objects as a copy template, or as a type.

- Copy templates
  Use copy templates to create independent copies of the library object.
- Types
  Create instances of objects of the "Types" folder and use the instances in your project. The instances are bound to their respective type. Changes to an instance also change all other instances. Types are marked by a green triangle in the "Libraries" task card.
- Managing the library objects
  You can only copy and move library objects within the same library.

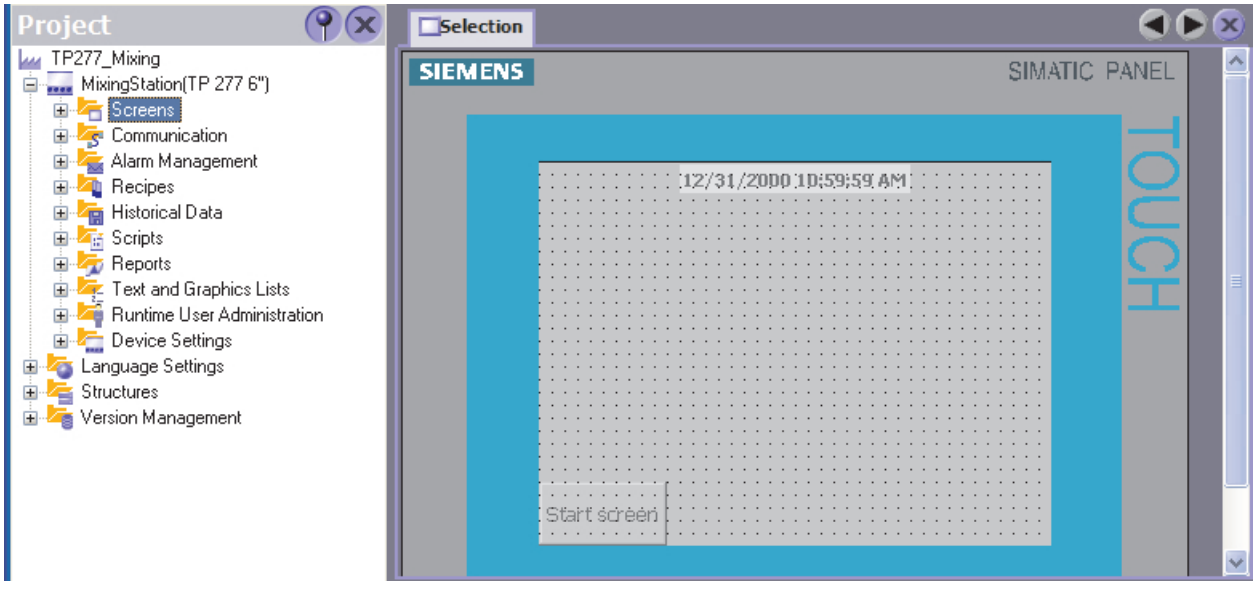
For more detailed information, see:

AUTOHOTSPOT

12.14.3 Screens and templates

Screens and templates in WinCC flexible

In WinCC flexible, you create screens that an operator can use to control and monitor machines and plants. When you create your screens, the object templates provided support you in visualizing your plant, displaying processes and defining process values.



The project has a template for every HMI device. You can centrally configure the function keys and objects for your project in these templates.

Every screen based on this template will contain the function keys and objects that you configured in the template. Changes to an object or of a function key assignment in the template are applied to the object in all the screens, which are based on this template.

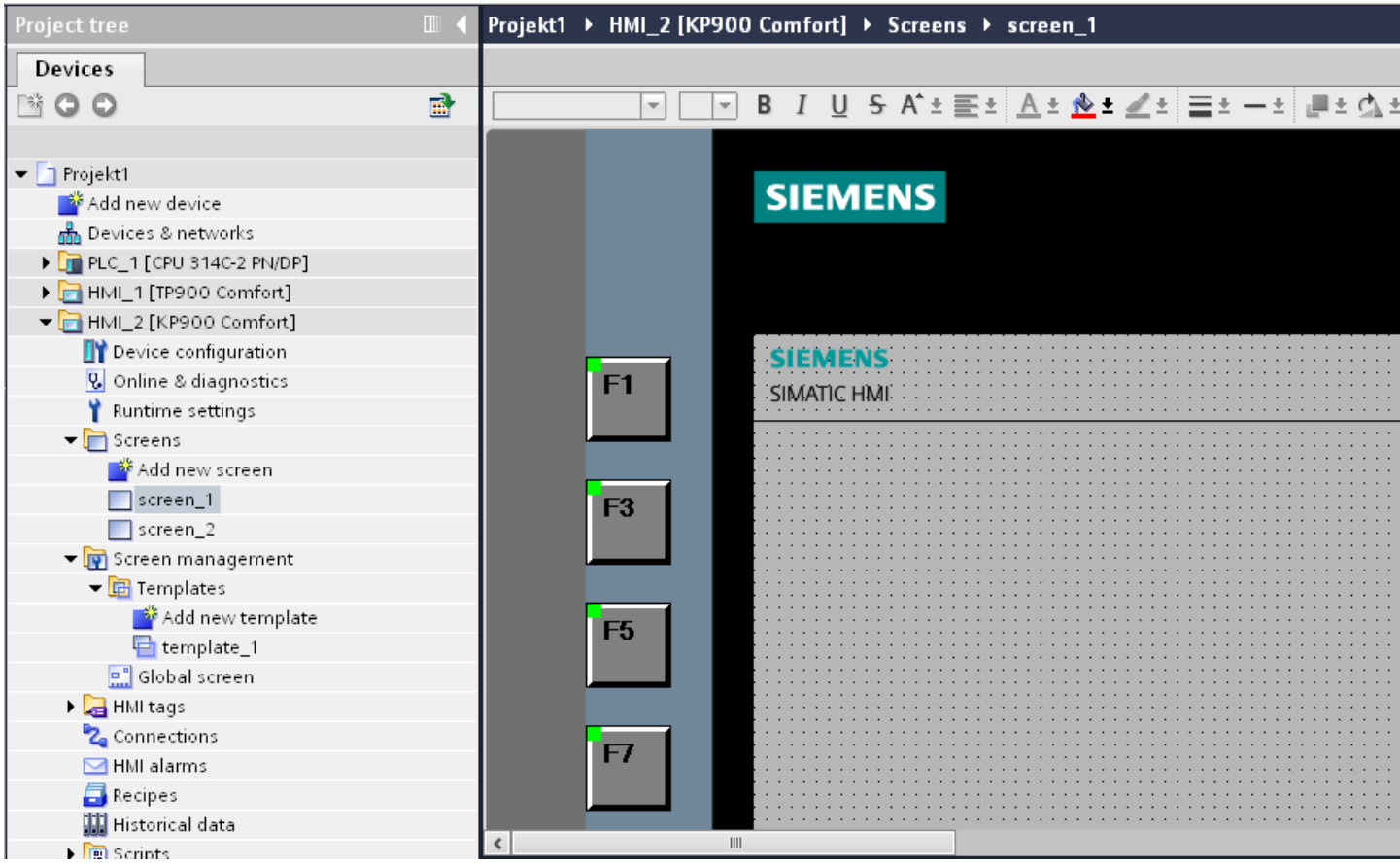
How do I configure screens and templates with WinCC in the TIA Portal?

In WinCC, you also configure "Templates" and a "Global screen" along with the "Screens".

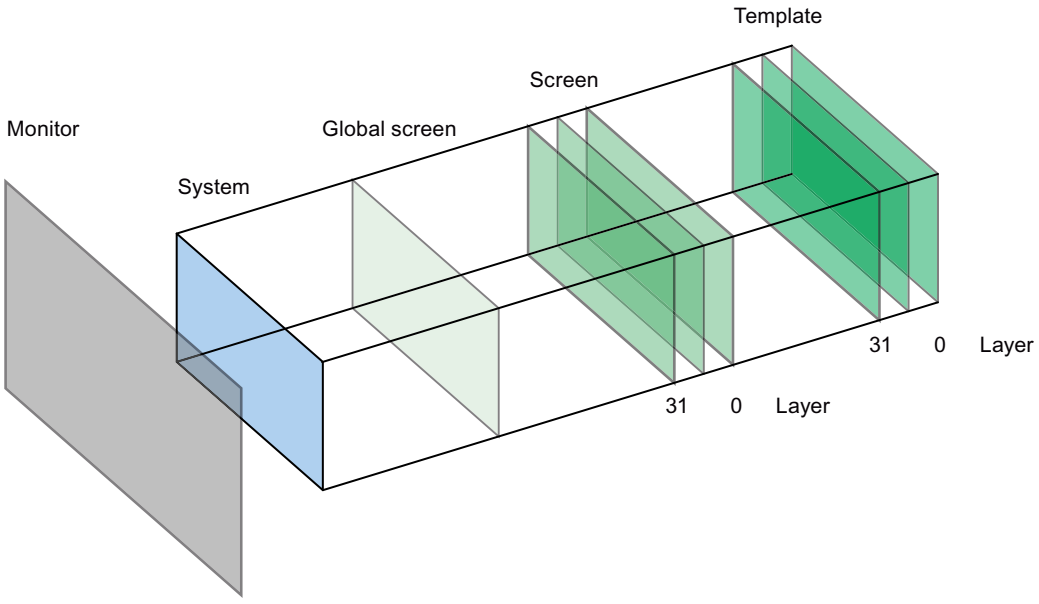
You determine functions and objects in the template which then apply to all screens based on this template. You can create multiple templates in WinCC.

In the "Global screen", define the elements which are independent of the template used for all screens of an HMI device. The "Alarm window" and "Alarm indicator" objects are available for use as global objects. For HMI devices with function keys, configure the function keys in the "Global Screen" editor.

You can also configure a "System Diagnostics Window" in the global screen of Comfort Panels.



Excluding the controls, the screens are displayed in runtime in the following order:



For more detailed information, see:

AUTOHOTSPOT

Using technology functions 13

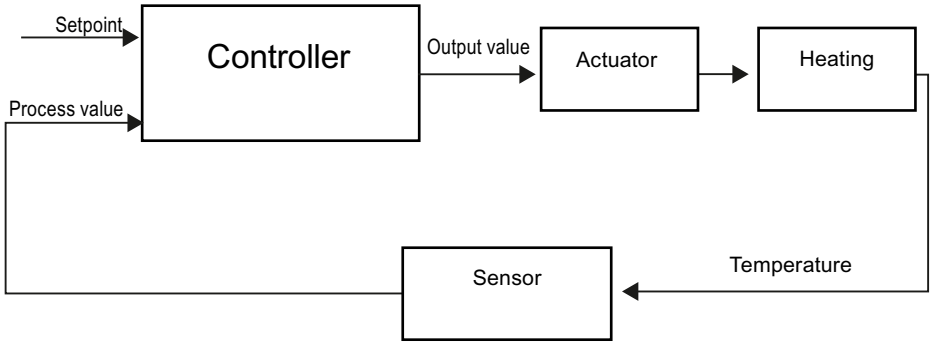
13.1 PID control

13.1.1 Principles for control

13.1.1.1 Controlled system and actuators

Controlled system

Room temperature control by means of a heating system is a simple example of a controlled system. A sensor measures the room temperature and transfers the value to a controller. The controller compares the current room temperature with a setpoint and calculates an output value (manipulated variable) for heating control.



A properly set PID controller reaches this setpoint as quickly as possible and then holds it a constant value. After a change in the output value, the process value often changes only with a time delay. The controller has to compensate for this response.

Actuators

The actuator is an element of the controlled system and is influenced by the controller. Its function modifies mass and energy flows.

The table below provides an overview of actuator applications.

Application	Actuator
Liquid and gaseous mass flow	Valve, shutter, gate valve
Solid mass flow, e.g., bulk material	Articulated baffle, conveyor, vibrator channel
Flow of electrical power	Switching contact, contactor, relay, thyristor
	Variable resistor, variable transformer, transistor

Actuators are distinguished as follows:

- Proportional actuators with constant actuating signal
  These elements set degrees of opening, angular positions or positions in proportion to the output value. The output value has an analog effect on the process within the control range. Actuators in this group include spring-loaded pneumatic drives, as well as motorized drives with position feedback for which a position control system is formed.
  An continuous controller, such as PID_Compact, generates the output value.
- Proportional actuators with pulse-width modulated signal
  These actuators are used to generate the output of pulses with a length proportional to the output value within the sampling time intervals. The actuator - e.g. a heating resistor or cooling apparatus - is switched on in isochronous mode for durations that differ depending on the output value.
  The actuating signal can assume unipolar "On" or "Off" states, or represent bipolar states such as "open/close", "forward/backward", "accelerate/brake".
  The output value is generated by a two-step controller such as PID_Compact with pulse-width modulation.
- Actuators with integral action and three-step actuating signal
  Actuators are frequently operated by motors with an on period that is proportional to the actuator travel of the choke element. This includes elements such as valves, shutters, and gate valves. In spite of their different design, all of these actuators follow the effect of an integral action at the input of the controlled system.
  A step controller, such as PID_3Step. generates the output value.

13.1.1.2 Controlled systems

The properties of a controlled system can hardly be influenced as these are determined by the technical requirements of the process and machinery. Acceptable control results can only be achieved by selecting a suitable controller type for the specific controlled system and adapting the controller to the time response of the controlled system. Therefore, it is is indispensable for the configuration of the proportional, integral and derivative actions of the controller to have precise knowledge of the type and parameters of the controlled system.

Controlled system types

Controlled systems are classified based on their time response to step changes of the output value.

We distinguish between the following controlled systems:

- Self-regulating controlled systems
  - Proportional-action controlled systems
  - PT1 controlled systems
  - PT2 controlled systems
- Non-self-regulating controlled systems
- Controlled systems with and without dead time

Self-regulating controlled systems

Proportional-action controlled systems

In proportional-action controlled systems, the process value follows the output value almost immediately. The ratio between the process value and output value is defined by the proportional Gain of the controlled system.

Examples:

- Gate valve in a piping system
- Voltage dividers
- Step-down function in hydraulic systems

PT1 controlled systems

In a PT1 controlled system, the process value initially changes in proportion to the change of the output value. The rate of change of the process value is reduced as a function of the time until the end value is reached, i.e., it is delayed.

Examples:

- Spring damping system
- Charge of RC elements
- Water container that is heated with steam.

The time constants are often identical for heating and cooling processes, or for charging and discharge characteristics. With different time constants, controlling is clearly more complex.

PT2 controlled systems

In a PT2 controlled system, the process value does not immediately follow a step change of the output value, i.e., it increases in proportion to the positive rate of rise and then approaches the setpoint at a decreasing rate of rise. The controlled system shows a proportional response characteristic with second order delay element.

Examples:

- Pressure control
- Flow rate control
- Temperature control

Non-self-regulating controlled systems

Non-self-regulating controlled systems have an integral response. The process value approaches an infinite maximum value.

Example:

- Liquid flow into a container

Controlled systems with dead time

A dead time always represents the runtime or transport time that has to expire before a change to the system input can be measured at the system output.

In controlled systems with dead time, the process value change is delayed by the amount of the dead time.

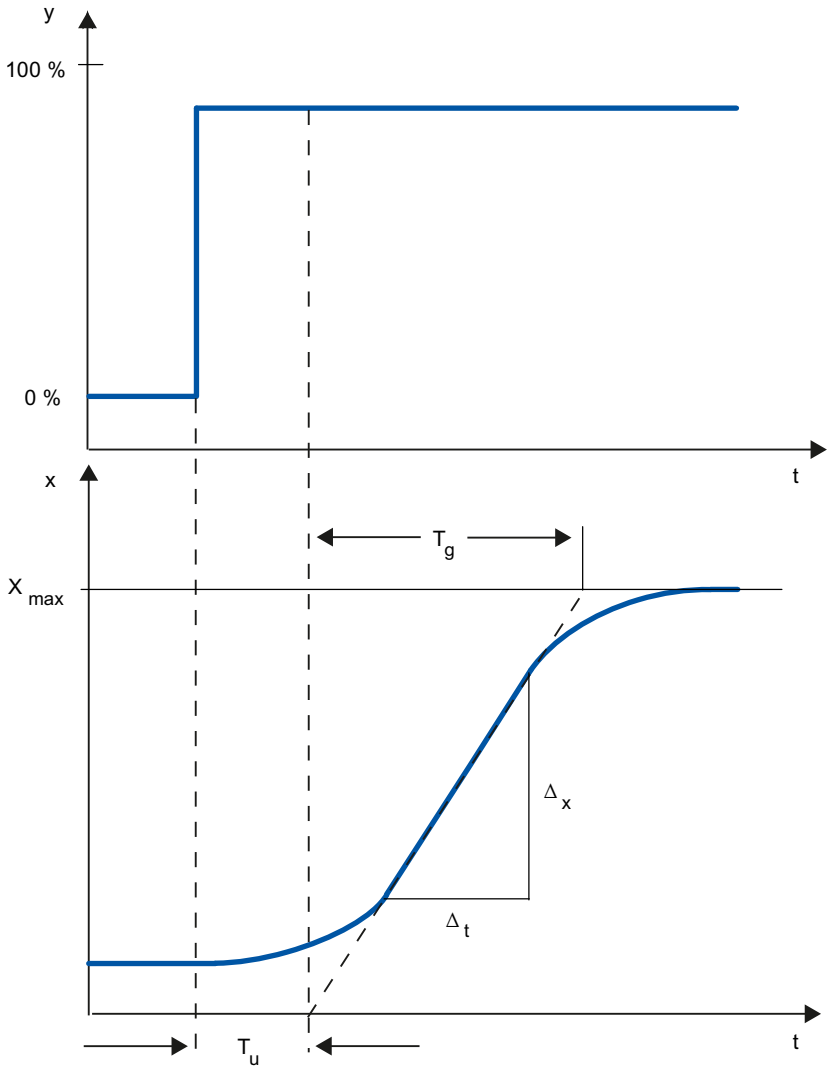
Example:

Conveyor

13.1.1.3 Characteristic values of the control section

Determining the time response from the step response

Time response of the controlled system can be determined based on the time characteristic of process value x following a step change of output value y. Most controlled systems are self-regulating controlled systems.



The time response can be determined by approximation using the variables Delay time T_u, Recovery time T_g and Maximum value X_{max}. The variables are determined by applying tangents

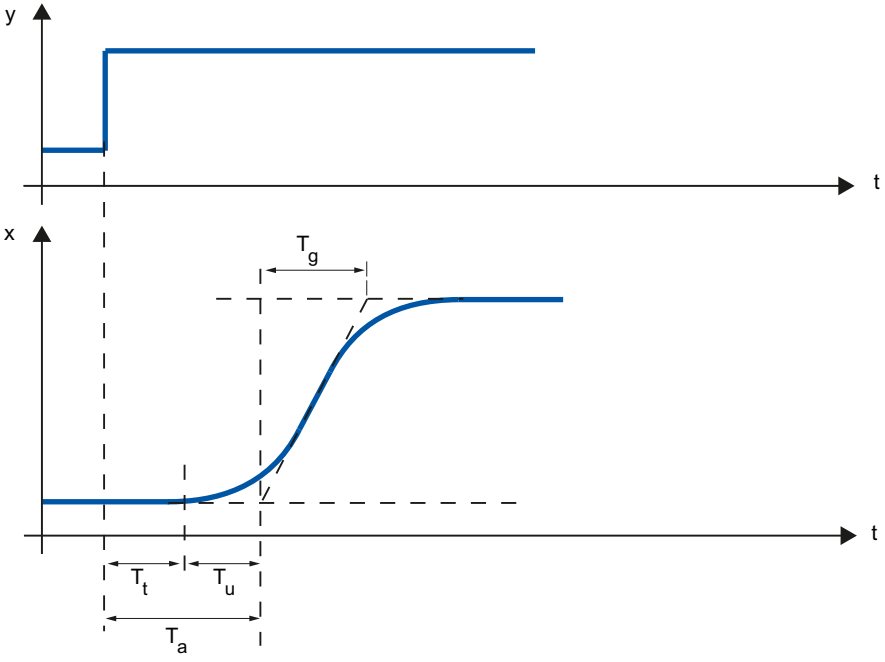
to the maximum value and the inflection point of the step response. In many situations, it is not possible to record the response characteristic up to the maximum value because the process value cannot exceed specific values. In this case, the rate of rise v_{max} is used to identify the controlled system ($v_{max} = \Delta_x/\Delta_t$).

The controllability of the controlled system can be estimated based on the ratio T_u/T_g, or $T_u \times v_{max}/X_{max}$. Rule:

Process type	T_u / T_g	Suitability of the controlled system for controlling
I	< 0,1	can be controlled well
II	0.1 to 0.3	can still be controlled
III	> 0,3	difficult to control

Influence of the dead time on the controllability of a controlled system

A controlled system with dead time and recovery reacts as follows to a jump of the output value.



T_t	Dead time
T_u	Delay time
T_g	Recovery time
y	Output value
x	Process value

The controllability of a self-regulating controlled system with dead time is determined by the ratio of T_t to T_g. T_t must be small compared to T_g. Rule:

$T_t/T_g \leq 1$

Response rate of controlled systems

Controlled systems can be judged on the basis of the following values:

T_u < 0.5 min, T_g < 5 min = fast controlled system

T_u > 0.5 min, T_g > 5 min = slow controlled system

Parameters of certain controlled systems

Physical quantity	Controlled system	Delay time T_u	Recovery time T_g	Rate of rise v_{max}
Temperature	Small electrically heated furnace	0.5 to 1 min	5 to 15 min	Up to 60 K/min.
	Large electrically heated annealing furnace	1 to 5 min	10 to 20 min	Up to 20 K/min.
	Large gas-heated annealing furnace	0.2 to 5 min	3 to 60 min	1 to 30 K/min
	Distillation tower	1 to 7 min	40 to 60 min	0.1 to 0.5° C/s
	Autoclaves (2.5 m^3)	0.5 to 0.7 min	10 to 20 min	Not specified
	High-pressure autoclaves	12 to 15 min	200 to 300 min	Not specified
	Steam superheater	30 s to 2.5 min	1 to 4 min	2°C/s
	Injection molding machines	0.5 to 3 min	3 to 30 min	5 to 20 K/min
	Extruders	1 to 6 min	5 to 60 min	
	Packaging machines	0.5 to 4 min	3 to 40 min	2 to 35 K/min
	Room heating	1 to 5 min	10 to 60 min	1° C/min
Flow rate	Pipeline with gas	0 to 5 s	0.2 to 10 s	Not relevant
	Pipeline with liquid	None	None	
Pressure	Gas pipeline	None	0.1 s	Not relevant
	Drum boiler with gas or oil firing	None	150 s	Not relevant
	Drum boiler with impact grinding mills	1 to 2 min	2 to 5 min	Not relevant
Vessel level	Drum boiler	0.6 to 1 min	Not specified	0.1 to 0.3 cm/s
Speed	Small electric drive	None	0.2 to 10 s	Not relevant
	Large electric drive	None	5 to 40 s	Not relevant
	Steam turbine	None	Not specified	50 min^{-1}
Voltage	Small generators	None	1 to 5 s	Not relevant
	Large generators	None	5 to 10 s	Not relevant

13.1.1.4 Pulse controller

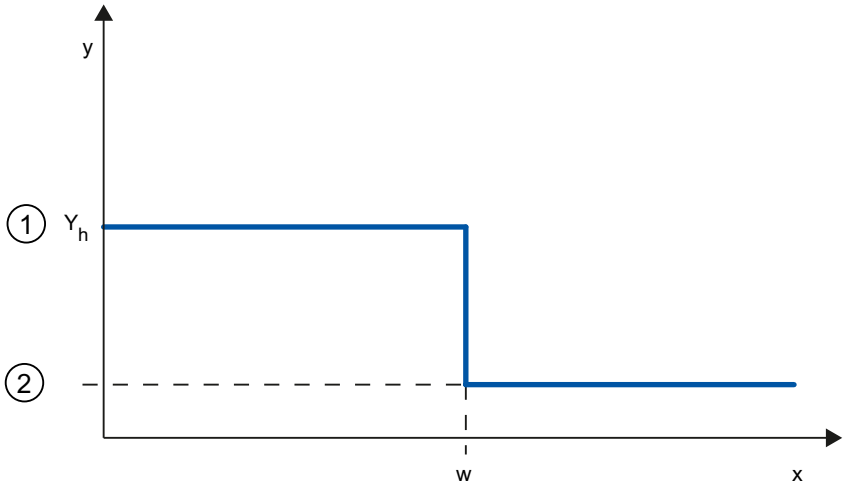
Two-step controllers without feedback

Two-step controllers have the state "ON" and "OFF" as the switching function. This corresponds to 100% or 0% output. This behavior generates a sustained oscillation of process value x around setpoint w.

The amplitude and duration of the oscillation increase in proportion to the ratio between the delay time T_u and recovery time T_g of the controlled system. These controllers are used mainly

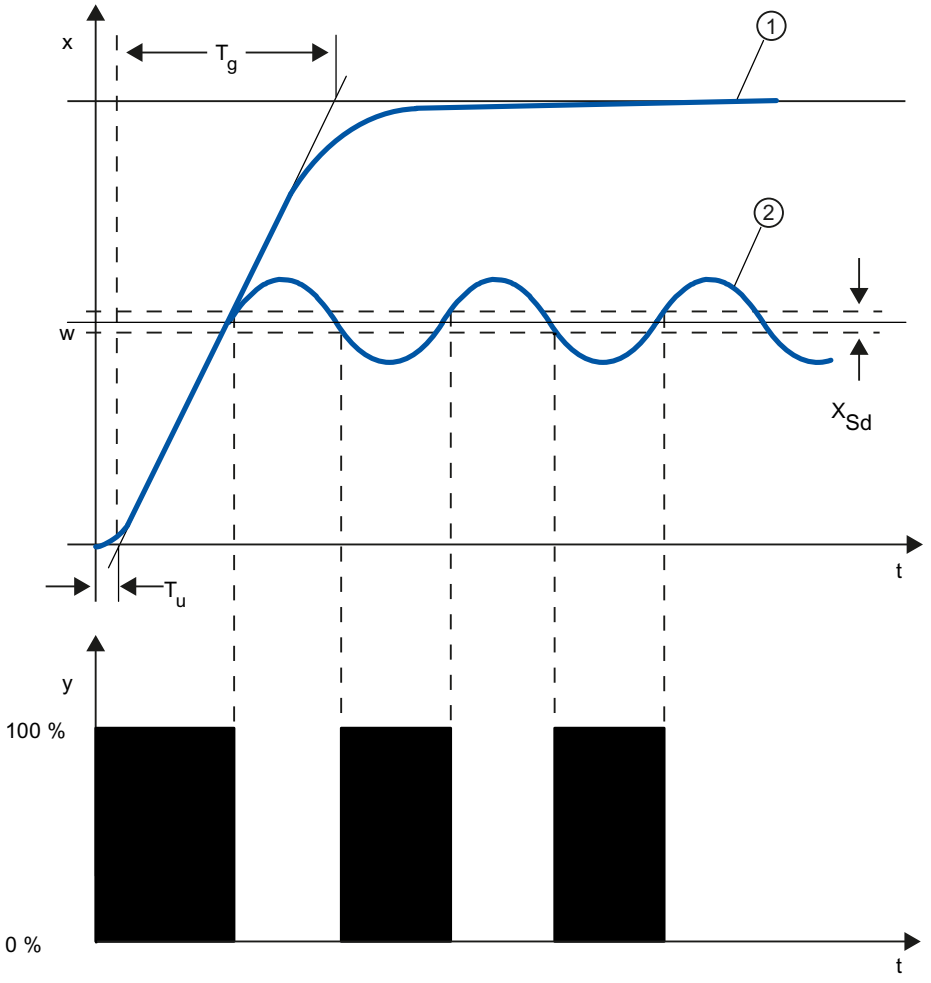
for simple temperature control systems (such as electrically directly heated furnaces) or as limit-value signaling units.

The following diagram shows the characteristic of a two-step controller



①	ON
②	OFF
Y_h	Control range
w	Setpoint

The following diagram shows the control function of a two-step controller



①	Response characteristic without controller
②	Response characteristic with two-step controller
T_u	Delay time
T_g	Recovery time
X_{Sd}	Switching difference

Two-step controllers with feedback

The behavior of two-step controllers in the case of controlled systems with larger delay times, such as furnaces where the functional space is separated from the heating, can be improved by the use of electronic feedback.

The feedback is used to increase the switching frequency of the controller, which reduces the amplitude of the process value. In addition, the control-action results can be improved substantially in dynamic operation. The limit for the switching frequency is set by the output level. It should not exceed 1 to 5 switches per minute at mechanical actuators, such as relays and contactors. In the case of voltage and current outputs with downstream thyristor or Triac controllers high switching frequencies can be selected that exceed the limit frequency of the controlled system by far.

Since the switching pulses can no longer be determined at the output of the controlled system, results comparable with those of continuous controllers are obtained.

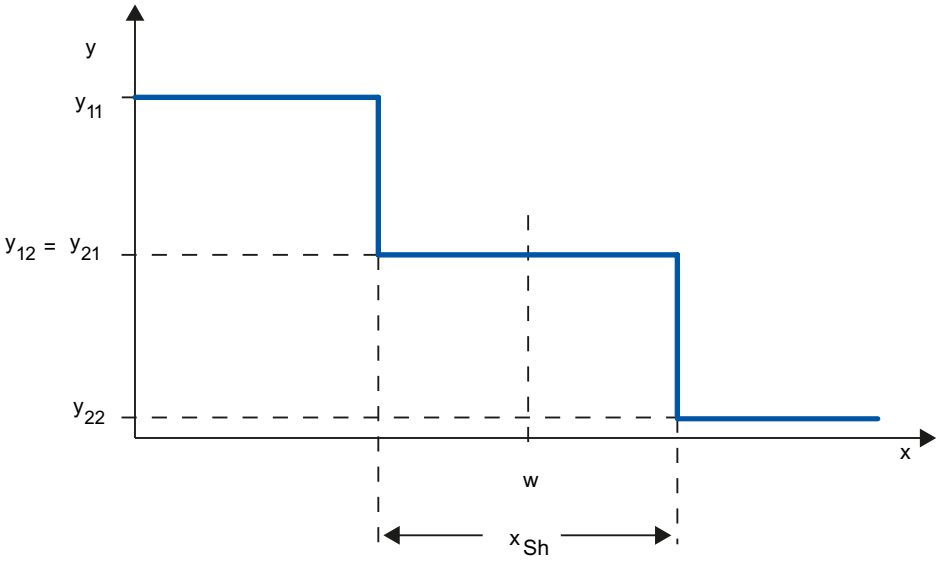
The output value is generated by pulse-width modulation of the output value of a continuous controller.

Two-step controllers with feedback are used for temperature control in furnaces, at processing machines in the plastics, textile, paper, rubber and foodstuff industries as well as for heating and cooling devices.

Three-step controllers

Three-step controllers are used for heating / cooling. These controllers have two switching points as their output. The control-action results are optimized through electronic feedback structures. Fields of applications for such controllers are heating, low-temperature, climatic chambers and tool heating units for plastic-processing machines.

The following diagram shows the characteristic of a three-step controller

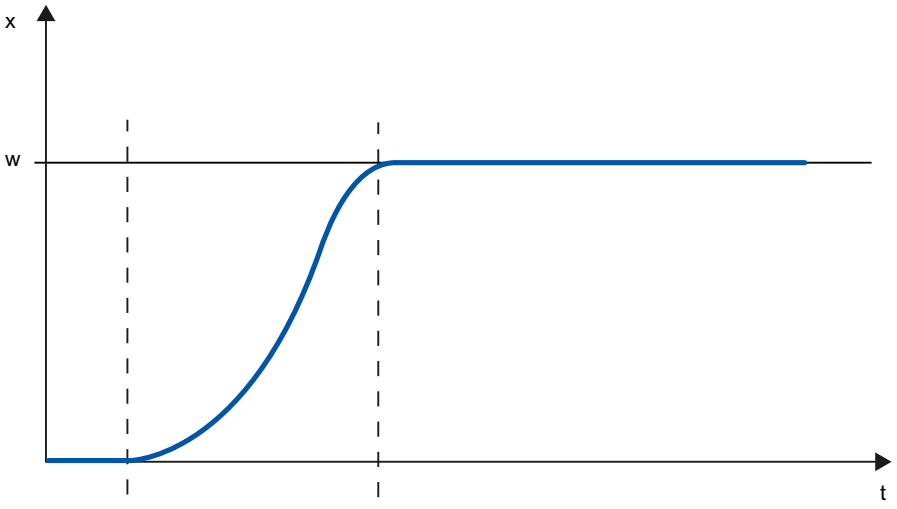


y	Output value, e.g. y11 = 100% heating y12 = 0% heating y21 = 0% cooling y22 = 100% cooling
x	Physical quantity of the process value, e.g., temperature in° C
w	Setpoint
x_{Sh}	Distance between Switching Point 1 and Switching Point 2

13.1.1.5 Response to setpoint changes and disturbances

Response to setpoint changes

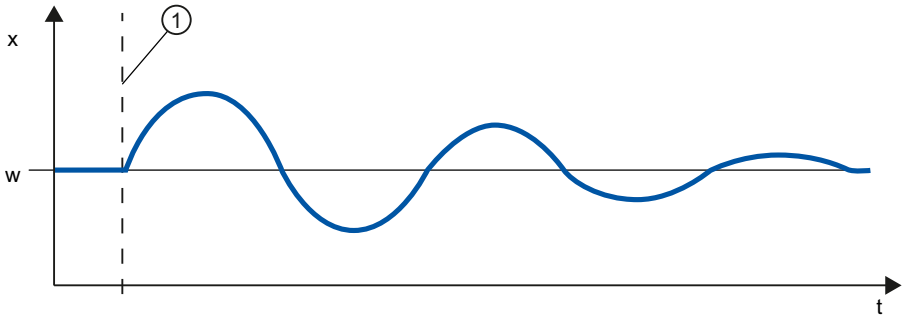
The process value should follow a setpoint change as quickly as possible. The response to setpoint changes is improved by minimizing fluctuation of the process value and the time required to reach the new setpoint.



x	Process value
w	Setpoint

Response to disturbances

The setpoint is influenced by disturbance variables. The controller has to eliminate the resulting control deviations in the shortest time possible. The response to disturbances is improved by minimizing fluctuation of the process value and the time required to reach the new setpoint.



x	Process value
w	Setpoint
①	Influencing a disturbance variable

Disturbance variables are corrected by a controller with integral action. A persistent disturbance variable does not reduce control quality because the control deviation is relatively constant. Dynamic disturbance variables have a more significant impact on control quality because of control deviation fluctuation. The control deviation is eliminated again only by means of the slow acting integral action.

A measurable disturbance variable can be included in the controlled system. This inclusion would significantly accelerated the response of the controller.

13.1.1.6 Control Response at Different Feedback Structures

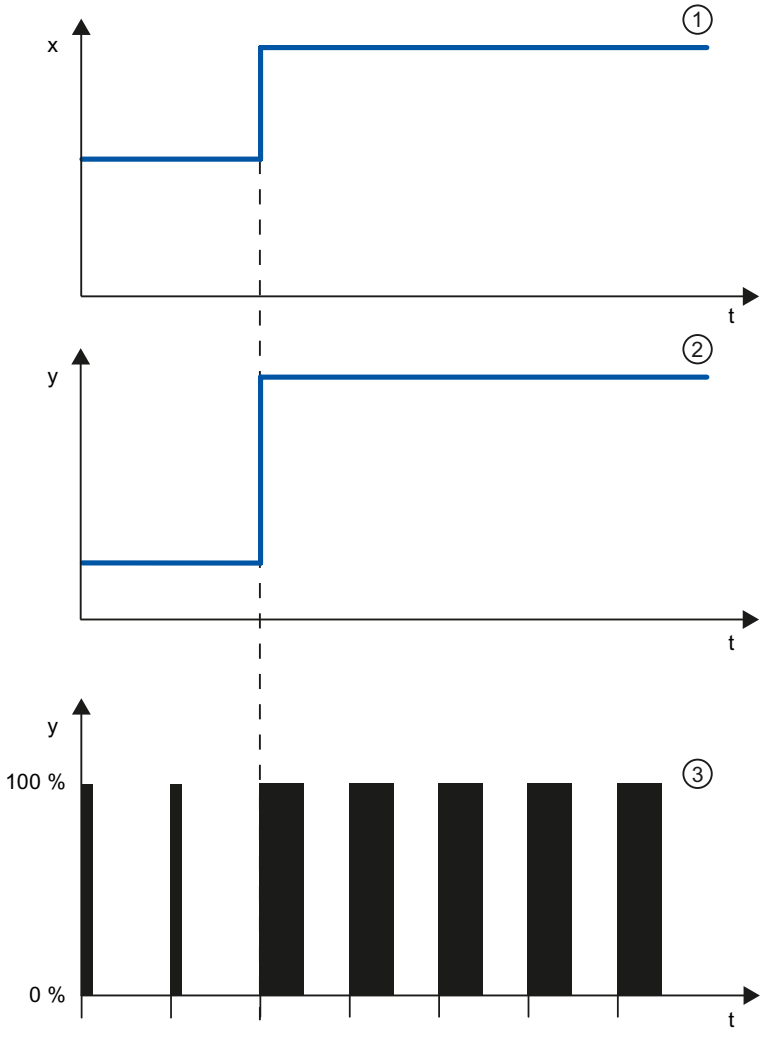
Control behavior of controllers

A precise adaptation of the controller to the time response of the controlled system is decisive for the controller's precise settling to the setpoint and optimum response to disturbance variables.

The feedback circuit can have a proportional action (P), proportional-derivative action (PD), proportional-integral action (PI), or proportional-integral-derivative action (PID).

If step functions are to be triggered by control deviations, the step responses of the controllers differ depending on their type.

Step response of a proportional action controller



① Control deviation
② Output value of a continuous controller
③ Output value of a pulse controller

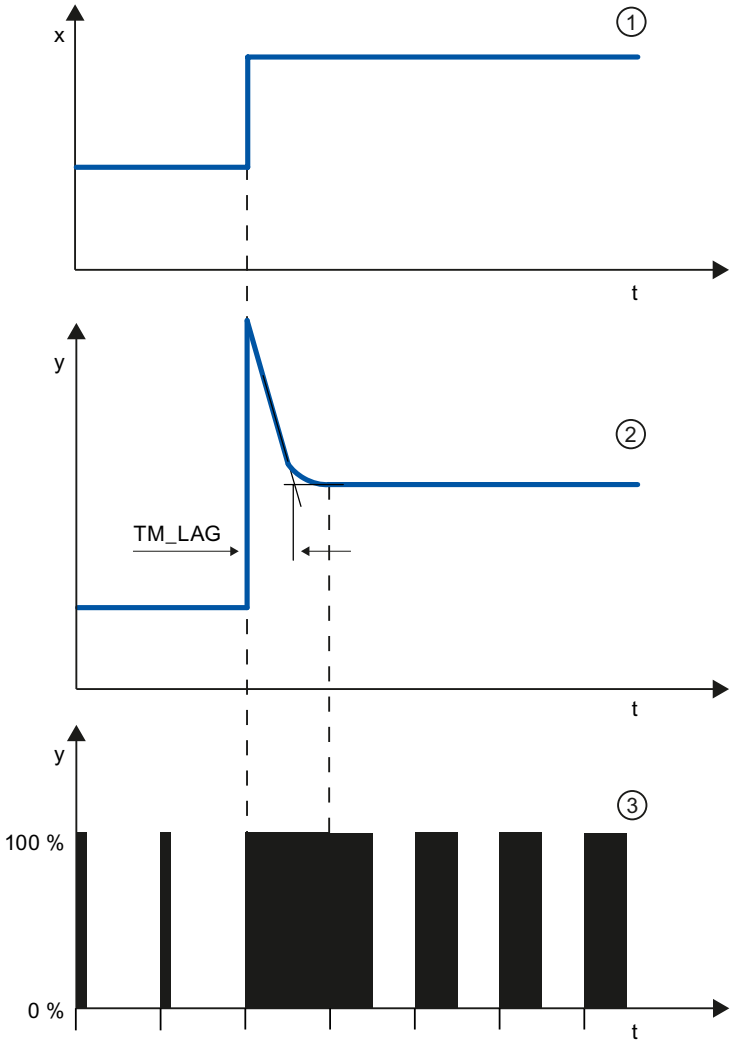
Equation for proportional action controller

Output value and control deviation are directly proportional, meaning:

Output value = proportional gain × control deviation

$y = GAIN \times x$

Step response of a PD-action controller



①	Control deviation
②	Output value of a continuous controller
③	Output value of a pulse controller
TM_LAG	Delay of the Derivative action

Equation for PD-action controller

The following applies for the step response of the PD-action controller in the time range:

$$y = GAIN \cdot X_W \cdot \left(1 + \frac{TD}{TM_LAG} \cdot e^{\frac{-t}{TM_LAG}} \right)$$

t = time interval since the step of the control deviation

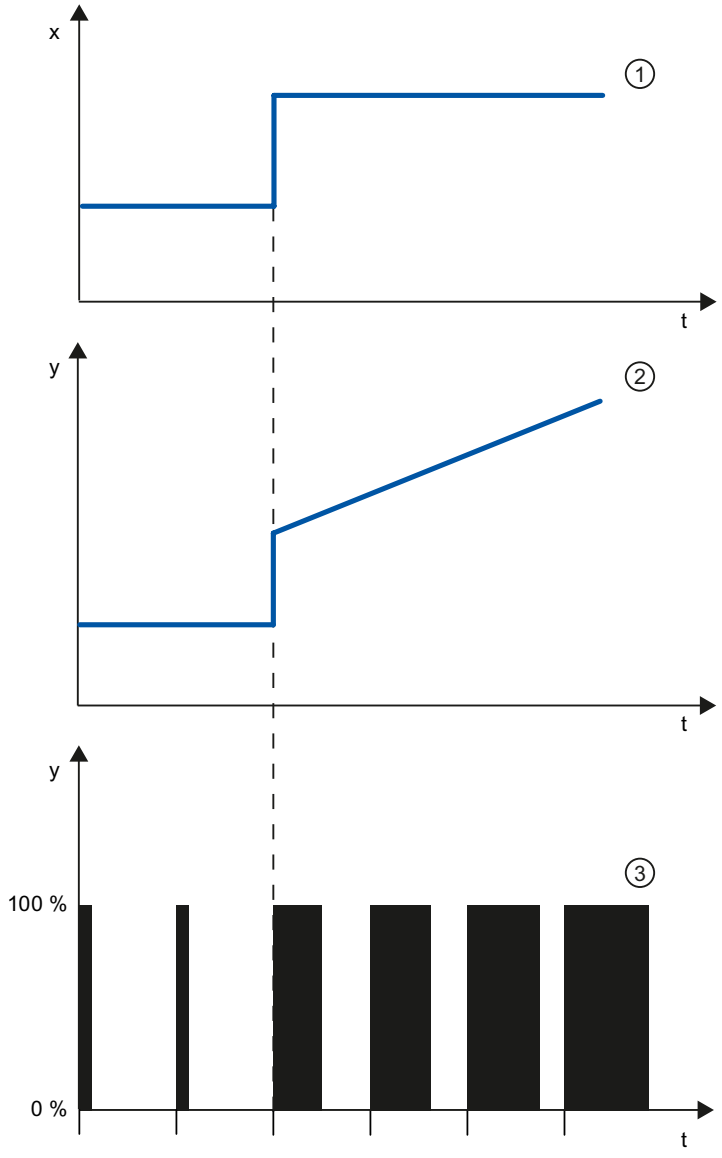
The derivative action generates a output value as a function of the rate of change of the process value. A derivative action by itself is not suitable for controlling because the output value only

follows a step of the process value. As long as the process value remains constant, the output value will no longer change.

The response to disturbances of the derivative action is improved in combination with a proportional action. Disturbances are not corrected completely. The good dynamic response is advantageous. A well attenuated, non-oscillating response is achieved during approach and setpoint change.

A controller with derivative action is not appropriate if a controlled system has pulsing measured quantities, for example, in the case of pressure or flow control systems.

Step response of a PI-action controller



① Control deviation
② Output value of a continuous controller
③ Output value of a pulse controller

An integral action in the controller adds the control deviation as a function of the time. This means that the controller corrects the system until the control deviation is eliminated. A sustained control deviation is generated at controllers with proportional action only. This effect can be eliminated by means of an integral action in the controller.

In practical experience, a combination of the proportional, integral and derivative actions is ideal, depending on the requirements placed on the control response. The time response of the individual components can be described by the controller parameters proportional gain GAIN, integral action time TI (integral action), and derivative action time TD (derivative action).

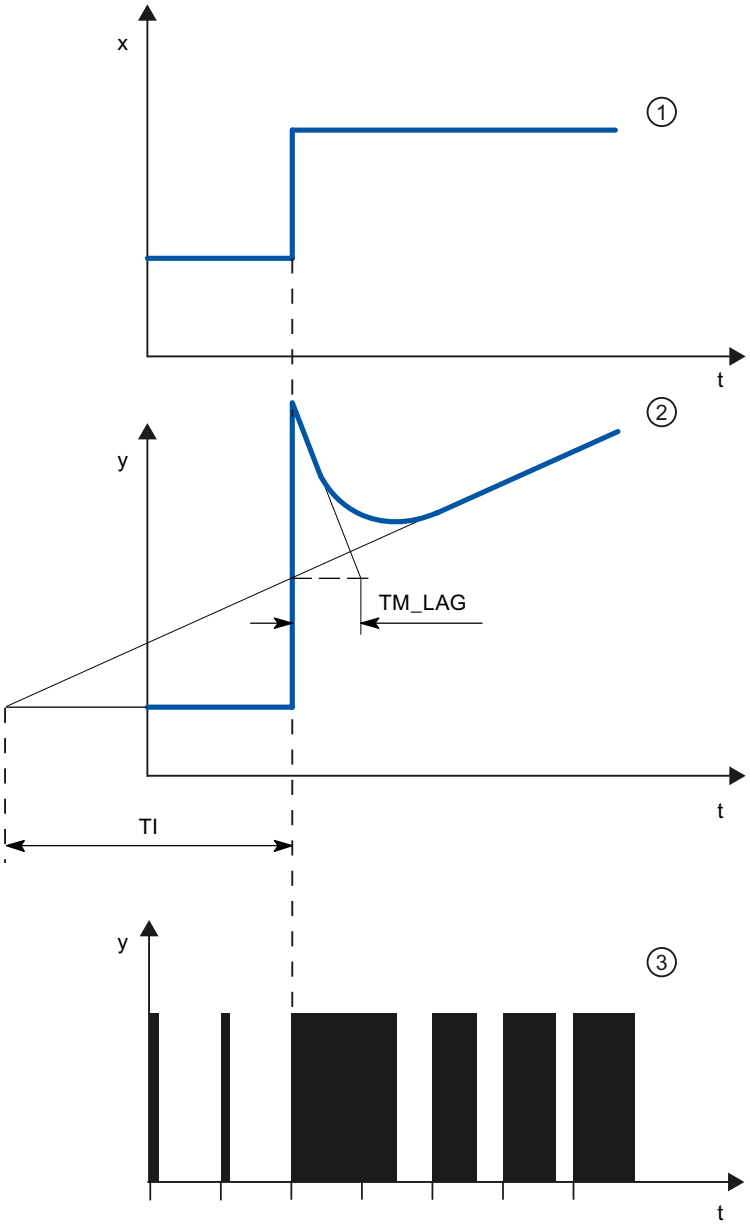
Equation for PI-action controller

The following applies for the step response of the PI-action controller in the time range:

$$y = GAIN \cdot X_W \cdot \left(1 + \frac{1}{TI \cdot t}\right)$$

t = time interval since the step of the control deviation

Step response of a PID controller



①	Control deviation
②	Output value of a continuous controller
③	Output value of a pulse controller
TM_LAG	Delay of the Derivative action
T_i	Integral action time

Equation for PID controller

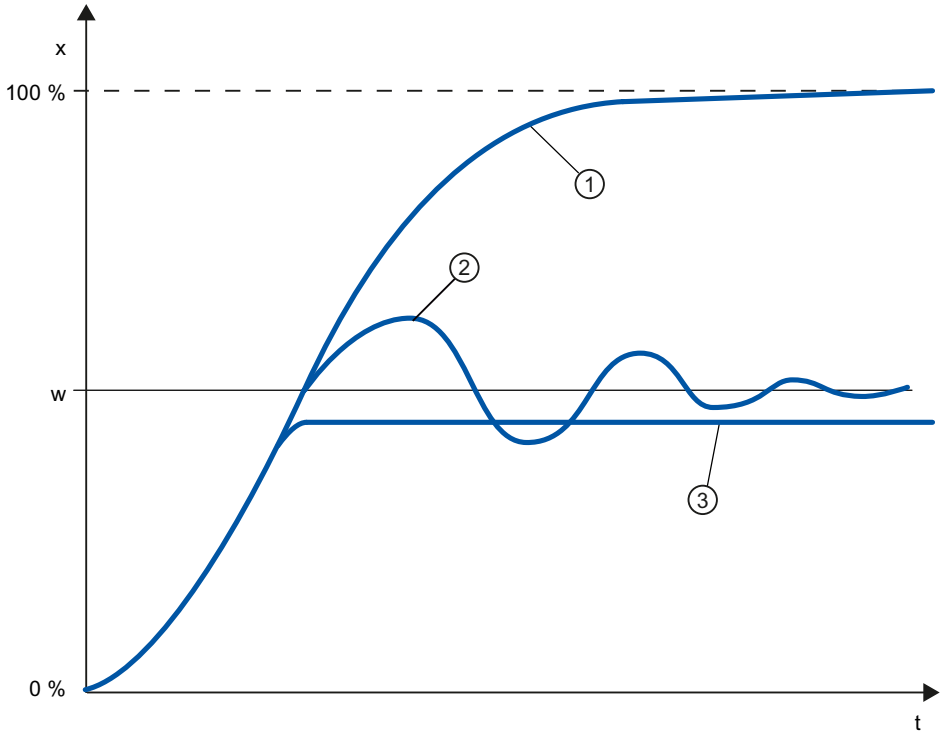
The following applies for the step response of the PID controller in the time range:

$$y = GAIN \cdot X_w \cdot \left(1 + \frac{1}{TI \cdot t} + \frac{TD}{TM_LAG} \cdot e^{\frac{-t}{TM_LAG}} \right)$$

t = time interval since the step of the control deviation

Response of a controlled system with different controller structures

Most of the controller systems occurring in process engineering can be controlled by means of a controller with PI-action response. In the case of slow controlled system with a large dead time, for example temperature control systems, the control result can be improved by means of a controller with PID action.



①	No controller
②	PID controller
③	PD-action controller
w	Setpoint
x	Process value

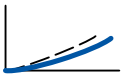
Controllers with PI and PID action have the advantage that the process value does not have any deviation from the setpoint value after settling. The process value oscillates over the setpoint during approach.

13.1.1.7 Selection of the controller structure for specified controlled systems

Selection of the Suitable Controller Structures

To achieve optimum control results, select a controller structure that is suitable for the controlled system and that you can adapt to the controlled system within specific limits.

The table below provides an overview of suitable combinations of a controller structure and controlled system.

Controlled system		Controller structure			
		P	PD	PI	PID
	With dead time only	Unsuitable	Unsuitable	Suitable	Unsuitable
	PT1 with dead time	Unsuitable	Unsuitable	Well suited	Well suited
	PT2 with dead time	Unsuitable	Suited conditionally	Well suited	Well suited
	Higher order	Unsuitable	Unsuitable	Suited conditionally	Well suited
	Not self-regulating	Well suited	Well suited	Well suited	Well suited

The table below provides an overview of suitable combinations of a controller structure and physical quantity.

Physical quantity	Controller structure			
	P	PD	PI	PID
	Sustained control deviation		No sustained control deviation	
Temperature	For low performance requirements and proportional action controlled systems with $T_u/Tg < 0,1$	Well suited	The most suitable controller structures for high performance requirements (except for specially adapted special controllers)	
Pressure	Suitable, if the delay time is inconsiderable	Unsuitable	The most suitable controller structures for high performance requirements (except for specially adapted special controllers)	
Flow rate	Unsuitable, because required GAIN range is usually too large	Unsuitable	Suitable, but integral action controller alone often better	Hardly required

13.1.1.8 PID parameter settings

Rule of Thumb for the Parameter Setting

Controller structure	Setting
P	GAIN ≈ v_{max} × T_u [° C]
PI	GAIN ≈ 1.2 × v_{max} × T_u [° C] TI ≈ 4 × T_u [min]
PD	GAIN ≈ 0.83 × v_{max} × T_u [° C] TD ≈ 0.25 × v_{max} × T_u [min] TM_LAG ≈ 0.5 × TD[min]
PID	GAIN ≈ 0.83 × v_{max} × T_u [° C] TI ≈ 2 × T_u [min] TD ≈ 0.4 × T_u [min] TM_LAG ≈ 0.5 × TD[min]
PD/PID	GAIN ≈ 0.4 × v_{max} × T_u [° C] TI ≈ 2 × T_u [min] TD ≈ 0.4 × T_u [min] TM_LAG ≈ 0.5 × TD[min]

Instead of $v_{max} = \Delta_x / \Delta_t$, you can use X_{max} / T_g.

In the case of controllers with PID structure the setting of the integral action time and differential-action time is usually coupled with each other.

The ratio TI / TD lies between 4 and 5 and is optimal for most controlled systems.

Non-observance of the differential-action time TD is uncritical at PD controllers.

In the case of PI and PID controllers, control oscillations occur if the integral action time TI has been select by more than half too small.

An integral action time that is too large slows down the settling times of disturbances. One cannot expect that the control loops operate "optimally" after the first parameter settings. Experience shows that adjusting is always necessary, when a system exists that is "difficult to control" with $T_u / T_g > 0.3$.

13.1.2 Configuring a software controller

13.1.2.1 Overview of software controller

For the configuration of a software controller, you need an instruction with the control algorithm and a technology object. The technology object for a software controller corresponds with the instance DB of the instruction. The configuration of the controller is saved in the technology object. In contrast to the instance DBs of other instructions, technology objects are not stored for the program resources, but rather under CPU > Technology objects.

Technology objects and instructions

CPU	Library	Instruction	Technology object	Description
S7-1200	Compact PID	PID_Compact V1.x	PID_Compact V1.x	Universal PID controller with integrated tuning
S7-1200		PID_3Step V1.x	PID_3Step V1.x	PID controller with integrated tuning for valves
S7-1500 S7-1200 V4.x		PID_Compact V2.x	PID_Compact V2.x	Universal PID controller with integrated tuning
S7-1500 S7-1200 V4.x		PID_3Step V2.x	PID_3Step V2.x	PID controller with integrated tuning for valves
S7-1500 ≥ V1.7 S7-1200 ≥ V4.1		PID_Temp V1.x	PID_Temp V1.x	Universal PID temperature controller with integrated tuning
S7-1500/300/400	PID basic functions	CONT_C	CONT_C	Continuous controller
S7-1500/300/400		CONT_S	CONT_S	Step controller for actuators with integrating behavior
S7-1500/300/400		PULSEGEN	-	Pulse generator for actuators with proportional behavior
S7-1500/300/400		TCONT_CP	TCONT_CP	Continuous temperature controller with pulse generator
S7-1500/300/400		TCONT_S	TCONT_S	Temperature controller for actuators with integrating behavior
S7-300/400	PID Self Tuner	TUN_EC	TUN_EC	Optimization of a continuous controller
S7-300/400		TUN_ES	TUN_ES	Optimization of a step controller
S7-300/400	Standard PID Control (PID Professional optional package)	PID_CP	PID_CP	Continuous controller with pulse generator
S7-300/400		PID_ES	PID_ES	Step controller for actuators with integrating behavior
S7-300/400		LP_SCHED	-	Distribute controller calls

CPU	Library	Instruction	Technology object	Description
S7-300/400	Modular PID Control (PID Professional optional package)	A_DEAD_B	-	Filter interfering signal from control deviation
S7-300/400		CRP_IN	-	Scale analog input signal
S7-300/400		CRP_OUT	-	Scale analog output signal
S7-300/400		DEAD_T	-	Delay output of input signal
S7-300/400		DEADBAND	-	Suppress small fluctuations to the process value
S7-300/400		DIF	-	Differentiate input signals over time
S7-300/400		ERR_MON		Monitor control deviation
S7-300/400		INTEG	-	Integrate input signals over time
S7-300/400		LAG1ST	-	First-order delay element
S7-300/400		LAG2ND	-	Second-order delay element
S7-300/400		LIMALARM	-	Report limit values
S7-300/400		LIMITER	-	Limiting the manipulated variable
S7-300/400		LMNGEN_C	-	Determine manipulated variable for continuous controller
S7-300/400		LMNGEN_S	-	Determine manipulated variable for step controller
S7-300/400		NONLIN	-	Linearize encoder signal
S7-300/400		NORM	-	Scale process value physically
S7-300/400		OVERRIDE	-	Switch manipulated variable from 2 PID controllers to 1 actuator
S7-300/400		PARA_CTL	-	Switch parameter sets
S7-300/400		PID	-	PID algorithm
S7-300/400		PUSLEGEN_M	-	Generate pulse for proportional actuators
S7-300/400		RMP_SOAK	-	Specify setpoint according to ramp / soak
S7-300/400		ROC_LIM	-	Limit rate of change
S7-300/400		SCALE_M	-	Scale process value
S7-300/400		SP_GEN	-	Specify setpoint manually
S7-300/400		SPLT_RAN	-	Split manipulated variable range
S7-300/400		SWITCH	-	Switch analog values
S7-300/400		LP_SCHED_M	-	Distribute controller calls

13.1.2.2 Steps for the configuration of a software controller

All SW-controllers are configured according to the same scheme:

Step	Description
1	Add technology object (Page 5997)
2	Configure technology object (Page 5998)
3	Call instruction in the user program (Page 5999)
4	Download technology object to device (Page 6000)
5	Commission software controller (Page 6001)
6	Save optimized PID parameters in the project (Page 6001)

Step	Description
7	Comparing values (Page 6003)
8	Display instances of a technology object (Page 6023)

13.1.2.3 Add technology objects

Add technology object in the project navigator

When a technology object is added, an instance DB is created for the instruction of this technology object. The configuration of the technology object is stored in this instance DB.

Requirement

A project with a CPU has been created.

Procedure

To add a technology object, proceed as follows:

1. Open the CPU folder in the project tree.
2. Open the "Technology objects" folder.
3. Double-click "Add new object".
   The "Add new object" dialog box opens.
4. Click on the "PID" button.
   All available PID-controllers for this CPU are displayed.
5. Select the instruction for the technology object, for example, PID_Compact.
6. Enter an individual name for the technology object in the "Name" input field.
7. Select the "Manual" option if you want to change the suggested data block number of the instance DB.
8. Click "Further information" if you want to add own information to the technology object.
9. Confirm with "OK".

Result

The new technology object has been created and stored in the project tree in the "Technology objects" folder. The technology object is used if the instruction for this technology object is called in a cyclic interrupt OB.

Note

You can select the "Add new and open" check box at the bottom of the dialog box. This opens the configuration of the technology object after adding has been completed.

13.1.2.4 Configure technology objects

The properties of a technology object on a S7-1200 CPU can be configured in two ways.

- In the Inspector window of the programming editor
- In the configuration editor

The properties of a technology object on a S7-300/400 CPU can only be configured in the configuration editor.

Inspector window of the programming editor

In the Inspector window of the programming editor you can only configure the parameters required for operation.

The offline values of the parameters are also shown in online mode. You can only change the online values in the commissioning window.

To open the Inspector window of the technology object, follow these steps:

1. Open the "Program blocks" folder in the project tree.
2. Double click the block (cyclic interrupt OB) in which you open the instruction of the SW-controller.
   The block is opened in the work area.
3. Click on the instruction of the SW-controller.
4. In the Inspector window, select the "Properties" and "Configuration" tabs consecutively.

Configuration window

For each technology object, there is a specific configuration window in which you can configure all properties.

To open the configuration window of the technology object, follow these steps:

1. Open the "Technology objects" folder in the project tree.
2. Open the technology object in the project tree.
3. Double-click the "Configuration" object.

Symbols

Icons in the area navigation of the configuration and in the Inspector window show additional details about the completeness of the configuration:

	The configuration contains default values and is complete. The configuration exclusively contains default values. With these default values the use of the technology object is possible without further changes.
	The configuration contains user-defined or automatically adjusted values and is complete All input fields of the configuration contain valid values and at least one default setting was changed.
	The configuration is incomplete or faulty At least one input field or a collapsible list contains no or one invalid value. The corresponding field or the drop-down list box has a red background. When clicked, the roll-out error message indicates the cause of the error.

The properties of a technology object are described in detail in the section for the technology object.

13.1.2.5 Call instruction in the user program

The instruction of the software controller must be called in a cyclic interrupt OB. The sampling time of the software controller is determined by the interval between the calls in the cyclic interrupt OB.

Requirement

The cyclic interrupt OB is created and the cycle time of the cyclic interrupt OB is correctly configured.

Procedure

Proceed as follows to call the instruction in the user program:

1. Open the CPU folder in the project tree.
2. Open the "Program blocks" folder.
3. Double-click the cyclic interrupt OB.
   The block is opened in the work area.
4. Open the "Technology" group in the "Instructions" window and the "PID Control" folder.
   The folder contains all instructions for software controllers that can be configured on the CPU.
5. Select the instruction and drag it to your cyclic interrupt OB.
   The "Call options" dialog box opens.
6. Select a technology object or type the name for a new technology object from the "Name" list.

Result

If the technology object does not exist yet, it is added. The instruction is added in the cyclic interrupt OB. The technology object is assigned to this call of the instruction.

13.1.2.6 Downloading technology objects to device

A new or modified configuration of the technology object must be downloaded to the CPU for the online mode. The following characteristics apply when downloading retentive data:

- **Software (changes only)**
  - S7-1200, S7-1500:
    Retentive data is retained.
  - S7-300/400:
    Retentive data is updated immediately. CPU does not change to Stop.
- **Download PLC program to device and reset**
  - S7-1200, S7-1500:
    Retentive data is updated at the next change from Stop to RUN. The PLC program can only be downloaded completely.
  - S7-300/400:
    Retentive data is updated at the next change from Stop to RUN.

Downloading retentive data to an S7-1200 or S7-1500 CPU

Note

The download and reset of the PLC program during ongoing system operation can result in serious damages or injuries in the case of malfunctions or program errors.

Make sure that dangerous states cannot occur before you download and reset the PLC program.

Proceed as follows to download the retentive data:

1. Select the entry of the CPU in the project tree.
2. Select the command "Download and reset PLC program" from the "Online" menu.
   - If you have not established an online connection yet, the "Extended download" dialog opens. In this case, set all required parameters for the connection and click "Download".
   - If the online connection has been defined, the project data is compiled, if necessary, and the dialog "Load preview" opens. This dialog displays messages and recommends actions necessary for download.
3. Check the messages.
   As soon as download is possible, the "Download" button becomes active.
4. Click on "Download".
   The complete PLC program is downloaded and the "Load results" dialog opens. This dialog displays the status and the actions after the download.
5. If the modules are to restart immediately after the download, select the check box "Start all".
6. Close the dialog "Download results" with "Finish".

Result

The complete PLC program is downloaded to the device. Blocks that only exist online in the device are deleted. By downloading all affected blocks and by deleting any blocks in the device that are not required, you avoid inconsistencies between the blocks in the user program.

The messages under "Info > General" in the Inspector window indicate whether the download was successful.

13.1.2.7 Commissioning software controller

Procedure

To open the "Commissioning" work area of the technology object, follow these steps:

1. Open the "Technology objects" folder in the project tree.
2. Open the technology object in the project tree.
3. Double-click the "Commissioning" object.

The commissioning functions are specific for each controller and are described there.

13.1.2.8 Save optimized PID parameter in the project

The software controller is optimized in the CPU. Through this, the values in the instance-DB on the CPU no longer agree with those in the project.

To update the PID parameter in the project with the optimized PID parameters, proceed as follows:

Requirement

- An online connection to the CPU is established and the CPU is in "RUN" mode.
- The functions of the commissioning window have been enabled by means of the "Start" button.

Procedure

1. Open the CPU folder in the project tree.
2. Open the "Technology objects" folder.
3. Open a technology object.
4. Double click on "Commissioning".
5. Click on the  icon "Upload PID parameters".
6. Save the project.

Result

The currently active PID parameters are stored in the project data. When reloading the project data in the CPU, the optimized parameters are used.

13.1.2.9 Comparing values

Comparison display and boundary conditions

The "Compare values" function provides the following options:

- Comparison of configured start values of the project with the start values in the CPU and the actual values
- Direct editing of actual values and the start values of the project
- Immediate detection and display of input errors with suggested corrections
- Backup of actual values in the project
- Transfer of start values of the project to the CPU as actual values

Icons and operator controls

The following icons and operator controls are available:

Icon	Function
	Start value PLC matches the configured Start value project
	Start value PLC does not match the configured Start value project
	The comparison of the Start value PLC with the configured Start value project cannot be performed
	At least one of the two comparison values has a process-related or syntax error.
	Transfers actual values to the offline project
	Transfers updated start values in the project to the CPU (initialize setting values)
	Opens the "Compare values" dialog

Boundary conditions

The "Compare values" function is available for S7-1200 and S7-1500 without limitations.

The following limitation applies to S7-300 and S7-400:

In monitoring mode, an S7-300/S7-400 cannot transfer the start values to the CPU. These values cannot be displayed online with "Compare values".

The actual values of the technology object are displayed and can be changed directly.

Comparing values

The procedure is shown in the following using "PID Parameters" as an example.

Requirements

- A project with a software controller is configured.
- The project is downloaded to the CPU.
- The configuration dialog is open in the project navigator.

Procedure

1. Open the desired software controller in the project navigation.
2. Double-click the "Configuration" object.
3. Navigate within the configuration window to the "PID Parameters" dialog.
4. Click the  icon to activate monitoring mode.
   The icons and operator controls (Page 6002) of the "Compare values" function are shown behind the parameters.
5. Click the desired parameter in the input box and change the parameter values manually by entering them directly.
   - If the background of the input box is gray, this value is a read-only value and cannot be changed.
   - To change the values in the "PID Parameters" dialog, enable manual entry by selecting the "Enable manual entry" check box beforehand.
6. Click the  icon to open the dialog for the start values.
   This dialog indicates two values of the parameter:
   - Start value in CPU: The start value in the CPU is shown in the top part.
   - Start value in the project: The configured start value in the project is shown in the bottom part.
7. Enter the desired value in the input box for the project.

Error detection

The input of incorrect values is detected. Corrections are suggested in this case.

If you enter a value with incorrect syntax, a rollout containing the corresponding error message opens below the parameter. The incorrect value is not applied.

If you enter a value that is incorrect for the process, a dialog opens containing the error message and a suggested correction:

- Click "No" to accept this suggested correction and correct your input.
- Click "OK" to apply the incorrect value.

NOTICE
Malfunctions of the controller
Values incorrect for the process can result in controller malfunctions.

Backing up actual values

Click the icon to transfer the actual controller values to the start values of your configured project.

Transferring project values to the CPU

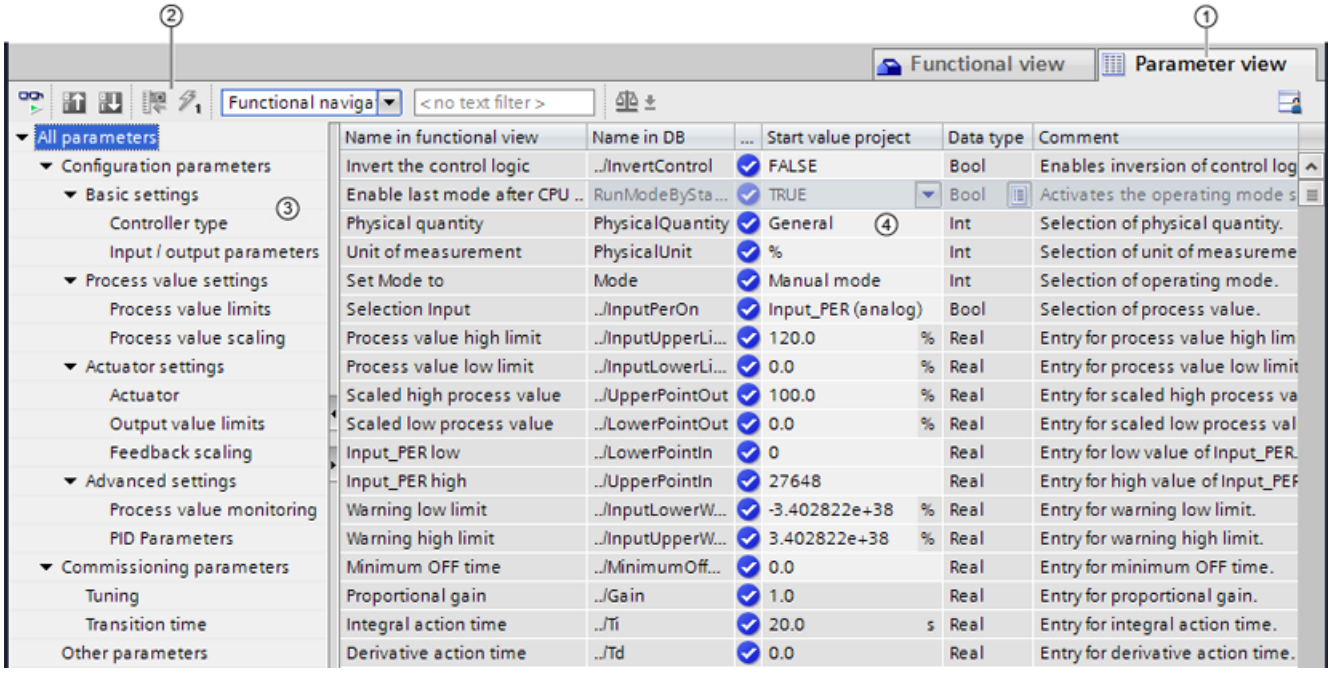
Click the icon to transfer the configured values of your project to the CPU.

CAUTION
Prevent personal injury and property damage!
Downloading and resetting of the user program while the plant is operating may result in significant property damage and severe personal injuries in the event of malfunctions or program errors.
Make sure that dangerous states cannot occur before you download and reset the user program.

13.1.2.10 Parameter view

Introduction to the parameter view

The Parameter view provides you with a general overview of all relevant parameters of a technology object. You obtain an overview of the parameter settings and can easily change them in offline and online mode.



① "Parameter view" tab

② Toolbar (Page 6007)

③ Navigation (Page 6007)

④ Parameter table (Page 6008)

Function scope

The following functions are available for analyzing the parameters of the technology objects and for enabling targeted monitoring and modification.

Display functions:

- Display of parameter values in offline and online mode
- Display of status information of the parameters
- Display of value deviations and option for direct correction
- Display of configuration errors
- Display of value changes as a result of parameter dependencies

- Display of all memory values of a parameter: Start value PLC, Start value project, Monitor value
- Display of the parameter comparison of the memory values of a parameter

Operator control functions:

- Navigation for quickly changing between the parameters and parameter structures.
- Text filter for faster searches for particular parameters.
- Sorting function for customizing the order of parameters and parameter groups to requirements.
- Memory function for backing up structural settings of the Parameter view.
- Monitoring and modifying of parameter values online.
- Change display format of value.
- Function for saving a snapshot of parameter values of the CPU in order to capture momentary situations and to respond to them.
- Function for applying a snapshot of parameter values as start values.
- Download of modified start values to the CPU.
- Comparison functions for comparing parameter values with one another.

Validity

The Parameter view described here is available for the following technology objects:

- PID_Compact
- PID_3Step
- PID_Temp
- CONT_C (S7-1500 only)
- CONT_S (S7-1500 only)
- TCONT_CP (S7-1500 only)
- TCONT_S (S7-1500 only)
- TO_Axis_PTO (S7-1200 Motion Control)
- TO_Positioning_Axis (S7-1200 Motion Control)
- TO_CommandTable_PTO (S7-1200 Motion Control)
- TO_CommandTable (S7-1200 Motion Control)

Structure of the parameter view

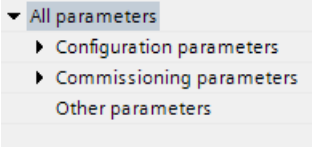
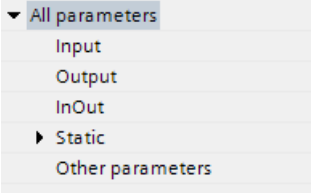
Toolbar

The following functions can be selected in the toolbar of the parameter view.

Icon	Function	Explanation
	Monitor all	Starts the monitoring of visible parameters in the active Parameter view (online mode).
	Create snapshot of monitor values and accept setpoints of this snapshot as start values	Applies the current monitor values to the “Snapshot” column and updates the start values in the project. Only in online mode for PID_Compact, PID_3Step and PID_Temp.
	Initialize setpoints	Transfers the start values updated in the project to the CPU. Only in online mode for PID_Compact, PID_3Step and PID_Temp.
	Create snapshot of monitor values	Applies the current monitor values to the “Snapshot” column. Only in online mode.
	Modify all selected parameters immediately and once	This command is executed once and as quickly as possible without reference to any particular point in the user program. Only in online mode.
	Select navigation structure	Toggles between functional navigation and data navigation.
	Text filter...	After entry of a character string: Display of all parameters containing the specified string in one of the currently visible columns.
	Selection of compare values	Selection of parameter values that are to be compared with one another in online mode (Start value project, Start value PLC, Snapshot) Only in online mode.
	Save window settings	Saves your display settings for the Parameter view (e.g., selected navigation structure, activated table columns, etc.)

Navigation

Within the "Parameter view" tab, the following alternative navigation structures can be selected.

Navigation		Explanation
Functional navigation		In the functional navigation, the structure of the parameters is based on the structure in the configuration dialog ("Functional view" tab), commissioning dialog, and diagnostics dialog. The last group "Other parameters" contains all other parameters of the technology object.
Data navigation		In the data navigation, the structure of the parameters is based on the structure in the instance DB / technology DB. The last group "Other parameters" contains the parameters that are not contained in the instance DB / technology DB.

You can use the "Select navigation structure" drop-down list to toggle the navigation structure.

Parameter table

The table below shows the meaning of the individual columns of the parameter table. You can show or hide the columns as required.

- Column "Offline" = X: Column is visible in offline mode.
- Column "Online" = X: Column is visible in online mode (online connection to the CPU).

Column	Explanation	Offline	Online
Name in functional view	Name of the parameter in the functional view. The display field is empty for parameters that are not configured via the technology object.	X	X
Full name in DB	Complete path of the parameter in the instance DB / technology DB. The display field is empty for parameters that are not contained in the instance DB / technology DB.	X	X
Name in DB	Name of the parameter in the instance DB / technology DB. If the parameter is part of a structure or UDT, the prefix ". ./" is added. The display field is empty for parameters that are not contained in the instance DB / technology DB.	X	X
Status of configuration	Display of the completeness of the configuration using status symbols. see Status of configuration (offline) (Page 6016)	X	
Compare result	Result of the "Compare values" function. This column is shown if there is an online connection and the "Monitor all" button is selected.		X
Start value project	Configured start value in the project. Error indication if entered values have a syntax or process-related error.	X	X
Default value	Value that is pre-assigned to the parameter. The display field is empty for parameters that are not contained in the instance DB / technology DB.	X	X
Snapshot	Snapshot of the current values in the CPU (monitor values). Error indication if values have a process-related error.	X	X
Start value PLC	Start value in the CPU. This column is shown if there is an online connection and the "Monitor all" button is selected. Error indication if values have a process-related error.		X
Monitor value	Current value in the CPU. This column is shown if there is an online connection and the "Monitor all" button is selected. Error indication if values have a process-related error.		X
Modify value	Value that is to be used to change the monitor valuet. This column is shown if there is an online connection and the "Monitor all" button is selected. Error indication if entered values have a syntax or process-related error.		X
Selection for transmission	Selection of the Modify values that are to be transmitted using the "Modify all selected parameters immediately and once" button. This column is displayed together with the "Modify value" column.		X

Column	Explanation	Offline	Online
Minimum value	Minimum process-related value of the parameter. If the minimum value is dependent on other parameters, it is defined: • Offline: By the Start value project. • Online: By the Monitor values.	X	X
Maximum value	Maximum process-related value of the parameter. If the maximum value is dependent on other parameters, it is defined: • Offline: By the Start value project. • Online: By the Monitor values.	X	X
Setpoint	Designates the parameter as a setpoint. These parameters can be initialized online.	X	X
Data type	Data type of the parameter. The display field is empty for parameters that are not contained in the instance DB / technology DB.	X	X
Retain	Designates the value as a retentive value. The values of retentive parameters are retained even after the voltage supply is switched off.	X	X
Accessible from HMI	Indicates whether the HMI can access this parameter during runtime.	X	X
Visible in HMI	Indicates whether the parameter is visible in the selection list of the HMI by default.	X	X
Comment	Brief description of the parameter.	X	X

See also

Comparing values (Page 6002)

Opening the parameter view

Requirement

The technology object has been added in the project tree, i.e., the associated instance DB / technology DB of the instruction has been created.

Procedure

1. Open the "Technology objects" folder in the project tree.
2. Open the technology object in the project tree.
3. Double-click the "Configuration" object.
4. Select the "Parameter view" tab in the top right corner.

Result

The Parameter view opens. Each displayed parameter is represented by one row in the parameter table.

The displayable parameter properties (table columns) vary depending on whether you are working with the Parameter view in offline or online mode.

In addition, you can selectively display and hide individual table columns.

See also

Default setting of the parameter view (Page 6010)

Default setting of the parameter view

Default settings

To enable you to work efficiently with the Parameter view, you can customize the parameter display and save your settings.

The following customizations are possible and can be saved:

- Show and hide columns
- Change column width
- Change order of the columns
- Toggle navigation
- Select parameter group in the navigation
- Selection of compare values

Show and hide columns

To show or hide columns in the parameter table, follow these steps:

1. Position the cursor in the header of the parameter table.
2. Select the "Show/Hide" command in the shortcut menu.
   The selection of available columns is displayed.
3. To show a column, select the check box for the column.
4. To hide a column, clear the check box for the column.

or

1. Position the cursor in the header of the parameter table.
2. Select the "Show all columns" command in the shortcut menu if all columns of the offline or online mode are to be displayed.

Some columns can only be displayed in online mode: see Parameter table (Page 6008).

Change column width

To customize the width of a column so that all texts in the rows can be read, follow these steps:

1. Position the cursor in the header of the parameter table to the right of the column to be customized until the shape of the cursor changes to a cross.
2. Then double-click this location.

or

1. Open the shortcut menu on the header of the parameter table.
2. Click
   - "Optimize column width" or
   - "Optimize width of all columns".

If the column width setting is too narrow, the complete content of individual fields are shown if you hover the cursor briefly over the relevant field.

Change order of the columns

The columns of the parameter table can be arranged in any way.

To change the order of the columns, follow these steps:

1. Click on the column header and use a drag-and-drop operation to move it to the desired location.
   When you release the mouse button, the column is anchored to the new position.

Toggle navigation

To toggle the display form of the parameters, follow these steps:

1. Select the desired navigation in the “Select navigation structure” drop-down list.
   - Data navigation
   - Functional navigation

See also Navigation (Page 6007).

Select parameter group in the navigation

Within the selected navigation, you choose between the “All parameters” display or the display of a subordinate parameter group of your choice.

1. Click the desired parameter group in the navigation.
   The parameter table only displays the parameters of the parameter group.

Selection of compare values (online)

To set the compare values for the “Compare values” function, follow these steps:

1. Select the desired compare values in the “Selection of compare values” drop-down list.
   - Start value project / Start value PLC
   - Start value project / Snapshot
   - Start value PLC / Snapshot

The “Start value project / Start value PLC” option is set by default.

Saving the default setting of the Parameter view

To save the above customizations of the Parameter view, follow these steps:

1. Customize the Parameter view according to your requirements.
2. Click the “Save window settings” button  at the top right of the Parameter view.

Working with the parameter view

Overview

The following table provides an overview of the functions of the Parameter view in online and offline mode described in the following.

- Column "Offline" = X: This function is possible in offline mode.
- Column "Online" = X: This function is possible in online mode.

Function/action	Offline	Online
Filtering the parameter table (Page 6013)	X	X
Sorting the parameter table (Page 6013)	X	X
Transferring parameter data to other editors (Page 6014)	X	X
Indicating errors (Page 6014)	X	X
Editing start values in the project (Page 6015)	X	X
Status of configuration (offline) (Page 6016)	X	
Monitoring values online in the parameter view (Page 6016)		X
Create snapshot of monitor values (Page 6018)		X
Modifying values (Page 6019)		X
Comparing values (Page 6020)		X
Applying values from the online program as start values (Page 6021)		X
Initializing setpoints in the online program (Page 6022)		X

Filtering the parameter table

You can filter the parameters in the parameter table in the following ways:

- With the text filter
- With the subgroups of the navigation

Both filter methods can be used simultaneously.

With the text filter

Texts that are visible in the parameter table can be filtered. This means only texts in displayed parameter rows and columns can be filtered.

1. Enter the desired character string for filtering in the “Text filter...” input box.
   The parameter table displays only the parameters containing the character string.

The text filtering is reset.

- When another parameter group is selected in the navigation.
- When navigation is changed from data navigation to functional navigation, or vice versa.

With the subgroups of the navigation

1. Click the desired parameter group in the navigation, e.g., "Static".
   The parameter table only shows the static parameters. You can select further subgroups for some groups of the navigation.
2. Click “All parameters” in the navigation if all parameters are to be shown again.

Sorting the parameter table

The values of the parameters are arranged in rows. The parameter table can be sorted by any displayed column.

- In columns containing numerical values, sorting is based on the magnitude of the numerical value.
- In text columns, sorting is alphabetical.

Sorting by column

1. Position the cursor in the header cell of the desired column.
   The background of this cell turns blue.
2. Click the column header.

Result

The entire parameter table is sorted by the selected column. A triangle with tip facing up appears in the column header.

Clicking the column header again changes the sorting as follows:

- Symbol "▲": Parameter table is sorted in ascending order.
- Symbol "▼": Parameter table is sorted in descending order.
- No symbol: The sorting is removed again. The parameter table assumes the default display.

The "../" prefix in the "Name in DB" column is ignored when sorting.

Transferring parameter data to other editors

After selecting an entire parameter row of the parameter table, you can use the following:

- Drag-and-drop
- <Ctrl+C>/<Ctrl+V>
- Copy/Paste via shortcut menu

Transfer parameters to the following editors of the TIA Portal:

- Program editor
- Watch table
- Signal table for trace function

The parameter is inserted with its full name: See information in "Full name in DB" column.

Indicating errors

Error indication

Parameter assignment errors that result in compilation errors (e.g., limit violation) are indicated in the Parameter view.

Every time a value is input in the Parameter view, a check is made for process-related and syntax errors and the result is indicated.

Bad values are indicated by:

- Red error symbol in the "Status of configuration" (offline mode) or "Compare result" (online mode, depending on the selected comparison type) columns

and/or

- Table field with red background
  If you click the bad field, a roll-out error message appears with information of the permissible value range or the required syntax (format)

Compilation error

From the error message of the compiler, you can directly open the Parameter view (functional navigation) containing the parameter causing the error in situations where the parameter is not displayed in the configuration dialog.

Editing start values in the project

With the Parameter view, you can edit the start values in the project in offline mode and online mode.

- You make value changes in the “Start value project” column of the parameter table.
- In the “Status of configuration” column of the parameter table, the progress of the configuration is indicated by the familiar status symbols from the configuration dialog of the technology object.

Boundary conditions

- If other parameters depend on the parameter whose start value was changed, the start value of the dependent parameters are also adapted.
- If a parameter of a technology object is not editable, it is also not editable in the parameter view. The ability to edit a parameter can also depend on the values of other parameters.

Defining new start values

To define start values for parameters in the Parameter view, follow these steps:

1. Open the Parameter view of the technology object.
2. Enter the desired start values in the "Start value project" column. The value must match the data type of the parameter and must not exceed the value range of the parameter. The limits of the value range can be seen in the “Maximum value” and “Minimum value” columns.

The "Status of configuration" column indicates the progress of the configuration with colored symbols.

See also Status of configuration (offline) (Page 6016)

Following adaptation of the start values and downloading of the technology object to the CPU, the parameters take the defined value at startup if they are not declared as retentive (“Retain” column).

Error indication

When a start value is input, a check is made for process-related and syntax errors and the result is indicated.

Bad start values are indicated by:

- Red error symbol in the "Status of configuration" (offline mode) or "Compare result" (online mode, depending on the selected comparison type) columns

and/or

- Red background in the “Start value project” field
  If you click on the bad field, a roll-out error message appears with information of the permissible value range or the necessary syntax (format)

Correcting bad start values

1. Correct bad start values using information from the roll-out error message.
   Red error symbol, red field background, and roll-out error message are no longer displayed.

The project cannot be successfully compiled unless the start values are error-free.

Status of configuration (offline)

The status of the configuration is indicated by icons:

- In the “Status of configuration” column in the parameter table
- In the navigation structure of the functional navigation and data navigation

Symbol in “Status of configuration” column

Symbol	Meaning
	The start value of the parameter corresponds to the default value and is valid. A start value has not yet been defined by the user.
	The start value of the parameter contains a value defined by the user or an automatically adjusted value. The start value is different than the default value. The start value is error-free and valid.
	The start value of the parameter is invalid (syntax or process-related error). The input box has a red background. When clicked, the roll-out error message indicates the cause of the error.
	Only for S7-1200 Motion Control: The start value of the parameter is valid but contains warnings. The input box has a yellow background.
	The parameter is not relevant in the current configuration.

Symbol in the navigation

The symbols in the navigation indicate the progress of the configuration in the same way as in the configuration dialog of the technology object.

See also

Configure technology objects (Page 5998)

Monitoring values online in the parameter view

You can monitor the values currently taken by the parameters of the technology object in the CPU (monitor values) directly in the Parameter view.

Requirements

- There is an online connection.
- The technology object is downloaded to the CPU.

- The program execution is active (CPU in "RUN").
- The Parameter view of the technology object is open.

Procedure

1. Start the monitoring by clicking .
   As soon as the Parameter view is online, the following columns are additionally displayed:
   - Compare result
   - Start value PLC
   - Monitor value
   - Modify value
   - Selection for transmission

   The "Monitor value" column shows the current parameter values on the CPU.
   Meaning of the additional columns: see Parameter table (Page 6008)
2. Stop the monitoring by clicking  again.

Display

All columns that are only available online have an orange background:

- Values in light-orange cells can be changed.
- Values in cells with a dark orange background cannot be changed.

Change display format of value

The display format of the value can be selected via the shortcut menu of a table row in the Parameter view of the technology object.

The display format of the following values can be changed both in online mode and in offline mode:

- Start value project
- Start value PLC
- Maximum value
- Minimum value
- Snapshot
- Monitor value
- Default value
- Modify value

The set display format applies to all values of the table row.

The following display formats of the value can be changed:

- Default
- Hex
- Octal
- Bin
- Dec (+/-)
- DEC

Depending on the parameter selected in the parameter view, only the supported display formats can be selected.

Requirements

- The Parameter view of the technology object is open.

Procedure

To change the display format of the value, proceed as follows:

1. Select one or more table rows in which you want to change the display format.
2. Select the "Display format" command in the shortcut menu.
3. Select the desired display format.

Note

To change the display format of a certain data type in multiple table rows, sort the Parameter view by this data type. Then select the first and last table row with this data type while keeping the <Shift> key pressed and change the display format for the selected table rows.

Create snapshot of monitor values

You can back up the current values of the technology object on the CPU (monitor values) and display them in the Parameter view.

Requirements

- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").
- The Parameter view of the technology object is open.
- The “Monitor all” button  is selected.

Procedure

To show the current parameter values, follow these steps:

1. In the Parameter view, click the “Create snapshot of monitor values" icon .

Result

The current monitor values are transferred once to the "Snapshot" column of the parameter table.

You can analyze the values "frozen" in this way while the monitor values continue to be updated in the "Monitor values" column.

Modifying values

With the Parameter view, you can modify values of the technology object in the CPU.

You can assign values to the parameter once (Modify value) and modify them immediately. The modify request is executed as quickly as possible without reference to any particular point in the user program.

DANGER

Danger when modifying:

Changing the parameter values while the plant is operating may result in severe damage to property and personal injury in the event of malfunctions or program errors.

Make sure that dangerous states cannot occur before you use the "Modify" function.

Requirements

- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").
- The Parameter view of the technology object is open.
- The “Monitor all” button is selected.
- The parameter can be modified (associated field in the "Modify value" column has a light-orange background).

Procedure

To modify parameters immediately, follow these steps:

1. Enter the desired modify values in the “Modify values” column of the parameter table.
2. Check whether the check box for modifying is selected in the "Select for transmission" column.
   The modify values and associated check boxes of dependent parameters are automatically adapted at the same time.
3. Click the “Modify all selected parameters immediately and once” icon .

The selected parameters are modified once and immediately with the specified values and can be monitored in the "Modify values" column. The check boxes for modifying in the "Selection for transmission" column are automatically cleared after the modify request is complete.

Error indication

When a start value is input, a check is made immediately for process-related and syntax errors and the result is indicated.

Bad start values are indicated by:

- Red background in the “Modify value” field

and

- If you click the bad field, a roll-out error message appears with information of the permissible value range or the necessary syntax (format)

Bad modify values

- Modify values with process-related errors can be transmitted.
- Modify values with syntax errors **cannot** be transmitted.

Comparing values

You can use comparison functions to compare the following memory values of a parameter:

- Start value project
- Start value PLC
- Snapshot

Requirements

- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").
- The Parameter view of the technology object is open.
- The “Monitor all” button  is selected.

Procedure

To compare the start values on the various target systems, follow these steps:

1. Click the "Selection of compare values" icon .
   A selection list containing the comparison options opens:
   - Start value project - Start value PLC (default setting)
   - Start value project - Snapshot
   - Start value PLC - Snapshot
2. Select the desired comparison option.
   The selected comparison option is executed as follows:
   - A scales symbol appears in the header cells of the two columns selected for comparison.
   - Symbols are used in the "Compare result" column to indicate the result of the comparison of the selected columns.

Symbol in "Compare result" column

Symbol	Meaning
	The compare values are equal and error-free.
	The compare values are not equal and error-free.
	At least one of the two compare values has a process-related or syntax error.
	The comparison cannot be performed. At least one of the two comparison values is not available (e.g. snapshot).
	Comparison of the value is inappropriate since it is not relevant in one of the configurations.

Symbol in the navigation

The symbols are shown in the same way in the navigation if the comparison result applies to at least one of the parameters below the displayed navigation structure.

Applying values from the online program as start values

In order to apply optimized values from the CPU to the project as start values, you create a snapshot of the monitor values. Values of the snapshot marked as a "Setpoint" are then applied to the project as start values.

Requirements

- The technology object is of type "PID_Compact" or "PID_3Step".
- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").

- The Parameter view of the technology object is open.
- The “Monitor all” button is selected.

Procedure

To apply optimized values from the CPU, follow these steps:

1. Click the "Create snapshot of monitor values and accept setpoints of this snapshot as start values" icon .

Result

The current monitor values are applied to the "Snapshot" column and their setpoints are copied to the "Start value project" column as new start values.

Note

Applying values of individual parameters

You can also apply the values of individual parameters that are not marked as a setpoint from the "Snapshot" column to the "Start values project" column. To do so, copy the values and insert them into the "Start value project" column using the "Copy" and "Paste" commands in the shortcut menu.

Initializing setpoints in the online program

You can initialize all parameters that are marked as a "Setpoint" in the Parameter view with new values in the CPU in one step. In so doing, the start values are downloaded from the project to the CPU. The CPU remains in "RUN" mode.

To avoid data loss on the CPU during a cold restart or warm restart, you must also download the technology object to the CPU.

DANGER

Danger when changing parameter values

Changing the parameter values while the plant is operating may result in severe damage to property and personal injury in the event of malfunctions or program errors.

Make sure that dangerous states cannot occur before you reinitialize the setpoints.

Requirements

- The technology object is of type "PID_Compact" or "PID_3Step".
- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").

- The Parameter view of the technology object is open.
- The “Monitor all” button is selected.
- The parameters marked as a "Setpoint" have a "Start value project" that is free of process-related and syntax errors

Procedure

To initialize all setpoints, follow these steps:

1. Enter the desired values in the "Start value project" column.
   Ensure that the start values are free of process-related and syntax errors.
2. Click the "Initialize setpoints" icon .

Result

The setpoints in the CPU are initialized with the start values from the project.

13.1.2.11 Display instance DB of a technology object.

An instance DB, in which the parameter and static variables are saved, is created for each technology object.

Procedure

To display the instance DB of a technology object, proceed as follows:

1. Open the CPU folder in the project tree.
2. Open the "Technology objects" folder.
3. Highlight a technology object.
4. Select the command "Open DB editor" in the shortcut menu.

13.1.3 Using PID_Compact (S7-1200, S7-1500)

13.1.3.1 Technology object PID_Compact (S7-1200, S7-1500)

The technology object PID_Compact provides a continuous PID controller with integrated optimization. You can alternatively configure a pulse controller. Both manual and automatic mode are possible.

PID-Compact continuously acquires the measured process value within a control loop and compares it with the required setpoint. From the resulting control deviation, the instruction PID_Compact calculates an output value by which the process value is adapted as quickly

and stable as possible to the setpoint. The output value for the PID controller consists of three actions:

- **Proportional** action
  The proportional action of the output value increases in proportion to the control deviation.
- **I** action
  The integral action of the output value increases until the control deviation has been balanced.
- **D** action
  The derivative action increases with the rate of change of control deviation. The process value is corrected to the setpoint as quickly as possible. The derivative action will be reduced again if the rate of change of control deviation drops.

The instruction PID_Compact calculates the proportional, integral and derivative parameters for your controlled system during pretuning. Fine tuning can be used to tune the parameters further. You do not need to manually determine the parameters.

Additional information

- Overview of software controller (Page 5994)
- Add technology objects (Page 5997)
- Configure technology objects (Page 5998)
- Configuring PID_Compact V2 (Page 6024)
- Configuring PID_Compact V1 (Page 6044)

FAQ

For more information, see the following FAQs in the Siemens Industry Online Support:

- Entry ID 79047707 (https://support.industry.siemens.com/cs/ww/en/view/79047707)

13.1.3.2 PID_Compact V2 (S7-1200, S7-1500)

Configuring PID_Compact V2 (S7-1200, S7-1500)

Basic settings V2 (S7-1200, S7-1500)

Introduction V2 (S7-1200, S7-1500)

Configure the following properties of the "PID_Compact" technology object under "Basic settings" in the Inspector window or in the configuration window:

- Physical quantity
- Control logic
- Start-up behavior after reset

- Setpoint (only in the Inspector window)
- Process value (only in the Inspector window)
- Output value (only in the Inspector window)

Setpoint, process value and output value

You can only configure the setpoint, process value and output value in the Inspector window of the programming editor. Select the source for each value:

- Instance DB
  The value saved in the instance DB is used.
  Value must be updated in the instance DB by the user program.
  There should be no value at the instruction.
  Change via HMI possible.
- Instruction
  The value connected to the instruction is used.
  The value is written to the instance DB each time the instruction is called.
  No change via HMI possible.

Control mode V2 (S7-1200, S7-1500)

Physical quantity

Select the physical quantity and unit of measurement for setpoint, process value, and disturbance variable in the "Controller type" group. Setpoint, process value, and disturbance variable is displayed in this unit of measurement.

Control logic

An increase of the output value is generally intended to cause an increase in the process value. This is referred to as a normal control logic.

PID_Compact does not work with negative proportional gain. Select the check box "Invert control logic" to reduce the process value with a higher output value.

Examples

- Opening the drain valve will reduce the level of a container's contents.
- Increasing cooling will reduce the temperature.

Startup characteristics

1. To switch to "Inactive" mode after CPU restart, clear the "Activate Mode after CPU restart" check box.
   To switch to the operating mode saved in the Mode parameter after CPU restart, select the "Activate Mode after CPU restart" check box.
2. In the "Set Mode to" drop-down list, select the mode that is to be enabled after a complete download to the device.
   After a complete download to the device, PID_Compact starts in the selected operating mode. With each additional restart, PID_Compact starts in the mode that was last saved in Mode.

Example

You have selected the "Activate Mode after CPU restart" check box and the entry "Pretuning" in the "Set Mode to" list. After a complete download to the device, PID_Compact starts in the "Pretuning" mode. If pretuning is still active, PID_Compact starts in "Pretuning" mode again after restart of the CPU. If pretuning was successfully completed and automatic mode is active, PID_Compact starts in "Automatic mode" after restart of the CPU.

Setpoint V2 (S7-1200, S7-1500)

Procedure

Proceed as follows to define a fixed setpoint:

1. Select "Instance DB".
2. Enter a setpoint, e.g. 80° C.
3. Delete any entry in the instruction.

Proceed as follows to define a variable setpoint:

1. Select "Instruction".
2. Enter the name of the REAL variable in which the setpoint is saved.
   Program-controlled assignment of various values to the REAL variable is possible, for example for the time controlled change of the setpoint.

Process value V2 (S7-1200, S7-1500)

PID_Compact will scale the value of the analog input to the physical quantity if you use the analog input value directly.

You will need to write a program for processing if you wish first to process the analog input value. The process value is, for example, not directly proportional to the value at the analog input. The processed process value must be in floating point format.

Procedure

Proceed as follows to use the analog input value without processing:

1. Select the entry "Input_PER" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the address of the analog input.

Proceed as follows to use the processed process value in floating point format:

1. Select the entry "Input" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the name of the variable in which the processed process value is saved.

Output value V2 (S7-1200, S7-1500)

PID_Compact offers three output values. Your actuator will determine which output value you use.

- Output_PER
  The actuator is triggered via an analog output and controlled with a continuous signal, e.g. 0...10V, 4...20mA.
- Output
  The output value needs to be processed by the user program, for example because of nonlinear actuator response.
- Output_PWM
  The actuator is controlled via a digital output. Pulse width modulation creates minimum ON and minimum OFF times.

Procedure

Proceed as follows to use the analog output value:

1. Select the entry "Output_PER (analog)" in the drop-down list "Output".
2. Select "Instruction".
3. Enter the address of the analog output.

Proceed as follows to process the output value using the user program:

1. Select the entry "Output" in the drop-down list "Output".
2. Select "Instance DB".
   The calculated output value is saved in the instance data block.
3. For the preparation of the output value, use the output parameter Output.
4. Transfer the processed output value to the actuator via a digital or analog CPU output.

Proceed as follows to use the digital output value:

1. Select the entry "Output_PWM" in the drop-down list "Output".
2. Select "Instruction".
3. Enter the address of the digital output.

Process value settings V2 (S7-1200, S7-1500)

Process value scaling V2 (S7-1200, S7-1500)

If you have configured the use of Input_PER in the basic setting, you must convert the value of the analog input to the physical quantity of the process value. The current configuration is displayed in the Input_PER display.

Input_PER will be scaled using a low and high value pair if the process value is directly proportional to the value of the analog input.

Procedure

To scale the process value, follow these steps:

1. Enter the low pair of values in the "Scaled low process value" and "Low" input fields.
2. Enter the high pair of values in the "Scaled high process value" and "High" input boxes.

Default settings for the value pairs are stored in the hardware configuration. To use the value pairs from the hardware configuration, follow these steps:

1. Select the PID_Compact instruction in the programming editor.
2. Interconnect Input_PER with an analog input in the basic settings.
3. Click the "Automatic setting" button in the process value settings.

The existing values will be overwritten with the values from the hardware configuration.

Process value limits V2 (S7-1200, S7-1500)

You must specify an appropriate absolute high limit and low limit for the process value as limit values for your controlled system. As soon as the process value violates these limits, an error occurs (ErrorBits = 0001h). Tuning is canceled when the process value limits are violated. You can configure how PID_Compact reacts to an error in automatic mode in the output value settings.

Advanced settings V2 (S7-1200, S7-1500)

Process value monitoring V2 (S7-1200, S7-1500)

Configure a warning high and low limit for the process value in the "Process value monitoring" configuration window. If one of the warning limits is exceeded or undershot during operation, a warning will be displayed at the PID_Compact instruction:

- At the InputWarning_H output parameter if the warning high limit has been exceeded
- At the InputWarning_L output parameter if the warning low limit has been undershot

The warning limits must be within the process value high and low limits.

The process value high and low limits will be used if you do not enter values.

Example

Process value high limit = 98 °C; warning high limit = 90 °C

Warning low limit = 10 °C; process value low limit = 0 °C

PID_Compact will respond as follows:

Process value	InputWarning_H	InputWarning_L	Error-Bits	Operating mode
> 98 °C	TRUE	FALSE	0001h	Inactive or Substitute output value with error monitoring
≤ 98 °C and > 90 °C	TRUE	FALSE	0000h	Automatic mode
≤ 90 °C and ≥ 10 °C	FALSE	FALSE	0000h	Automatic mode
< 10 °C and ≥ 0 °C	FALSE	TRUE	0000h	Automatic mode
< 0 °C	FALSE	TRUE	0001h	Inactive or Substitute output value with error monitoring

In the output value settings, you can specify the reaction of PID_Compact when the process value high limit or low limit is violated.

See also

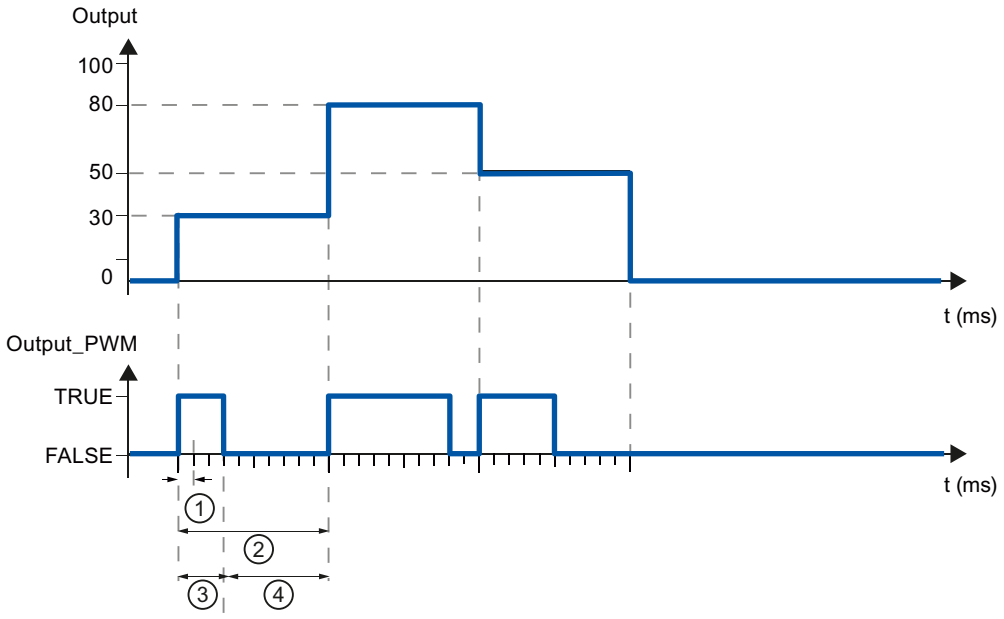
Parameters State and Mode V2 (Page 4219)

PWM limits V2 (S7-1200, S7-1500)

The value at the output parameter Output is transformed into a pulse sequence that is output at output parameter Output_PWM by means of a pulse width modulation. Output is calculated in the PID algorithm sampling time, Output_PWM is output in the PID_Compact sampling time.

The PID algorithm sampling time is determined during pretuning or fine tuning. If manually setting the PID parameters, you will also need to configure the PID algorithm sampling time. The PID_Compact sampling time is equivalent to the cycle time of the calling OB.

The pulse duration is proportional to the value at Output and is always an integer multiple of the PID_Compact sampling time.



① PID_Compact sampling time
② PID algorithm sampling time
③ Pulse duration
④ Break time

The "Minimum ON time" and the "Minimum OFF time" are rounded to an integer multiple of the PID_Compact sampling time.

A pulse or a break is never shorter than the minimum ON or OFF time. The inaccuracies this causes are added up and compensated in the next cycle.

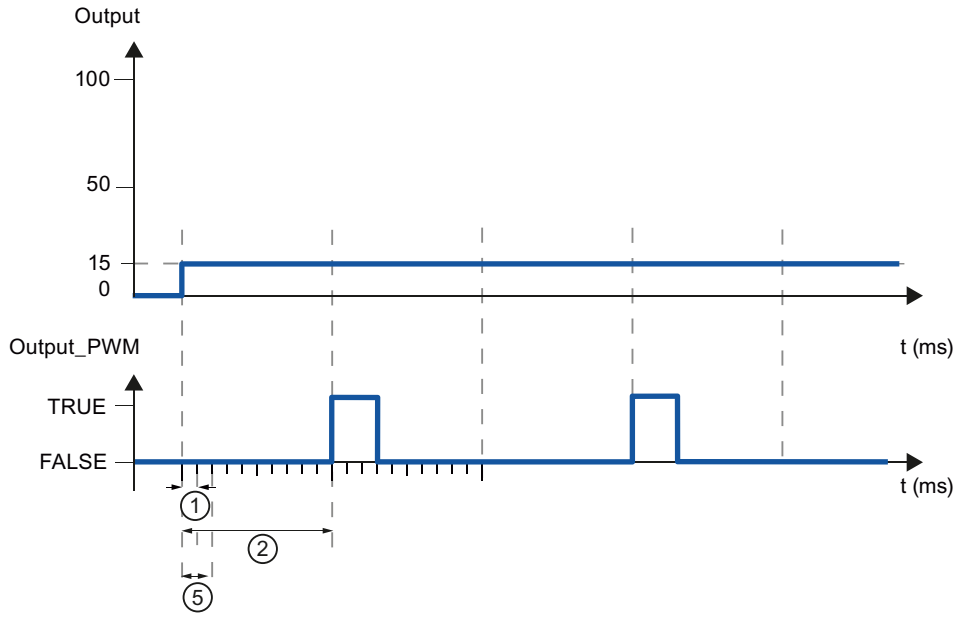
Example

PID_Compact sampling time = 100 ms

PID algorithm sampling time = 1000 ms

Minimum ON time = 200 ms

Output is a constant 15%. The smallest pulse that PID_Compact can output is 20%. In the first cycle, no pulse is output. In the second cycle, the pulse not output in the first cycle is added to the pulse of the second cycle.



① PID_Compact sampling time
② PID algorithm sampling time
⑤ Minimum ON time

In order to minimize operation frequency and conserve the actuator, extend the minimum ON and OFF times.

If you are using "Output" or "Output_PER", you must configure the value 0.0 for the minimum ON and OFF times.

Note

The minimum ON and OFF times only affect the output parameter Output_PWM and are not used for any pulse generators integrated in the CPU.

Output value V2 (S7-1200, S7-1500)

Output value limits

In the "Output value limits" configuration window, configure the absolute limits of your output value in percent. Absolute output value limits are not violated in neither manual mode nor automatic mode. If an output value outside the limits is specified in manual mode, the effective value is limited in the CPU to the configured limits.

The output value limits must match the control logic.

The valid output value limit values depend on the Output used.

Output	-100.0 to 100.0%
Output_PER	-100.0 to 100.0%
Output_PWM	0.0 to 100.0%

Reaction to error

NOTICE
Your system may be damaged. If you output "Current value while error pending " or "Substitute output value while error pending" in the event of an error, PID_Compact remains in automatic mode. This may cause a violation of the process value limits and damage your system. It is essential to configure how your controlled system reacts in the event of an error to protect your system from damage.

PID_Compact is preset so that the controller stays active in most cases in the event of an error. If errors occur frequently in controller mode, this default reaction has a negative effect on the control response. In this case, check the Errorbits parameter and eliminate the cause of the error.

PID_Compact generates a programmable output value in response to an error:

- Zero (inactive)
  PID_Compact outputs 0.0 as output value for all errors and switches to "Inactive" mode. The controller is only reactivated by a falling edge at Reset or a rising edge at ModeActivate.
- Current value while error is pending
  If the following errors occur in **automatic mode**, PID_Compact returns to automatic mode as soon as the errors are no longer pending.
  If one or more of the following errors occur, PID_Compact stays in automatic mode:
  - 0001h: The "Input" parameter is outside the process value limits.
  - 0800h: Sampling time error
  - 40000h: Invalid value at Disturbance parameter.

  If one or more of the following errors occur in **automatic mode**, PID_Compact switches to "Substitute output value with error monitoring" mode and outputs the last valid output value:
  - 0002h: Invalid value at Input_PER parameter.
  - 0200h: Invalid value at Input parameter.
  - 0400h: Calculation of output value failed.
  - 1000h: Invalid value at Setpoint parameter.

  If an error occurs in **manual mode**, PID_Compact continues using the manual value as the output value. If the manual value is invalid, the substitute output value is used. If the manual value and substitute output value are invalid, the output value low limit is used.
  If the following error occurs during a **pretuning or fine tuning**, PID_Compact remains in active mode:
  - 0020h: Pretuning is not permitted during fine tuning.

  When any other error occurs, PID_Compact cancels the tuning and switches to the mode from which tuning was started.
  As soon as no errors are pending, PID_Compact returns to automatic mode.
- Substitute output value while error is pending
  PID_Compact outputs the substitute output value.
  If the following error occurs, PID_Compact stays in "Substitute output value with error monitoring" mode and outputs the output value low limit:
  - 20000h: Invalid value at SubstituteOutput tag.

  For all other errors, PID_Compact reacts as described for "Current value while error is pending".

See also

Parameters State and Mode V2 (Page 4219)

PID parameters V2 (S7-1200, S7-1500)

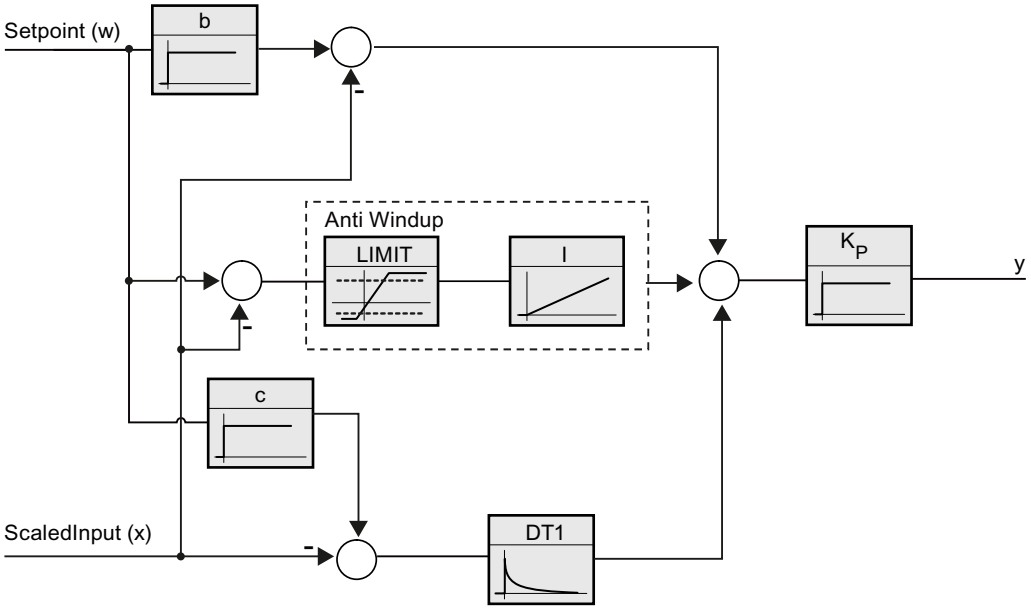
The PID parameters are displayed in the "PID Parameters" configuration window. The PID parameters will be adapted to your controlled system during controller tuning. You do not need to enter the PID parameters manually.

The PID algorithm operates according to the following equation:

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
y	Output value of the PID algorithm
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integral action time
a	Derivative delay coefficient (derivative delay T1 = a × T_D)
T_D	Derivative action time
c	Derivative action weighting

The diagram below illustrates the integration of the parameters into the PID algorithm:



All PID parameters are retentive. If you enter the PID parameters manually, you must completely download PID_Compact.

Downloading technology objects to device (Page 6000)

Proportional gain

The value specifies the proportional gain of the controller. PID_Compact does not work with a negative proportional gain. Control logic is inverted under Basic settings > Controller type.

Integral action time

The integral action time determines the time behavior of the integral action. The integral action is deactivated with integral action time = 0.0.

Derivative action time

The derivative action time determines the time behavior of the derivative action. Derivative action is deactivated with derivative action time = 0.0.

Derivative delay coefficient

The derivative delay coefficient delays the effect of the derivative action.

Derivative delay = derivative action time × derivative delay coefficient

- 0.0: Derivative action is effective for one cycle only and therefore almost not effective.
- 0.5: This value has proved useful in practice for controlled systems with **one** dominant time constant.
- > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed.

Proportional action weighting

The proportional action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Proportional action for setpoint change is fully effective
- 0.0: Proportional action for setpoint change is not effective

The proportional action is always fully effective when the process value is changed.

Derivative action weighting

The derivative action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Derivative action is fully effective upon setpoint change
- 0.0: Derivative action is not effective upon setpoint change

The derivative action is always fully effective when the process value is changed.

PID algorithm sampling time

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the cycle time. All other functions of PID_Compact are executed at every call.

If you use Output_PWM, the accuracy of the output signal is determined by the ratio of the PID algorithm sampling time to the cycle time of the OB. The PID algorithm sampling time corresponds to the time period of the pulse width modulation. The cycle time should be at least 10 times the PID algorithm sampling time.

Rule for tuning

Select whether PI or PID parameters are to be calculated in the "Controller structure" drop-down list.

- **PID**
  Calculates PID parameters during pretuning and fine tuning.
- **PI**
  Calculates PI parameters during pretuning and fine tuning.
- **User-defined**
  The drop-down list displays "User-defined" if you have configured different controller structures for pretuning and fine tuning via a user program.

See also

Selection of the controller structure for specified controlled systems (Page 5993)

Commissioning PID_Compact V2 (S7-1200, S7-1500)

Pretuning V2 (S7-1200, S7-1500)

The pretuning determines the process response to a jump change of the output value and searches for the point of inflection. The PID parameters are calculated from the maximum rate of rise and dead time of the controlled system. You obtain the best PID parameters when you perform pretuning and fine tuning.

The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise. This is most likely the case in operating modes "Inactive" and "manual mode". The PID parameters are backed up before being recalculated.

Requirement

- The "PID_Compact" instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE

- Reset = FALSE
- PID_Compact is in one of the following modes: "Inactive", "Manual mode", or "Automatic mode".
- The setpoint and the process value lie within the configured limits (see "Process value monitoring" configuration).
- The difference between setpoint and process value is greater than 30% of the difference between process value high limit and process value low limit.
- The distance between the setpoint and the process value is > 50% of the setpoint.

Procedure

To perform pretuning, follow these steps:

1. Double-click the "PID_Compact > Commissioning" entry in the project tree.
2. Select the entry "Pretuning" in the "Tuning mode" drop-down list.
3. Click the "Start" icon.
   - An online connection will be established.
   - Value recording is started.
   - Pretuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

   Note

   Click the "Stop" icon when the progress bar has reached 100% and it can be assumed the controller tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

If pretuning was performed without an error message, the PID parameters have been tuned. PID_Compact switches to automatic mode and uses the tuned parameters. The tuned PID parameters will be retained during power OFF and a restart of the CPU.

If pretuning is not possible, PID_Compact responds with the configured reaction to errors.

See also

Parameters State and Mode V2 (Page 4219)

Fine tuning V2 (S7-1200, S7-1500)

Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are tuned for the operating point from the amplitude and frequency of this oscillation. All PID parameters are recalculated from the results. PID parameters from fine tuning usually have better master control and disturbance characteristics than PID parameters from pretuning. You obtain the best PID parameters when you perform pretuning and fine tuning.

PID_Compact automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. The PID parameters are backed up before being recalculated.

Requirement

- The PID_Compact instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- The setpoint and the process value lie within the configured limits.
- The control loop has stabilized at the operating point. The operating point is reached when the process value corresponds to the setpoint.
- No disturbances are expected.
- PID_Compact is in one of the following operating modes: Inactive, automatic mode, or manual mode.

Process depends on initial situation

Fine tuning can be started from the following operating modes: "Inactive", "automatic mode", or "manual mode". Fine tuning proceeds as follows when started from:

- Automatic mode
  Start fine tuning from automatic mode if you wish to improve the existing PID parameters through tuning.
  PID_Compact controls the system using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start.
- Inactive or manual mode
  If the requirements for pretuning are met, pretuning is started. The determined PID parameters will be used for control until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start. If pretuning is not possible, PID_Compact responds with the configured reaction to errors.
  An attempt is made to reach the setpoint with the minimum or maximum output value if the process value for pretuning is already too near the setpoint. This can produce increased overshoot.

Procedure

To perform fine tuning, follow these steps:

1. Select the entry "Fine tuning" in the "Tuning mode" drop-down list.
2. Click the "Start" icon.
   - An online connection will be established.
   - Value recording is started.
   - The process of fine tuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

Note

Click the "Stop" icon in the "Tuning mode" group when the progress bar has reached 100% and it is to be assumed that tuning is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

If no errors occurred during fine tuning, the PID parameters have been tuned. PID_Compact switches to automatic mode and uses the tuned parameters. The tuned PID parameters will be retained during power OFF and a restart of the CPU.

If errors occurred during "fine tuning", PID_Compact responds with the configured response to errors.

See also

Parameters State and Mode V2 (Page 4219)

"Manual" mode V1 (S7-1200, S7-1500)

The following section describes how you can use the "manual mode" operating mode in the commissioning window of the "PID_Compact" technology object. Manual mode is also possible when an error is pending.

Requirement

- The "PID_Compact" instruction is called in a cyclic interrupt OB.
- An online connection to the CPU has been established and the CPU is in the "RUN" mode.

Procedure

Use "Manual mode" in the commissioning window if you want to test the controlled system by specifying a manual value. To define a manual value, follow these steps:

1. Click the "Start" icon.
2. Select the "Manual mode" check box in the "Online status of controller" area. PID_Compact operates in manual mode. The most recent current output value remains in effect.
3. Enter the manual value in the "Output" field as a % value.
4. Click the  icon.

Result

The manual value is written to the CPU and immediately goes into effect.

Clear the "Manual mode" check box if the output value is to be specified again by the PID controller. The switchover to automatic mode is bumpless.

See also

Parameters State and Mode V2 (Page 4219)

Override control with PID_Compact V2 (S7-1200, S7-1500)

Override control

In case of override control, two or more controllers share one actuator. Only one controller has access to the actuator at any time and influences the process.

A logic operation decides which controller has access to the actuator. This decision is often made based on a comparison of the output values of all controllers, for example, in case of a maximum selection, the controller with the largest output value gets access to the actuator.

The selection based on the output value requires that all controllers operate in automatic mode. The controllers that do not have an effect on the actuator are updated. This is necessary to prevent windup effects and their negative impacts on the control response and the switchover between the controllers.

PID_Compact supports override controls as of version 2.3 by offering a simple process for updating the controllers that are not active: By using the tags OverwriteInitialOutputValue and PIDCtrl.PIDInit, you can pre-assign the integral action of the controller in automatic mode as though the PID algorithm had calculated Output = OverwriteInitialOutputValue for the output value in the last cycle. To do this, OverwriteInitialOutputValue is interconnected with the output value of the controller that currently has access to the actuator. By setting the bit PIDCtrl.PIDInit, you trigger the pre-assignment of the integral action as well as the restart of the controller cycle and the PWM period. The subsequent calculation of the output value in the current cycle takes place based on the pre-assigned (and synchronized for all controllers) integral action as well as the proportional action and integral action from the current control deviation. The derivative action is not active during the call with PIDCtrl.PIDInit = TRUE and therefore does not contribute to the output value.

This procedure ensures that the calculation of the current output value and thus the decision on which controller is to have access to the actuator is only based on the current process state and the PI parameters. Windup effects for controllers that are not active and thus incorrect decisions of the switchover logic are prevented.

Requirements

- PIDCtrl.PIDInit is only effective if the integral action is activated (Retain.CtrlParams.Ti tag > 0.0).
- You must assign PIDCtrl.PIDInit and OverwriteInitialOutputValue in your user program yourself (see example below). PID_Compact does not automatically change these tags.
- PIDCtrl.PIDInit is only effective when PID_Compact is in automatic mode (parameter State = 3)
- If possible, select the sampling time of the PID algorithm (Retain.CtrlParams.Cycle tag) in such a way that it is identical for all controllers, and call all controllers in the same cyclic interrupt OB. In this way, you ensure that the switchover does not take place within a controller cycle or a PWM period.

Note

Constant adaptation of the output value limits

Instead of the active updating of the controllers without access to the actuator described here, this is implemented alternatively by constant adaptation of the output value limits in other controller systems.

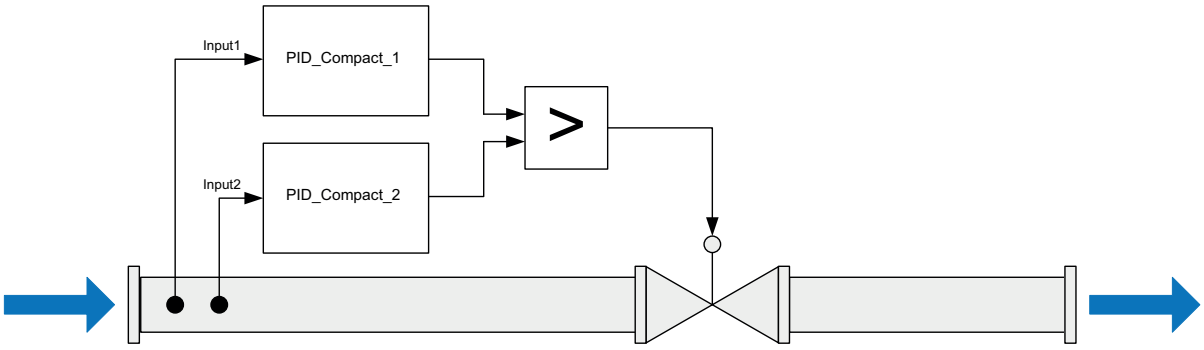
This is not possible with PID_Compact, because a change of the output value limits is not supported in automatic mode.

Example: Control of a gas pipeline

PID_Compact is used for control of a gas pipeline.

The main goal is to control the flow rate Input1. The controller PID_Compact_1 is used for this purpose. In addition, the pressure Input2 (measured in flow direction in front of the valve) is to be kept below the high limit with the limiting controller PID_Compact_2.

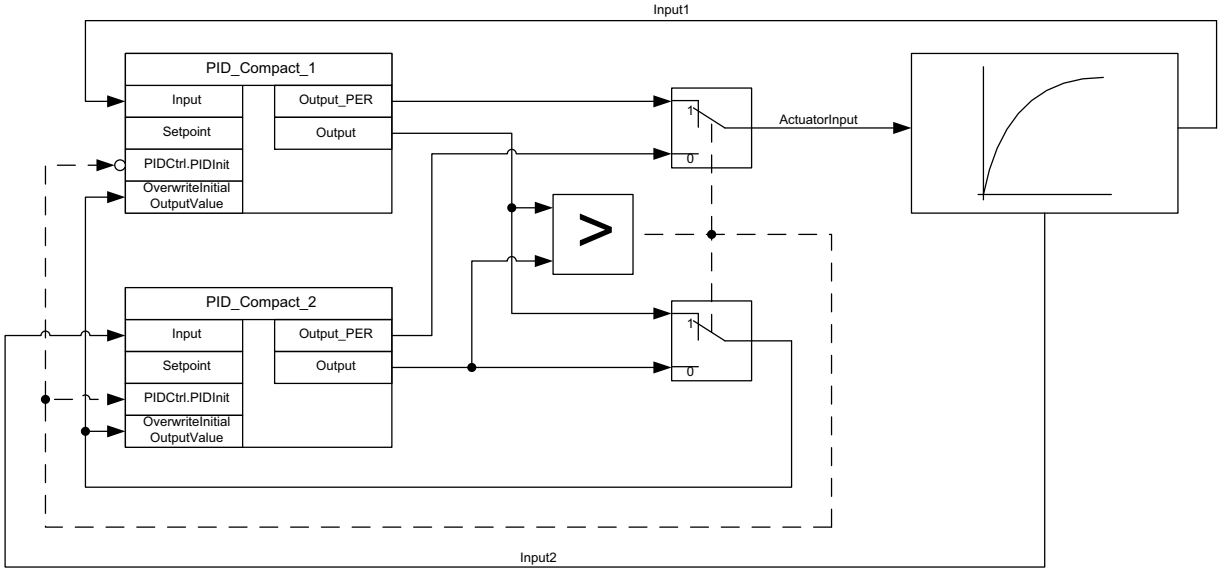
Flow rate and pressure are controlled by a single solenoid valve. The output value of the controller corresponds to the valve opening: The valve is opened when the output value increases. This means the flow rate increases (normal control logic) while the pressure drops (inverted control logic).



The valve is controlled with the output value of PID_Compact in I/O format (parameter Output_PER) by writing the program tag ActuatorInput.

The setpoint for the flow rate is specified at the parameter PID_Compact_1.Setpoint.

The pressure high limit is specified as setpoint at the parameter PID_Compact_2.Setpoint.



Both controllers must share one valve as shared actuator. The logic that decides which controller gets access to the actuator is implemented by a maximum selection of the output value (in Real format, parameter Output) in this case. Because the output value corresponds to the opening of the valve, the controller that requires the larger valve opening gets the control.

Note

Activate inversion of the control logic

Because a decrease of the actual value (pressure) is to be achieved with the pressure regulator PID_Compact_2 when the output value increases (valve opening), the inversion of the control logic must be activated: PID_Compact_2.Config.InvertControl = TRUE.

In normal operation of the plant, the actual value of the flow rate corresponds to the setpoint. The flow controller PID_Compact_1 has settled on a stationary output value PID_Compact_1.Output. The actual value of the pressure in normal operation is significantly

below the high limit that is specified as setpoint for PID_Compact_2. The pressure regulator therefore wants to close the valve even further to increase the pressure, which means it will calculate an output value PID_Compact_2.Output that is smaller than the output value of the flow controller PID_Compact_1.Output. The maximum selection of the switchover logic therefore gives the flow controller PID_Compact_1 continued access to the actuator. In addition, it is ensured that PID_Compact_2 is updated by means of the assignments PID_Compact_2.OverwriteInitialOutputValue = PID_Compact_1.Output and PID_Compact_2.PIDCtrl.PIDInit = TRUE.

If the pressure now approaches the high limit or exceeds it, for example due to a fault, the pressure regulator PID_Compact_2 calculates a higher output value to open the valve even further and thus reduce the pressure. If PID_Compact_2.Output is greater than PID_Compact_1.Output, the pressure regulator PID_Compact_2 receives access to the actuator through the maximum selection and opens it. It is ensured that PID_Compact_1 is updated by means of the assignments PID_Compact_1.OverwriteInitialOutputValue = PID_Compact_2.Output and PID_Compact_1.PIDCtrl.PIDInit = TRUE.

The pressure is reduced while the flow rate increases and can no longer be kept at the setpoint.

Once the fault has been remedied, the pressure will continue to drop and the opening of the valve is reduced by the pressure regulator. If the flow controller calculates a larger opening as output value, the plant returns to normal operation so that the flow controller PID_Compact_1 once again has access to the actuator.

This example can be implemented with the following SCL program code:

```
"PID_Compact_1"(Input := "Input1");
"PID_Compact_2"(Input := "Input2");
IF "PID_Compact_1".Output >= "PID_Compact_2".Output THEN
  "ActuatorInput" := "PID_Compact_1".Output_PER;
  "PID_Compact_1".PIDCtrl.PIDInit := FALSE;
  "PID_Compact_2".PIDCtrl.PIDInit := TRUE;
  "PID_Compact_2".OverwriteInitialOutputValue :=
  "PID_Compact_1".Output;
ELSE
  "ActuatorInput" := "PID_Compact_2".Output_PER;
  "PID_Compact_1".PIDCtrl.PIDInit := TRUE;
  "PID_Compact_2".PIDCtrl.PIDInit := FALSE;
  "PID_Compact_1".OverwriteInitialOutputValue :=
  "PID_Compact_2".Output;
END_IF;
```

Simulating PID_Compact V2 with PLCSIM (S7-1200, S7-1500)

Note

Simulation with PLCSIM

The simulation of PID_Compact V2.x with PLCSIM for CPU S7-1200 is not supported.

PID_Compact V2.x can only be simulated for CPU S7-1500 with PLCSIM.

For the simulation with PLCSIM, the time behavior of the simulated PLC is not exactly identical to that of a "real" PLC. The actual cycle clock of a cyclic interrupt OB can have larger fluctuations with a simulated PLC than with "real" PLCs.

In the standard configuration, PID_Compact determines the time between calls automatically and monitors them for fluctuations.

For the simulation of PID_Compact with PLCSIM, for example, a sampling time error (ErrorBits = DW#16#00000800) can therefore be detected.

This results in ongoing tuning being aborted.

The response in automatic mode depends on the value of the ActivateRecoverMode tag.

To prevent this from happening, you should configure PID_Compact for simulation with PLCSIM as follows:

- CycleTime.EnEstimation = FALSE
- CycleTime.EnMonitoring = FALSE
- CycleTime.Value: Assign the cycle clock of the calling cyclic interrupt OB in seconds to this tag.

13.1.3.3 PID_Compact V1 (S7-1200, S7-1500)

Configuring PID_Compact V1 (S7-1200, S7-1500)

Basic settings V1 (S7-1200, S7-1500)

Introduction V1 (S7-1200, S7-1500)

Configure the following properties of the "PID_Compact" technology object under "Basic settings" in the Inspector window or in the configuration window:

- Physical quantity
- Control logic
- Start-up behavior after reset
- Setpoint (only in the Inspector window)
- Process value (only in the Inspector window)
- Output value (only in the Inspector window)

Setpoint, process value and output value

You can only configure the setpoint, process value and output value in the Inspector window of the programming editor. Select the source for each value:

- Instance DB
  The value saved in the instance DB is used.
  Value must be updated in the instance DB by the user program.
  There should be no value at the instruction.
  Change via HMI possible.
- Instruction
  The value connected to the instruction is used.
  The value is written to the instance DB each time the instruction is called.
  No change via HMI possible.

Control mode V1 (S7-1200, S7-1500)

Physical quantity

Select the unit of measurement and physical quantity for the setpoint and process value in the "Controller type" group. The setpoint and process value will be displayed in this unit.

Control logic

An increase of the output value is generally intended to cause an increase in the process value. This is referred to as a normal control logic.

PID_Compact does not work with negative proportional gain. Select the check box "Invert control logic" to reduce the process value with a higher output value.

Examples

- Opening the drain valve will reduce the level of a container's contents.
- Increasing cooling will reduce the temperature.

Start-up behavior after reset

To change straight to the last active mode after restarting the CPU, select the "Enable last mode after CPU restart" check box.

PID_Compact will remain in "Inactive" mode if the check box is cleared.

Setpoint V1 (S7-1200, S7-1500)

Procedure

Proceed as follows to define a fixed setpoint:

1. Select "Instance DB".
2. Enter a setpoint, e.g. 80° C.
3. Delete any entry in the instruction.

Proceed as follows to define a variable setpoint:

1. Select "Instruction".
2. Enter the name of the REAL variable in which the setpoint is saved.
   Program-controlled assignment of various values to the REAL variable is possible, for example for the time controlled change of the setpoint.

Process value V1 (S7-1200, S7-1500)

PID_Compact will scale the value of the analog input to the physical quantity if you use the analog input value directly.

You will need to write a program for processing if you wish first to process the analog input value. The process value is, for example, not directly proportional to the value at the analog input. The processed process value must be in floating point format.

Procedure

Proceed as follows to use the analog input value without processing:

1. Select the entry "Input_PER" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the address of the analog input.

Proceed as follows to use the processed process value in floating point format:

1. Select the entry "Input" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the name of the variable in which the processed process value is saved.

Output value V1 (S7-1200, S7-1500)

PID_Compact offers three output values. Your actuator will determine which output value you use.

- Output_PER
  The actuator is triggered via an analog output and controlled with a continuous signal, e.g. 0...10V, 4...20mA.
- Output
  The output value needs to be processed by the user program, for example because of nonlinear actuator response.
- Output_PWM
  The actuator is controlled via a digital output. Pulse width modulation creates minimum ON and minimum OFF times.

Procedure

Proceed as follows to use the analog output value:

1. Select the entry "Output_PER (analog)" in the drop-down list "Output".
2. Select "Instruction".
3. Enter the address of the analog output.

Proceed as follows to process the output value using the user program:

1. Select the entry "Output" in the drop-down list "Output".
2. Select "Instance DB".
   The calculated output value is saved in the instance data block.
3. For the preparation of the output value, use the output parameter Output.
4. Transfer the processed output value to the actuator via a digital or analog CPU output.

Proceed as follows to use the digital output value:

1. Select the entry "Output_PWM" in the drop-down list "Output".
2. Select "Instruction".
3. Enter the address of the digital output.

Process value settings V1 (S7-1200, S7-1500)

Configure the scaling of your process value and specify the process value absolute limits In the "Process value settings" configuration window.

Scaling the process value

If you have configured the use of Input_PER in the basic settings, you will need to convert the value of the analog input into the physical quantity of the process value. The current configuration will be displayed in the Input_PER display.

Input_PER will be scaled using a low and high value pair if the process value is directly proportional to the value of the analog input.

1. Enter the low pair of values in the "Scaled low process value" and "Low" input fields.
2. Enter the high pair of values in the "Scaled high process value" and "High" input boxes.

Default settings for the value pairs are saved in the hardware configuration. Proceed as follows to use the value pairs from the hardware configuration:

1. Select the instruction PID_Compact in the programming editor.
2. Connect Input_PER with an analog input in the basic settings.
3. Click on the "Automatic setting" button in the process value settings.

The existing values will be overwritten with the values from the hardware configuration.

Monitoring process value

Specify the absolute high and low limit of the process value. As soon as these limits are violated during operation, the controller switches off and the output value is set to 0%. You must enter reasonable limits for your controlled system. Reasonable limits are important during optimization to obtain optimal PID parameters.

The default for the "High limit process value" is 120 %. At the I/O input, the process value can be a maximum of 18% higher than the standard range (overrange). An error is no longer reported for a violation of the "High limit process value". Only a wire-break and a short-circuit are recognized and the PID_Compact switches to "Inactive" mode.

WARNING

If you set very high process value limits (for example $-3.4*10^{38}...+3.4*10^{38}$), process value monitoring will be disabled. Your system may then be damaged if an error occurs.

See also

Process value monitoring V1 (Page 6049)

PWM limits V1 (Page 6049)

Output value limits V1 (Page 6051)

PID parameters V1 (Page 6052)

Advanced settings V1 (S7-1200, S7-1500)

Process value monitoring V1 (S7-1200, S7-1500)

Configure a warning high and low limit for the process value in the "Process value monitoring" configuration window. If one of the warning limits is exceeded or undershot during operation, a warning will be displayed at the PID_Compact instruction:

- At the InputWarning_H output parameter if the warning high limit has been exceeded
- At the InputWarning_L output parameter if the warning low limit has been undershot

The warning limits must be within the process value high and low limits.

The process value high and low limits will be used if you do not enter values.

Example

Process value high limit = 98° C; warning high limit = 90° C

Warning low limit = 10° C; process value low limit = 0° C

PID_Compact will respond as follows:

Process value	InputWarning_H	InputWarning_L	Operating mode
> 98° C	TRUE	FALSE	Inactive
≤ 98° C and > 90° C	TRUE	FALSE	Automatic mode
≤ 90° C and ≥ 10° C	FALSE	FALSE	Automatic mode
< 10° C and ≥ 0° C	FALSE	TRUE	Automatic mode
< 0° C	FALSE	TRUE	Inactive

See also

Process value settings V1 (Page 6047)

PWM limits V1 (Page 6049)

Output value limits V1 (Page 6051)

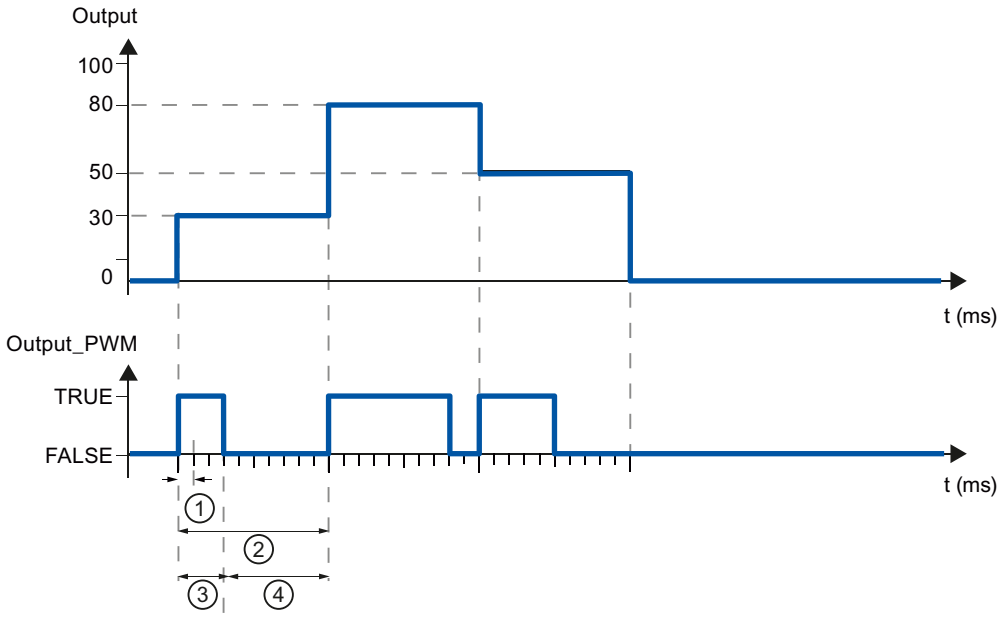
PID parameters V1 (Page 6052)

PWM limits V1 (S7-1200, S7-1500)

The value at the output parameter Output is transformed into a pulse sequence that is output at output parameter Output_PWM by means of a pulse width modulation. Output is calculated in the PID algorithm sampling time, Output_PWM is output in the PID_Compact sampling time.

The PID algorithm sampling time is determined during pretuning or fine tuning. If manually setting the PID parameters, you will also need to configure the PID algorithm sampling time. The PID_Compact sampling time is equivalent to the cycle time of the calling OB.

The pulse duration is proportional to the value at Output and is always an integer multiple of the PID_Compact sampling time.



① PID_Compact sampling time
② PID algorithm sampling time
③ Pulse duration
④ Break time

The "Minimum ON time" and the "Minimum OFF time" are rounded to an integer multiple of the PID_Compact sampling time.

A pulse or a break is never shorter than the minimum ON or OFF time. The inaccuracies this causes are added up and compensated in the next cycle.

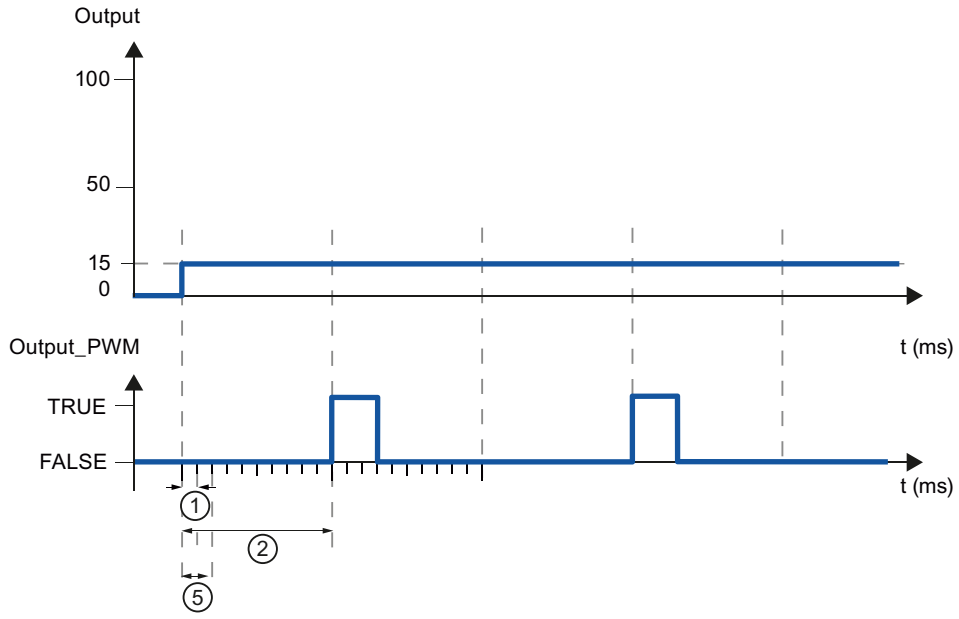
Example

PID_Compact sampling time = 100 ms

PID algorithm sampling time = 1000 ms

Minimum ON time = 200 ms

Output is a constant 15%. The smallest pulse that PID_Compact can output is 20%. In the first cycle, no pulse is output. In the second cycle, the pulse not output in the first cycle is added to the pulse of the second cycle.



① PID_Compact sampling time
② PID algorithm sampling time
⑤ Minimum ON time

In order to minimize operation frequency and conserve the actuator, extend the minimum ON and OFF times.

If you are using "Output" or "Output_PER", you must configure the value 0.0 for the minimum ON and OFF times.

Note

The minimum ON and OFF times only affect the output parameter Output_PWM and are not used for any pulse generators integrated in the CPU.

See also

Process value settings V1 (Page 6047)

Process value monitoring V1 (Page 6049)

Output value limits V1 (Page 6051)

PID parameters V1 (Page 6052)

Output value limits V1 (S7-1200, S7-1500)

In the "Output value limits" configuration window, configure the absolute limits of your output value in percent. Absolute output value limits are not violated in neither manual mode nor in automatic mode. If a output value outside the limits is specified in manual mode, the effective value is limited in the CPU to the configured limits.

The valid output value limit values depend on the Output used.

Output	-100.0 to 100.0
Output_PER	-100.0 to 100.0
Output_PWM	0.0 to 100.0

PID_Compact sets the output value to 0.0 if an error occurs. 0.0 must therefore always be within the output value limits. You will need to add an offset to Output and Output_PER in the user program if you want an output value low limit of greater than 0.0.

See also

Process value settings V1 (Page 6047)

Process value monitoring V1 (Page 6049)

PWM limits V1 (Page 6049)

PID parameters V1 (Page 6052)

PID parameters V1 (S7-1200, S7-1500)

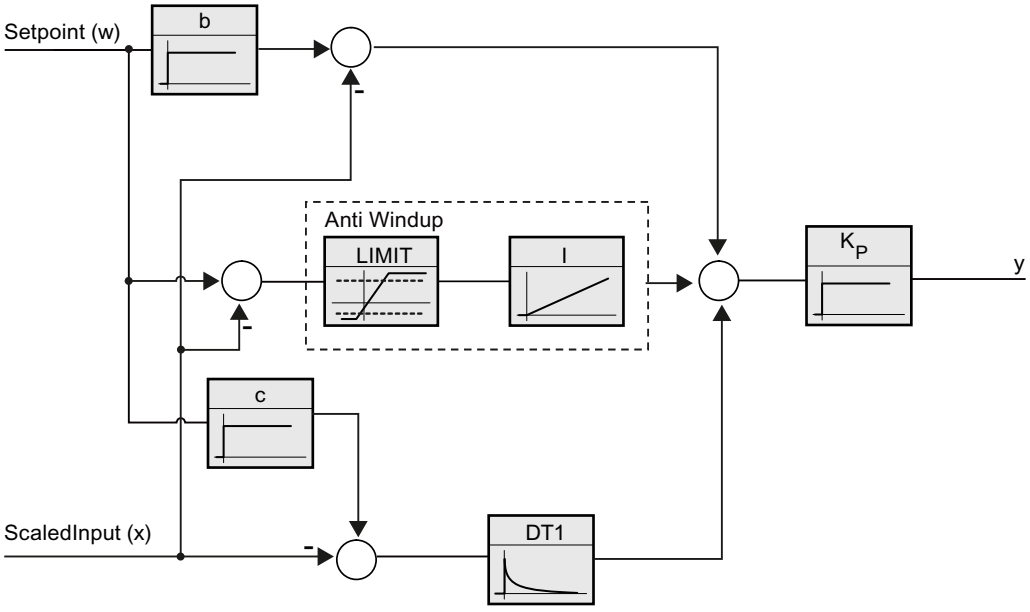
The PID parameters are displayed in the "PID Parameters" configuration window. The PID parameters will be adapted to your controlled system during controller tuning. You do not need to enter the PID parameters manually.

The PID algorithm operates according to the following equation:

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
y	Output value of the PID algorithm
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integral action time
a	Derivative delay coefficient (derivative delay T1 = a × T_D)
T_D	Derivative action time
c	Derivative action weighting

The diagram below illustrates the integration of the parameters into the PID algorithm:



All PID parameters are retentive. If you enter the PID parameters manually, you must completely download PID_Compact.

AUTOHOTSPOT

Proportional gain

The value specifies the proportional gain of the controller. PID_Compact does not work with a negative proportional gain. Control logic is inverted under Basic settings > Controller type.

Integral action time

The integral action time determines the time behavior of the integral action. The integral action is deactivated with integral action time = 0.0.

Derivative action time

The derivative action time determines the time behavior of the derivative action. Derivative action is deactivated with derivative action time = 0.0.

Derivative delay coefficient

The derivative delay coefficient delays the effect of the derivative action.

Derivative delay = derivative action time × derivative delay coefficient

- 0.0: Derivative action is effective for one cycle only and therefore almost not effective.
- 0.5: This value has proved useful in practice for controlled systems with **one** dominant time constant.
- > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed.

Proportional action weighting

The proportional action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Proportional action for setpoint change is fully effective
- 0.0: Proportional action for setpoint change is not effective

The proportional action is always fully effective when the process value is changed.

Derivative action weighting

The derivative action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Derivative action is fully effective upon setpoint change
- 0.0: Derivative action is not effective upon setpoint change

The derivative action is always fully effective when the process value is changed.

PID algorithm sampling time

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the cycle time. All other functions of PID_Compact are executed at every call.

If you use Output_PWM, the accuracy of the output signal is determined by the ratio of the PID algorithm sampling time to the cycle time of the OB. The PID algorithm sampling time corresponds to the time period of the pulse width modulation. The cycle time should be at least 10 times the PID algorithm sampling time.

Rule for tuning

Select whether PI or PID parameters are to be calculated in the "Controller structure" drop-down list.

- **PID**
  Calculates PID parameters during pretuning and fine tuning.
- **PI**
  Calculates PI parameters during pretuning and fine tuning.
- **User-defined**
  The drop-down list displays "User-defined" if you have configured different controller structures for pretuning and fine tuning via a user program.

See also

Selection of the controller structure for specified controlled systems (Page 5993)

Downloading technology objects to device (Page 6000)

Commissioning PID_Compact V1 (S7-1200, S7-1500)

Commissioning V1 (S7-1200, S7-1500)

The commissioning window helps you commission the PID controller. You can monitor the values for the setpoint, process value and output value along the time axis in the trend view. The following functions are supported in the commissioning window:

- Controller pretuning
- Controller fine tuning
  Use fine tuning for fine adjustments to the PID parameters.
- Monitoring the current closed-loop control in the trend view
- Testing the controlled system by specifying a manual output value

All functions require an online connection to the CPU to have been established.

Basic handling

- Select the desired sampling time in the "Sampling time" drop-down list.
  All values in the commissioning window are updated in the selected update time.
- Click the "Start" icon in the measuring group if you want to use the commissioning functions.
  Value recording is started. The current values for the setpoint, process value and output value are entered in the trend view. Operation of the commissioning window is enabled.
- Click the "Stop" icon if you want to end the commissioning functions.
  The values recorded in the trend view can continue to be analyzed.

Closing the commissioning window will terminate recording in the trend view and delete the recorded values.

See also

Pretuning V1 (Page 6055)

Fine tuning V1 (Page 6057)

"Manual" mode V1 (Page 6058)

Pretuning V1 (S7-1200, S7-1500)

The pretuning determines the process response to a jump change of the output value and searches for the point of inflection. The tuned PID parameters are calculated as a function of the maximum slope and dead time of the controlled system.

The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise. The PID parameters are backed up before being recalculated.

Requirement

- The "PID_Compact" instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- PID_Compact is in "inactive" or "manual" mode.
- The setpoint may not be changed during controller tuning. PID_Compact will otherwise be deactivated.
- The setpoint and the process value lie within the configured limits (see "Process value monitoring" configuration).
- The difference between setpoint and process value is greater than 30% of the difference between process value high limit and process value low limit.
- The distance between the setpoint and the process value is > 50% of the setpoint.

Procedure

To perform pretuning, follow these steps:

1. Double-click the "PID_Compact > Commissioning" entry in the project tree.
2. Select the entry "Pretuning" in the "Tuning mode" drop-down list.
3. Click the "Start" icon.
   - An online connection will be established.
   - Value recording is started.
   - Pretuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

   Note

   Click the "Stop" icon when the progress bar has reached 100% and it is to be assumed the controller tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

If pretuning was performed without an error message, the PID parameters have been tuned. PID_Compact switches to automatic mode and uses the tuned parameters. The tuned PID parameters will be retained during power OFF and a restart of the CPU.

If pretuning is not possible, PID_Compact will change to "Inactive" mode.

See also

Parameters State and sRet.i_Mode V1 (Page 4244)

Commissioning V1 (Page 6055)

Fine tuning V1 (Page 6057)

"Manual" mode V1 (Page 6058)

Fine tuning V1 (S7-1200, S7-1500)

Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are optimized for the operating point from the amplitude and frequency of this oscillation. All PID parameters are recalculated on the basis of the findings. PID parameters from fine tuning usually have better master control and disturbance behavior than PID parameters from pretuning.

PID_Compact automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. The PID parameters are backed up before being recalculated.

Requirement

- The PID_Compact instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- The setpoint and the process value lie within the configured limits (see "Process value monitoring" configuration).
- The control loop has stabilized at the operating point. The operating point is reached when the process value corresponds to the setpoint.
- No disturbances are expected.
- The setpoint may not be changed during controller tuning.
- PID_Compact is in inactive mode, automatic mode or manual mode.

Process depends on initial situation

Fine tuning can be started in "inactive", "automatic" or "manual" mode. Fine tuning proceeds as follows when started in:

- Automatic mode
  Start fine tuning in automatic mode if you wish to improve the existing PID parameters using controller tuning.
  PID_Comact will regulate using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start.
- Inactive or manual mode
  If the requirements for pretuning are met, pretuning is started. The PID parameters established will be used for adjustment until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start. If pretuning is not possible, PID_Compact will change to "Inactive" mode.
  An attempt is made to reach the setpoint with a minimum or maximum output value if the process value for pretuning is already too near the setpoint. This can produce increased overshoot.

Procedure

Proceed as follows to carry out "fine tuning":

1. Select the entry "Fine tuning" in the "Tuning mode" drop-down list.
2. Click the "Start" icon.
   - An online connection will be established.
   - Value recording is started.
   - The process of fine tuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

Note

Click the "Stop" icon in the "Tuning mode" group when the progress bar has reached 100% and it is to be assumed the controller tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

The PID parameters will have been optimized if fine tuning has been executed without errors. PID_Compact changes to automatic mode and uses the optimized parameters. The optimized PID parameters will be retained during power OFF and a restart of the CPU.

If errors occurred during "fine tuning", PID_Compact will change to "inactive" mode.

See also

Parameters State and sRet.i_Mode V1 (Page 4244)

Commissioning V1 (Page 6055)

Pretuning V1 (Page 6055)

"Manual" mode V1 (Page 6058)

"Manual" mode V1 (S7-1200, S7-1500)

The following section describes how you can use the "Manual" operating mode in the commissioning window of the "PID Compact" technology object.

Requirement

- The "PID_Compact" instruction is called in a cyclic interrupt OB.
- An online connection to the CPU has been established and the CPU is in the "RUN" mode.
- The functions of the commissioning window have been enabled via the "Start" icon.

Procedure

Use "Manual mode" in the commissioning window if you want to test the process by specifying a manual value. To define a manual value, proceed as follows:

1. Select the check box "Manual mode" in the "Online status of the controller" area. PID_Compact operates in manual mode. The most recent current output value remains in effect.
2. Enter the manual value in the "Output" field as a % value.
3. Click the control icon .

Result

The manual value is written to the CPU and immediately goes into effect.

Note

PID_Compact continues to monitor the process value. If the process value limits are exceeded, PID_Compact is deactivated.

Clear the "Manual mode" check box if the output value is to be specified again by the PID controller. The change to automatic mode is bumpless.

See also

Parameters State and sRet.i_Mode V1 (Page 4244)

Commissioning V1 (Page 6055)

Pretuning V1 (Page 6055)

Fine tuning V1 (Page 6057)

Simulating PID_Compact V1 with PLCSIM (S7-1200, S7-1500)

Note

Simulation with PLCSIM

For the simulation with PLCSIM, the time behavior of the simulated PLC is not exactly identical to that of a "real" PLC. The actual cycle clock of a cyclic interrupt OB can have larger fluctuations with a simulated PLC than with "real" PLCs.

In the standard configuration, PID_Compact determines the time between calls automatically and monitors them for fluctuations.

For a simulation of PID_Compact with PLCSIM, for example, a sampling time error (ErrorBits = DW#16#00000800) can therefore be detected.

PID_Compact switches to "Inactive" mode (State = 0) in this case.

To prevent this from happening, you should configure PID_Compact for simulation with PLCSIM as follows:

- sb_EnCyclEstimation = FALSE
- sb_EnCyclMonitoring = FALSE
- sPid_Calc.r_Cycle: Assign the cycle clock of the calling cyclic interrupt OB in seconds to this tag.

13.1.4 Using PID_3Step (S7-1200, S7-1500)

13.1.4.1 Technology object PID_3Step (S7-1200, S7-1500)

The technology object PID_3Step provides a PID controller with tuning for valves or actuators with integral response.

You can configure the following controllers:

- Three-point step controller with position feedback
- Three-point step controller without position feedback
- Valve controller with analog output value

PID_3Step continuously acquires the measured process value within a control loop and compares it with the setpoint. From the resulting control deviation, PID_3Step calculates an

output value through which the process value reaches the setpoint as quickly and steadily as possible. The output value for the PID controller consists of three actions:

- **Proportional** action
  The proportional action of the output value increases in proportion to the control deviation.
- **I** action
  The integral action of the output value increases until the control deviation has been balanced.
- **D** action
  The derivative action increases with the rate of change of control deviation. The process value is corrected to the setpoint as quickly as possible. The derivative action will be reduced again if the rate of change of control deviation drops.

The instruction PID_3Step calculates the proportional, integral and derivative parameters for your controlled system during pretuning. Fine tuning can be used to tune the parameters further. You do not need to manually determine the parameters.

Additional information

- Overview of software controller (Page 5994)
- Add technology objects (Page 5997)
- Configure technology objects (Page 5998)
- Configuring PID_3Step V2 (Page 6061)
- Configuring PID_3Step V1 (Page 6079)

Principle

For more information, see the following FAQs in the Siemens Industry Online Support:

- Entry ID 68011827 (https://support.industry.siemens.com/cs/ww/en/view/68011827)

13.1.4.2 PID_3Step V2 (S7-1200, S7-1500)

Configuring PID_3Step V2 (S7-1200, S7-1500)

Basic settings V2 (S7-1200, S7-1500)

Introduction V2 (S7-1200, S7-1500)

Configure the following properties of the "PID_3Step" technology object under "Basic settings" in the Inspector window or in the configuration window:

- Physical quantity
- Control logic
- Start-up behavior after reset

- Setpoint (only in the Inspector window)
- Process value (only in the Inspector window)
- Output value (only in the Inspector window)
- Position feedback (only in the Inspector window)

Setpoint, process value, output value and position feedback

You can only configure the setpoint, process value, output value and position feedback in the Inspector window of the programming editor. Select the source for each value:

- Instance DB
  The value saved in the instance DB is used.
  Value must be updated in the instance DB by the user program.
  There should be no value at the instruction.
  Change via HMI possible.
- Instruction
  The value connected to the instruction is used.
  The value is written to the instance DB each time the instruction is called.
  No change via HMI possible.

Control mode V2 (S7-1200, S7-1500)

Physical quantity

Select the physical quantity and unit of measurement for setpoint, process value, and disturbance variable in the "Controller type" group. Setpoint, process value, and disturbance variable is displayed in this unit of measurement.

Control logic

An increase of the output value is generally intended to cause an increase in the process value. This is referred to as a normal control logic.

PID_3Step does not work with negative proportional gain. Select the check box "Invert control logic" to reduce the process value with a higher output value.

Examples

- Opening the drain valve will reduce the level of a container's contents.
- Increasing cooling will reduce the temperature.

Startup characteristics

1. To switch to "Inactive" mode after CPU restart, clear the "Activate Mode after CPU restart" check box.
   To switch to the operating mode saved in the Mode parameter after CPU restart, select the "Activate Mode after CPU restart" check box.
2. In the "Set Mode to" drop-down list, select the mode that is to be enabled after a complete download to the device.
   After a complete download to the device, PID_3Step starts in the selected operating mode. With each additional restart, PID_3Step starts in the mode that was last saved in Mode.

Example

You have selected the "Activate Mode after CPU restart" check box and the entry "Pretuning" in the "Set Mode to" list. After a complete download to the device, PID_3Step starts in the "Pretuning" mode. If pretuning is still active, PID_3Step starts in "Pretuning" mode again after restart of the CPU. If pretuning was successfully completed and automatic mode is active, PID_3Step starts in "Automatic mode" after restart of the CPU.

Setpoint V2 (S7-1200, S7-1500)

Procedure

Proceed as follows to define a fixed setpoint:

1. Select "Instance DB".
2. Enter a setpoint, e.g. 80° C.
3. Delete any entry in the instruction.

Proceed as follows to define a variable setpoint:

1. Select "Instruction".
2. Enter the name of the REAL variable in which the setpoint is saved.
   Program-controlled assignment of various values to the REAL variable is possible, for example for the time controlled change of the setpoint.

Process value V2 (S7-1200, S7-1500)

PID_3Step will scale the value of the analog input to the physical quantity if you use the analog input value directly.

You will need to write a program for processing if you wish first to process the analog input value. The process value is, for example, not directly proportional to the value at the analog input. The processed process value must be in floating point format.

Procedure

Proceed as follows to use the analog input value without processing:

1. Select the entry "Input_PER" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the address of the analog input.

Proceed as follows to use the processed process value in floating point format:

1. Select the entry "Input" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the name of the variable in which the processed process value is saved.

Position feedback V2 (S7-1200, S7-1500)

Position feedback configuration depends upon the actuator used.

- Actuator without position feedback
- Actuator with digital endstop signals
- Actuator with analog position feedback
- Actuator with analog position feedback and endstop signals

Actuator without position feedback

Proceed as follows to configure PID_3Step for an actuator without position feedback:

1. Select the entry "No Feedback" in the drop-down list "Feedback".

Actuator with digital endstop signals

Proceed as follows to configure PID_3Step for an actuator with endstop signals:

1. Select the entry "No Feedback" in the drop-down list "Feedback".
2. Activate the "Actuator endstop signals" check box.
3. Select "Instruction" as source for Actuator_H and Actuator_L.
4. Enter the addresses of the digital inputs for Actuator_H and Actuator_L.

Actuator with analog position feedback

Proceed as follows to configure PID_3Step for an actuator with analog position feedback:

1. Select the entry "Feedback" or "Feedback_PER" in the drop-down list "Feedback".
   - Use the analog input value for Feedback_PER. Configure Feedback_PER scaling in the actuator settings.
   - Process the analog input value for Feedback using your user program.
2. Select "Instruction" as source.
3. Enter the address of the analog input or the variable of your user program.

Actuator with analog position feedback and endstop signals

Proceed as follows to configure PID_3Step for an actuator with analog position feedback and endstop signals:

1. Select the entry "Feedback" or "Feedback_PER" in the drop-down list "Feedback".
2. Select "Instruction" as source.
3. Enter the address of the analog input or the variable of your user program.
4. Activate the "Actuator endstop signals" check box.
5. Select "Instruction" as source for Actuator_H and Actuator_L.
6. Enter the addresses of the digital inputs for Actuator_H and Actuator_L.

Output value V2 (S7-1200, S7-1500)

PID_3Step offers an analog output value (Output_PER) and digital output values (Output_UP, Output_DN). Your actuator will determine which output value you use.

- Output_PER
  The actuator has a relevant motor transition time and is triggered via an analog output and controlled with a continuous signal, e.g. 0...10 V or 4...20 mA. The value at Output_PER corresponds to the target position of the valve, e.g. Output_PER = 13824, when the valve is to be opened by 50%.
  For auto-tuning and anti windup behavior, for example, PID_3Step takes into consideration that the analog output value has a delayed effect on the process due to the motor transition time. If no relevant motor transition time is in effect in your process (e.g. with solenoid valves), so that the output value has a direct and full effect on the process, use PID_Compact instead.
- Output_UP, Output_DN
  The actuator has a relevant motor transition time and is controlled by two digital outputs.
  Output_UP moves the valve in the open state direction.
  Output_DN moves the valve in the closed state direction.

The motor transition time is taken into consideration in the calculation of the analog output value as well as in the calculation of the digital output values. It is mainly required for correct operation during auto-tuning and the anti-windup behavior. You should therefore configure the motor transition time under "Actuator settings" with the value that the motor requires to move the actuator from the closed to the opened state.

Procedure

Proceed as follows to use the analog output value:

1. Select the entry "Output (analog)" in the drop-down list "Output".
2. Select "Instruction".
3. Enter the address of the analog output.

Proceed as follows to use the digital output value:

1. Select the entry "Output (digital)" in the drop-down list "Output".
2. Select "Instruction" for Output_UP and Output_DN.
3. Enter the addresses of the digital outputs.

Proceed as follows to process the output value using the user program:

1. Select the entry corresponding to the actuator in the drop-down list "Output".
2. Select "Instruction".
3. Enter the name of the tag you are using to process the output value.
4. Transfer the processed output value to the actuator by means of an analog or digital CPU output.

Process value settings V2 (S7-1200, S7-1500)

Process value scaling V2 (S7-1200, S7-1500)

If you have configured the use of Input_PER in the basic setting, you must convert the value of the analog input to the physical quantity of the process value. The current configuration is displayed in the Input_PER display.

Input_PER will be scaled using a low and high value pair if the process value is directly proportional to the value of the analog input.

Procedure

To scale the process value, follow these steps:

1. Enter the low pair of values in the "Scaled low process value" and "Low" text boxs.
2. Enter the high pair of values in the "Scaled high process value" and "High" input boxes.

Default settings for the value pairs are stored in the hardware configuration. To use the value pairs from the hardware configuration, follow these steps:

1. Select the PID_3Step instruction in the programming editor.
2. Interconnect Input_PER with an analog input in the basic settings.
3. Click the "Automatic setting" button in the process value settings.

The existing values will be overwritten with the values from the hardware configuration.

Process value limits V2 (S7-1200, S7-1500)

You must specify an appropriate absolute high limit and low limit for the process value as limit values for your controlled system. As soon as the process value violates these limits, an error occurs (ErrorBits = 0001h). Tuning is canceled when the process value limits are violated. You can specify how PID_3Step responds to errors in automatic mode in the actuator settings.

Final controlling element settings V2 (S7-1200, S7-1500)

Final controlling element V2 (S7-1200, S7-1500)

Actuator-specific times

Configure the motor transition time and the minimum ON and OFF times to prevent damage to the actuator. You can find the specifications in the actuator data sheet.

The motor transition time is the time in seconds the motor requires to move the actuator from the closed to the opened state. You can measure the motor transition time during commissioning.

The motor transition time is taken into consideration in the calculation of the analog output value as well as in the calculation of the digital output values. It is mainly required for correct operation during auto-tuning and the anti-windup behavior.

If no relevant motor transition time is in effect in your process (e.g. with solenoid valves), so that the output value has a direct and full effect on the process, use PID_Compact instead.

The motor transition time is retentive. If you enter the motor transition time manually, you must completely download PID_3Step.

Downloading technology objects to device (Page 6000)

If you are using "Output_UP" or "Output_DN", you can reduce the switching frequency with the minimum on and minimum OFF time.

The on or off times calculated are totaled in automatic mode and only become effective when the sum is greater than or equal to the minimum on or OFF time.

Manual_UP = TRUE or Manual_DN = TRUE in manual mode operates the actuator for at least the minimum ON or OFF time.

If you have selected the analog output value Output_PER, the minimum ON time and the minimum OFF time are not evaluated and cannot be changed.

Reaction to error

PID_3Step is preset so that the controller stays active in most cases in the event of an error. If errors occur frequently in controller mode, this default reaction has a negative effect on the control response. In this case, check the Errorbits parameter and eliminate the cause of the error.

NOTICE
Your system may be damaged. If you output "Current value while error pending" or "Substitute output value while error pending" in the event of an error, PID_3Step remains in automatic mode even if the process value limits are violated. This may damage your system. It is essential to configure how your controlled system reacts in the event of an error to protect your system from damage.

PID_3Step generates a programmable output value in the case of an error:

- Current value
  PID_3Step is switched off and no longer modifies the actuator position.
- Current value for error while error is pending
  The controller functions of PID_3Step are switched off and the position of the actuator is no longer changed.
  If the following errors occur in automatic mode, PID_3Step returns to automatic mode as soon as the errors are no longer pending.
  - 0002h: Invalid value at Input_PER parameter.
  - 0200h: Invalid value at Input parameter.
  - 0400h: Calculation of output value failed.
  - 1000h: Invalid value at Setpoint parameter.
  - 2000h: Invalid value at Feedback_PER parameter.
  - 4000h: Invalid value at Feedback parameter.
  - 8000h: Error during digital position feedback.
  - 20000h: Invalid value at SavePosition tag.

  If one or more of the following errors occur, PID_3Step stays in automatic mode:
  - 0001h: The Input parameter is outside the process value limits.
  - 0800h: Sampling time error
  - 40000h: Invalid value at Disturbance parameter.

  PID_3Step remains in manual mode if an error occurs in manual mode.
  If an error occurs during tuning or transition time measurement, PID_3Step switches to the mode in which tuning or transition time measurement was started. Only in the event of the following error is tuning not aborted:
  - 0020h: Pretuning is not permitted during fine tuning.

- Substitute output value
  PID_3Step moves the actuator to the substitute output value and then switches off.
- Substitute output value while error is pending
  PID_3Step moves the actuator to the substitute output value. When the substitute output value is reached, PID_3Step reacts as it does with "Current value for while error is pending".

Enter the substitute output value in "%".

Only substitute output values 0% and 100% can be approached precisely in the case of actuators without analog position feedback. A substitute output value not equal to 0% or 100% is approached via an internally simulated position feedback. This procedure does not, however, allow the exact approach of substitute output value.

All substitute output values can be approached precisely with actuators with analog position feedback.

Scaling V2 position feedback (S7-1200, S7-1500)

Scaling position feedback

If you have configured the use of Feedback_PER in the basic settings, you will need to convert the value of the analog input into %. The current configuration will be displayed in the "Feedback" display.

Feedback_PER is scaled using a low and high value pair.

1. Enter the low pair of values in the "Low endstop" and "Low" input boxes.
2. Enter the high pair of values in the "High endstop" and "High" input boxes.

"Low endstop" must be less than "High endstop"; "Low" must be less than "High".

The valid values for "High endstop" and "Low endstop" depend upon:

- No Feedback, Feedback, Feedback_PER
- Output (analog), Output (digital)

Output	Feedback	Low endstop	High endstop
Output (digital)	No Feedback	Cannot be set (0.0%)	Cannot be set (100.0%)
Output (digital)	Feedback	-100.0% or 0.0%	0.0% or +100.0%
Output (digital)	Feedback_PER	-100.0% or 0.0%	0.0% or +100.0%
Output (analog)	No Feedback	Cannot be set (0.0%)	Cannot be set (100.0%)
Output (analog)	Feedback	-100.0% or 0.0%	0.0% or +100.0%
Output (analog)	Feedback_PER	-100.0% or 0.0%	0.0% or +100.0%

Output value limits V2 (S7-1200, S7-1500)

Limiting the output value

You can exceed or undershoot the output value limits during the transition time measurement and with mode = 10. The output value is limited to these values in all other modes.

Enter the absolute output value limits in the "Output value high limit" and "Output value low limit" input boxes. The output value limits must be within "Low endstop" and "High endstop".

If no Feedback is available and Output (digital) is set, you cannot limit the output value. Output_UP and Output_DN are then reset upon Actuator_H = TRUE or Actuator_L = TRUE. If no endstop signals are available, Output_UP and Output_DN are reset after a travel time of 150% of the motor actuating time.

The default value of 150% can be adjusted using the tagConfig.VirtualActuatorLimit. As of PID_3Step Version 2.3 the monitoring and limiting of the travel time can be deactivated with Config.VirtualActuatorLimit = 0.0.

Advanced settings V2 (S7-1200, S7-1500)

Actual value monitoring V2 (S7-1200, S7-1500)

Configure a warning high and low limit for the process value in the "Process value monitoring" configuration window. If one of the warning limits is exceeded or undershot during operation, a warning will be displayed at the PID_3Step instruction:

- At the InputWarning_H output parameter if the warning high limit has been exceeded
- At the InputWarning_L output parameter if the warning low limit has been undershot

The warning limits must be within the process value high and low limits.

The process value high and low limits will be used if you do not enter values.

Example

Process value high limit = 98° C; warning high limit = 90° C

Warning low limit = 10° C; process value low limit = 0° C

PID_3Step will respond as follows:

Process value	InputWarning_H	InputWarning_L	Error-Bits	Operating mode
> 98° C	TRUE	FALSE	0001h	As configured
≤ 98° C and > 90° C	TRUE	FALSE	0000h	Automatic mode
≤ 90° C and ≥ 10° C	FALSE	FALSE	0000h	Automatic mode
< 10° C and ≥ 0° C	FALSE	TRUE	0000h	Automatic mode
< 0° C	FALSE	TRUE	0001h	As configured

In the actuator settings, you can configure the response of PID_3Step when the process value high limit or low limit is violated.

PID parameters V2 (S7-1200, S7-1500)

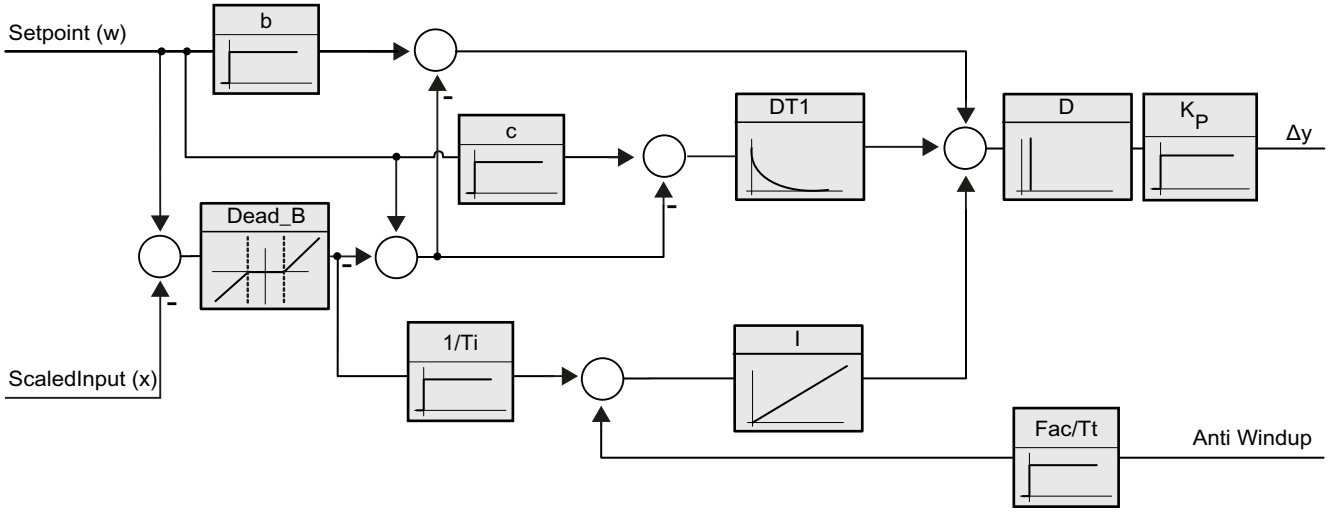
The PID parameters are displayed in the "PID Parameters" configuration window. The PID parameters will be adapted to your controlled system during controller tuning. You do not need to enter the PID parameters manually.

The PID algorithm operates according to the following equation:

$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
Δy	Output value of the PID algorithm
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integration time
a	Derivative delay coefficient (derivative delay T1 = a × T_D)
T_D	Derivative action time
c	Derivative action weighting

The diagram below illustrates the integration of the parameters into the PID algorithm:



All PID parameters are retentive. If you enter the PID parameters manually, you must completely download PID_3Step.

Downloading technology objects to device (Page 6000)

Proportional gain

The value specifies the proportional gain of the controller. PID_3Step does not work with a negative proportional gain. Control logic is inverted under Basic settings > Controller type.

Integration time

The integration time determines the time behavior of the integral action. The integral action is deactivated with integration time = 0.0.

Derivative action time

The derivative action time determines the time behavior of the derivative action. Derivative action is deactivated with derivative action time = 0.0.

Derivative delay coefficient

The derivative delay coefficient delays the effect of the derivative action.

Derivative delay = derivative action time × derivative delay coefficient

- 0.0: Derivative action is effective for one cycle only and therefore almost not effective.
- 0.5: This value has proved useful in practice for controlled systems with **one** dominant time constant.
- > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed.

Proportional action weighting

The proportional action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Proportional action for setpoint change is fully effective
- 0.0: Proportional action for setpoint change is not effective

The proportional action is always fully effective when the process value is changed.

Derivative action weighting

The derivative action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Derivative action is fully effective upon setpoint change
- 0.0: Derivative action is not effective upon setpoint change

The derivative action is always fully effective when the process value is changed.

PID algorithm sampling time

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the PID_3Step sampling time. All other functions of PID_3Step are executed at every call.

Dead band width

The deadband suppresses the noise component in the steady controller state. The dead band width specifies the size of the dead band. The dead band is off if the dead band width is 0.0.

If values not equal to 1.0 are configured for the proportional action weighting or the derivative action weighting, setpoint changes even within the dead zone affect the output value. Process value changes within the dead zone do not affect the output value, regardless of the weighting.

See also

Selection of the controller structure for specified controlled systems (Page 5993)

Commissioning PID_3Step V2 (S7-1200, S7-1500)

Pretuning V2 (S7-1200, S7-1500)

The pretuning determines the process response to a pulse of the output value and searches for the point of inflection. The tuned PID parameters are calculated as a function of the maximum slope and dead time of the controlled system. You obtain the best PID parameters when you perform pretuning and fine tuning.

The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise. This is most likely the case in operating modes "Inactive" and "manual mode". The PID parameters are backed up before being recalculated.

The setpoint is frozen during pretuning.

Requirement

- The PID_3Step instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- The motor transition time has been configured or measured.
- PID_3Step is in one of the following modes: "Inactive", "Manual mode", or "Automatic mode".
- The setpoint and the process value lie within the configured limits (see "Process value settings" configuration).

Procedure

To perform pretuning, follow these steps:

1. Double-click the "PID_3Step > Commissioning" entry in the project tree.
2. Select the entry "Pretuning" in the "Tuning mode" drop-down list in the working area "Tuning".
3. Click the "Start" icon.
   - An online connection will be established.
   - Value recording is started.
   - Pretuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

   Note

   Click the "Stop" icon when the progress bar has reached 100% and it is to be assumed the controller tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

If pretuning was performed without an error message, the PID parameters have been tuned. PID_3Step switches to automatic mode and uses the tuned parameters. The tuned PID parameters will be retained during power OFF and a restart of the CPU.

If pretuning is not possible, PID_3Step responds with the configured reaction to errors.

Fine tuning V2 (S7-1200, S7-1500)

Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are tuned for the operating point from the amplitude and frequency of this oscillation. All PID parameters are recalculated from the results. PID parameters from fine tuning usually have better master control and disturbance characteristics than PID parameters from pretuning. You obtain the best PID parameters when you perform pretuning and fine tuning.

PID_3Step automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. The PID parameters are backed up before being recalculated.

The setpoint is frozen during fine tuning.

Requirement

- The PID_3Step instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- The motor transition time has been configured or measured.

- The setpoint and the process value lie within the configured limits (see "Process value settings" configuration).
- The control loop has stabilized at the operating point. The operating point is reached when the process value corresponds to the setpoint.
- No disturbances are expected.
- PID_3Step is in inactive mode, automatic mode or manual mode.

Process depends on initial situation

Fine tuning proceeds as follows when started from:

- Automatic mode
  Start fine tuning from automatic mode if you wish to improve the existing PID parameters through tuning.
  PID_3Step controls the system using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start.
- Inactive or manual mode
  Pretuning is always started first. The determined PID parameters will be used for control until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start.

Procedure

To perform fine tuning, follow these steps:

1. Select the entry "Fine tuning" in the "Tuning mode" drop-down list.
2. Click the "Start" icon.
   - An online connection will be established.
   - Value recording is started.
   - The process of fine tuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

Note

Click the "Stop" icon in the "Tuning mode" group when the progress bar has reached 100% and it is to be assumed the controller tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

If no errors occurred during fine tuning, the PID parameters have been tuned. PID_3Step switches to automatic mode and uses the tuned parameters. The tuned PID parameters will be retained during power OFF and a restart of the CPU.

If errors occurred during fine tuning, PID_3Step responds with the configured response to errors.

Commissioning with manual PID parameters V2 (S7-1200, S7-1500)

Requirement

- The PID_3Step instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- The motor transition time has been configured or measured.
- PID_3Step is in "inactive" mode.
- The setpoint and the process value lie within the configured limits (see "Process value settings" configuration).

Procedure

Proceed as follows to commission PID_3Step with manual PID parameters:

1. Double-click on "PID_3Step > Configuration" in the project tree.
2. Click on "Advanced settings > PID Parameters" in the configuration window.
3. Select the check box "Enable direct input".
4. Enter the PID parameters.
5. Double-click the "PID_3Step > Commissioning" entry in the project tree.
6. Establish an online connection to the CPU.
7. Load the PID parameters to the CPU.
8. Click the "Start PID_3Step" icon.

Result

PID_3Step changes to automatic mode and controls using the current PID parameters.

See also

PID parameters V2 (Page 6070)

Measuring the motor transition time V2 (S7-1200, S7-1500)

Introduction

PID_3Step requires the motor transition time to be as accurate as possible for good controller results. The data in the actuator documentation contains average values for this type of actuator. The value for the specific actuator used may differ.

You can measure the motor transition time during commissioning if you are using actuators with position feedback or endstop signals. The output value limits are not taken into consideration during the motor transition time measurement. The actuator can travel to the high or the low endstop.

The motor transition time cannot be measured if neither position feedback nor endstop signals are available.

Actuators with analog position feedback

Proceed as follows to measure motor transition time with position feedback:

Requirement

- Feedback or Feedback_PER has been selected in the basic settings and the signal has been connected.
- An online connection to the CPU has been established.

1. Select the "Use position feedback" check box.
2. Enter the position to which the actuator is to be moved in the "Target position" input field. The current position feedback (starting position) will be displayed. The difference between "Target position" and "Position feedback" must be at least 50% of the valid output value range.
3. Click the "Start" icon.

Result

The actuator is moved from the starting position to the target position. Time measurement starts immediately and ends when the actuator reaches the target position. The motor transition time is calculated according to the following equation:

Motor transition time = (output value high limit – output value low limit) × Measuring time / AMOUNT (target position – starting position).

The progress and status of transition time measurement are displayed. The transition time measured is saved in the instance data block on the CPU and displayed in the "Measured transition time" field. When the transition time measurement is ended and ActivateRecoverMode = TRUE, PID_3Step switches to the operating mode from which the

transition time measurement was started. If the transition time measurement is ended and ActivateRecoverMode = FALSE, PID_3Step changes to "Inactive" mode.

Note

Click on the icon  "Upload measured transition time" to load the motor transition time measured to the project.

Actuators with endstop signals

Proceed as follows to measure the transition time of actuators with endstop signals:

Requirement

- The "Endstop signals" check box in the basic settings has been selected and Actuator_H and Actuator_L are connected.
- An online connection to the CPU has been established.

Proceed as follows to measure motor transition time with endstop signals:

1. Select the "Use actuator endstop signals" check box.
2. Select the direction in which the actuator is to be moved.
   - Open - Close - Open
     The actuator is moved first to the high endstop, then to the low endstop and then back to the high endstop.
   - Close - Open - Close
     The actuator is moved first to the low endstop, then to the high endstop and then back to the low endstop.
3. Click the "Start" icon.

Result

The actuator is moved in the selected direction. Time measurement will start once the actuator has reached the first endstop and will end when the actuator reaches this endstop for the second time. The motor transition time is equal to the time measured divided by two.

The progress and status of transition time measurement are displayed. The transition time measured is saved in the instance data block on the CPU and displayed in the "Measured transition time" field. When the transition time measurement is ended and ActivateRecoverMode = TRUE, PID_3Step switches to the operating mode from which the transition time measurement was started. If the transition time measurement is ended and ActivateRecoverMode = FALSE, PID_3Step changes to "Inactive" mode.

Cancelling transition time measurement

PID_3Step switches to "Inactive" mode if you cancel transition time measurement by pressing the Stop button.

Simulating PID_3Step V2 with PLCSIM (S7-1200, S7-1500)

Note

Simulation with PLCSIM

The simulation of PID_3Step V2.x with PLCSIM for CPU S7-1200 is not supported.

PID_3Step V2.x can only be simulated for CPU S7-1500 with PLCSIM.

For the simulation with PLCSIM, the time behavior of the simulated PLC is not exactly identical to that of a "real" PLC. The actual cycle clock of a cyclic interrupt OB can have larger fluctuations with a simulated PLC than with "real" PLCs.

In the standard configuration, PID_3Step determines the time between calls automatically and monitors them for fluctuations.

For a simulation of PID_3Step with PLCSIM, for example, a sampling time error ((ErrorBits = DW#16#00000800) can therefore be detected.

This results in ongoing tuning being aborted.

The response in automatic mode depends on the value of the ActivateRecoverMode tag.

To prevent this from happening, you should configure PID_3Step for simulation with PLCSIM as follows:

- CycleTime.EnEstimation = FALSE
- CycleTime.EnMonitoring = FALSE
- CycleTime.Value: Assign the cycle clock of the calling cyclic interrupt OB in seconds to this tag.

13.1.4.3 PID_3Step V1 (S7-1200, S7-1500)

Configuring PID_3Step V1 (S7-1200, S7-1500)

Basic settings V1 (S7-1200, S7-1500)

Introduction V1 (S7-1200, S7-1500)

Configure the following properties of the "PID_3Step" technology object under "Basic settings" in the Inspector window or in the configuration window:

- Physical quantity
- Control logic
- Start-up behavior after reset
- Setpoint (only in the Inspector window)
- Process value (only in the Inspector window)
- Output value (only in the Inspector window)
- Position feedback (only in the Inspector window)

Setpoint, process value, output value and position feedback

You can only configure the setpoint, process value, output value and position feedback in the Inspector window of the programming editor. Select the source for each value:

- Instance DB
  The value saved in the instance DB is used.
  Value must be updated in the instance DB by the user program.
  There should be no value at the instruction.
  Change via HMI possible.
- Instruction
  The value connected to the instruction is used.
  The value is written to the instance DB each time the instruction is called.
  No change via HMI possible.

Control mode V1 (S7-1200, S7-1500)

Physical quantity

Select the unit of measurement and physical quantity for the setpoint and process value in the "Controller type" group. The setpoint and process value will be displayed in this unit.

Control logic

An increase of the output value is generally intended to cause an increase in the process value. This is referred to as a normal control logic.

PID_3Step does not work with negative proportional gain. Select the check box "Invert control logic" to reduce the process value with a higher output value.

Examples

- Opening the drain valve will reduce the level of a container's contents.
- Increasing cooling will reduce the temperature.

Start-up behavior after reset

To change straight to the last active mode after restarting the CPU, select the "Enable last mode after CPU restart" check box.

PID_3Step will remain in "Inactive" mode if the check box is cleared.

Setpoint V1 (S7-1200, S7-1500)

Procedure

Proceed as follows to define a fixed setpoint:

1. Select "Instance DB".
2. Enter a setpoint, e.g. 80° C.
3. Delete any entry in the instruction.

Proceed as follows to define a variable setpoint:

1. Select "Instruction".
2. Enter the name of the REAL variable in which the setpoint is saved.
   Program-controlled assignment of various values to the REAL variable is possible, for example for the time controlled change of the setpoint.

Process value V1 (S7-1200, S7-1500)

PID_3Step will scale the value of the analog input to the physical quantity if you use the analog input value directly.

You will need to write a program for processing if you wish first to process the analog input value. The process value is, for example, not directly proportional to the value at the analog input. The processed process value must be in floating point format.

Procedure

Proceed as follows to use the analog input value without processing:

1. Select the entry "Input_PER" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the address of the analog input.

Proceed as follows to use the processed process value in floating point format:

1. Select the entry "Input" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the name of the variable in which the processed process value is saved.

Position feedback V1 (S7-1200, S7-1500)

Position feedback configuration depends upon the actuator used.

- Actuator without position feedback
- Actuator with digital endstop signals
- Actuator with analog position feedback
- Actuator with analog position feedback and endstop signals

Actuator without position feedback

Proceed as follows to configure PID_3Step for an actuator without position feedback:

1. Select the entry "No Feedback" in the drop-down list "Feedback".

Actuator with digital endstop signals

Proceed as follows to configure PID_3Step for an actuator with endstop signals:

1. Select the entry "No Feedback" in the drop-down list "Feedback".
2. Activate the "Actuator endstop signals" check box.
3. Select "Instruction" as source for Actuator_H and Actuator_L.
4. Enter the addresses of the digital inputs for Actuator_H and Actuator_L.

Actuator with analog position feedback

Proceed as follows to configure PID_3Step for an actuator with analog position feedback:

1. Select the entry "Feedback" or "Feedback_PER" in the drop-down list "Feedback".
   - Use the analog input value for Feedback_PER. Configure Feedback_PER scaling in the actuator settings.
   - Process the analog input value for Feedback using your user program.
2. Select "Instruction" as source.
3. Enter the address of the analog input or the variable of your user program.

Actuator with analog position feedback and endstop signals

Proceed as follows to configure PID_3Step for an actuator with analog position feedback and endstop signals:

1. Select the entry "Feedback" or "Feedback_PER" in the drop-down list "Feedback".
2. Select "Instruction" as source.
3. Enter the address of the analog input or the variable of your user program.
4. Activate the "Actuator endstop signals" check box.
5. Select "Instruction" as source for Actuator_H and Actuator_L.
6. Enter the addresses of the digital inputs for Actuator_H and Actuator_L.

Output value V1 (S7-1200, S7-1500)

PID_3Step offers an analog output value (Output_PER) and digital output values (Output_UP, Output_DN). Your actuator will determine which output value you use.

- Output_PER
  The actuator has a relevant motor transition time and is triggered via an analog output and controlled with a continuous signal, e.g. 0...10 V or 4...20 mA. The value at Output_PER corresponds to the target position of the valve, e.g. Output_PER = 13824, when the valve is to be opened by 50%.
  For auto-tuning and anti windup behavior, for example, PID_3Step takes into consideration that the analog output value has a delayed effect on the process due to the motor transition time. If no relevant motor transition time is in effect in your process (e.g. with solenoid valves), so that the output value has a direct and full effect on the process, use PID_Compact instead.
- Output_UP, Output_DN
  The actuator has a relevant motor transition time and is controlled by two digital outputs.
  Output_UP moves the valve in the open state direction.
  Output_DN moves the valve in the closed state direction.

The motor transition time is taken into consideration in the calculation of the analog output value as well as in the calculation of the digital output values. It is mainly required for correct operation during auto-tuning and the anti-windup behavior. You should therefore configure the motor transition time under "Actuator settings" with the value that the motor requires to move the actuator from the closed to the opened state.

Procedure

Proceed as follows to use the analog output value:

1. Select the entry "Output (analog)" in the drop-down list "Output".
2. Select "Instruction".
3. Enter the address of the analog output.

Proceed as follows to use the digital output value:

1. Select the entry "Output (digital)" in the drop-down list "Output".
2. Select "Instruction" for Output_UP and Output_DN.
3. Enter the addresses of the digital outputs.

Proceed as follows to process the output value using the user program:

1. Select the entry corresponding to the actuator in the drop-down list "Output".
2. Select "Instruction".
3. Enter the name of the tag you are using to process the output value.
4. Transfer the processed output value to the actuator by means of an analog or digital CPU output.

Process value settings V1 (S7-1200, S7-1500)

Configure the scaling of your process value and specify the process value absolute limits In the "Process value settings" configuration window.

Scaling the process value

If you have configured the use of Input_PER in the basic settings, you will need to convert the value of the analog input into the physical quantity of the process value. The current configuration will be displayed in the Input_PER display.

Input_PER will be scaled using a low and high value pair if the process value is directly proportional to the value of the analog input.

1. Enter the low pair of values in the "Scaled low process value" and "Low" input fields.
2. Enter the high pair of values in the "Scaled high process value" and "High" input boxes.

Default settings for the value pairs are saved in the hardware configuration. Proceed as follows to use the value pairs from the hardware configuration:

1. Select the instruction PID_3Step in the programming editor.
2. Connect Input_PER to an analog input in the basic settings.
3. Click on the "Automatic setting" button in the process value settings.
   The existing values will be overwritten with the values from the hardware configuration.

Monitoring process value

Specify the absolute high and low limit of the process value. You must enter reasonable limits for your controlled system. Reasonable limits are important during optimization to obtain optimal PID parameters. The default for the "High limit process value" is 120 %. At the I/O input, the process value can be a maximum of 18% higher than the standard range (overrange). This setting ensures that an error is no longer signaled due to a violation of the "Process value high limit". Only a wire-break and a short-circuit are recognized and PID_3Step reacts according to the configured reaction to error.

NOTICE
Your system may be damaged.
If you set very high process value limits (for example $-3.4*10^{38}$...$+3.4*10^{38}$), process value monitoring will be disabled. Your system may then be damaged if an error occurs. You need to configure useful process value limits for your controlled system.

V1 final controlling element setting (S7-1200, S7-1500)

Actuator-specific times

Configure the motor transition time and the minimum ON and OFF times to prevent damage to the actuator. You can find the specifications in the actuator data sheet.

The motor transition time is the time in seconds the motor requires to move the actuator from the closed to the opened state. The maximum time that the actuator is moved in one direction is 110% of the motor transition time. You can measure the motor transition time during commissioning.

The motor transition time is taken into consideration in the calculation of the analog output value as well as in the calculation of the digital output values. It is mainly required for correct operation during auto-tuning and the anti-windup behavior.

If no relevant motor transition time is in effect in your process (e.g. with solenoid valves), so that the output value has a direct and full effect on the process, use PID_Compact instead.

If you are using "Output_UP" or "Output_DN", you can reduce the switching frequency with the minimum on and minimum OFF time.

The on or off times calculated are totaled in automatic mode and only become effective when the sum is greater than or equal to the minimum on or OFF time.

A rising edge at Manual_UP or Manual_DN in manual mode will operate the actuator for at least the minimum on or OFF time.

If you have selected the analog output value Output_PER, the minimum ON time and the minimum OFF time are not evaluated and cannot be changed.

Reaction to error

PID_3Step is preset so that the controller stays active in most cases in the event of an error. If errors occur frequently in controller mode, this default reaction has a negative effect on the control response. In this case, check the Errorbits parameter and eliminate the cause of the error.

PID_3Step generates a programmable output value in response to an error:

- Current value
  PID_3Step is switched off and no longer modifies the actuator position.
- Current value for error while error is pending
  The controller functions of PID_3Step are switched off and the position of the actuator is no longer changed.
  If the following errors occur in automatic mode, PID_3Step returns to automatic mode as soon as the errors are no longer pending.
  - 0002h: Invalid value at Input_PER parameter.
  - 0200h: Invalid value at Input parameter.
  - 0800h: Sampling time error
  - 1000h: Invalid value at Setpoint parameter.
  - 2000h: Invalid value at Feedback_PER parameter.
  - 4000h: Invalid value at Feedback parameter.
  - 8000h: Error during digital position feedback.

  If one of these error occurs in manual mode, PID_3Step remains in manual mode.
  If an error occurs during the tuning or transition time measurement, PID_3Step is switched off.

- Substitute output value
  PID_3Step moves the actuator to the substitute output value and then switches off.
- Substitute output value while error is pending
  PID_3Step moves the actuator to the substitute output value. When the substitute output value is reached, PID_3Step reacts as it does with "Current value for while error is pending".

Enter the substitute output value in "%".

Only substitute output values 0% and 100% can be approached precisely in the case of actuators without analog position feedback. The actuator is moved in one direction at 110% of the motor transition time to ensure the high or low endstop is reached. There endstop signals take priority. A substitute output value not equal to 0% or 100% is approached via an internally simulated position feedback. This procedure does not, however, allow the exact approach of substitute output value.

All substitute output values can be approached precisely with actuators with analog position feedback.

Scaling position feedback

If you have configured the use of Feedback_PER in the basic settings, you will need to convert the value of the analog input into %. The current configuration will be displayed in the "Feedback" display.

Feedback_PER is scaled using a low and high value pair.

1. Enter the low pair of values in the "Low endstop" and "Low" input boxes.
2. Enter the high pair of values in the "High endstop" and "High" input boxes.

"Low endstop" must be less than "High endstop"; "Low" must be less than "High".

The valid values for "High endstop" and "Low endstop" depend upon:

- No Feedback, Feedback, Feedback_PER
- Output (analog), Output (digital)

Output	Feedback	Low endstop	High endstop
Output (digital)	No Feedback	Cannot be set (0.0%)	Cannot be set (100.0%)
Output (digital)	Feedback	-100.0% or 0.0%	0.0% or +100.0%
Output (digital)	Feedback_PER	-100.0% or 0.0%	0.0% or +100.0%
Output (analog)	No Feedback	Cannot be set (0.0%)	Cannot be set (100.0%)
Output (analog)	Feedback	-100.0% or 0.0%	0.0% or +100.0%
Output (analog)	Feedback_PER	-100.0% or 0.0%	0.0% or +100.0%

Limiting the output value

You can only exceed or undershoot the output value limits during the transition time measurement. The output value is limited to these values in all other modes.

Enter the absolute output value limits in the "Output value high limit" and "Output value low limit" input boxes. The output value limits must be within "Low endstop" and "High endstop".

If no Feedback is available and Output (digital) is set, you cannot limit the output value. The digital outputs are reset with Actuator_H = TRUE or Actuator_L = TRUE, or after a travel time amounting to 110% of the motor transition time.

Advanced settings V1 (S7-1200, S7-1500)

Actual value monitoring V1 (S7-1200, S7-1500)

Configure a warning high and low limit for the process value in the "Process value monitoring" configuration window. If one of the warning limits is exceeded or undershot during operation, a warning will be displayed at the PID_3Step instruction:

- At the InputWarning_H output parameter if the warning high limit has been exceeded
- At the InputWarning_L output parameter if the warning low limit has been undershot

The warning limits must be within the process value high and low limits.

The process value high and low limits will be used if you do not enter values.

Example

Process value high limit = 98° C; warning high limit = 90° C

Warning low limit = 10° C; process value low limit = 0° C

PID_3Step will respond as follows:

Process value	InputWarning_H	InputWarning_L	Operating mode
> 98° C	TRUE	FALSE	Inactive
≤ 98° C and > 90° C	TRUE	FALSE	Automatic mode
≤ 90° C and ≥ 10° C	FALSE	FALSE	Automatic mode
< 10° C and ≥ 0° C	FALSE	TRUE	Automatic mode
< 0° C	FALSE	TRUE	Inactive

PID parameters V1 (S7-1200, S7-1500)

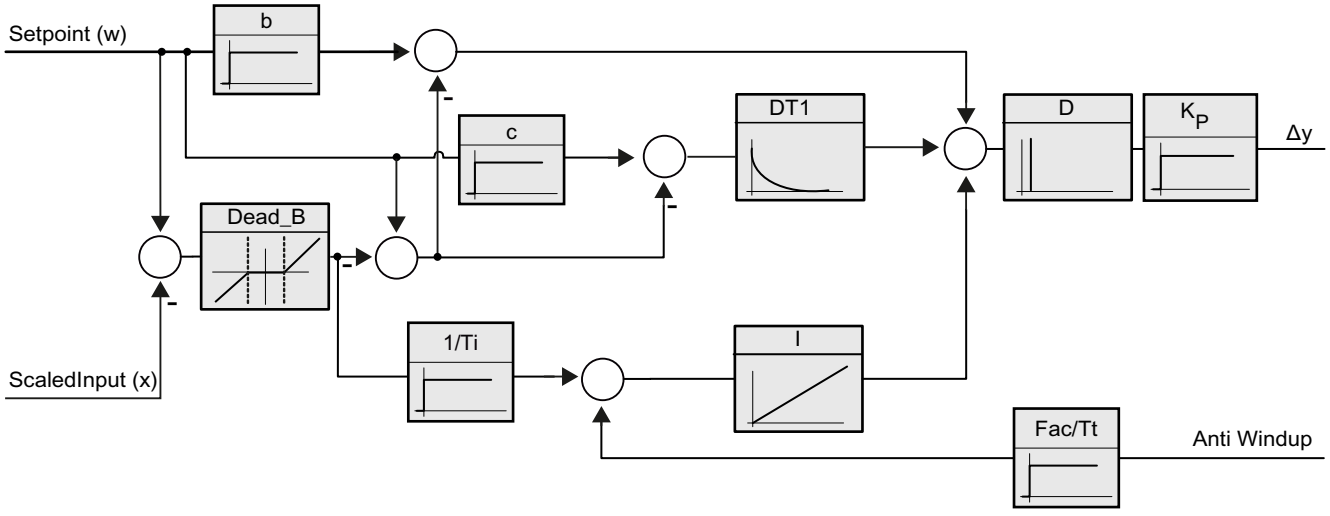
The PID parameters are displayed in the "PID Parameters" configuration window. The PID parameters will be adapted to your controlled system during controller tuning. You do not need to enter the PID parameters manually.

The PID algorithm operates according to the following equation:

$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
Δy	Output value of the PID algorithm
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integration time
a	Derivative delay coefficient (derivative delay T1 = a × T_D)
T_D	Derivative action time
c	Derivative action weighting

The diagram below illustrates the integration of the parameters into the PID algorithm:



All PID parameters are retentive. If you enter the PID parameters manually, you must completely download PID_3Step.

Downloading technology objects to device (Page 6000)

Proportional gain

The value specifies the proportional gain of the controller. PID_3Step does not work with a negative proportional gain. Control logic is inverted under Basic settings > Controller type.

Integration time

The integration time determines the time behavior of the integral action. The integral action is deactivated with integration time = 0.0.

Derivative action time

The derivative action time determines the time behavior of the derivative action. Derivative action is deactivated with derivative action time = 0.0.

Derivative delay coefficient

The derivative delay coefficient delays the effect of the derivative action.

Derivative delay = derivative action time × derivative delay coefficient

- 0.0: Derivative action is effective for one cycle only and therefore almost not effective.
- 0.5: This value has proved useful in practice for controlled systems with **one** dominant time constant.
- > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed.

Proportional action weighting

The proportional action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Proportional action for setpoint change is fully effective
- 0.0: Proportional action for setpoint change is not effective

The proportional action is always fully effective when the process value is changed.

Derivative action weighting

The derivative action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Derivative action is fully effective upon setpoint change
- 0.0: Derivative action is not effective upon setpoint change

The derivative action is always fully effective when the process value is changed.

PID algorithm sampling time

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of the PID algorithm represents the time between two calculations of the output value. It is calculated during tuning and rounded to a multiple of the PID_3Step sampling time. All other functions of PID_3Step are executed at every call.

Dead band width

The deadband suppresses the noise component in the steady controller state. The dead band width specifies the size of the dead band. The dead band is off if the dead band width is 0.0.

If values not equal to 1.0 are configured for the proportional action weighting or the derivative action weighting, setpoint changes even within the dead zone affect the output value.

Process value changes within the dead zone do not affect the output value, regardless of the weighting.

See also

Selection of the controller structure for specified controlled systems (Page 5993)

Commissioning PID_3Step V1 (S7-1200, S7-1500)

Commissioning V1 (S7-1200, S7-1500)

You can monitor the setpoint, process value and output value over time in the "Tuning" working area. The following commissioning functions are supported in the curve plotter:

- Controller pretuning
- Controller fine tuning
- Monitoring the current closed-loop control in the trend view

All functions require an online connection to the CPU to have been established.

Basic handling

- Select the desired sampling time in the "Sampling time" drop-down list.
  All values in the tuning working area are updated in the selected update time.
- Click the "Start" icon in the measuring group if you want to use the commissioning functions.
  Value recording is started. The current values for the setpoint, process value and output value are entered in the trend view. Operation of the commissioning window is enabled.
- Click the "Stop" icon if you want to end the commissioning functions.
  The values recorded in the trend view can continue to be analyzed.
- Closing the commissioning window will terminate recording in the trend view and delete the recorded values.

Pretuning V1 (S7-1200, S7-1500)

The pretuning determines the process response to a pulse of the output value and searches for the point of inflection. The tuned PID parameters are calculated as a function of the maximum slope and dead time of the controlled system.

The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise. The PID parameters are backed up before being recalculated.

The setpoint is frozen during pretuning.

Requirement

- The PID_3Step instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- PID_3Step is in "inactive" or "manual" mode.
- The setpoint and the process value lie within the configured limits (see "Process value settings" configuration).

Procedure

To perform pretuning, follow these steps:

1. Double-click the "PID_3Step > Commissioning" entry in the project tree.
2. Select the entry "Pretuning" in the "Tuning mode" drop-down list in the working area "Tuning".
3. Click the "Start" icon.
   - An online connection will be established.
   - Value recording is started.
   - Pretuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

   Note

   Click the "Stop" icon when the progress bar has reached 100% and it is to be assumed the controller tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

If pretuning was performed without an error message, the PID parameters have been tuned. PID_3Step switches to automatic mode and uses the tuned parameters. The tuned PID parameters will be retained during power OFF and a restart of the CPU.

If pretuning is not possible, PID_3Step changes to "Inactive" mode.

Fine tuning V1 (S7-1200, S7-1500)

Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are optimized for the operating point from the amplitude and frequency of this oscillation. All PID parameters are recalculated on the basis of the findings. PID parameters from fine tuning usually have better master control and disturbance behavior than PID parameters from pretuning.

PID_3Step automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. The PID parameters are backed up before being recalculated.

The setpoint is frozen during fine tuning.

Requirement

- The PID_3Step instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- The motor transition time has been configured or measured.
- The setpoint and the process value lie within the configured limits (see "Process value settings" configuration).
- The control loop has stabilized at the operating point. The operating point is reached when the process value corresponds to the setpoint.
- No disturbances are expected.
- PID_3Step is in inactive mode, automatic mode or manual mode.

Process depends on initial situation

Fine tuning proceeds as follows when started in:

- Automatic mode
  Start fine tuning in automatic mode if you wish to improve the existing PID parameters using controller tuning.
  PID_3Step will regulate using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start.
- Inactive or manual mode
  Pretuning is always started first. The PID parameters established will be used for adjustment until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start.

Procedure

Proceed as follows to carry out "fine tuning":

1. Select the entry "Fine tuning" in the "Tuning mode" drop-down list.
2. Click the "Start" icon.
   - An online connection will be established.
   - Value recording is started.
   - The process of fine tuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

   Note

   Click the "Stop" icon in the "Tuning mode" group when the progress bar has reached 100% and it is to be assumed the controller tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

The PID parameters will have been optimized if fine tuning has been executed without errors. PID_3Step changes to automatic mode and uses the optimized parameters. The optimized PID parameters will be retained during power OFF and a restart of the CPU.

If errors occurred during fine tuning, PID_3Step will change to "inactive" mode.

Commissioning with manual PID parameters V1 (S7-1200, S7-1500)

Procedure

Proceed as follows to commission PID_3Step with manual PID parameters:

1. Double-click on "PID_3Step > Configuration" in the project tree.
2. Click on "Advanced settings > PID Parameters" in the configuration window.
3. Select the check box "Enable direct input".
4. Enter the PID parameters.
5. Double-click on "PID_3Step > Commissioning" in the project tree.
6. Establish an online connection to the CPU.
7. Load the PID parameters to the CPU.
8. Click on the "Activate controller" icon.

Result

PID_3Step changes to automatic mode and controls using the current PID parameters.

Measuring the motor transition time V1 (S7-1200, S7-1500)

Introduction

PID_3Step requires the motor transition time to be as accurate as possible for good controller results. The data in the actuator documentation contains average values for this type of actuator. The value for the specific actuator used may differ.

You can measure the motor transition time during commissioning if you are using actuators with position feedback or endstop signals. The output value limits are not taken into consideration during the motor transition time measurement. The actuator can travel to the high or the low endstop.

The motor transition time cannot be measured if neither position feedback nor endstop signals are available.

Actuators with analog position feedback

Proceed as follows to measure motor transition time with position feedback:

Requirement

- Feedback or Feedback_PER has been selected in the basic settings and the signal has been connected.
- An online connection to the CPU has been established.

1. Select the "Use position feedback" check box.
2. Enter the position to which the actuator is to be moved in the "Target position" input field. The current position feedback (starting position) will be displayed. The difference between "Target position" and "Position feedback" must be at least 50% of the valid output value range.
3. Click the  "Start transition time measurement" icon.

Result

The actuator is moved from the starting position to the target position. Time measurement starts immediately and ends when the actuator reaches the target position. The motor transition time is calculated according to the following equation:

Motor transition time = (output value high limit – output value low limit) × Measuring time / AMOUNT (target position – starting position).

The progress and status of transition time measurement are displayed. The transition time measured is saved in the instance data block on the CPU and displayed in the "Measured transition time" field. PID_3Step will change to "Inactive" mode once transition time measurement is complete.

Note

Click on the icon  "Upload measured transition time" to load the motor transition time measured to the project.

Actuators with endstop signals

Proceed as follows to measure the transition time of actuators with endstop signals:

Requirement

- The "Endstop signals" check box in the basic settings has been selected and Actuator_H and Actuator_L are connected.
- An online connection to the CPU has been established.

Proceed as follows to measure motor transition time with endstop signals:

1. Select the "Use actuator endstop signals" check box.
2. Select the direction in which the actuator is to be moved.
   - Open - Close - Open
     The actuator is moved first to the high endstop, then to the low endstop and then back to the high endstop.
   - Close - Open - Close
     The actuator is moved first to the low endstop, then to the high endstop and then back to the low endstop.
3. Click the  "Start transition time measurement" icon.

Result

The actuator is moved in the selected direction. Time measurement will start once the actuator has reached the first endstop and will end when the actuator reaches this endstop for the second time. The motor transition time is equal to the time measured divided by two.

The progress and status of transition time measurement are displayed. The transition time measured is saved in the instance data block on the CPU and displayed in the "Measured transition time" field. PID_3Step will change to "Inactive" mode once transition time measurement is complete.

Cancelling transition time measurement

PID_3Step will change to "Inactive" mode immediately if you cancel transition time measurement. The actuator will stop being moved. You can reactive PID-3Step in the curve plotter.

Simulating PID_3Step V1 with PLCSIM (S7-1200, S7-1500)

Note

Simulation with PLCSIM

For the simulation with PLCSIM, the time behavior of the simulated PLC is not exactly identical to that of a "real" PLC. The actual cycle clock of a cyclic interrupt OB can have larger fluctuations with a simulated PLC than with "real" PLCs.

In the standard configuration, PID_3Step determines the time between calls automatically and monitors them for fluctuations.

For a simulation of PID_3Step with PLCSIM, for example, a sampling time error (ErrorBits = DW#16#00000800) can therefore be detected.

This results in ongoing tuning being aborted.

The response in automatic mode depends on the value of the ActivateRecoverMode tag.

To prevent this from happening, you should configure PID_3Step for simulation with PLCSIM as follows:

- CycleTime.EnEstimation = FALSE
- CycleTime.EnMonitoring = FALSE
- CycleTime.Value: Assign the cycle clock of the calling cyclic interrupt OB in seconds to this tag.

13.1.5 Using PID_Temp (S7-1200, S7-1500)

13.1.5.1 Technology object PID_Temp (S7-1200, S7-1500)

The PID_Temp technology object provides a continuous PID controller with integrated tuning. PID_Temp is especially designed for temperature control and is suited for heating or heating/cooling applications. Two outputs are available for this purpose, one each for heating and cooling. PID_Temp can also be used for other control tasks. PID_Temp is cascadable and can be used in manual or automatic mode.

PID_Temp continuously acquires the measured process value within a control loop and compares it with the set setpoint. From the resulting control deviations, the instruction PID_Temp calculates the output value for heating and/or cooling which is used to adjust the process value to the setpoint. The output values for the PID controller consist of three actions:

- Proportional action
  The proportional action of the output value increases in proportion to the control deviation.
- Integral action
  The integral action of the output value increases until the control deviation has been balanced.
- Derivative action
  The derivative action increases with the rate of change of control deviation. The process value is corrected to the setpoint as quickly as possible. The derivative action will be reduced again if the rate of change of control deviation drops.

The instruction PID_Temp calculates the proportional, integral and derivative parameters for your controlled system during "pretuning". "Fine tuning" can be used to tune the parameters further. You do not need to manually determine the parameters.

Either a fixed cooling factor or two PID parameter sets can be used for heating-and-cooling applications.

Additional information

- Overview of software controller (Page 5994)
- Add technology objects (Page 5997)
- Configure technology objects (Page 5998)
- Configuring PID_Temp (Page 6097)

13.1.5.2 Configuring PID_Temp (S7-1200, S7-1500)

Basic settings (S7-1200, S7-1500)

Introduction (S7-1200, S7-1500)

Configure the following properties of the "PID_Temp" technology object under "Basic settings" in the Inspector window or in the configuration window:

- Physical quantity
- Start-up behavior after reset
- Source and input of the setpoint (only in the Inspector window)
- Selection of the process value
- Source and input of the process value (only in the Inspector window)
- Selection of the heating output value
- Source and input of the heating output value (only in the Inspector window)
- Activation and selection of the cooling output value
- Source and input of the cooling output value (only in the Inspector window)
- Activation of PID_Temp as master or slave of a cascade
- Number of slaves
- Selection of the master (only in the Inspector window)

Setpoint, process value, heating output value and cooling output value

You can select the source and enter values or tags for the setpoint, process value, heating output value and cooling output value in the Inspector window of the programming editor.

Select the source for each value:

- Instance DB:
  The value saved in the instance DB is used. The value must be updated by the user program in the instance DB. There should be no value at the instruction. Can be changed using HMI.
- Instruction:
  The value connected to the instruction is used. The value is written to the instance DB each time the instruction is called. Cannot be changed using HMI.

Controller type (S7-1200, S7-1500)

Physical quantity

Select the unit of measurement and physical quantity for the setpoint and the process value in the "Controller type" group. The setpoint and the process value are displayed in this unit.

Startup characteristics

1. To switch to "Inactive"mode after CPU restart, clear the "Activate Mode after CPU restart"check box.
   To switch to the operating mode saved in the Mode parameter after CPU restart, select the "Activate Mode after CPU restart" check box.
2. In the "Set Mode to" drop-down list, select the mode that is to be enabled after a complete download to the device.
   After a complete "Download to device", PID_Temp starts in the selected operating mode. With each additional restart, PID_Temp starts in the mode that was last saved in Mode. When selecting pretuning or fine tuning, you also have to set or reset the Heat.EnableTuning and Cool.EnableTuning tags in order to choose between tuning for heating and tuning for cooling.

Example:

You have selected the "Activate Mode after CPU restart" check box and the "Pretuning" entry in the "Set Mode to" list. After a complete "Download to device", PID_Temp starts in the "Pretuning" mode. If pretuning is still active, PID_Temp starts in "Pretuning" mode again after restart of the CPU (heating/cooling depends on the tags Heat.EnableTuning and Cool.EnableCooling). If pretuning was successfully completed and automatic mode is active, PID_Temp starts in "Automatic mode" after restart of the CPU.

Setpoint (S7-1200, S7-1500)

Procedure

Proceed as follows to define a fixed setpoint:

1. Select "Instance DB".
2. Enter a setpoint, e.g. 80° C.
3. Delete any entry in the instruction.

Proceed as follows to define a variable setpoint:

1. Select "Instruction".
2. Enter the name of the REAL tag in which the setpoint is saved.
   Program-controlled assignment of various values to the REAL tag is possible, for example for the time-controlled change of the setpoint.

Process value (S7-1200, S7-1500)

PID_Temp will scale the value of the analog input to the physical quantity if you use the analog input value directly.

You will need to write a program for processing if you wish first to process the analog input value. The process value is, for example, not directly proportional to the value at the analog input. The processed process value must be in floating point format.

Procedure

Proceed as follows to use the analog input value without processing:

1. Select the entry "Input_PER" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the address of the analog input.

Proceed as follows to use the processed process value in floating point format:

1. Select the entry "Input" in the drop-down list "Input".
2. Select "Instruction" as source.
3. Enter the name of the variable in which the processed process value is saved.

Heating and cooling output value (S7-1200, S7-1500)

The PID_Temp instruction provides a PID controller with integrated tuning for temperature processes. PID_Temp is suitable for heating or heating-and-cooling applications.

PID_Temp provides the following output values. Your actuator will determine which output value you use.

- OutputHeat
  Heating output value (floating-point format): The output value for heating needs to be processed by the user program, for example, because of non-linear actuator response.
- OutputHeat_PER
  Analog heating output value: The actuator for heating is triggered via an analog output and controlled with a continuous signal, e.g. 0...10 V, 4...20 mA.
- OutputHeat_PWM
  Pulse-width modulated heating output value: The actuator for heating is controlled via a digital output. Pulse width modulation creates variable ON and OFF times.
- OutputCool
  Cooling output value (floating-point format): The output value for cooling needs to be processed by the user program, for example because of non-linear actuator response.

- OutputCool_PER
  Analog cooling output value: The actuator for cooling is triggered via an analog output and controlled with a continuous signal, e.g. 0...10 V, 4...20 mA.
- OutputCool_PWM
  Pulse-width modulated cooling output value: The actuator for cooling is controlled via a digital output. Pulse width modulation creates variable ON and OFF times.

The cooling output is only available if it was activated via the "Activate cooling" check box.

- If the check box is cleared, the output value of the PID algorithm (PidOutputSum) is scaled and output at the outputs for heating.
- If the check box is selected, positive output values of the PID algorithm (PidOutputSum) are scaled and output at the outputs for heating. Negative output values of the PID algorithm are scaled and output at the outputs for cooling. You can choose between two methods for output value calculation at the output settings.

Note

Note:

- The OutputHeat_PWM, OutputHeat_PER, OutputCool_PWM, OutputCool_PER outputs are only calculated if you select these correspondingly from the drop-down list.
- The OutputHeat output is always calculated.
- The OutputCool output is calculated if the check box for cooling is selected.
- The "Activate cooling" check box is only available if the controller is not configured as a master in a cascade.

Procedure

Proceed as follows to use the analog output value:

1. Select the entry "OutputHeat_PER" or "OutputCool_PER" in the drop-down list "OutputHeat" or "OutputCool".
2. Select "Instruction".
3. Enter the address of the analog output.

Proceed as follows to use the pulse-width modulated output value:

1. Select the entry "OutputHeat_PWM" or "OutputCool_PWM" in the drop-down list "OutputHeat" or "OutputCool".
2. Select "Instruction".
3. Enter the address of the digital output.

Proceed as follows to process the output value using the user program:

1. Select the entry "OutputHeat" or "OutputCool" in the drop-down list "OutputHeat" or "OutpuCool".
2. Select "Instruction".

3. Enter the name of the variable you are using to process the output value.
4. Transfer the processed output value to the actuator by means of an analog or digital CPU output.

Cascade (S7-1200, S7-1500)

If a PID_Temp instance receives its setpoint from a higher-level master controller and outputs its output value in turn to a subordinate slave controller, this PID_Temp instance is both a master controller and a slave controller simultaneously. Both configurations listed below then have to be carried out for such a PID_Temp instance. This is the case, for example, for the middle PID_Temp instance in a cascade control system with three concatenated measured variables and three PID_Temp instances.

Configuring a controller as master in a cascade

A master controller defines the setpoint of a slave controller with its output.

In order to use PID_Temp as master in a cascade, you have to deactivate the cooling in the basic settings. In order to configure this PID_Temp instance as a master controller in a cascade, activate the "Controller is master" check box. The selection of the output value for heating is set automatically to OutputHeat.

OutputHeat_PWM and OutputHeat_PER cannot be used at a master in a cascade.

Subsequently specify the number of directly subordinate slave controllers that receive their setpoint from this master controller.

If no own scaling function is used when assigning the OutputHeat parameter of the master to the Setpoint parameter of the slave, it may be necessary to adapt the output value limits and the output scaling of the master to the setpoint/process value range of the slave. This can be done in the output settings of the master in the "OutputHeat / OutputCool" section.

Configuring a controller as a slave in a cascade

A slave controller receives its setpoint (Setpoint parameter) from the output of its master controller (OutputHeat parameter).

In order to configure this PID_Temp instance as a slave controller in a cascade, activate the "Controller is slave" check box in the basic settings.

Subsequently select the PID_Temp instance that is to be used as the master controller for this slave controller in the Inspector window of the programming editor. The Master and Setpoint parameters of the slave controller are interconnected with the selected master controller through this selection (the existing interconnections at these parameters are overwritten). This interconnection allows the exchange of information and the setpoint specification between master and slave. If required, the interconnection can be changed subsequently at the Setpoint parameter of the slave controller in order, for example, to insert an additional filter. The interconnection at the parameter Master may not be changed subsequently.

The "Controller is master" check box has to be selected and the number of slaves has to be configured correctly at the selected master controller. The master controller has to be called before the slave controller in the same cyclic interrupt OB.

Additional information

Additional information about program creation, configuration and commissioning when PID_Temp is used in cascade control systems is available under Cascade control with PID_Temp (Page 6126).

Process value settings (S7-1200, S7-1500)

Process value limits (S7-1200, S7-1500)

You must specify an appropriate absolute high limit and low limit for the process value as limit values for your controlled system. As soon as the process value violates these limits, an error occurs (ErrorBits = 0001h). Tuning is canceled when the process value limits are violated. You can specify how PID_Temp responds to errors in automatic mode in the output settings.

Process value scaling (S7-1200, S7-1500)

If you have configured the use of Input_PER in the basic settings, you will need to convert the value of the analog input into the physical quantity of the process value. The current configuration is displayed in the Input_PER display.

Input_PER is scaled using a low and high value pair if the process value is directly proportional to the value of the analog input.

Procedure

To scale the process value, follow these steps:

1. Enter the low pair of values in the "Scaled low process value" and "Low" input fields.
2. Enter the high pair of values in the "Scaled high process value" and "High" input fields.

Default settings for the value pairs are saved in the hardware configuration. Proceed as follows to use the value pairs from the hardware configuration:

1. Select the instruction PID_Temp in the programming editor.
2. Interconnect Input_PER with an analog input in the basic settings.
3. Click on the "Automatic setting" button in the process value settings.

The existing values are overwritten with the values from the hardware configuration.

Output settings (S7-1200, S7-1500)

Basic settings of output (S7-1200, S7-1500)

Method for heating and cooling

If cooling is activated in the basic settings, two methods are available for calculating the PID output value:

- PID parameter switching (Config.AdvancedCooling = TRUE):
  The output value calculation for cooling takes place by means of a separate PID parameter set. Based on the calculated output value and the control deviation, the PID algorithm decides whether the PID parameter for heating or cooling is used. This method is suitable if the heating and cooling actuators have different time responses and different gains. Pretuning and fine tuning for cooling are only available if this method is selected.
- Cooling factor (Config.AdvancedCooling = FALSE):
  Output value calculation for cooling is effected with the PID parameters for heating under consideration of the configurable cooling factor Config.CoolFactor. This method is suitable if the heating and cooling actuators have a similar time response but different gains. If this method is selected, pretuning and fine tuning for cooling as well as the PID parameter set for cooling are not available. You can only execute the tuning for heating.

Cooling factor

If the cooling factor is selected as the method for heating/cooling, this factor is used in the calculation of the output value for cooling. This allows different gains of heating and cooling actuators to be taken into account.

The cooling factor is not set automatically or adjusted during tuning. You have to configure the correct cooling factor manually by using the ratio "Heating actuator gain/Cooling actuator gain".

Example: Cooling factor = 2.0 means that the heating actuator gain is twice as high as the cooling actuator gain.

The cooling factor is only effective and can only be changed if "Cooling factor" is selected as the method for heating/cooling.

Reaction to error

NOTICE
Your system may be damaged. If you output "Current value while error is pending " or "Substitute output value while error is pending" in the event of an error, PID_Temp remains in automatic mode or in manual mode. This may cause a violation of the process value limits and damage your system. It is essential to configure how your controlled system reacts in the event of an error to protect your system from damage.

PID_Temp is preset so that the controller stays active in most cases in the event of an error.

If errors occur frequently in controller mode, this default reaction has a negative effect on the control response. In this case, check the ErrorBits parameter and eliminate the cause of the error.

PID_Temp generates a programmable output value in response to an error:

- Zero (inactive)
  At all errors, PID_Temp switches to the "Inactive" operating mode and outputs the following:
  - 0.0 as PID output value (PidOutputSum)
  - 0.0 as output value for heating (OutputHeat) and output value for cooling (OutputCool)
  - 0 as analog output value for heating (OutputHeat_PER) and analog output value for cooling (OutputCool_PER)
  - FALSE as PWM output value for heating (OutputHeat_PWM) and PWM output value for cooling (OutputCool_PWM)

  This is independent of the configured output value limits and the scaling. The controller is only reactivated by a falling edge at Reset or a rising edge at ModeActivate.
- Current value while error is pending
  The error response depends on the error occurring and the operating mode.
  If one or more of the following errors occur in automatic mode, PID_Temp stays in automatic mode:
  - 0000001h: The Input parameter is outside the process value limits.
  - 0000800h: Sampling time error
  - 0040000h: Invalid value at Disturbance parameter.
  - 8000000h: Error during the calculation of the PID parameters.

  If one or more of the following errors occur in automatic mode, PID_Temp switches to "Substitute output value with error monitoring" mode and outputs the last valid PID output value (PidOutputSum):
  - 0000002h: Invalid value at Input_PER parameter.
  - 0000200h: Invalid value at Input parameter.
  - 0000400h: Calculation of output value failed.
  - 0001000h: Invalid value at Setpoint or SubstituteSetpoint parameter.

  The values at the outputs for heating and cooling resulting from the PID output value are produced by the configured output scaling.
  As soon as the errors are no longer pending, PID_Temp switches back to automatic mode.
  If an error occurs during manual mode, PID_Temp remains in manual mode and continues to use the manual value as the PID output value.
  If the manual value is invalid, the configured substitute output value is used.
  If the manual value and substitute output value are invalid, the low limit of the PID output value for heating (Config.Output.Heat.PidLowerLimit) is used.
  If the following error occurs during pretuning or fine tuning, PID_Temp remains in active mode:
  - 0000020h: Pretuning is not permitted during fine tuning.

  When any other error occurs, PID_Temp cancels the tuning and switches to the mode from which tuning was started.
- Substitute output value while error is pending

PID_Temp behaves as described at "Current value while error is pending", but outputs the configured substitute output value (SubstituteOutput) as a PID output value (PidOutputSum) in "Substitute output value with error monitoring" operating mode.
The values at the outputs for heating and cooling resulting from the PID output value are produced by the configured output scaling.
In the case of controllers with activated cooling output (Config.ActivateCooling = TRUE), enter:

– A positive substitute output value to output the value at the outputs for heating.

– A negative substitute output value to output the value at the outputs for cooling.

If the following error occurs, PID_Temp stays in "Substitute output value with error monitoring" mode and outputs the low limit of the PID output value for heating (Config.Output.Heat.PidLowerLimit):

– 0020000h: Invalid value at SubstituteOutput tag.

Output value limits and scaling (S7-1200, S7-1500)

Depending on the operating mode, the PID output value (PidOutputSum) is calculated automatically by the PID algorithm or by the manual value (ManualValue) or the configured substitute output value (SubstituteOutput).

The PID output value is limited depending on the configuration:

- If the cooling is deactivated in the basic settings (Config.ActivateCooling = FALSE), the value is limited to the high limit of the PID output value (heating) (Config.Output.Heat.PidUpperLimit) and the low limit of the PID output value (heating) (Config.Output.Heat.PidLowerLimit).
  You can configure both limits at the horizontal axis of the scaling characteristic line in the "OutputHeat / OutputCool" section. These are displayed in the "OutputHeat_PWM / OutputCool_PWM" and "OutputHeat_PER / OutputCool_PER" sections, but cannot be changed.
- If the cooling is activated in the basic settings (Config.ActivateCooling = TRUE), the value is limited to the high limit of the PID output value (Config.Output.Heat.PidUpperLimit) and the low limit of the PID output value (cooling) (Config.Output.Cool.PidLowerLimit).
  You can configure both limits at the horizontal axis of the scaling characteristic line in the "OutputHeat / OutputCool" section. These are displayed in the "OutputHeat_PWM / OutputCool_PWM" and "OutputHeat_PER / OutputCool_PER" sections, but cannot be changed.
  The low limit of the PID output value (heating) (Config.Output.Heat.PidLowerLimit) and the high limit of the PID output value (cooling) (Config.Output.Cool.PidUpperLimit) cannot be changed and have to be assigned the value 0.0.

The PID output value is scaled and output at the outputs for heating and cooling. Scaling can be specified separately for each output and is specified across 2 value pairs each, consisting of a limit value of the PID output value and a scaling value:

Output	Value pair	Parameter
OutputHeat	Value pair 1	PID output value high limit (heating) Config.Output.Heat.PidUpperLimit, Scaled high output value (heating) Config.Output.Heat.UpperScaling
	Value pair 2	PID output value low limit (heating) Config.Output.Heat.PidLowerLimit, Scaled low output value (heating) Config.Output.Heat.LowerScaling
OutputHeat_PWM	Value pair 1	PID output value high limit (heating) Config.Output.Heat.PidUpperLimit, Scaled high PWM output value (heating) Config.Output.Heat.PwmUpperScaling
	Value pair 2	PID output value low limit (heating) Config.Output.Heat.PidLowerLimit, Scaled low PWM output value (heating) Config.Output.Heat.PwmLowerScaling
OutputHeat_PER	Value pair 1	PID output value high limit (heating) Config.Output.Heat.PidUpperLimit, Scaled high analog output value (heating) Config.Output.Heat.PerUpperScaling
	Value pair 2	PID output value low limit (heating) Config.Output.Heat.PidLowerLimit, Scaled low analog output value (heating) Config.Output.Heat.PerLowerScaling
OutputCool	Value pair 1	PID output value low limit (cooling) Config.Output.Cool.PidLowerLimit, Scaled high output value (cooling) Config.Output.Cool.UpperScaling
	Value pair 2	PID output value high limit (cooling) Config.Output.Cool.PidUpperLimit, Scaled low output value (cooling) Config.Output.Cool.LowerScaling
OutputCool_PWM	Value pair 1	PID output value low limit (cooling) Config.Output.Cool.PidLowerLimit, Scaled high PWM output value (cooling) Config.Output.Cool.PwmUpperScaling
	Value pair 2	PID output value high limit (cooling) Config.Output.Cool.PidUpperLimit, Scaled low PWM output value (cooling) Config.Output.Cool.PwmLowerScaling

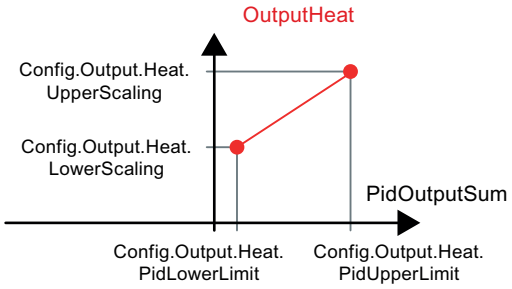
Output	Value pair	Parameter
OutputCool_PER	Value pair 1	PID output value low limit (cooling) Config.Output.Cool.PidLowerLimit, Scaled high analog output value (cooling) Config.Output.Cool.PerUpperScaling
	Value pair 2	PID output value high limit (cooling) Config.Output.Cool.PidUpperLimit, Scaled low analog output value (cooling) Config.Output.Cool.PerLowerScaling

The low limit of PID output value (heating) (Config.Output.Heat.PidLowerLimit) has to have the value 0.0, if the cooling is activated (Config.ActivateCooling = TRUE).

The high limit of PID output value (cooling) (Config.Output.Cool.PidUpperLimit) must always have the value 0.0.

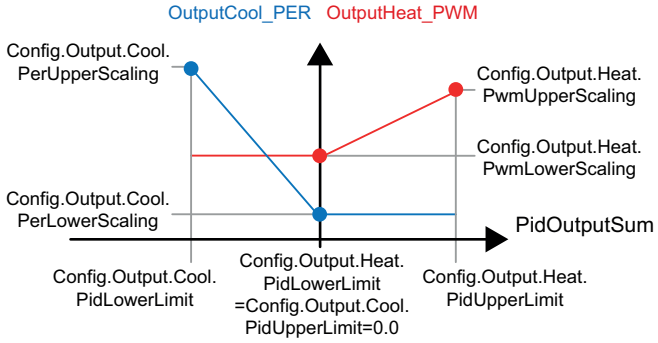
Example:

Output scaling when the OutputHeat output is used (cooling deactivated. The low limit of PID output value (heating) (Config.Output.Heat.PidLowerLimit) may be unequal to 0.0):



Example:

Output scaling when the OutputHeat_PWM and OutputCool_PER outputs are used (cooling activated. The low limit of PID output value (heating) (Config.Output.Heat.PidLowerLimit) must be 0.0):



With the exception of the "Inactive" operating mode, the value at an output always lies between its scaled high output value and the scaled low output value, for example for OutputHeat always

between the scaled high output value (heating) (Config.Output.Heat.UpperScaling) and the scaled low output value (heating) (Config.Output.Heat.LowerScaling).

If you want to limit the value at the associated output, you therefore have to adapt these scaling values as well.

You can configure the scaling values of an output at the vertical axes of the scaling characteristic line. Each output has two separate scaling values. These can only be changed for OutputHeat_PWM, OutputCool_PWM, OutputHeat_PER and OutputCool_PER if the corresponding output is selected in the basic settings. The cooling has to be activated additionally in the basic settings at all the outputs for cooling.

The trend view in the commissioning dialog box only records the values of OutputHeat and OutputCool, irrespective of the selected output in the basic settings. Therefore, if necessary, adapt the scaling values for OutputHeat or OutputCool if you use OutputHeat_PWM or OutputHeat_PER or OutputCool_PWM or OutputCool_PER and want to use the trend view in the commissioning dialog.

Advanced settings (S7-1200, S7-1500)

Process value monitoring (S7-1200, S7-1500)

Configure a warning high and low limit for the process value in the "Process value monitoring" configuration window. If one of the warning limits is exceeded or undershot during operation, a warning is displayed at the PID_Temp instruction:

- At the InputWarning_H output parameter if the warning high limit has been exceeded
- At the InputWarning_L output parameter if the warning low limit has been undershot

The warning limits must be within the process value high and low limits.

The process value high and low limits are used if you do not enter values.

Example

Process value high limit = 98° C; warning high limit = 90° C

Warning low limit = 10° C; process value low limit = 0° C

PID_Temp will respond as follows:

Process value	InputWarning_H	InputWarning_L	ErrorBits
> 98 °C	TRUE	FALSE	0001h
≤ 98° C and > 90° C	TRUE	FALSE	0000h
≤ 90° C and ≥ 10° C	FALSE	FALSE	0000h
< 10° C and ≥ 0° C	FALSE	TRUE	0000h
< 0° C	FALSE	TRUE	0001h

You can configure the response of PID_Temp when the process value high limit or low limit is violated in the output settings.

PWM limits (S7-1200, S7-1500)

The PID output value PidOutputSum is scaled and transformed via a pulse width modulation into a pulse train that is output at the output parameter OutputHeat_PWM or OutputCool_PWM. The "Sampling time of PID algorithm" represents the time between two calculations of the PID output value. The sampling time is used as time period of the pulse width modulation.

During heating, the PID output value is always calculated in the "Sampling time of PID algorithm for heating".

Calculation of the PID output value during cooling depends on the type of cooling selected in "Basic settings Output":

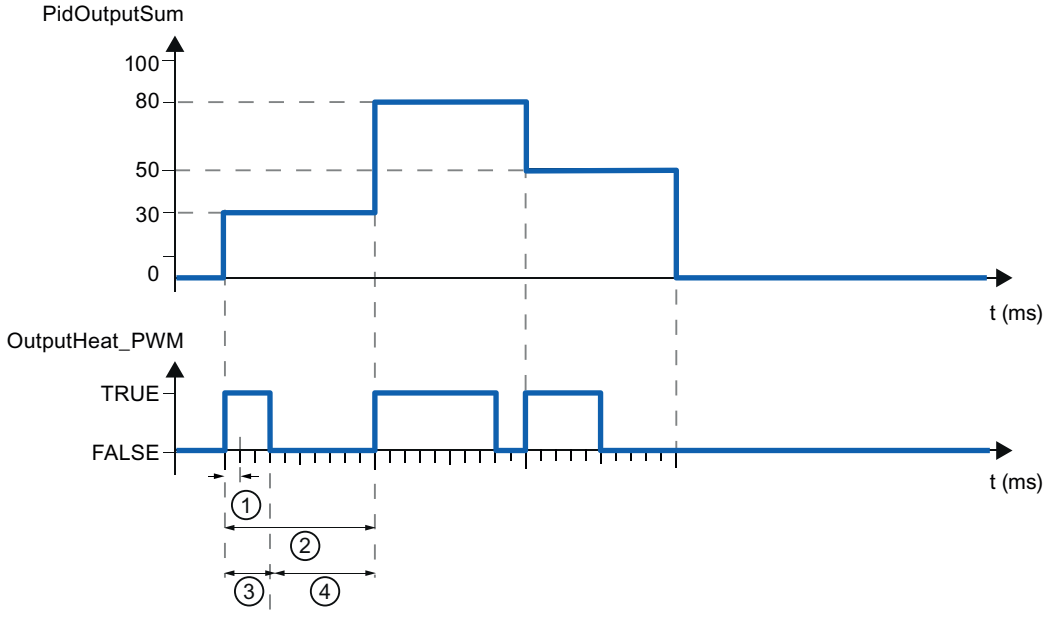
- If the cooling factor is used, the "Sampling time of PID algorithm for heating" applies.
- If the PID parameter switching is used, the "Sampling time of PID algorithm for cooling" applies.

OutputHeat_PWM and OutputCool_PWM are output in the sampling time PID_Temp (corresponds to the cycle time of the calling OB).

The PID algorithm sampling time for heating or cooling is determined during pretuning or fine tuning. If you set the PID parameters manually, you will also need to configure the PID algorithm sampling time for heating or cooling. The PID_Temp sampling time is equivalent to the cycle time of the calling OB.

The pulse duration is proportional to the PID output value and is always an integer multiple of the PID_Temp sampling time.

Example for OutputHeat_PWM



① PID_Temp sampling time
② PID algorithm sampling time for heating
③ Pulse duration
④ Break time

The "Minimum ON time" and the "Minimum OFF time" can be set separately for heating and cooling, rounded to an integer multiple of the PID_Temp sampling time.

A pulse or a break is never shorter than the minimum ON or OFF time. The inaccuracies this causes are added up and compensated in the next cycle.

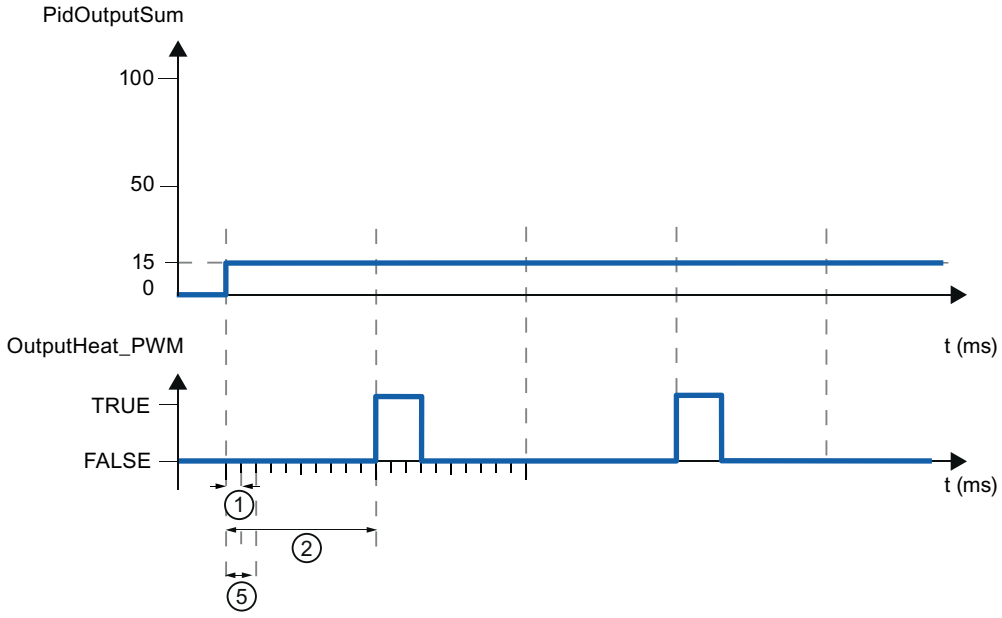
Example for OutputHeat_PWM

PID_Temp sampling time = 100 ms

PID algorithm sampling time = 1000 ms

Minimum ON time = 200 ms

The PID output value PidOutputSum amounts to 15% constantly. The smallest pulse that PID_Temp can output corresponds to 20%. In the first cycle, no pulse is output. In the second cycle, the pulse not output in the first cycle is added to the pulse of the second cycle.



① PID_Temp sampling time

② PID algorithm sampling time for heating

⑤ Minimum ON time

In order to minimize operation frequency and conserve the actuator, extend the minimum ON and OFF times.

If you have selected OutputHeat/OutputCool or OutputHeat_PER/OutputCool_PER as the output in the basic settings, the minimum ON time and the minimum OFF time are not evaluated and cannot be changed.

If the "Sampling time of PID algorithm" (Retain.CtrlParams.Heat.Cycle or Retain.CtrlParams.Cool.Cycle) and thus the period duration of the pulse width modulation is very high when OutputHeat_PWM or OutputCool_PWM is used, you can specify a deviating shorter period duration at the parameters Config.Output.Heat.PwmPeriode or

Config.Output.Cool.PwmPeriode in order to improve smoothness of the process value (see also PwmPeriode tag (Page 4379)).

Note

The minimum ON and OFF times only affect the output parameters OutputHeat_PWM or OutputCool_PWM and are not used for any pulse generators integrated in the CPU.

PID parameters (S7-1200, S7-1500)

The PID parameters are displayed in the "PID Parameters" configuration window.

If cooling is activated in the basic settings and PID parameter switching is selected as the method for heating/cooling in the output settings, two parameter sets are available: One for heating and one for cooling.

In this case, the PID algorithm decides on the basis of the calculated output value and the control deviation whether the PID parameters for heating or cooling are used.

If cooling is deactivated or the cooling factor is selected as the method for heating/cooling, the parameter set for heating is always used.

During tuning, the PID parameters are adapted to the controlled system with the exception of the dead zone width that has to be configured manually.

PID_Temp is a PIDT1 controller with anti-windup and weighting of the proportional and derivative actions.

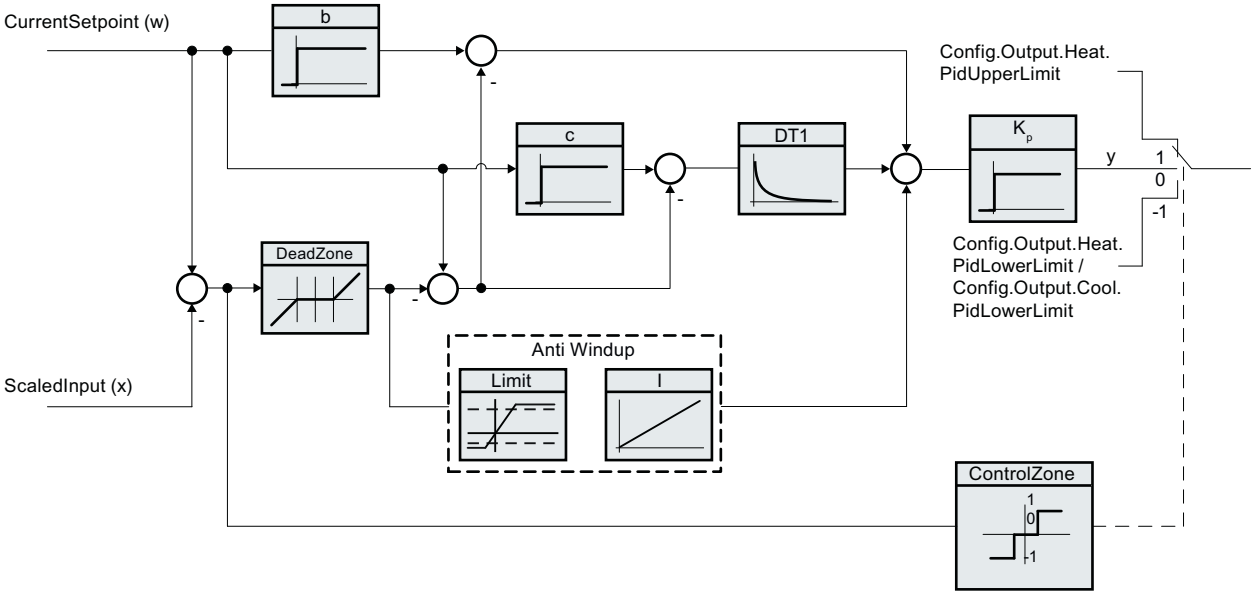
The PID algorithm operates according to the following equation (control zone and dead zone deactivated):

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description	Associated parameters of the PID_Temp instruction
y	Output value of the PID algorithm	-
K_p	Proportional gain	Retain.CtrlParams.Heat.Gain Retain.CtrlParams.Cool.Gain CoolFactor
s	Laplace operator	-
b	Proportional action weighting	Retain.CtrlParams.Heat.PWeighting Retain.CtrlParams.Cool.PWeighting
w	Setpoint	CurrentSetpoint
x	Process value	ScaledInput
T_I	Integral action time	Retain.CtrlParams.Heat.Ti Retain.CtrlParams.Cool.Ti
T_D	Derivative action time	Retain.CtrlParams.Heat.Td Retain.CtrlParams.Cool.Td

Symbol	Description	Associated parameters of the PID_Temp instruction
a	Coefficient for derivative-action delay (Derivative delay T1 = a × T_D)	Retain.CtrlParams.Heat.TdFiltRatio Retain.CtrlParams.Cool.TdFiltRatio
c	Derivative action weighting	Retain.CtrlParams.Heat.DWeighting Retain.CtrlParams.Cool.DWeighting
DeadZone	Dead zone width	Retain.CtrlParams.Heat.DeadZone Retain.CtrlParams.Cool.DeadZone
ControlZone	Control zone width	Retain.CtrlParams.Heat.ControlZone Retain.CtrlParams.Cool.ControlZone

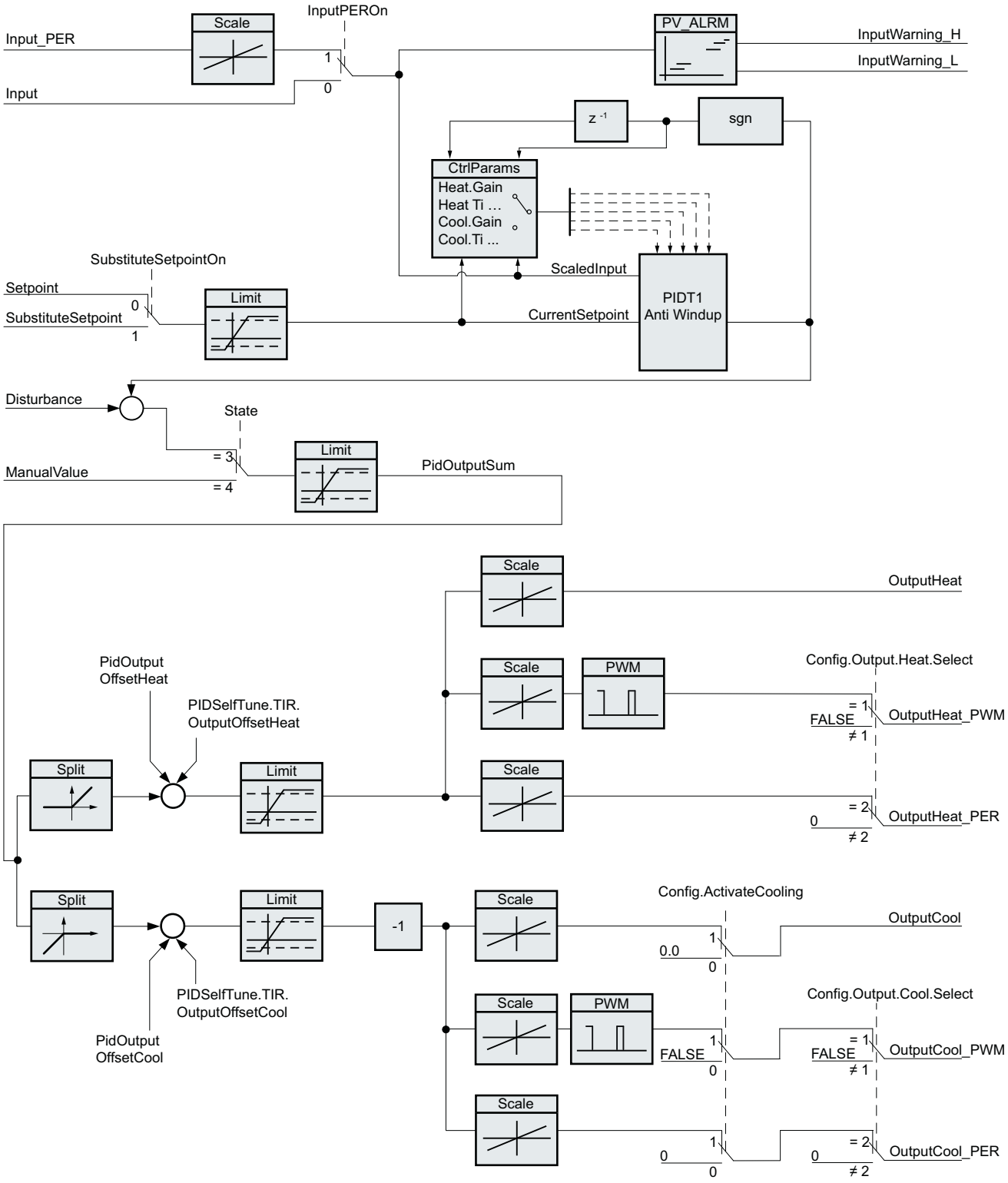
The diagram below illustrates the integration of the parameters into the PID algorithm:



All PID parameters are retentive. If you enter the PID parameters manually, you must completely download PID_Temp (Downloading technology objects to device (Page 6000)).

PID_Temp block diagram

The following block diagram shows how the PID algorithm is integrated in the PID_Temp.



Proportional gain

The value specifies the proportional gain of the controller. PID_Temp does not operate with a negative proportional gain and only supports the normal control direction, meaning that an increase in the process value is achieved by an increase in the PID output value (PidOutputSum).

Integral action time

The integral action time determines the time behavior of the integral action. The integral action is deactivated with integral action time = 0.0.

Derivative action time

The derivative action time determines the time behavior of the derivative action. Derivative action is deactivated with derivative action time = 0.0.

Derivative delay coefficient

The derivative delay coefficient delays the effect of the derivative action.

Derivative delay = derivative action time × derivative delay coefficient

- 0.0: Derivative action is effective for one cycle only and therefore almost not effective.
- 0.5: This value has proved useful in practice for controlled systems with one dominant time constant.
- > 1.0: The greater the coefficient, the longer the effect of the derivative action is delayed.

Proportional action weighting

The proportional action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Proportional action for setpoint change is fully effective
- 0.0: Proportional action for setpoint change is not effective

The proportional action is always fully effective when the process value is changed.

Derivative action weighting

The derivative action may weaken with changes to the setpoint.

Values from 0.0 to 1.0 are applicable.

- 1.0: Derivative action is fully effective upon setpoint change
- 0.0: Derivative action is not effective upon setpoint change

The derivative action is always fully effective when the process value is changed.

PID algorithm sampling time

The controlled system needs a certain amount of time to respond to changes in the output value. It is therefore not advisable to calculate the output value in every cycle. The sampling time of "PID algorithm" represents the time between two calculations of the PID output value. It is calculated during tuning and rounded to a multiple of the PID_Temp sampling time (cycle time of the cyclic interrupt OB). All other functions of PID_Temp are executed at every call.

If you use OutputHeat_PWM or OutputCool_PWM, the sampling time of the PID algorithm is used as the period duration of the pulse width modulation. The accuracy of the output signal is determined by the ratio of the PID algorithm sampling time to the cycle time of the OB. The cycle time should be no more than a tenth of the PID algorithm sampling time.

The sampling time of the PID algorithm that is used as the period duration of the pulse width modulation at OutputCool_PWM depends on the method for heating/cooling selected in "Basic settings Output":

- If the cooling factor is used, the "sampling time of the PID algorithm for heating" also applies to OutputCool_PWM.
- If PID parameter switching is used, the "sampling time PID algorithm for cooling" applies as the period duration for OutputCool_PWM.

If the sampling time of the PID algorithm and thus the period duration of the pulse width modulation is very high when OutputHeat_PWM or OutputCool_PWM is used, you can specify a deviating shorter period duration at the parameters Config.Output.Heat.PwmPeriode or Config.Output.Cool.PwmPeriode in order to improve smoothness of the process value.

Dead zone width

If the process value is affected by noise, the noise can also have an effect on the output value. The output value may fluctuate considerably when controller gain is high and the derivative action is activated. If the process value lies within the dead zone around the setpoint, the control deviation is suppressed so that the PID algorithm does not react and unnecessary fluctuations of the output value are reduced.

The dead zone width for heating or cooling is not set automatically during tuning. You have to correctly configure the dead zone width manually. The dead zone is deactivated by setting the dead zone width = 0.0.

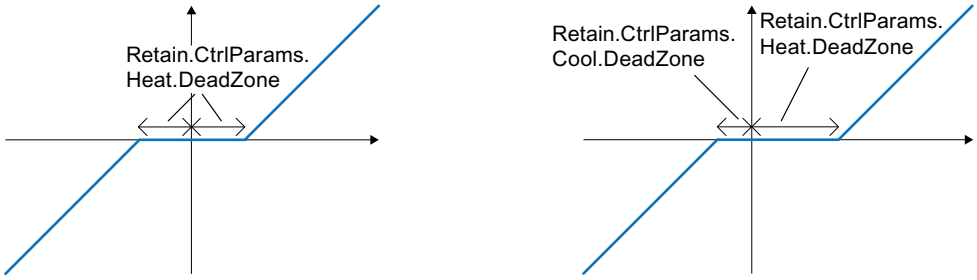
When the dead zone is switched on, the result can be a permanent control deviation (deviation between setpoint and actual value). This can have a negative effect on fine tuning.

If cooling is activated in the basic settings and PID parameter switching is selected as the method for heating/cooling in the output settings, the dead zone lies between "Setpoint - dead zone width (heating)" and "Setpoint + dead zone width (cooling)".

If cooling is deactivated in the basic settings or the cooling factor is used, the dead zone lies symmetrically between "Setpoint - dead zone width (heating)" and "Setpoint + dead zone width (heating)".

If values not equal to 1.0 are configured for the proportional action weighting or the derivative action weighting, setpoint changes even within the dead zone affect the output value.

Process value changes within the dead zone do not affect the output value, regardless of the weighting.



Dead zone with deactivated cooling or cooling factor (left) or activated cooling and PID parameter switching (right). The x / horizontal axis displays the control deviation = setpoint - process value. The y / vertical axis shows the output signal of the dead zone that is passed to the PID algorithm.

Control zone width

If the process value exits the control zone around the setpoint, the minimum or maximum output value is output. This means that the process value reaches the setpoint faster.

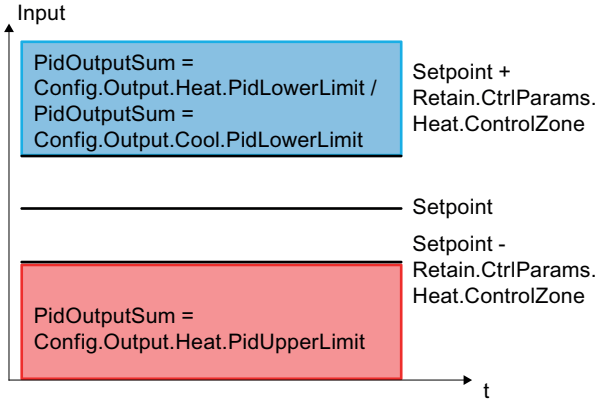
If the process value lies within the control zone around the setpoint, the output value is calculated by the PID algorithm.

The control zone width for heating or cooling is only set automatically during the pretuning, if "PID (temperature)" is selected as the controller structure for cooling or heating.

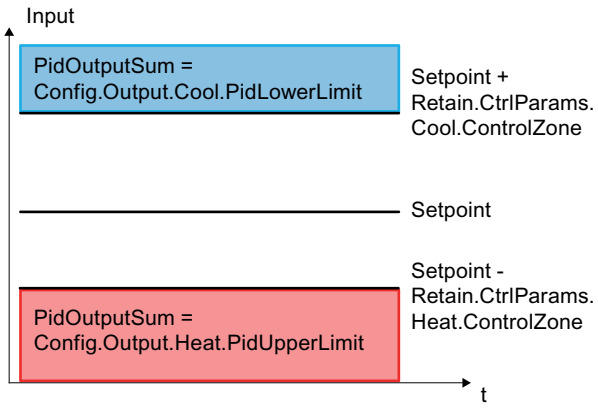
The control zone is deactivated by setting the control zone width = 3.402822e+38.

If cooling is deactivated in the basic settings or the cooling factor is used, the control zone lies symmetrically between "Setpoint - control zone width (heating)" and "Setpoint + control zone width (heating)".

If cooling is activated in the basic settings and PID parameter switching is selected as the method for heating/cooling in the output settings, the control zone lies between "Setpoint - control zone width (heating)" and "Setpoint + control zone width (cooling)".



Control zone with deactivated cooling or cooling factor.



Control zone with activated cooling and PID parameter switching.

Rule for tuning

Select whether PI or PID parameters are to be calculated in the "Controller structure" drop-down list. You can specify the rules for tuning for heating and for tuning for cooling separately.

- PID (temperature)
  Calculates PID parameters during pretuning and fine tuning.
  Pretuning is designed for temperature processes and results in a slower and rather asymptotic control response with lower overshoot than with the "PID" option. Fine tuning is identical to the "PID" option.
  The control zone width is determined automatically during pretuning only if this option is selected.
- PID
  Calculates PID parameters during pretuning and fine tuning.
- PI
  Calculates PI parameters during pretuning and fine tuning.
- User-defined
  The drop-down list displays "User-defined" if you have configured different controller structures for pretuning and fine tuning via a user program or the parameter view.

13.1.5.3 Commissioning PID_Temp (S7-1200, S7-1500)

Commissioning (S7-1200, S7-1500)

The commissioning window helps you commission the PID controller. You can monitor the values for the setpoint, process value and the output values for heating and cooling along the time axis in the trend view. The following functions are supported in the commissioning window:

- Controller pretuning
- Controller fine tuning
  Use fine tuning for fine adjustments to the PID parameters.
- Monitoring the current closed-loop control in the trend view

- Testing the controlled system by specifying a manual PID output value and a substitute setpoint
- Saving the actual values of the PID parameters to an offline project.

All functions require an online connection to the CPU.

The online connection to the CPU is established, if it does not exist already, and operation of the commissioning window is enabled by means of the "Monitor all" or "Start" buttons of the trend view.

Operation of the trend view

- Select the desired sampling time in the "Sampling time" drop-down list.
  All the values of the trend view are updated in the selected sampling time.
- Click the "Start" icon in the Measurement group if you want to use the trend view.
  Value recording is started. The current values for the setpoint, process value and output values for heating and cooling are entered in the trend view.
- Click the "Stop" icon if you want to end the trend view.
  The values recorded in the trend view can continue to be analyzed.

Closing the commissioning window will terminate recording in the trend view and delete the recorded values.

Pretuning (S7-1200, S7-1500)

The pretuning determines the process response to a jump change of the output value and searches for the point of inflection. The tuned PID parameters are calculated as a function of the maximum slope and dead time of the controlled system. You obtain the best PID parameters when you perform pretuning and fine tuning.

The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise. This is most likely the case in operating modes "Inactive" or "Manual mode". The PID parameters are backed up before being recalculated.

PID_Temp offers different pretuning types depending on the configuration:

- Pretuning heating
  A jump is output at the output value heating, the PID parameters for heating are calculated and then the setpoint is used as the control variable in automatic mode.
- Pretuning heating and cooling
  A jump is output at the output value heating.
  As soon as the process value is close to the setpoint, a jump change is output at the output value cooling.
  The PID parameters for heating (Retain.CtrlParams.Heat structure) and cooling (Retain.CtrlParams.Cool structure) are calculated and then the setpoint is used as the control variable in automatic mode.
- Pretuning cooling
  A jump is output at the output value cooling.
  The PID parameters for cooling are calculated and then the setpoint is used as the control variable in automatic mode.

If you want to tune the PID parameters for heating and cooling, you can expect a better control response with "Pretuning heating" followed by "Pretuning cooling" rather than with "Pretuning heating and cooling". However, carrying out pretuning in two steps takes more time.

General requirements

- The PID_Temp instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- PID_Temp is in one of the following modes: "Inactive", "Manual mode", or "Automatic mode".
- The setpoint and the process value lie within the configured limits (see Process value monitoring (Page 6109) configuration).

Requirements for pretuning heating

- The difference between setpoint and process value is greater than 30% of the difference between process value high limit and process value low limit.
- The distance between the setpoint and the process value is greater than 50% of the setpoint.
- The setpoint is greater than the process value.

Requirements for pretuning heating and cooling

- The cooling output in the "Basic settings" is activated (Config.ActivateCooling = TRUE).
- The PID parameter switching in the "Basic settings of output value" is activated (Config.AdvancedCooling = TRUE).
- The difference between setpoint and process value is greater than 30% of the difference between process value high limit and process value low limit.
- The distance between the setpoint and the process value is greater than 50% of the setpoint.
- The setpoint is greater than the process value.

Requirements for pretuning cooling

- The cooling output in the "Basic settings" is activated (Config.ActivateCooling = TRUE).
- The PID parameter switching in the "Basic settings of output value" is activated (Config.AdvancedCooling = TRUE).
- "Pretuning heating" or "Pretuning heating and cooling" has been carried out successfully (PIDSelfTune.SUT.ProcParHeatOk = TRUE). The same setpoint should be used for all tunings.
- The difference between setpoint and process value is smaller than 5% of the difference between process value high limit and process value low limit.

Procedure

To perform pretuning, follow these steps:

1. Double-click the "PID_Temp > Commissioning" entry in the project tree.
2. Activate the "Monitor all" button or start the trend view.
   An online connection will be established.
3. Select the desired pretuning entry from the "Tuning mode" drop-down list.
4. Click the "Start" icon.
   - Pretuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

Note

Click the "Stop" icon when the progress bar ("Progress" tag) has not changed for a long period and it is to be assumed that the tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

Result

If pretuning was performed without an error message, the PID parameters have been tuned. PID_Temp switches to automatic mode and uses the tuned parameters. The tuned PID parameters will be retained during power OFF and a restart of the CPU.

If pretuning is not possible, PID_Temp responds with the configured reaction to errors.

Fine tuning (S7-1200, S7-1500)

Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are tuned for the operating point from the amplitude and frequency of this oscillation. The PID parameters are recalculated from the results. PID parameters from fine tuning usually have better master control and disturbance characteristics than PID parameters from pretuning. You obtain the best PID parameters when you perform pretuning and fine tuning.

PID_Temp automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value. The PID parameters are backed up before being recalculated.

PID_Temp offers different fine tuning types depending on the configuration:

- Fine tuning heating:
  PID_Temp generates an oscillation of the process value with periodic changes at the output value heating and calculates the PID parameters for heating.
- Fine tuning cooling:
  PID_Temp generates an oscillation of the process value with periodic changes at the output value cooling and calculates the PID parameters for cooling.

Temporary tuning offset for heating/cooling controllers

If PID_Temp is used as a heating/cooling controller (Config.ActivateCooling = TRUE), the PID output value (PidOutputSum) at the setpoint has to fulfill the following requirements so that process value oscillation can be generated and fine tuning can be carried out successfully:

- Positive PID output value for fine tuning heating
- Negative PID output value for fine tuning cooling

If this condition is not fulfilled, you can specify a temporary offset for fine tuning that is output at the opposing output.

- Offset for cooling output (PIDSelfTune.TIR.OutputOffsetCool) with fine tuning heating.
  Before starting tuning, enter a negative tuning offset cooling that is smaller than the PID output value (PidOutputSum) at the setpoint in the stationary state.
- Offset for heating output (PIDSelfTune.TIR.OutputOffsetHeat) at fine tuning cooling
  Before starting tuning, enter a positive tuning offset heating that is greater than the PID output value (PidOutputSum) at the setpoint in the stationary state.

The defined offset is balanced by the PID algorithm so that the process value remains at the setpoint. The height of the offset allows the PID output value to be adapted correspondingly so that it fulfills the requirement mentioned above.

To avoid larger overshoots of the process value when defining the offset, it can also be increased in several steps.

If PID_Temp exits the fine tuning mode, the tuning offset is reset.

Example: Specification of an offset for fine tuning cooling

- Without offset
  - Setpoint = Process value (ScaledInput) = 80 °C
  - PID output value (PidOutputSum) = 30.0
  - Output value heating (OutputHeat) = 30.0
  - Output value cooling (OutputCool) = 0.0
    Oscillation of the process value around the setpoint cannot be generated with the cooling output alone. Fine tuning would fail here.
- With offset for heating output (PIDSelfTune.TIR.OutputOffsetHeat) = 80.0
  - Setpoint = Process value (ScaledInput) = 80 °C
  - PID output value (PidOutputSum) = -50.0
  - Output value heating (OutputHeat) = 80.0
  - Output value cooling (OutputCool) = -50.0
    Thanks to the specification of an offset for the heating output, the cooling output can now generate oscillation of the process value around the setpoint. Fine tuning can now be carried out successfully.

General requirements

- The PID_Temp instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- The setpoint and the process value lie within the configured limits (see "Process value settings" configuration).
- The control loop has stabilized at the operating point. The operating point is reached when the process value corresponds to the setpoint.
  When the dead zone is switched on, the result can be a permanent control deviation (deviation between setpoint and actual value). This can have a negative effect on fine tuning.
- No disturbances are expected.
- PID_Temp is in inactive mode, automatic mode or manual mode.

Requirements for fine tuning heating

- Heat.EnableTuning = TRUE
- Cool.EnableTuning = FALSE
- If PID_Temp is configured as a heating-and-cooling controller (Config.ActivateCooling = TRUE), the heating output has to be active at the operating point where tuning is to be carried out.
  PidOutputSum > 0.0 (see tuning offset)

Requirements for fine tuning cooling

- Heat.EnableTuning = FALSE
- Cool.EnableTuning = TRUE
- The cooling output is activated (Config.ActivateCooling = TRUE).
- The PID parameter switching is activated (Config.AdvancedCooling = TRUE).
- The cooling output has to be active at the operating point where tuning is to be carried out.
  PidOutputSum < 0.0 (see tuning offset)

Process depends on initial situation

Fine tuning can be started from the following operating modes: "Inactive", "automatic mode", or "manual mode".

Fine tuning proceeds as follows when started from:

- Automatic mode with PIDSelfTune.TIR.RunIn = FALSE (default)
  Start fine tuning from automatic mode if you wish to improve the existing PID parameters through tuning.
  PID_Temp controls the system using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start.
- Inactive, manual mode or automatic mode with PIDSelfTune.TIR.RunIn = TRUE
  An attempt is made to reach the setpoint with the minimum or maximum output value (two-point control):
  - With minimum or maximum output value heating at fine tuning heating.
  - With minimum or maximum output value cooling for fine tuning cooling.

  This can produce increased overshoot. Fine tuning starts when the setpoint is reached. If the setpoint cannot be reached, PID_Temp does not automatically abort tuning.

Procedure

To perform fine tuning, follow these steps:

1. Double-click the "PID_Temp > Commissioning" entry in the project tree.
2. Activate the "Monitor all" button or start the trend view.
   An online connection will be established.
3. Select the desired fine tuning entry from the "Tuning mode" drop-down list.
4. If required (see tuning offset), specify a tuning offset and wait until the stationary state is reached again.
5. Click the "Start" icon.
   - The process of fine tuning is started.
   - The "Status" field displays the current steps and any errors that may have occurred. The progress bar indicates the progress of the current step.

Note

Click the "Stop" icon in the "Tuning mode" group if the progress bar ("Progress" tag) has not changed for a long period and it is to be assumed that the tuning function is blocked. Check the configuration of the technology object and, if necessary, restart controller tuning.

In the following phases in particular, tuning is not aborted automatically if the setpoint cannot be reached.

- "Attempting to reach setpoint for heating with two-point control."
- "Attempting to reach setpoint for cooling with two-point control."

Result

If fine tuning was performed without errors, the PID parameters have been tuned. PID_Temp switches to automatic mode and uses the tuned parameters. The tuned PID parameters will be retained during power OFF and a restart of the CPU.

If errors occurred during fine tuning, PID_Temp responds with the configured response to errors.

"Manual" mode (S7-1200, S7-1500)

The following section describes how you can use "Manual mode" in the commissioning window of the "PID_Temp" technology object.

Manual mode is also possible when an error is pending.

Requirement

- The "PID_Temp" instruction is called in a cyclic interrupt OB.
- An online connection to the CPU has been established.
- The CPU is in "RUN" mode.

Procedure

If you want to test the controlled system by specifying a manual value, use "Manual mode" in the commissioning window.

To define a manual value, follow these steps:

1. Double-click the "PID_Temp > Commissioning" entry in the project tree.
2. Activate the "Monitor all" button or start the trend view.
   An online connection will be established.
3. Select the "Manual mode" check box in the "Online status of controller" area.
   PID_Temp operates in manual mode. The most recent current output value remains in effect.
4. Enter the manual value in the editable field as a % value.
   If cooling is activated in the basic settings, enter the manual value as follows:
   - Enter a positive manual value to output the value at the outputs for heating.
   - Enter a negative manual value to output the value at the outputs for cooling.
5. Click the icon.

Result

The manual value is written to the CPU and immediately goes into effect.

Clear the "Manual mode" check box if the output value is to be specified again by the PID controller.

The switchover to automatic mode is bumpless.

Substitute setpoint (S7-1200, S7-1500)

The following section describes how you can use the substitute setpoint in the commissioning window of the "PID_Temp" technology object.

Requirement

- The "PID_Temp" instruction is called in a cyclic interrupt OB.
- An online connection to the CPU has been established.
- The CPU is in "RUN" mode.

Procedure

If you want to use a different value as the setpoint than that specified at the "Setpoint" parameter (for example to tune a slave in a cascade), use the substitute setpoint in the commissioning window.

Proceed as follows to specify a substitute setpoint:

1. Double-click the "PID_Temp > Commissioning" entry in the project tree.
2. Activate the "Monitor all" button or start the trend view.
   An online connection will be established.
3. Select the "Subst.Setpoint" check box in the "Online status of controller" section.
   The substitute setpoint (SubstituteSetpoint tag) is initialized with the most recently updated setpoint and now used.
4. Enter the substitute setpoint in the editable field.
5. Click the icon.

Result

The substitute setpoint is written to the CPU and immediately goes into effect.

Clear the "Subst.Setpoint" check box if the value at the "Setpoint" parameter is to be used again as setpoint.

The switchover is not bumpless.

Cascade commissioning (S7-1200, S7-1500)

Information about cascade commissioning with PID_Temp is available under Commissioning (Page 6131).

13.1.5.4 Cascade control with PID_Temp (S7-1200, S7-1500)

Introduction (S7-1200, S7-1500)

In cascade control, several control loops are nested within each other. In the process, slaves receive their setpoint (Setpoint) from the output value (OutputHeat) of the respective higher-level master.

A prerequisite for establishing a cascade control system is that the controlled system can be divided into subsystems, each with its own measured variable.

Setpoint specification for the controlled variable is carried out at the outmost master.

The output value of the innermost slave is applied to the actuator and thus acts on the controlled system.

The following major advantages result from the use of a cascade control system in comparison with a single-loop control system:

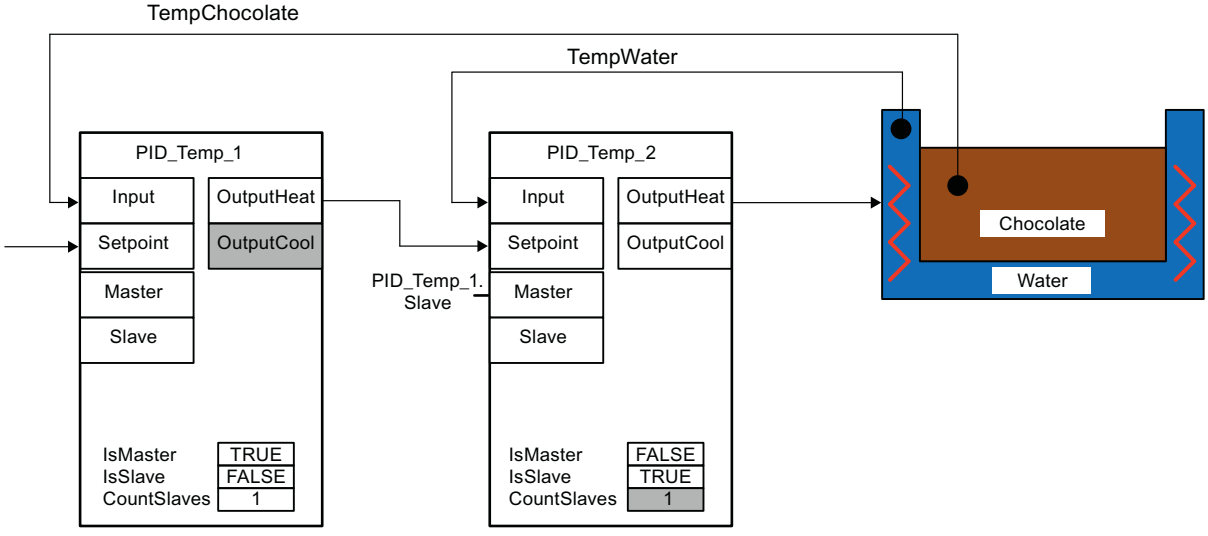
- Thanks to the additional subordinate control loops, disturbances which occur there are corrected quickly. Their influence on the controlled variable is reduced considerably. The disturbance behavior is thus improved.
- The subordinate control loops act in linearizing form. The negative effects of such non-linearities on the controlled variable are thus moderated.

PID_Temp offers the following functionality especially for use in cascade control systems:

- Specification of a substitute setpoint
- Exchange of status information between master and slave (for example, current operating mode)
- Different Anti-Wind-Up modes (response of the master to limitation of its slave)

Example

The following block diagram shows a cascade control system with PID_Temp using the simplified example of a chocolate melting unit:



The PID_Temp_1 master compares the process value of the chocolate temperature (TempChocolate) with the setpoint specification by the user at the Setpoint parameter. Its output value OutputHeat forms the setpoint of the slave PID_Temp_2.

PID_Temp_2 attempts to regulate the process value of the water-bath temperature (TempWater) to this setpoint. The output value of PID_Temp_2 acts directly on the actuator of the controlled system (heating of the water bath) and thus influences the water-bath temperature. The water-bath temperature in turn has an effect on the chocolate temperature.

FAQ

For more information, see the following FAQs in the Siemens Industry Online Support:

- Entry ID 103526819 (https://support.industry.siemens.com/cs/ww/en/view/103526819)

See also

Program creation (Page 6128)

Program creation (S7-1200, S7-1500)

Observe the following points during program creation:

- Number of PID_Temp instances
  The number of different PID_Temp instances called up in a cyclic interrupt OB has to agree with the number of concatenated measured variables in the process.
  There are two concatenated measured variables in the example: TempChocolate and TempWater. Therefore two PID_Temp instances are required.
- Call sequence
  A master has to be called before its slaves in the same cyclic interrupt OB.
  The outermost master at which the user setpoint is specified is called first.
  The slave whose setpoint is specified by the outermost master is called next, etc.
  The innermost slave that acts on the actuator of the process with its output value is called last.
  In the example, PID_Temp_1 is called before PID_Temp_2.
- Interconnection of the measured variables
  The outermost master is interconnected with the outermost measured variable that is to be regulated to the user setpoint.
  The innermost slave is interconnected with the innermost measured variable that is influenced directly by the actuator.
  Interconnection of the measured variables with PID_Temp is carried out with the parameters Input or Input_PER.
  In the example, the outermost measured variable TempChocolate is interconnected with PID_Temp_1 and the innermost measured variable TempWater with PID_Temp_2.

- Interconnection of the output value of the master to the setpoint of the slave
  The output value (OutputHeat) of a master has to be assigned to the setpoint (Setpoint) of its slave.
  This interconnection can be carried out in the programming editor or automatically in the Inspector window of the slave in the basic settings via the selection of the master.
  If required, you can insert your own filter or scaling functions, for example in order to adapt the output value range of the master to the setpoint/process value range of the slave.
  In the example, OutputHeat of PID_Temp_1 is assigned to Setpoint of PID_Temp_2.
- Interconnection of the interface for information exchange between master and slave
  The "Slave" parameter of a master has to be assigned to the "Master" parameter of all its directly subordinate slaves (which receive their setpoint from this master). The assignment should be carried out via the interface of the slave in order to allow the interconnection of a master with multiple slaves and the display of the interconnection in the Inspector window of the slave in the basic settings.
  This interconnection can be carried out in the programming editor or automatically in the Inspector window of the slave in the basic settings via the selection of the master.
  The Anti-Wind-Up functionality and the evaluation of the slave operating modes at the master can only function correctly if this interconnection is carried out.
  In the example, the "Slave" parameter of PID_Temp_1 is assigned to the "Master" parameter of PID_Temp_2.

Program code of the example using SCL (without assignment of the output value of the slave to the actuator):

```
"PID_Temp_1"(Input:="TempChocolate");

"PID_Temp_2"(Input:="TempWater", Master := "PID_Temp_1".Slave,
Setpoint := "PID_Temp_1".OutputHeat);
```

See also

PID_Temp ActivateRecoverMode tag (Page 4375)

Configuration (S7-1200, S7-1500)

You can carry out the configuration via your user program, the configuration editor or the Inspector window of the PID_Temp call.

When using PID_Temp in a cascade control system, ensure the correct configuration of the settings specified below.

If a PID_Temp instance receives its setpoint from a superior master controller and outputs its output value in turn to a subordinate slave controller, this PID_Temp instance is both a master controller and a slave controller simultaneously. Both configurations listed below have to be carried out for such a PID_Temp instance. This is the case, for example, for the middle PID_Temp instance in a cascade control system with three concatenated measured variables and three PID_Temp instances.

Configuration of a master

Setting in the configuration editor or Inspector window	DB parameter	Explanation
Basic settings → Cascade: Activate "Controller is master" check box	Config.Cascade.IsMaster = TRUE	Activates this controller as a master in a cascade
Basic settings → Cascade: Number of slaves	Config.Cascade.CountSlaves	Number of directly subordinate slaves that receive their setpoint directly from this master
Basic settings → Input/output parameters: Selection of the output value (heating) = OutputHeat	Config.Output.Heat.Select = 0	The master only uses the output parameter OutputHeat. OutputHeat_PWM and OutputHeat_PER are deactivated.
Basic settings → Input/output parameters: Clear "Activate cooling" check box	Config.ActivateCooling = FALSE	The cooling has to be deactivated at a master.
Output settings → Output limits and scaling → OutputHeat / OutputCool: PID output value low limit (heating), PID output value high limit (heating), Scaled low output value (heating), Scaled high output value (heating)	Config.Output.Heat.PidLowerLimit, Config.Output.Heat.PidUpperLimit, Config.Output.Heat.LowerScaling, Config.Output.Heat.UpperScaling	If no own scaling function is used when assigning OutputHeat of the master to Setpoint of the slave, it may be necessary to adapt the output value limits and the output scaling of the master to the setpoint/process value range of the slave.
This tag is not available in the Inspector window or in the functional view of the configuration editor. You can change it via the parameter view of the configuration editor.	Config.Cascade.AntiWindUpMode	The Anti-Wind-Up mode determines how the integral action of this master is treated if directly subordinate slaves reach their output value limits. Options are: • AntiWindUpMode = 0: The AntiWindUp functionality is deactivated. The master does not react to the limitation of its slaves. • AntiWindUpMode = 1 (default): The integral action of the master is reduced in the relationship "Slaves in limitation/ Number of slaves". This reduces the effects of the limitation on the control behavior. • AntiWindUpMode = 2: The integral action of the master is held as soon as a slave is in limitation.

Configuration of a slave

Setting in the configuration editor or Inspector window	DB parameter	Explanation
Basic settings → Cascade: Select the "Controller is slave" check box	Config.Cascade.IsSlave = TRUE	Activates this controller as a slave in a cascade

Commissioning (S7-1200, S7-1500)

After compiling and loading of the program, you can start commissioning of the cascade control system.

Begin with the innermost slave at commissioning (implementation of tuning or change to automatic mode with existing PID parameters) and continue outwards until the outermost master has been reached.

In the above example, commissioning starts with PID_Temp_2 and is continued with PID_Temp_1.

Tuning the slave

Tuning of PID_Temp requires a constant setpoint. Therefore, activate the substitute setpoint of a slave (SubstituteSetpoint and SubstituteSetpointOn tags) to tune the slave or set the associated master to manual mode with a corresponding manual value. This ensures that the setpoint of the slave remains constant during tuning.

Tuning the master

In order for a master to influence the process or to carry out tuning, all the downstream slaves have to be in automatic mode and their substitute setpoint has to be deactivated. A master evaluates these conditions through the interface for information exchange between master and slave (Master parameter and Slave parameter) and displays the current state at the AllSlaveAutomaticState and NoSlaveSubstituteSetpoint tags. Corresponding status messages are output in the commissioning editor.

Status message in the commissioning editor of the master	DB parameter of the master	Correction
One or more slaves are not in automatic mode.	AllSlaveAutomaticState = FALSE, NoSlaveSubstituteSetpoint = TRUE	First, carry out commissioning of all downstream slaves. Ensure that the following conditions are fulfilled before carrying out tuning or activating manual mode or automatic mode of the master: • All downstream slaves are in automatic mode (state = 3). • All downstream slaves have deactivated the substitute setpoint (SubstituteSetpointOn = FALSE).
One or more slaves have activated the substitute setpoint.	AllSlaveAutomaticState = TRUE, NoSlaveSubstituteSetpoint = FALSE	
One or more slaves are not in automatic mode and have activated the substitute setpoint.	AllSlaveAutomaticState = FALSE, NoSlaveSubstituteSetpoint = FALSE	

If pretuning or fine tuning is started for a master, PID_Temp aborts tuning in the following cases and displays an error with ErrorBits = DW#16#0200000:

- One or more slaves are not in automatic mode (AllSlaveAutomaticState = FALSE)
- One or more slaves have activated the substitute setpoint (NoSlaveSubstituteSetpoint = FALSE).

The subsequent operating mode changeover depends on ActivateRecoverMode.

Substitute setpoint (S7-1200, S7-1500)

In order to specify a setpoint, PID_Temp offers a substitute setpoint at the SubstituteSetpoint tag in addition to the Setpoint parameter. This can be activated by setting SubstituteSetpointOn = TRUE or by selecting the corresponding check box in the commissioning editor.

The substitute setpoint allows you to specify the setpoint temporarily directly at the slave, for example during commissioning or tuning.

In this case, the interconnection of the output value of the master with the setpoint of the slave that is required for normal operation of the cascade control system does not have to be changed in the program

In order for a master to influence the process or to carry out tuning, the substitute setpoint has to be deactivated at all downstream slaves.

You can monitor the currently effective setpoint as it is used by the PID algorithm for calculation at the CurrentSetpoint tag.

Operating modes and fault response (S7-1200, S7-1500)

The master or slave of a PID_Temp instance does not change the operating mode of this PID_Temp instance.

If a fault occurs at one of its slaves, the master remains in its current operating mode.

If a fault occurs at its master, the slave remains in its current operating mode. However, further operation of the slave then depends on the fault and the configured fault response of the master since the output value of the master is used as the setpoint of the slave:

- If ActivateRecoverMode = TRUE is configured at the master. and the fault does not prevent the calculation of OutputHeat, the fault does not have any effect on the slave.
- If ActivateRecoverMode = TRUE is configured at the master and the fault prevents the calculation of OutputHeat, the master outputs the last output value or the configured substitute output value SubstituteOutput, depending on SetSubstituteOutput. This is then used by the slave as the setpoint.
  PID_Temp is preconfigured so that the substitute output value 0.0 is output in this case (ActivateRecoverMode = TRUE, SetSubstituteOutput = TRUE, SubstituteOutput = 0.0). Configure a suitable substitute output value for your application or activate the use of the last valid PID output value (SetSubstituteOutput = FALSE).
- If ActivateRecoverMode = FALSE is configured at the master, the master changes to the "Inactive" mode when a fault occurs and outputs OutputHeat = 0.0. The slave then uses 0.0 as the setpoint.

The fault response is located in the output settings in the configuration editor.

13.1.5.5 Multi-zone controlling with PID_Temp (S7-1200, S7-1500)

Introduction

In a multi-zone control system, several sections, so-called zones, of a plant are controlled simultaneously to different temperatures. A multi-zone control system is characterized by the mutual influence of the temperature zones through thermal coupling, i.e. the process value of one zone can influence the process value of a different zone through thermal coupling. The strength that this influence has depends on the structure of the plant and the selected operating points of the zones.

Example: Extrusion plant as it is used, for example, in plastics processing.

The substance mixture that passes through the extruder has to be controlled to different temperatures for optimal processing. For example, different temperatures can be required at the filling point of the extruder than at the outlet nozzle. The individual temperature zones mutually influence each other through thermal coupling.

When PID_Temp is used in multi-zone control systems, each temperature zone is controlled by a separate PID_Temp instance.

Observe the following explanations if you want to use the PID_Temp in a multi-zone control system.

Separate pretuning for heating and cooling

Initial commissioning of a plant as a rule begins with the carrying out of pretuning in order to carry out initial setting of the PID parameters and control to the operating point. The pretuning for multi-zone control systems is often carried out simultaneously for all zones.

PID_Temp offers the possibility of carrying out pretuning for heating and cooling in one step (Mode = 1, Heat.EnableTuning = TRUE, Cool.EnableTuning = TRUE) for controllers with activated cooling and PID parameter switching as the method for heating/cooling (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE).

However, it is advisable not to use this tuning for simultaneous pretuning of several PID_Temp instances in a multi-zone control system. Instead, first carry out the pretuning for heating (Mode = 1, Heat.EnableTuning = TRUE, Cool.EnableTuning = FALSE) and the pretuning for cooling (Mode = 1, Heat.EnableTuning = FALSE, Cool.EnableTuning = TRUE) separately.

Pretuning for cooling should not be started until all zones have completed pretuning for heating and have reached their operating points.

This reduces mutual influencing through thermal coupling between the zones during tuning.

Adapting the delay time

If PID_Temp is used in a multi-zone control system with strong thermal couplings between the zones, you should ensure that the adaption of the delay time is deactivated for pretuning with PIDSelfTune.SUT.AdaptDelayTime = 0. Otherwise, the determination of the delay time can be incorrect if the cooling of a zone is prevented by the thermal influence of other zones during the adapting of the delay time (heating is deactivated in this phase).

Temporary deactivation of cooling

PID_Temp offers the possibility of deactivating cooling temporarily in automatic mode for controllers with active cooling (Config.ActivateCooling = TRUE) by setting DisableCooling = TRUE.

This ensures that this controller does not cool in automatic mode during commissioning while the controllers of other zones have not yet completed tuning of heating. The tuning could otherwise be influenced negatively by the thermal coupling between the zones.

Procedure

You can proceed as follows during the commissioning of multi-zone control systems with relevant thermal couplings:

1. Set DisableCooling = TRUE for all controllers with activated cooling.
2. Set PIDSelfTune.SUT.AdaptDelayTime = 0 for all controllers.
3. Specify the desired setpoints (Setpoint parameter) and start pretuning for heating (Mode = 1, Heat.EnableTuning = TRUE, Cool.EnableTuning = FALSE) simultaneously for all controllers.
4. Wait until all the controllers have completed pretuning for heating.
5. Set DisableCooling = FALSE for all controllers with activated cooling.
6. Wait until the process values of all the zones are steady and close to the respective setpoint. If the setpoint cannot be reached permanently for a zone, the heating or cooling actuator is too weak.
7. Start pretuning for cooling (Mode = 1, Heat.EnableTuning = FALSE, Cool.EnableTuning = TRUE) for all controllers with activated cooling.

Note

Limit violation of the process value

If the cooling is deactivated in automatic mode with DisableCooling = TRUE, this can cause the process value to exceed the setpoint and the process value limits while DisableCooling = TRUE. Observe the process values and intervene, if appropriate, if you use DisableCooling.

Note

Multi-zone control systems

For multi-zone control systems, the thermal couplings between the zones can result in increased overshoots, permanent or temporary violation of limits and permanent or temporary control deviations during commissioning or operation. Observe the process values and be ready to intervene. Depending on the system, it can be necessary to deviate from the procedure described above.

Synchronization of several fine tuning processes

If fine tuning is started from automatic mode with PIDSelfTune.TIR.RunIn = FALSE, PID_Temp tries to reach the setpoint with PID controlling and the current PID parameters. The actual tuning does not start until the setpoint is reached. The time required to reach the setpoint can be different for the individual zones of a multi-zone control system.

If you want to carry out fine tuning for several zones simultaneously, PID_Temp offers the possibility to synchronize these by waiting with the further tuning steps after the setpoint has been reached.

Procedure

This ensures that all the controllers have reached their setpoint when the actual tuning steps start. This reduces mutual influencing through thermal coupling between the zones during tuning.

Proceed as follows for controllers for whose zones you want to carry out fine tuning simultaneously:

1. Set PIDSelfTune.TIR.WaitForControlIn = TRUE for all controllers.
   These controllers have to be in automatic mode with PIDSelfTune.TIR.RunIn = FALSE.
2. Specify the desired setpoints (Setpoint parameters) and start fine tuning for all controllers.
3. Wait until PIDSelfTune.TIR.ControlInReady = TRUE at all controllers.
4. Set PIDSelfTune.TIR.FinishControlIn = TRUE for all controllers.

All controllers then start the actual tuning simultaneously.

13.1.5.6 Override control with PID_Temp (S7-1200, S7-1500)

Override control

In case of override control, two or more controllers share one actuator. Only one controller has access to the actuator at any time and influences the process.

A logic operation decides which controller has access to the actuator. This decision is often made based on a comparison of the output values of all controllers, for example, in case of a maximum selection, the controller with the largest output value gets access to the actuator.

The selection based on the output value requires that all controllers operate in automatic mode. The controllers that do not have an effect on the actuator are updated. This is necessary to prevent windup effects and their negative impacts on the control response and the switchover between the controllers.

PID_Temp supports override controls as of version 1.1 by offering a simple process for updating the controllers that are not active: By using the tags OverwriteInitialOutputValue and PIDCtrl.PIDInit, you can pre-assign the integral action of the controller in automatic mode as though the PID algorithm had calculated PidOutputSum = OverwriteInititalOutputValue for the PID output value in the last cycle. To do this, OverwriteInitialOutputValue is interconnected with the PID output value of the controller that currently has access to the actuator. By setting the bit PIDCtrl.PIDInit, you trigger the pre-assignment of the integral action as well as the restart of the controller cycle and the PWM period. The subsequent calculation of the PID output value in the current cycle takes place based on the pre-assigned (and synchronized for all controllers)

integral action as well as the proportional action and integral action from the current control deviation. The derivative action is not active during the call with PIDCtrl.PIDInit = TRUE and therefore does not contribute to the output value.

This procedure ensures that the calculation of the current PID output value and thus the decision on which controller is to have access to the actuator is only based on the current process state and the PI parameters. Windup effects for controllers that are not active and thus incorrect decisions of the switchover logic are prevented.

Requirement

- PIDCtrl.PIDInit is only effective if the integral action is activated (tags Retain.CtrlParams.Heat.Ti and Retain.CtrlParams.Cool.Ti > 0.0).
- You must assign PIDCtrl.PIDInit and OverwriteInitialOutputValue in your user program yourself (see example below). PID_Temp does not automatically change these tags.
- PIDCtrl.PIDInit is only effective when PID_Temp is in automatic mode (parameter State = 3).
- If possible, select the sampling time of the PID algorithm (Retain.CtrlParams.Heat.Cycle and Retain.CtrlParams.Cool.Cycle tags) so that it is identical for all controllers, and call all controllers in the same cyclic interrupt OB. In this way, you ensure that the switchover does not take place within a controller cycle or a PWM period.

Note

Constant adaptation of the output value limits

Instead of the active updating of the controllers without access to the actuator described here, this is implemented alternatively by constant adaptation of the output value limits in other controller systems.

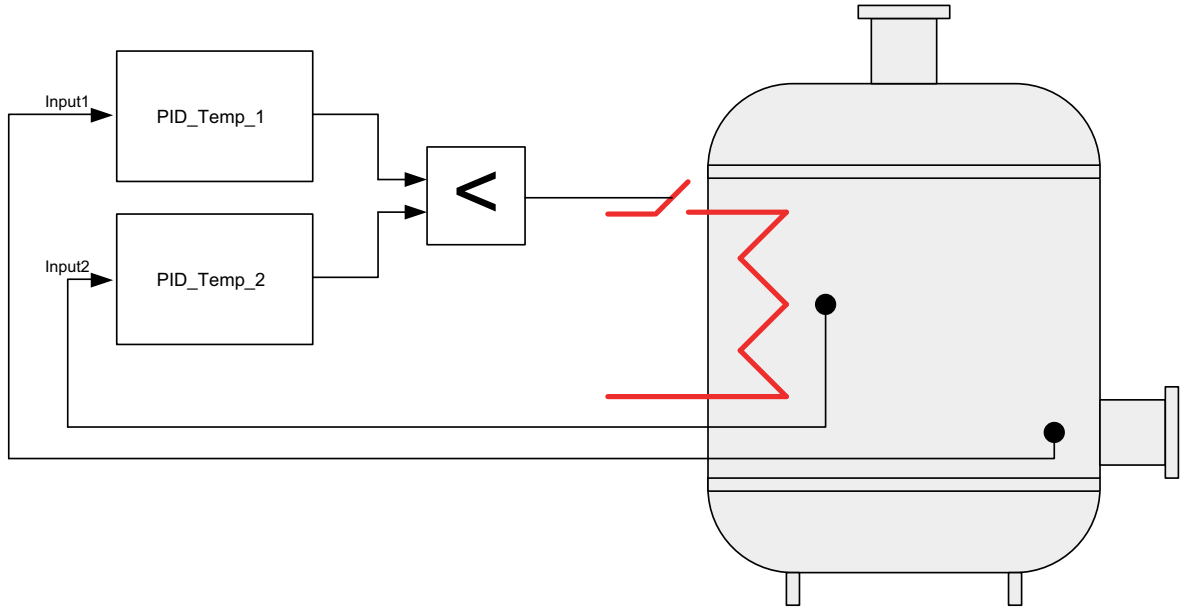
This is not possible with PID_Temp, because a change of the output value limits is not supported in automatic mode.

Example: Control of a large boiler

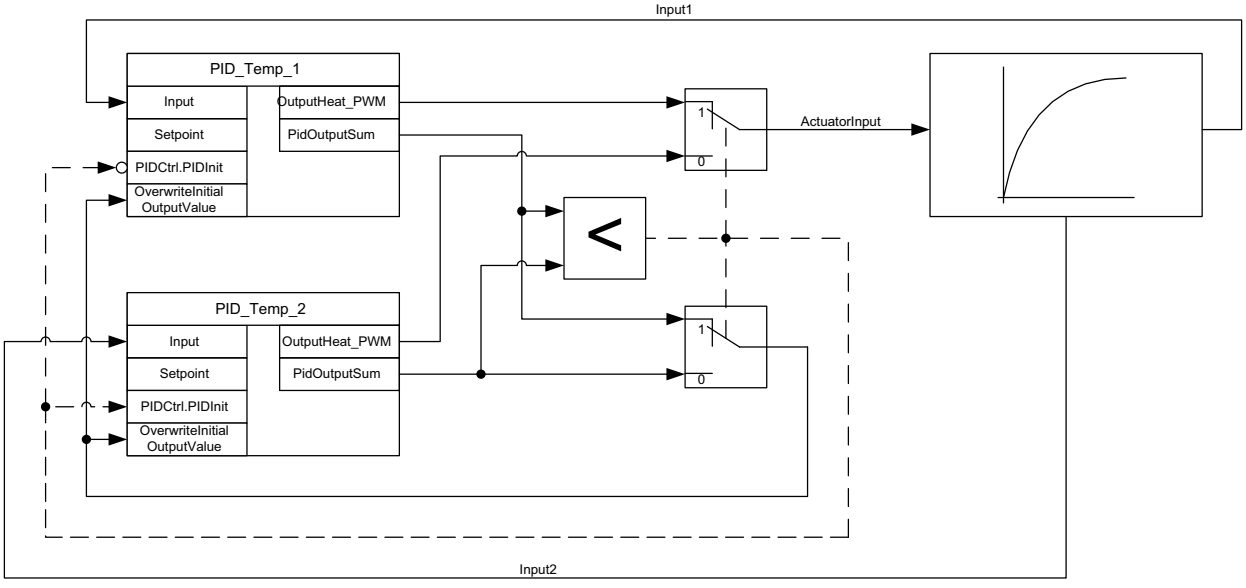
PID_Temp is used for control of a large boiler.

The main goal is to control the temperature Input1. The controller PID_Temp_1 is used for this purpose. In addition, the temperature Input2 is to be kept below a high limit at an additional measuring point with the limiting controller PID_Temp_2.

Both temperatures are influenced by only one heater. The output value of the controller corresponds to the heating power.



The heater is controlled with the pulse-width modulated output value of PID_Temp (parameter OutputHeat_PWM) by writing the program tag ActuatorInput. The setpoint for the temperature Input1 is specified at the parameter PID_Temp_1.Setpoint. The temperature high limit for the additional measuring point is specified as setpoint at the parameter PID_Temp_2.Setpoint.



Both controllers must share one heater as shared actuator. The logic that decides which controller gets access to the actuator is implemented by a minimum selection of the PID output value (in Real format, parameter PidOutputSum) in this case. Because the PID output value corresponds to the heating power, the controller that requires lower heating power gets the control.

In normal operation of the plant, the process value of the main controlled variable corresponds to the setpoint. The main controller PID_Temp_1 has settled on a stationary PID output value

PID_Temp_1.PidOutputSum. The process value of the limiting controller Input2 in normal operation is significantly below the high limit that is specified as setpoint for für PID_Temp_2. The limiting controller therefore wants to increase the heating power to increase its process value, which means it will calculate a PID output value PID_Temp_2.PidOutputSum that is greater than the main controller PID_Temp_1.PidOutputSum. The minimum selection of the switchover logic therefore gives the main controller PID_Temp_1 continued access to the actuator. In addition, it is ensured that PID_Temp_2 is updated by means of the assignments PID_Temp_2.OverwriteInitialOutputValue = PID_Temp_1.PidOutputSum and PID_Temp_2.PIDCtrl.PIDInit = TRUE.

If Input2 now approaches the high limit or exceeds it, for example due to a fault, the limiting controller PID_Temp_2 calculates a smaller PID output value to restrict the heating power and thus reduce Input2. If PID_Temp_2.PidOutputSum is smaller than PID_Temp_1.PidOutputSum, the limiting controller PID_Temp_2 receives access to the actuator through the minimum selection and reduces the heating power. It is ensured that PID_Temp_1 is updated by means of the assignments PID_Temp_1.OverwriteInitialOutputValue = PID_Temp_2.PidOutputSum and PID_Temp_1.PIDCtrl.PIDInit = TRUE.

The temperature at the additional measuring point Input2 drops. The temperature of the main controlled variable Input1 drops as well and cannot be held at the setpoint any longer.

Once the fault has been remedied, the Input2 will continue to drop and the heating power is further increased by the limiting controller. As soon as the main controller has calculated a lower heating power as output value, the plant returns to normal operation so that the main controller PID_Temp_1 once again has access to the actuator. This example can be implemented with the following SCL program code:

```
"PID_Temp_1"(Input := "Input1");
"PID_Temp_2"(Input := "Input2");
IF "PID_Temp_1".PidOutputSum <= "PID_Temp_2".PidOutputSum THEN
  "ActuatorInput" := "PID_Temp_1".OutputHeat_PWM;
  "PID_Temp_1".PIDCtrl.PIDInit := FALSE;
  "PID_Temp_2".PIDCtrl.PIDInit := TRUE;
  "PID_Temp_2".OverwriteInitialOutputValue := "PID_Temp_1".PidOutputSum;
ELSE
  "ActuatorInput" := "PID_Temp_2".OutputHeat_PWM;
  "PID_Temp_1".PIDCtrl.PIDInit := TRUE;
  "PID_Temp_2".PIDCtrl.PIDInit := FALSE;
  "PID_Temp_1".OverwriteInitialOutputValue := "PID_Temp_2".PidOutputSum;
END_IF;
```

13.1.5.7 Simulating PID_Temp with PLCSIM (S7-1200, S7-1500)

Note

Simulation with PLCSIM

The simulation of PID_Temp with PLCSIM for CPU S7-1200 is not supported.

PID_TEMP can be simulated only for CPU S7-1500 with PLCSIM.

For the simulation with PLCSIM, the time behavior of the simulated PLC is not exactly identical to that of a "real" PLC. The actual cycle clock of a cyclic interrupt OB can have larger fluctuations with a simulated PLC than with "real" PLCs.

In the standard configuration, PID_Temp determines the time between calls automatically and monitors them for fluctuations.

For the simulation of PID_Temp with PLCSIM, for example, a sampling time error (ErrorBits = DW#16#00000800) can therefore be detected.

This results in ongoing tuning being aborted.

The response in automatic mode depends on the value of the ActivateRecoverMode tag.

To prevent this from happening, you should configure PID_Temp for simulation with PLCSIM as follows:

- CycleTime.EnEstimation = FALSE
- CycleTime.EnMonitoring = FALSE
- CycleTime.Value: Assign the cycle clock of the calling cyclic interrupt OB in seconds to this tag.

13.2 Using S7-1200 Motion Control (S7-1200)

13.2.1 Introduction (S7-1200)

13.2.1.1 Motion functionality of the CPU S7-1200 (S7-1200)

The TIA Portal, together with the motion control functionality of the CPU S7-1200, supports you in controlling stepper motors and servo motors:

- You configure the positioning axis and command table technology objects in the TIA Portal. The CPU S7-1200 uses these technology objects to control the outputs that control the drives.
- In the user program you control the axis by means of Motion Control instructions and initiate motion commands of your drive.

See also

Hardware components for motion control (Page 6141)

Integration of the positioning axis technology object (Page 6177)

Tools of the positioning axis technology object (Page 6180)

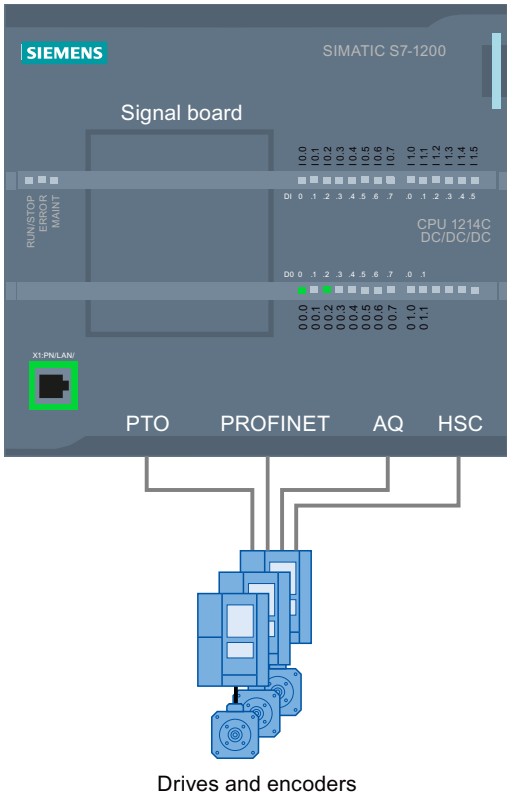
Use of the command table technology object (Page 6241)

Command table technology object tools (Page 6241)

http://www.automation.siemens.com/mcms/topics/en/simatic/simatic-technology/integrated-functions/simatic-s7-1200/Pages/Default.aspx (http://www.automation.siemens.com/mcms/topics/en/simatic/simatic-technology/integrated-functions/simatic-s7-1200/Pages/Default.aspx)

13.2.1.2 Hardware components for motion control (S7-1200)

The representation below shows the basic hardware configuration for a motion control application with the CPU S7-1200.



CPU S7-1200

CPU S7-1200 combines the functionality of a programmable logic controller with motion control functionality for operation of drives. The motion control functionality takes over the control and monitoring of the drives.

Signal board

You add further inputs and outputs to the CPU with the signal boards.

You can use the digital outputs as pulse generator outputs for controlling drives as required. In CPUs with relay outputs, the pulse signal cannot be output on the onboard outputs because the relays do not support the necessary switching frequencies. To be able to work with the PTO (Pulse Train Output) on these CPUs, you must use a signal board with digital outputs.

You can use the analog outputs for controlling connected analog drives as required.

PROFINET

Use the PROFINET interface to establish the online connection between the CPU S7-1200 and the programming device. In addition to the online functions of the CPU, additional commissioning and diagnostic functions are available for motion control.

PROFINET continues to support the PROFIdrive profile for connecting PROFIdrive capable drives and encoders.

Drives and encoders

Drives permit the movement of the axis. Encoders provide the actual position for the closed loop position control of the axis.

The table below shows the connection possibilities for drives and encoders:

Drive connection	Closed/open loop control of axis	Encoder connection
PTO (Pulse Train Output) (Stepper motors and servo motors with pulse interface)	Position-controlled	-
Analog output (AQ)	Position-controlled	• Encoder on high-speed counter (HSC) • Encoder on technology module (TM) • Encoder on PROFINET
PROFINET	Position-controlled	• Encoder on drive • Encoder on high-speed counter (HSC) • Encoder on technology module (TM) • Encoder on PROFINET

Ordering information for CPU firmware V4.1

The order information listed below applies to the currently installed product phase (without any installed Hardware Support Packages) of the TIA Portal.

Name	MLFB - article number
CPU 1211C DC/DC/DC	6ES7211-1AE40-0XB0
CPU 1211C AC/DC/RLY	6ES7211-1BE40-0XB0
CPU 1211C DC/DC/RLY	6ES7211-1HE40-0XB0
CPU 1212C DC/DC/DC	6ES7212-1AE40-0XB0
CPU 1212C AC/DC/RLY	6ES7212-1BE40-0XB0
CPU 1212C DC/DC/RLY	6ES7212-1HE40-0XB0
CPU 1214C DC/DC/DC	6ES7214-1AG40-0XB0
CPU 1214C AC/DC/RLY	6ES7214-1BG40-0XB0
CPU 1214C DC/DC/RLY	6ES7214-1HG40-0XB0
CPU 1214FC DC/DC/DC	6ES7214-1AF40-0XB0
CPU 1214FC DC/DC/RLY	6ES7214-1HF40-0XB0
CPU 1215C DC/DC/DC	6ES7215-1AG40-0XB0

Name	MLFB - article number
CPU 1215C AC/DC/RLY	6ES7215-1BG40-0XB0
CPU 1215C DC/DC/RLY	6ES7215-1HG40-0XB0
CPU 1215FC DC/DC/DC	6ES7215-1AF40-0XB0
CPU 1215FC DC/DC/RLY	6ES7215-1HF40-0XB0
CPU 1217C DC/DC/DC	6ES7217-1AG40-0XB0
Signal board DI4 x DC 24 V (200 kHz)	6ES7221-3BD30-0XB0
Signal board DI4 x DC 5 V (200 kHz)	6ES7 221-3AD30-0XB0
Signal board DQ4 x DC 24 V (200 kHz)	6ES7222-1BD30-0XB0
Signal board DQ4 x DC 5 V (200 kHz)	6ES7222-1AD30-0XB0
Signal board DI2/DQ2 x DC 24 V (20 kHz)	6ES7223-0BD30-0XB0
Signal board DI2/DQ2 x DC 24 V (200 kHz)	6ES7223-3BD30-0XB0
Signal board DI2/DQ2 x DC 5 V (200 kHz)	6ES7223-3AD30-0XB0
Signal board AQ1 x 12 bit (±10 V, 0 to 20 mA)	6ES7 232-4HA30-0XB0

Use a Hardware Support Package (HSP) to install new hardware components. The hardware component will then be available in the hardware catalog.

See also

Motion functionality of the CPU S7-1200 (Page 6140)

CPU outputs relevant for motion control (Page 6143)

13.2.2 Basics for working with S7-1200 Motion Control (S7-1200)

13.2.2.1 Stepper motor on the PTO (S7-1200)

CPU outputs relevant for motion control (S7-1200)

The number of usable drives depends on the number of PTOs (pulse train outputs) and the number of available pulse generator outputs.

The following tables provide information about the relevant dependencies:

Maximum number of PTOs

4 PTOs are available for each CPU with technology version V4. This means a maximum of 4 drives can be controlled.

Signal type of the PTO

Depending on the signal type of the PTO, 1-2 pulse generator outputs are required per PTO (drive):

Signal type	Number of pulse generator outputs
Pulse A and direction B (direction output disabled)[1]	1
Pulse A and direction B [1]	2
Clock up A and clock down B	2
A/B phase-shifted	2
A/B phase-shifted - quadruple	2

[1] The direction output must be on-board or on a signal board.

Usable pulse generator outputs and limit frequencies

The relay variants of the CPUs can only access the pulse generator outputs of a signal board.

Depending on the CPU and signal board, the following pulse signal generator outputs can be used with the following limit frequencies:

On-board	Q0.0	Q0.1	Q0.2	Q0.3	Q0.4	Q0.5	Q0.6	Q0.7	Q1.0	Q1.1
CPU 1211 (DC/DC/DC)	100 kHz	100 kHz	100 kHz	100 kHz	-	-	-	-	-	-
CPU 1212 (DC/DC/DC)	100 kHz	100 kHz	100 kHz	100 kHz	20 kHz	20 kHz	-	-	-	-
CPU 1214(F) (DC/DC/DC)	100 kHz	100 kHz	100 kHz	100 kHz	20 kHz	20 kHz	20 kHz	20 kHz	20 kHz	20 kHz
CPU 1215(F) (DC/DC/DC)	100 kHz	100 kHz	100 kHz	100 kHz	20 kHz	20 kHz	20 kHz	20 kHz	20 kHz	20 kHz
CPU 1217 (DC/DC/DC)	1 MHz	1 MHz	1 MHz	1 MHz	100 kHz	100 kHz	100 kHz	100 kHz	100 kHz	100 kHz
Signal board	**Qx.0**	**Qx.1**	**Qx.2**	**Qx.3**	-	-	-	-	-	-
Signal board DI2/DQ2 x DC24V 20kHz	20 kHz	20 kHz	-	-	-	-	-	-	-	-
Signal board DI2/DQ2 x DC24V 200kHz	200 kHz	200 kHz	-	-	-	-	-	-	-	-
Signal board DQ4 x DC24V 200kHz	200 kHz	200 kHz	200 kHz	200 kHz	-	-	-	-	-	-
Signal board DI2/DQ2 x DC5V 200kHz	200 kHz	200 kHz	-	-	-	-	-	-	-	-
Signal board DQ4 x DC5V 200kHz	200 kHz	200 kHz	200 kHz	200 kHz	-	-	-	-	-	-

The low limit frequency is always 1Hz.

The pulse generator outputs can be freely assigned to the PTOs.

Note

If pulse generator outputs with different limit frequencies are used in accordance with the signal type, the low limit frequency is used in each case.

Signal type "Pulse A and direction B" is an exception. With this type of signal, the limit frequency of the pulse generator output is always used.

Note

Access to pulse generator outputs via the process image

The firmware takes control via the corresponding pulse generator and direction outputs if the PTO (Pulse Train Output) is selected and assigned to an axis.

With this takeover of the control function, the connection between the process image and I/O output is also disconnected. Although the user has the option of writing the process image of pulse generator and direction outputs via the user program or watch table, this is not transferred to the I/O output. Accordingly, it is also not possible to monitor the I/O output via the user program or watch table. The information read reflects the value of the process image and does not match the real status of the I/O output.

For all other CPU outputs that are not used permanently by the CPU firmware, the status of the I/O output can be controlled or monitored via the process image, as usual.

Outputs for drive signals

For motion control, you can optionally parameterize a drive interface for "Drive enabled" and "Drive ready".

When using the drive interface the digital output for the drive enable and the digital input for "drive ready" can be freely selected.

Acceleration/deceleration limits

The following limits apply to acceleration and deceleration:

Acceleration/deceleration	Value
Minimum acceleration/deceleration	5.0E-3 pulses/s^2
Maximum acceleration/deceleration	9.5E+9 pulses/s^2

Jerk limits

The following limits apply to the jerk:

Jerk	Value
Minimum jerk	4.0E-3 pulses/s^3
Maximum jerk	1.0E+10 pulses/s^3

See also

CPU outputs relevant for motion control (technology version V1...3) (Page 6364)

How the pulse interface works (Page 6146)

Relationship between the signal type and the direction of travel (Page 6146)

Hardware and software limit switches (Page 6164)

Jerk limit (Page 6165)

Homing (Page 6166)

Hardware components for motion control (Page 6141)

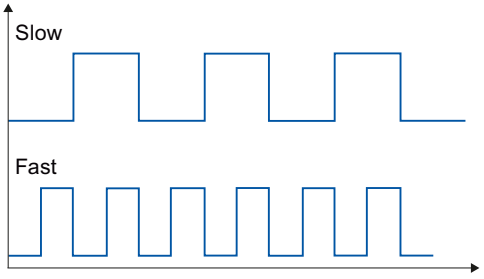
Integration of the positioning axis technology object (Page 6177)

Tools of the positioning axis technology object (Page 6180)

How the pulse interface works (S7-1200)

Depending on the settings of the stepper motor, each pulse affects the movement of the stepper motor by a specific angle. If the stepper motor is set to 1000 pulses per revolution, for example, it moves 0.36° per pulse.

The speed of the stepper motor is determined by the number of pulses per time unit.



(The statements made here also apply to servo motors with pulse interface.)

See also

CPU outputs relevant for motion control (Page 6143)

Relationship between the signal type and the direction of travel (Page 6146)

Hardware and software limit switches (Page 6164)

Jerk limit (Page 6165)

Homing (Page 6166)

Integration of the positioning axis technology object (Page 6177)

Tools of the positioning axis technology object (Page 6180)

Relationship between the signal type and the direction of travel (S7-1200)

The CPU outputs the velocity and direction of travel via two outputs.

The relationships between the configuration and direction of travel differ depending on the selected signal type. You can configure the following signal types in the axis configuration under "Basic parameters > General":

- "PTO (pulse A and direction B)"
- "PTO (clock up A and clock down B)" (as of V4)
- "PTO (A/B phase-shifted)" (as of V4)
- "PTO (A/B phase-shifted - quadruple)" (as of V4)

You configure the direction under "Extended Parameters > Mechanics" in the axis configuration. If you select the "Invert direction" option, the direction logic described below for the respective signal type is inverted.

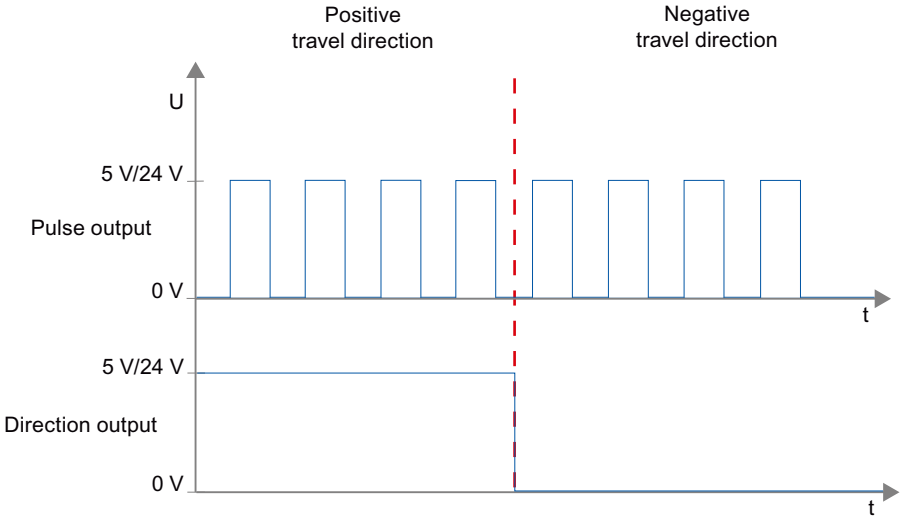
PTO (pulse A and direction B)

The pulse output pulses and the direction output level are evaluated for this signal type.

The pulses are output via the pulse output of the CPU. The direction output of the CPU specifies the direction of rotation of the drive:

- 5 V/24 V at direction output ⇒ positive direction of rotation
- 0 V at direction output ⇒ negative direction of rotation

The specified voltage depends on the hardware used. The indicated values do not apply to the differential outputs of CPU 1217.

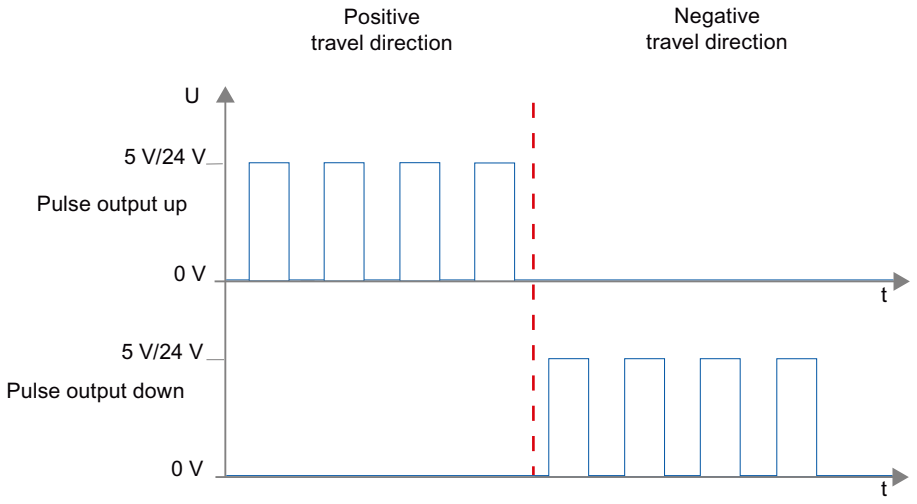


PTO (clock up A and clock down B) (as of V4)

The pulses of one output are evaluated for this signal type.

The pulse for the positive direction is output via the "Pulse output up" The pulse for the negative direction is output via the "Pulse output down"

The specified voltage depends on the hardware used. The indicated values do not apply to the differential outputs of CPU 1217.



PTO (A/B phase-shifted) (as of V4)

The positive edges of one output in each case are evaluated for this signal type.

The pulse is output via the "Signal A" output and phase-shifted via the "Signal B" output. The phase shifting between the outputs defines the direction of rotation:

- Signal A leads signal B by 90° ⇒ positive direction of rotation
- Signal B leads signal A by 90° ⇒ negative direction of rotation

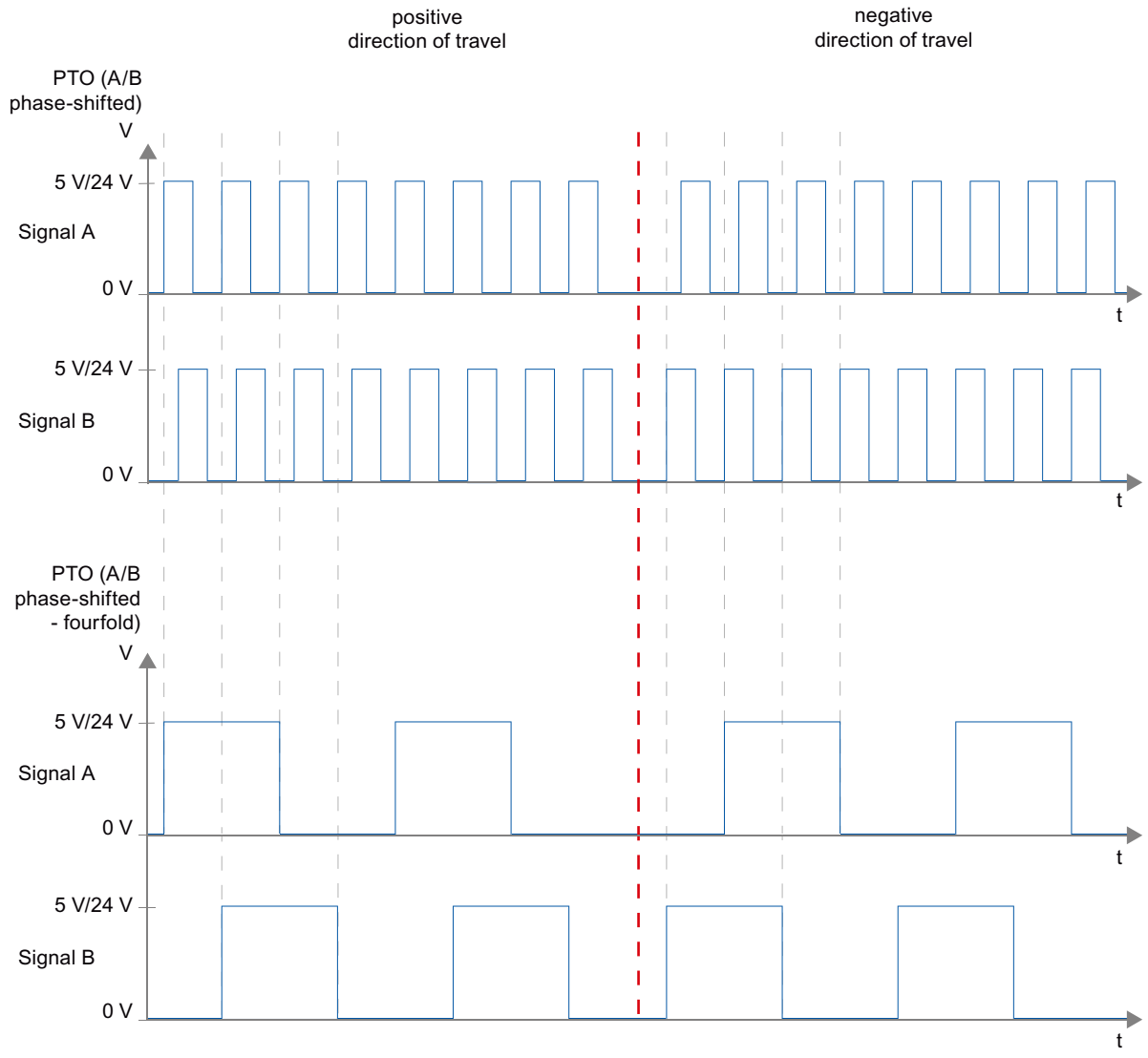
PTO (A/B phase-shifted - quadruple) (as of V4)

The positive and negative edges of both outputs are evaluated for this signal type. A pulse period has four edges with two phases (A and B). The pulse frequency at the output is therefore reduced to a quarter.

The pulse is output via the "Signal A" output and phase-shifted via the "Signal B" output. The phase shifting between the outputs defines the direction of rotation:

- Signal A leads signal B by 90° ⇒ positive direction of rotation
- Signal B leads signal A by 90° ⇒ negative direction of rotation

The specified voltage depends on the hardware used. The indicated values do not apply to the differential outputs of CPU 1217.



Invert direction

If you select the "Invert rotation signal" option, the direction logic is inverted:

- **PTO (pulse A and direction B)**
  - 0 V at direction output (low level) ⇒ positive direction of rotation
  - 5 V/24 V at direction output (high level) ⇒ negative direction of rotation

  The specified voltage depends on the hardware used. The voltages indicated do not apply to the differential outputs of CPU 1217.
- **PTO (clock up A and clock down B)**
  The outputs "Pulse output down" and "Pulse output up" are swapped.

- **PTO (A/B phase-shifted)**
  The "Signal A" and "Signal B" outputs are swapped.
- **"PTO (A/B phase-shifted - quadruple)**
  The "Signal A" and "Signal B" outputs are swapped.

See also

CPU outputs relevant for motion control (Page 6143)

How the pulse interface works (Page 6146)

Hardware and software limit switches (Page 6164)

Jerk limit (Page 6165)

Homing (Page 6166)

Integration of the positioning axis technology object (Page 6177)

Tools of the positioning axis technology object (Page 6180)

13.2.2.2 PROFIdrive drive / analog drive connection (S7-1200)

Drive and encoder connection (S7-1200)

A drive and an encoder are assigned to a positioning axis with drive connection via PROFIdrive/ analog drive connection.

Drives with PROFIdrive capability are connected by means of PROFIdrive telegrams. The setpoint is specified via PROFIdrive telegrams.

Drives with analog setpoint interfaces are connected using an analog output and an optional enable signal. The setpoint is specified via an analog output.

Options for connecting

Drives with PROFIdrive capability are connected via the PROFINET interface of the CPU.

Drives with analog setpoint interface are connected with the CPU via one of the following connections:

- Analog output via signal board
- Analog output on-board
- Analog output via analog output module

The following connection options are available for an encoder:

- Encoder on the PROFIdrive drive
- Encoder on technology module

- PROFIdrive encoder directly on PROFINET IO
  (With these encoders, the encoder value is always transferred via PROFIdrive telegrams per PROFIBUS or PROFINET.)
- Encoder on high speed counter (HSC)
  With this connection option, the encoder signals are connected directly to an HSC, which forms the encoder values from them. Depending on the CPU in use, up to 6 HSC encoders can be used.

Maximum number of axes

You can control up to a maximum of 8 drives via PROFIdrive or the analog drive connection (the number is independent of the simulation status of the axis).

Automatic transfer of drive and encoder parameters in the device (S7-1200)

For operation, identical reference values for the drive and encoder connections must be set in the controller and in the drive and encoder.

The speed setpoint NSET and the actual speed value NACT are transferred in the PROFIdrive telegram as a percentage of the reference speed. The reference value for the speed must be set identically in the controller and in the drive.

The resolution of the actual value in the PROFIdrive telegram must likewise be set identically in the controller and in the drive and encoder modules

Automatic transfer of parameters

For SINAMICS drives as of V4.x and PROFIdrive encoders as of product version A16, the drive and encoder parameters can be automatically transferred to the CPU.

The corresponding parameters are transferred after the (re)initialization of the technology object and (re)start of the drives and the CPU. Changes in the drive configuration are transferred after restart of the drive or technology object.

The successful transfer of the parameters can be verified in the controller in the variables of the <TO>.StatusDrive.AdaptionState = 1 and <TO>.StatusSensor[1].AdaptionState = 1 technology objects.

Parameters

The controller settings are made in the TIA Portal under "Technology object > Configuration > Basic parameters > Drive/encoder".

The drive and encoder settings are made in the configuration or the respective hardware.

The following table compares the settings in the TIA Portal and the corresponding drive/ encoder parameters:

Setting in the TIA Portal	Controller variable in the technology data block	Drive parameter	Automatic transfer
Drive			
Telegram number	Input address telegram <TO>.Actor.Interface.AddressIn	Telegram number P922	-
	Output address telegram <TO>.Actor.Interface.AddressOut		
Reference speed in [1/min]	<TO>.Actor.DriveParameter.Reference-Speed	(SINAMICS drives: P2000)	X
Maximum speed of motor in [1/min]	<TO>.Actor.DriveParameter.MaxSpeed	(SINAMICS drives: P1082)	X
Drive	<TO>.Actor.Type 0 = analog drive connection 1 = PROFIdrive 2 = PTO (pulse train output)	-	-
Encoder			
Telegram	<TO>.Sensor[n].Interface.AddressIn	P922	-
	<TO>.Sensor[n].Interface.Addressout		
Encoder type • Linear incremental • Linear absolute • Rotary incremental • Rotary absolute	<TO>.Sensor[n].System 0: Rotary 1: Linear	P979[1] Bit0 Encoder 1 P979[11] Bit0 Encoder 2	X
	<TO>.Sensor[n].Type 0: Incremental 1: Absolute	P979[5] Encoder 1 P979[15] Encoder 2	-
Resolution (linear encoder) The grid spacing is specified on the name-plate of the encoder as a separation dis-tance of the marks on the linear measuring system.	<TO>.Sensor[n].Parameter.Resolution	P979[2] Encoder 1 P979[12] Encoder 2	X
Steps per revolution (rotary encoder)	<TO>.Sensor[n].Parameter.StepsPerRe-volution	P979[2] Encoder 1 P979[12] Encoder 2	X
Number of bits for fine resolution XIST1 (cyclic actual encoder value, linear or ro-tary encoder)	<TO>.Sensor[n].Parameter.FineResolu-tionXist1	P979[3] Encoder 1 P979[13] Encoder 2	X
Number of bits for fine resolution XIST2 (absolute encoder value, linear or rotary encoder)	<TO>.Sensor[n].Parameter.FineResolu-tionXist2	P979[4] Encoder 1 P979[14] Encoder 2	X
Differentiable encoder revolutions (rotary absolute encoder)	<TO>.Sensor[n].Parameter.Determina-bleRevolutions	P979[5] Encoder 1 P979[15] Encoder 2	X

See also

Configuration - Drive - PTO (Pulse Train Output) (Page 6186)

Configuration - Drive - Analog drive connection (Page 6189)

Configuration - Drive - PROFIdrive (Page 6190)

Configuration - Encoder - Encoder on PROFINET/PROFIBUS (Page 6192)

Configuration - Encoder - Encoder on high-speed counter (HSC) (Page 6194)

PROFIdrive (S7-1200)

PROFIdrive is the standardized standard profile for drive technology in the connection of drives and encoders via PROFINET IO. Drives and encoders that support the PROFIdrive profile are connected according to the PROFIdrive standard.

Communication between controller and drive/encoder is by means of various PROFIdrive telegrams. Each of the telegrams has a standardized structure. Depending on the application, you can select the applicable telegram. Control words and status words as well as setpoints and actual values are transmitted in the PROFIdrive telegrams.

Telegrams for PROFIdrive

The setpoint of a positioning axis is transferred to a drive via PROFIdrive telegram 1, 2 3 or 4.

The encoder value is transmitted either in a telegram together with the setpoint (telegram 3 and 4), or in a separate encoder telegram (telegram 81 or telegram 83).

The following table shows the supported PROFIdrive telegrams for the assignment of drives and encoders:

Telegram	Brief description
Standard telegrams	
1	• 16 bit speed setpoint (NSET) • 16 bit actual speed (NACT)
2	• 32 bit speed setpoint (NSET) • 32-bit actual speed (NACT) • Signs of life
3	• 32 bit speed setpoint (NSET) • 32-bit actual speed (NACT) • 1 encoder • Signs of life
4	• 32 bit speed setpoint (NSET) • 32-bit actual speed (NACT) • 2 encoders • Signs of life
Standard telegrams encoder	
81	• 1 encoder • Signs of life
83	• 32-bit actual speed (NACT) • 1 encoder • Signs of life

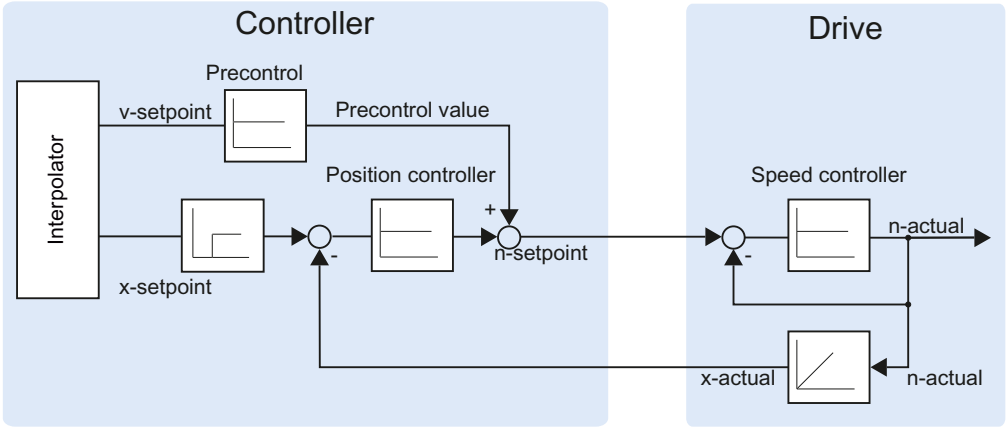
Closed loop control (S7-1200)

Drives which are connected via PROFIdrive or an analog drive interface are generally operated under position control. If service is required, the axis can also be operated without position control.

The position controller is a P controller with precontrol of velocity.

Controller structure

The following figure shows the controller structure of an S7-1200 Motion Control:



The MC-Interpolator [OB92] calculates the setpoint position for the axis. The difference between the setpoint and actual position is multiplied by the gain factor of the position controller. The resulting value is added to the precontrol value and output as setpoint speed to the drive via PROFIdrive or analog output.

The encoder records the actual position of the axis and returns it to the controller via a PROFIdrive telegram or an HSC (high speed counter).

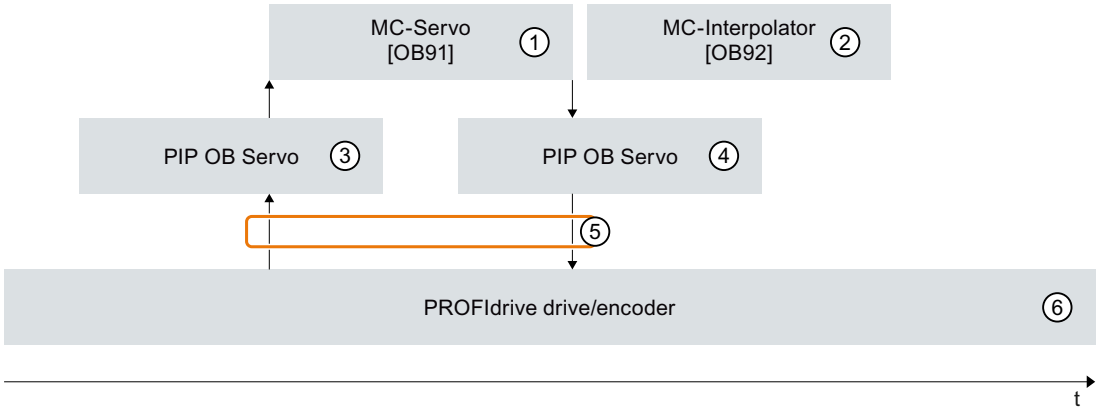
Data connection PROFIdrive drive / PROFIdrive encoder (S7-1200)

The data connection of PROFIdrive drives and PROFIdrive encoders occurs either directly via the PROFIdrive telegram or via a data block.

Use the connection via data block if you want to influence or evaluate telegram contents in the user program for process-specific reasons.

Principle of data connection directly to the drive/encoder

The following simplified function chart shows the direct data connection to PROFIdrive drives and PROFIdrive encoders by means of telegrams:



①	The organization block "MC-Servo" calculates the position controller. At the start of "MC-Servo", the input telegram of the drive or encoder is read (⑥ -> ⑤ -> ③). If an organization block "MC-PreServo" was added, the telegram is read at the beginning of "MC-PreServo". At the end of "MC-Servo", the output telegram is written to the drive or encoder (⑥ -> ⑤ -> ③). If an organization block "MC-PostServo" was added, the telegram is written at the end of "MC-PostServo".
②	In every Motion application cycle, the organization block "MC-Interpolator" is called after the "MC-Servo". In "MC-Interpolator", the Motion Control instructions are evaluated, setpoints are generated for the next Motion application cycle and the technology object is monitored.
③	The process image partition "PIP OB servo" of the inputs is updated in the Motion application cycle.
④	The process image partition "PIP OB servo" of the outputs is updated in the Motion application cycle.
⑤	Telegram exchange via the I/O addresses of the controller and the drive or the encoder.
⑥	PROFIdrive drive or PROFIdrive encoder

Principle of data connection via data block

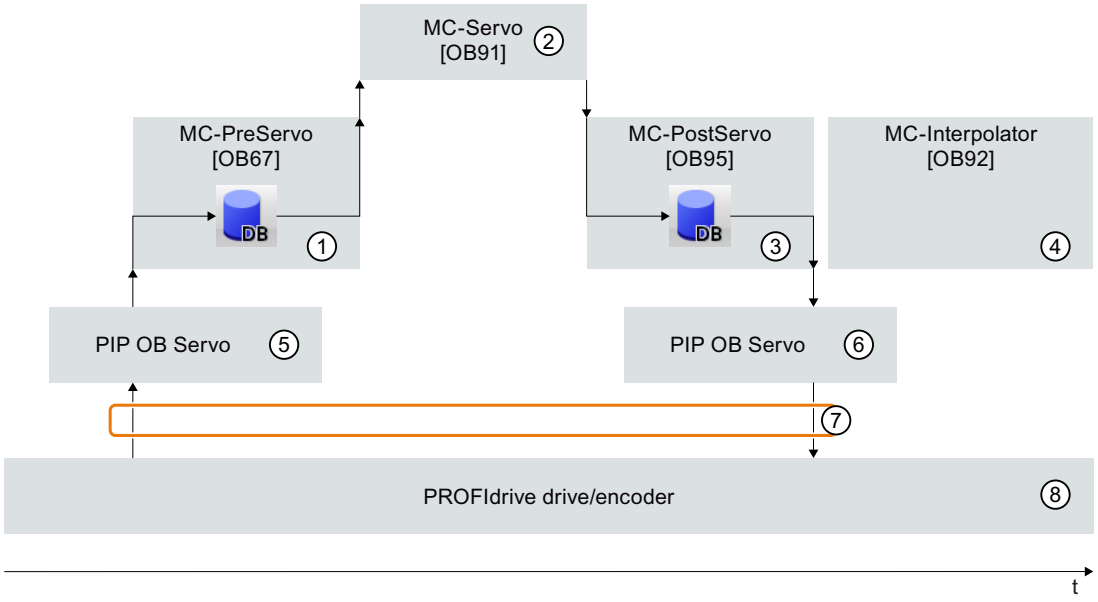
The following simplified function chart shows the data connection to PROFIdrive drives and PROFIdrive encoders via data block: The following sections provide details on the execution.

To influence or evaluate telegram contents for process-specific reasons, a data block must be connected in between as a data interface (see ① and ③).

To do this, use the organization blocks "MC-PreServo" and "MC-PostServo" to achieve a high quality of position control.

In this case, the telegram exchange via the I/O addresses of the controller and the drive or the encoder has to be programmed in the user program of the "MC-PreServo" and the "MC-PostServo".

The organization blocks "MC-PreServo" and "MC-PostServo" can be added in the project tree with the command "Add new block".



①	The "MC-PreServo" is called before the "MC-Servo". In the user program of the "MC-PreServo", transfer the content of the input telegram from the process image partition "PIP OB servo" ⑤ to the data block of the data interface. In the other user program of the "MC-PreServo", the input area of the telegram can be processed or evaluated.
②	The organization block "MC-Servo" calculates the position controller. At the start of "MC-Servo", the input telegram of the drive or encoder is read from the data block of the data interface (① -> ②). At the end of "MC-Servo", the output telegram of the drive or encoder is written to the data block of the data interface (② -> ③).
③	The "MC-PostServo" is called after the "MC-Servo". In the user program of the "MC-PostServo", the output area of the telegram can be processed or evaluated. At the end of the user program of the "MC-PostServo", transfer the content of the output telegram from the data interface of the data block to the process image partition "PIP OB servo" ⑥.
④	In every Motion application cycle, the organization block "MC-Interpolator" is called after the "MC-PostServo". In "MC-Interpolator", the Motion Control instructions are evaluated, setpoints are generated for the next Motion application cycle and the technology object is monitored.
⑤	The process image partition "PIP OB servo" of the inputs is updated in the Motion application cycle.
⑥	The process image partition "PIP OB servo" of the outputs is updated in the Motion application cycle.
⑦	Telegram exchange via the I/O addresses of the controller and the drive or the encoder.
⑧	PROFIdrive drive or PROFIdrive encoder

Basic procedure for the data connection via a data block

To use the data connection via the data block, follow the steps described below. The data connection can be configured separately for a PROFIdrive drive and encoder.

Creating the data block for data connection

The data block for the data connection must be created by the user. The data block must contain a data structure of data type "PD_TELx" for the data connection. Here, "x" stands for the telegram number of the drive or encoder configured in the device configuration.

To create the data block, follow the steps described below:

1. Create a new data block of type "Global DB".
2. Select the data block in the project tree and select the shortcut menu command "Properties".
3. Under Attributes, disable the following attributes and confirm with OK
   "Only store in load memory"
   "Data block write-protected in the device"
   "Optimized block access"
4. Open the data block in the block editor.
5. Insert a "PD_TELx" type variable in the block editor.

This variable contains the "Input" variable structure for the input area of the telegram and the "Output" variable structure for the output area of the telegram.

Note

"Input" and "Output" relate to the view of the closed loop position control. For example, the input area contains the actual values of the drive and the output area contains the setpoints for the drive.

The data block may contain the data structures of multiple axes and encoders and other contents.

Configuring data connection via a data block

Proceed as decribed below for the configuration of the axis:

1. Open the configuration window "Hardware interface > Drive" or "Hardware interface > Encoder".
2. In the Data block drop-down list, select "Data block".
3. In the "Data block" field, select the previously created data block.
   Open this data block and select the tag name defined for the drive or encoder.

Adding a PLC variable for telegram access

To enable access to the input and output areas of the telegram, create the following PLC variable.

For the PLC variable of the input area, follow the steps described below:

1. Open the "PLC variables" folder in the project tree and have all variables displayed.
2. Add a new variable and assign a unique name, made up for example from the name of the axis or the encoder, the telegram type and the address area.

3. Enter the type "PD_TELx_IN" textually in the "Data type" column.
4. Enter the telegram input address of the drive/encoder in the "Address" column.
   You can find the address in the device configuration of the drive or encoder.

Follow the same procedure for the PLC variable of the output area and select "PD_TELx_OUT" as the data type and the telegram output address of the drive/encoder as the address.

Programming MC-PreServo and MC-PostServo

MC-PreServo

The user program of the "MC-PreServo" must read the input area of the telegram and transfer it to the data block of the data connection.

Assign the previously defined PLC tag of the input area to the tag structure "Input" of the data block in the "MC-PreServo" user program.

With further instructions, you can edit the data of the tag structure "Input" of the data block before it is then transferred to "MC-Servo" and processed in MC_Servo" .

MC-PostServo

After it has been processed, "MC-Servo" transfers the output area of the telegram to the tag structure "Output" of the data block.

The content of the tag structure "Output" of the data block must be written to the telegram output address in the "MC-PostServo" user program.

Assign the "Output" tag structure of the data block to the previously defined PLC tag of the output area in the "MC-PostServo" user program.

If the output area is to be modified, this must be done before the assignment instruction.

NOTICE
Machine damage
Improper manipulation of drive and encoder telegrams may result in unwanted movements of the drive.
Check your user program in regard to consistency in the drive and encoder connection.

See also

PROFIdrive frame (Page 6289)

Configuration - Drive - PROFIdrive (Page 6190)

Configuration - Encoder - Encoder on PROFINET/PROFIBUS (Page 6192)

Data connection drive with analog drive connection (Page 6159)

Organization Blocks for Motion Control (Page 6160)

Data connection drive with analog drive connection (S7-1200)

The data connection of drives with analog drive interface can alternatively be made via a data block.

Use the connection via a data block if you want to adapt the analog setpoint in the user program for process-related reasons.

Principle of data connection via data block

At the end of position control by MC-Servo [OB91], the setpoint of the analog drive is written to the assigned analog output.

To adapt the analog setpoint for process-related reasons, a data interface via a data block must be connected in between.

The setpoint of the analog drive can be edited via the MC-PostServo [OB95] organization block in the data block and can then be written to the I/O address.

The MC-PostServo is called after the MC-Servo. The organization block MC-PostServo can be programmed by the user and must be added with the command "Add new block".

The procedure in principle

To use the data connection via the data block, follow the steps described below. The data connection can be configured separately for drives with analog drive interface and PROFIdrive encoders. You can find information on data connection of the PROFIdrive encoders in the section Data connection PROFIdrive drive / PROFIdrive encoder (Page 6154).

Creating the data block for data connection

The data block must be created on the user side.

To create the data block, follow the steps described below:

1. Create a new data block of type "Global DB".
2. Select the data block in the project tree and select "Properties" from the shortcut menu.
3. Under Attributes, disable the following attributes and confirm with OK
   "Only store in load memory"
   "Data block write-protected in the device"
   "Optimized block access"
4. Open the data block in the block editor.
5. Insert a variable of the "WORD" data type in the block editor.

Configuring data connection via a data block

Proceed in the configuration as described below ("Analog drive connection" must be selected in the Basic parameters > General configuration window):

1. Open the configuration window Basic parameters > Drive.
2. Select the previously defined variable of the data block in the "Analog output" box.

Set analog output address in the process image TPA OB Servo

To achieve a good level of control quality, the address area of the analog output must be within the "TPA OB Servo" process image.

Proceed as described below:

1. Open the module of the analog output in the device configuration.
2. Open the "General" tab
3. Select the "I/O addresses"
4. Select the organization block "MC-Servo". "TPA OB Servo" is automatically selected as the process image.
5. Assign a variable name to the analog output in the "I/O variables" tab.

Program MC-PostServo

Assign the variable of the data block to the variable of the analog output in the MC-PostServouser program.

At the end of MC-PostServo , the output area of "TPA OB Servo" is written to the I/O.

⚠ WARNING
Improper manipulation of the drive setpoint may endanger humans and machines.
Take adequate precautions to prevent danger to humans and machines.

Process response (S7-1200)

Organization Blocks for Motion Control (S7-1200)

Description

When you create a "Positioning axis" technology object with a PROFIdrive drive or with an analog drive interface, organization blocks for processing the technology objects are created automatically. The Motion Control functionality of the technology objects creates its own execution level, and is called according to the Motion Control application cycle.

The following blocks are created:

- **MC-Servo [OB91]**
  Calculation of the Position Controller
- **MC-Interpolator [OB92]**
  Evaluation of the motion control instructions, generation of setpoints and monitoring functionality

The organization blocks are protected (know-how protection). The program code cannot be viewed or changed.

The frequency relationship of the two organization blocks to one another is always 1:1. MC-Servo [OB91] is always executed before MC-Interpolator [OB92].

You can set the Motion Control application cycle and the priority of the organization blocks according to your requirements for control quality and system load.

Motion Control application cycle

You can set the Motion Control application cycle, in which the MC-Servo [OB91] is called, in the properties of the organization block in "General > Cycle Time".

The MC-Servo [OB91] is called cyclically with the specified "application cycle".

The selected Motion Control application cycle must be long enough to be able to process all technology objects in one cycle. If the processing time of the technology objects is longer than the application cycle, overflows (Page 6162) will occur.

To avoid disruptions in the program execution on the CPU, set the Motion Control application cycle depending on the number of axes used as follows:

Motion Control application cycle = 2 ms + (number of position-controlled axes × 2 ms)

The following table shows the resulting Motion Control application cycle as an example according to the number of position-controlled axes:

Number of axes	Motion Control application cycle
1	4 ms
2	6 ms
4	8 ms
8	10 ms

For SINAMICS, the following should continue to apply:

- Motion Control application cycle (MC-Servo) ≥ SINAMICS drive process image (parameter P2048) ≥ bus clock cycle

All times should be selected as integral multiples of one another.

Priority

You can configure the priority of the organization blocks as needed in their properties under "General > Properties > Priority":

- **MC-Servo [OB91]**
  Priority 17 to 26 (default value 25)
- **MC-Interpolator [OB92]**
  Priority 17 to 26 (default value 24)

The priority of MC-Servo [OB91] must be at least one higher than the priority of the MC-Interpolator [OB92].

MC-PreServo [OB67] and MC-PostServo [OB95]

Organization blocks MC-PreServo [OB67] and MC-PostServo [OB95] are programmable and are called in the configured application cycle. MC-PreServo [OB67] is called directly before MC-Servo [OB91]. MC-PostServo [OB95] is called directly after MC-Servo [OB91].

See also

Data connection PROFIdrive drive / PROFIdrive encoder (Page 6154)

Process image partition "PIP OB Servo" (S7-1200)

For optimal control, assign all I/O modules used by Motion Control (e.g. for technology modules, hardware limit switches) to the process image partition "PIP OB servo". The assignment causes the I/O modules to be processed simultaneously with the technology object.

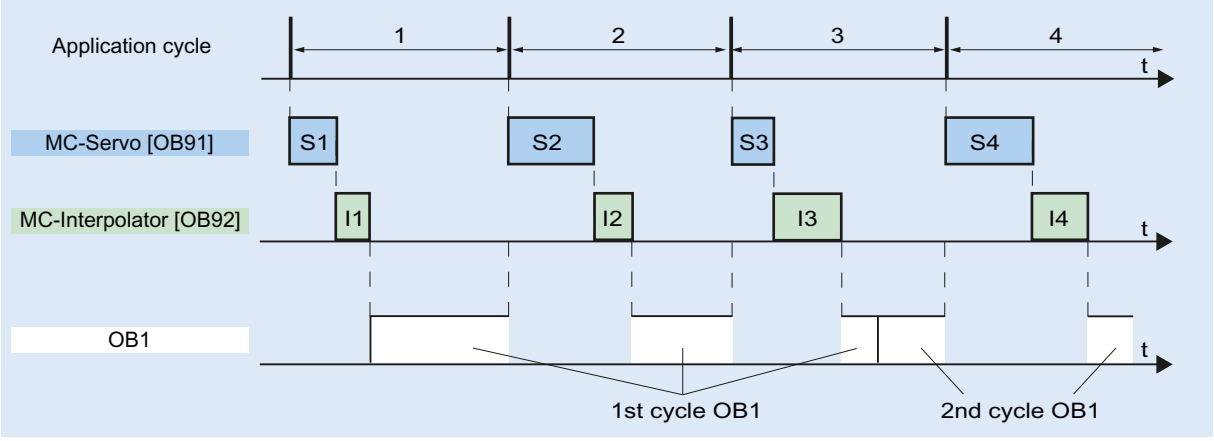
Operational Sequence and Timeouts (S7-1200)

When processing the Motion Control functionality, the organization blocks MC-Servo [OB91] and MC-Interpolator [OB92] are called and processed in each application cycle (processing also occurs in the STOP operating state of the CPU). The remaining cycle time is available for the processing of your user program.

For error-free program execution, keep to the following rules:

- In each application cycle, MC-Servo [OB91] must be started and executed completely.
- In every application cycle, the relevant MC-Interpolator [OB92] must at least be started.

The following figure shows an example of the error-free operational sequence for the processing of organization block OB1:



Overflows

If the set application cycle is not adhered to, for example because the application cycle is too short, overflows can occur.

An overflow of the MC-Servo [OB91], MC-Interpolator [OB92], MC_PreServo [OB67] and MC_PostServo [OB95] is entered in the diagnostic buffer of the CPU and results in setting the CPU to STOP.

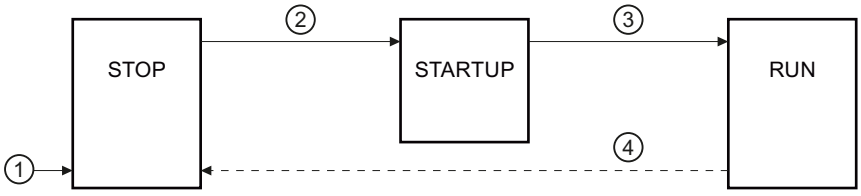
MC-PreServo, MC-Servo, MC-PostServo and MC-Interpolator are stopped. If necessary, you can evaluate the entry in the diagnostic buffer via a time error OB (OB80).

Operating modes (S7-1200)

This section examines the behavior of Motion Control in each operating mode, and in the transitions between operating modes. A general description of the operating modes can be found in system manual S7-1200.

Operating modes and transitions

The CPU has three operating modes: STOP, STARTUP and RUN. The following figure shows the operating modes and the operating mode transitions:



STOP mode

In STOP mode the user program is not processed and all process outputs are disabled. Thus no Motion Control jobs are executed.

The technology data blocks of the position-controlled axes are updated.

STARTUP mode

Before the CPU starts processing of the cyclical user program, the startup OBs are processed one time.

In STARTUP mode, the process outputs are disabled. Motion Control jobs are rejected.

The technology data blocks of the position-controlled axes are updated.

RUN mode

The user program is processed in RUN mode.

In RUN mode, the Motion Control commands programmed in OB1 are called and processed cyclically (other execution levels are possible).

The technology data blocks are updated.

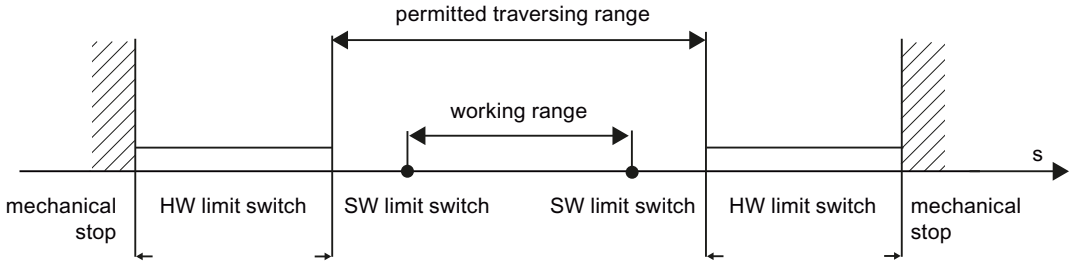
Operating mode transitions

The following table shows the behavior of Motion Control in the transitions between the operating modes:

No.	Operating mode transition	Behavior
①	POWER ON → STOP	The CPU performs a restart of the technology objects. The technology objects are reinitialized with the values from the load memory.
②	STOP → STARTUP	The technology objects are initialized with the start values of the CPU.
③	STARTUP → RUN	The process outputs are enabled.
④	RUN → STOP	When the CPU changes to RUN mode after STOP mode, all technology objects are disabled in accordance with the error response "remove enablement". Running Motion Control jobs are terminated.

13.2.2.3 Hardware and software limit switches (S7-1200)

Use the hardware and software limit switches to limit the "permitted traversing range" and the "working range" of your positioning axis technology object. The relationships are shown in the following diagram:



Hardware limit switches are limit switches that limit the maximum "permitted traversing range" of the axis. Hardware limit switches are physical switching elements that must be connected to interrupt-capable inputs of the CPU.

Software limit switches limit the "working range" of the axis. They should fall inside the hardware limit switches relative to the traversing range. Since the positions of the software limit switches can be flexibly set, the working range of the axis can be adapted on an individual basis, depending on the current traversing profile. In contrast to hardware limit switches, software limit switches are implemented exclusively via the software and do not require their own switching elements.

Hardware and software limit switches must be activated prior to use in the configuration or in the user program. Software limit switches are only active after homing the axis.

See also

CPU outputs relevant for motion control (Page 6143)

How the pulse interface works (Page 6146)

Relationship between the signal type and the direction of travel (Page 6146)

Jerk limit (Page 6165)

Homing (Page 6166)

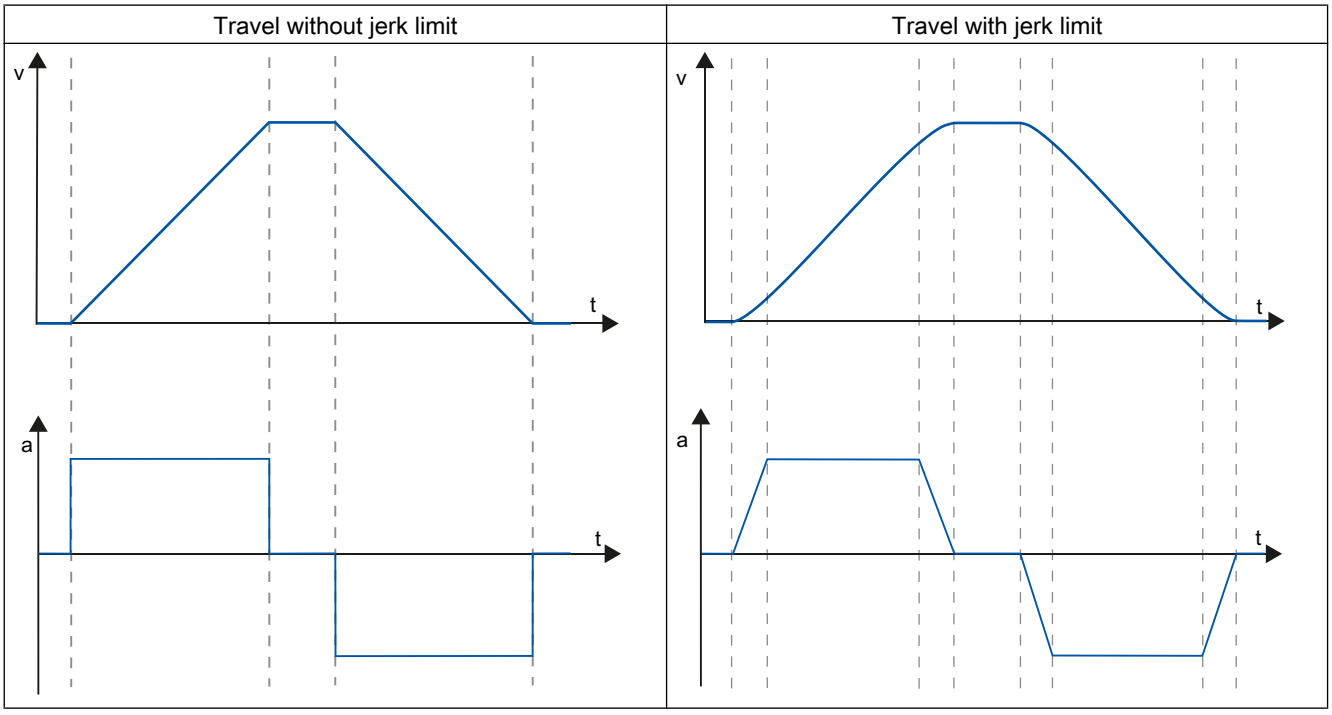
Integration of the positioning axis technology object (Page 6177)

Tools of the positioning axis technology object (Page 6180)

Position limits (Page 6198)

13.2.2.4 Jerk limit (S7-1200)

With the jerk limit you can reduce the stresses on your mechanics during an acceleration and deceleration ramp. The acceleration and deceleration value is not changed abruptly when the jerk limiter is active; it is gradually increased and decreased. The figure below shows the velocity and acceleration curve without and with jerk limit.



The jerk limit gives a "smoothed" velocity profile of the axis motion. This ensures, for example, soft starting and braking of a conveyor belt.

See also

Behavior of the axis when using the jerk limit (Page 6208)

CPU outputs relevant for motion control (Page 6143)

How the pulse interface works (Page 6146)

Relationship between the signal type and the direction of travel (Page 6146)

Hardware and software limit switches (Page 6164)

Homing (Page 6166)

Integration of the positioning axis technology object (Page 6177)

Tools of the positioning axis technology object (Page 6180)

13.2.2.5 Homing (S7-1200)

Homing means matching the axis coordinates of the technology object to the real, physical location of the drive. For position-controlled axes the entries and displays for the position refer exactly to these axis coordinates. Therefore, agreement between the axis coordinates and the real situation is extremely important. This step is necessary to ensure that the absolute target position of the axis is also achieved exactly with the drive.

In the S7-1200 CPU, axis homing is implemented with the motion control instruction, "MC_Home". The "Homed" status is displayed in the tags of the technology object <Axis name>.StatusBits.HomingDone . The following homing modes exist:

Homing modes

- Active homing
  In active homing mode, the motion control instruction "MC_Home" performs the required reference point approach. When the homing switch is detected, the axis is homed according to the configuration. Active traversing motions are aborted.
- Passive homing
  During passive homing, the "MC_Home" Motion Control instruction does not carry out any homing motion. The traversing motion required for this must be implemented by the user via other Motion Control instructions. When the homing switch is detected, the axis is homed according to the configuration. Active traversing motions are not aborted upon start of passive homing.
- Direct homing absolute
  The axis position is set regardless of the homing switch. Active traversing motions are not aborted. The value of input parameter "Position" of motion control instruction "MC_Home" is set immediately as the reference point of the axis.
- Direct homing relative
  The axis position is set regardless of the homing switch. Active traversing motions are not aborted. The following statement applies to the axis position after homing:
  New axis position = current axis position + value of parameter "Position" of instruction "MC_Home".

Resetting the "Homed" status

The "Homed" status of a technology object (<Axis name>.StatusBits.HomingDone) is reset under the following conditions:

- **Drive connection via PTO (Pulse Train Output):**
  - Start an "MC_Home" command for active homing
    (After successful completion of the homing operation, the "Homed" status is set again.)
  - Disabling of axis by the "MC_Power" Motion Control instruction
  - Changeover between automatic mode and manual control
  - After POWER OFF -> POWER ON of the CPU
  - After CPU restart (RUN-STOP -> STOP-RUN)
- **Technology objects with incremental actual values:**
  - Start an "MC_Home" command for active homing
    (After successful completion of the homing operation, the "Homed" status is set again.)
  - Error in the encoder system, or encoder failure
  - Restart of the technology object
  - After POWER OFF → POWER ON of the CPU
  - Memory reset
  - Modification of the encoder configuration
- **Technology objects with absolute actual values:**
  - Errors in the sensor system/encoder failure
  - Replacement of the CPU
  - Modification of the encoder configuration
  - Restoration of the CPU factory settings
  - Transfer of a different project to the controller

See also

CPU outputs relevant for motion control (Page 6143)

How the pulse interface works (Page 6146)

Relationship between the signal type and the direction of travel (Page 6146)

Hardware and software limit switches (Page 6164)

Jerk limit (Page 6165)

Integration of the positioning axis technology object (Page 6177)

Tools of the positioning axis technology object (Page 6180)

Homing (positioning axis technology object as of V2) (Page 6211)

13.2.3 Guidelines on use of motion control (S7-1200)

The guidelines described here present the basic procedure for using motion control with the CPU S7-1200.

Requirements

To use the positioning axis technology object, a project with a CPU S7-1200 must be created.

Procedure

Follow the steps below in the order given to use motion control with the CPU S7-1200. Use the following links for this purpose:

1. Adding a positioning axis technology object (Page 6182)
2. Working with the configuration dialog (Page 6183)
3. Download to CPU (Page 6260)
4. Function test of the axis in the commissioning window (Page 6261)
5. Programming (Page 6266)
6. Diagnostics of the axis control (Page 6285)

13.2.4 Using versions (S7-1200)

13.2.4.1 Overview of versions (S7-1200)

The relationship between the relevant versions for S7-1200 Motion Control can be found in the following table:

Technology version

You can check the currently selected technology version in the "Instructions" task card in the folder "Technology > Motion Control" and in the "Technology object > Add new object" dialog.

Select the technology version in the "Instructions" task card in the folder "Technology > Motion Control".

If a technology object with an alternative version is added in the "Add new object" dialog, the technology version will also be changed.

Note

The selection of an alternative technology version will also affect the Motion Control Instructions version (task card).

The technology objects and Motion Control instructions will only be converted to the selected version upon compilation or "Download to device".

Version of the technology object

The version of a technology object can be checked in the inspector window under "Properties > General > Information" in the "Version" box.

Motion Control instruction version

The Motion Control instruction version can be checked in the inspector window under "Properties > General > Information" in the "Version" box.

If the Motion Control instruction version used is not in line with the following compatibility list, the relevant Motion Control instructions will be highlighted in the program editor.

Compatibility list

Technology		CPU	Technology object	Motion Control Instruction
V1.0		V1.0, V2.0, V2.1, V2.2, V3.0	Axis V1.0	MC_Power V1.0 MC_Reset V1.0 MC_Home V1.0 MC_Halt V1.0 MC_MoveAbsolute V1.0 MC_MoveRelative V1.0 MC_MoveVelocity V1.0 MC_MoveJog V1.0
V2.0	Innovations: • Jerk limit • Command table • MC_ChangeDynamic	V2.1, V2.2, V3.0	Axis V2.0, Command table V2.0	MC_Power V2.0 MC_Reset V2.0 MC_Home V2.0 MC_Halt V2.0 MC_MoveAbsolute V2.0 MC_MoveRelative V2.0 MC_MoveVelocity V2.0 MC_MoveJog V2.0 MC_CommandTable V2.0 MC_ChangeDynamic V2.0
V3.0	Innovation: Load in RUN mode	V2.2, V3.0, V4.0	Axis V3.0, Command table V3.0	MC_Power V3.0 MC_Reset V3.0 MC_Home V3.0 MC_Stop V3.0 MC_MoveAbsolute V3.0 MC_MoveRelative V3.0 MC_MoveVelocity V3.0 MC_MoveJog V3.0 MC_CommandTable V3.0 MC_ChangeDynamic V3.0

Technology		CPU	Technology object	Motion Control Instruction
V4.0	Innovations: • MC_ReadParam • MC_WriteParam • Standardization of S7-1200 and S7-1500 Motion Control technology data blocks.	V4.0	Positioning axis V4.0, Command table V4.0	MC_Power V4.0 MC_Reset V4.0 MC_Home V4.0 MC_Halt V4.0 MC_MoveAbsolute V4.0 MC_MoveRelative V4.0 MC_MoveVelocity V4.0 MC_MoveJog V4.0 MC_CommandTable V4.0 MC_ChangeDynamic V4.0 MC_ReadParam V4.0 MC_WriteParam V4.0

Technology		CPU	Technology object	Motion Control Instruction
V5.0	Innovations: • Drive connection by means of PROFIdrive • Analog drive connection • Position control for PROFIdrive/analog drive connection • Position monitoring for PROFIdrive/analog drive connection • MC-Servo [OB91] • MC-Interpolator [OB92]	V4.1	Positioning axis V5.0, Command table V5.0	MC_Power V5.0 MC_Reset V5.0 MC_Home V5.0 MC_Halt V5.0 MC_MoveAbsolute V5.0 MC_MoveRelative V5.0 MC_MoveVelocity V5.0 MC_MoveJog V5.0 MC_CommandTable V5.0 MC_ChangeDynamic V5.0 MC_ReadParam V5.0 MC_WriteParam V5.0
V6.0	Innovations: • MC-PreServo • MC-PostServo • Data connection directly to the SINAMICS drive or via data block • Data connection directly to the analog output of a drive with analog drive connection or via a data block • Transfer of drive and encoder parameters in the device for PROFIdrive drives and encoders. • Move position-controlled drives without position control for servicing purposes • Simulation of position-controlled drives without configured or existing hardware • Level selection when configuring the hardware limit switch • Support PROFIdrive Telegram 4	V4.2	Positioning axis V6.0, Command table V6.0	MC_Power V6.0 MC_Reset V6.0 MC_Home V6.0 MC_Halt V6.0 MC_MoveAbsolute V6.0 MC_MoveRelative V6.0 MC_MoveVelocity V6.0 MC_MoveJog V6.0 MC_CommandTable V6.0 MC_ChangeDynamic V6.0 MC_ReadParam V6.0 MC_WriteParam V6.0

See also

Changing a technology version (Page 6172)

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

Status of limit switch (Page 6176)

13.2.4.2 Changing a technology version (S7-1200)

Before you can access all the benefits of a new technology version, you may need to set / modify the technology version for existing projects.

Note

Compatibility of the technology object tags

When switching between V1...3 and ≥ V4, please see the compatibility list (Page 6172) when using tags of the technology object in the user program, monitoring tables, etc .

Setting/changing a technology version

To set or change the technology version, follow these steps:

1. Open the program editor (e.g., by opening the OB1).
2. In the "Instructions" task card, select the desired technology version in the "Technology > Motion Control" folder.
3. Save and compile the project. Pay attention to any error information that is displayed during compilation. Deal with the causes of the errors indicated.
4. Check the configuration of the technology objects.
5. If necessary, adapt the tag names in the following objects in line with the compatibility list.

- User program
- Watch tables
- Force tables
- HMI configuration
- Trace configuration

See also

Overview of versions (Page 6168)

Status of limit switch (Page 6176)

13.2.4.3 Compatibility list of variables V1...3 <-> V4...5 (S7-1200)

The technology data blocks for S7-1200 Motion Control and S7-1500 Motion Control have been standardized within the framework of the V4 technology. As of V4, this has resulted in new tags and tag names for the positioning axis and command table technology objects.

Observe the information in the following tables if you have used tags of the technology objects in the user program and you want to convert the project from V1...3 to V4 or higher (or vice versa).

The tags listed in the "Automatic conversion V1...3 to ≥ V4" column are converted automatically when the project is compiled. Tag names in monitoring and force tables or the HMI or trace configuration are not converted.

The following tags are new or have been adapted and may have to be corrected in the user program, watch tables, etc.:

Config tags (positioning axis)

Tag name V1.0 to V3.0	Tag name as of V4.0	Automatic conversion V1..3 to ≥ V4
<Axis name>.Config.DynamicDefaults.Acceleration	<Axis name>.DynamicDefaults.Acceleration	Yes
<Axis name>.Config.DynamicDefaults.Deceleration	<Axis name>.DynamicDefaults.Deceleration	Yes
<Axis name>.Config.DynamicDefaults.EmergencyDeceleration	<Axis name>.DynamicDefaults.EmergencyDeceleration	Yes
<Axis name>.Config.DynamicDefaults.Jerk	<Axis name>.DynamicDefaults.Jerk	Yes
<Axis name>.Config.DynamicDefaults.JerkActive	Not available The jerk is activated if the configured jerk is > 0.004 pulse/s^3.	No
<Axis name>.Config.DynamicLimits.MaxVelocity	<Axis name>.DynamicLimits.MaxVelocity	Yes
<Axis name>.Config.DynamicLimits.MinVelocity	<Axis name>.DynamicLimits.MinVelocity	Yes
<Axis name>.Config.General.LengthUnit	<Axis name>.Units.LengthUnit	Yes
<Axis name>.Config.Homing.AutoReversal	<Axis name>.Homing.AutoReversal	Yes
<Axis name>.Config.Homing.Direction	<Axis name>.Homing.ApproachDirection	Yes
<Axis name>.Config.Homing.FastVelocity	<Axis name>.Homing.ApproachVelocity	Yes
<Axis name>.Config.Homing.Offset	<Axis name>.Sensor[1].ActiveHoming.HomePositionOffset	Yes
<Axis name>.Config.Homing.SideActiveHoming	<Axis name>.Sensor[1].ActiveHoming.SideInput	Yes
<Axis name>.Config.Homing.SidePassiveHoming	<Axis name>.Sensor[1].PassiveHoming.SideInput	Yes
<Axis name>.Config.Homing.SlowVelocity	<Axis name>.Homing.ReferencingVelocity	Yes
<Axis name>.Config.Homing.SwitchedLevel	<Axis name>.Sensor[1].ActiveHoming.SwitchLevel <Axis name>.Sensor[1].PassiveHoming.SwitchLevel	No
<Axis name>.Config.Mechanics.InverseDirection	<Axis name>.Actor.InverseDirection	Yes
<Axis name>.Config.Mechanics.LeadScrew	<Axis name>.Mechanics.LeadScrew	Yes
<Axis name>.Config.Mechanics.PulsesPerDriveRevolution	<Axis name>.Actor.DriveParameter.PulsesPerDriveRevolution	Yes

Tag name V1.0 to V3.0	Tag name as of V4.0	Automatic conversion V1..3 to ≥ V4
<Axis name>.Config.PositionLimits_HW.Active	<Axis name>.PositionLimitsHW.Active	Yes
<Axis name>.Config.PositionLimits_HW.MaxSwitchedLevel	<Axis name>.PositionLimitsHW.MaxSwitchLevel	Yes
<Axis name>.Config.PositionLimits_HW.MinSwitchedLevel	<Axis name>.PositionLimitsHW.MinSwitchLevel	Yes
<Axis name>.Config.PositionLimits_SW.Active	<Axis name>.PositionLimitsSW.Active	Yes
<Axis name>.Config.PositionLimits_SW.MaxPosition	<Axis name>.PositionLimitsSW.MaxPosition	Yes
<Axis name>.Config.PositionLimits_SW.MinPosition	<Axis name>.PositionLimitsSW.MinPosition	Yes
Not available	<Axis name>.Actor.DirectionMode	No
Not available	<Axis name>.Actor.Type	No
Not available	<Axis name>.Sensor[1].ActiveHoming.Mode	No
Not available	<Axis name>.Sensor[1].PassiveHoming.Mode	No

ErrorBits tags (positioning axis)

Tag name V1.0 to V3.0	Tag name as of V4.0	Automatic conversion V1..3 to ≥ V4
<Axis name>.ErrorBits.HwLimitMax	<Axis name>.ErrorBits.HWLimit (Note the new status bits and the section Status of the limit switch (Page 6176).)	No
<Axis name>.ErrorBits.HwLimitMin		
<Axis name>.ErrorBits.SwLimitMaxExceeded	<Axis name>.ErrorBits.SWLimit (Note the new status bits and the section Status of the limit switch (Page 6176).)	No
<Axis name>.ErrorBits.SwLimitMaxReached		
<Axis name>.ErrorBits.SwLimitMinExceeded		
<Axis name>.ErrorBits.SwLimitMinReached		
Not available	<Axis name>.ErrorBits.DirectionFault	No

MotionStatus tags (positioning axis)

Tag name V1.0 to V3.0	Tag name as of V4.0	Automatic conversion V1..3 to ≥ V4
<Axis name>.MotionStatus.Distance	<Axis name>.StatusPositioning.Distance	Yes
<Axis name>.MotionStatus.Position	<Axis name>.Position	Yes
<Axis name>.MotionStatus.TargetPosition	<Axis name>.StatusPositioning.TargetPosition	Yes
<Axis name>.MotionStatus.Velocity	<Axis name>.Velocity	Yes

StatusBits tags (positioning axis)

Tag name V1.0 to V3.0	Tag name as of V4.0	Automatic conversion V1..3 to ≥ V4
<Axis name>.StatusBits.Homing	<Axis name>.StatusBits.HomingCommand	Yes
<Axis name>.StatusBits.SpeedCommand	<Axis name>.StatusBits.VelocityCommand	Yes
Not available	<Axis name>.StatusBits.HWLimitMaxActive	No
Not available	<Axis name>.StatusBits.HWLimitMinActive	No
Not available	<Axis name>.StatusBits.SWLimitMaxActive	No
Not available	<Axis name>.StatusBits.SWLimitMinActive	No

Tags (command table)

Tag name V1.0 to V3.0	Tag name as of V4.0	Automatic conversion V1..3 to ≥ V4
<Command table>.Config.Command[n].Position	<Command table>.Command[n].Position	Yes
<Command table>.Config.Command[n].Velocity	<Command table>.Command[n].Velocity	Yes
<Command table>.Config.Command[n].Duration	<Command table>.Command[n].Duration	Yes
<Command table>.Config.Command[n].NextStep	<Command table>.Command[n].NextStep	Yes
<Command table>.Config.Command[n].StepCode	<Command table>.Command[n].StepCode	Yes

See also

Overview of versions (Page 6168)

Changing a technology version (Page 6172)

Tags of the positioning axis technology object V4...5 (Page 6436)

13.2.4.4 Compatibility list of variables V4...5 <-> V6 (S7-1200)

The technology data blocks for S7-1200 Motion Control and S7-1500 Motion Control have continued to be standardized within the framework of the V6 technology. As of V6, this has resulted in new tag names for the technology object positioning axis.

Observe the information in the following tables if you have used tags of the technology objects in the user program and you want to convert the project from V4...5 to ≥ V6 or higher (or vice versa).

The tags listed in the "Automatic conversion V4...5 to ≥ V6" column are converted automatically when the project is compiled. Tag names in monitoring and force tables, HMI and trace configurations are not converted.

The following tags are new or have been adapted and may need to be corrected in the user program, monitoring tables, etc.:

Config variables (positioning axis)

Variable name V4.0 to V5.0	Variable name as of V6.0	Automatic conversion V4...5 to ≥ V6
<Axis name>.PositionLimitsSW.Active	<Axis name>.PositionLimits_SW.Active	Yes
<Axis name>.PositionLimitsSW.MinPosition	<Axis name>.PositionLimits_SW.MinPosition	Yes
<Axis name>.PositionLimitsSW.MaxPosition	<Axis name>.PositionLimits_SW.MaxPosition	Yes
<Axis name>.PositionLimitsHW.Active	<Axis name>.PositionLimits_HW.Active	Yes
<Axis name>.PositionLimitsHW.MinSwitchLevel	<Axis name>.PositionLimits_HW.MinSwitchLevel	Yes
<Axis name>.PositionLimitsHW.MinSwitchAddress	<Axis name>.PositionLimits_HW.MinSwitchAddress	Yes

13.2.4.5 Status of limit switch (S7-1200)

The status and error bits for the display of the reached limit switch have been adapted in version V4.

In order to replicate the behavior of the error bits of versions V1...3, use the following logical operators:

V1...3	V4 or higher
<Axis name>.ErrorBits.HwLimitMin	<Axis name>.ErrorBits.HWLimit AND <Axis name>.StatusBits.HWLimitMinActive
<Axis name>.ErrorBits.HwLimitMax	<Axis name>.ErrorBits.HWLimit AND <Axis name>.StatusBits.HWLimitMaxActive
<Axis name>.ErrorBits.SwLimitMinReached	<Axis name>.ErrorBits.SWLimit AND (<Axis name>.Position = <Axis name>.PositioningLimits_SW.MinPosition)
<Axis name>.ErrorBits.SwLimitMinExceeded	<Axis name>.ErrorBits.SWLimit AND (<Axis name>.Position < <Axis name>.PositioningLimits_SW.MinPosition)
<Axis name>.ErrorBits.SwLimitMaxReached	<Axis name>.ErrorBits.SWLimit AND (<Axis name>.Position = <Axis name>.PositioningLimits_SW.MaxPosition)
<Axis name>.ErrorBits.SwLimitMaxExceeded	<Axis name>.ErrorBits.SWLimit AND (<Axis name>.Position > <Axis name>.PositioningLimits_SW.MaxPosition)

See also

Overview of versions (Page 6168)

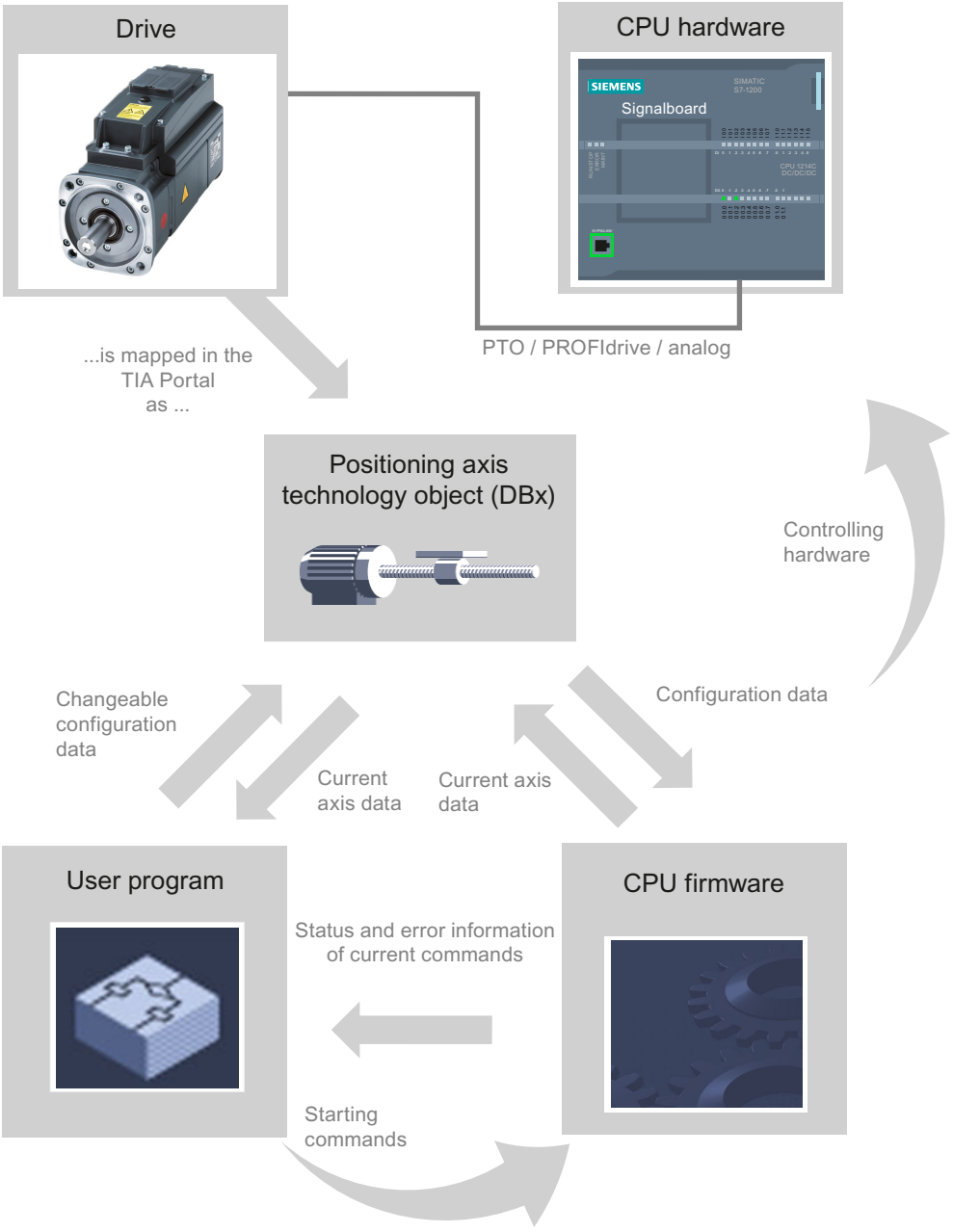
Changing a technology version (Page 6172)

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

13.2.5 Positioning axis technology object (S7-1200)

13.2.5.1 Integration of the positioning axis technology object (S7-1200)

The following representation shows the relationships between the hardware and software components that are implemented when using the positioning axis technology object:



CPU hardware

The physical drive is controlled and monitored by the CPU hardware.

Drive

The drive represents the unit of power unit and motor. You can use stepper motors and servo motors with pulse, PROFIdrive or analog interfaces.

Positioning axis technology object

The physical drive including mechanics is mapped in the TIA Portal as a positioning axis technology object. To do this, configure the positioning axis technology object with the following parameters:

- Selection of the PTOs (Pulse Train Output)/PROFIdrive drives/analog outputs to be used and configuration of the drive interface
- Parameter for mechanics and gear transmission of the drive (or the machine or system)
- Parameters for position limits and position monitoring
- Parameters for dynamics and homing
- Parameters for the control loop

The configuration of the positioning axis technology object is saved in the technology object (data block). This data block also forms the interface between the user program and the CPU firmware. The current axis data is saved in the data block of the technology object at the runtime of the user program.

User program

You start Motion Control instructions jobs in the CPU firmware with the user program. The following jobs for controlling the axis are possible:

- Enable and disable axis
- Position axis absolutely
- Position axis relatively
- Move axis with velocity set point
- Run axis commands as movement sequence (technology as of V2, PTO only)
- Moving axes in jog mode
- Stop axis
- Reference axis; set reference point
- Change dynamic settings of axis
- Continuously read motion data of the axis
- Read and write variable of the axis
- Acknowledge error

You determine the command parameters with the input parameters of the Motion Control instructions and the axis configuration. The output parameters of the instruction give you up to date information about the status and any errors of the command.

Before starting a command for the axis, you must enable the axis with the Motion Control instruction "MC_Power".

You can read out configuration data and current axis data with the variables of the technology object. You can change individual, changeable variables of the technology object (e.g. the current acceleration) from the user program.

You can also change the dynamic settings of the axis with the Motion Control instruction "MC_ChangeDynamic" and write additional configuration data with "MC_WriteParam". You can read the current motion status of the axis with the Motion Control instruction "MC_ReadParam".

CPU firmware

The motion control jobs started in the user program are processed in the CPU firmware. When using the axis control panel, Motion Control jobs are triggered by operating the axis control panel. The CPU firmware performs the following jobs depending on the configuration:

- Calculate the exact motion profile for motion jobs and emergency stop situations
- Position control for drive connection via PROFIdrive/analog drive connection
- Control of the pulse and direction signal for drive connection via PTO
- Control of the drive enable
- Monitoring of the drive and the hardware and software limit switches
- Up to date feedback of status and error information to the Motion Control instructions in the user program
- Writing of current axis data into the data block of the technology object

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

CPU outputs relevant for motion control (Page 6143)

Relationship between the signal type and the direction of travel (Page 6146)

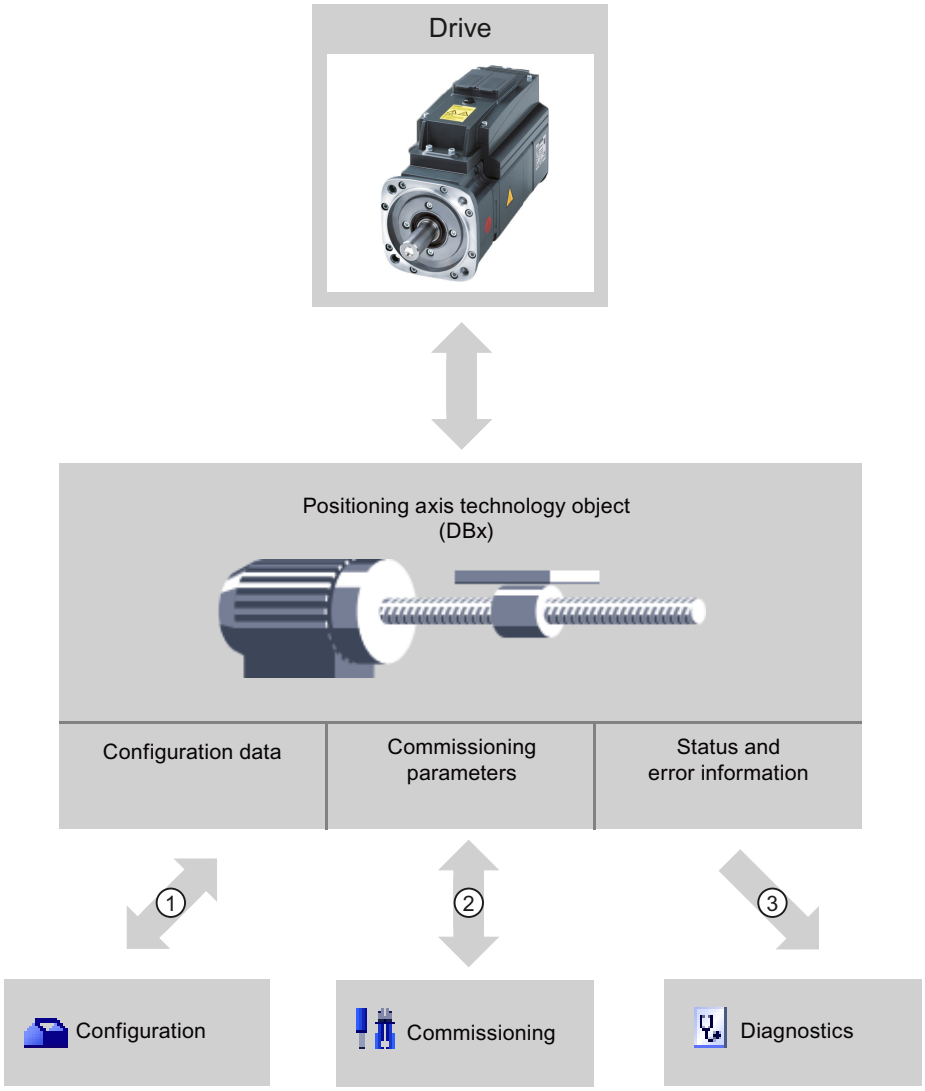
Tools of the positioning axis technology object (Page 6180)

Hardware and software limit switches (Page 6164)

Homing (Page 6166)

13.2.5.2 Tools of the positioning axis technology object (S7-1200)

The TIA Portal provides the "Configuration", "Commissioning", and "Diagnostics" tools for the positioning axis technology object. The following representation shows the interaction of the three tools with the technology object and the drive:



①	Reading and writing of configuration data of the technology object
②	Drive control via the technology object. Read the axis status for display in the axis control panel. Optimization of the position control
③	Readout of the current status and error information of the technology object Additional telegram information is displayed for PROFIdrive drives.

Configuration

Use the "Configuration" tool to configure the following properties of the positioning axis technology object:

- Selection of the PTOs (Pulse Train Output)/PROFIdrive drives/analog outputs to be used and configuration of the drive interface
- Properties of the mechanics and the transmission ratio of the drive (or machine/plant)
- Properties of the position limits and the position monitoring
- Properties of the dynamics and the homing
- Parameters of the control loop

Save the configuration in the data block of the technology object.

Commissioning

Use the "Commissioning" tool to test the function of your axis without having to create a user program. When the tool is started, the axis control panel will be displayed. The following commands are available on the axis control panel:

- Enabling and disabling the axis
- Move axis in jog mode
- Position axis in absolute and relative terms
- Home axis
- Acknowledge errors

The dynamic values can be adjusted accordingly for the motion commands. The axis control panel also shows the current axis status.

With drive connection via PROFIdrive / analog output, tuning supports you in determining the optimal gain for the control loop.

Diagnostics

Use the "Diagnostics" tool to keep track of the current status and error information for the axis and drive.

See also

CPU outputs relevant for motion control (Page 6143)

Relationship between the signal type and the direction of travel (Page 6146)

Integration of the positioning axis technology object (Page 6177)

Hardware and software limit switches (Page 6164)

Homing (Page 6166)

Configuring the positioning axis technology object (Page 6183)

Axis control panel (Page 6261)

Axis - Diagnostics (Page 6285)

13.2.5.3 Adding a positioning axis technology object (S7-1200)

Requirements

A project with a CPU S7-1200 has been created.

Procedure

To add a positioning axis technology object in the project tree, follow these steps:

1. Open the "CPU > Technology objects" folder in the project tree.
2. Double-click the "Add new object" command.
   The "Add new object" dialog opens.
3. Select the "Motion Control" technology.
4. Open the "Motion Contro > S7-1200 Motion Control" folder.
5. Select the desired technology version in the "Version" column.
6. Select the "TO_PositioningAxis" object.
7. Enter the name of the axis in the "Name" input box.
8. To change the automatically assigned data block number, select the "Manual" option.
9. To display additional information about the technology object, click "Additional information".
10. Confirm your entry with "OK".

Result

The new technology object is created and saved to the "Technology objects" folder in the project tree.

The organization blocks MC-Servo [OB91] and MC-Interpolator [OB92] are automatically created in the "Program blocks" folder. The technology objects are processed in these organization blocks. The position controller is calculated in the MC-Servo [OB91]. The MC-Interpolator [OB92] takes over the evaluation of the Motion Control instructions, the setpoint generation and the monitoring functionality.

See also

Guidelines on use of motion control (Page 6168)

13.2.5.4 Configuring the positioning axis technology object (S7-1200)

Working with the configuration dialog (S7-1200)

You configure the properties of the technology object in the configuration window. Proceed as follows to open the configuration window of the technology object:

1. Open the group of the required technology object in the project tree.
2. Double-click the "Configuration" object.

The configuration is divided into the following categories:

- **Basic parameters**
  The basic parameters contain all the parameters which must be configured for a functioning axis.
- **Extended parameters**
  The advanced parameters include parameters to adapt to your drive or your plant.

Configuration window icons

Icons in the area navigation of the configuration show additional details about the status of the configuration:

	The configuration contains default values and is complete. The configuration contains only default values. With these default values you can use the technology object without additional changes.
	The configuration contains user-defined or automatically adapted values and is complete. All input fields of the configuration contain valid values and at least one preset value has changed.
	The configuration is incomplete or incorrect At least one input field or drop-down list contains an invalid value. The corresponding field or the drop-down list is displayed on a red background. Click the roll-out error message to display the cause of the error.
	The configuration is valid but contains warnings For example, only one hardware limit switch is configured. Depending on the application, the lacking configuration of a hardware limit switch may result in a hazard. The corresponding field or the drop-down list is displayed on a yellow background.

See also

Guidelines on use of motion control (Page 6168)

Basic parameters (Page 6184)

Extended parameters (Page 6196)

Monitor values (S7-1200)

If there is an online connection to the CPU, the icon "Monitor all"  is displayed in the configuration dialog of the technology object.

The "Monitor all" function provides the following options:

- Comparison of configured start values of the project with the start values in the CPU and the actual values
- Direct editing of actual values and the start values of the project
- Immediate detection and display of input errors with suggested corrections
- Backup of the actual values in the project by manual transfer to the start value of the project

Icons and operator controls

If there is an online connection to the CPU, the actual values are displayed at the parameters.

In addition to the actual values of the parameters, the following symbols appear:

Icon	Description
	Start value in CPU matches the configured Start value in the project
	Start value in CPU does not match the configured Start value in the project
	A comparison of the start value in the CPU with the configured start value in the project cannot be performed because the selected CPU module does not support this comparison.
	The value is not comparable with any significance since it is not relevant in one of the configurations.
	Use the button to show the start value of the CPU and the start value of the project for the respective parameter.

The actual value and the start value in the project can be changed directly and then downloaded to the CPU. The change of the actual value is transferred directly to the CPU for directly modifiable parameters.

Basic parameters (S7-1200)

Configuration - General (S7-1200)

Configure the basic properties of the positioning axis technology object in the "General" configuration window.

Axis name

Define the name of the axis or the name of the positioning axis technology object in this field. The technology object is listed under this name in the project tree.

Drive

Select the type of drive connection:

- **PTO (Pulse Train Output)**
  The drive is connected via a pulse generator output, an optional enable output and an optional ready input.
- **Analog drive connection**
  The drive is connected via an analog output, an encoder, an optional enable output and an optional ready input.
  All movements of the axis are position-controlled.
- **PROFIdrive**
  The drive is connected via PROFINET/PROFIBUS. Communication between controller and drive is by means of PROFIdrive telegrams.
  All movements of the axis are position-controlled.

If you select the "Analog drive connection" or "PROFIdrive", additional elements are added to the navigation of the configuration:

- Encoder
- Modulo
- Position supervisions (positioning supervision, following error and standstill signal)
- Control loop

In the additional configuration windows, you configure the encoders that are to be connected and the resulting options for position control and position monitoring.

Unit of measurement position

In the drop-down list, select the desired measurement unit for the dimension system of the axis. The selected measurement unit is used for further configuration of the positioning axis technology object and for displaying the current axis data.

The values at the input parameters (Position, Distance, Velocity, ...) of the Motion Control instructions also refer to this measurement unit.

Note

Select the drive connection and the measurement unit of the position at the beginning of the axis configuration.

With a subsequent change, the parameters are reset or re-initialized, which requires you to check the parameters of the configuration dialogs once again.

You may have to adapt the values of the input parameters of the Motion Control instructions to the new unit of measurement in the user program.

Simulation

In the drop-down list, select whether or not the drive and the encoder are to be simulated. The simulation can be selected for the analog drive connection or for a PROFIdrive drive. The configuration of the drive and encoder hardware is not required for simulation mode (potential errors in the drive and encoder configuration are ignored).

Application: The drive is simulated, for example, for commissioning and later operated with the hardware that may be configured.

The "Simulation" operating mode can be controlled/changed during runtime of the user program with a download or tag (<Achsname>.Simulation.Mode). The axis must be disabled for this (Enable = FALSE).

In simulation mode, setpoints are not output to the drive and actual values are not read in from the drive/encoder. Hardware limit switches and homing switches have no effect.

The following table shows Motion Control instructions with adapted behavior in simulation mode.

Motion Control instruction	Behavior in simulation mode
MC_Power	The axis is enabled immediately without waiting for feedback from the drive.
MC_Home	Homing jobs are executed immediately without simulated axis motion.

PTO drives work without control loop. No separate simulation function is required in order to simulate a PTO drive when the PTO drive is not connected.

See also

CPU outputs relevant for motion control (Page 6143)

Relationship between the signal type and the direction of travel (Page 6146)

Configuration - General ("Axis" technology object V1...3) (Page 6368)

Configuration - Drive (S7-1200)

Configuration - Drive - PTO (Pulse Train Output) (S7-1200)

In the "Drive" configuration window, configure the pulse generator and the enable and feedback of the drive.

Hardware interface

The pulses are output to the power unit of the drive by fixed assigned digital outputs.

In CPUs with relay outputs, the pulse signal cannot be output at these outputs because the relays do not support the necessary switching frequencies. To be able to work with the PTO (Pulse Train Output) on these CPUs, you must use a signal board with digital outputs.

Note

The PTO requires the functionality of a high-speed counter (HSC). An internal HSC is used for this, the count of which cannot be evaluated.

Pulse generator

In the drop-down list, select the PTO (Pulse Train Output) to control the stepper motor or servo motor by means of pulse interface. If you have not used the pulse generators and high-speed counters elsewhere in the device configuration, the hardware interface can be configured automatically. In this case, the PTO selected in the drop-down list is displayed with a white background.

If PTO (PulseTrain Output) is selected, the "Device configuration" button takes you to the parameter assignment of the pulse options in the device configuration of the CPU. This may be necessary if there is a conflict because the PTO is being used at the other end or the parameters have been changed by the user.

Signal type

Select the signal type in the drop-down list. The following signal types are available:

- **PTO (pulse A and direction B)**
  A pulse output and a direction output are used for controlling the stepper motor.
- **PTO (clock up A and clock down B)**
  One pulse output each for motion in positive direction and negative direction is used for controlling the stepper motor.
- **PTO (A/B phase-shifted)**
  Both pulse outputs for Phase A and for Phase B run at the same frequency.
  The period of the pulse outputs is evaluated at the drive end as a step.
  The phase offset between Phase A and Phase B determines the direction of the motion.
- **PTO (A/B phase offset - quadruple)**
  Both pulse outputs for Phase A and for Phase B run at the same frequency.
  All positive edges and all negative edges of Phase A and Phase B are evaluated as a step at the drive end.
  The phase offset between Phase A and Phase B determines the direction of the motion.

The following table shows the parameters to be configured depending on the signal type:

Signal type/parameter		Description
PTO (pulse A and direction B)		
	Pulse output	Select the pulse output for motion in positive direction in this field. You can select the output using a symbolic address or assign it to an absolute address.
	Activate direction output	With this option, you enable or disable the direction output. The motion direction is restricted when you disable the direction output.
	Direction output	Select the output for the direction output in this field. You can select the output using a symbolic address or assign it to an absolute address.
PTO (clock up A and clock down B)		
	Pulse output forward	Select the pulse output for motion in positive direction in this field. You can select the output using a symbolic address or assign it to an absolute address.
	Pulse output backward	Select the pulse output for motion in negative direction in this field. You can select the output using a symbolic address or assign it to an absolute address.
PTO (A/B phase offset)/PTO (A/B phase offset - quadruple)		
	Signal A	Select the pulse output for Phase A signals in this field. You can select the output using a symbolic address or assign it to an absolute address.
	Signal B	Select the pulse output for Phase B signals in this field. You can select the output using a symbolic address or assign it to an absolute address.

Drive enable and feedback

In this area, you configure the output for drive enable and the input for the "Drive ready" feedback of the drive:

- **Enable output**
  Select the enable output for the drive enable in this field.
- **Ready input**
  Select the ready input for the "Drive ready" feedback of the drive in this field

Drive enable is controlled by Motion Control instruction "MC_Power" and enables power to the drive. The drive signals "Drive ready" to the CPU if it is ready to start executing movement after receiving the drive enable.

If the drive does not have any interfaces of this type, you do not have to configure the parameters. In this case, select the value TRUE for the ready input.

Configuration - Drive - Analog drive connection (S7-1200)

In the "Drive" configuration window, configure the analog output and the enable and feedback of the drive.

Hardware interface

The speed setpoint is output to the power unit of the drive by means of permanently assigned analog outputs.

Configure the inputs and outputs for the control of the drive in this area:

- **Analog output**
  In this field, select the PLC variable of the analog output via which the drive is controlled. When you open the autocompletion, all output addresses are displayed with 16 bits (WORD, INT, UINT). The variable of a data block with the WORD data type can be selected for data connection via a data block.
  You can also enter an address, for example QW20. If the address is valid, the name "Axis_1_AnalogOutput" is generated for this address and inserted in the variable table. For the address to be valid, it needs to be occupied by the appropriate data type and a HW module.
- **Selection of enable output**
  Select an available output as the enable output for the drive enable in this field.
- **Selection of ready input**
  Select the ready input for the "Drive ready" feedback of the drive in this field

Drive enable is controlled by Motion Control instruction "MC_Power" and enables power to the drive. The drive signals "Drive ready" to the CPU if it is ready to start executing movement after receiving the drive enable. If the drive does not have any interfaces of this type, you do not have to configure the parameters. In this case, select the value TRUE for the ready input.

Data exchange with the drive

In this area, you can configure the scaling of the setpoint speed:

- **Reference speed**
  The reference speed of the drive is the speed, with which the drive spins when there is an output of 100% at the analog output. The reference speed must be configured in the drive, and transferred into the configuration of the technology object.
  The analog value that is output at 100% depends on the type of the analog output. As an example, for an analog output with ±10 V, the value 10 V is output at 100%.
- **Maximum speed**
  In this field, specify the maximum speed of the drive.
  The maximum speed is limited by the drive and by the value range of the analog output. In the simplest situation, the reference speed and maximum speed are identical.
  Analog outputs can be overloaded by approximately 17%. If the drive permits overloading, you can use this to operate an analog output as a limit in the -117% to 117% range.
- **Invert drive direction**
  To invert the rotation direction of the drive, select the check box.

Configuration - Drive - PROFIdrive (S7-1200)

In the "Drive" configuration window, configure the data connection and the parameters of the PROFIdrive drive.

PROFIdrive drive (as of V6)

- **Data connection**
  In the drop-down list, select whether the data connection is to be made directly with the drive device or via an editable data block in the user program.
- **Drive** (for data connection: "Drive")
  In the "Drive" field, select an already configured PROFIdrive drive.
- **Data block** (for data connection: "Data block")
  Select a previously created data block which contains a variable structure of the data type "PD_TELx" ("x" stands for the telegram number to be used).

Data exchange with the drive

In this area, you can configure the data exchange between the drive and controller:

- **Drive telegram** (for data connection: "Data block" not selectable)
  In the drop-down list, check or select the telegram of the drive. The specification must match the device configuration of the drive.
- **Input/output address**
  The fields show the symbolic and absolute input and output address of the telegram.
- **Invert drive direction**
  To invert the rotation direction of the drive, select the check box.
- **Automatic transfer of drive parameters in the device**
  Select the check box if you want the drive parameters "Reference speed" and "Maximum speed" to be transferred as values from the drive configuration to the CPU. The drive parameters are transferred from the bus after the (re-)initialization of the technology object and (re)start of the drives and the CPU.
  Alternatively, you must synchronize the following parameters manually:
  - **Reference speed**
    Configure the reference speed to match the one in the configuration of the drive.
    On the bus, the value 16#4000 is transferred, for example, which corresponds to 100% of the reference speed.
  - **Maximum speed**
    Configure the maximum speed of the drive in this field.
    The maximum speed is obtained from the configuration of the drive. A maximum of -200% to +200% of the reference speed can be transmitted over the bus. The maximum speed can thus be twice the reference speed at maximum.

Note

Automatic transfer of drive parameters is only possible with SINAMICS drives as of V4.x. For this, "Drive" must be selected as the data connection in the configuration window.

See also

Data connection PROFIdrive drive / PROFIdrive encoder (Page 6154)

Automatic transfer of drive and encoder parameters in the device (Page 6151)

Configuration - Encoder (S7-1200)

Encoder connection (S7-1200)

Depending on the selection of the encoder connection, you configure various parameters in the "Encoder" configuration window. The following encoder connections are possible:

- Encoder on high-speed counter (HSC) (Page 6194)
- PROFIdrive encoder on PROFINET/PROFIBUS (encoder on drive, encoder on technology module, PROFIdrive encoder) (Page 6192)

Configuration - Encoder - Encoder on PROFINET/PROFIBUS (S7-1200)

Encoder selection

In the "PROFIdrive encoder" box, select a PROFIdrive encoder on PROFINET.

- **Data connection**
  In the drop-down list, select whether the data connection should be established directly with the encoder or via a data block that can be edited in the user program.
- **PROFIdrive encoder/data block**
  Select a previously configured PROFIdrive encoder in this configuration field.
  The following encoders can be selected:
  - **Connection to drive (not with analog drive connection)**
    The encoder is connected to the drive. The encoder signals are evaluated by the drive and transmitted to the controller as part of the drive telegram (telegram 3 or 4) (the encoders of the telegrams from other drives cannot be used).
    The encoder is configured using the configuration of the PROFIdrive drive.
  - **Encoder on technology module (TM)**
    Select a previously configured technology module and the channel to be used. Only technology modules set to the "Position input for Motion Control" mode are displayed for selection.
    If no technology module is available for selection, change to the device configuration and add a technology module.
    You can identify the technology modules suitable for position detection for Motion Control in the documentation for the technology module and the catalog data.
  - **PROFIdrive encoder on PROFINET/PROFIBUS (PROFIdrive)**
    In the "PROFIdrive encoder" field, select a previously configured encoder on PROFINET/PROFIBUS. Switch to the device configuration in the network view and add an encoder, in the event that no encoder is offered for selection.

  If "Data block" was selected for the data connection, a previously created data block containing a tag structure of data type "PD_TELx" must be selected here ("x" stands for the telegram number to be used by which the encoder is connected).

Data exchange with encoder

In this area, you can configure the data exchange between the encoder and controller:

- **Encoder telegram** (for data connection: "Data block" not selectable)
  In the drop-down list, select the telegram of the encoder. The specification must match the device configuration.
- **Input/output address**
  The fields show the symbolic and absolute input and output address of the telegram.

- **Invert encoder direction**
  To invert the actual value of the encoder, select the check box.
- **Automatic transfer of encoder parameters in the device**
  Select the check box if you want to transfer the encoder parameters from the encoder configuration to the CPU. The encoder parameters are transferred from the bus after the (re-)initialization of the technology object and (re)start of the encoder and the CPU. The encoder type must be the same in the configuration of the axis and in the configuration of the drive.

  Note

  Automatic transfer of encoder parameters is only possible with PROFIdrive encoders as of product version A16. For this, "Encoder" must be selected as the data connection in the configuration window.

  A product version > V4.x is required to use an encoder on the SINAMICS drive.

  The parameters must be adjusted manually if there is no automatic transfer of encoder parameters. You can find the parameters to be synchronized in the section Automatic transfer of drive and encoder parameters in the device (Page 6151).

Encoder type

Set the employed encoder type in the "Encoder type" box. The following encoder types can be selected:

- **Linear incremental**
- **Linear absolute**
- **Rotary incremental**
- **Rotary absolute**

Configure the various parameters depending on the selected encoder type. Depending on the selected encoder type, configure the following parameters:

Encoder type/parameter		Description
Linear incremental		
	Distance between two increments	In this field, you configure the distance between two steps of the encoder.
	Fine resolution - Bits in incr. actual value (Gn_XIST1)	In this field, configure the number of bits for fine resolution within the incremental actual value (Gn_XIST1).
Linear absolute		
	Distance between two increments	In this field, you configure the distance between two steps of the encoder.
	Fine resolution - Bits in incr. actual value (Gn_XIST1)	In this field, configure the number of bits for fine resolution within the incremental actual value (Gn_XIST1).
	Fine resolution - Bits in abs. actual value (Gn_XIST2)	In this field, configure the number of reserved bits for the multiplication factor of the absolute value of the fine resolution (Gn_XIST2).
Rotary incremental		

Encoder type/parameter		Description
	Steps per revolution	In this field, configure the number of steps that the encoder resolves per revolution.
	Fine resolution - Bits in incr. actual value (Gn_XIST1)	In this field, configure the number of bits for fine resolution within the incremental actual value (Gn_XIST1).
Rotary absolute		
	Steps per revolution	In this field, configure the number of steps that the encoder resolves per revolution.
	Number of revolutions	In this field, configure the number of revolutions that the absolute value encoder can detect.
	Fine resolution - Bits in incr. actual value (Gn_XIST1)	In this field, configure the number of bits for fine resolution within the incremental actual value (Gn_XIST1).
	Fine resolution - Bits in abs. actual value (Gn_XIST2)	In this field, configure the number of reserved bits for the multiplication factor of the absolute value of the fine resolution (Gn_XIST2).

See also

Data connection PROFIdrive drive / PROFIdrive encoder (Page 6154)

Configuring technology modules for Motion Control (Page 6238)

Configuration - Encoder - Encoder on high-speed counter (HSC) (S7-1200)

Selection of high-speed counter (HSC)

In the high-speed counter box, select the high-speed counter to which the encoder transfers the actual value.

Check the filter times of the two high-speed counter digital inputs that are used. The filter times should be short enough to ensure reliable recording of the pulses.

HSC interface

Select the operating mode of the high-speed counter in the "Operating mode" box.

Depending on the operating mode, configure the various inputs:

Operating mode/parameter		Description
Two-phase		
	Clock generator forward	In this field, select the input for counting up. You can select the input using a symbolic address or assign it to an absolute address. The frequency and the location (on-board, signal board) of the input are displayed next to the address box.
	Clock generator backward	In this field, select the input for counting down. You can select the input using a symbolic address or assign it to an absolute address. The frequency and the location (on-board, signal board) of the input are displayed next to the address box.
A/B counter / A/B counter quadruple		
	Clock generator A	In this field, select the input for Phase A signals. You can select the input using a symbolic address or assign it to an absolute address. The frequency and the location (on-board, signal board) of the input are displayed next to the address box.
	Clock generator B	In this field, select the input for Phase B signals. You can select the input using a symbolic address or assign it to an absolute address. The frequency and the location (on-board, signal board) of the input are displayed next to the address box.

Invert encoder direction

To invert the actual value of the encoder, select the check box.

Automatic transfer of encoder parameters in the device

This selection is not possible when using encoders on the high-speed counter (HSC).

Encoder type

Select the encoder type in the "Encoder type" box. The following encoder types can be selected:

- **Linear incremental**
- **Rotary incremental**

Configure the various parameters depending on the selected encoder type. Depending on the selected encoder type, configure the following parameters:

Encoder type/parameter		Description
Linear incremental		
	Distance between two increments	In this field, you configure the distance between two steps of the encoder.
	Fine resolution - Bits in incr. actual value (Gn_XIST1)	In this field, configure the number of bits for fine resolution within the incremental actual value (Gn_XIST1).

Encoder type/parameter		Description
Rotary incremental		
	Steps per revolution	In this field, configure the number of steps that the encoder resolves per revolution.
	Fine resolution - Bits in incr. actual value (Gn_XIST1)	In this field, configure the number of bits for fine resolution within the incremental actual value (Gn_XIST1).

Extended parameters (S7-1200)

Mechanics (S7-1200)

Configuration - Mechanics - PTO (Pulse Train Output) (S7-1200)

Configure the mechanical properties of the drive in the "Mechanics" configuration window.

Pulses per motor revolution

Configure the number of pulses required for one revolution of the motor in this box.

Limits (independent of the selected unit of measurement):

- 0 < Pulse per motor revolution ≤ 2147483647

Load motion per motor revolution

In this box, configure the load distance per motor revolution covered by the mechanical system of your unit.

Limits (independent of the selected unit of measurement):

- 0.0 < Distance per revolution ≤ 1.0e12

Permitted direction of rotation (technology version as of V4)

Configure this box to determine whether the mechanics of your system are to move in both directions or only in the positive or negative direction.

If you have not activated the direction output of the pulse generator in the "PTO (pulse A and direction B)" mode, the selection is limited to the positive or negative direction.

Invert direction

You can use the "Invert direction" check box to adapt the control system to the direction logic of the drive.

The direction logic is inverted according to the selected mode of the pulse generator:

- **PTO (pulse A and direction B)**
  - 0 V at direction output ⇒ positive direction of rotation
  - 5 V/24 V at direction output ⇒ negative direction of rotation

  The specified voltage depends on the hardware used. The indicated values do not apply to the differential outputs of CPU 1217.
- **PTO (clock up A, clock down B)**
  The outputs "Pulse output down" and "Pulse output up" are swapped.
- **PTO (A/B phase-shifted)**
  The "Phase A" and "Phase B" outputs are swapped.
- **"PTO (A/B phase-shifted - quadruple)**
  The "Phase A" and "Phase B" outputs are swapped.

Configuration - Mechanics - PROFIdrive/analog drive connection (S7-1200)

Configure the mechanical properties of the drive and its encoder in the "Mechanics" configuration window.

Encoder mounting type

In the drop-down list, select how the encoder is mounted on the mechanism. The following encoder installation types are possible:

- **On motor shaft**
- **External measuring system** (rotary encoders only)

Position parameters

Depending on the selected encoder installation type, configure the following position parameters:

Encoder installation type/position parameter		Description
On the motor shaft		
	Load motion per motor revolution	In this field, configure the load distance for one motor revolution.
External measuring system		
	Load motion per motor revolution	In this field, configure the load distance for one motor revolution.
	Distance per encoder revolution	In this field, configure the distance recorded by the external measuring system per encoder revolution.

Configuration - Modulo (PROFIdrive/analog drive connection only) (S7-1200)

Use the "Modulo" setting if you want to limit the traversing range to a recurring distance based on the product length / product cycle. The modulo function is only possible in position-controlled operation of the axis.

When "modulo" is enabled, the position value of the technology object is represented by means of a recurring modulo range. The modulo range is defined by the start value and the length.

For example, to limit the position value of an axis to one full rotation, the modulo range can be defined with start value = 0° and length = 360°. With an encoder resolution of 0.1°/encoder step, the position value is represented in the modulo range 0.0° to 359.9°. If the axis in this example moves to the position 400°, the actual position 40° (400° - 360°) is reached.

The position of the axis is controlled with the Motion Control instruction "MC_MoveAbsolute" in the modulo function.

You determine the approach direction of the target position using the input parameter "Direction". The following parameter values are available:

- 0: The sign for the velocity ("Velocity" parameter) determines the motion direction.
- 1: Target position is approached in a positive direction.
- 2: Target position is approached in a negative direction.
- 3: Starting from the current position, the technology selects the shortest distance to the target position.

Enable modulo

Select the "Enable modulo" check box to use a recurring reference system for the axis (for example, 0.0° to 359.9°).

Modulo start value

In this field, define the position at which the modulo range should begin (for example, 0°).

Modulo length

In this field, define the length of the modulo range (for example, 360°).

See also

MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)

Position limits (S7-1200)

Requirements for hardware limit switches (S7-1200)

Use only hardware limit switches that remain permanently switched after being approached. This switching status may only be revoked after a return to the valid travel range.

See also

Configuration - Position limits (Page 6199)

Response of the axis when position limits are triggered (Page 6201)

Changing the configuration of the position limits in the user program (Page 6203)

Configuration - Position limits (S7-1200)

Configure the hardware and software limit switches of the axis in the "Position limits" configuration window.

Enable HW limit switches

Activate the function of the low and high hardware limit switch with this check box. The hardware limit switches can be used for purposes of direction reversal during a homing procedure. For details, refer to the configuration description for homing.

Enable SW limit switch

Activate the function of the low and high software limit switch with this check box.

Note

Activated software limit switches act only on a homed axis.

Input low / high HW limit switch

Select the digital input for the low or high hardware limit switch from the drop-down list.

The input should be interrupt-capable to guarantee short response times. However, the input of PTO axes must be interrupt-capable.

The digital onboard CPU inputs and the digital inputs of a plugged signal board can be selected as interrupt-capable inputs for the HW limit switches.

Note

The digital inputs are set to a filter time of 6.4 ms by default. If these are used as hardware limit switches, undesired decelerations may occur. If this occurs, reduce the filter time for the relevant digital inputs.

The filter time can be set under "Input filter" in the device configuration of the digital inputs.

Select level

In the drop-down list, select the signal level available at the CPU when the hardware limit switch is approached.

- Selection of "Low level" (normally closed contact)
  0 V (FALSE) at CPU input corresponds to hardware limit switch approached
- Selection of "High level" (normally open contact)
  5 V / 24 V (TRUE) at the CPU input = hardware limit switch approached (the actual voltage depends on the hardware used)

High and low software limit switch

Enter the position value of the low and high software limit switch in these boxes.

Limits (independent of the selected unit of measurement):

- -1.0e12 ≤ low software limit switch ≤ 1.0e12
- -1.0e12 ≤ high software limit switch ≤ 1.0e12

The value of the high software limit switch must be greater than or equal to the value of the low software limit switch.

See also

Requirements for hardware limit switches (Page 6198)

Response of the axis when position limits are triggered (Page 6201)

Changing the configuration of the position limits in the user program (Page 6203)

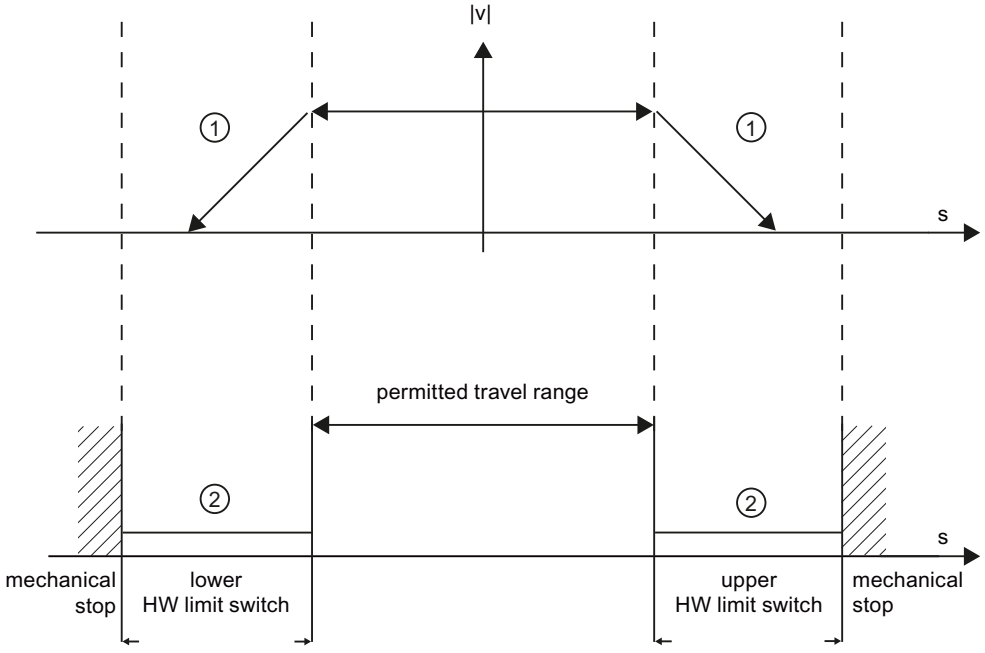
Configuration - Homing - Active (Page 6211)

Response of the axis when position limits are triggered (S7-1200)

Behavior of axis when hardware limit switches are approached

When a hardware limit switch is approached, the axis behaves differently depending on the drive connection:

- **Drive connection via PROFIdrive / analog output**
  When a hardware limit switch is approached, the axis is disabled and, depending on the configuration, braked at the drive and brought to a standstill. You must select the deceleration sufficiently large in the drive so that the axis stops reliably before the mechanical stop.
- **Drive connection via PTO (Pulse Train Output)**
  When the hardware limit switches are approached, the axis brakes to a standstill at the configured emergency deceleration. You must select the emergency deceleration sufficiently large so that the axis stops reliably before the mechanical stop. The following diagram presents the behavior of the axis after it approaches the hardware limit switches:



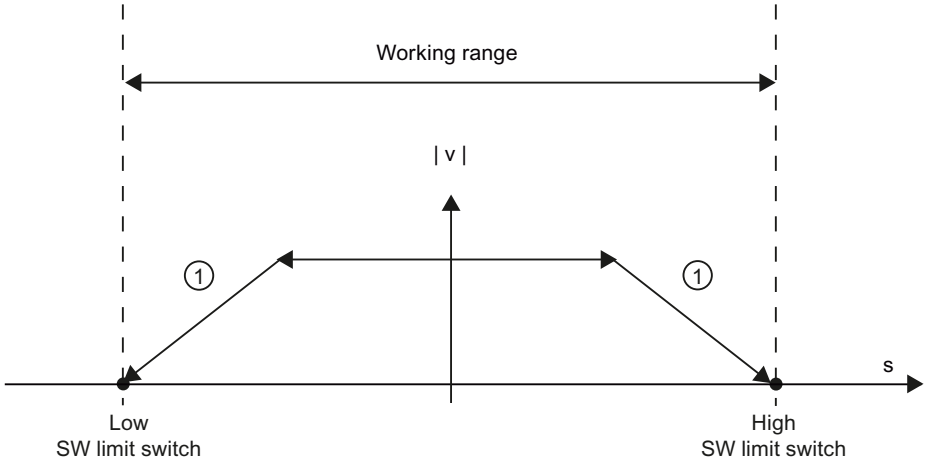
①	The axis brakes at the configured emergency stop deceleration (PTO) or the deceleration configured in the drive (PROFIdrive or analog drive interface) until standstill occurs.
②	Range in which the HW limit switches signal the status "approached".

The "HW limit switch approached" error is displayed in the initiating Motion Control instruction, at "MC_Power", and in the technology object variables. Instructions for eliminating errors can be found in the Appendix under "List of ErrorIDs and ErrorInfos".

Behavior of axis when software limit switches are reached

If software limit switches are activated, an active motion is stopped at the position of the software limit switch. The axis is braked at the configured deceleration.

The following diagram presents the behavior of the axis until it reaches the software limit switches:



①	The axis brakes to a standstill at the configured deceleration.

The "SW limit switch approached" error is displayed in the initiating Motion Control instruction, at "MC_Power", and in the technology object variables. Instructions for eliminating errors can be found in the Appendix under "List of ErrorIDs and ErrorInfos".

When a software limit switch is overtraveled, the axis behaves differently depending on the drive connection:

- **Drive connection via PROFIdrive / analog output**
  When a software limit switch is overtraveled, the axis is disabled and, depending on the configuration, braked at the drive and brought to a standstill.
- **Drive connection via PTO (Pulse Train Output)**
  You can learn about the behavior of the axis when a software limit switch is overtraveled in the sections "Software limit switches in conjunction with a homing operation (Page 6296)" and "Software limit switches in conjunction with dynamic changes (Page 6301)".

Use additional hardware limit switches if a mechanical endstop is located after the software limit switches and there is a risk of mechanical damage.

See also

Requirements for hardware limit switches (Page 6198)

Configuration - Position limits (Page 6199)

Changing the configuration of the position limits in the user program (Page 6203)

Changing the configuration of the position limits in the user program (S7-1200)

You can change the following configuration parameters during runtime of the user program in the CPU:

Hardware limit switches

You can also activate and deactivate the hardware limit switches during runtime of the user program. Use the following technology object variable for this purpose:

- <axis name>.PositionLimits_HW.Active

Refer to the description of the technology object variables (Page 6328) in the appendix for information on when changes to the configuration parameter take effect.

Software limit switches

You can also activate and deactivate the software limit switches and change their position values during runtime of the user program. Use the following technology object variables for this purpose:

- <axis name>.PositionLimits_SW.Active
  for activating and deactivating the software limit switches
- <axis name>.PositionLimits_SW.MinPosition
  for changing the position of the low software limit switch
- <axis name>.PositionLimits_SW.MaxPosition
  for changing the position of the high software limit switch

Refer to the description of the technology object variables (Page 6328) in the appendix for information on when changes to the configuration parameters take effect.

See also

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)

Requirements for hardware limit switches (Page 6198)

Configuration - Position limits (Page 6199)

Response of the axis when position limits are triggered (Page 6201)

Dynamics (S7-1200)

Configuration - Dynamics - General (S7-1200)

Configure the maximum velocity, the start/stop velocity, the acceleration and deceleration, and the jerk limit (positioning axis technology object as of V2) of the axis in the "General dynamics" configuration window.

Unit of velocity limits

Select the unit of measurement with which you want to set the velocity limits in the drop-down list. The unit set here depends on the unit of measurement set under "Configuration - General" and serves only for easier input. This provides the possibility, for example, to enter the maximum speed as a speed value of the motor in 1/min.

Maximum velocity / Start/stop velocity

Define the maximum permissible velocity and the start/stop velocity of the axis in these boxes. The start/stop velocity is the minimum permissible velocity of the axis and can only be configured for drive connection via PTO (Pulse Train Output).

For drive connection via PROFIdrive or analog output, the start/stop velocity is fixed at zero. The maximum velocity when connecting via PROFdrive or analog output amounts to $1e^{+12}$ of the selected measurement unit (e.g. mm/s, °/s, ...).

Limits:

The limits indicated below refer to the "Pulses/s" unit of measurement:

Positioning axis technology object (PTO) as of V4

- 1 ≤ start/stop velocity ≤ 20000 (signal board 20 kHz)
  1 ≤ start/stop velocity ≤ 200000 (signal board 200 kHz)
  1 ≤ start/stop velocity ≤ 100000 (onboard CPU outputs 100 kHz)
  1 ≤ start/stop velocity ≤ 20000 (on-board CPU outputs 20 kHz)
  1 ≤ start/stop velocity ≤ 1000000 (onboard CPU outputs 1 MHz CPU 1217)
- 1 ≤ maximum velocity ≤ 20000 (signal board 20 kHz)
  1 ≤ maximum velocity ≤ 200000 (signal board 200 kHz)
  1 ≤ maximum velocity ≤ 100000 (onboard CPU outputs 100 kHz)
  1 ≤ maximum velocity ≤ 20000 (on-board CPU outputs 20 kHz)
  1 ≤ maximum velocity ≤ 1000000 (onboard CPU outputs 1 MHz CPU 1217)

You can learn about the limits for the technology object positioning axis < V4 in the appendix Outputs of the CPU relevant for Motion Control (technology version V1...3) (Page 6364).

The value of the maximum velocity must be greater or equal to the value of the start/stop velocity.

The limits for other units of measurement must be converted by the user to conform to the given mechanics.

Acceleration / Deceleration - Ramp-up time / Ramp-down time

Set the desired acceleration in the "Ramp-up time" or "Acceleration" boxes. The desired deceleration can be set in the "Ramp-down time" or "Deceleration" boxes.

The relation between the ramp-up time and acceleration and the ramp-down time and deceleration is shown in the following equations:

$$\textit{Ramp-up time} = \frac{\textit{Maximum velocity} - \textit{Start/stop velocity}}{\textit{Acceleration}}$$

$$\text{Ramp-down time} = \frac{\text{Maximum velocity} - \text{Start/stop velocity}}{\text{Deceleration}}$$

Motion jobs started in the user program are performed with the selected acceleration / deceleration.

The limits for acceleration and deceleration with drive connection via PTO (Pulse Train Output) can be found in section CPU outputs relevant for motion control (Page 6143).

Note

Changes to the velocity limits ("start/stop velocity" and "maximum velocity") influence the acceleration and deceleration values of the axis. The ramp-up and ramp-down times are retained.

Enable jerk limit (positioning axis technology object as of V2)

Enable the jerk limit with this check box.

Axis acceleration and deceleration is not stopped abruptly when the jerk limit is activated; it is adjusted gently according to the set step or smoothing time.

Note

In coordination with S7-1500 Motion Control, the check box is no longer displayed as a parameter in the technology data block as of V4. By disabling the check box, the jerk value is set to 0.0.

Smoothing time / jerk (positioning axis technology object as of V2)

You can input the parameters of the jerk limit in the "Smoothing time" box or alternatively in the "Jerk" box.

- Set the desired jerk for acceleration and deceleration ramp in the "Jerk" box.
- Set the desired smoothing time for the acceleration ramp in the "Smoothing time" box.

Note

Smoothing time V2...3

The set smoothing time visible in the configuration only applies to the acceleration ramp.

If the values for acceleration and deceleration differ, the smoothing time of the deceleration ramp is calculated according to the jerk of the acceleration ramp and used. (See also Behavior of the axis when using the jerk limit (Page 6208)

The smoothing time of the deceleration is adapted as follows:

- **Acceleration > deceleration**
  The smoothing time used for the deceleration ramp is shorter than that for the acceleration ramp.
- **Acceleration < deceleration**
  The smoothing time used for the deceleration ramp is greater than that for the acceleration ramp.
- **Acceleration = deceleration**
  The smoothing times of the acceleration and deceleration ramp are equal.

The relation between smoothing times and jerk is shown in the following equation:

$$\text{Smoothing time (acceleration ramp)} = \frac{\text{Acceleration}}{\text{Jerk}}$$

$$\text{Smoothing time (deceleration ramp)} = \frac{\text{Deceleration}}{\text{Jerk}}$$

Motion jobs started in the user program are performed with the selected jerk.

The limits for jerk with drive connection via PTO (Pulse Train Output) can be found in section CPU outputs relevant for motion control (Page 6143).

For PROFIdrive drives and drives with analog drive interface, the limit is $1E^{+12}$.

See also

Behavior of the axis when using the jerk limit (Page 6208)

Hardware components for motion control (Page 6141)

CPU outputs relevant for motion control (Page 6143)

Configuration - Dynamics - Emergency stop (Page 6207)

Changing the configuration of dynamics in the user program (Page 6209)

Configuration - Dynamics - Emergency stop (S7-1200)

Configure the emergency stop deceleration of the axis in the "Dynamics emergency stop" configuration window. In the event of an error, and when disabling the axis, the axis is brought to a standstill with this deceleration using the Motion Control instruction "MC_Power" (input parameter StopMode = 0 or 2).

Velocity

The velocity values configured in the "General dynamics" configuration window are once again displayed in this information area.

Deceleration

Set the deceleration value for emergency stop in the "Emergency deceleration" or "Emergency stop ramp-down time" field.

The relation between emergency stop ramp-down time and emergency deceleration is shown in the following equation:

$$\textit{Emergency stop ramp-down time} = \frac{\textit{Maximum velocity - Start/stop velocity}}{\textit{Emergency deceleration}}$$

The specified emergency deceleration must be sufficient to bring the axis to a standstill in a timely manner in the event of an emergency (for example, when the hardware limit switch is approached prior to reaching the mechanical endstop).

The configured maximum velocity of the axis must be used as a basis for selecting the emergency deceleration.

Limit values:

The limits indicated below refer to the "Pulses/s^2" unit of measurement.

- As of CPU firmware V3
  0.005 ≤ emergency stop deceleration ≤ 9.5E^9
- CPU Firmware V1...2
  0.28 ≤ emergency stop deceleration ≤ 9.5E^9

The limits for other units of measurement must be converted to conform to the given mechanics.

The limits for jerk with drive connection via PTO (Pulse Train Output) can be found in section CPU outputs relevant for motion control (Page 6143).

For PROFIdrive drives and drives with analog drive interface, the limit is 1E^{+12}.

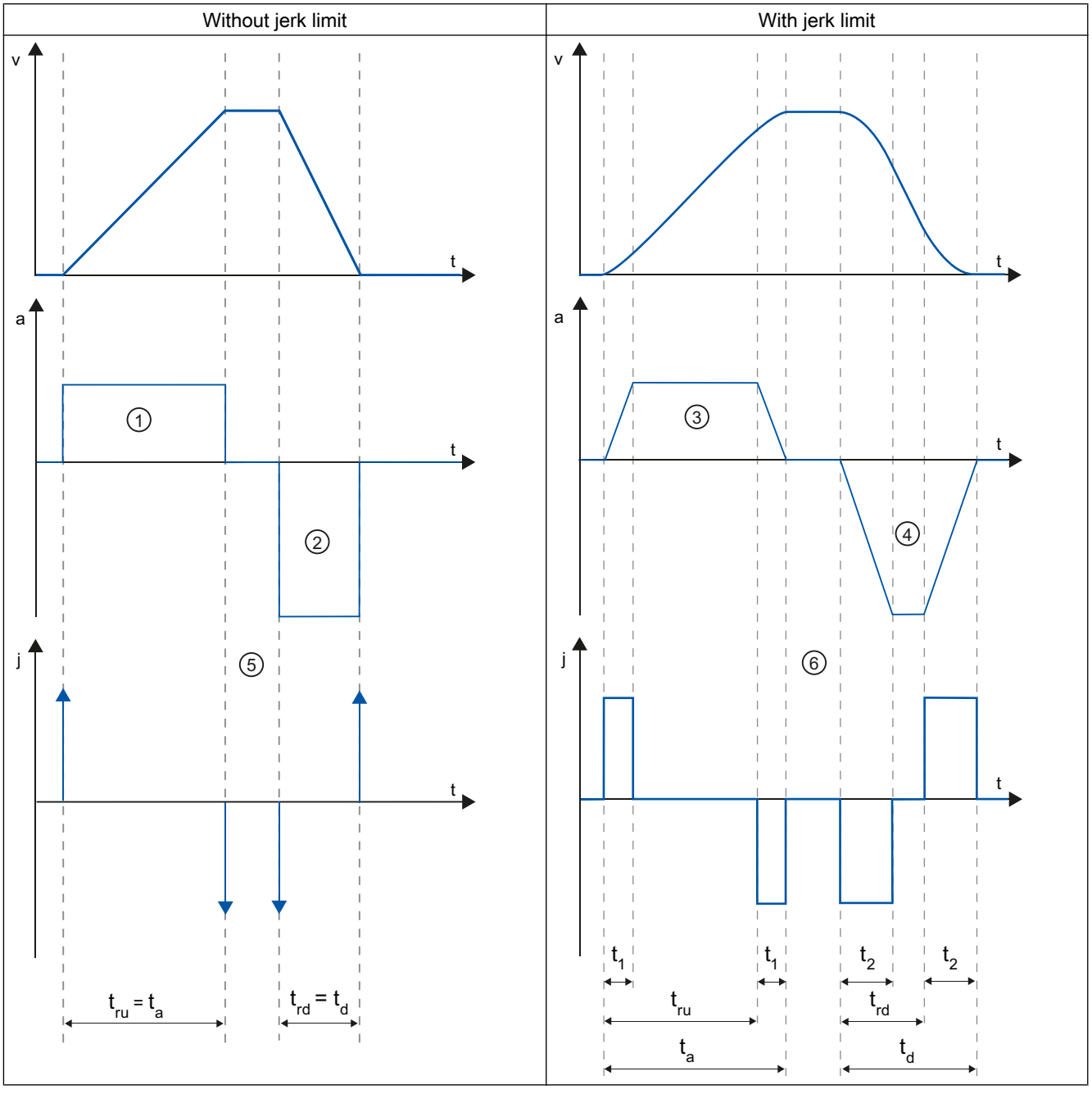
See also

Configuration - Dynamics - General (Page 6203)

Changing the configuration of dynamics in the user program (Page 6209)

Behavior of the axis when using the jerk limit (S7-1200)

Axis acceleration and deceleration is not stopped abruptly when the jerk limit is activated; it is adjusted gently according to the set step or smoothing time. The diagram below details the behavior of the axis with and without activated jerk limit:



t	Time axis
v	Velocity
a	Acceleration
j	Jerk
t_{ru}	Rampup time
t_a	Time taken for the axis to accelerate
t_{rd}	Deceleration time
t_d	Time taken for the axis to decelerate
t_1	Smoothing time of the acceleration ramp
t_2	Smoothing time of the deceleration ramp

The example shows travel in which the deceleration value ② is twice the acceleration value ①. The resulting ramp-down time t_{rd} is therefore only half the length of the ramp-up time t_{ru}.

Acceleration ① and deceleration ② change abruptly without a jerk limit. Acceleration ① and deceleration ② change gradually with activated jerk limiter. As the jerk applies to entire motion, the rate is the same for the increase and decrease in acceleration and deceleration.

The step value j becomes infinitely high ⑤ as soon as the change is made without jerk limit. The step is limited to the configured value ⑥ when the jerk limit is activated.

The smoothing time t_1 given in the configuration applies to the acceleration ramp. The deceleration ramp smoothing time t_2 is calculated from the configured jerk value and the configured deceleration.

See also

Configuration - Dynamics - General (Page 6203)

Changing the configuration of dynamics in the user program (S7-1200)

You can change the following configuration parameters during runtime of the user program in the CPU:

Acceleration and deceleration

You can also change the values for acceleration and deceleration during runtime of the user program. Use the following technology object variables for this purpose:

- <Axis name>.DynamicDefaults.Acceleration
  for changing acceleration
- <Axis name>.DynamicDefaults.Deceleration
  for changing deceleration

Refer to the description of the technology object variables (Page 6328) in the appendix for information on when changes to the configuration parameters take effect.

Emergency stop deceleration

You can also change the value for the emergency stop deceleration during runtime of the user program. Use the following technology object variable for this purpose:

- <Axis name>.DynamicDefaults.EmergencyDeceleration

Refer to the description of the technology object variables in the appendix for information on when changes to the configuration parameter take effect.

Note

After changes to this parameter, it may be necessary to adapt the positions of the hardware limit switches and other safety-relevant settings.

Jerk limit

You can also activate and deactivate the jerk limit at runtime of the user program and change the value for the jerk. For this, use the technology object tag <axis name>.DynamicDefaults.Jerk. For technology objects < V4, the tag <axis name>.Config.DynamicDefaults.JerkActive must be set to FALSE to change the values.

The following applies to PTO axes:

- If you enter a value ≥ 0.004 mm/s^3 for the jerk, the jerk limit is enabled with the specified value.
- If you enter a value < 0.004mm/s^3 for the jerk, the jerk limit is disabled.

For position-controlled axes, the jerk limit is disabled for a value of 0.0, and activated for values > $1e^{-9}$.

Refer to the description of the technology object variables in the appendix for information on when changes to the configuration parameter take effect.

See also

Changing configuration of dynamic values in user program ("Axis" technology object V1...3) (Page 6375)

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

Tags of the positioning axis technology object V4...5 (Page 6436)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)

Configuration - Dynamics - General (Page 6203)

Configuration - Dynamics - Emergency stop (Page 6207)

Homing (positioning axis technology object as of V2) (S7-1200)

Configuration - Homing - Active (S7-1200)

Configure the necessary parameters for active homing in the "Active homing" configuration window. Active homing is started using Motion Control instruction "MC_Home" with input parameter "Mode" = 3.

Select homing mode (drive connection via PROFIdrive V5 or higher only)

Select one of the following homing modes:

- Use zero mark via PROFIdrive telegram and proximity switch
- Use zero mark via PROFIdrive telegram
- Use homing mark via digital input

If you have selected drive connection via PTO (Pulse Train Output) or analog output with HSC as the encoder, the only homing mode available is "Use homing mark via digital input".

Digital inputs

In this area, you configure the homing switch:

- **Input homing switch**
  Select the digital input for the homing switch in this field.

  Note

  The digital inputs are set to a filter time of 6.4 ms by default.

  When the digital inputs are used as a homing switch, this can result in undesired decelerations and thus inaccuracies. Depending on the homing velocity and extent of the homing switch, the home position may not be detected. The filter time can be set under "Input filter" in the device configuration of the digital inputs.

  The specified filter time must be less than the duration of the input signal at the homing switch.

  For drive connection via PTO (Pulse Train Output):
  The input must be interrupt-capable. The onboard CPU inputs and the inputs of an inserted signal board can be selected as inputs for the homing switch.

- **Select level**
  In the drop-down list, select the level of the homing switch that is to be used for homing.

- **Permit auto reverse at HW limit switch**
  Activate the check box to use the hardware limit switch as a reversing cam for the homing procedure. The hardware limit switches must be enabled for the reversal of direction (at least the hardware limit switch in the direction of approach must be configured).
  If the hardware limit switch is reached during active homing, the axis brakes at the configured deceleration (not with the emergency stop deceleration) and reverses direction. The homing switch is then sensed in reverse direction.
  If the direction reversal is not active and the axis reaches the hardware limit switch during active homing, the homing procedure is aborted with an error and the axis is braked at the emergency stop deceleration.

  Note

  If possible, use one of the following measures to ensure that the machine does not travel to a mechanical endstop in the event of a direction reversal:

  - Keep the approach velocity low.
  - Increase the configured acceleration/deceleration.
  - Increase the distance between the hardware limit switch and the mechanical endstop.

Approach/homing direction

With the direction selection, you determine the approach direction used during active homing to search for the homing switch, as well as the homing direction. The homing direction specifies the travel direction the axis uses to approach the configured end of the homing switch to carry out the homing operation.

Side of homing switch

This is where you select whether the axis is to be homed on the top or bottom side of the homing switch.

Approach velocity

In this field, specify the velocity at which the homing switch is to be searched for during the homing procedure.

Limits (independent of the selected unit of measurement):

- Start/stop velocity ≤ approach velocity ≤ maximum velocity

Homing velocity

In this field, specify the velocity at which the homing switch is to be approached for homing.

Limits (independent of the selected unit of measurement):

- Start/stop velocity ≤ Homing velocity ≤ Maximum velocity

Home position offset

If the desired home position deviates from the position of the homing switch, the home position offset can be specified in this field.

If the value does not equal 0, the axis executes the following actions following homing at the homing switch:

1. Move the axis at the homing velocity by the value of the home position offset
2. Upon reaching the "home position offset", the axis is at the home position that was specified in input parameter "Position" of the "MC_Home" Motion Control instruction.

Limits (independent of the selected unit of measurement):

- -1.0e12 ≤ home position offset ≤ 1.0e12

Home position

The position configured in the Motion Control instruction "MC_Home" is used as the home position.

Configuration - Homing - Passive (S7-1200)

Configure the necessary parameters for passive homing in the "Homing - Passive" configuration window.

The movement for passive homing must be triggered by the user (e.g. using an axis motion command). Passive homing is started using Motion Control instruction "MC_Home" with input parameter "Mode" = 2.

Select homing mode (drive connection via PROFIdrive V5 or higher only)

Select one of the following homing modes:

- **Use zero mark via PROFIdrive telegram and proximity switch**
  The system checks for when the proximity switch is reached. After the proximity switch is reached and is left again in the assigned homing direction, zero mark detection is enabled via the PROFIdrive telegram. When the zero mark is reached in the pre-selected direction, then the actual position of the technology object is set to the homing mark position.
- **Use zero mark via PROFIdrive telegram**
  The system enables zero mark detection as soon as the actual value of the technology object moves in the assigned homing direction. When the zero mark is reached in the specified homing direction, the actual position of the technology object is set to the homing mark position.
- **Use homing mark via digital input**
  The system checks the state of the digital input as soon as the actual value of the axis or encoder moves in the assigned homing direction. When the homing mark is reached (setting of the digital input) in the specified homing direction, the actual position of the technology object is set to the homing mark position.

If you have selected drive connection via PTO (Pulse Train Output), a homing mark via a digital input is used by default.

Digital inputs

In this area, you configure the homing switch:

- **Input homing switch**
  Select the digital input for the homing switch in this field. The input must be interrupt-capable. The onboard CPU inputs and the inputs of an inserted signal board can be selected as inputs for the homing switch.

  Note

  The digital inputs are set to a filter time of 6.4 ms by default.

  When the digital inputs are used as a homing switch, this can result in undesired decelerations and thus inaccuracies. Depending on the homing velocity and extent of the homing switch, the home position may not be detected. The filter time can be set under "Input filter" in the device configuration of the digital inputs.

  The specified filter time must be less than the duration of the input signal at the homing switch.

- **Select level**
  In the drop-down list, select the level of the homing switch that is to be used for homing.

Side of homing switch

This is where you select whether the axis is to be homed on the top or bottom side of the homing switch.

Home position

The position configured in the Motion Control instruction "MC_Home" is used as the home position.

Note

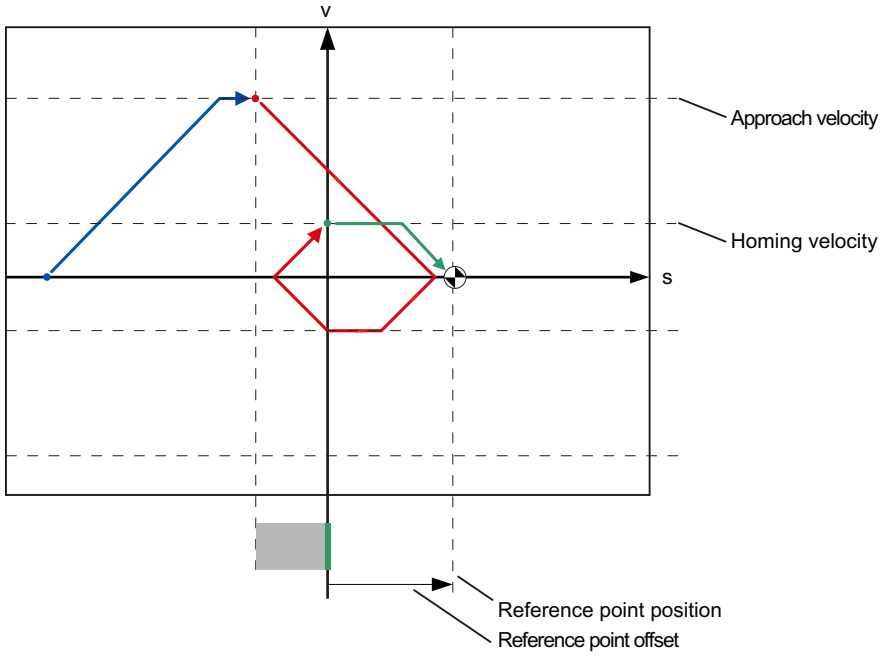
If passive homing is carried out without an axis motion command (axis at a standstill), homing will be executed upon the next rising or falling edge at the homing switch.

Sequence - Active homing (S7-1200)

You start active homing with motion control instruction "MC_Home" (input parameter Mode = 3). The "Position" input parameter specifies the absolute home position. Alternatively, you can start active homing on the axis control panel for test purposes.

The diagram below shows an example of a characteristic curve for an active home position approach with the following configuration parameters:

- "Homing mode" = "Use homing mark via digital input"
- "Approach/homing direction" = "Positive direction"
- "Side of homing switch" = "Top side"
- Value of "home position offset" > 0



Search for homing switch (blue curve section)

When active homing starts, the axis accelerates to the configured "approach velocity" and searches at this velocity for the homing switch. The tag <axis name>.StatusBits.HomingDone is set to FALSE.

Reference point approach (red curve section)

When the homing switch is detected, the axis in this example brakes and reverses, to be homed to the configured side of the homing switch at the configured homing velocity. Homing causes the tag <axis name>.StatusBits.HomingDone to change to TRUE.

Travel to home position offset (green curve segment)

After homing, the axis moves at the homing velocity along the path to the home position offset. There the axis is at the homing point position that was specified in input parameter "Position" of the "MC_Home" Motion Control instruction.

See also

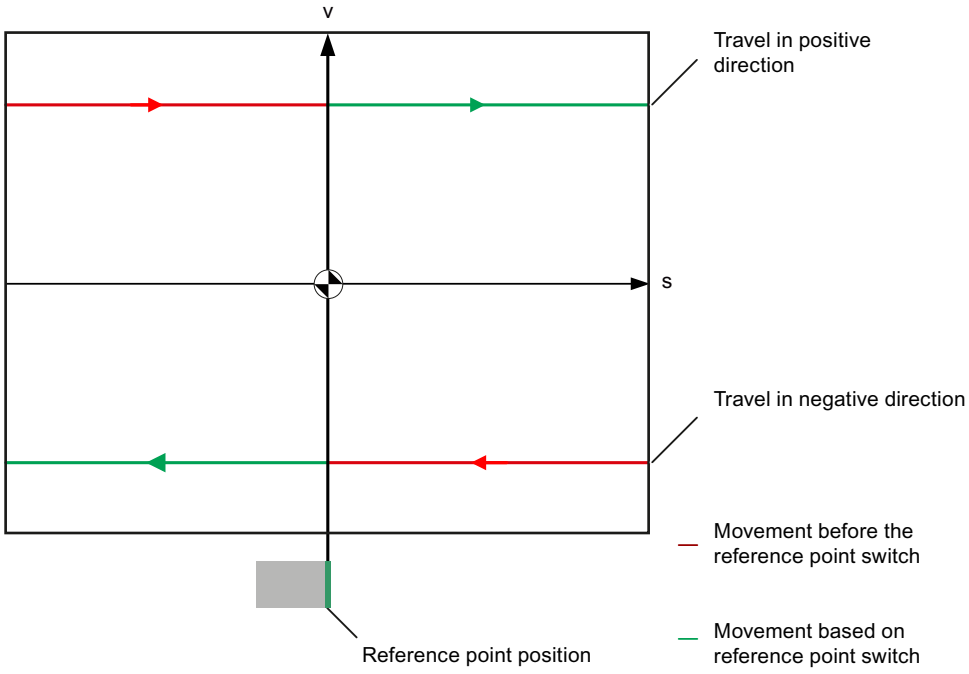
Configuration - Homing - General (Axis technology object V2...3) (Page 6373)

Sequence - Passive homing (S7-1200)

Passive homing is started with Motion Control instruction "MC_Home" (input parameter Mode = 2). Input parameter "Position" specifies the absolute reference point position.

The diagram below shows an example of a characteristic curve for passive homing with the following configuration parameters:

- "Side of homing switch" = "Top side"
- "Homing mode" = "Use homing mark via digital input"



Movement towards homing switch (red section of curve)

The Motion Control instruction "MC_Home" does not itself carry out any homing motion when passive homing is started. The travel required for reaching the homing switch must be implemented by the user via other motion control instructions such as "MC_MoveRelative". The tag <axis name>.StatusBits.HomingDone remains TRUE during passive homing if the axis has already been homed.

Axis homing (transition from red to green section of curve)

The axis is homed when the configured side of the homing switch is reached. The current position of the axis is set to the home position. This is specified at the "Position" parameter of the "MC_Home" Motion Control instruction. The tag <axis name>.StatusBits.HomingDone will be set to "TRUE" if the axis has not been homed before. The travel previously started is not canceled.

Movement beyond homing switch (green section of curve)

Following homing at the homing switch, the axis continues and completes the previously started travel with the corrected axis position.

Changing the homing configuration in the user program (S7-1200)

With positioning axis technology object as of V2, you can change the following configuration parameters during runtime of the user program in the CPU:

Passive homing

You can change the end of the homing switch for passive homing during the user program runtime. Use the following technology object tag for this purpose:

- <Axis name>.Sensor[1].PassiveHoming.SideInput
  for changing the side of the homing switch
- <Axis name>.Sensor[1].PassiveHoming.Mode
  for changing the homing mode

Refer to the description of the technology object tags (Page 6436) in the appendix for information on when changes to the configuration parameter take effect.

Active homing

You can change the direction of approach, the side of the homing switch, the approach velocity, the homing velocity, and the home position offset for active homing during the program runtime of the user program. Use the following technology object tags for this purpose:

- <Axis name>.Homing.AutoReversal
  for changing the auto reverse at the HW limit switch
- <Axis name>.Homing.ApproachDirection
  for changing the approach/homing direction

- <Axis name>.Sensor[1].ActiveHoming.SideInput
  for changing the side of the homing switch
- <Axis name>.Homing.ApproachVelocity
  for changing the approach velocity
- <Axis name>.Homing.ReferencingVelocity
  for changing the homing velocity
- <Axis name>.Sensor[1].ActiveHoming.HomePositionOffset
  for changing the home position offset
- <Axis name>.Sensor[1].ActiveHoming.Mode
  for changing the homing mode

Refer to the description of the technology object tags in the appendix for information on when changes to the configuration parameter take effect.

See also

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

MC_ChangeDynamic: Change dynamic settings of axis V4...5 (Page 4159)

Positioning monitoring (S7-1200)

Configuration - Position monitoring (PROFIdrive and analog drive connection only) (S7-1200)

In the "Position monitoring" configuration window, configure the criteria for monitoring the target position.

Position monitoring monitors the behavior of the actual position at the end of the setpoint calculation. As soon as the setpoint velocity reaches the value zero, the actual position value must be located within a tolerance time in the positioning window. The actual value must not exit the positioning window during the minimum dwell time.

If the actual position reaches the positioning window within the tolerance time and remains in the positioning window for the minimum dwell time, the status bit <axis name>.StatusBits.Done is set. This completes a motion command.

Position monitoring does not make any distinction between how the setpoint interpolation was completed. The end of setpoint interpolation can, for example, be reached as follows:

- By the setpoint reaching the target position
- By position-controlled stopping during the motion through the Motion Control instruction "MC_Halt"

In the following cases, the axis is stopped by the position monitoring and a positioning error (ErrorID 16#800F) is displayed at the Motion Control instruction:

- The actual value does not reach the positioning window within the tolerance time.
- The actual value exits the positioning window during the minimum dwell time.

Positioning window

In this field, configure the size of the positioning window.

Tolerance time

In this field, configure the tolerance time within which the position value must reach the positioning window.

Minimum dwell time in positioning window

In this field, configure the minimum dwell time for which the actual position value must be located in the positioning window.

Configuration - Following error (PROFIdrive and analog drive connection only) (S7-1200)

In the "Following error" configuration window, you configure the permissible deviation of the actual position of the axis from the setpoint position.

The following error is the difference between the setpoint position and the actual position value of the axis. The transmission times of the setpoint to the drive and of the actual value to the controller are taken into account in the calculation of the following error.

The following error is monitored based on a velocity-dependent following error limit. The permissible following error depends on the setpoint velocity.

A constant permissible following error can be specified for velocities lower than an adjustable velocity low limit. Above this low velocity limit, the permissible following error increases in proportion to the setpoint velocity. The maximum following error is permitted at the maximum velocity.

If the permitted following error is exceeded, the axis is stopped and an error (ErrorID 16#800D) is displayed at the Motion Control instruction.

Enable following error monitoring

Select the check box to enable following error monitoring.

When following error monitoring is enabled, the axis is stopped in the error range (orange).

Maximum following error

In this field, configure the following error that is permissible at maximum velocity.

Following error

In this field, configure the permissible following error for low velocities (without dynamic adaptation).

Start dynamic adjustment

In this field, configure the velocity above which the following error should be dynamically adapted. Above this velocity, the following error up to the maximum velocity will be adapted to the maximum following error.

Maximum velocity

This box shows the maximum velocity configured under "Dynamics > General".

Configuration - Standstill signal (PROFIdrive and analog drive connection only) (S7-1200)

In the "Standstill signal" configuration window, configure the criteria for standstill detection.

To display the standstill (<Axis name>.StatusBits.StandStill), the velocity of the axis must remain in the standstill window for the minimum dwell time.

Standstill window

In this field, configure the size of the standstill window.

Minimum dwell time in standstill window

In this field, configure the minimum dwell time in the standstill window.

Configuration - Control loop (PROFIdrive and analog drive connection only) (S7-1200)

In the "Control loop" configuration window, configure the precontrol and the gain Kv of the position control loop.

The Kv factor affects the following parameters:

- Positioning accuracy and stop control
- Uniformity of motion
- Positioning time

The better the mechanical conditions of the axis are (high stiffness), the higher you can configure the Kv factor. This reduces the following error, and a higher dynamic response is achieved.

The "Tuning (Page 6264)" function supports you in determining the optimum gain for the position control of the axis.

Precontrol

In this field, configure the velocity precontrol of the position control loop as a percentage.

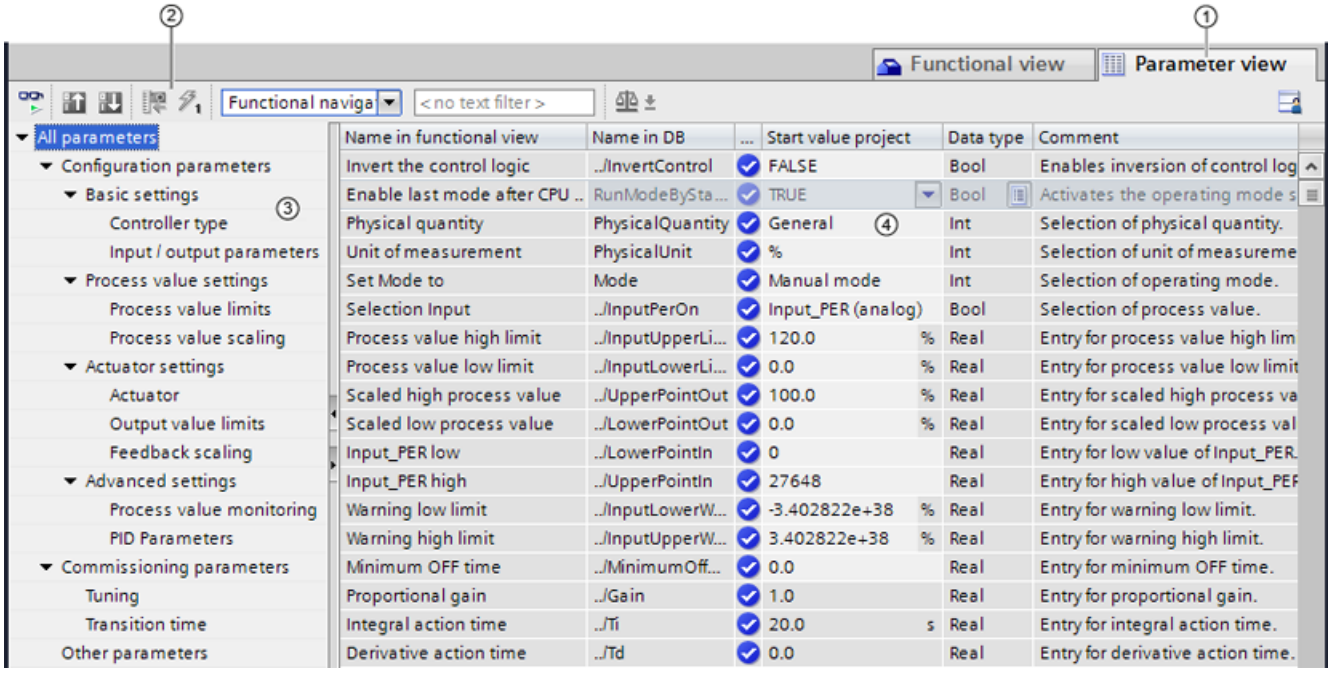
Gain (Kv factor)

In this field, you configure the gain Kv of the position control loop.

Parameter view (S7-1200)

Introduction to the parameter view (S7-1200)

The Parameter view provides you with a general overview of all relevant parameters of a technology object. You obtain an overview of the parameter settings and can easily change them in offline and online mode.



① "Parameter view" tab

② Toolbar (Page 6007)

③ Navigation (Page 6007)

④ Parameter table (Page 6008)

Function scope

The following functions are available for analyzing the parameters of the technology objects and for enabling targeted monitoring and modification.

Display functions:

- Display of parameter values in offline and online mode
- Display of status information of the parameters
- Display of value deviations and option for direct correction
- Display of configuration errors
- Display of value changes as a result of parameter dependencies

- Display of all memory values of a parameter: Start value PLC, Start value project, Monitor value
- Display of the parameter comparison of the memory values of a parameter

Operator control functions:

- Navigation for quickly changing between the parameters and parameter structures.
- Text filter for faster searches for particular parameters.
- Sorting function for customizing the order of parameters and parameter groups to requirements.
- Memory function for backing up structural settings of the Parameter view.
- Monitoring and modifying of parameter values online.
- Function for saving a snapshot of parameter values of the CPU in order to capture momentary situations and to respond to them.
- Function for applying a snapshot of parameter values as start values.
- Download of modified start values to the CPU.
- Comparison functions for comparing parameter values with one another.

Validity

The Parameter view described here is available for the following technology objects:

- PID_Compact
- PID_3Step
- PID_Temp
- CONT_C (S7-1500 only)
- CONT_S (S7-1500 only)
- TCONT_CP (S7-1500 only)
- TCONT_S (S7-1500 only)
- TO_Axis_PTO (S7-1200 Motion Control)
- TO_Positioning_Axis (S7-1200 Motion Control)
- TO_CommandTable_PTO (S7-1200 Motion Control)
- TO_CommandTable (S7-1200 Motion Control)

Structure of the parameter view (S7-1200)

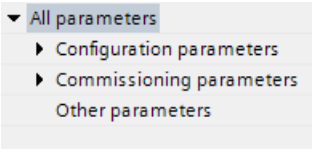
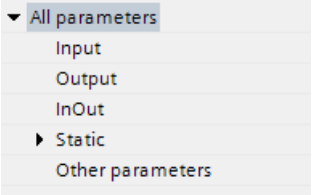
Toolbar (S7-1200)

The following functions can be selected in the toolbar of the parameter view.

Icon	Function	Explanation
	Monitor all	Starts the monitoring of visible parameters in the active Parameter view (online mode).
	Create snapshot of monitor values and accept setpoints of this snapshot as start values	Applies the current monitor values to the “Snapshot” column and updates the start values in the project. Only in online mode for PID_Compact, PID_3Step and PID_Temp.
	Initialize setpoints	Transfers the start values updated in the project to the CPU. Only in online mode for PID_Compact, PID_3Step and PID_Temp.
	Create snapshot of monitor values	Applies the current monitor values to the “Snapshot” column. Only in online mode.
	Modify all selected parameters immediately and once	This command is executed once and as quickly as possible without reference to any particular point in the user program. Only in online mode.
	Select navigation structure	Toggles between functional navigation and data navigation.
	Text filter...	After entry of a character string: Display of all parameters containing the specified string in one of the currently visible columns.
	Selection of compare values	Selection of parameter values that are to be compared with one another in online mode (Start value project, Start value PLC, Snapshot) Only in online mode.
	Save window settings	Saves your display settings for the Parameter view (e.g., selected navigation structure, activated table columns, etc.)

Navigation (S7-1200)

Within the "Parameter view" tab, the following alternative navigation structures can be selected.

Navigation		Explanation
Functional navigation		In the functional navigation, the structure of the parameters is based on the structure in the configuration dialog ("Functional view" tab), commissioning dialog, and diagnostics dialog. The last group "Other parameters" contains all other parameters of the technology object.
Data navigation		In the data navigation, the structure of the parameters is based on the structure in the instance DB / technology DB. The last group "Other parameters" contains the parameters that are not contained in the instance DB / technology DB.

You can use the "Select navigation structure" drop-down list to toggle the navigation structure.

Parameter table (S7-1200)

The table below shows the meaning of the individual columns of the parameter table. You can show or hide the columns as required.

- Column "Offline" = X: Column is visible in offline mode.
- Column "Online" = X: Column is visible in online mode (online connection to the CPU).

Column	Explanation	Offline	Online
Name in functional view	Name of the parameter in the functional view. The display field is empty for parameters that are not configured via the technology object.	X	X
Full name in DB	Complete path of the parameter in the instance DB / technology DB. The display field is empty for parameters that are not contained in the instance DB / technology DB.	X	X
Name in DB	Name of the parameter in the instance DB / technology DB. If the parameter is part of a structure or UDT, the prefix ". ./" is added. The display field is empty for parameters that are not contained in the instance DB / technology DB.	X	X
Status of configuration	Display of the completeness of the configuration using status symbols. see Status of configuration (offline) (Page 6016)	X	
Compare result	Result of the "Compare values" function. This column is shown if there is an online connection and the "Monitor all" button  is selected.		X
Start value project	Configured start value in the project. Error indication if entered values have a syntax or process-related error.	X	X
Default value	Value that is pre-assigned to the parameter. The display field is empty for parameters that are not contained in the instance DB / technology DB.	X	X
Snapshot	Snapshot of the current values in the CPU (monitor values). Error indication if values have a process-related error.	X	X
Start value PLC	Start value in the CPU. This column is shown if there is an online connection and the "Monitor all" button  is selected. Error indication if values have a process-related error.		X
Monitor value	Current value in the CPU. This column is shown if there is an online connection and the "Monitor all" button  is selected. Error indication if values have a process-related error.		X
Modify value	Value that is to be used to change the monitor valuet. This column is shown if there is an online connection and the "Monitor all" button  is selected. Error indication if entered values have a syntax or process-related error.		X
Selection for transmission 	Selection of the Modify values that are to be transmitted using the "Modify all selected parameters immediately and once" button. This column is displayed together with the "Modify value" column.		X

Column	Explanation	Offline	Online
Minimum value	Minimum process-related value of the parameter. If the minimum value is dependent on other parameters, it is defined: • Offline: By the Start value project. • Online: By the Monitor values.	X	X
Maximum value	Maximum process-related value of the parameter. If the maximum value is dependent on other parameters, it is defined: • Offline: By the Start value project. • Online: By the Monitor values.	X	X
Setpoint	Designates the parameter as a setpoint. These parameters can be initialized online.	X	X
Data type	Data type of the parameter. The display field is empty for parameters that are not contained in the instance DB / technology DB.	X	X
Retain	Designates the value as a retentive value. The values of retentive parameters are retained even after the voltage supply is switched off.	X	X
Accessible from HMI	Indicates whether the HMI can access this parameter during runtime.	X	X
Visible in HMI	Indicates whether the parameter is visible in the selection list of the HMI by default.	X	X
Comment	Brief description of the parameter.	X	X

See also

Comparing values (Page 6002)

Opening the parameter view (S7-1200)

Requirement

The technology object has been added in the project tree, i.e., the associated instance DB / technology DB of the instruction has been created.

Procedure

1. Open the "Technology objects" folder in the project tree.
2. Open the technology object in the project tree.
3. Double-click the "Configuration" object.
4. Select the "Parameter view" tab in the top right corner.

Result

The Parameter view opens. Each displayed parameter is represented by one row in the parameter table.

The displayable parameter properties (table columns) vary depending on whether you are working with the Parameter view in offline or online mode.

In addition, you can selectively display and hide individual table columns.

See also

Default setting of the parameter view (Page 6010)

Default setting of the parameter view (S7-1200)

Default settings

To enable you to work efficiently with the Parameter view, you can customize the parameter display and save your settings.

The following customizations are possible and can be saved:

- Show and hide columns
- Change column width
- Change order of the columns
- Toggle navigation
- Select parameter group in the navigation
- Selection of compare values

Show and hide columns

To show or hide columns in the parameter table, follow these steps:

1. Position the cursor in the header of the parameter table.
2. Select the "Show/Hide" command in the shortcut menu.
   The selection of available columns is displayed.
3. To show a column, select the check box for the column.
4. To hide a column, clear the check box for the column.

or

1. Position the cursor in the header of the parameter table.
2. Select the "Show all columns" command in the shortcut menu if all columns of the offline or online mode are to be displayed.

Some columns can only be displayed in online mode: see Parameter table (Page 6008).

Change column width

To customize the width of a column so that all texts in the rows can be read, follow these steps:

1. Position the cursor in the header of the parameter table to the right of the column to be customized until the shape of the cursor changes to a cross.
2. Then double-click this location.

or

1. Open the shortcut menu on the header of the parameter table.
2. Click
   - "Optimize column width" or
   - "Optimize width of all columns".

If the column width setting is too narrow, the complete content of individual fields are shown if you hover the cursor briefly over the relevant field.

Change order of the columns

The columns of the parameter table can be arranged in any way.

To change the order of the columns, follow these steps:

1. Click on the column header and use a drag-and-drop operation to move it to the desired location.
   When you release the mouse button, the column is anchored to the new position.

Toggle navigation

To toggle the display form of the parameters, follow these steps:

1. Select the desired navigation in the “Select navigation structure” drop-down list.
   - Data navigation
   - Functional navigation

See also Navigation (Page 6007).

Select parameter group in the navigation

Within the selected navigation, you choose between the “All parameters” display or the display of a subordinate parameter group of your choice.

1. Click the desired parameter group in the navigation.
   The parameter table only displays the parameters of the parameter group.

Selection of compare values (online)

To set the compare values for the “Compare values” function, follow these steps:

1. Select the desired compare values in the “Selection of compare values” drop-down list.
   - Start value project / Start value PLC
   - Start value project / Snapshot
   - Start value PLC / Snapshot

The “Start value project / Start value PLC” option is set by default.

Saving the default setting of the Parameter view

To save the above customizations of the Parameter view, follow these steps:

1. Customize the Parameter view according to your requirements.
2. Click the “Save window settings” button at the top right of the Parameter view.

Working with the parameter view (S7-1200)

Overview (S7-1200)

The following table provides an overview of the functions of the Parameter view in online and offline mode described in the following.

- Column "Offline" = X: This function is possible in offline mode.
- Column "Online" = X: This function is possible in online mode.

Function/action	Offline	Online
Filtering the parameter table (Page 6013)	X	X
Sorting the parameter table (Page 6013)	X	X
Transferring parameter data to other editors (Page 6014)	X	X
Indicating errors (Page 6014)	X	X
Editing start values in the project (Page 6015)	X	X
Status of configuration (offline) (Page 6016)	X	
Monitoring values online in the parameter view (Page 6016)		X
Create snapshot of monitor values (Page 6018)		X
Modifying values (Page 6019)		X
Comparing values (Page 6020)		X
Applying values from the online program as start values (Page 6021)		X
Initializing setpoints in the online program (Page 6022)		X

Filtering the parameter table (S7-1200)

You can filter the parameters in the parameter table in the following ways:

- With the text filter
- With the subgroups of the navigation

Both filter methods can be used simultaneously.

With the text filter

Texts that are visible in the parameter table can be filtered. This means only texts in displayed parameter rows and columns can be filtered.

1. Enter the desired character string for filtering in the “Text filter...” input box.
   The parameter table displays only the parameters containing the character string.

The text filtering is reset.

- When another parameter group is selected in the navigation.
- When navigation is changed from data navigation to functional navigation, or vice versa.

With the subgroups of the navigation

1. Click the desired parameter group in the navigation, e.g., "Static".
   The parameter table only shows the static parameters. You can select further subgroups for some groups of the navigation.
2. Click “All parameters” in the navigation if all parameters are to be shown again.

Sorting the parameter table (S7-1200)

The values of the parameters are arranged in rows. The parameter table can be sorted by any displayed column.

- In columns containing numerical values, sorting is based on the magnitude of the numerical value.
- In text columns, sorting is alphabetical.

Sorting by column

1. Position the cursor in the header cell of the desired column.
   The background of this cell turns blue.
2. Click the column header.

Result

The entire parameter table is sorted by the selected column. A triangle with tip facing up appears in the column header.

Clicking the column header again changes the sorting as follows:

- Symbol "▲": Parameter table is sorted in ascending order.
- Symbol "▼": Parameter table is sorted in descending order.
- No symbol: The sorting is removed again. The parameter table assumes the default display.

The "../" prefix in the "Name in DB" column is ignored when sorting.

Transferring parameter data to other editors (S7-1200)

After selecting an entire parameter row of the parameter table, you can use the following:

- Drag-and-drop
- <Ctrl+C>/<Ctrl+V>
- Copy/Paste via shortcut menu

Transfer parameters to the following editors of the TIA Portal:

- Program editor
- Watch table
- Signal table for trace function

The parameter is inserted with its full name: See information in "Full name in DB" column.

Indicating errors (S7-1200)

Error indication

Parameter assignment errors that result in compilation errors (e.g., limit violation) are indicated in the Parameter view.

Every time a value is input in the Parameter view, a check is made for process-related and syntax errors and the result is indicated.

Bad values are indicated by:

- Red error symbol in the "Status of configuration" (offline mode) or "Compare result" (online mode, depending on the selected comparison type) columns

and/or

- Table field with red background
  If you click the bad field, a roll-out error message appears with information of the permissible value range or the required syntax (format)

Compilation error

From the error message of the compiler, you can directly open the Parameter view (functional navigation) containing the parameter causing the error in situations where the parameter is not displayed in the configuration dialog.

Editing start values in the project (S7-1200)

With the Parameter view, you can edit the start values in the project in offline mode and online mode.

- You make value changes in the “Start value project” column of the parameter table.
- In the “Status of configuration” column of the parameter table, the progress of the configuration is indicated by the familiar status symbols from the configuration dialog of the technology object.

Boundary conditions

- If other parameters depend on the parameter whose start value was changed, the start value of the dependent parameters are also adapted.
- If a parameter of a technology object is not editable, it is also not editable in the parameter view. The ability to edit a parameter can also depend on the values of other parameters.

Defining new start values

To define start values for parameters in the Parameter view, follow these steps:

1. Open the Parameter view of the technology object.
2. Enter the desired start values in the "Start value project" column. The value must match the data type of the parameter and must not exceed the value range of the parameter. The limits of the value range can be seen in the “Maximum value” and “Minimum value” columns.

The "Status of configuration" column indicates the progress of the configuration with colored symbols.

See also Status of configuration (offline) (Page 6016)

Following adaptation of the start values and downloading of the technology object to the CPU, the parameters take the defined value at startup if they are not declared as retentive (“Retain” column).

Error indication

When a start value is input, a check is made for process-related and syntax errors and the result is indicated.

Bad start values are indicated by:

- Red error symbol in the "Status of configuration" (offline mode) or "Compare result" (online mode, depending on the selected comparison type) columns

and/or

- Red background in the “Start value project” field
  If you click on the bad field, a roll-out error message appears with information of the permissible value range or the necessary syntax (format)

Correcting bad start values

1. Correct bad start values using information from the roll-out error message.
   Red error symbol, red field background, and roll-out error message are no longer displayed.

The project cannot be successfully compiled unless the start values are error-free.

Status of configuration (offline) (S7-1200)

The status of the configuration is indicated by icons:

- In the “Status of configuration” column in the parameter table
- In the navigation structure of the functional navigation and data navigation

Symbol in “Status of configuration” column

Symbol	Meaning
	The start value of the parameter corresponds to the default value and is valid. A start value has not yet been defined by the user.
	The start value of the parameter contains a value defined by the user or an automatically adjusted value. The start value is different than the default value. The start value is error-free and valid.
	The start value of the parameter is invalid (syntax or process-related error). The input box has a red background. When clicked, the roll-out error message indicates the cause of the error.
	Only for S7-1200 Motion Control: The start value of the parameter is valid but contains warnings. The input box has a yellow background.
	The parameter is not relevant in the current configuration.

Symbol in the navigation

The symbols in the navigation indicate the progress of the configuration in the same way as in the configuration dialog of the technology object.

See also

Configure technology objects (Page 5998)

Monitoring values online in the parameter view (S7-1200)

You can monitor the values currently taken by the parameters of the technology object in the CPU (monitor values) directly in the Parameter view.

Requirements

- There is an online connection.
- The technology object is downloaded to the CPU.

- The program execution is active (CPU in "RUN").
- The Parameter view of the technology object is open.

Procedure

1. Start the monitoring by clicking .
   As soon as the Parameter view is online, the following columns are additionally displayed:
   - Compare result
   - Start value PLC
   - Monitor value
   - Modify value
   - Selection for transmission

   The "Monitor value" column shows the current parameter values on the CPU.
   Meaning of the additional columns: see Parameter table (Page 6008)
2. Stop the monitoring by clicking  again.

Display

All columns that are only available online have an orange background:

- Values in light-orange cells can be changed.
- Values in cells with a dark orange background cannot be changed.

Create snapshot of monitor values (S7-1200)

You can back up the current values of the technology object on the CPU (monitor values) and display them in the Parameter view.

Requirements

- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").
- The Parameter view of the technology object is open.
- The “Monitor all” button  is selected.

Procedure

To show the current parameter values, follow these steps:

1. In the Parameter view, click the “Create snapshot of monitor values" icon .

Result

The current monitor values are transferred once to the "Snapshot" column of the parameter table.

You can analyze the values "frozen" in this way while the monitor values continue to be updated in the "Monitor values" column.

Modifying values (S7-1200)

With the Parameter view, you can modify values of the technology object in the CPU.

You can assign values to the parameter once (Modify value) and modify them immediately. The modify request is executed as quickly as possible without reference to any particular point in the user program.

⚠ DANGER

Danger when modifying:

Changing the parameter values while the plant is operating may result in severe damage to property and personal injury in the event of malfunctions or program errors.

Make sure that dangerous states cannot occur before you use the "Modify" function.

Requirements

- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").
- The Parameter view of the technology object is open.
- The “Monitor all” button is selected.
- The parameter can be modified (associated field in the "Modify value" column has a light-orange background).

Procedure

To modify parameters immediately, follow these steps:

1. Enter the desired modify values in the “Modify values” column of the parameter table.
2. Check whether the check box for modifying is selected in the "Select for transmission" column.
   The modify values and associated check boxes of dependent parameters are automatically adapted at the same time.
3. Click the “Modify all selected parameters immediately and once” icon .

The selected parameters are modified once and immediately with the specified values and can be monitored in the "Modify values" column. The check boxes for modifying in the

"Selection for transmission" column are automatically cleared after the modify request is complete.

Error indication

When a start value is input, a check is made immediately for process-related and syntax errors and the result is indicated.

Bad start values are indicated by:

- Red background in the “Modify value” field

and

- If you click the bad field, a roll-out error message appears with information of the permissible value range or the necessary syntax (format)

Bad modify values

- Modify values with process-related errors can be transmitted.
- Modify values with syntax errors **cannot** be transmitted.

Comparing values (S7-1200)

You can use comparison functions to compare the following memory values of a parameter:

- Start value project
- Start value PLC
- Snapshot

Requirements

- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").
- The Parameter view of the technology object is open.
- The “Monitor all” button  is selected.

Procedure

To compare the start values on the various target systems, follow these steps:

1. Click the "Selection of compare values" icon .
   A selection list containing the comparison options opens:
   - Start value project - Start value PLC (default setting)
   - Start value project - Snapshot
   - Start value PLC - Snapshot
2. Select the desired comparison option.
   The selected comparison option is executed as follows:
   - A scales symbol appears in the header cells of the two columns selected for comparison.
   - Symbols are used in the "Compare result" column to indicate the result of the comparison of the selected columns.

Symbol in "Compare result" column

Symbol	Meaning
	The compare values are equal and error-free.
	The compare values are not equal and error-free.
	At least one of the two compare values has a process-related or syntax error.
	The comparison cannot be performed. At least one of the two comparison values is not available (e.g. snapshot).
	Comparison of the value is inappropriate since it is not relevant in one of the configurations.

Symbol in the navigation

The symbols are shown in the same way in the navigation if the comparison result applies to at least one of the parameters below the displayed navigation structure.

Applying values from the online program as start values (S7-1200)

In order to apply optimized values from the CPU to the project as start values, you create a snapshot of the monitor values. Values of the snapshot marked as a "Setpoint" are then applied to the project as start values.

Requirements

- The technology object is of type "PID_Compact" or "PID_3Step".
- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").

- The Parameter view of the technology object is open.
- The “Monitor all” button is selected.

Procedure

To apply optimized values from the CPU, follow these steps:

1. Click the "Create snapshot of monitor values and accept setpoints of this snapshot as start values" icon .

Result

The current monitor values are applied to the "Snapshot" column and their setpoints are copied to the "Start value project" column as new start values.

Note

Applying values of individual parameters

You can also apply the values of individual parameters that are not marked as a setpoint from the "Snapshot" column to the "Start values project" column. To do so, copy the values and insert them into the "Start value project" column using the "Copy" and "Paste" commands in the shortcut menu.

Initializing setpoints in the online program (S7-1200)

You can initialize all parameters that are marked as a "Setpoint" in the Parameter view with new values in the CPU in one step. In so doing, the start values are downloaded from the project to the CPU. The CPU remains in "RUN" mode.

To avoid data loss on the CPU during a cold restart or warm restart, you must also download the technology object to the CPU.

DANGER

Danger when changing parameter values

Changing the parameter values while the plant is operating may result in severe damage to property and personal injury in the event of malfunctions or program errors.

Make sure that dangerous states cannot occur before you reinitialize the setpoints.

Requirements

- The technology object is of type "PID_Compact" or "PID_3Step".
- There is an online connection.
- The technology object is downloaded to the CPU.
- The program execution is active (CPU in "RUN").

- The Parameter view of the technology object is open.
- The “Monitor all” button is selected.
- The parameters marked as a "Setpoint" have a "Start value project" that is free of process-related and syntax errors

Procedure

To initialize all setpoints, follow these steps:

1. Enter the desired values in the "Start value project" column.
   Ensure that the start values are free of process-related and syntax errors.
2. Click the "Initialize setpoints" icon .

Result

The setpoints in the CPU are initialized with the start values from the project.

Configuring technology modules for Motion Control (S7-1200)

Overview (S7-1200)

The following technology modules can be used as the encoder connection in S7-1200 Motion Control.

ET 200MP	ET 200 SP
TM Count 2x24V (Page 6240)	TM Count 1x24V (Page 6240)
TM PosInput 2 (Page 6238)	TM PosInput 1 (Page 6238)

Technology modules can be used centrally or distributed in the system. However, isochronous mode is supported only with decentralized use.

The following section describes how to configure the technology modules as encoder:

See also

TM Count 1x24V / TM Count 2x24V (Page 6240)

TM PosInput 1 / TM PosInput 2 (Page 6238)

TM PosInput 1 / TM PosInput 2 (S7-1200)

For use with S7-1200 Motion Control, the following parameters must be configured:

Configuration	
Technology module **TM PosInput 1 / TM PosInput 2**	**Technology object** **Axis**
Basic parameters > Channel 0/1 > Operating mode	–
Select "Position input for Motion Control" mode	

Configuration	
Technology module **TM PosInput 1 / TM PosInput 2**	**Technology object** **Axis**
Basic parameters > Channel 0/1 > Module parameters	**Basic parameters > Encoder**
–	Select encoder and corresponding channel
Signal type • Incremental encoder • Absolute encoder	Select encoder type corresponding to configuration for technology module • Incremental • Absolute
–	**Data exchange with encoder**
	Telegram "DP_TEL83_STANDARD" is automatically selected after the selection of the encoder.
	Disable "Automatic transfer of encoder parameter values in the device"
	Select rotary or linear measuring system type
Signal evaluation • Single • Double • Quadruple	Select fine resolution corresponding to configuration for technology module • Incremental encoder: – 0 = Single – 1 = Double *or* – 2 = Quadruple • Absolute encoder: – 0 (= single)
Enter increments per revolution	• Rotary type: Enter increments per revolution corresponding to configuration at technology module (1:1) • Linear type: Enter distance per revolution
–	**Hardware interface > Data exchange with drive**
• Rotary type: Enter reference speed corresponding to configuration for technology object (1:1) • Linear type: Configuration not relevant	Enter reference speed
I/O addresses	–
Organization block: MC-Servo	
Process image: PIP OB servo	
Select "Isochronous mode"	

"–" No configuration for technology module/technology object is required for these parameters

See also

Overview (Page 6238)

TM Count 1x24V / TM Count 2x24V (Page 6240)

TM Count 1x24V / TM Count 2x24V (S7-1200)

For use with S7-1200 Motion Control, the following parameters must be configured:

Configuration	
Technology module **TM Count 1x24V / TM Count 2x24V**	**Technology object** **Axis**
Basic parameters > Channel 0/1 > Operating mode	–
Select "Position input for Motion Control" mode	
Basic parameters > Channel 0/1 > Module parameters	**Hardware interface > Encoder**
–	Select encoder and corresponding channel
Signal type • Incremental encoder	Select encoder type corresponding to configuration for technology module • Incremental
–	**Hardware interface > Data exchange with encoder**
	Telegram "DP_TEL83_STANDARD" is automatically selected after the selection of the encoder.
	Disable "Automatic transfer of encoder parameter values in the device"
	Select rotary or linear measuring system type
Signal evaluation • Single • Double • Quadruple	Select fine resolution corresponding to configuration for technology module • 0 = Single • 1 = Double • 2 = Quadruple
Enter increments per revolution	• Rotary type: Enter increments per revolution corresponding to configuration at technology module (1:1) • Linear type: Enter distance per increment
–	**Hardware interface > Data exchange with drive**
• Rotary type: Enter reference speed corresponding to configuration for technology object (1:1) • Linear type: Configuration not relevant	Enter reference speed
I/O addresses	–
Organization block: MC-Servo	
Process image: PIP OB servo	
Select "Isochronous mode"	

See also

Overview (Page 6238)

TM PosInput 1 / TM PosInput 2 (Page 6238)

13.2.6 Technology object command table (S7-1200)

13.2.6.1 Use of the command table technology object (S7-1200)

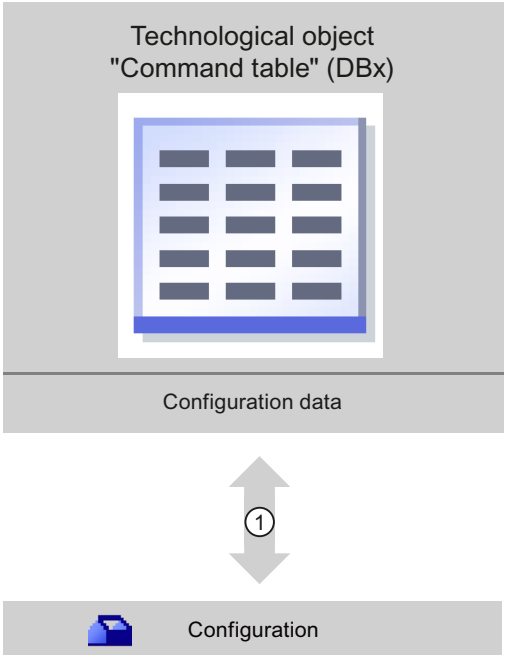
The technology object "Command table" allows you to combine multiple individual axis control commands in one movement sequence. The technology object can be used as of technology version V2 for axes with drive connection via PTO (Pulse Train Output).

You configure the movement sequence as a table in a configuration dialog.

The motion profile of the movement sequences can be checked on a graph before the project is loaded to the CPU. The command tables created are then linked to an axis and used in the user program with the "MC_CommandTable" Motion Control instruction. You can process part or all of the command table.

13.2.6.2 Command table technology object tools (S7-1200)

The "Configuration" tool is provided in the TIA Portal for the "Command Table" technology object. The representation below shows the interaction of the tool with the technology object:



①	Writing and reading the configuration of the technology object

Configuration

Configure the following properties of the "Command Table" technology object with the "Configuration" tool:

- You can create one or more movement sequences by configuring individual jobs.
- You can configure the graphic display to check your movement sequence using an axis already configured or a configurable default axis.

The movement sequence data are saved in the data block of the technology object.

13.2.6.3 Adding the technological object command table (S7-1200)

Requirements

- A project with a CPU S7-1200 has been created.
- The CPU firmware version is V2.1 or higher

Procedure

Proceed as follows to add a "Command table" technology object in the project tree:

1. Open the "CPU > Technology objects" folder in the project tree.
2. Double-click the "Add new object" command.
   The "Add new object" dialog opens.
3. Select the "Motion Control" technology.
4. Open the "Motion Contro > S7-1200 Motion Control" folder.
5. Select the desired technology version in the "Version" column.
6. Select the "TO_CommandTable" object.
7. Enter the name of the command table in the "Name" input box.
8. To change the automatically assigned data block number, select the "Manual" option.
9. To display additional information about the technology object, click "Additional information".
10. Confirm your entry with "OK".

Result

The new technology object is created and saved to the "Technology objects" folder in the project tree.

13.2.6.4 Configuring the command table technology object (S7-1200)

Working with the configuration dialog (S7-1200)

You configure the properties of the technology object in the configuration window. Proceed as follows to open the configuration window of the technology object:

1. Open the group of the required technology object in the project tree.
2. Double-click the "Configuration" object.

The configuration is divided into the following categories:

- **Basic parameters**
  The basic parameters contain all parameters which must be configured for a functional command table.
- **Extended parameters**
  The extended parameters contain the parameters of the default axis or display the parameter values of the axis selected.

Configuration window icons

Icons in the area navigation of the configuration show additional details about the status of the configuration:

	**The configuration contains default values and is complete.** The configuration contains only default values. With these default values you can use the technology object without additional changes.
	**The configuration contains user-defined or automatically adapted values and is complete.** All input fields of the configuration contain valid values and at least one preset value has changed.
	**The configuration is incomplete or incorrect** At least one input field or drop-down list contains an invalid value. The corresponding field or the drop-down list is displayed on a red background. Click the roll-out error message to display the cause of the error.
	**The configuration contains mutually incompatible parameter values** The configuration contains parameter values that contradict each other either in size or logic. The corresponding field or the drop-down list is displayed on a yellow background.

See also

Guidelines on use of motion control (Page 6168)

Basic parameters (Page 6244)

Extended parameters (Page 6257)

Monitor values (S7-1200)

If there is an online connection to the CPU, the icon "Monitor all"  is displayed in the configuration dialog of the technology object.

The "Monitor all" function provides the following options:

- Comparison of configured start values of the project with the start values in the CPU and the actual values
- Direct editing of actual values and the start values of the project
- Immediate detection and display of input errors with suggested corrections
- Backup of the actual values in the project by manual transfer to the start value of the project

Icons and operator controls

If there is an online connection to the CPU, the actual values are displayed at the parameters.

In addition to the actual values of the parameters, the following symbols appear:

Icon	Description
	Start value in CPU matches the configured Start value in the project
	Start value in CPU does not match the configured Start value in the project
	A comparison of the start value in the CPU with the configured start value in the project cannot be performed because the selected CPU module does not support this comparison.
	The value is not comparable with any significance since it is not relevant in one of the configurations.
	Use the button to show the start value of the CPU and the start value of the project for the respective parameter.

The actual value and the start value in the project can be changed directly and then downloaded to the CPU. The change of the actual value is transferred directly to the CPU for directly modifiable parameters.

Basic parameters (S7-1200)

Configuration - General (S7-1200)

Configure the name of the technology object in the "General" configuration window.

Name

Define the name of the command table or the name of the "Command table" technology object in this field. The technology object is listed under this name in the project tree.

See also

Configuration - Command table (Page 6245)

Shortcut menu commands - Command table (Page 6248)

Working with the trend diagram (Page 6249)

Shortcut menu commands - Curve chart (Page 6253)

Transition from "Complete command" to "Blend motion" (Page 6254)

Changing the command table configuration in the user program (Page 6256)

Configuration - Command table (S7-1200)

Create the desired movement sequence in the "Command Table" configuration window and check the result against the graphic view in the trend diagram.

Note

Small deviations are possible between the time behavior and position in the trend shown and the real movement of the axis. Movements in response to software limit switches being reached are not shown.

Enable warnings

Activate the display of warnings in the command table with this check box.

Use axis parameters of

From the drop-down list, select which axis parameters are to be used for selecting the graphic view of and checking the movement sequence. Select "Default axis" if you have yet to add an axis to the "Technology object" folder or wish to use values which have not been configured in any of the available axes. You configure the properties of the default axis under "Advanced parameters".

The axis parameters of the axis selected at the "Axis" parameter are used to process the command table in the user program.

Column: Step

Shows the step number of the command.

Column: Command type

In this column, select the command types which are to be used for processing the command table. Up to 32 commands can be entered. The commands will be processed in sequence. You can choose between the following entries and command types:

- **Empty**
  The entry serves as a placeholder for any commands to be added. The empty entry is ignored when the command table is processed.
- **Halt**
  Stop axis
  (the command only takes effect after a "Velocity set point" command)
- **Positioning Relative**
  Position axis relatively

- **Positioning Absolute**
  Position axis absolutely (the axis must be homed for this)
- **Velocity set point**
  Move axis at set velocity
- **Wait**
  Waits until the given period is over. Wait does not stop active travel.
- **Separator**
  Adds a Separator line above the selected line. The Separator line acts as a range limit for the graphic display of the trend view.
  Use the Separator lines if you wish to process parts of the command table.

Column: Position/travel path

Enter the position or travel path for the selected command in this column:

- **Command "Positioning Relative"**
  The command will move the axis by the given travel path.
- **Command "Positioning Absolute"**
  The command will move the axis by the given position.
  The axis must be homed for this.
- **Separator**
  The value given specifies the start position for the graphic display.

Limit values (independent of the selected user unit):

- -1.0e12 ≤ position / distance ≤ -1.0e-12
- 1.0e-12 ≤ position / distance ≤ 1.0e12
- Position / travel path = 0.0

Column: Velocity

In this column, you enter the velocity for the selected command:

- **Command "Positioning Relative"**
  The command will move the axis at the given velocity.
  The given velocity will not be reached if the travel path selected is not large enough.
- **Command "Positioning Absolute"**
  The command will move the axis at the given velocity.
  The given velocity will not be reached if the target position is too close to the starting position.
- **Command " Velocity set point"**
  The command will move the axis at the given velocity.
  The given velocity will not be reached during the command if too short a runtime is selected.

Limit values (independent of the selected user unit):

- For the commands: "Positioning Relative" and "Positioning Absolute"
  - 1.0e-12 ≤ velocity ≤ 1.0e12
- For the command: "Velocity set point"
  - -1.0e12 ≤ velocity ≤ -1.0e-12
  - 1.0e-12 ≤ velocity ≤ 1.0e12
  - Velocity = 0.0

Column: Duration

Enter the duration of the selected command in this column:

- **Command " Velocity set point"**
  The command will move the axis for the specified duration. The duration includes both the acceleration phase and the constant travel phase. The next command will be processed once the duration is over.
- **Command "Wait"**
  Waits until the given duration is over.

Limit values (independent of the selected user unit):

- 0.001s ≤ duration ≤ 64800s

Column: Next step

Select the mode of transition to the next step from the drop-down list:

- **Complete command**
  The command will be completed. The next command will be processed immediately.
- **Blend motion**
  The motion of the current command will be blended with the motion of the following command. The transition mode "Blend motion" is available with command types "Positioning Relative" and "Positioning Absolute".
  Motion will be blended with motions of the following command types:
  - Positioning Relative
  - Positioning Absolute
  - Velocity set point

No blending occurs with other command types.

For a detailed description of the response of the axis when a command is appended or overlapped, see section Transition from "Complete command" to "Blend motion" (Page 6254)

Column: Step code

Enter a numerical value / bit pattern in this column which is to be output at the "StepCode" output parameter of the "MC_CommandTable" Motion Control instruction while the command is being processed.

Limit values:

- 0 ≤ code number ≤ 65535

See also

Configuration - General (Page 6244)

Shortcut menu commands - Command table (Page 6248)

Working with the trend diagram (Page 6249)

Shortcut menu commands - Curve chart (Page 6253)

Transition from "Complete command" to "Blend motion" (Page 6254)

Changing the command table configuration in the user program (Page 6256)

Shortcut menu commands - Command table (S7-1200)

The following shortcut menu commands are available in the command table:

Insert empty line

Adds an empty line above the selected line.

This shortcut menu command can only be executed if there are enough empty lines at the end of the command table.

Add empty line

Adds an empty line below the selected line.

This shortcut menu command can only be executed if there are enough empty lines at the end of the command table.

Insert separator line

Adds a separator line above the selected line.

You cannot have two consecutive separator lines.

Add separator line

Adds a separator line below the selected line.

You cannot have two consecutive separator lines, nor can you add a separator line at the end of the command table.

Cut

Removes the selected lines or content of the selected cell and saves them/it in the clipboard.

Selected lines will be deleted and the subsequent lines of the command table shifted up.

Copy

Copies the selected lines or content of the selected cell and saves them/it in the clipboard.

Paste

- Selected lines:
  Pastes the lines from the clipboard into the table above the selected line.
- Selected cell:
  Pastes the content of the clipboard into the selected line.

This shortcut menu command can only be executed if there are enough empty lines at the end of the command table.

Replace

Replaces the selected lines with the lines in the clipboard.

Delete

Deletes the selected lines. The lines below in the command table shift up.

See also

Configuration - General (Page 6244)

Configuration - Command table (Page 6245)

Working with the trend diagram (Page 6249)

Shortcut menu commands - Curve chart (Page 6253)

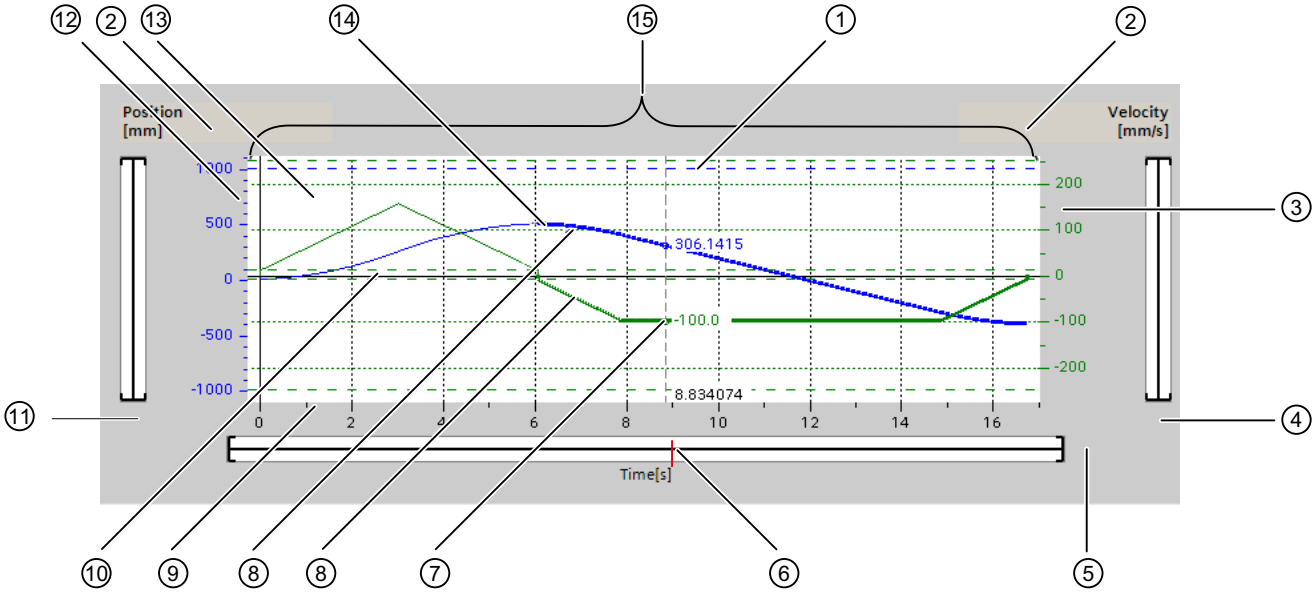
Transition from "Complete command" to "Blend motion" (Page 6254)

Changing the command table configuration in the user program (Page 6256)

Working with the trend diagram (S7-1200)

The following tools and information are available in the trend view:

Trend view and components



①	Ruler
②	Selecting the grid
③	Velocity axis scale range
④	Scroll bar, velocity axis
⑤	Scroll bar time axis
⑥	Ruler position marking
⑦	Velocity curve
⑧	Curve section of a selected command
⑨	Time axis scale range
⑩	Start/stop velocity
⑪	Scroll bar, position axis
⑫	Position axis scale range
⑬	Software limit switch position
⑭	Position curve
⑮	Trend view

Selecting separator sections

If the command table consists of multiple sections separated by separators, you can select these sections in the trend view by selecting a command in the section.

Selecting commands

Commands can be selected in the trend view and in the command table:

- Click on a point on the velocity or position curve in the trend view. The corresponding command will be highlighted in the command table.
- Select a command in the command table.
  The corresponding section of curve will be highlighted.

Selecting the visible range of the trend view

Follow the steps below to adjust the section of the trend view to be displayed:

Select the scaling in the shortcut menu:

- Scale to curves:
  Scales the axes so the position and velocity curves are visible.
- Scale to curves and limits:
  Scales the axes so the position and velocity curves, the positions of the activated software limit switches and the minimum and maximum velocity limits are visible.

The view selected will be marked in the shortcut menu with a tick.

Selecting the section to be shown within the range:

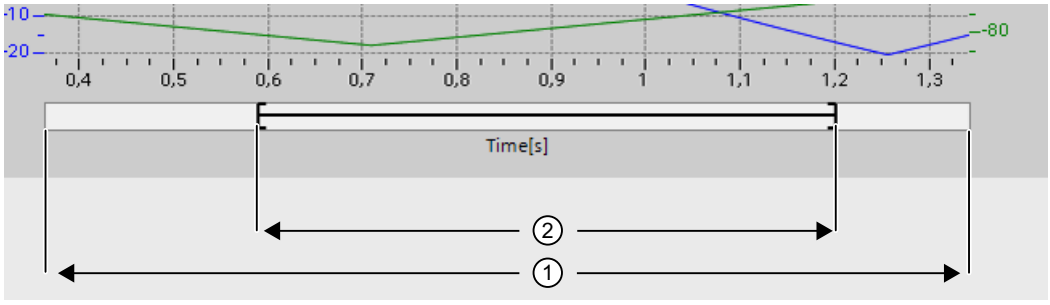
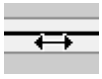
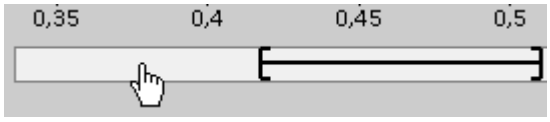
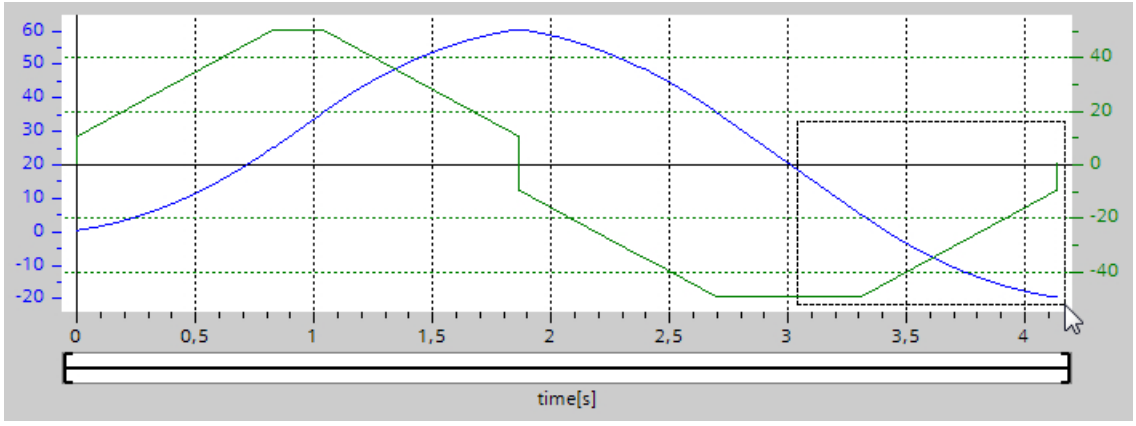


Figure 13-1 CmdTable_Scle01_new

①	Range which the curve values and / or limits are within. (see Selecting in the shortcut menu)
②	Selected range to be shown in the trend window. You set the range with the margin cursor at the right-hand and left-hand margin.  You set the position within range ① with the drag cursor.  You can also define the position by clicking in range ①. 

Selecting the section to be shown with the mouse:

Drag a section of the trend view by clicking and dragging with the mouse. The section of curve selected will be enlarged once you release the mouse.



Undoing the last change to the section:

Select the shortcut command "Undo zoom" to undo the last change to the section.

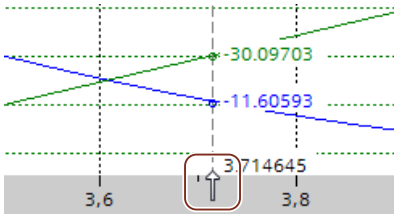
Synchronizing the grid

Click on the axis scales to select whether the grid is to be synchronized with the position axis or velocity axis.

Reading off curve values from the ruler

Activate the ruler using the shortcut menu command "Show ruler".

You can move the ruler to any point on the curves using the ruler cursor.



See also

Configuration - General (Page 6244)

Configuration - Command table (Page 6245)

Shortcut menu commands - Command table (Page 6248)

Shortcut menu commands - Curve chart (Page 6253)

Transition from "Complete command" to "Blend motion" (Page 6254)

Changing the command table configuration in the user program (Page 6256)

Shortcut menu commands - Curve chart (S7-1200)

The following shortcut menu commands are available in the curve window:

Zoom 100%

Selects a zoom factor which will show 100% of the curve values and / or limits.

Undo zoom

Undoes the last zoom change.

Scaling on trends

Scales the axes so the position and velocity trends are visible.

Scaling on trends and limits

Scales the axes so the position and velocity trends, the positions of the activated software limit switches and the minimum and maximum velocity limits are visible.

Show velocity limits

Shows the lines of the velocity limits.

Show software limit switches

Shows the lines of the software limit switches.

Show measuring ruler

Fades the measuring ruler in / out

Use the measuring ruler when you want to see the individual values of the trends.

See also

Configuration - General (Page 6244)

Configuration - Command table (Page 6245)

Shortcut menu commands - Command table (Page 6248)

Working with the trend diagram (Page 6249)

Transition from "Complete command" to "Blend motion" (Page 6254)

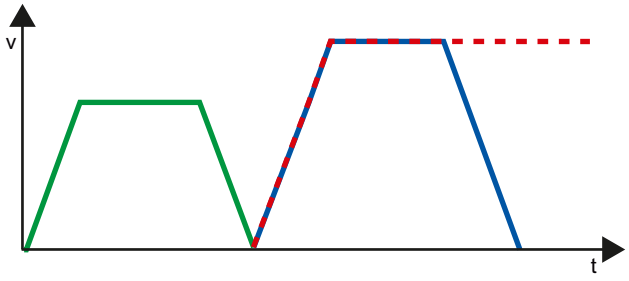
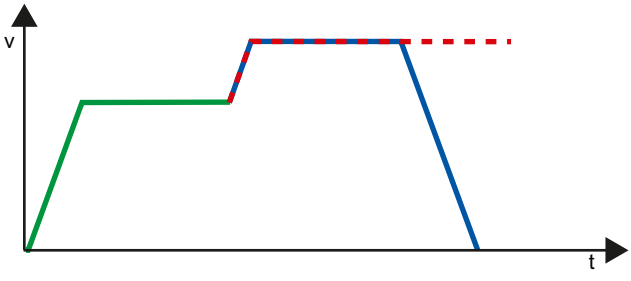
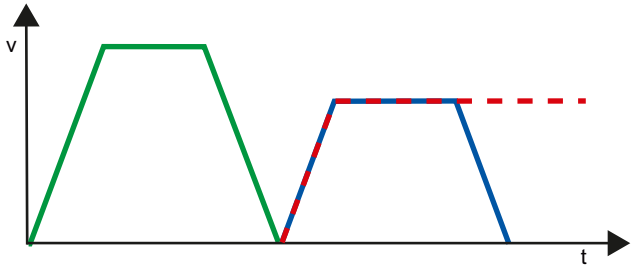
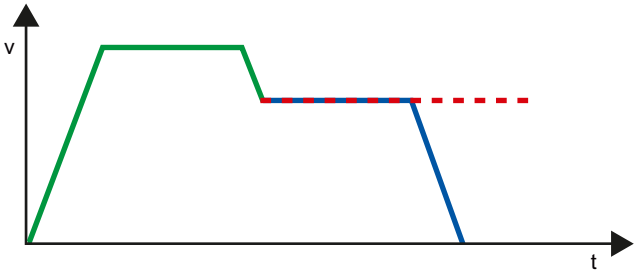
Changing the command table configuration in the user program (Page 6256)

Transition from "Complete command" to "Blend motion" (S7-1200)

The charts below show the transition between movements in various different transition modes in the "Next step" column:

Motion transition with preceding positioning jobs

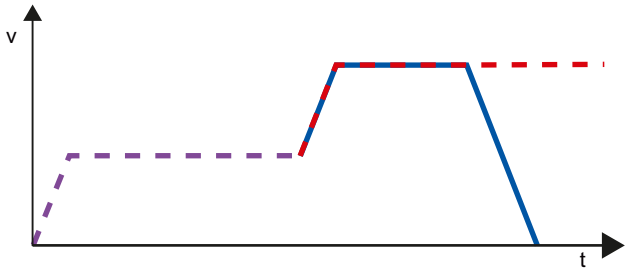
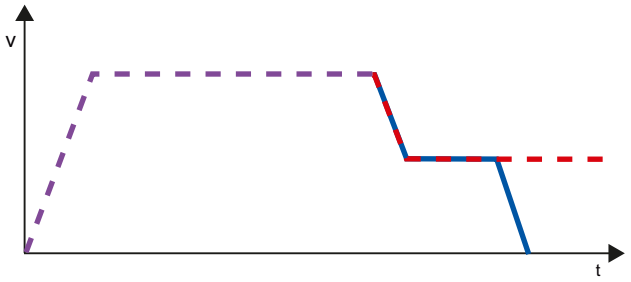
The following diagrams show a command sequence with two motion tasks. The first command is for positioning (green). The second command is for velocity (red) or positioning (blue):

Complete job	Blend motion
Transition from lower to higher velocity  A job with high velocity is appended to a previous positioning job. The positioning job terminates at its target position at velocity "0". The second job starts from standstill.	**Transition from lower to higher velocity**  A job with high velocity is overlapped with a previous positioning job. The first positioning job terminates without standstill at its target position. The second job starts with the new velocity.
Transition from higher to lower velocity  A job with low velocity is appended to a previous positioning job. The positioning job terminates at its target position at velocity "0". The second job starts from standstill.	**Transition from higher to lower velocity**  A job with low velocity is overlapped with a previous positioning job. The first positioning job terminates without standstill at its target position. The first job starts with the new velocity.

—	1. Job "Positioning Relative" or "Positioning Absolute"
- -	2. Job "Velocity set point"
—	2. Job "Positioning Relative" or "Positioning Absolute"

Motion transition with preceding velocity jobs

The following diagrams show a command sequence with two motion tasks. The first command is for velocity (violet). The second command is for velocity (red) or positioning (blue):

Transition from lower to higher velocity	Transition from higher to lower velocity
	
A job with a high velocity is appended to a previous velocity job. The first velocity job ends after the defined runtime. The second job starts with the new velocity.	A command with lower velocity is blended with a previous velocity command. The second job starts with the new velocity.

	1. Job "Velocity set point"
	2. Job "Velocity set point"
	2. Job "Positioning Relative" or "Positioning Absolute"

See also

Configuration - General (Page 6244)

Configuration - Command table (Page 6245)

Shortcut menu commands - Command table (Page 6248)

Working with the trend diagram (Page 6249)

Shortcut menu commands - Curve chart (Page 6253)

Changing the command table configuration in the user program (Page 6256)

Changing the command table configuration in the user program (S7-1200)

You can change the following configuration parameters during runtime of the user program in the CPU:

Commands and corresponding values

You can also change the parameters of the command table during the runtime of the user program. Use the following technology object tags for this purpose:

- <Table name>.Command[1..32].Type
  for changing the command type
- <Table name>.Command[1..32].Position
  for changing the position/travel distance

- <Table name>.Command[1..32].Velocity
  for changing the velocity
- <Table name>.Command[1..32].Duration
  for changing the duration
- <Table name>.Command[1..32].NextStep
  for changing the parameter "Next step"
- <Table name>.Command[1..32].StepCode
  for changing the step code

Refer to the description of the technology object tags (Page 6472) in the appendix for information on when changes to the configuration parameters take effect.

See also

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

Configuration - General (Page 6244)

Configuration - Command table (Page 6245)

Shortcut menu commands - Command table (Page 6248)

Working with the trend diagram (Page 6249)

Shortcut menu commands - Curve chart (Page 6253)

Transition from "Complete command" to "Blend motion" (Page 6254)

Extended parameters (S7-1200)

Configuration - Extended parameters (S7-1200)

Configure the basic properties of the chart view of the "Command table" technology object in the "Extended parameters" configuration window.

Note

If the default axis has been selected under "Use axis parameters of", the unit of measurement can be edited. If a configured axis has been selected, the unit of measurement for this axis will be displayed.

Use axis parameters of

From the drop-down list, select which axis parameters are to be used for selecting the graphic view of and checking the movement sequence. Select "Default axis" if you have yet to add an axis to the "Technology object" folder or wish to use values which have not been configured in any of the available axes.

The axis parameters of the axis selected at the "Axis" parameter will be used to process the command table in the user program.

Unit of measurement position

Enter the unit of measurement for the default axis in this field. If a preconfigured axis has been selected under "Use axis parameters of", the unit of measurement configured in these parameter will be displayed.

Copy axis parameters

Select the direction of copy and the axis for copying the axis parameters. You can copy the axis parameters of the default axis to the selected axis or accept the axis parameters of the selected axis for the default axis. Use the "Apply configuration" button to copy the axis parameters according to your configuration.

Configuration - Dynamics (S7-1200)

Configure the acceleration and deceleration and the jerk limit for the default axis in the "Dynamics" configuration window.

Note

If the default axis has been selected under "Use axis parameters of", the following fields can be edited. If a configured axis has been selected, the values of this axis will be displayed.

Acceleration / deceleration

Set the desired acceleration of the default axis in the "Acceleration" field. The desired deceleration can be set in the "Deceleration" field.

Motion jobs configured in the command table will be calculated with the selected acceleration / deceleration.

Limit values:

- 1.0e-12 ≤ acceleration ≤ 1.0e12
- 1.0e-12 ≤ deceleration ≤ 1.0e12

Activate jerk limit

Enable the jerk limit with this check box.

Jerk

Set the desired jerk for ramping up and ramping down in the "Jerk" field.

Motion jobs configured in the command table will be calculated with the selected jerk.

Limit values:

- 1.0e-12 ≤ jerk ≤ 1.0e12

Configuration - Limit values (S7-1200)

Configure the maximum velocity, the start/stop velocity and the software limit switches of the default axis in the "Limits" configuration window.

Note

If the default axis has been selected under "Use axis parameters of", the following fields can be edited. If a configured axis has been selected, the values of this axis will be displayed.

Maximum velocity / Start/stop velocity

Define the maximum permissible velocity and the start/stop velocity of the default axis in these boxes. The start/stop velocity is the minimum permissible velocity of the default axis.

Limit values:

- 1.0e-12 ≤ start/stop velocity ≤ 1.0e12
  Start/stop velocity = 0.0
- 1.0e-12 ≤ maximum velocity ≤ 1.0e12
  Maximum velocity = 0.0

The value of the maximum velocity must be greater or equal to the value of the start/stop velocity.

Enable software limit switches

Activate the function of the low and high software limit switch with this check box. Movements in response to software limit switches being reached are not shown in the trend view.

Low / high software limit switch

Enter the position value of the low and high software limit switch in these boxes.

Limit values:

- -1.0e12 ≤ low software limit switch ≤ -1.0e-12
  1.0e-12 ≤ low software limit switch ≤ 1.0e12
  Low software limit switch = 0.0
- -1.0e12 ≤ high software limit switch ≤ -1.0e-12
  1.0e-12 ≤ high software limit switch ≤ 1.0e12
  High software limit switch = 0.0

The value of the high software limit switch must be greater than or equal to the value of the low software limit switch.

13.2.7 Download to CPU (S7-1200)

The data of the Motion Control technology objects are saved in the data blocks. The conditions for downloading of "blocks" therefore apply when loading a new or modified technology object.

⚠ CAUTION

Possible malfunctions of the axis when loading without hardware configuration

The hardware configuration is modified when the following modifications are made to the axis or encoder configuration:

- Modification of the pulse generator (PTO)
- Modification of the HW limit switch address
- Modification of the homing switch address
- Modification of the address of the PROFIdrive telegram
- Modification of the address of the analog output
- Modification of address of enable output or ready input

If the modified configuration of the axis or encoder is loaded with the shortcut menu commands "Software" or "Software (all blocks)" without downloading the hardware configuration, the axis may malfunction as a result.

Ensure that the current hardware configuration is downloaded to the CPU under the listed conditions.

Download in CPU S7-1200 RUN mode (from firmware version V2.2)

For CPU S7-1200 from firmware version V2.2, when loading in CPU RUN mode it is checked whether it is possible to load without stopping the CPU.

The following conditions apply when loading data blocks in RUN mode:

	Download to load memory	Download to work memory
Data block modified values	Yes	No
Data block modified structure	Yes (as of firmware V4)	Yes (as of firmware V4) • When downloading with reinitialization • For variables in system reserve for downloading without reinitialization
	No (firmware V2.2...3)	No (firmware V2.2...3)
New data block	Yes	Yes
Data block deleted	Yes	Yes

Also note the following when deleting data blocks and downloading data blocks with reinitialization:

- The axis must be disabled when downloading a positioning axis technology object.
- When downloading a command table technology object, no MC_CommandTable command with this command table must be active (parameter "Busy" = FALSE).
- When downloading an MC_Power instance data block, no MC_Power instruction must be active (parameter "Busy" = FALSE).

From technology version V3.0, Motion Control technology objects (data blocks) can also be downloaded in CPU RUN mode.

Technology objects lower than V3.0 cannot be downloaded in CPU RUN mode.

Select one of the actions described below to download the modified version of a Motion Control technology object (from version V3.0) to the work memory:

- **Technology object positioning axis and command table**
  Change the CPU operating mode from STOP to RUN.
- **Technology object positioning axis**
  Disable the axis and execute a "Restart" using the Motion Control instruction "MC_Reset".
- **Technology object command table**
  Ensure that the command table is not being used. Download the data block of the command table to the work memory using the extended instruction "READ_DBL".

See also

Guidelines on use of motion control (Page 6168)

MC_Reset: Acknowledge fault, restart technology object V4...5 (Page 4134)

13.2.8 Commissioning (S7-1200)

13.2.8.1 Axis control panel (S7-1200)

Use the axis control panel to move the axis in manual mode, to optimize the axis settings, and to test your system.

The axis control panel can only be used if an online connection to the CPU is established.

Note

Response times of the axis control panel

The response time during axis control panel operation depends on the communication load of the CPU. Close all other online windows of the TIA Portal to minimize the response time.

You can adjust the timeout in the start dialog.

"Manual control" button

Click "Manual control" to move the axis in manual control mode. Start by disabling the axis in the user program using Motion Control instruction "MC_Power". In "Manual control" mode, the axis control panel takes over master control for the axis functions. The user program has no influence on the axis functions until manual control is ended.

CAUTION

Additional axes in automatic mode

The Manual control is active for one axis only. If additional axes are in automatic mode, dangerous situations may arise as a result.

In this happens, disable all other axes.

"Automatic mode" button

Click "Automatic mode" to end the "Manual control" mode. The axis control panel passes back the master control and the axis can be controlled by the user program again. The axis must be re-enabled in the user program and homed, if required.

Complete all active traversing motions before switching to automatic control; otherwise, the axis will be braked with the emergency stop deceleration.

"Enable" button

Click "Enable" to enable the axis in "Manual control" mode. When the axis is enabled, the axis control panel functions can be used.

If the axis cannot be enabled because certain conditions are not met, note the error message in the "Error message" box. Information on eliminating errors is available in the appendix under "List of ErrorIDs and ErrorInfos". After the error has been corrected, enable the axis again.

"Disable" button

Click "Disable" if you want to temporarily disable the axis in "Manual control" mode.

"Command" area

Operation in the "Command" area is only possible if the axis is enabled. You can select one of the following command inputs:

- **Jog**
  This command is equivalent to Motion Control command "MC_MoveJog" in the user program.
- **Positioning**
  This command is equivalent to the Motion Control commands "MC_MoveAbsolute" and "MC_MoveRelative" in the user program. The axis must be homed for absolute positioning.
- **Homing**
  This command is equivalent to Motion Control command "MC_Home" in the user program.
  - The "Set reference point" button corresponds to Mode = 0 (direct homing absolute)
  - The "Active homing" button corresponds to Mode = 3 (active homing)

  For active homing, the homing switch must be configured in the axis configuration. The values for approach velocity, homing velocity, and reference position offset are taken from the axis configuration unchanged.

Depending on the selection, the relevant boxes for entry of setpoints and the buttons for starting the command are displayed.

Jerk limitation can be activated and deactivated using the "Activate jerk limit" button. By default, the jerk is applied with 10% of the configured value. This value can be changed as required.

"Axis status" area

If "Manual control" mode is activated, the current axis status and drive status are shown in the "Axis status" area. The current position and velocity of the axis are displayed at "Process values".

Click "Acknowledge" to acknowledge all cleared errors.

The "Info message" box displays advanced information about the status of the axis.

"Current values" area

This area displays the current position and velocity of the axis.

Error message

The "Error message" box shows the current error. In "Manual control" mode, the error entry can be deleted by pressing the "Acknowledge" button once the error is eliminated.

Note

Initial values for velocity, acceleration / deceleration and jerk

For safety reasons, the "Velocity", "Acceleration/deceleration" and "Jerk" parameters are initialized with values equivalent to only 10% of the configured values when the axis control panel is activated. The "Jerk" parameter is only used for technology object "Axis" V2.0 and higher.

The values in the configuration view displayed when you select "Extended parameters > Dynamics > General" are used for initialization.

The "Velocity" parameter on the control panel is derived from the "Maximum velocity" and the "Acceleration/deceleration" parameters from "Acceleration" in the configuration.

The "Velocity", "Acceleration/deceleration" and "Jerk" parameters can be changed in the axis control panel; this does not affect the values in the configuration.

See also

Guidelines on use of motion control (Page 6168)

13.2.8.2 Tuning (S7-1200)

The movement of axes with drive connection via PROFIdrive/analog output is position-controlled.

The "Tuning" function supports you in determining the optimal gain (Kv factor) for the control loop (Page 6220) of the axis. The axis velocity profile is recorded by means of the Trace function for this purpose for the duration of a configurable positioning movement. Then you can evaluate the recording, and adapt the gain accordingly.

The "Tuning" function for the positioning axis technology object can be found in the project tree under "Technology object > Commissioning".

The "Tuning" dialog is divided into the following areas:

- Master control
- Axis
- Optimize gain setting
- Trace

Note

No transfer of the parameters

The configured parameter values are discarded after master control is returned. Transfer the values as needed into your configuration.

Start of optimization

The trace is started at the same time the optimization starts. A timeout can be adapted in the start dialog for this.

Master control

In this area, you can take over master control of the technology object, or return it to your user program:

- **"Activate" button**
  With the "Activate" button, you set up an online connection to the CPU and take over master control for the selected technology object. Note the following when taking over master control:
  - To take over master control, the technology object must be disabled in the user program.
  - Until master control is returned, the user program has no influence on the functions of the technology object. Motion Control jobs from the user program to the technology object are rejected with the error ("ErrorID" = 16#8203).

  CAUTION

  Additional axes in automatic mode

  The master control is only applied for the selected technology object. If additional axes are in automatic mode, dangerous situations may arise as a result.

  In this happens, disable all other axes.

- **"Deactivate" button**
  With the "Deactivate" button, you return master control to your user program.

Axis

In this area, enable or disable the technology object for operation with the axis control panel/ tuning:

- **"Enable" button**
  With the "Enable" button, you enable the selected technology object.
- **"Disable" button**
  With the "Disable" button, you disable the selected technology object.

Optimize gain setting

You make the settings for optimization of the gain in this area:

- **"Forward" button**
  With the "Forward" button, you start a test step for optimization in the positive direction.
- **"Backward" button**
  With the "Backward" button, you start a test step for optimization in the negative direction.
- **"Stop" button**
  You can use the "Stop" button to end the current movement for optimization and end trace recording.
- **"Customize dynamics" check box**
  Select this option to adapt the acceleration and the maximum acceleration for the optimization.
- **Acceleration**
  In this field, you configure the acceleration for a test step.
- **Maximum velocity**
  In this field, you configure the maximum velocity for a test step.
- **Measurement duration**
  The measurement duration is recalculated and entered depending on the selected acceleration, velocity and distance.
  You can adapt the value of the measurement duration afterwards.
- **Gain**
  In this field, you configure the actual gain of the position controller (Kv).
  The gain takes effect when it is entered. If the gain of the position controller (Kv) is too large, this can lead to an error on the drive.

Trace

The Trace function is displayed in the lower area of the "Tuning" dialog.

With each test step, a Trace recording of the required parameters is automatically started and displayed after completion of the test step. After master control has been returned, the Trace recording is deleted.

You will find a full description of the Trace function in the section on using the trace and logic analyzer function in the TIA Portal help.

13.2.9 Programming (S7-1200)

13.2.9.1 Overview of the Motion Control statements (S7-1200)

You control the axis with the user program using Motion Control instructions. The instructions start Motion Control commands that execute the desired functions.

The status of the Motion Control commands and any errors that occur during their execution can be obtained from the output parameters of the Motion Control instructions. The following Motion Control instructions are available:

- MC_Power: Enable, disable axis as of V6 (Page 4094)
- MC_Reset: Acknowledge fault, restart technology object as of V6 (Page 4099)
- MC_Home: Home axes, set reference point as of V6 (Page 4100)
- MC_Halt: Stop axis as of V6 (Page 4105)
- MC_MoveAbsolute: Absolute positioning of axis as of V6 (Page 4108)
- MC_MoveRelative: Relative positioning of axis as of V6 (Page 4112)
- MC_MoveVelocity: Move axis at set velocity as of V6 (Page 4115)
- MC_MoveJog: Move axis in jog mode as of V6 (Page 4119)
- MC_CommandTable: Run axis commands as motion sequence as of V6 (Page 4122)
- MC_ChangeDynamic: Change dynamic settings of axis as of V6 (Page 4124)
- MC_ReadParam: Continuously read motion data of a positioning axis as of V6 (Page 4126)
- MC_WriteParam: Write tag of positioning axis as of V6 (Page 4128)

See also

Creating a user program (Page 6267)

Programming notes (Page 6270)

Behavior of the Motion Control commands after POWER OFF and restart (Page 6272)

Monitoring active commands (Page 6272)

Error displays of the Motion Control statements (Page 6283)

13.2.9.2 Creating a user program (S7-1200)

In the section below you learn how to create a user program with the basic configuration for controlling your axis. All available axis functions are controlled using the Motion Control instructions to be inserted.

Requirement

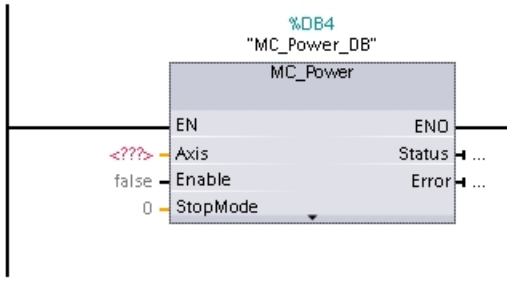
- The technology object has been created and configured without errors.

Before creating and testing the user program, it is advisable to test the function of the axis and the corresponding parts of the system with the axis command table.

Procedure

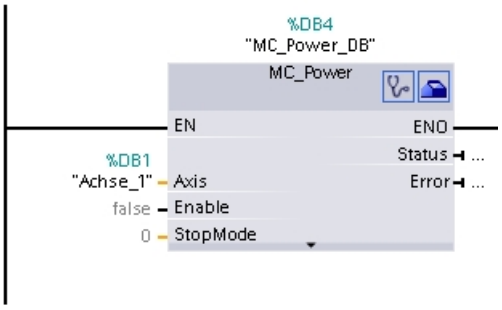
Proceed as follows to create the user program in accordance with the principles described below:

1. In the project tree, double-click your code block (the code block must be called in the cyclic program).
   The code block is opened in the programming editor and all available instructions are displayed.
2. Open the "Technology" category and the "Motion Control" and "S7-1200 Motion Control" folders.
3. Use a drag-and-drop operation to move the "MC_Power" instruction to the desired network of the code block.
   The dialog box for defining the instance DB opens.
4. In the next dialog box, select from the following alternatives:
   Single instance
   Click "Single instance" and select whether you want to define the name and number of the instance DB automatically or manually.
   Multi-instance
   Click "Multi-instance" and select whether you want to define the name of the multi-instance automatically or manually.
5. Click "OK".
   The Motion Control instruction "MC_Power" is inserted into the network.



Parameters marked with "<???>" must be initialized; default values are assigned to all other parameters.
Parameters displayed in black are required for use of the Motion Control instruction.

6. Select the technology object in the project tree and drag-and-drop it on <???>.



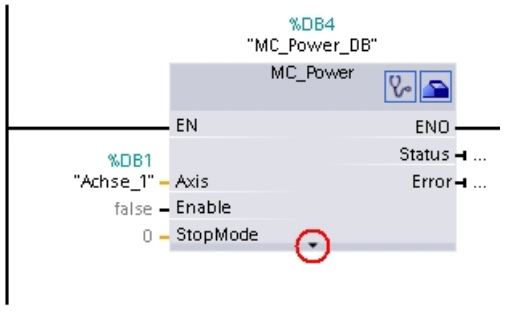
After the selection of the technology object data block, the following buttons are available:



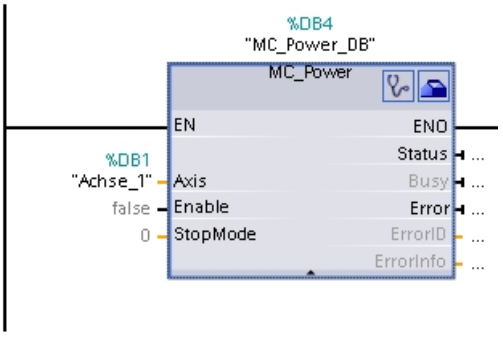
Click the stethoscope icon if you want to open the diagnostics dialog for the technology object.



Click the toolbox icon if you want to open the configuration view of the technology object.



Click the arrow down icon to view additional parameters of the Motion Control instruction.



The grayed-out parameters now visible can be used optionally.

7. Add your choice of Motion Control instructions in accordance with steps 3 to 6.

Result

You have created the basic configuration for axis control in the user program.

Initialize the input parameters of Motion Control instructions in other parts of the user program to initiate the desired jobs for the "Axis" technology object.

Evaluate the output parameters of the Motion Control instructions and the tags of the data block to track the initiated jobs and the status of the axis.

Refer to the detailed description for details on the parameters of Motion Control instructions.

See also

Overview of the Motion Control statements (Page 6266)

Programming notes (Page 6270)

Behavior of the Motion Control commands after POWER OFF and restart (Page 6272)

Monitoring active commands (Page 6272)

Error displays of the Motion Control statements (Page 6283)

13.2.9.3 Programming notes (S7-1200)

When creating your user program, note the following information:

- **Cyclic call of utilized motion control instructions**
  The current status of command execution is available via the output parameters of the motion control instruction. The status is updated with every call of the motion control instruction. Therefore, make sure that the utilized motion control instructions are called cyclically.
- **Transfer of parameter values of a motion control instruction**
  The parameter values pending for the input parameters are transferred with a positive edge at input parameter "Execute" when the block is called.
  The motion control command is started with these parameter values. Parameter values that are subsequently changed for the motion control instruction are not transferred until the next start of the motion control command.
  Exceptions to this are input parameters "StopMode" of motion control instruction "MC_Power" and "Velocity" of motion control instruction "MC_MoveJog". A change in the input parameter is also applied with "Enable" = TRUE or "JogForward" and "JogBackward".

- **Programming under consideration of the status information**
  In a stepwise execution of motion control jobs, make sure to wait for the active command to finish before starting a new command. Use the status messages of the motion control instruction and the "StatusBits" tag of the technology object to check for completion of the active command.
  In the examples below, observe the indicated sequence. Failure to observe the sequence will display an axis or command error.
  - **Axis enable with motion control instruction "MC_Power"**
    You must enable the axis before it can take on motion jobs. Use an AND operation of tag <Axis name>.StatusBits.Enable = TRUE with output parameter Status = TRUE of motion control instruction "MC_Power" to verify that the axis is enabled.
  - **Acknowledge error with motion control instruction "MC_Reset"**
    Prior to starting a motion control command, errors requiring acknowledgement must be acknowledged with "MC_Reset". Eliminate the cause of the error and acknowledge the error with motion control instruction "MC_Reset". Verify that the error has been successfully acknowledged before initiating a new command. For this purpose, use an AND operation of tag <Axis name>.StatusBits.Error = FALSE with output parameter Done = TRUE of motion control instruction "MC_Reset".
  - **Home axis with motion control instruction "MC_Home"**
    Before you can start an MC_MoveAbsolute command, the axis must be homed. Use an AND operation of tag <Axis name>.StatusBits.HomingDone = TRUE with output parameter Done = TRUE of motion control instruction "MC_Home" to verify that the axis has been homed.
- **Override of motion control command processing**
  Motion control jobs for moving an axis can also be executed as overriding jobs.
  If a new motion control command is started for an axis while another motion control command is active, the active command is overridden by the new command before the existing command is completely executed. The overridden command signals this using CommandAborted = TRUE in the motion control instruction. It is possible to override an active MC_MoveRelative command with a MC_MoveAbsolute command.
- **Avoiding multiple use of the same instance**
  All relevant information of a motion control command is stored in its instance.
  Do not start a new command using this instance, if you want to track the status of the current command. Use different instances if you want to track the commands separately. If the same instance is used for multiple motion control commands, the status and error information of the individual commands will overwrite each other.
  In the user program, each axis must be called with a separate call of the Motion Control instruction "MC_Power" with a separate instance data block.
- **Call of motion control instructions in different priority classes (run levels)**
  Motion Control instructions with the same instance may not be called in different priority classes without interlocking. To learn how to call locked motion control instructions, refer to "Tracking commands from higher priority classes (run levels) (Page 6294)".

See also

Overview of the Motion Control statements (Page 6266)

Creating a user program (Page 6267)

Behavior of the Motion Control commands after POWER OFF and restart (Page 6272)

Monitoring active commands (Page 6272)

Error displays of the Motion Control statements (Page 6283)

Tracking jobs from higher priority classes (execution levels) (Page 6294)

13.2.9.4 Behavior of the Motion Control commands after POWER OFF and restart (S7-1200)

A POWER OFF or CPU-STOP aborts all active motion control jobs. All CPU outputs, including pulse and direction outputs, are reset.

After a subsequent POWER ON or CPU restart (CPU RUN), the technology objects and the motion control jobs will be reinitialized.

All actual data of the technology objects as well as all status and error information of the previously active motion control jobs are reset to their initial values.

Before the axis can be reused, it must be enabled again using the Motion Control instruction "MC_Power". If homing is required, the axis must be homed again with Motion Control instruction "MC_Home". When an absolute encoder is used, homing is retained after POWER OFF.

See also

Overview of the Motion Control statements (Page 6266)

Creating a user program (Page 6267)

Programming notes (Page 6270)

Monitoring active commands (Page 6272)

Error displays of the Motion Control statements (Page 6283)

13.2.9.5 Monitoring active commands (S7-1200)

Monitoring active commands (S7-1200)

There are three typical groups for tracking active Motion Control commands:

- **Motion control instructions with output parameter "Done"**
- **Motion control instruction "MC_MoveVelocity"**
- **Motion control instruction "MC_MoveJog"**

Motion control instructions with "Done" output parameter (S7-1200)

Motion control instructions with the output parameter "Done" are started via input parameter "Execute" and have a defined conclusion (for example, with Motion Control instruction "MC_Home": Homing was successful). The command is complete and the axis is at a standstill.

The commands of the following Motion Control instructions have a defined conclusion:

- MC_Reset
- MC_Home
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_CommandTable (technology object as of V2)
- MC_ChangeDynamic (technology object as of V2)
- MC_WriteParam (as of technology object V4)
- MC_ReadParam (as of technology object V4)

The output parameter "Done" indicates the value TRUE, if the command has been successfully completed.

The output parameters "Busy", "CommandAborted", and "Error" signal that the command is still being processed, has been aborted or an error is pending. The Motion Control instruction "MC_Reset" cannot be aborted and thus has no "CommandAborted" output parameter. The Motion Control instruction "MC_ChangeDynamic" is completed immediately and therefore has no "Busy" or "CommandAborted" output parameters.

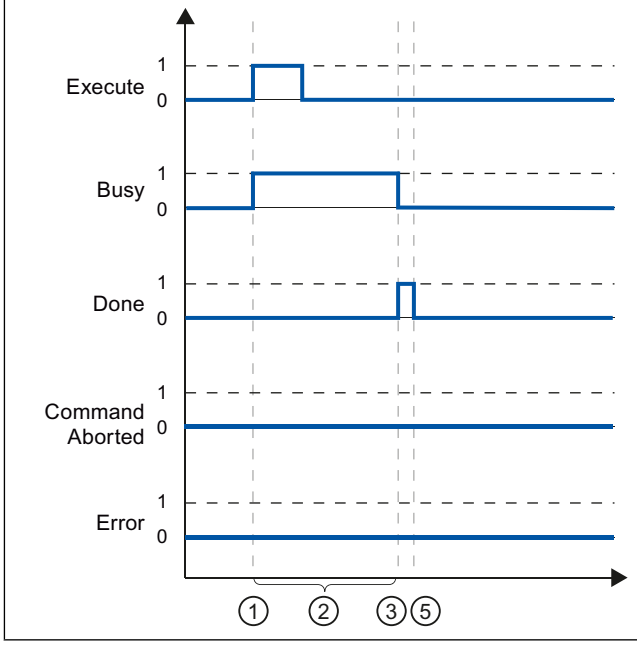
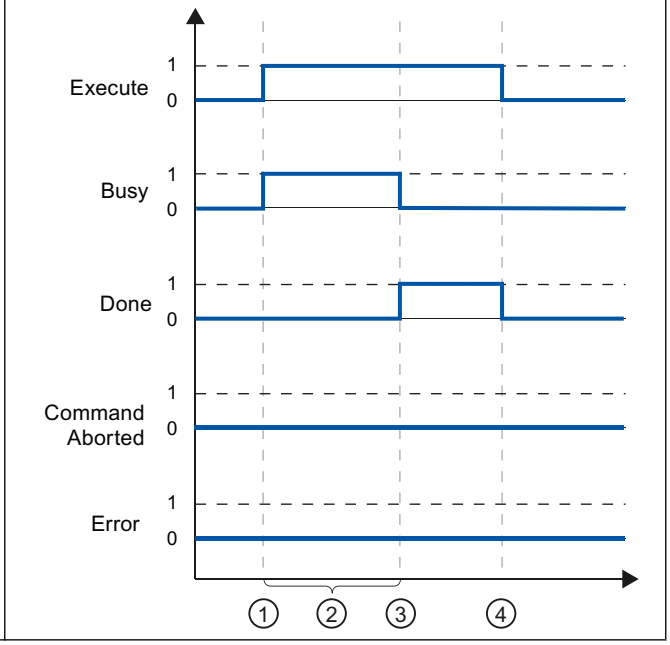
During execution of the Motion Control command, the output parameter "Busy" indicates the value TRUE. If the command has been completed, aborted, or stopped by an error, the output parameter "Busy" changes its value to FALSE. This change occurs regardless of the signal at input parameter "Execute".

Output parameters "Done", "CommandAborted", and "Error" indicate the value TRUE for at least one cycle. These status messages are latched while input parameter "Execute" is set to TRUE.

The behavior of the status bits is presented below for various example situations:

Complete execution of command

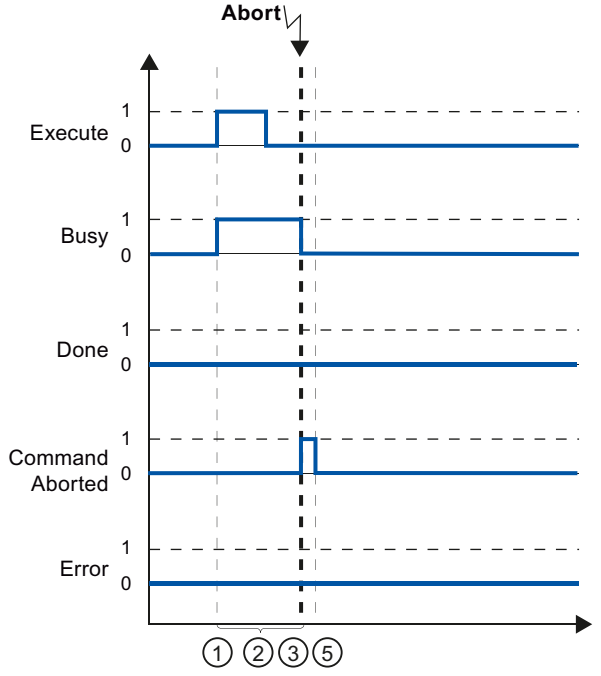
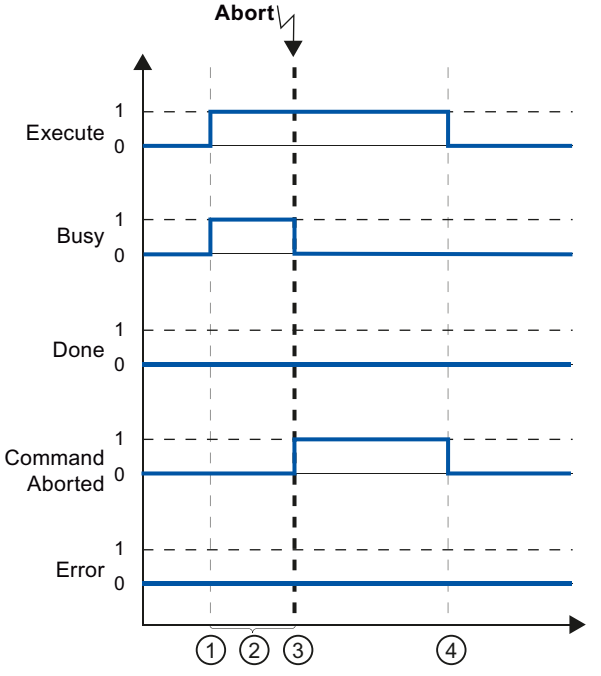
If the Motion Control command has been completely executed by the time of its conclusion, this is indicated by the value TRUE in output parameter "Done". The signal status of the input parameter "Execute" influences the display duration at the output parameter "Done":

"Execute" changes its value to FALSE **during processing** of the command	"Execute" changes its value to FALSE **after completion** of the command
	

①	The command is started with a positive edge at the input parameter "Execute". Depending on the programming, "Execute" can still be reset to the value FALSE during the command, or the value TRUE can be retained until after completion of the command.
②	While the command is active, the output parameter "Busy" indicates the value TRUE.
③	With conclusion of the command (for example, for Motion Control instruction "MC_Home": Homing was successful), output parameter "Busy" changes to FALSE and "Done" to TRUE.
④	If "Execute" retains the value TRUE until after completion of the command, then "Done" also remains TRUE and changes its value to FALSE together with "Execute".
⑤	If "Execute" has been set to FALSE before the command is complete, "Done" indicates the value TRUE for only one execution cycle.

Abort command

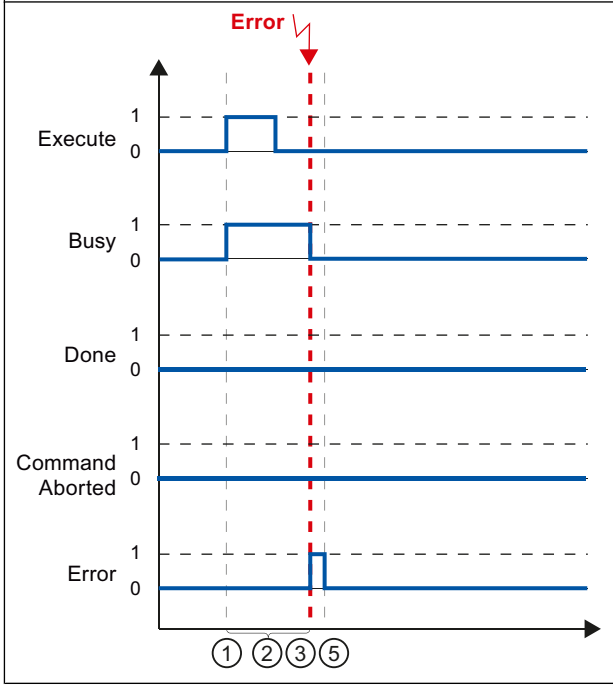
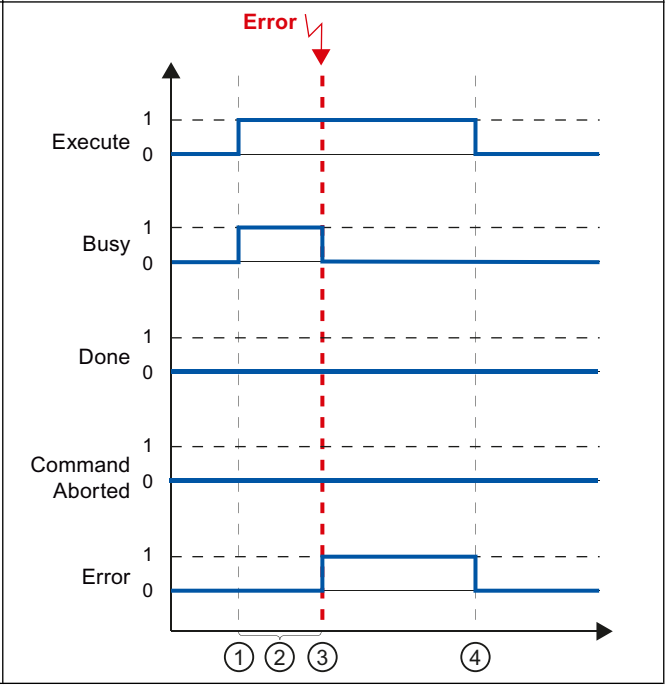
If the Motion Control command is aborted during execution, this is indicated by the value TRUE in output parameter "CommandAborted". The signal status of the input parameter "Execute" influences the display duration at the output parameter "CommandAborted":

"Execute" changes its value to FALSE **before** the command is aborted.	"Execute" changes its value to FALSE **after** the command is aborted.
	

①	The command is started with a positive edge at the input parameter "Execute". Depending on the programming, "Execute" can still be reset to the value FALSE during the command, or the value TRUE can be retained until after completion of the command.
②	While the command is active, the output parameter "Busy" indicates the value TRUE.
③	During command execution, the command is aborted by another Motion Control command. If the command is aborted, output parameter "Busy" changes to FALSE and "CommandAborted" to TRUE.
④	If "Execute" retains the value TRUE until after the command is aborted, then "CommandAborted" also remains TRUE and changes its value to FALSE together with "Execute".
⑤	If "Execute" has been set to FALSE before the command is aborted, "CommandAborted" indicates the value TRUE for only one execution cycle.

Error during command execution

If an error occurs during execution of the Motion Control command, this is indicated by the value TRUE in the output parameter "Error". The signal status of the input parameter "Execute" influences the display duration at the output parameter "Error":

"Execute" changes its value to FALSE **before** the error occurs	"Execute" changes its value to FALSE **after** the error occurs
	

①	The command is started with a positive edge at the input parameter "Execute". Depending on the programming, "Execute" can still be reset to the value FALSE during the command, or the value TRUE can be retained until after completion of the command.
②	While the command is active, the output parameter "Busy" indicates the value TRUE.
③	An error occurred during command execution. When the error occurs, the output parameter "Busy" changes to FALSE and "Error" to TRUE.
④	If "Execute" retains the value TRUE until after the error occurs, then "Error" also remains TRUE and only changes its value to FALSE together with "Execute".
⑤	If "Execute" has been set to FALSE before the error occurs, "Error" indicates the value TRUE for only one execution cycle.

Motion control instruction MC_MoveVelocity (S7-1200)

A "MC_MoveVelocity" command is started with a positive edge at the "Execute" parameter. The command objective is fulfilled when the assigned velocity is reached and the axis travels at constant velocity. When the assigned velocity is reached and maintained, this is indicated in the "InVelocity" parameter with the value TRUE.

The motion of the axis can, for example, be stopped with an "MC_Halt" command.

The output parameters "Busy", "CommandAborted", and "Error" signal that the command is still being processed, has been aborted or an error is pending.

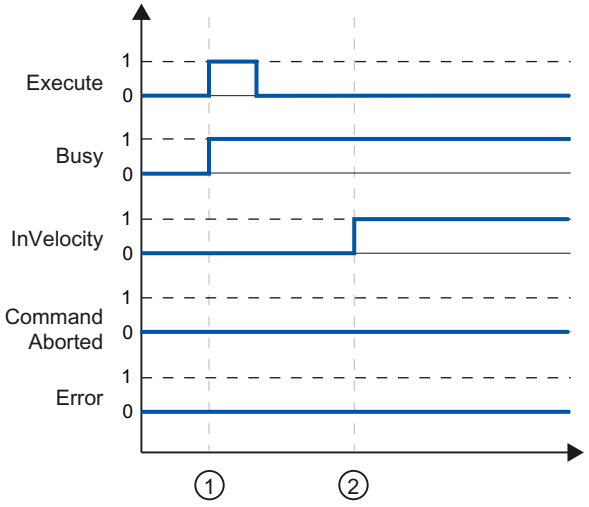
During execution of the Motion Control command, the output parameter "Busy" indicates the value TRUE. If the command has been stopped by another command or by an error, the output parameter "Busy" changes its value to FALSE. This change occurs regardless of the signal at the input parameter "Execute".

The output parameters "CommandAborted" and "Error" show the value TRUE for at least one cycle. These status messages are latched while input parameter "Execute" is set to TRUE.

The behavior of the status bits is presented below for various example situations:

The parameterized velocity is reached

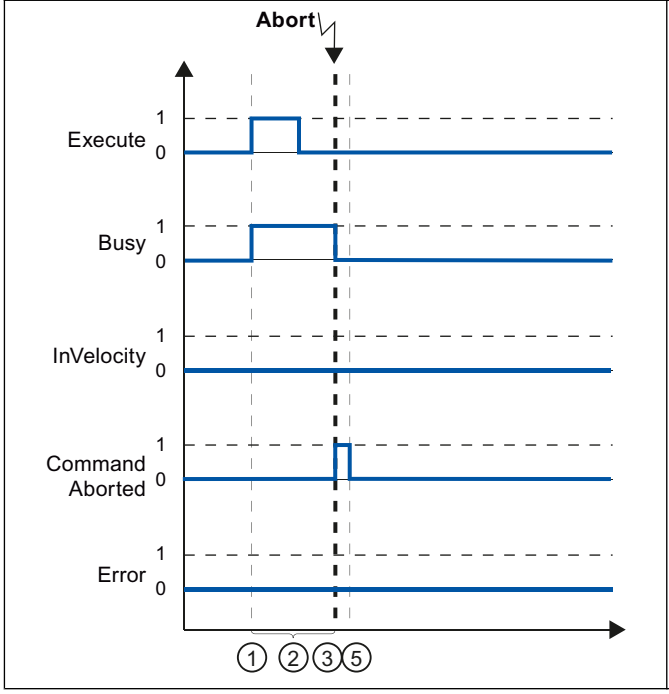
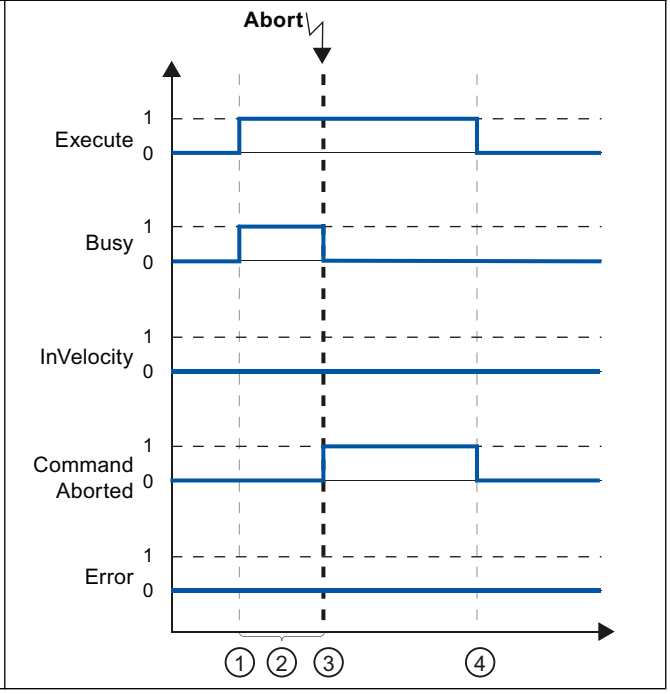
If the Motion Control command has been executed by the time the parameterized velocity is reached, this is indicated by the value TRUE in output parameter "InVelocity". The parameter "Execute" has no effect on the indication duration in the "InVelocity" parameter.



①	The job is started with a positive edge at the "Execute" parameter. Depending on the programming, "Execute" can be reset to the FALSE value before or after the parameterized velocity has been reached. While the job is active, the parameter "Busy" shows the value TRUE.
②	When the assigned velocity is reached, the "InVelocity" parameter changes to TRUE. The "Busy" and "InVelocity" parameters retain the TRUE value until the "MC_MoveVelocity" command is overridden by another Motion Control command or stopped by an error.

The command is aborted prior to reaching the parameterized velocity

If the Motion Control command is aborted before the parameterized velocity is reached, this is indicated by the value TRUE in output parameter "CommandAborted". The signal status of input parameter "Execute" influences the display duration at the output parameter "CommandAborted".

"Execute" changes its value to FALSE **before** the command is aborted.	"Execute" changes its value to FALSE **after** the command is aborted.
	

①	The command is started with a positive edge at the input parameter "Execute". Depending on the programming, "Execute" can still be reset to the value FALSE during the command, or the value TRUE can be retained until after the command is aborted.
②	While the command is active, the output parameter "Busy" indicates the value TRUE.
③	During command execution, the command is aborted by another Motion Control command. If the command is aborted, output parameter "Busy" changes to FALSE and "CommandAborted" to TRUE.
④	If "Execute" retains the value TRUE until after the command is aborted, then "CommandAborted" also remains TRUE and changes its status to FALSE together with "Execute".
⑤	If "Execute" has been reset to FALSE before the command is aborted, "CommandAborted" indicates the value TRUE for only one execution cycle.

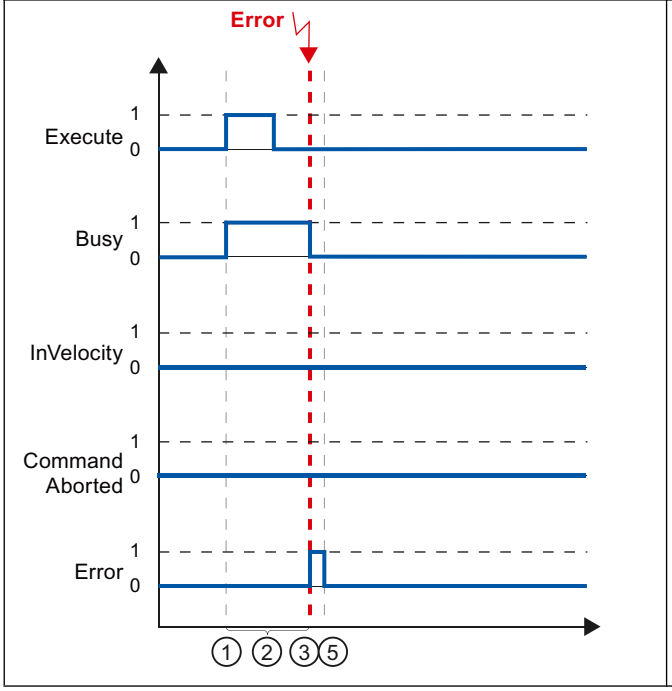
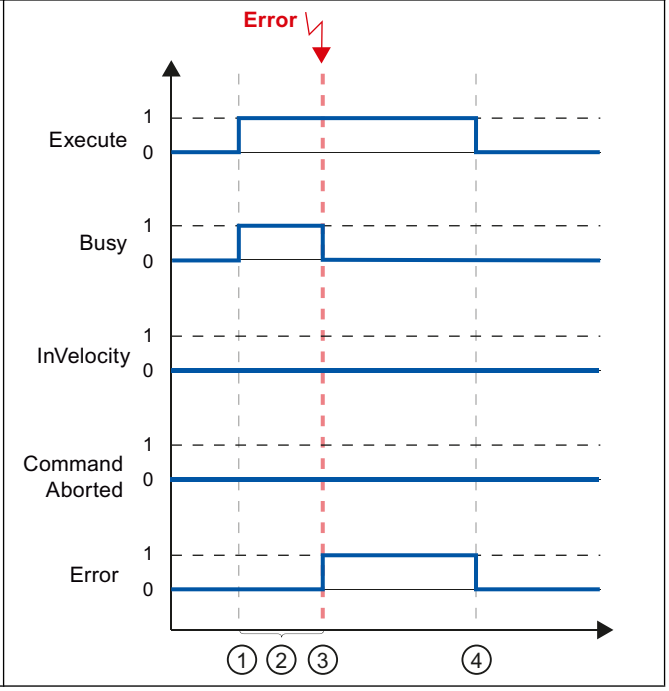
Note

Under the following conditions, an abort is not indicated in output parameter "CommandAborted":

The parameterized velocity has been reached, input parameter "Execute" has the value FALSE, and a new Motion Control command is initiated.

An error has occurred prior to reaching the parameterized velocity

If an error occurs during execution of the Motion Control command before the parameterized velocity has been reached, this is indicated by the value TRUE in the output parameter "Error". The signal status of the input parameter "Execute" influences the display duration at the output parameter "Error":

"Execute" changes its value to FALSE **before** the error occurs	"Execute" changes its value to FALSE **after** the error occurs
	

①	The command is started with a positive edge at the input parameter "Execute". Depending on the programming, "Execute" can still be reset to the value FALSE during the command, or the value TRUE can be retained until after the error has occurred.
②	While the command is active, the output parameter "Busy" indicates the value TRUE.
③	An error occurred during command execution. When the error occurs, the output parameter "Busy" changes to FALSE and "Error" to TRUE.
④	If "Execute" retains the value TRUE until after the error has occurred, then "Error" also remains TRUE and only changes its status to FALSE together with "Execute".
⑤	If "Execute" has been reset to FALSE before the error occurs, "Error" indicates the value TRUE for only one execution cycle.

Note

Under the following conditions, an error is not indicated in output parameter "Error":

The parameterized velocity has been reached, input parameter "Execute" has the value FALSE, and an axis error occurs (software limit switch is approached, for example).

The error of the axis is only indicated in the "MC_Power" Motion Control instruction.

Motion control instruction MC_MoveJog (S7-1200)

The commands of Motion Control instruction "MC_MoveJog" implement a jog operation.

The motion control commands "MC_MoveJog" do not have a defined end. The command objective is fulfilled when the parameterized velocity is reached for the first time and the axis travels at constant velocity. When the parameterized velocity is reached, this is indicated by the value TRUE in output parameter "InVelocity".

The order is complete when input parameter "JogForward" or "JogBackward" has been set to the value FALSE and the axis has come to a standstill.

The output parameters "Busy", "CommandAborted", and "Error" signal that the command is still being processed, has been aborted or an error is pending.

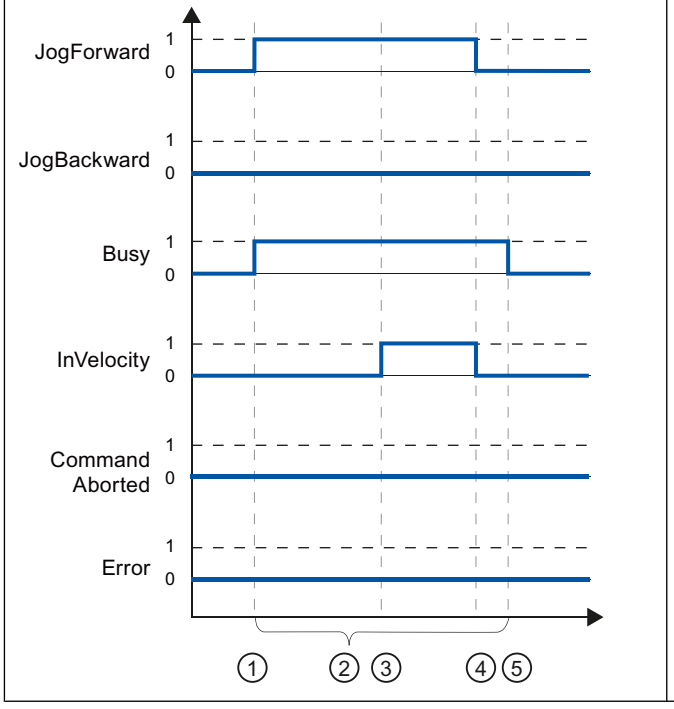
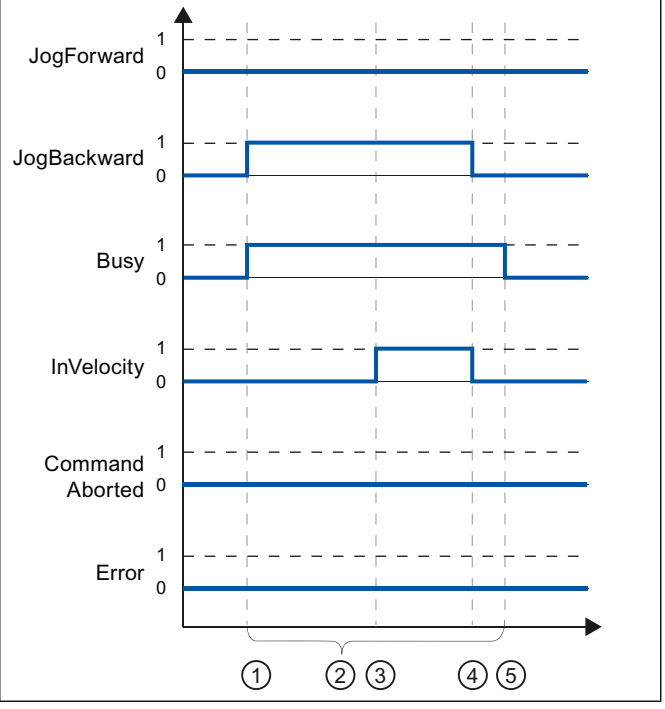
During processing of the motion control command, the output parameter "Busy" indicates the value TRUE. If the command has been completed, aborted, or stopped by an error, the output parameter "Busy" changes its value to FALSE.

The output parameter "InVelocity" indicates the status TRUE, as long as the axis is moving at the parameterized velocity. The output parameters "CommandAborted" and "Error" indicate the status for at least one cycle. These status messages are latched as long as either input parameter "JogForward" or "JogBackward" is set to TRUE.

The behavior of the status bits is presented below for various example situations:

The parameterized velocity is reached and maintained

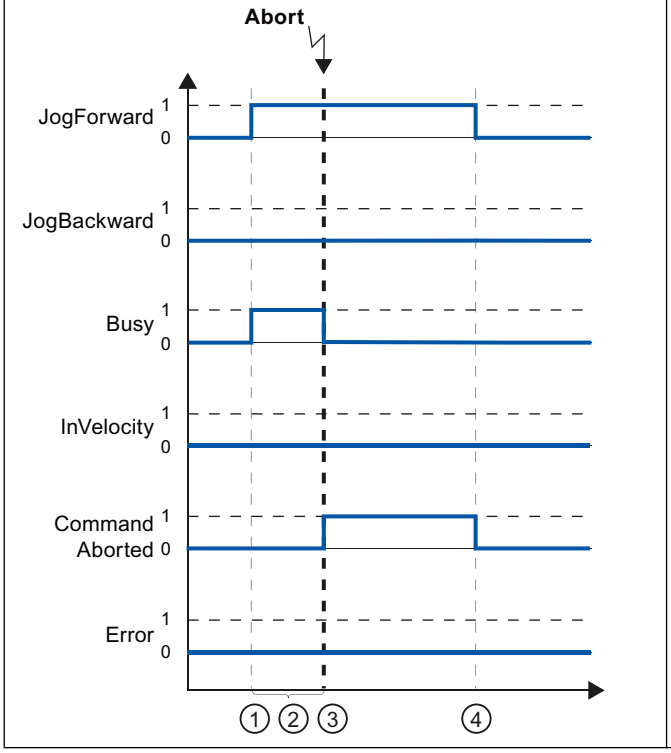
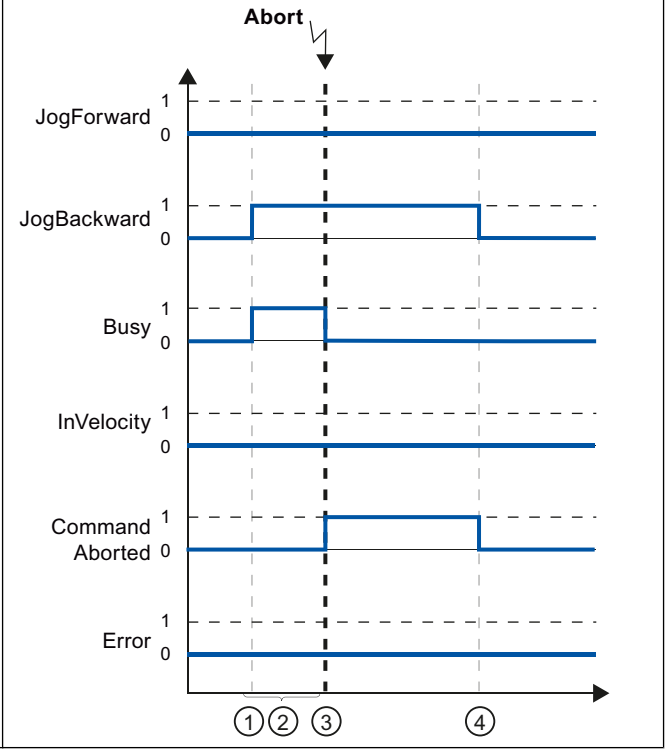
If the motion control command has been executed by the time the parameterized velocity is reached, this is indicated by the value TRUE in output parameter "InVelocity".

Jog mode is controlled by input parameter "JogForward"	Jog mode is controlled by input parameter "JogBackward".
	

①	The command is started with a positive edge at the input parameter "JogForward" or "JogBackward".
②	While the command is active, the output parameter "Busy" indicates the value TRUE.
③	When the parameterized velocity is reached, the output parameter "InVelocity" changes to TRUE.
④	When the input parameter "JogForward" or "JogBackward" is reset to the value FALSE, the axis motion ends. The axis starts to decelerate. As a result, the axis no longer moves at constant velocity and the output parameter "InVelocity" changes its status to FALSE.
⑤	If the axis has come to a standstill, the motion control command is complete and the output parameter "Busy" changes its value to FALSE.

The command is aborted during execution

If the motion control command is aborted during execution, this is indicated by the value TRUE in output parameter "CommandAborted". The behavior is independent of whether or not the parameterized velocity has been reached.

Jog mode is controlled by input parameter "JogForward".	Jog mode is controlled by input parameter "JogBackward".
	

①	The command is started with a positive edge at the input parameter "JogForward" or "JogBackward".
②	While the command is active, the output parameter "Busy" indicates the value TRUE.
③	During command execution, the command is aborted by another motion control command. If the command is aborted, output parameter "Busy" changes to FALSE and "CommandAborted" to TRUE.
④	When the input parameter "JogForward" or "JogBackward" is reset to the value FALSE, the output parameter "CommandAborted" changes its value to FALSE.

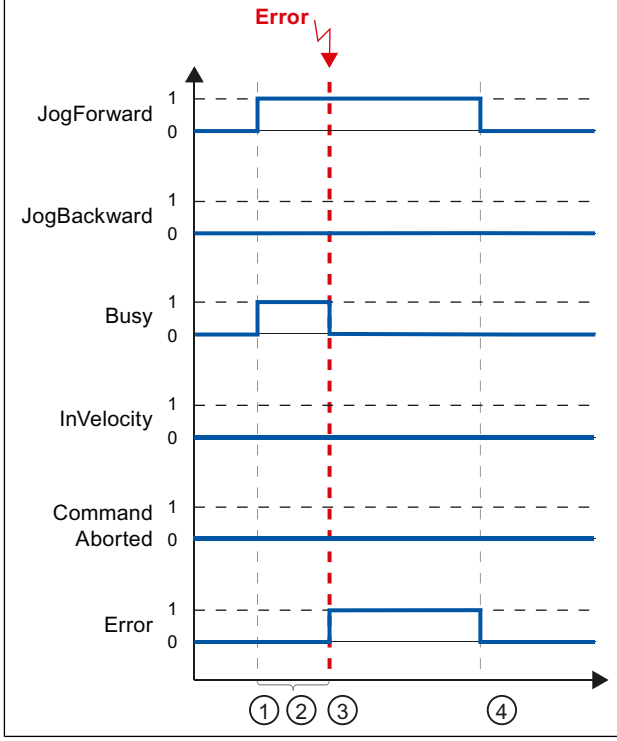
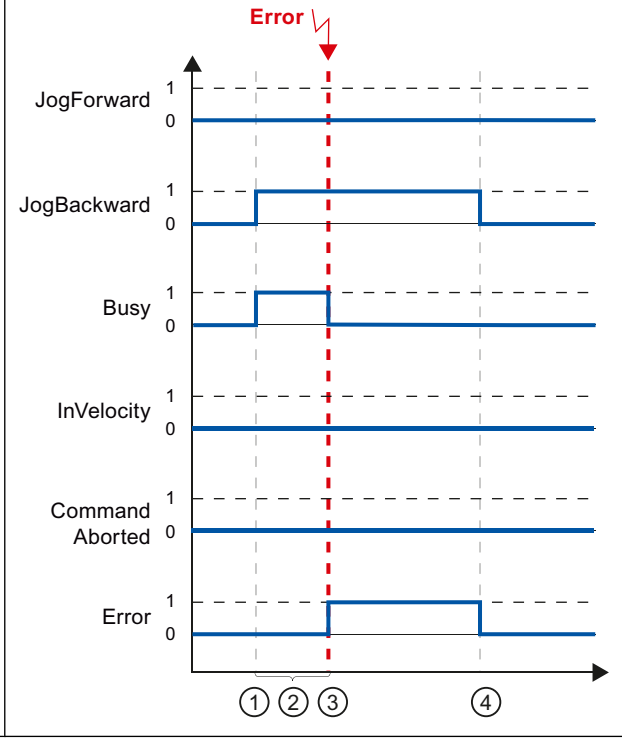
Note

The command abort is indicated in the output parameter "CommandAborted" for only one execution cycle, if all conditions below are met:

The input parameters "JogForward" and "JogBackward" have the value FALSE (but the axis is still decelerating) and a new motion control command is initiated.

An error has occurred during command execution

If an error occurs during execution of the motion control command, this is indicated by the value TRUE in output parameter "Error". The behavior is independent of whether or not the parameterized velocity has been reached.

Jog mode is controlled by input parameter "JogForward".	Jog mode is controlled by input parameter "JogBackward".
	

①	The command is started with a positive edge at the input parameter "JogForward" or "JogBackward".
②	While the command is active, the output parameter "Busy" indicates the value TRUE.
③	An error occurred during command execution. When the error occurs, the output parameter "Busy" changes to FALSE and "Error" to TRUE.
④	When the input parameter "JogForward" or "JogBackward" is reset to the value FALSE, the output parameter "Error" changes its value to FALSE.

Note

An error occurrence is indicated in the output parameter "Error" for only one execution cycle, if all the conditions below are met:

The input parameters "JogForward" and "JogBackward" have the value FALSE (but the axis is still decelerating) and a new error occurs (software limit switch is approached, for example).

13.2.9.6 Error displays of the Motion Control statements (S7-1200)

The Motion Control instructions indicate any errors in motion control commands and the technology object at the output parameters "Error", "ErrorID" and "ErroInfo" of the Motion Control instructions.

Error display at output parameters "Error", "ErrorID" and "ErrorInfo"

If the output parameter "Error" indicates the value TRUE, the complete command, or portions thereof, could not be executed. The cause of the error is indicated by the value in output parameter "ErrorID". Detailed information about the cause of the error is returned by the value in output parameter ErrorInfo. We distinguish between the following error classes for error indication:

- **Operating error with axis stop (for example, "HW limit switch was approached")**
  Operating errors with axis stop are errors that occur during runtime of the user program. If the axis is in motion, it is stopped with the configured deceleration or emergency stop deceleration, depending on the error. The errors are indicated in the error-triggering Motion Control instruction and in the Motion Control instruction "MC_Power".
- **Operating error without axis stop (for example, "Axis is not homed")**
  Operating errors without axis stop are errors that occur during runtime of the user program. If the axis is in motion, the motion is continued. The errors are only indicated in the Motion Control instruction which triggers the error.
- **Configuration error in Motion Control instruction (for example "Incorrect value in parameter "Velocity")**
  Parameterization errors occur when incorrect information is specified in the input parameters of Motion Control instructions. If the axis is in motion, the motion is continued. The errors are only indicated in the Motion Control instruction which triggers the error.
- **Configuration error on technology object "Axis" (for example, "Value for "Acceleration" is invalid")**
  A configuration error exists if one or more parameters are incorrectly configured in the axis configuration or if editable configuration data have been modified incorrectly during runtime of the program. An axis in motion is stopped with the configured emergency stop deceleration. The error is indicated in the error-triggering Motion Control instruction and in Motion Control instruction "MC_Power".

- **Configuration error on technology object "Command table" (for example "Value for "Velocity" is invalid")**
  There is a configuration error if one or more parameters are incorrectly set in the axis command table or if programmable configuration data have been modified incorrectly during runtime of the program. If the axis is in motion, the motion is continued. The errors are only indicated in the "MC_CommandTable" Motion Control instruction.
- **Internal error**
  When an internal error occurs, the axis is stopped. The errors are indicated in the error-triggering Motion Control instruction and, in some cases, in the Motion Control instruction "MC_Power".

A detailed description of the ErrorIDs and ErrorInfos, as well as their remedies, is available in the Appendix (Page 6386).

See also

Overview of the Motion Control statements (Page 6266)

Creating a user program (Page 6267)

Programming notes (Page 6270)

Behavior of the Motion Control commands after POWER OFF and restart (Page 6272)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

Monitoring active commands (Page 6272)

13.2.9.7 Restart of technology objects (S7-1200)

Description

After the CPU is switched on, or after technology objects are downloaded into the CPU, the system automatically initializes the technology objects with the start values from the technology data block. If restart-relevant changes are detected during a reload into the CPU, a restart of the technology object is automatically performed.

If restart-relevant data have been changed in RUN mode by the user program, then the technology object must be reinitialized by the user in order for the changes to be used.

If changes in the technology data block should also be retained after the restart of the technology object, then you must write the changes to the start value in load memory using the extended instruction "WRIT_DBL".

Restart necessary

If a restart of the technology object is necessary, this will be indicated under "Technology object > Diagnostics > Status and error bits > Status messages > Restart required", and in the tags of the technology object <Axis name>.StatusBits.RestartRequired.

Restarting a technology object

A restart of the technology object is triggered by the user by means of the "MC_Reset" Motion Control instruction, with parameter "Restart" = TRUE.

A restart resets the "Homed" status of a technology object with incremental actual values (<Axis name>.StatusBits.HomingDone).

13.2.10 Axis - Diagnostics (S7-1200)

13.2.10.1 Status and error bits (technology objects as of V4) (S7-1200)

You use the "Status and error bits" diagnostic function to monitor the most important status and error messages for the axis in the TIA Portal. The diagnostic function display is available in online mode in "Manual control" mode and in "Automatic control" when the axis is active. The status error messages have the following meaning:

Status messages

Status message - Axis	Description
Enabled	The axis is enabled and ready to be controlled via Motion Control commands. (Tag of technology object: <Axis name>.StatusBits.Enable)
Homed	The axis is homed and is capable of executing absolute positioning commands of Motion Control instruction "MC_MoveAbsolute". The axis does not have to be homed for relative positioning. Special situations: • During active homing, the status is FALSE. • If a homed axis undergoes passive homing, the status is set to TRUE during passive homing. (Tag of technology object: <Axis name>.StatusBits.HomingDone)
Axis error	An error has occurred in the "Axis" technology object. Additional information about the error is available in automatic control at the ErrorID and ErrorInfo parameters of the Motion Control instructions. In manual mode, the "Error message" box of the axis control panel displays detailed information about the cause of error. (Tag of technology object: <Axis name>.StatusBits.Error)
Control panel active	The "Manual control" mode was enabled in the axis control panel. The axis control panel has control priority over the "Axis" technology object. The axis cannot be controlled from the user program. (Tag of technology object: <Axis name>.StatusBits.ControlPanelActive)
Restart required	A modified configuration of the axis was downloaded to the load memory in CPU RUN mode. To download the modified configuration to the work memory, you need to restart the axis. Use the Motion Control instruction MC_Reset to do this. (Tag of technology object: <Axis name>.StatusBits.RestartRequired)

Status message - Drive	Description
Ready	The drive is ready for operation. (Tag of technology object: <Axis name>.StatusBits.DriveReady)
Drive error	The drive has reported an error due to loss of its "Drive ready" signal. (Tag of technology object: <Axis name>.ErrorBits.DriveFault)

Status message - Motion	Description
Standstill	The axis is at a standstill. (Tag of technology object: <Axis name>.StatusBits.StandStill)
Acceleration	The axis accelerates. (Tag of technology object: <Axis name>.StatusBits.Accelerating)
Constant velocity	The axis travels at constant velocity. (Tag of technology object: <Axis name>.StatusBits.ConstantVelocity)
Deceleration	The axis decelerates (slows down). (Tag of technology object: <Axis name>.StatusBits.Decelerating)

Status message - Motion type	Description
Positioning	The axis executes a positioning command of the Motion Control instruction "MC_MoveAbsolute", "MC_MoveRelative" or the axis control panel. (Tag of technology object: <Axis name>.StatusBits.PositioningCommand)
Travel with velocity specification	The axis executes a command with velocity specification of the Motion Control instruction "MC_MoveVelocity", "MC_MoveJog" or the axis control panel. (Tag of technology object: <Axis name>.StatusBits.VelocityCommand)
Homing	The axis executes a homing command of the Motion Control instruction "MC_Home" or the axis control panel. (Tag of technology object: <Axis name>.StatusBits.HomingCommand)
Command table active	The axis is controlled by Motion Control instruction "MC_CommandTable". (Tag of technology object: <Axis name>.StatusBits.CommandTableActive)

Limit switch status messages

Limit switch status message	Description
Low SW limit switch has been reached	A software limit switch was reached or exceeded. (Tag of technology object: <Axis name>.StatusBits.SWLimitMinActive)
High SW limit switch has been reached	A hardware limit switch was reached or exceeded. (Tag of technology object: <Axis name>.StatusBits.SWLimitMaxActive)
Low HW limit switch was reached	The low hardware limit switch was reached or exceeded. (Tag of technology object: <Axis name>.StatusBits.HWLimitMinActive)
High HW limit switch was reached	The high hardware limit switch was reached or exceeded. (Tag of technology object: <Axis name>.StatusBits.HWLimitMaxActive)

Error messages

Error message	Description
SW limit switch has been reached	A software limit switch was reached or exceeded. (Tag of technology object: <Axis name>.ErrorBits.SWLimit)
HW limit switch has been reached	A hardware limit switch was reached or exceeded. (Tag of technology object: <Axis name>.ErrorBits.HWLimit)
Invalid direction of motion	The motion direction of the command does not match the configured motion direction. (Tag of technology object: <Axis name>.ErrorBits.DirectionFault)
PTO already in use	A second axis is using the same PTO (Pulse Train Output) and HSC (High Speed Counter) and is enabled with "MC_Power". (Tag of technology object: <Axis name>.ErrorBits.HWUsed)
Encoder	Error in the encoder system. (Tag of technology object: <Axis name>.ErrorBits.SensorFault)
Data exchange	Error in communication with a connected device. (Tag of technology object: <Axis name>.ErrorBits.CommunicationFault)
Positioning	The axis was not correctly positioned at the end of a positioning motion. (Tag of technology object: <Axis name>.ErrorBits.PositionigFault)
Following error	The maximum permitted following error was exceeded. (Tag of technology object: <Axis name>.ErrorBits.FollowingErrorFault)
Encoder values are invalid	The encoder values are invalid. (Tag of technology object: <Axis name>.StatusSensor.State)
Configuration error	The "Axis" technology object was incorrectly configured or editable configuration data were modified incorrectly during runtime of the user program. (Tag of technology object: <Axis name>.ErrorBits.ConfigFault)
Internal error	An internal error has occurred. (Tag of technology object: <Axis name>.ErrorBits.SystemFault)

The output window below shows the first reported and still unacknowledged error.

See also

StatusBits tags V4...5 (Page 6463)

ErrorBits tags V4...5 (Page 6468)

Diagnostics - Status and error bits ("Axis" technology object V1...3) (Page 6383)

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

Motion status (Page 6288)

13.2.10.2 Motion status (S7-1200)

Use the "Motion status" diagnostic function to monitor the motion status of the axis in the TIA Portal. The diagnostic function display is available in online mode in "Manual control" mode and in "Automatic control" when the axis is active. The displayed status information has the following meaning:

Status	Description
Current position	The "Current position" box indicates the current axis position. If the axis is not homed, the value indicates the position value relative to the enable position of the axis. (Tag of technology object: <Axis name>.Position)
Current velocity	The "Current velocity" box indicates the current axis velocity. (Tag of technology object: <Axis name>.Velocity)
Target position	The "Target position" box indicates the current target position of an active positioning command or of the axis command table. The value of the "Target position" is only valid during execution of a positioning command. (Tag of technology object: <Axis name>.StatusPositioning.TargetPosition)
Remaining travel distance	The "Remaining travel distance" box indicates the travel distance currently remaining for an active positioning command or the axis command table. The "Remaining travel distance" value is only valid during execution of a positioning command. (Tag of technology object: <Axis name>.StatusPositioning.Distance)

See also

Position tag V4...5 (Page 6436)

Velocity tag V4...5 (Page 6437)

StatusPositioning tags V4...5 (Page 6460)

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

Position tag as of V6 (Page 6328)

Velocity tag as of V6 (Page 6329)

MotionStatus tags V1...3 (Page 6430)

Status and error bits (technology objects as of V4) (Page 6285)

13.2.10.3 Dynamics settings (S7-1200)

Use the "Dynamics settings" diagnostic function to monitor the dynamic limits of the axis in the TIA Portal. The diagnostic function display is available in online mode in "Manual control" mode and in "Automatic control" when the axis is active. The displayed status information has the following meaning:

Dynamic limit	Description
Acceleration	The "Acceleration" box indicates the currently configured acceleration of the axis. (Tag of technology object: <Axis name>.DynamicDefaults.Acceleration)
Deceleration	The "Deceleration" box indicates the currently configured deceleration of the axis. (Tag of technology object: <Axis name>.DynamicDefaults.Deceleration)

Dynamic limit	Description
Emergency deceleration	The "Emergency deceleration" box indicates the currently configured emergency stop deceleration of the axis. (Tag of technology object: <Axis name>.DynamicDefaults.EmergencyDeceleration)
Jerk (axis technology object as of V2)	The "Velocity" box indicates the current axis step velocity configured. (Tag of technology object: <Axis name>.DynamicDefaults.Jerk)

See also

DynamicDefaults tags V4...5 (Page 6451)

Compatibility list of variables V1...3 <-> V4...5 (Page 6172)

13.2.10.4 PROFIdrive frame (S7-1200)

The "Technology object > Diagnostics > PROFIdrive telegram" diagnostics function is used in the TIA Portal to monitor the PROFIdrive telegrams returned by the drive and encoder. The display of the Diagnostics function is available in online operation.

"Drive" area

This area displays the following parameters contained in the PROFIdrive telegram from the drive to the controller:

- Status words "SW1" and "SW2"
- The setpoint speed that was output to the drive (NSET)
- The actual speed that was signaled from the drive (NACT)

"Encoder" area

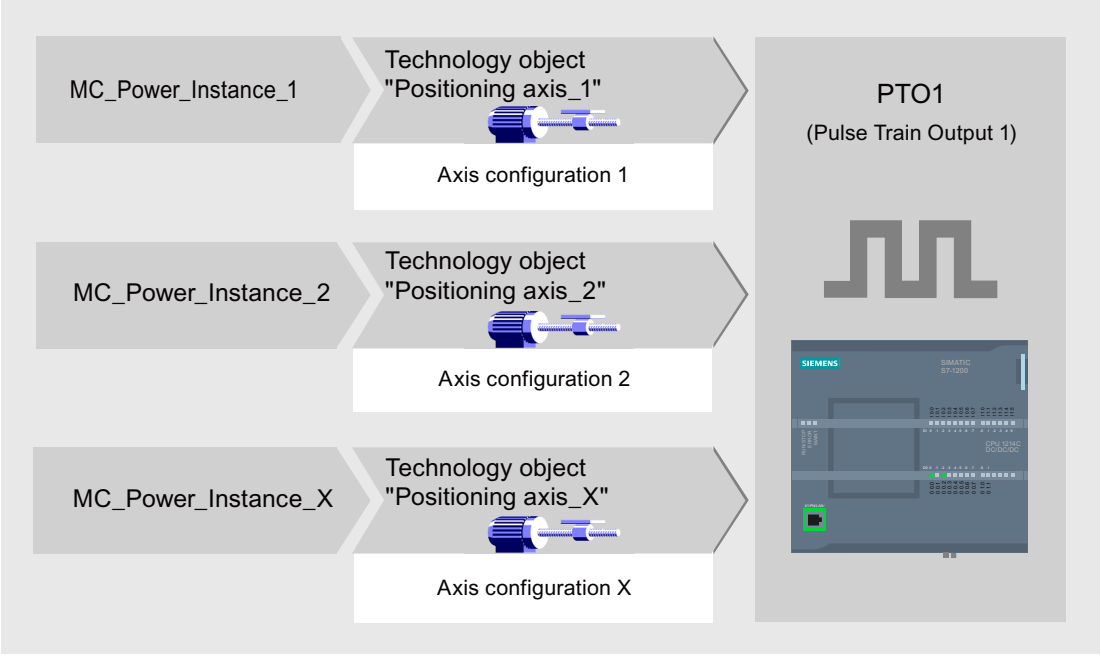
This area displays the following parameters contained in the PROFIdrive telegram from the encoder to the controller:

- Status word "G1_ZSW"
- The actual position value "G1_XIST1" (cyclic actual encoder value)
- The actual position value "G1_XIST2" (absolute value of the encoder)

13.2.11 Appendix (S7-1200)

13.2.11.1 Using multiple axes with the same PTO (S7-1200)

Use the Motion Control functionality of the CPU S7-1200 to run multiple positioning axis technology objects with the same PTO (Pulse Train Output) and thus with the same CPU outputs. This is appropriate, for example, if different axis configurations are to be used for different production sequences via one PTO. As described below, it is possible to switch between these axis configurations as often as necessary. The following diagram presents the basic functional relationships:

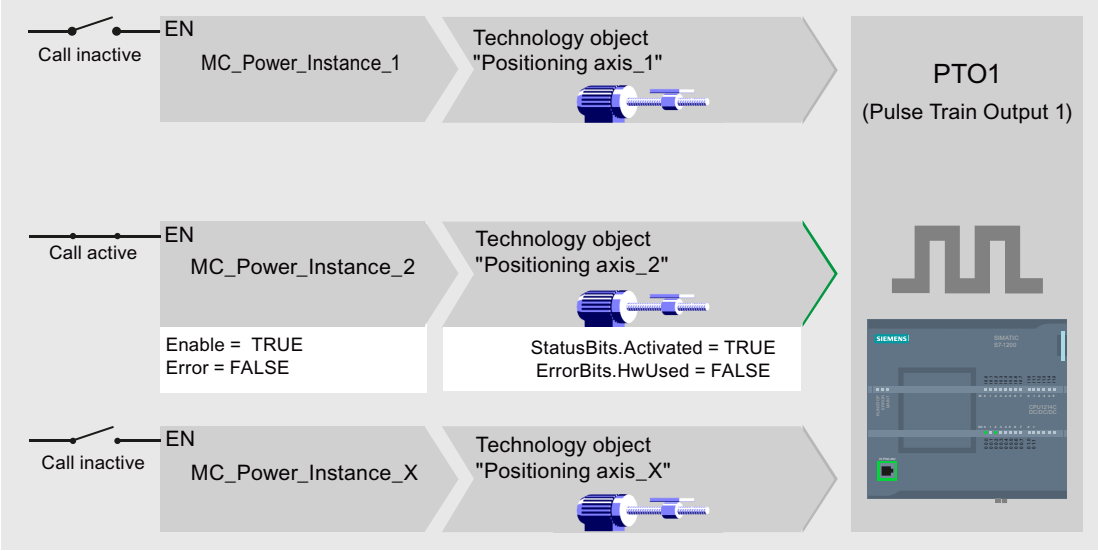


In this example, several positioning axis technology objects, each with its own axis configuration, use the same PTO. Each axis must be called in the user program with a separate call of Motion Control instruction "MC_Power" with a separate instance data block. Only one axis at a time may use the PTO. The axis that is currently using the PTO indicates this with tag <Axis name>.StatusBits.Activated = TRUE.

Switchover of the positioning axis technology object

The program scheme described below shows you how to switch between different technology objects and, thus, between different axis configurations. To use the same PTO with multiple axes without error indications, only the Motion Control instructions of the axis currently being used may be called.

The following diagram presents this principle using Motion Control instruction "MC_Power" as an example:



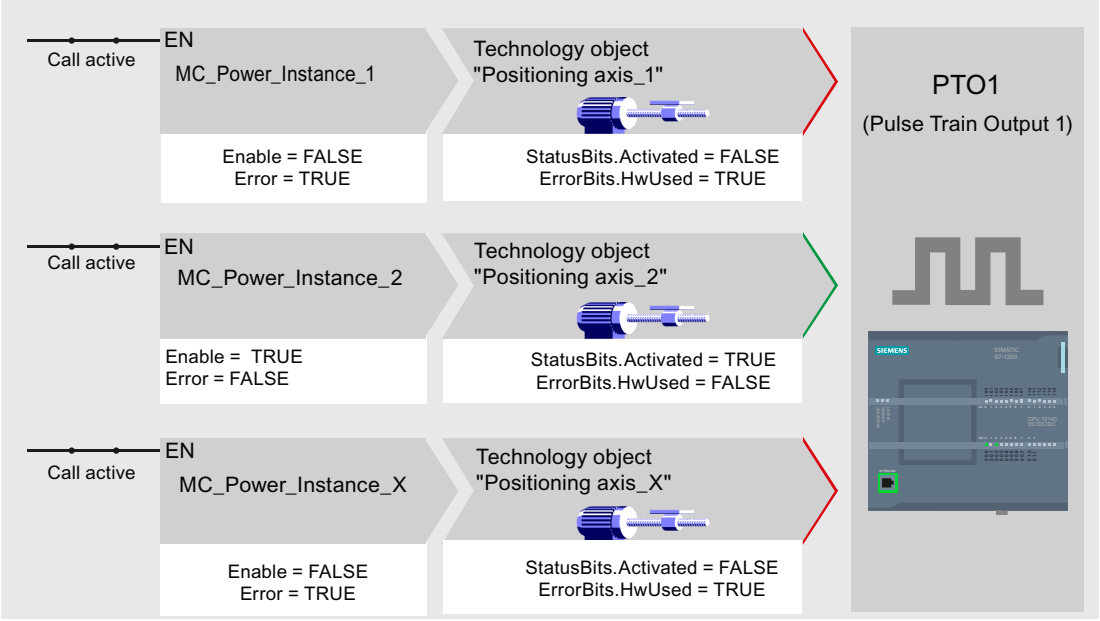
The tags of the activated axis ("Positioning axis_2" here) show the following typical indicators in the user program:

- <Axis name>.StatusBits.Activated = TRUE
- <Axis name>.ErrorBits.HWUsed = FALSE

To switch from one positioning axis technology object to another, follow the steps described below. In the example, a switch is made from "Positioning axis_2" to "Positioning axis_1":

1. End any active traversing motions of activated "Positioning axis_2"
2. Disable "Positioning axis_2" with the associated Motion Control instruction "MC_Power" using input parameter Enable = FALSE
3. To verify that "Positioning axis_2" has been disabled, use an AND operation of output parameter Status = FALSE of Motion Control instruction "MC_Power" and technology object tag <Axis name>.StatusBits.Enable = FALSE.
4. Deactivate the conditional call of the Motion Control instructions for "Positioning axis_2".
5. Activate the conditional call of the Motion Control instruction for "Positioning axis_1". At the first call of the corresponding Motion Control instruction "MC_Power", "Positioning axis_2" is deactivated and "Positioning axis_1" is activated.
6. Enable "Positioning axis_1" with the associated Motion Control instruction "MC_Power" using input parameter Enable = TRUE.
7. To verify that "Positioning axis_1" has been enabled, use an AND operation of output parameter Status = TRUE of Motion Control instruction "MC_Power" and technology object tag <Axis name>.StatusBits.Enable = TRUE.

It is also always possible to cyclically call all Motion Control instructions of all axes working with a single PTO.



When an axis is enabled (here "Positioning axis_2"), this axis becomes active.

In contrast to the conditional call, the Motion Control instructions of the deactivated axes (here "Positioning axis_1" and "Positioning axis_x") indicate errors. The tags of these axes indicate the status <Axis name>.StatusBits.Activated = FALSE and <Axis name>.ErrorBits.HWUsed = TRUE.

Use the conditional call of the Motion Control instructions if you want to implement the user program without error indicators.

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Using multiple drives with the same PTO (Page 6293)

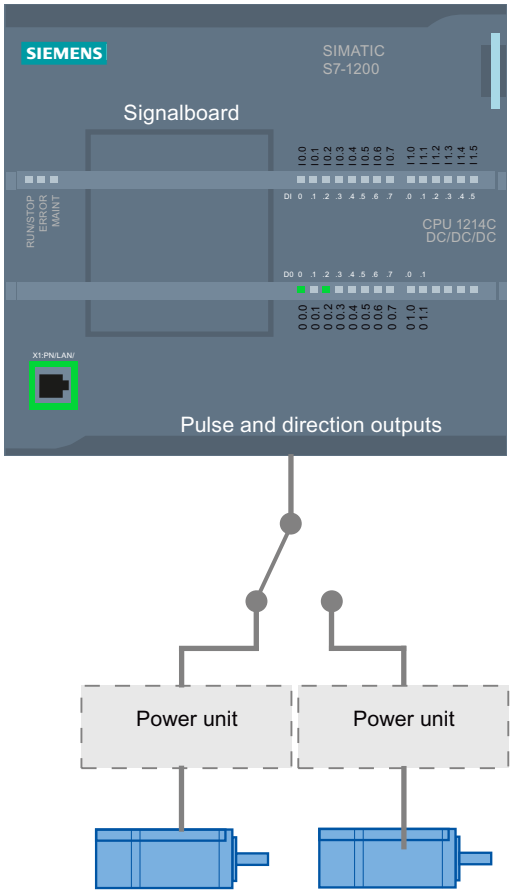
Tracking jobs from higher priority classes (execution levels) (Page 6294)

Special cases when using software limit switches for drive connection via PTO (Page 6296)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

13.2.11.2 Using multiple drives with the same PTO (S7-1200)

If multiple drives are to be used, they can be run with a common PTO (Pulse Train Output) using changeover. The following diagram represents the basic circuit design:



The changeover between drives can be controlled, if required, by the user program via a digital output. If different axis configurations are required for the different drives, a changeover between these configurations is required for the PTO. For additional information on this topic, refer to "Using multiple axes with the same PTO (Page 6290)".

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Using multiple axes with the same PTO (Page 6290)

Tracking jobs from higher priority classes (execution levels) (Page 6294)

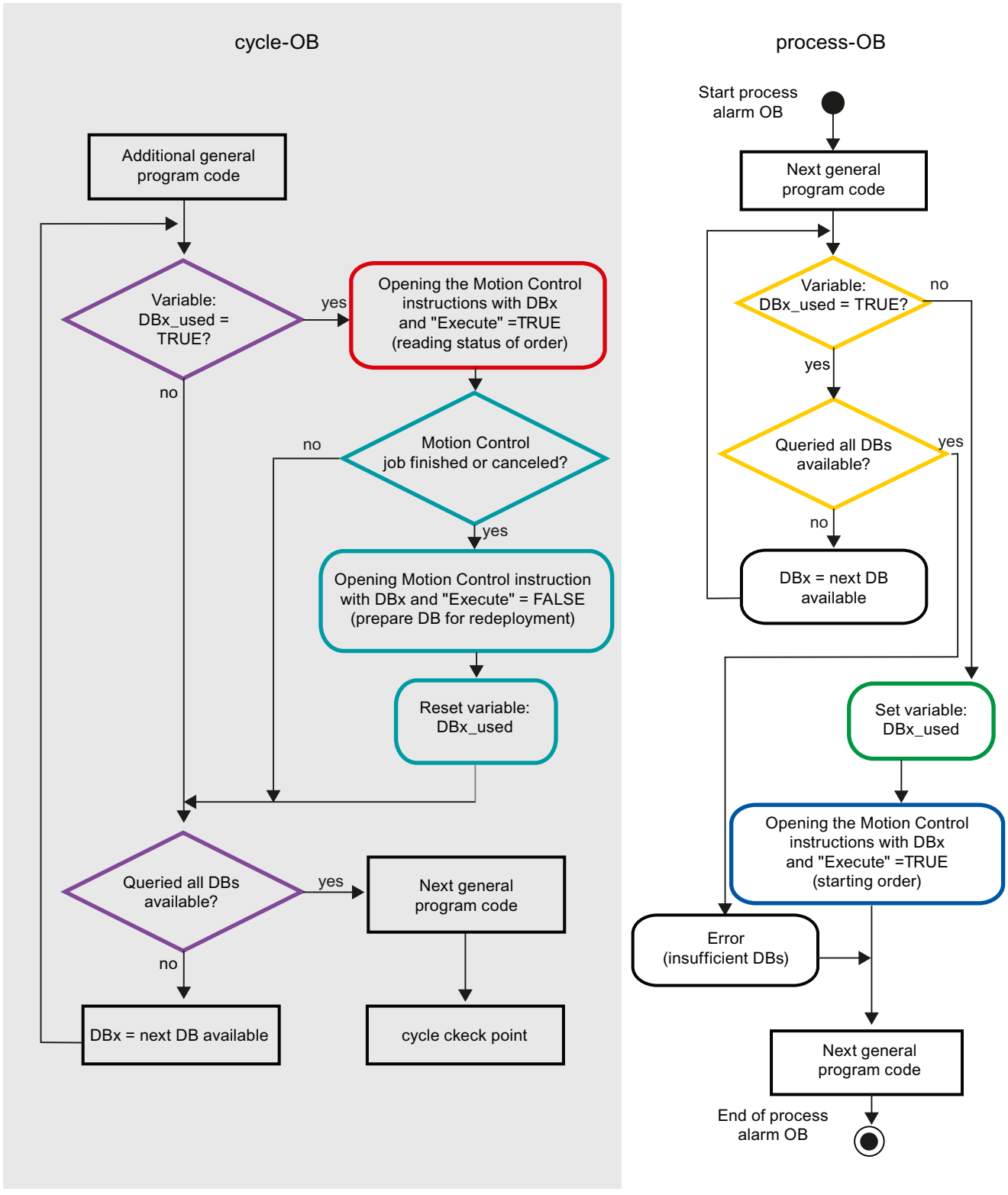
Special cases when using software limit switches for drive connection via PTO (Page 6296)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

13.2.11.3 Tracking jobs from higher priority classes (execution levels) (S7-1200)

Depending on the application, it may be necessary to start Motion Control commands (for example, interrupt-controlled) in a higher priority class (execution level).

The Motion Control instructions must be called at short intervals for status monitoring. Motion Control commands cannot be sufficiently closely monitored if the higher priority Motion Control commands are called only once or at too great an interval. Tracking in such cases can be carried out in the cycle OB. An instance data block that is not currently being utilized must be available for each start of a higher priority Motion Control command. Refer to the following flow chart to see how you start Motion Control commands in a higher priority class (for example, hardware interrupt OB) and continue tracking in the program cycle OB:



Depending on the frequency of the Motion Control commands you want to start, you will have to generate a sufficient number of instance data blocks. Users determine which instance data block is currently used in the DBx_used tags.

Start of Motion Control command in the hardware interrupt OB

Binary queries of the DBx_used tags (orange) are used to find an instance data block not currently in use. If such an instance data block is found, the utilized instance data block is marked as "used" (green) and the Motion Control command is started with this instance data block (blue).

Any other program sections of the hardware interrupt OB are then executed, followed by a return to the program cycle OB.

Tracking of started Motion Control commands in the program cycle OB

All instance data blocks available in the cycle OB are checked to determine if they are currently in use by means of the DBx_used tag (violet).

If an instance data block is in use (Motion Control command is being processed), the Motion Control instruction with this instance data block and input parameter Execute = TRUE is called to read out the status messages (red).

If the command is complete or has been aborted, the following actions are taken next (blue green):

- Call of Motion Control instruction with input parameter Execute = FALSE
- Resetting the DBx_used tag

This completes the command tracking, and the instance data block is now available for use again.

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Using multiple axes with the same PTO (Page 6290)

Using multiple drives with the same PTO (Page 6293)

Special cases when using software limit switches for drive connection via PTO (Page 6296)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

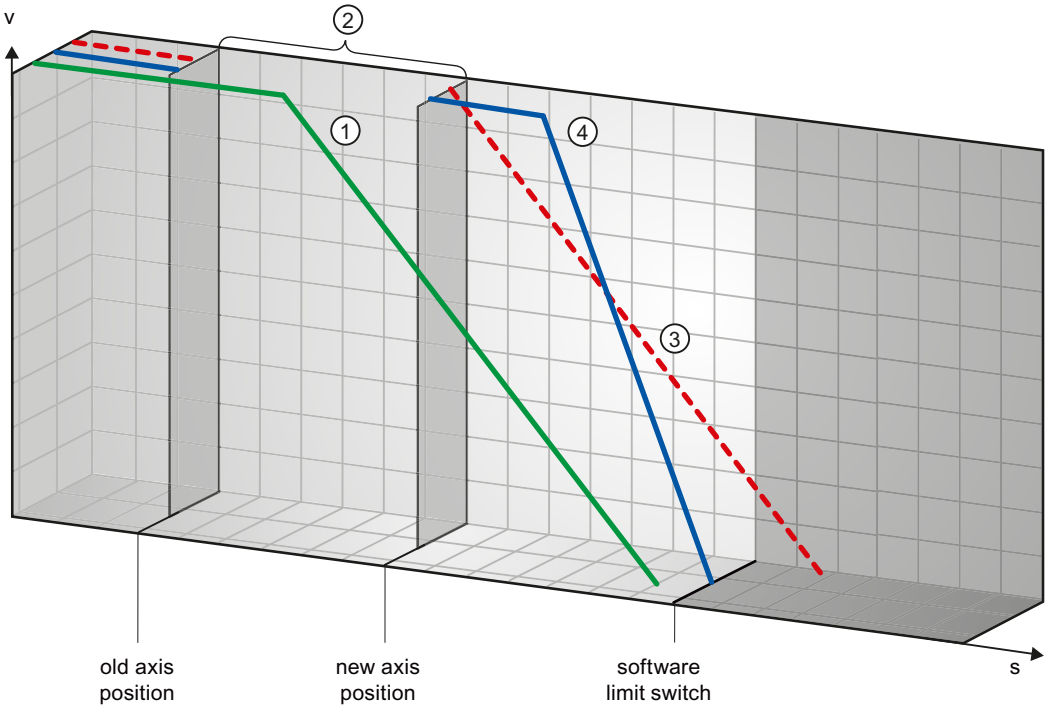
13.2.11.4 Special cases when using software limit switches for drive connection via PTO (S7-1200)

Software limit switches in conjunction with a homing operation (S7-1200)

Due to unfavorably parameterized homing jobs, the braking action of the axis may be influenced at the software limit switch. Take the following examples into consideration when developing your program.

Example 1:

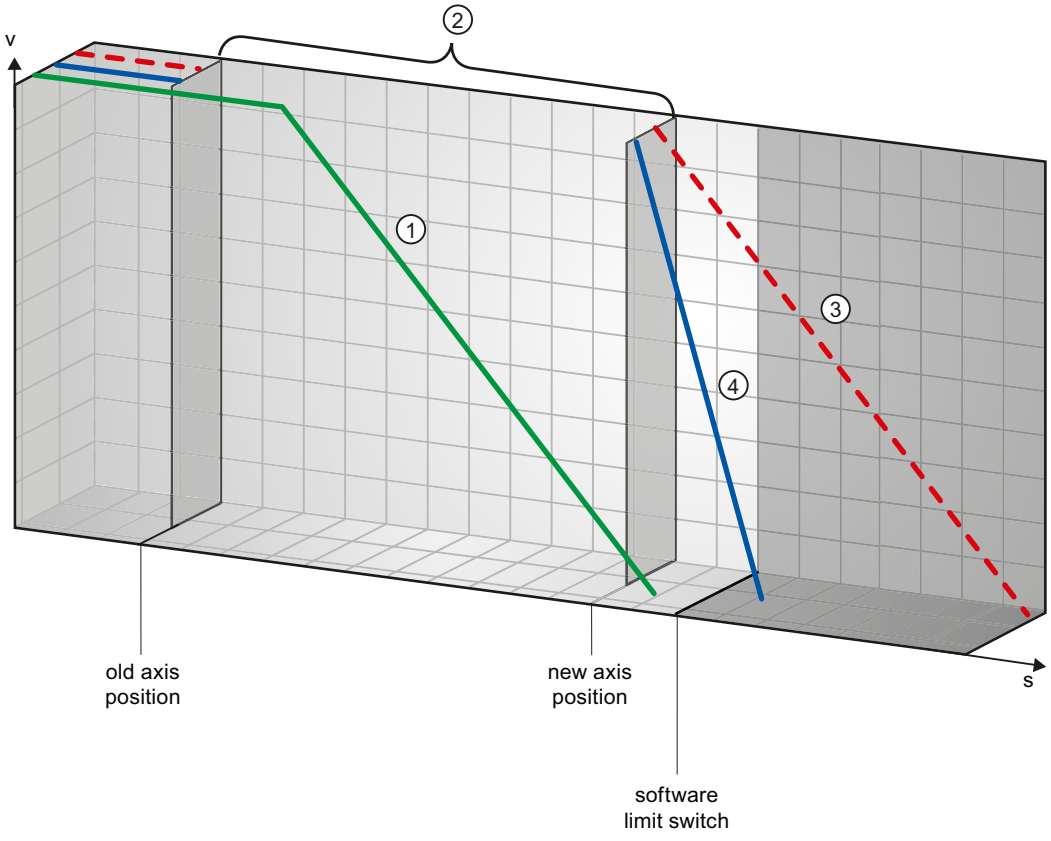
During a travel command, a homing job (for example, Set reference point) offsets the current axis position in the direction of the software limit switch. It is still possible to bring the axis to a standstill before reaching the software limit switch:



①	The green curve shows the motion **without** the homing job. The axis brakes at the configured deceleration and comes to a standstill at a position before the software limit switch.
②	A new axis position is set as a result of the homing job. The area between the old and new axis position is thus "skipped".
③	Based on the new axis position, the axis would theoretically be stopped with the configured deceleration at a position after the software limit switch (red curve).
④	Because braking with the configured deceleration is no longer sufficient, the axis actually follows the blue curve. Following constant motion, the axis brakes at the configured emergency stop deceleration and comes to a standstill at the position of the software limit switch.

Example 2:

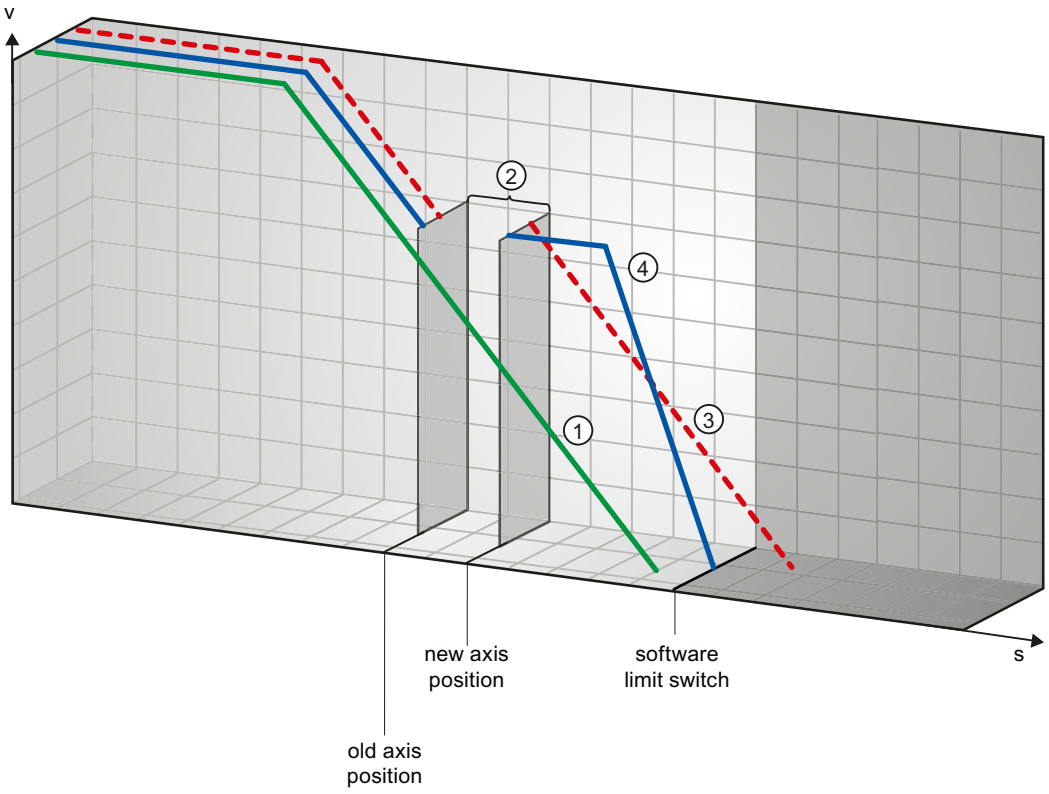
During a travel command, a homing job (for example, Set reference point) offsets the current axis position in the direction of the software limit switch. In contrast to example 1, it is no longer possible to bring the axis to a standstill before reaching the software limit switch. The axis overruns the position of the software limit switch.



①	The green curve shows the motion **without** the homing job. The axis brakes at the configured deceleration and comes to a standstill at a position before the software limit switch.
②	A new axis position is set as a result of the homing job. The area between the old and new axis position is thus "skipped".
③	Based on the new axis position, the axis would theoretically be stopped with the configured deceleration at a position well after the software limit switch (red curve).
④	Because braking with the configured deceleration is no longer sufficient, the axis actually follows the blue curve. The axis brakes with the configured emergency stop deceleration. However, the emergency stop deceleration is not sufficient to stop the axis at the position of the software limit switch. The position of the software limit switch is overrun.

Example 3:

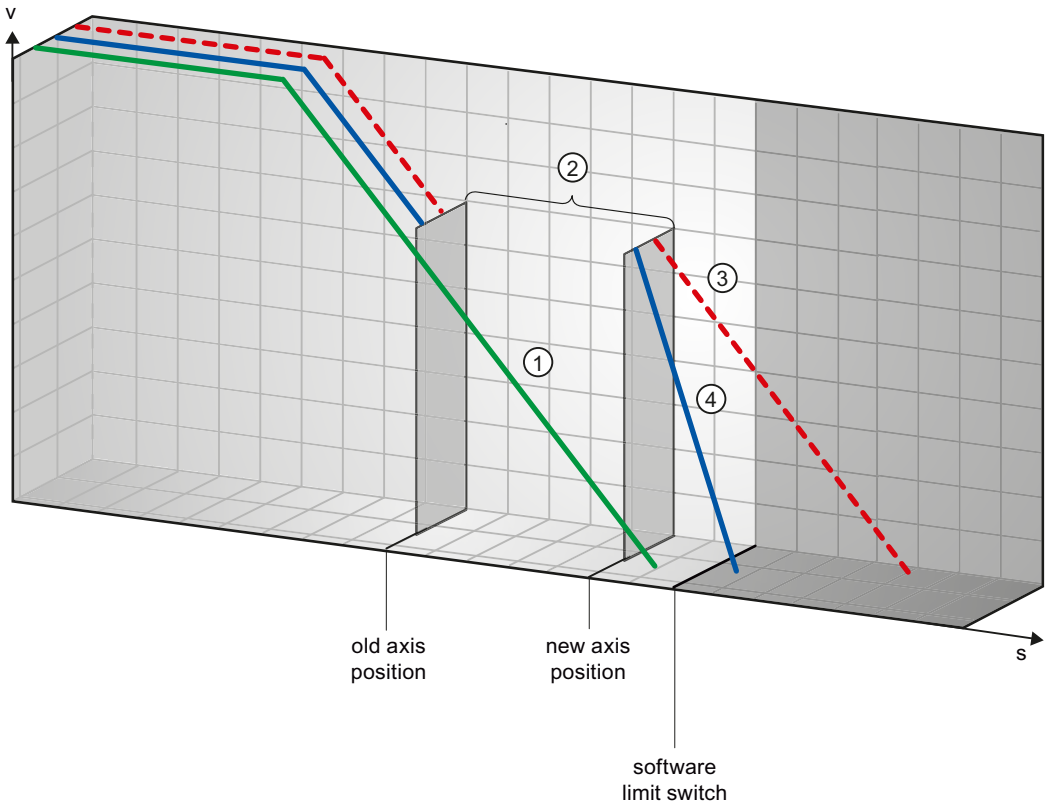
During a braking operation, a homing job (for example, Set reference point) offsets the current axis position in the direction of the software limit switch. It is still possible to bring the axis to a standstill before reaching the software limit switch:



①	The green curve shows the motion **without** the homing job. The axis brakes at the configured deceleration and comes to a standstill at a position before the software limit switch.
②	A new axis position is set as a result of the homing job. The area between the old and new axis position is thus "skipped".
③	Based on the new axis position, the axis would theoretically be stopped with the configured deceleration at a position after the software limit switch (red curve).
④	Because braking with the configured deceleration is no longer sufficient, the axis actually follows the blue curve. Following constant motion, the axis brakes at the configured emergency stop deceleration and comes to a standstill at the position of the software limit switch.

Example 4:

During a braking operation, a homing job (for example, Set reference point) offsets the current axis position in the direction of the software limit switch. In contrast to example 3, it is no longer possible to bring the axis to a standstill before reaching the software limit switch. The axis overruns the position of the software limit switch.



①	The green curve shows the motion **without** the homing job. The axis brakes at the configured deceleration and comes to a standstill at a position before the software limit switch.
②	A new axis position is set as a result of the homing job. The area between the old and new axis position is thus "skipped".
③	Based on the new axis position, the axis would theoretically be stopped with the configured deceleration at a position well after the software limit switch (red curve).
④	Because braking with the configured deceleration is no longer sufficient, the axis actually follows the blue curve. The axis brakes with the configured emergency stop deceleration. However, the emergency stop deceleration is not sufficient to stop the axis at the position of the software limit switch. The position of the software limit switch is overrun.

See also

Software limit switches and software limit switch position changes. (Page 6300)

Software limit switches in conjunction with dynamic changes (Page 6301)

Response of the axis when position limits are triggered (Page 6201)

Software limit switches and software limit switch position changes. (S7-1200)

An incorrect change in the position of the software limit switch during the runtime of the user program can abruptly reduce the distance between the current axis position and the position of the software limit switch.

The axis response is similar to that described in Software limit switches in conjunction with a homing operation (Page 6296).

See also

Software limit switches in conjunction with a homing operation (Page 6296)

Software limit switches in conjunction with dynamic changes (Page 6301)

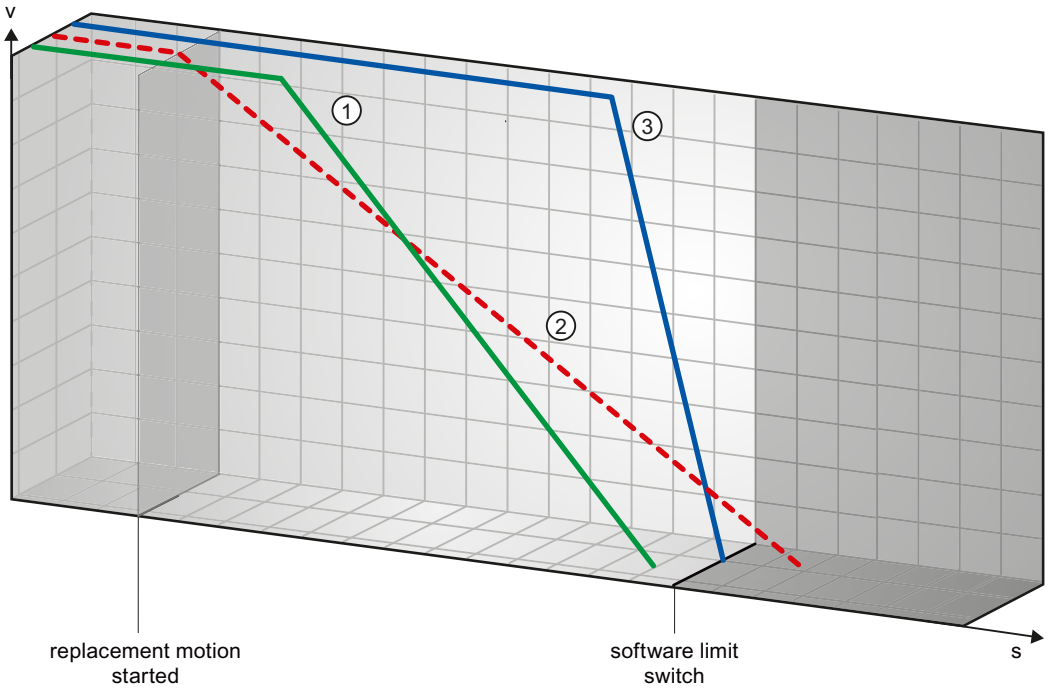
Response of the axis when position limits are triggered (Page 6201)

Software limit switches in conjunction with dynamic changes (S7-1200)

It is possible to influence the deceleration of the axis in the area of the software limit switches in conjunction with overriding motion commands. This applies when the overriding motion command is started with a lower deceleration (tag <Axis name>.DynamicDefaults.Deceleration). Take the following examples into consideration when developing your program.

Example 1:

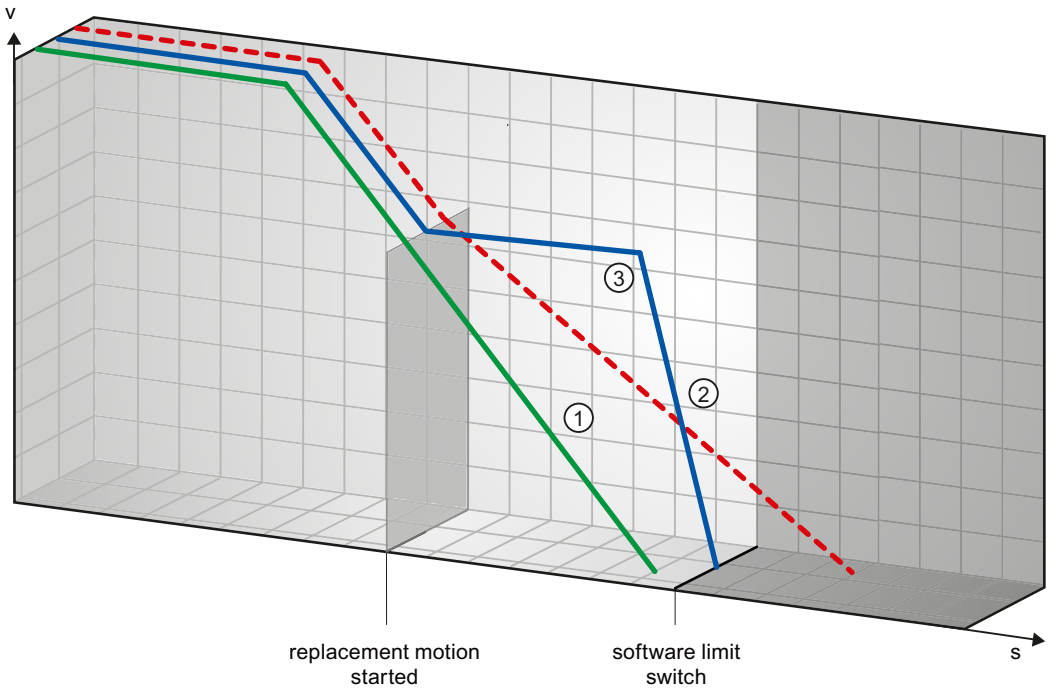
During axis motion, an active motion command is overridden by another motion command with a lower deceleration:



①	The green curve shows the motion of an active command **without** this command being overridden. The axis brakes at the configured deceleration and comes to a standstill at a position before the software limit switch.
②	Based on the overriding motion command with lower deceleration, the axis would theoretically be stopped with the configured deceleration at a position after the software limit switch (red curve).
③	Because braking with the configured deceleration of the overriding motion command is no longer sufficient, the axis actually follows the blue curve. Following a constant motion, the axis brakes at the emergency stop deceleration and comes to a standstill at the position of the software limit switch.

Example 2:

During braking of the axis, an active motion command is overridden by another motion command with a lower deceleration:



①	The green curve shows the motion of an active command **without** this command being overridden. The axis brakes at the configured deceleration and comes to a standstill at a position before the software limit switch.
②	Based on the overriding motion command with lower deceleration, the axis would theoretically be stopped at a position well after the software limit switch (red curve).
③	Because braking with the configured deceleration of the overriding motion command is no longer sufficient, the axis actually follows the blue curve. Following a constant motion, the axis brakes at the emergency stop deceleration and comes to a standstill at the position of the software limit switch.

See also

Software limit switches in conjunction with a homing operation (Page 6296)

Software limit switches and software limit switch position changes. (Page 6300)

Response of the axis when position limits are triggered (Page 6201)

13.2.11.5 Reducing velocity for a short positioning duration (S7-1200)

The CPU can reduce the velocity of a positioning command when the planned positioning duration is < 2 ms.

The velocity of command execution will then be reduced for the entire duration. The reduced velocity (pulses per s) is calculated as follows:

- Reduced velocity = Number of pulses to be output * 500Hz

Velocity is **not** reduced if the planned positioning duration is >= 2 ms.

13.2.11.6 Dynamic adjustment of start/stop velocity (S7-1200)

The configuration of your velocity limits (start/stop velocity and maximum velocity), the dynamic values (acceleration, deceleration and jerk) and the target speed of the traversing command may under certain circumstances result in the start/stop velocity being dynamically adjusted by the CPU.

This is for example the case when, due to a low configured start/stop velocity, the time required for the first pulses would be longer than that possible for the entire acceleration. The first pulse is in these cases output at a greater velocity than the configured start/stop velocity. The subsequent pulses are also dynamically adjusted to ensure the acceleration process can be completed in the specified time.

Ensure in the event of any pulse loss that the hardware (drive) you use is adjusted to this situation, or change the dynamic settings of your axis to avoid dynamic adjustment of the start/stop velocity.

13.2.11.7 Move the axis without position control for servicing (S7-1200)

If service is required, it may be necessary to move a PROFIdrive drive or a drive with an analog drive interface without position control.

This may be the case for example with invalid or incorrect encoder values.

The following Motion Control instructions influence the status of the position control:

Enable position-controlled drives without position control with MC_Power

Start the axis with the Motion Control instruction MC_Power StartMode = 0 in non-position-controlled operation.

The non-position-controlled mode is in effect until another Motion Control instruction changes the status of the position control.

MC_MoveVelocity

MC_MoveVelocity with PositionControlled = FALSE forces non-position-controlled operation.

MC_MoveVelocity with PositionControlled = TRUE forces position-controlled operation.

The selected position-controlled operation is retained even after the termination of MC_MoveVelocity.

MC_MoveJog

MC_MoveJog with PositionControlled = FALSE forces non-position-controlled operation.

MC_MoveJog with PositionControlled = TRUE forces position-controlled operation.

The selected position-controlled operation is retained even after the termination of MC_MoveJog.

MC_Home, MC_MoveRelative, MC_MoveAbsolute

The Motion Control instructions MC_Home, MC_MoveRelative and MC_MoveAbsolute force position-controlled operation.

The position control remains active even when the command has ended.

MC_Halt

The Motion Control instruction MC_Halt is executed in position-controlled and non-position-controlled operation.

The status of the position control remains unchanged.

See also

MC_Power: Enable, disable axis as of V6 (Page 4094)

MC_Halt: Stop axis as of V6 (Page 4105)

13.2.11.8 List of ErrorIDs and ErrorInfos (technology objects as of V6) (S7-1200)

The following table lists all ErrorIDs and ErrorInfos that can be indicated in Motion Control instructions. In addition to the cause of the error, remedies for eliminating the error are also listed:

Depending on the error reaction, the axis is stopped in the case of operating errors with stop of axis. The following error reactions are possible:

- **Remove enable**
  The setpoint zero is output and the enable is removed. The axis is braked depending on the configuration in the drive, and is brought to a standstill.
- **Stop with emergency stop ramp**
  Active motion commands are aborted. The axis is braked with the emergency stop deceleration configured under "Technology object > Extended parameters > Dynamics > Emergency stop ramp" without any jerk limit and brought to a standstill.

Operating error with axis stop

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#8000		Drive error, loss of "Drive ready"		-
	16#0001	-	Acknowledge error with instruction "MC_Reset"; provide drive signal; restart command, if necessary	

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#8001		**Low SW limit switch has been tripped**		-
	16#000E	The position of the low SW limit switch was reached with the currently configured deceleration	Acknowledge the error with instruction "MC_Reset"; use a motion command to move the axis in the positive direction out of the range of the SW limit switch	
	16#000F	The position of the low SW limit switch was reached with the emergency stop deceleration		
	16#0010	The position of the low SW limit switch was exceeded with the emergency stop deceleration		Remove enable
16#8002		**High SW limit switch has been tripped**		-
	16#000E	The position of the high SW limit switch was reached with the currently configured deceleration	Acknowledge the error with instruction "MC_Reset"; use a motion command to move the axis in the negative direction out of the range of the SW limit switch	
	16#000F	The position of the high SW limit switch was reached with the emergency stop deceleration		
	16#0010	The position of the high SW limit switch was exceeded with the emergency stop deceleration		Remove enable
16#8003		**Low HW limit switch was reached**		For drive connection via PTO (Pulse Train Output): Stop with emergency stop ramp For drive connection via PROFIdrive/analog output: Remove enable
	16#000E	The low HW limit switch was reached. The axis was stopped with the emergency stop deceleration. (During an active homing procedure, the homing switch was not found)	Acknowledge the error for an enabled axis with instruction "MC_Reset"; use a motion command to move the axis in the positive direction out of the range of the HW limit switch.	
16#8004		**High HW limit switch was reached**		For drive connection via PTO (Pulse Train Output): Stop with emergency stop ramp For drive connection via PROFIdrive/analog output: Remove enable
	16#000E	The high HW limit switch was reached. The axis was stopped with the emergency stop deceleration. (During an active homing procedure, the homing switch was not found)	Acknowledge the error for an enabled axis with instruction "MC_Reset"; use a motion command to move the axis in the negative direction out of the range of the HW limit switch.	

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#8005		**PTO/HSC are already being used by another axis**		-
	16#0001	-	**The axis was configured incorrectly:** Correct the configuration of the PTO (Pulse Train Output) / HSC (High Speed Counter) and download it to the controller	
			More than one axis is to run with one PTO: Another axis is using the PTO / HSC. If the current axis is to assume the control, the other axis must be disabled with "MC_Power" Enable = FALSE. (See also Using multiple axes with the same PTO (Page 6290))	
16#8006		**A communication error in the control panel has occurred**		Remove enable
	16#0012	A timeout has occurred	Check the cable connection and press the "Manual control" button again	
16#8007		**The axis cannot be enabled**		-
	16#0025	Restarting	Wait until the axis restart is complete.	
	16#0026	Executing loading process in RUN mode	Wait until the loading process is complete.	
16#8008		**Invalid direction of motion**		-
	16#002E	The selected motion direction is not allowed.	• Adjust the motion direction and restart the command. • Adjust the allowed direction of rotation in the technology object configuration under "Extended parameters > Mechanics". Restart the command.	
	16#002F	A reversing motion is not possible with the selected motion direction.		
16#8009		**Reference switch/encoder zero mark not found**		Stop with emergency stop ramp
	16#0033	Error in the configuration, hardware or installation of the encoder or at the homing switch.	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.	

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#800A		**Alarm message from encoder**		Remove enable
	16#0001	-	Check the device with regard to function, connections and I/Os.	
	16#0034	Hardware error at encoder		
	16#0035	Encoder dirty		
	16#0036	Error during reading of encoder absolute value	Compare the encoder type in the drive or encoder parameter P979 with the configuration data of the technology object.	
	16#0037	Zero mark monitoring of the encoder	Encoder reports error in zero mark monitoring (fault code 0x0002 in Gx_XIST2, see PROFIdrive profile). Check the plant for electromagnetic compatibility (EMC).	
	16#0038	Encoder is in "Parking" state	• Search for the cause of the error in the connected drive or encoder. • Check whether the error message was possibly triggered by a commissioning action at the drive or encoder.	
	16#0040	PROFIdrive: Encoder on bus failed (station failure).	Check the device with regard to function, connections and I/Os.	
	16#0041	PROFIdrive: Signs of life of encoder faulty.		
16#800B		**Range violation of the position**		Remove enable
	16#0039	Range violation in positive direction	Home the axis to a valid actual value range.	
	16#003A	Range violation in negative direction		
	16#003B	The change of the actual position in a position control clock cycle is greater than the modulo length.	Adjust the modulo length of the employed encoder.	

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#800C		**Alarm message from drive**		Remove enable
	16#0001	-	Check the device with regard to function, connections and I/Os. In the "Tuning" dialog box, select a smaller gain (Kv).	
	16#003C	PROFIdrive: Drive signal "Control requested" failed.		
	16#003D	PROFIdrive/analog drive connection: Drive has shut down.		
	16#003E	PROFIdrive: Drive on bus failed (station failure)		
	16#003F	PROFIdrive: Signs of life of drive faulty.	• Check the device with regard to function, connections and I/Os. • Compare the clock parameters of HW Config (PROFIBUS line, slave OM for drive or encoder) and the execution system. Tmapc and servo must be configured with the same clock cycle time.	
16#800D		**The permitted following error was exceeded**		Remove enable
	16#0001	-	• Check the configuration of the control loop. • Check the direction signal of the encoder. • Check the configuration of following error monitoring.	
16#800E		**Error at the hardware limit switch**		Remove enable
	16#0042	Illegal free travel direction with active hardware limit switch	The programmed direction of movement is disabled due to the active hardware limit switch. Retract the axis in the opposite direction.	
	16#0043	Hardware limit switch polarity is reversed, axis cannot be freed	Check the mechanical configuration of the hardware limit switch.	
	16#0044	Both hardware limit switches are active, axis cannot be freed		
16#800F		**Error in target range**		Remove enable
	16#0045	Target range not reached	Target range was not reached within the positioning tolerance time. • Check the configuration of the position monitoring. • Check the configuration of the control loop.	
	16#0046	Exit target range again	The target range was exited within the minimum dwell time. • Check the configuration of the position monitoring. • Check the configuration of the control loop.	

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#8010		**Position of the low SW limit switch is greater than that of the high SW limit switch when the axis is not a modulo axis**		Remove enable
	16#0001	-	Change the position of the software limit switches.	
16#8011		**Approach velocity to the homing switch / zero mark equals zero.**		Remove enable
	16#000A	Value is less than or equal to 0.	Approach velocity > Select zero	
16#8012		**Homing velocity for setting the home position equals zero.**		Remove enable
	16#000A	Value is less than or equal to 0.	Homing velocity > Select zero	
16#8013		**The axis cannot occupy the PTO because it is being used by "CTRL_PTO".**		Remove enable
	16#0001	-	Select another PTO in the configuration.	

Operating error without axis stop

ErrorID	ErrorInfo	Description	Remedy
16#8200		**Axis is not enabled**	
	16#0001	-	Enable the axis; restart the command
16#8201		**Axis has already been enabled by another "MC_Power" instance**	
	16#0001	-	Enable the axis with only one "MC_Power" instance
16#8202		**The maximum number of simultaneous motion control commands has been exceeded (max. 200 commands for drive connection via PTO (Pulse Train Output), max. 100 commands for drive connection via PROFIdrive/analog output)**	
	16#0001	-	Reduce the number of simultaneously active commands; restart the command A command is active if parameter "Busy" = TRUE in the Motion Control instruction.
16#8203		**Axis is currently operated in "Manual control" (axis control panel)**	
	16#0001	-	Exit "Manual control"; restart the command
16#8204		**Axis is not homed**	
	16#0001	-	Home the axis with instruction "MC_Home"; restart the command
16#8205		**The axis is currently controlled by the user program (the error is only displayed in the axis control panel)**	
	16#0013	The axis is enabled in the user program	Disable axis with instruction "MC_Power" and select "Manual control" again in the axis control panel
16#8206		**Technology object not activated yet**	
	16#0001	-	Enable the axis with instruction "MC_Power" Enable = TRUE or enable the axis in the axis control panel.
16#8207		**Command rejected**	

ErrorID	ErrorInfo	Description	Remedy
	16#0001	-	
	16#0016	Active homing is running; another homing method cannot be started.	Wait for active homing to finish or abort the active homing with a motion command, for example, "MC_Halt".
	16#0018	The axis cannot be moved with a command table while it is being actively or passively homed.	Wait until direct or passive homing is complete.
	16#0019	The axis cannot be actively or passively homed while a command table is being processed.	Wait for command table to finish or abort the command table with a motion command, for example, "MC_Halt".
	16#0052	The specified position exceeds the numerical limit.	Enter a valid position value at the Motion Control instruction.
	16#0053	The axis is ramping up.	Wait until the axis is ready for operation.
	16#0054	Actual value is invalid	To execute a "MC_Home" command, the actual values must be valid. Check the status of the actual values. The variable of the technology object <axis name>."StatusSensor.State" must show the value 2 (valid).
	16#006B	Call only prohibited in position-controlled mode	Enable the axis with position control with the "MC_Power" with StartMode = 1.
16#8208		**Difference between maximum and start/stop velocity is invalid**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#8209		**Invalid acceleration for technology object "Axis"**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#820A		**It is not possible to restart the axis**	
	16#0013	The axis is enabled in the user program	Disable the axis with the "MC_Power" instruction; restart again
	16#0027	The axis is currently being operated in "Manual control" (axis control panel)	Exit "Manual control"; restart again
	16#002C	The axis is not disabled.	Disable the axis; restart the command
	16#0047	The technology object is not ready for restart.	Download the project again.
	16#0048	Condition for restart of the technology object is not satisfied.	Disable the technology object.
16#820B		**It is not possible to execute the command table**	
	16#0026	Executing loading process in RUN mode	Wait until the loading process is complete.
16#820C		**No configuration available**	
	16#0001	-	Internal error Contact the hotline.
	16#0014	The selected hardware is used by another application	

Block parameter error

ErrorID	ErrorInfo	Description	Remedy
16#8400		**Invalid value at parameter "Position" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
16#8401		**Invalid value at parameter "Distance" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
16#8402		**Invalid value at parameter "Velocity" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0008	Value is greater than the configured maximum velocity	
	16#0009	Value is less than the configured start/stop velocity	
	16#0024	Value is less than 0	
	16#0030	Value has an incorrect number format or is outside the valid number range	
16#8403		**Invalid value at parameter "Direction" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; restart the command
16#8404		**Invalid value at parameter "Mode" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; restart the command
	16#0015	Active/passive homing is not configured	Correct the configuration and download it to the controller; enable the axis and restart the command
	16#0017	The direction reversal is activated at the hardware limit switch, despite the fact that the hardware limit switches are disabled	• Activate the HW limit switch using the variable <axis name>.PositionLimits_HW.Active = TRUE, restart the command • Correct the configuration and download it to the controller; enable the axis and restart the command
	16#0055	Invalid mode at incremental encoder	Start a homing process for an incremental encoder using parameter "Mode" = 0, 1, 2, 3.
	16#0056	Invalid mode at absolute encoder	Passive and active homing ("Mode" = 2, 3) are not possible for an absolute value encoder. Start a homing process for an absolute encoder using parameter "Mode" = 0, 1.
16#8405		**Invalid value at parameter "StopMode" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; enable the axis again
16#8406		**Simultaneous forward and backward jogging is not allowed**	
	16#0001	-	Take steps to ensure that parameters "JogForward" and "JogBackward" do not have signal status TRUE simultaneously; restart the command.

ErrorID	ErrorInfo	Description	Remedy
16#8407		**Switching to another axis with instruction "MC_Power" is only permitted after disabling the active axis.**	
	16#0001	-	Disable the active axis; it is then possible to switch to the other axis and enable it.
16#8408		**Invalid value at parameter "Axis" of the Motion Control instruction**	
	16#001A	The specified value does not match the required technology object version	Correct the value; restart the command
	16#001B	The specified value does not match the required technology object type	
	16#001C	The specified value is not a Motion Control technology data block	
16#8409		**Invalid value at parameter "CommandTable" of the Motion Control instruction**	
	16#001A	The specified value does not match the required technology object version	Correct the value; restart the command
	16#001B	The specified value does not match the required technology object type	
	16#001C	The specified value is not a Motion Control technology data block	
16#840A		**Invalid value at parameter "StartStep" of the Motion Control instruction**	
	16#000A	Value is less than or equal to 0.	Correct the value; restart the command
	16#001D	The start step is greater than the end step	
	16#001E	Value is greater than 32	
16#840B		**Invalid value at parameter "EndStep" of the Motion Control instruction**	
	16#000A	Value is less than or equal to 0.	Correct the value; restart the command
	16#001E	Value is greater than 32	
16#840C		**Invalid value at parameter "RampUpTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840D		**Invalid value at parameter "RampDownTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840E		**Invalid value at parameter "EmergencyRampTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840F		**Invalid value at parameter "JerkTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#8410		**Invalid value at parameter "Parameter" of the Motion Control instruction**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000B	Address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0028	Data type of VARIANT pointer "Parameter" and "Value" do not match.	Use a suitable data type; restart command
	16#0029	VARIANT pointer "Parameter" does not point to a data block of the technology object.	Correct the VARIANT pointer; restart the command
	16#002A	The value at the VARIANT pointer "Parameter" cannot be read.	Correct the VARIANT pointer; restart the command
	16#002B	The value at the VARIANT pointer "Parameter" cannot be written.	Correct the VARIANT pointer or value; restart the command
	16#002C	The axis is not disabled.	Disable the axis; restart the command
16#8411		**Invalid value at parameter "Value" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
16#8412		**Value at "Start Mode" parameter of the Motion Control instruction invalid**	
	16#0011	The selection value is invalid	Correct the selection value; enable the axis again

Configuration error of the axis

ErrorID	ErrorInfo	Description	Remedy
16#8600		**Parameter assignment of pulse generator (PTO is invalid**	
	16#000B	The address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0014	The selected hardware is used by another application	
16#8601		**Parameter assignment of the high-speed counter (HSC) is invalid**	
	16#000B	The address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0014	The selected hardware is used by another application	
16#8602		**Invalid parameter assignment of "Enable output"**	
	16#000B	The address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8603		**Invalid parameter assignment of "Ready input"**	
	16#000B	The address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8604		**Invalid "Pulses per motor revolution" value**	
	16#000A	Value is less than or equal to zero	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8605		**Invalid "Distance per revolution" value**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#000A	Value is less than or equal to zero	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8606		**Invalid "Start/stop velocity" value**	
	16#0002	The value has an invalid number format	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0007	The start/stop velocity is greater than the maximum velocity	
16#8607		**Invalid "maximum velocity" value**	
	16#0002	The value has an invalid number format	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8608		**Invalid "Acceleration" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8609		**Invalid "Deceleration" value**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#860A		**Invalid "Emergency stop deceleration" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#860B		**Value for position of the low SW limit switch is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
	16#0030	The position value of the low software limit switch is greater than that of the high software limit switch	
16#860C		**Value for position of the high SW limit switch is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary

ErrorID	ErrorInfo	Description	Remedy
16#860D		**Invalid address of the low HW limit switch**	
	16#000B	Invalid address	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000C	The address of the falling edge is invalid	
	16#000D	The address of the rising edge is invalid	
16#860E		**Invalid address of the high HW limit switch**	
	16#000B	Invalid address	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000C	The address of the falling edge is invalid	
	16#000D	The address of the rising edge is invalid	
16#860F		**Invalid "home position offset" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8610		**Invalid "approach velocity" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0008	The velocity is greater than the maximum velocity	
	16#0009	The velocity is less than the start/stop velocity	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8611		**Invalid "Homing velocity" value**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0008	The velocity is greater than the maximum velocity	
	16#0009	The velocity is less than the start/stop velocity	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8612		**Invalid address of the homing switch**	
	16#000B	Invalid address	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000C	The address of the falling edge is invalid	
	16#000D	The address of the rising edge is invalid	
16#8613		**During active homing, direction reversal at the hardware limit switch is activated although the hardware limit switches are not configured**	
	16#0001	-	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8614		**Invalid "Jerk" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#001F	Value is greater than the maximum jerk	
	16#0020	Value is less than the minimum jerk	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8615		**Value for "Unit of measurement" is invalid**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8616		**Address of homing switch is invalid (passive homing as of V4)**	

ErrorID	ErrorInfo	Description	Remedy
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8617		**Value of variable <axis name>.Sensor.Sensor[1].ActiveHoming.Mode is invalid**	
	16#0011	The selection value is invalid (Valid value: 2 = Homing via digital input)	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8618		**Value of variable <axis name>.Sensor.Sensor[1].PassiveHoming.Mode is invalid**	
	16#0011	The selection value is invalid (Valid value: 2 = Homing via digital input)	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8619		**Value of variable <axis name>.Actor.Type is invalid**	
	16#0011	The selection value is invalid (Valid value: 2 = Connection via pulse interface)	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#861A		**Value for "Permitted direction of rotation" is invalid**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#002D	"Both directions" not allowed when direction output is deactivated	
16#861B		**Faulty load gear factors**	
	16#0031	Valid is invalid.	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#861C		**Illegal combination of data for homing with incremental encoder**	

ErrorID	ErrorInfo	Description	Remedy
	16#0031	Valid is invalid.	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#861D		**The set encoder mounting type is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].MountingMode**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#861E		**The configuration of the measuring wheel circumference of the encoder is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.DistancePerRevolution**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#861F		**The configuration for the resolution of the linear encoder is faulty. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.Resolution**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8620		**The set fine resolution for Gn_XIST1 is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.FineResolutionXist1**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8621		**The set fine resolution for Gn_XIST1 in <Axis name>.Sensor.Sensor[1].Parameter.FineResolutionXist1 is not consistent with the setting in PROFIdrive parameter P979**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary

ErrorID	ErrorInfo	Description	Remedy
16#8622		**Invalid value for the configuration date <Axis name>.Actor.Interface.AddressIn or <Axis name>.Actor.Interface.AddressOut**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8623		**The value set in the variable <axis name>.Sensor.Sensor[1].Type is invalid.**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8624		**The set encoder system is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].System**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8625		**Parameter of position monitoring is faulty. Invalid value in <Axis name>.PositioningMonitoring.MinDwellTime**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8626		**Parameter of position monitoring is faulty. Invalid value in <Axis name>.PositioningMonitoring.Window**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8627		**The configuration of the PROFIdrive interface of the actual value is faulty. Invalid value in <Axis name>.Sensor.Sensor[1].Interface.AddressIn or <Axis name>.Sensor.Sensor[1].Interface.AddressOut**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8628		**Faulty controller factors**	
	16#0030	Value has an incorrect number format or is outside the valid number range	The value for the gain or the precontrol of the control loop is faulty. • Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary (<Axis name>.PositionControl.Kv, <Axis name>.PositionControl.Kpc)
16#8629		**Limit for standstill signal is faulty. Invalid value in <Axis name>.StandStillSignal.VelocityThreshold**	

ErrorID	ErrorInfo	Description	Remedy
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862A		**Parameter of position monitoring is faulty. Invalid value in <Axis name>.PositioningMonitoring.ToleranceTime**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862B		**Inconsistent PROFIBUS parameterization; the sum of Ti and To is greater than one DP cycle**	
	16#0030	Value has an incorrect number format or is outside the valid number range	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#862C		**Parameter of standstill monitoring is faulty. Invalid value in <Axis name>.StandStillSignal.MinDwellTime**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862D		**Parameter of following error monitoring is faulty. Invalid value in <Axis name>.FollowingError.MinValue**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862E		**Invalid value for the configuration date <Axis name>.Modulo.Length**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862F		**Invalid value for the configuration date <Axis name>.Modulo.StartValue**	

ErrorID	ErrorInfo	Description	Remedy
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8630		**Invalid value for the configuration date <Axis name>.Actor.DriveParameter.ReferenceSpeed**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8631		**The set fine resolution for Gn_XIST2 is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.FineResolutionXist2**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8632		**The number of determinable encoder revolutions is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.DeterminableRevolutions**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8634		**Parameter of the following error monitoring is faulty. Invalid value in <Axis name>.FollowingError.MaxValue**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8635		**Parameter of the following error monitoring is faulty. Invalid value in <Axis name>.FollowingError.MinVelocity**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8636		**Controller factor is incorrect. Invalid value of the precontrol factor <Axis name>.PositionControl.Kpc**	

ErrorID	ErrorInfo	Description	Remedy
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8637		**Invalid value for the configuration date <Axis name>.Sensor.Sensor[1].Interface.Type**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8638		**Invalid value for the configuration date <Axis name>.Sensor.Sensor[1].Interface.HSC**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8639		**Error at the drive**	
	16#0049	Configuration error at device	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#004A	The technology needs a smaller servo clock.	Internal system error. Check the project for consistency and reload it into the controller.
	16#004B	Device driver not initialized during ramp-up.	To enable a technology object, the actuator driver must be initialized. Execute the command again later.
16#863A		**Communication to the drive is faulty**	
	16#004C	Configuration error at device	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.
	16#004D	The device driver needs a smaller servo clock.	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.
	16#004E	Error in internal communication with the device	Check the project for consistency and reload it into the controller.
16#863B		**Error at encoder**	
	16#0049	Configuration error at device	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#004A	The technology needs a smaller servo clock.	Internal system error. Check the project for consistency and reload it into the controller.
	16#004B	Device driver not initialized during ramp-up.	To enable a technology object, the actuator driver must be initialized. Execute the command again later.
16#863C		**Communication with encoder is faulty**	

ErrorID	ErrorInfo	Description	Remedy
	16#004C	Configuration error at device	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.
	16#004D	The device driver needs a smaller servo clock.	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.
	16#004E	Error in internal communication with the device	Check the project for consistency and reload it into the controller.
16#863D		**Communication to the device (drive or encoder) is faulty**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0055	The requested logical address is invalid.	• Connect a suitable device. • Check the device (I/Os). • Check the topological configuration in HW Config. • Compare the configuration of HW Config and the technology object.
	16#0056	The requested logical output address is invalid.	
	16#0057	The requested logical output address is invalid.	
16#863E		**Value of variable "ControlPanel.Input.TimeOut" is invalid (axis control panel)**	
	16#0030	Value has an incorrect number format or is outside the valid number range	Correct the value in the variables of the technology object <axis name>.ControlPanel.Input.TimeOut. The value is specified in milliseconds (ms).
16#863F		**Invalid value for the configuration date <Axis name>.Actor.DriveParameter.MaxSpeed**	
	16#0030	Value has an incorrect number format or is outside the valid number range	Correct the reference value in the drive and in the configuration of the technology object to Actuator.MaxSpeed/2. With analog drive connection, correct the reference value in the drive and in the configuration of the technology object to Actuator.MaxSpeed/1.17.
16#8640		**Error with automatic transfer of drive parameters in the device**	
	16#0030	Value has an incorrect number format or is outside the valid number range	Correct the value.
	16#0059	The device is not assigned to any SINAMICS drive unit or does not support the services required for the adaptation.	Disable automatic transfer of the parameters in the device. Complete the required parameters in the axis configuration and reload the axis configuration in the device.
	16#005A	Automatic transfer of parameters canceled due to insufficient resources.	
	16#005B	The parameters can only be automatically transferred when the device is connected directly to an I/O area.	

ErrorID	ErrorInfo	Description	Remedy
	16#005C	Maximum speed/velocity (p1082): Either the parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	Check the causes. Disable automatic transfer of the parameters in the device if you cannot eliminate the causes. Complete the required parameters in the axis configuration and reload the axis configuration in the device.
	16#005D	Maximum torque/force (p1520): Either the parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
	16#005E	Maximum torque/force (p1521): Either the parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
	16#005F	Fine resolution torque/force limiting (p1544): Either the parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
	16#0060	Nominal speed/velocity (p2000): Either the parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
	16#0061	Nominal torque/force (p2003): Either the parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
16#8641		**Error with automatic transfer of encoder parameters in the device**	
	16#0030	Value has an incorrect number format or is outside the valid number range	Correct the value.
	16#0059	The device is not assigned to any SINAMICS drive unit or does not support the services required for the adaptation.	Disable automatic transfer of the parameters in the device. Complete the required parameters in the axis configuration and reload the axis configuration in the device.
	16#005A	Automatic transfer of parameters canceled due to insufficient resources.	
	16#005B	The parameters can only be automatically transferred when the device is connected directly to an I/O area.	

ErrorID	ErrorInfo	Description	Remedy
	16#0062	Encoder system (r0979[1/11].0): Either a parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	Check the causes. Disable automatic transfer of the parameters in the device if you cannot eliminate the causes. Complete the required parameters in the axis configuration and reload the axis configuration in the device.
	16#0063	Encoder resolution (r0979[2/12]): Either a parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
	16#0064	Encoder fine resolution Gx_XIST1 (r0979[3/13]): Either a parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
	16#0065	Encoder fine resolution Gx_XIST2 (r0979[4/14]): Either a parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
	16#0066	Number of determinable encoder revolutions (r0979[5/15]): Either a parameter does not exist or its value cannot be read or is outside the permitted limits. Reading of parameters has been aborted due to an error signaled by the hardware.	
16#8642		**Configuration is internally adapted**	
	16#0067	1: Illegal value for Actor.MaxSpeed (Actor.MaxSpeed greater than 2*Actor.ReferenceSpeed) Remedy: set e.g. P2000 = P1082 in drive.	Adjust values.
16#8643		**Inconsistency between the configuration of the axis and the drive configuration**	
	16#0068	The configured telegram type is not compatible with the telegram type on the device (P922 or P2079).	Synchronize the telegram type in the configuration of the axis and in the configuration of the drive.
	16#0069	A function module with linear motor is set in the drive.	Set rotation motor in drive.
16#8644		**Inconsistency between the configuration of the axis and the encoder configuration**	
	16#0068	The configured telegram type is not compatible with the telegram type on the device (P922 or P2079).	Synchronize the telegram type in the configuration of the axis and in the configuration of the encoder.
	16#006A	The encoder on the drive is not an absolute encoder (P979).	Synchronize the encoder type in the configuration of the axis and in the encoder configuration.
16#8645		**The maximum velocity cannot be reached with the configured drive and axis parameters.**	
	16#0001	-	Adapt drive or axis configuration and load configuration in the device.
16#8646		**Illegal value in Sensor.Interface.Number**	

ErrorID	ErrorInfo	Description	Remedy
	16#0030	Value has an incorrect number format or is outside the valid number range	Correct the value and load configuration in the device.
16#8647		**Simulation is not supported for PTO axes**	
	16#0001	-	Disable simulation

Configuration error of the command table

ErrorID	ErrorInfo	Description	Remedy
16#8700		**Value for "Command type" in the command table is invalid**	
	16#0001	-	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
16#8701		**Value for "Position / travel path" in the command table is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
16#8702		**Value for "Velocity" in the command table is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0008	Value is greater than the configured maximum velocity	
	16#0009	Value is less than the configured start/stop velocity	
16#8703		**Value for "Duration" in the command table is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0021	Value is greater than 64800 s	
	16#0022	Value is less than 0.001 s	
16#8704		**Value for "Next step" in the command table is invalid**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0023	The command transition is not permitted for this command	

Internal errors

ErrorID	ErrorInfo	Description	Remedy
16#8FFF		**Internal error**	
	16#F0**	-	POWER OFF and POWER ON the CPU If this does not work, contact Customer Support. Have the following information ready: • ErrorID • ErrorInfo • Diagnostic buffer entries

13.2.11.9 Tags of the positioning axis technology object as of V6 (S7-1200)

Position tag as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Position				
Setpoint position of the axis (indicated in the configured unit of measurement) If the axis is not homed, the tag indicates the position value relative to the enable position of the axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Motion status (Page 6288)

Tags of the positioning axis technology object V4...5 (Page 6436)

Tags of the command table technology object V1...3 (Page 6470)

Velocity tag as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Velocity				
Setpoint velocity of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Motion status (Page 6288)

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

ActualPosition tag as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.ActualPosition				
Actual position of the axis (indicated in the configured unit of measurement) If the axis is not homed, the tag indicates the position value relative to the enable position of the axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

ActualVelocity tag as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.ActualVelocity				
Actual velocity of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Actuator tags as of V6 (S7-1200)

Legend

Data type	Data type of variable
Start value	Start value of variable

Access	Access to the variable in the user program:	
	R	The variable can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the variable can be written with the Motion Control instruction "MC_WriteParam".
	-	The variable cannot be used in the user program.
Effective	Specifies when a change to the variable takes effect:	
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
HMI	The variable can be used in an HMI system (according to the access to the variable in the user program).	

<Axis name>.Actor.Type				
Drive connection • Positioning axis technology object as of V5: 0 = The drive is connected via an analog output. All movements of the axis are position-controlled. 1 = The drive is connected via PROFIdrive telegrams. All movements of the axis are position-controlled. 2 = The drive is connected via a pulse interface. • Positioning axis technology object V4: The drive is connected via a pulse interface.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	2	R	-	X

<Axis name>.Actor.InverseDirection				
Invert direction FALSE = The direction is not inverted. TRUE = The direction is inverted.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X
		WP	2	-

<Axis name>.Actor.DirectionMode				
Permitted direction of rotation 0 = Both directions 1 = Positive direction 2 = Negative direction				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Int	0	R	-	X
		WP	2	-

<Axis name>.Actor.DataAdaption				
Automatic transfer of drive parameters in the device 0 = Disabled 1 = Enabled				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	0	R	-	X

<AxisName>.Actor.Interface.AddressIn (Positioning axis technology object as of V5)				
Input address for the PROFIdrive telegram (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<AxisName>.Actor.Interface.AddressOut (Positioning axis technology object as of V5)				
Output address for the PROFIdrive telegram (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Actor.Interface.EnableDriveOutput				
Enable output (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Actor.Interface.DriveReadyInput				
Ready input (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Actor.Interface.PTO				
Pulse output (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	0	-	-	-

<AxisName>.Actor.DriveParameter.ReferenceSpeed (Positioning axis technology object as of V5)				
Reference value (100%) for the setpoint speed of the drive (N-set) The setpoint speed is transmitted in the PROFIdrive telegram as a standardized value from -200% to 200% of the "ReferenceSpeed". For setpoint specification via an analog output, the analog output can be operated in the range from -117% to 117%, provided the drive permits this.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	3000.0	R	-	X

<AxisName>.Actor.DriveParameter.MaxSpeed (Positioning axis technology object as of V5)				
Maximum value for the setpoint speed of the drive (N-set) (PROFIdrive: MaxSpeed ≤ 2 × ReferenceSpeed Analog setpoint: MaxSpeed ≤ 1.17 × ReferenceSpeed)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	3000.0	R	-	X

<Axis name>.Actor.DriveParameter.PulsesPerDriveRevolution				
Pulses per motor revolution				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	1000.0	R	-	X
		WP	2	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1] tags (S7-1200)

Sensor[1] tags as of V6 (S7-1200)

Legend

Data type	Data type of variable	
Start value	Start value of variable The start value can be overwritten by the axis configuration.	
Access	Access to the variable in the user program:	
	R	The variable can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the variable can be written with the Motion Control instruction "MC_WriteParam".
	-	The variable cannot be used in the user program.

Effective	Specifies when a change to the variable takes effect:
HMI	The variable can be used in an HMI system (according to the access to the variable in the user program).

<axis name>.Sensor.Sensor[1].Type				
Encoder type (internal parameter) 0 = incremental 1 = Absolute				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	0	R	-	-

<axis name>.Sensor.Sensor[1].InverseDirection				
Inversion of the actual value FALSE: Actual value is not inverted TRUE: Actual value is inverted				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<axis name>.Sensor.Sensor[1].System				
Encoder system 0 = Linear encoder 1 = Rotary encoder				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	1	R	-	X

<axis name>.Sensor.Sensor[1].MountingMode				
Type of encoder mounting 0 = Drive end 2 = External				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	0	R	-	X

<axis name>.Sensor.Sensor[1].DataAdaption				
Automatic transfer of encoder parameters in the device 0 = Disabled 1 = Enabled				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	0	R	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1].Interface tags as of V6 (S7-1200)

Legend

Data type	Data type of variable
Start value	Start value of variable The start value can be overwritten by the axis configuration.
Access	Access to the variable in the user program:
	- The variable cannot be used in the user program.
Effective	Specifies when a change to the variable takes effect.
HMI	The variable can be used in an HMI system (according to the access to the variable in the user program).

<axis name>.Sensor.Sensor[1].Interface.Type				
Encoder connection (internal parameter) 0 = PROFIdrive encoder on PROFINET 1 = Encoder on technology module (TM) 2 = Encoder on drive 4 = Encoder on high-speed counter				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	4	-	-	-

<axis name>.Sensor.Sensor[1].Interface.AddressIn				
Input address for the PROFIdrive telegram (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<axis name>.Sensor.Sensor[1].Interface.AddressOut				
Output address for the PROFIdrive telegram (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<axis name>.Sensor.Sensor[1].Interface.HSC				
High-speed counter to which the encoder transfers the actual value (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	-	-	-	-

<axis name>.Sensor.Sensor[1].Interface.Number				
Encoder number				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	1	-	-	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1].Parameter tags as of V6 (S7-1200)

Legend

Data type	Data type of variable	
Start value	Start value of variable The start value can be overwritten by the axis configuration.	
Access	Access to the variable in the user program:	
	R	The variable can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the variable can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the variable takes effect:	
HMI	The variable can be used in an HMI system (according to the access to the variable in the user program).	

<axis name>.Sensor.Sensor[1].Parameter.Resolution				
Resolution of a linear encoder (offset between two encoder pulses)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.001	R	-	X

<axis name>.Sensor.Sensor[1].Parameter.StepsPerRevolution				
Increments per rotary encoder revolution				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	2048	R	-	X

<axis name>.Sensor.Sensor[1].Parameter.FineResolutionXist1				
Number of bits for fine resolution Gn_XIST1 (cyclic actual encoder value)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	11	R	-	X

<axis name>.Sensor.Sensor[1].Parameter.FineResolutionXist2				
Number of bits for fine resolution Gn_XIST2 (absolute value of the encoder)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	9	R	-	X

<axis name>.Sensor.Sensor[1].Parameter.DeterminableRevolutions				
Number of differentiable encoder revolutions for a multi-turn absolute value encoder (For a single-turn absolute value encoder = 1; for an incremental encoder = 0)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	1	R	-	X

<axis name>.Sensor.Sensor[1].Parameter.DistancePerRevolution				
Load distance per revolution of an externally mounted encoder				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	100.0	R	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1].ActiveHoming tags as of V6 (S7-1200)

Legend

Data type	Data type of variable	
Start value	Start value of variable The start value can be overwritten by the axis configuration.	
Access	Access to the variable in the user program:	
	RW	The variable can be read and written in the user program. The variable can be written with Motion Control instruction "MC_WriteParam".
	R	The variable can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the variable can be written with the Motion Control instruction "MC_WriteParam".
	-	The variable cannot be used in the user program.

Effective	Specifies when a change to the variable takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<axis name>.StatusBits.Activated variable changes from FALSE → TRUE), disabled, or enabled
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
	8	For drive connection via PTO (Pulse Train Output): When an active homing command is started
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The variable can be used in an HMI system (according to the access to the variable in the user program).	

<axis name>.Sensor.Sensor[1].ActiveHoming.Mode				
Active homing mode • Positioning axis technology object as of V5: 0 = Zero mark via PROFIdrive telegram (not PTO) 1 = Zero mark via PROFIdrive telegram and proximity switch (not PTO) 2 = Homing via digital input • Positioning axis technology object V4: 2 = Homing via digital input				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	2	R	-	X
		WP	2	-

<axis name>.Sensor.Sensor[1].ActiveHoming.SideInput				
End of the homing switch to which the axis is homed during active homing FALSE = Lower end TRUE = Upper end				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 8, 10	X

<axis name>.Sensor.Sensor[1].ActiveHoming.DigitalInputAddress				
Symbolic input address of the homing switch (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<axis name>.Sensor.Sensor[1].ActiveHoming.HomePositionOffset				
Home position offset (active homing) (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RW	1, 8, 10	X

<axis name>.Sensor.Sensor[1].ActiveHoming.SwitchLevel				
Selection of signal level that is present at the CPU input when the homing switch is reached FALSE = Low level (Low-active) TRUE = High level (High-active)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	RW	1, 8, 10	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1].PassiveHoming tags as of V6 (S7-1200)

Legend

Data type	Data type of variable	
Start value	Start value of variable The start value can be overwritten by the axis configuration.	
Access	Access to the variable in the user program:	
	RW	The variable can be read and written in the user program. The variable can be written with Motion Control instruction "MC_WriteParam".
	R	The variable can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the variable can be written with the Motion Control instruction "MC_WriteParam".
	-	The variable cannot be used in the user program.
Effective	Specifies when a change to the variable takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<axis name>.StatusBits.Activated variable changes from FALSE → TRUE), disabled, or enabled
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
	7	For drive connection via PTO (Pulse Train Output): When a passive homing command is started
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The variable can be used in an HMI system (according to the access to the variable in the user program).	

<axis name>.Sensor.Sensor[1].PassiveHoming.Mode				
Passive homing mode • Positioning axis technology object as of V5: 0 = Zero mark via PROFIdrive telegram (not PTO) 1 = Zero mark via PROFIdrive telegram and proximity switch (not PTO) 2 = Homing via digital input • Positioning axis technology object V4: 2 = Homing via digital input				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	2	R	-	X
		WP	2	-

<axis name>.Sensor.Sensor[1].PassiveHoming.SideInput				
End of the homing switch to which the axis is homed during passive homing FALSE = Lower end TRUE = Upper end				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 7, 10	X

<axis name>.Sensor.Sensor[1].PassiveHoming.DigitalInputAddress				
Symbolic input address of the homing switch (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<axis name>.Sensor.Sensor[1].PassiveHoming.SwitchLevel				
Selection of level that is present at the CPU input when the homing switch is reached FALSE = Low level (Low-active) TRUE = High level (High-active)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	RW	1, 7, 10	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Units tag as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Units.LengthUnit				
Configured unit of measurement of the parameter -1 = "Pulse" 1005 = "°" (degrees) 1013 = "mm" 1010 =: "m" 1018 = "ft" 1019 = "in"				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Int	1013	R	-	X
		WP	2	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Mechanics tag as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".

Effective	Specifies when a change to the tag takes effect:	
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Mechanics.LeadScrew				
Distance per revolution (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Modulo tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Modulo.Enable				
Enable modulo FALSE = Modulo conversion deactivated TRUE = Modulo conversion activated When modulo conversion is enabled, a check is made for modulo length > 0.0				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.Modulo.Length				
Modulo length				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	360.0	R	-	X

<Axis name>.Modulo.StartValue				
Modulo start value				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	R	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

DynamicLimits tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.DynamicLimits.MaxVelocity				
Maximum velocity of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	250.0	R	-	X
		WP	2	-

<Axis name>.DynamicLimits.MinVelocity				
Start/stop velocity of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X
		WP	2	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

DynamicDefaults tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program. The tag can be written with Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE → TRUE), disabled, or enabled
	5	For drive connection via PTO (Pulse Train Output): The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable, or active MC_Home command is started (Mode = 3)
	6	For drive connection via PTO (Pulse Train Output): When a MC_MoveJog command is stopped
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.DynamicDefaults.Acceleration				
Acceleration of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	48.0	RW	1, 5, 6, 10	X

<Axis name>.DynamicDefaults.Deceleration				
Deceleration of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	48.0	RW	1, 5, 6, 10	X

<Axis name>.DynamicDefaults.EmergencyDeceleration				
Emergency stop deceleration of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	120.0	RW	1, 5, 6, 10	X

<Axis name>.DynamicDefaults.Jerk				
Jerk during acceleration and deceleration ramp of the axis (indicated in the configured unit of measurement) The jerk is activated if the configured jerk is greater than 0.00004 mm/s².				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	192.0	RW	1, 5, 10	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

PositionLimits_SW variables as of V6 (S7-1200)

Legend

Data type	Data type of variable	
Start value	Start value of variable The start value can be overwritten by the axis configuration.	
Access	Access to the variable in the user program:	
	RW	The variable can be read and written in the user program. The variable can be written with Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the variable takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<axis name>.StatusBits.Activated variable changes from FALSE → TRUE), disabled, or enabled
	5	For drive connection via PTO (Pulse Train Output): The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable, or active MC_Home command is started (Mode = 3)
	6	For drive connection via PTO (Pulse Train Output): When a MC_MoveJog command is stopped
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The variable can be used in an HMI system (according to the access to the variable in the user program).	

<axis name>.PositionLimits_SW.Active				
Activation of the software limit switches FALSE = The software limit switches are deactivated. TRUE = The software limit switches are activated.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 5, 6, 10	X

<axis name>.PositionLimits_SW.MinPosition				
Position of the low software limit switch (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	-10000.0	RW	1, 5, 6, 10	X

<axis name>.PositionLimits_SW.MaxPosition				
Position of the high software limit switch (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10000.0	RW	1, 5, 6, 10	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

PositionLimits_HW variables as of V6 (S7-1200)

Legend

Data type	Data type of variable	
Start value	Start value of variable The start value can be overwritten by the axis configuration.	
Access	Access to the variable in the user program:	
	RW	The variable can be read and written in the user program. The variable can be written with Motion Control instruction "MC_WriteParam".
	R	The variable can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the variable can be written with the Motion Control instruction "MC_WriteParam".
	-	The variable cannot be used in the user program.

Effective	Specifies when a change to the variable takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<axis name>.StatusBits.Activated variable changes from FALSE → TRUE), disabled, or enabled
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
	5	For drive connection via PTO (Pulse Train Output): The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable, or active MC_Home command is started (Mode = 3)
	6	For drive connection via PTO (Pulse Train Output): When a MC_MoveJog command is stopped
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The variable can be used in an HMI system (according to the access to the variable in the user program).	

<axis name>.PositionLimits_HW.Active				
Activation of the hardware limit switches FALSE = The hardware limit switches are deactivated. TRUE = The hardware limit switches are activated.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 5, 6, 10	X

<axis name>.PositionLimits_HW.MinSwitchLevel				
Selection of signal level that is present at the CPU input when the low hardware limit switch is reached FALSE = Low level (Low-active) TRUE = High level (High-active)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X
		WP	2	-

<axis name>.PositionLimits_HW.MinSwitchAddress				
Symbolic input address of the low hardware limit switch (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<axis name>.PositionLimits_HW.MaxSwitchLevel				
Selection of signal level that is present at the CPU input when the high hardware limit switch is reached FALSE = Low level (Low-active) TRUE = High level (High-active)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X
		WP	2	-

<axis name>.PositionLimits_HW.MaxSwitchAddress				
Input address of the high hardware limit switch (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Homing tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program. The tag can be written with Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE → TRUE), disabled, or enabled
	8	For drive connection via PTO (Pulse Train Output): When an active homing command is started
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Homing.AutoReversal				
Activation of auto reverse at the hardware limit switch during active homing FALSE = Auto reversal at the hardware limit switch is deactivated. TRUE = Auto reversal at the hardware limit switch is activated.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 8, 10	X

<Axis name>.Homing.ApproachDirection				
Approach and homing direction of axis during active homing FALSE = Negative approach direction for finding the homing switch and negative homing direction TRUE = Positive approach direction for finding the homing switch and positive homing direction				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	RW	1, 8, 10	X

<Axis name>.Homing.ApproachVelocity				
Approach velocity of the axis during active homing (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	200.0	RW	1, 8, 10	X

<Axis name>.Homing.ReferencingVelocity				
Homing velocity of the axis during active homing (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	40.0	RW	1, 8, 10	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

PositionControl tag as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.PositionControl.Kv				
Proportional gain in the position control ("Kv" > 0.0)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X
		WP	10	-

<Axis name>.PositionControl.Kpc				
Velocity precontrol of the position control as a percentage				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	100.0	R	-	X
		WP	10	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

FollowingError tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.FollowingError.EnableMonitoring				
Enable following error monitoring FALSE = Following error monitoring disabled TRUE = Following error monitoring enabled				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	R	-	X

<Axis name>.FollowingError.MinValue				
Permissible following error at velocities below the value of "MinVelocity".				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X
		WP	10	-

<Axis name>.FollowingError.MaxValue				
Maximum permissible following error which may be reached at the maximum velocity.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	100.0	R	-	X
		WP	10	-

<Axis name>.FollowingError.MinVelocity				
MinValue" is permissible below this velocity and is held constant.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X
		WP	10	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

PositionMonitoring tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.PositioningMonitoring.ToleranceTime				
Tolerance time Maximum permitted duration from reaching velocity setpoint zero until entrance into the positioning window.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	1.0	R	-	X
		WP	10	-

<Axis name>.PositioningMonitoring.MinDwellTime				
Minimum dwell time in positioning window				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.1	R	-	X
		WP	10	-

<Axis name>.PositioningMonitoring.Window				
Positioning window				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	1.0	R	-	X
		WP	10	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

StandstillSignal tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.StandstillSignal.VelocityThreshold				
Velocity threshold If velocity is below this threshold, then the minimum dwell time begins.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	5.0	R	-	X
		WP	10	-

<Axis name>.StandstillSignal..MinDwellTime				
Minimum dwell time				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.01	R	-	X
		WP	10	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Simulation tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	For drive connection via PTO (Pulse Train Output): If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
	WD BL	For drive connection via PROFIdrive/analog output: The tag can be written to the start value in the load memory with the extended instruction "WRIT_DBL".
Effective	Specifies when a change to the tag takes effect.	
	9	With the restart of the technology object
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Simulation.Mode				
Simulation mode 0: No simulation, normal operation 1: Simulation mode In simulation mode, you can simulate axes without a real drive in the CPU.				
Data type	**Start value**	**Access**	**Effective**	**HMI**

<Axis name>.Simulation.Mode				
DInt	0	R	-	X
		WP	2	-
		WDBL	9	-

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

StatusPositioning tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.StatusPositioning.Distance				
Current distance of axis from target position (indicated in the configured unit of measurement) The value of the tag is only valid during execution of a positioning command with "MC_MoveAbsolute", "MC_MoveRelative", or the axis control panel.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

<Axis name>.StatusPositioning.TargetPosition				
Target position of the axis (indicated in the configured unit of measurement) The value of the tag is only valid during execution of a positioning command with "MC_MoveAbsolute", "MC_MoveRelative", or the axis control panel.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

<AxisName>.StatusPositioning.FollowingError (Positioning axis technology object as of V5)				
Current following error of the axis (indicated in the configured unit of measurement) FollowingError = 0.0 for drive connection via PTO (Pulse Train Output).				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

StatusDrive tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect:	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.StatusDrive.InOperation				
Operational status of the drive FALSE = Drive not ready. Setpoints will not be executed. TRUE = Drive ready. Setpoints can be executed.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusDrive.CommunicationOK				
Cyclic BUS communication between controller and drive FALSE = Communication not established TRUE = Communication established				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

StatusSensor tags as of V6 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect:	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.StatusSensor.State				
Status of the encoder value 0 = Invalid 1 = Wait for status valid 2 = Valid				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	0	RCCP, RP	-	X

<Axis name>.StatusSensor.CommunicationOK				
Cyclic BUS communication between controller and encoder FALSE = Communication not established TRUE = Communication established				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusSensor.AbsEncoderOffset				
Home point offset to the value of an absolute value encoder. The value will be retentively stored in the CPU.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

StatusBits tags as of V6 (S7-1200)

Legend

Data type	Data type of variable	
Start value	Start value of variable	
Access	Access to the variable in the user program:	
	RCCP	The variable can be read in the user program and is updated at each cycle control point.
	RP	The variable can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding variables is determined at the start of the command.
Effective	Specifies when a change to the variable takes effect.	
HMI	The variable can be used in an HMI system.	

<Axis name>.StatusBits.Activated				
Activation of the axis FALSE = The axis is not activated. TRUE = The axis is activated. It is connected to the assigned PTO (Pulse Train Output). The data of the technology data block will be updated cyclically.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Enable				
Enable status of the axis FALSE = The axis is not enabled. TRUE = The axis is enabled and ready to accept Motion Control commands.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.HomingDone				
Homing status of the axis FALSE = The axis is not homed. TRUE = The axis is homed and is capable of executing absolute positioning commands. The axis does not have to be homed for relative positioning. The status is FALSE during active homing. The status remains TRUE during passive homing if the axis was already homed beforehand.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Done				
Command execution on the axis FALSE = A Motion Control command is active on the axis. TRUE = A Motion Control command is not active on the axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Error				
Error status on the axis FALSE = No error is active on the axis. TRUE = An error occurred on the axis. Additional information about the error is available in automatic control at the "ErrorID" and "ErrorInfo" parameters of the Motion Control instructions. In manual mode, the "Error message" box of the axis control panel displays detailed information about the cause of error.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.StandStill				
Standstill of the axis FALSE = The axis is in motion. TRUE = The axis is at a standstill.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.PositioningCommand				
Execution of a positioning command FALSE = A positioning command is not active on the axis. TRUE = The axis is executing a positioning command of the "MC_MoveRelative" or MC_MoveAbsolute Motion Control instruction.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.VelocityCommand				
Execution of a command with velocity specification FALSE = A command with velocity specification is not active on the axis. TRUE = The axis is executing a motion command with velocity specification of the "MC_MoveVelocity" or MC_MoveJog Motion Control instruction.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.HomingCommand				
Execution of a homing command FALSE = A homing command is not active on the axis. TRUE = The axis is executing a homing command of the "MC_Home" Motion Control instruction.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.CommandTableActive				
Execution of a command table FALSE = A command table is not active on the axis. TRUE = The axis is controlled by "MC_CommandTable" Motion Control instruction.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.ConstantVelocity				
Constant velocity FALSE = The axis is accelerating, decelerating, or at a standstill. TRUE = The setpoint velocity has been reached. The axis is moving at constant velocity.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Accelerating				
Acceleration process FALSE = The axis is decelerating, moving at constant velocity, or at a standstill. TRUE = The axis is accelerating.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Decelerating				
Deceleration process FALSE = The axis is accelerating, moving at constant velocity, or at a standstill. TRUE = The axis is decelerating.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.ControlPanelActive				
Activation status of the axis control panel FALSE = "Automatic mode" is activated. The user program has control priority over the axis. TRUE = The "Manual control" mode was enabled in the axis control panel. The axis control panel has control priority over the axis. The axis cannot be controlled from the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.DriveReady				
Operational status of the drive FALSE = The drive is not ready. Setpoints will not be executed. TRUE = The drive is ready. Setpoints can be executed.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.RestartRequired				
Restart of axis required FALSE = A restart of the axis is not required. TRUE = Values were modified in the load memory. To download the values to the work memory while the CPU is in RUN mode, the axis must be restarted. Use the MC_Reset Motion Control instruction to do this.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.SWLimitMinActive				
Status of the low software limit switch FALSE = The axis is kept within its configured work area. TRUE = The low software limit switch was reached or exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.SWLimitMaxActive				
Status of the high software limit switch FALSE = The axis is kept within its configured work area. TRUE = The high software limit switch was reached or exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.HWLimitMinActive				
Status of the low hardware limit switch FALSE = The axis is kept within its configured permitted traversing range. TRUE = The low hardware limit switch was reached or exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.HWLimitMaxActive				
Status of the high hardware limit switch FALSE = The axis is kept within its configured permitted traversing range. TRUE = The high hardware limit switch was reached or exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<axis name>.StatusBits.AxisSimulation				
Simulation enabled FALSE = Simulation is disabled TRUE = Simulation is enabled				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<axis name>.StatusBits.NonPositionControlled				
Position control enabled FALSE = The axis is not in position-controlled mode. TRUE = The axis is in position-controlled mode.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

ErrorBits tags as of V6 (S7-1200)

Legend

Data type	Data type of tag
Start value	Start value of tag

Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.ErrorBits.SystemFault				
Internal system error				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.ConfigFault				
Incorrect configuration of the axis				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.DriveFault				
Error in the drive. Loss of the "Drive ready" signal.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.SWLimit				
Software limit switch reached or exceeded				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.HWLimit				
Hardware limit switch reached or exceeded				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP	-	X

<Axis name>.ErrorBits.DirectionFault				
Impermissible motion direction				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP, RP	-	X

<Axis name>.ErrorBits.HWUsed				
Another axis is using the same PTO (Pulse Train Output) and is enabled with "MC_Power".				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<AxisName>.ErrorBits.SensorFault (Positioning axis technology object as of V5)				
Error in the encoder system				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.CommunicationFault (Positioning axis technology object as of V5)				
Communication error Error in communication with a connected device.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.FollowingErrorFault (Positioning axis technology object as of V5)				
Maximum permitted following error exceeded				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.PositioningFault (Positioning axis technology object as of V5)				
Positioning error The axis was not correctly positioned at the end of a positioning motion.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

ControlPanel tags as of V6 (S7-1200)

The "ControlPanel" tags do not contain any user-relevant data. These tags cannot be accessed in the user program.

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Internal tags as of V6 (S7-1200)

The "Internal" tags do not contain any user-relevant data. These tags cannot be accessed in the user program.

See also

Tags of the positioning axis technology object V4...5 (Page 6436)

Tag of the axis technology object V1...3 (Page 6420)

Update of the technology object tags (S7-1200)

The status and error information of the axis indicated in the technology object tags is updated at each cycle control point.

Changes to the values of configuration tags do not take effect immediately. For information on the conditions under which a change takes effect, refer to the detailed description of the relevant tag.

13.2.11.10 Versions V1...5 (S7-1200)

CPU outputs relevant for motion control (technology version V1...3) (S7-1200)

The number of usable drives depends on the CPU, the number of PTOs (pulse train outputs) and the number of available pulse generator outputs.

The following tables provide information about the relevant dependencies:

Maximum number of PTOs

The maximum number of controllable PTOs (drives) depends on the article number of the CPU:

CPU article number	Number of PTOs
xxxxxxx-1xx**30**-xxxx	2
xxxxxxx-1xx**31**-xxxx	4

The maximum number of controllable PTOs (drives) applies regardless of the use of a signal board.

Usable pulse generator outputs

The CPU provides one pulse output and one direction output for controlling a stepper motor drive or a servo motor drive with pulse interface. The pulse output provides the drive with the pulses required for motor motion. The direction output controls the travel direction of the drive.

Pulse and direction outputs are permanently assigned to one another and form a pulse generator output. Onboard CPU outputs or outputs of a signal board can be used as pulse generator outputs. You select between onboard CPU outputs and outputs of the signal board during device configuration under Pulse generators (PTO/PWM) on the "Properties" tab.

The following table shows the number of usable drives per CPU or signal board:

CPU		On-board	Signal board				
			DI2/DO2 x DC24V 20kHz	DI2/DO2 x DC24V 200kHz	DO4 x DC24V 200kHz	DI2/DO2 x DC5V 200kHz	DO4 x DC5V 200kHz
CPU 1211C, CPU 1212C, CPU 1214C (MLFB - article number xxxxxxx-1xx**30**-xxxx)	DC/DC/DC	2	2	2	2	2	2
	AC/DC/RLY	-	1	1	2	1	2
	DC/DC/RLY	-	1	1	2	1	2
CPU 1211C (MLFB - article number xxxxxxx-1xx**31**-xxxx)	DC/DC/DC	2	3	3	4	3	4
	AC/DC/RLY	-	1	1	2	1	2
	DC/DC/RLY	-	1	1	2	1	2
CPU 1212C (MLFB - article number xxxxxxx-1xx**31**-xxxx)	DC/DC/DC	3	4	4	4	4	4
	AC/DC/RLY	-	1	1	2	1	2
	DC/DC/RLY	-	1	1	2	1	2
CPU 1214C (MLFB - article number xxxxxxx-1xx**31**-xxxx)	DC/DC/DC	4	4	4	4	4	4
	AC/DC/RLY	-	1	1	2	1	2
	DC/DC/RLY	-	1	1	2	1	2
CPU 1215C	DC/DC/DC	4	4	4	4	4	4
	AC/DC/RLY	-	1	1	2	1	2
	DC/DC/RLY	-	1	1	2	1	2

The following table shows the address assignment of the pulse and direction outputs:

CPU S7-1200		PTO1 outputs [1]		PTO2 outputs [2]		PTO3 outputs [1]		PTO4 outputs [2]	
		Pls.	Dir.	Pls.	Dir.	Pls.	Dir.	Pls.	Dir.
CPU 1211C, CPU 1212C, CPU 1214C, CPU1215C (DC/DC/DC)	CPU	Ax.0	Ax.1	Ax.2	Ax.3	Ax.4	Ax.5	Ax.6	Ax.7
	Signal board	Ay.0	Ay.1	Ay.2	Ay.3	Ay.0	Ay.1	Ay.2	Ay.3

CPU S7-1200		PTO1 outputs [1]		PTO2 outputs [2]		PTO3 outputs [1]		PTO4 outputs [2]	
		Pls.	Dir.	Pls.	Dir.	Pls.	Dir.	Pls.	Dir.
CPU 1211C, CPU 1212C, CPU 1214C, CPU 1215C (AC/DC/ RLY)	CPU	-	-	-	-	-	-	-	-
	Signal board	Ay.0	Ay.1	Ay.2	Ay.3	Ay.0	Ay.1	Ay.2	Ay.3
CPU 1211C, CPU 1212C, CPU 1214C, CPU 1215C (DC/DC/ RLY)	CPU	-	-	-	-	-	-	-	-
	Signal board	Ay.0	Ay.1	Ay.2	Ay.3	Ay.0	Ay.1	Ay.2	Ay.3

x = Initial byte address of onboard CPU outputs (default value = 0)

y = Initial byte address of signal board outputs (default value = 4)

[1] If a DC/DC/DC CPU variant is used together with a DI2/DO2 signal board, the signals of the PTO1/3 can be generated via the onboard CPU outputs or via the signal board.

[2] If a DC/DC/DC CPU variant is used together with a DO4 signal board, the signals for PTO1/3 or PTO2/4 can be generated via the onboard CPU outputs or via the signal board.

PTO3 and PTO4 are only available for CPUs with the article numbers xxxxxxx-1xx**31**-xxxx.

Note

Access to pulse generator outputs via the process image

The firmware takes control via the corresponding pulse generator outputs if the PTO (Pulse Train Output) is selected and assigned to an axis.

With this takeover of the control function, the connection between the process image and I/O output is also disconnected. Although the user has the option of writing the process image of pulse generator outputs via the user program or watch table, this is not transferred to the I/O output. Accordingly, it is also not possible to monitor the I/O output via the user program or watch table. The information read reflects the value of the process image and does not match the real status of the I/O output.

For all other CPU outputs that are not used permanently by the CPU firmware, the status of the I/O output can be controlled or monitored via the process image, as usual.

Limit frequencies of pulse outputs

The following limit frequencies apply to the pulse outputs:

Pulse output	Limit frequencies for technology object positioning axis V1	Limiting frequencies of the technology object positioning axis V2/V3 with CPU < V3.0	Limiting frequencies of the technology object positioning axis V2/V3 with CPU V3.0
On-board (MLFB article number xxxxxxx-1xx**30**-xxxx)	2 Hz ≤ f ≤ 100 kHz	2 Hz ≤ f ≤ 100 kHz	1 Hz ≤ f ≤ 100 kHz
On-board (MLFB - article number xxxxxxx-1xx**31**xxxx)	2 Hz ≤ f ≤ 100 kHz (PTO 1+2) 2 Hz ≤ f ≤ 20 kHz (PTO 3+4)	2 Hz ≤ f ≤ 100 kHz (PTO 1+2) 2 Hz ≤ f ≤ 20 kHz (PTO 3+4)	1 Hz ≤ f ≤ 100 kHz (PTO 1+2) 1 Hz ≤ f ≤ 20 kHz (PTO 3+4)
Signal board DI2/DO2 x DC24V 20kHz	2 Hz ≤ f ≤ 20 kHz	2 Hz ≤ f ≤ 20 kHz	1 Hz ≤ f ≤ 20 kHz
Signal board DI2/DO2 x DC24V 200kHz	2 Hz ≤ f ≤ 100 kHz	2 Hz ≤ f ≤ 200 kHz	1 Hz ≤ f ≤ 200 kHz
Signal board DO4 x DC24V 200kHz	2 Hz ≤ f ≤ 100 kHz	2 Hz ≤ f ≤ 200 kHz	1 Hz ≤ f ≤ 200 kHz
Signal board DI2/DO2 x DC5V 200kHz	2 Hz ≤ f ≤ 100 kHz	2 Hz ≤ f ≤ 200 kHz	1 Hz ≤ f ≤ 200 kHz
Signal board DO4 x DC5V 200kHz	2 Hz ≤ f ≤ 100 kHz	2 Hz ≤ f ≤ 200 kHz	1 Hz ≤ f ≤ 200 kHz

Drive signals

For motion control, you can optionally parameterize a drive interface for "Drive enabled" and "Drive ready". When using the drive interface the digital output for the drive enable and the digital input for "drive ready" can be freely selected.

Acceleration/deceleration limits

The following limits apply to acceleration and deceleration:

Acceleration/deceleration	Value (CPU < V3.0)	Value (CPU V3.0)
Minimum acceleration/deceleration	2.8E-1 pulses/s^2	5.0E-3 pulses/s^2
Maximum acceleration/deceleration	9.5E+9 pulses/s^2	9.5E+9 pulses/s^2

Jerk limits

The following limits apply to the jerk:

Jerk	Value (CPU < V3.0)	Value (CPU V3.0)
Minimum jerk	4.0E-2 pulses/s^3	4.0E-3 pulses/s^3
Maximum jerk	1.5E+8 pulses/s^3	1.0E+10 pulses/s^3

See also

CPU outputs relevant for motion control (Page 6143)

Configuration dialogs (S7-1200)

V1...3 (S7-1200)

Configuration - General ("Axis" technology object V1...3) (S7-1200)

Configure the basic properties of the "Axis" technology object in the "General" configuration window.

Axis name:

Define the name of the axis or the name of the "Axis" technology object in this box. The technology object is listed under this name in the project navigation.

Hardware interface

The pulses are output to the power unit of the drive by fixed assigned digital outputs.

In CPUs with relay outputs, the pulse signal cannot be output on these outputs because the relays do not support the necessary switching frequencies. A signal board with digital outputs must be used to enable you to work with the PTO (Pulse Train Output) on these CPUs.

Note

The PTO requires the functionality of a fast counter (HSC). An HSC is used for this purpose with CPU version <V3.0. The HSC is then no longer available to the user. An internal HSC is used for this with CPU version ≥ V3.0.

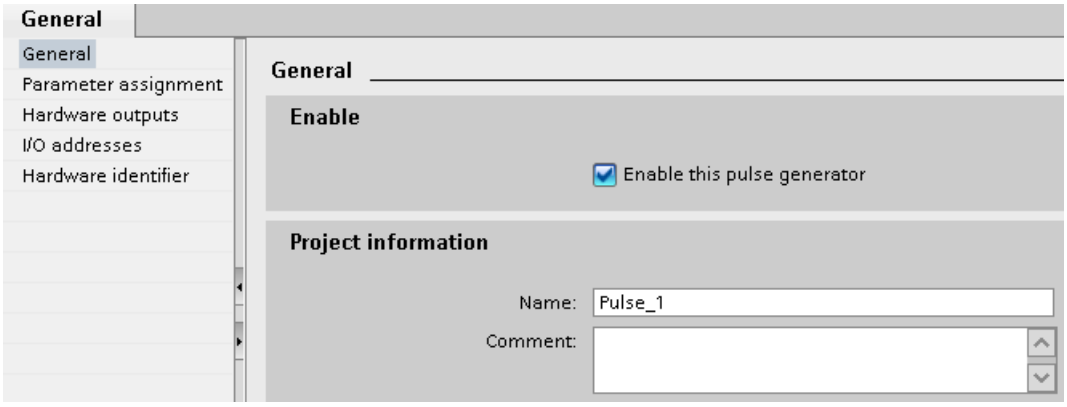
The count can not be evaluated from its input address.

The assignment between PTO and HSC is fixed. When the user activates PTO1, it is connected to the HSC1. If the PTO2 is activated, this is connected with the HSC2.

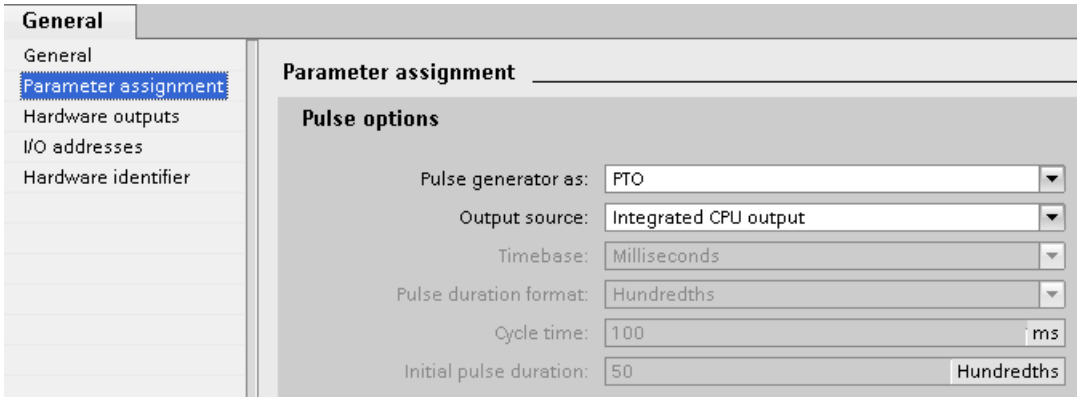
In the "Pulse generator selection" drop-down list, select the PTO (Pulse Train Output) which is to provide the pulses for controlling the stepper motors or servo motors with pulse interface. If the pulse generators and high-speed counters are not used elsewhere in the device configuration, the hardware interface can be configured automatically. In this case, the PTO selected in the drop-down list is displayed with a white background. The interfaces used are listed in the output boxes "Output source", "Pulse output", "Direction output" and "Assigned fast counter".

Proceed as follows if you wish to change the interfaces or if the PTO could not be automatically configured (entry in the "Pulse generator selection" drop-down list is highlighted in red):

1. Click on the "Device configuration" button.
   The pulse generator device configuration opens.
   Enlarge the property window of the device configuration if the configuration of the pulse generator is not visible.



2. Select the "Enable this pulse generator" check box.
3. Select the "Parameter assignment" entry in the area navigation.
   The "Parameter assignment" opens.



4. In the "Pulse generator as:" dropdown list select the "PTO" entry.
5. In the "Output source:" dropdown list select the "Integrated CPU output" or "Signal board output" entry. The "Signal board output" entry can only be selected for PTO1 or for PTO1 and PTO2 depending on the installed signal board. For more detailed information, see section: CPU outputs relevant for motion control (Page 6143)
6. Go back to the axis configuration.
   Unless the corresponding fast counter has already been used elsewhere, the PTO boxes of the "General" axis configuration are not shaded red. If this is not the case, correct the configuration based on the error messages.

User unit

Select the desired unit for the dimension system of the axis in the dropdown list. The selected unit is used for additional configuration of the "Axis" technology object and for displaying the current axis data.

The values at the input parameters (Position, Distance, Velocity, ...) of the Motion Control instructions also refer to this unit.

Note

Later changing of the dimension system may not be converted correctly in all the configuration windows of the technology object. In this case check the configuration of all axis parameters.

The values of the input parameters of the Motion Control instructions may have to be adapted to the new unit of measurement in the user program.

See also

Configuration - General (Page 6184)

Configuration - Homing ("Axis" technology object V1) (S7-1200)

Configure the parameters for active and passive homing in the "Homing" configuration window. The homing method is set using the "Mode" input parameter of the Motion Control instruction. Here, Mode = 2 means passive homing and Mode = 3 means active homing.

Reference point switch input

Select the digital input for the reference point switch from the drop-down list. The input must be interrupt-capable. The onboard CPU inputs and the inputs of an inserted signal board can be selected as inputs for the reference point switch.

Note

The digital inputs are set to a filter time of 6.4 ms by default.

When the digital inputs are used as a reference point switch, this can result in undesired decelerations and thus inaccuracies. Depending on the homing velocity and extent of the reference point switch, the reference point may not be detected. The filter time can be set under "Input filter" in the device configuration of the digital inputs.

The specified filter time must be less than the duration of the input signal at the reference point switch.

Permitting direction reversal after reaching the HW limit switch (active homing only)

Activate the check box to use the hardware limit switch as a reversing cam for the homing procedure. The hardware limit switches must be activated for direction reversal. If the CPU firmware V1.0 is used, both hardware limit switches must be configured. If CPU firmware as of V2.0 is used, only the hardware limit switch in the approach direction must be configured.

If the hardware limit switch is reached during active homing, the axis brakes at the configured deceleration (not with the emergency stop deceleration) and reverses direction. The reference point switch is then sensed in reverse direction.

If the direction reversal is not active and the axis reaches the hardware limit switch during active homing, the homing procedure is aborted with an error and the axis is braked at the emergency stop deceleration.

Note

Use one of the following measures to ensure that the machine does not travel to a mechanical endstop in the event of a direction reversal:

- Keep the approach velocity low
- Increase the configured acceleration/deceleration
- Increase the distance between hardware limit switch and mechanical stop

Approach / homing direction (active and passive homing)

With the direction selection, you determine the "approach direction" used during active homing to search for the reference point switch, as well as the homing direction. The homing direction specifies the travel direction the axis uses to approach the configured end of the reference point switch to carry out the homing operation.

Refer to the table under "Reference point switches" for the effect of the approach direction setting on passive homing.

Side of the reference point switch (active and passive homing)

- Active homing
  This is where you select whether the axis is homed on the low or high end of the reference point switch.

  Note

  Depending on the start position of the axis and the configuration of the homing parameters, the homing procedure sequence can differ from the diagram in the configuration window.

- Passive homing
  With passive homing, the traversing motions for purposes of homing must be implemented by the user via motion commands. The end of the reference point switch on which homing occurs depends on the following factors:

  – "Approach direction" configuration

  – "Reference point switch" configuration

  – Current travel direction during passive homing

  The table below presents details on the effect of factors:

Influencing factors:			Result:
Configuration Approach direction	**Configuration Reference point switch**	**Current travel direction**	**Homing on Reference point switch**
Positive	"Bottom side"	Positive direction	**Top side**
		Negative direction	**Bottom side**
Positive	"Top side"	Positive direction	**Bottom side**
		Negative direction	**Top side**
Negative	"Bottom side"	Positive direction	**Bottom side**
		Negative direction	**Top side**
Negative	"Top side"	Positive direction	**Top side**
		Negative direction	**Bottom side**

Velocity (active homing only)

In this box, specify the velocity at which the reference point switch is to be searched for during the homing procedure.

Limits (independent of the selected unit of measurement):

- Start/stop velocity ≤ approach velocity ≤ maximum velocity

Homing velocity (active homing only)

In this box, specify the velocity at which the axis approaches the reference point switch for homing.

Limits (independent of the selected unit of measurement):

- Start/stop velocity ≤ Homing velocity ≤ Maximum velocity

Home position offset (active homing only)

If the desired home position deviates from the position of the reference point switch, the home position offset can be specified in this box.

If the value does not equal 0, the axis executes the following actions following homing at the reference point switch:

1. Move the axis at the homing velocity by the value of the home position offset
2. Upon reaching the "home position offset", the axis is at the home position that was specified in input parameter "Position" of the "MC_Home" Motion Control instruction.

Limits (independent of the selected unit of measurement):

- -1.0e12 ≤ home position offset ≤ 1.0e12

Home position

The position configured in the Motion Control instruction "MC_Home" is used as the home position.

Configuration - Homing ("Axis" technology object V2...3) (S7-1200)

Configuration - Homing - General (Axis technology object V2...3) (S7-1200)

Configure the reference point switch input for active and passive homing in the "Homing - General" configuration window.

Reference point switch input

Select the digital input for the homing switch from the drop-down list. The input must be interrupt-capable. The on-board CPU inputs and the inputs of an inserted signal board can be selected as inputs for the homing switch.

Note

The digital inputs are set to a filter time of 6.4 ms by default.

When the digital inputs are used as a homing switch, this can result in undesired decelerations and thus inaccuracies. Depending on the homing velocity and extent of the homing switch, the home position may not be detected. The filter time can be set under "Input filter" in the device configuration of the digital inputs.

The specified filter time must be less than the duration of the input signal at the homing switch.

Active level

In the drop-list, select the level of the homing switch that is to be used for homing.

See also

Sequence - Active homing (Page 6215)

Configuration - Homing - Passive (Axis technology object V2...3) (S7-1200)

Configure the necessary parameters for passive homing in the "Homing - Passive" configuration window.

The movement for passive homing must be triggered by the user (e.g. using an axis motion command). Passive homing is started using Motion Control instruction "MC_Home" with input parameter "Mode" = 2.

Side of the homing switch

This is where you select whether the axis is to be homed on the low or high end of the homing switch.

Home position

The position configured in the Motion Control instruction "MC_Home" is used as the home position.

Note

If passive homing is carried out without an axis motion command (axis at a standstill), homing will be executed upon the next rising or falling edge at the homing switch.

Configuration - Homing - Active (Axis technology object V2...3) (S7-1200)

Configure the necessary parameters for active homing in the "Active homing" configuration window. Active homing is started using Motion Control instruction "MC_Home" with input parameter "Mode" = 3.

Permit auto reverse at the hardware limit switch

Activate the check box to use the hardware limit switch as a reversing cam for the homing procedure. The hardware limit switches must be enabled for the reversal of direction (at least the hardware limit switch in the direction of approach must be configured).

If the hardware limit switch is reached during active homing, the axis brakes at the configured deceleration (not with the emergency stop deceleration) and reverses direction. The homing switch is then sensed in reverse direction.

If the direction reversal is not active and the axis reaches the hardware limit switch during active homing, the homing procedure is aborted with an error and the axis is braked at the emergency stop deceleration.

Note

If possible, use one of the following measures to ensure that the machine does not travel to a mechanical endstop in the event of a direction reversal:

- Keep the approach velocity low.
- Increase the configured acceleration/deceleration.
- Increase the distance between the hardware limit switch and the mechanical endstop.

Approach/homing direction

With the direction selection, you determine the approach direction used during active homing to search for the homing switch, as well as the homing direction. The homing direction specifies the travel direction the axis uses to approach the configured end of the homing switch to carry out the homing operation.

Side of the homing switch

This is where you select whether the axis is to be homed on the low or high end of the homing switch.

Velocity

In this box, specify the velocity at which the homing switch is to be searched for during the homing procedure.

Limits (independent of the selected unit of measurement):

- Start/stop velocity ≤ approach velocity ≤ maximum velocity

Homing velocity

In this box, specify the velocity at which the reference point switch is to be approached for homing.

Limits (independent of the selected unit of measurement):

- Start/stop velocity ≤ Homing velocity ≤ Maximum velocity

Home position offset

If the desired home position deviates from the position of the homing switch, the home position offset can be specified in this box.

If the value does not equal 0, the axis executes the following actions following homing at the homing switch:

1. Move the axis at the homing velocity by the value of the home position offset
2. Upon reaching the "home position offset", the axis is at the home position that was specified in input parameter "Position" of the "MC_Home" Motion Control instruction.

Limits (independent of the selected unit of measurement):

- -1.0e12 ≤ home position offset ≤ 1.0e12

Home position

The position configured in the Motion Control instruction "MC_Home" is used as the home position.

Changing configuration of dynamic values in user program ("Axis" technology object V1...3) (S7-1200)

You can change the following configuration parameters during runtime of the user program in the CPU:

Acceleration and deceleration

You can also change the values for acceleration and deceleration during runtime of the user program. Use the following technology object tags for this purpose:

- <Axis name>.Config.DynamicDefaults.Acceleration
  for changing acceleration
- <Axis name>.Config.DynamicDefaults.Deceleration
  for changing deceleration

Refer to the description of the technology object tags (Page 6420) in the appendix for information on when changes to the configuration parameters take effect.

Emergency stop deceleration

You can also change the value for the emergency stop deceleration during runtime of the user program. Use the following technology object tag for this purpose:

- <Axis name>.Config.DynamicDefaults.EmergencyDeceleration

Refer to the description of the technology object tags in the appendix for information on when changes to the configuration parameter take effect.

NOTICE
After changes to this parameter, it may be necessary to adapt the positions of the hardware limit switches and other safety-relevant settings.

Jerk limit (as of technology object "Axis" V2.0)

You can also activate and deactivate the jerk limit at runtime of the user program and change the value for the jerk. Use the following technology object tags for this purpose:

- <Axis name>.Config.DynamicDefaults.JerkActive
  for activating and deactivating the jerk limit
- <Axis name>.Config.DynamicDefaults.Jerk
  for changing the jerk

Refer to the description of the technology object tags in the appendix for information on when changes to the configuration parameter take effect.

See also

Changing the configuration of dynamics in the user program (Page 6209)

V4 (S7-1200)

Configuration - General (positioning axis technology object as of V4) (S7-1200)

Configure the basic properties of the positioning axis technology object in the "General" configuration window.

Axis name

Define the name of the axis or the name of the positioning axis technology object in this field. The technology object is listed under this name in the project tree.

Hardware interface

The pulses are output to the power unit of the drive by fixed assigned digital outputs.

In CPUs with relay outputs, the pulse signal cannot be output on these outputs because the relays do not support the necessary switching frequencies. A signal board with digital outputs must be used to enable you to work with the PTO (Pulse Train Output) on these CPUs.

Note

The PTO requires the functionality of a high-speed counter (HSC). An internal HSC is used for this, the count of which cannot be evaluated.

Selecting the pulse generator

In the drop-down list, select the PTO (Pulse Train Output) which is to provide the pulses for controlling the stepper motors or servo motors with pulse interface. If the pulse generators and high-speed counters are not used elsewhere in the device configuration, the hardware interface can be configured automatically. In this case, the PTO selected in the drop-down list is displayed with a white background.

"Device configuration" button

This button takes you to the settings for the pulse options in the device configuration of the CPU.

Signal type

Select the desired signal type in the drop-down list. The following signal types are available:

- **PTO (pulse A and direction B)**
  A pulse output and a direction output are used for controlling the stepper motor.
- **PTO (count up A, count down B)**
  One pulse output each for motion in positive direction and negative direction is used for controlling the stepper motor.
- **PTO (A/B phase offset)**
  Both pulse outputs for Phase A and for Phase B run at the same frequency.
  The period of the pulse outputs is evaluated at the drive end as a step.
  The phase offset between Phase A and Phase B determines the direction of the motion.
- **PTO (A/B phase offset - quadruple)**
  Both pulse outputs for Phase A and for Phase B run at the same frequency.
  All positive edges and all negative edges of Phase A and Phase B are evaluated as a step at the drive end.
  The phase offset between Phase A and Phase B determines the direction of the motion.

Pulse output (signal type "PTO (pulse A and direction B)")

Select the desired output for the pulse output in this field.

You can select the output using a symbolic address or assign it to an absolute address.

Activate direction output (signal type "PTO (pulse A and direction B)")

You can deactivate and activate the direction output in "pulse and direction" mode. This option enables you to limit the motion direction.

Direction output (signal type "PTO (pulse A and direction B)")

Select the desired output for the direction output in this field.

You can select the output using a symbolic address or assign it to an absolute address.

Pulse output up (signal type "PTO (count up A, count down B)")

Select the desired pulse output for motions in positive direction in this field.

You can select the output using a symbolic address or assign it to an absolute address.

Pulse output down (signal type "PTO (count up A, count down B)")

Select the desired pulse output for motions in negative direction in this field.

You can select the output using a symbolic address or assign it to an absolute address.

Phase A (signal types "PTO (A/B phase offset)" and "PTO (A/B phase offset - quadruple)")

Select the desired pulse output for Phase A signals in this field.

You can select the output using a symbolic address or assign it to an absolute address.

Phase B (signal types "PTO (A/B phase offset)" and "PTO (A/B phase offset - quadruple)")

Select the desired pulse output for Phase B signals in this field.

You can select the output using a symbolic address or assign it to an absolute address.

User unit

Select the desired unit for the dimension system of the axis in the drop-down list. The selected unit is used for further configuration of the positioning axis technology object and for displaying the current axis data.

The values at the input parameters (Position, Distance, Velocity, ...) of the Motion Control instructions also refer to this unit.

Note

Later changing of the dimension system may not be converted correctly in all the configuration windows of the technology object. In this case check the configuration of all axis parameters.

The values of the input parameters of the Motion Control instructions may have to be adapted to the new unit of measurement in the user program.

Configuration - Drive signals (positioning axis technology object V4) (S7-1200)

Configure the output for drive enable and the input for the "Drive ready" feedback signal of the drive in the "Drive signals" configuration window.

Drive enable is controlled by Motion Control instruction "MC_Power" and enables power to the drive. The signal is provided to the drive via the output to be configured.

The drive signals "Drive ready" to the CPU if it is ready to start executing travel after receipt of drive enable. The "Drive ready" signal is reported back to the CPU via the input to be configured.

If the drive does not have any interfaces of this type, you will not have to configure the parameters. In this case, select the value TRUE for the ready input.

See also

Configuration - Mechanics (positioning axis technology object V4) (Page 6379)

Position limits (Page 6198)

Dynamics (Page 6203)

Homing (positioning axis technology object as of V2) (Page 6211)

Configuration - Mechanics (positioning axis technology object V4) (S7-1200)

Configure the mechanical properties of the drive in the "Mechanics" configuration window.

Increments per motor revolution

Configure the number of pulses required for one revolution of the motor in this box.

Limits (independent of the selected unit of measurement):

- 0 < Pulse per motor revolution ≤ 2147483647

Distance per revolution

In this box, configure the load distance per motor revolution covered by the mechanical system of your unit.

Limits (independent of the selected unit of measurement):

- 0.0 < Distance per revolution ≤ 1.0e12

Permitted direction of rotation (technology version as of V4)

Configure this box to determine whether the mechanics of your system are to move in both directions or only in the positive or negative direction.

If you have not activated the direction output of the pulse generator in the "PTO (pulse A and direction B)" mode, the selection is limited to the positive or negative direction.

Invert direction

You can use the "Invert direction" check box to adapt the control system to the direction logic of the drive.

The direction logic is inverted according to the selected mode of the pulse generator:

- **PTO (pulse A and direction B)**
  - 0 V at direction output ⇒ positive direction of rotation
  - 5 V/24 V at direction output ⇒ negative direction of rotation

  The specified voltage depends on the hardware used. The indicated values do not apply to the differential outputs of CPU 1217.
- **PTO (count up A, count down B)**
  The outputs "Pulse output down" and "Pulse output up" are swapped.
- **PTO (A/B phase offset)**
  The "Phase A" and "Phase B" outputs are swapped.
- **"PTO (A/B phase offset - quadruple)**
  The "Phase A" and "Phase B" outputs are swapped.

See also

Configuration - Drive signals (positioning axis technology object V4) (Page 6379)

Relationship between the signal type and the direction of travel (Page 6146)

Position limits (Page 6198)

Dynamics (Page 6203)

Homing (positioning axis technology object as of V2) (Page 6211)

Configuration - Homing - Passive (Positioning axis technology object V4) (S7-1200)

Configure the necessary parameters for passive homing in the "Homing - Passive" configuration window.

The movement for passive homing must be triggered by the user (e.g. using an axis motion command). Passive homing is started using Motion Control instruction "MC_Home" with input parameter "Mode" = 2.

Homing switch input

Select the digital input for the homing switch from the drop-down list. The input must be interrupt-capable. The on-board CPU inputs and the inputs of an inserted signal board can be selected as inputs for the homing switch.

Note

The digital inputs are set to a filter time of 6.4 ms by default.

When the digital inputs are used as a homing switch, this can result in undesired decelerations and thus inaccuracies. Depending on the homing velocity and extent of the homing switch, the home position may not be detected. The filter time can be set under "Input filter" in the device configuration of the digital inputs.

The specified filter time must be less than the duration of the input signal at the homing switch.

Active level

In the drop-list, select the level of the homing switch that is to be used for homing.

Side of the homing switch

This is where you select whether the axis is to be homed on the low or high end of the homing switch.

Home position

The position configured in the Motion Control instruction "MC_Home" is used as the home position.

Note

If passive homing is carried out without an axis motion command (axis at a standstill), homing will be executed upon the next rising or falling edge at the homing switch.

Configuration - Homing - Active (Positioning axis technology object V4) (S7-1200)

Configure the necessary parameters for active homing in the "Active homing" configuration window. Active homing is started using Motion Control instruction "MC_Home" with input parameter "Mode" = 3.

Homing switch input

Select the digital input for the homing switch from the drop-down list. The input must be interrupt-capable. The on-board CPU inputs and the inputs of an inserted signal board can be selected as inputs for the homing switch.

Note

The digital inputs are set to a filter time of 6.4 ms by default.

When the digital inputs are used as a homing switch, this can result in undesired decelerations and thus inaccuracies. Depending on the homing velocity and extent of the homing switch, the home position may not be detected. The filter time can be set under "Input filter" in the device configuration of the digital inputs.

The specified filter time must be less than the duration of the input signal at the homing switch.

Active level

In the drop-list, select the level of the homing switch that is to be used for homing.

Allowing direction reversal at the hardware limit switch

Activate the check box to use the hardware limit switch as a reversing cam for the homing procedure. The hardware limit switches must be enabled for the reversal of direction (at least the hardware limit switch in the direction of approach must be configured).

If the hardware limit switch is reached during active homing, the axis brakes at the configured deceleration (not with the emergency stop deceleration) and reverses direction. The homing switch is then sensed in reverse direction.

If the direction reversal is not active and the axis reaches the hardware limit switch during active homing, the homing procedure is aborted with an error and the axis is braked at the emergency stop deceleration.

Note

If possible, use one of the following measures to ensure that the machine does not travel to a mechanical endstop in the event of a direction reversal:

- Keep the approach velocity low.
- Increase the configured acceleration/deceleration.
- Increase the distance between the hardware limit switch and the mechanical endstop.

Approach/homing direction

With the direction selection, you determine the approach direction used during active homing to search for the homing switch, as well as the homing direction. The homing direction specifies the travel direction the axis uses to approach the configured end of the homing switch to carry out the homing operation.

Side of the homing switch

This is where you select whether the axis is to be homed on the low or high end of the homing switch.

Approach velocity

In this box, specify the velocity at which the homing switch is to be searched for during the homing procedure.

Limits (independent of the selected unit of measurement):

- Start/stop velocity ≤ approach velocity ≤ maximum velocity

Homing velocity

In this box, specify the velocity at which the homing switch is to be approached for homing.

Limits (independent of the selected unit of measurement):

- Start/stop velocity ≤ Homing velocity ≤ Maximum velocity

Home position offset

If the desired home position deviates from the position of the homing switch, the home position offset can be specified in this box.

If the value does not equal 0, the axis executes the following actions following homing at the homing switch:

1. Move the axis at the homing velocity by the value of the home position offset
2. Upon reaching the "home position offset", the axis is at the home position that was specified in input parameter "Position" of the "MC_Home" Motion Control instruction.

Limits (independent of the selected unit of measurement):

- -1.0e12 ≤ home position offset ≤ 1.0e12

Home position

The position configured in the Motion Control instruction "MC_Home" is used as the home position.

Diagnostics - Status and error bits ("Axis" technology object V1...3) (S7-1200)

You use the "Status and error bits" diagnostic function to monitor the most important status and error messages for the axis in the TIA Portal. The diagnostic function display is available in online mode in "Manual control" mode and in "Automatic control" when the axis is active. The status error messages have the following meaning:

Status of the axis

Status	Description
Enabled	The axis is enabled and ready to be controlled via Motion Control commands. (Tag of technology object: <Axis name>.StatusBits.Enable)
Homed	The axis is homed and is capable of executing absolute positioning commands of Motion Control instruction "MC_MoveAbsolute". The axis does not have to be homed for relative positioning. Special situations: • During active homing, the status is FALSE. • If a homed axis undergoes passive homing, the status is set to TRUE during passive homing. (Tag of technology object: <Axis name>.StatusBits.HomingDone)
Axis error	An error has occurred in the "Axis" technology object. Additional information about the error is available in automatic control at the ErrorID and ErrorInfo parameters of the Motion Control instructions. In manual mode, the "Error message" box of the axis control panel displays detailed information about the cause of error. (Tag of technology object: <Axis name>.StatusBits.Error)
Axis control panel enabled	The "Manual control" mode was enabled in the axis control panel. The axis control panel has control priority over the "Axis" technology object. The axis cannot be controlled from the user program. (Tag of technology object: <Axis name>.StatusBits.ControlPanelActive)
Restart necessary	A modified configuration of the axis was downloaded to the load memory in CPU RUN operating mode. To download the modified configuration to the work memory, you need to restart the axis. Use the Motion Control instruction MC_Reset to do this.

Drive status

Status	Description
Drive ready	The drive is ready for operation. (Tag of technology object: <Axis name>.StatusBits.DriveReady)
Drive error	The drive has reported an error due to loss of its "Drive ready" signal. (Tag of technology object: <Axis name>.ErrorBits.DriveFault)

Status of the axis motion

Status	Description
Standstill	The axis is at a standstill. (Tag of technology object: <Axis name>.StatusBits.StandStill)
Acceleration	The axis accelerates. (Tag of technology object: <Axis name>.StatusBits.Acceleration)
Constant velocity	The axis travels at constant velocity. (Tag of technology object: <Axis name>.StatusBits.ConstantVelocity)
Deceleration	The axis decelerates (slows down). (Tag of technology object: <Axis name>.StatusBits.Deceleration)

Status of the motion mode

Status	Description
Positioning	The axis executes a positioning command of the Motion Control instruction "MC_MoveAbsolute", "MC_MoveRelative" or the axis control panel. (Tag of technology object: <Axis name>.StatusBits.PositioningCommand)
Travel with velocity specification	The axis executes a command with velocity specification of the Motion Control instruction "MC_MoveVelocity", "MC_MoveJog" or the axis control panel. (Tag of technology object: <Axis name>.StatusBits.SpeedCommand)
Homing	The axis executes a homing command of the Motion Control instruction "MC_Home" or the axis control panel. (Tag of technology object: <Axis name>.StatusBits.Homing)
Command table active (as of technology object Axis V2.0)	The axis is controlled by Motion Control instruction "MC_CommandTable". (Tag of technology object: <Axis name>.StatusBits.CommandTableActive)

Error messages

Error	Description
Lower SW limit switch was reached	The lower software limit switch has been reached. (Tag of technology object: <Axis name>.ErrorBits.SwLimitMinReached)
Lower SW limit switch was exceeded	The lower software limit switch has been exceeded. (Tag of technology object: <Axis name>.ErrorBits.SwLimitMinExceeded)
Upper SW limit switch was reached	The upper software limit switch has been reached. (Tag of technology object: <Axis name>.ErrorBits.SwLimitMaxReached)
Upper SW limit switch was exceeded	The upper software limit switch has been exceeded. (Tag of technology object: <Axis name>.ErrorBits.SwLimitMaxExceeded)
Lower HW limit switch was reached	The lower hardware limit switch has been reached. (Tag of technology object: <Axis name>.ErrorBits.HwLimitMin)
Upper HW limit switch was reached	The upper hardware limit switch has been reached. (Tag of technology object: <Axis name>.ErrorBits.HwLimitMax)
PTO and HSC already in use	A second axis is using the same PTO (Pulse Train Output) and HSC (High Speed Counter) and is enabled with "MC_Power". (Tag of technology object: <Axis name>.ErrorBits.HwUsed)
Configuration error	The "Axis" technology object was incorrectly configured or editable configuration data were modified incorrectly during runtime of the user program. (Tag of technology object: <Axis name>.ErrorBits.ConfigFault)
Internal error	An internal error has occurred. (Tag of technology object: <Axis name>.ErrorBits.SystemFault)

See also

Status and error bits (technology objects as of V4) (Page 6285)

ErrorIDs and ErrorInfos (S7-1200)

List of ErrorIDs and ErrorInfos (technology objects V4...5) (S7-1200)

The following table lists all ErrorIDs and ErrorInfos that can be indicated in Motion Control instructions. In addition to the cause of the error, remedies for eliminating the error are also listed:

Depending on the error reaction, the axis is stopped in the case of operating errors with stop of axis. The following error reactions are possible:

- **Remove enable**
  The setpoint zero is output and the enable is removed. The axis is braked depending on the configuration in the drive, and is brought to a standstill.
- **Stop with emergency stop ramp**
  Active motion commands are aborted. The axis is braked with the emergency stop deceleration configured under "Technology object > Extended parameters > Dynamics > Emergency stop ramp" without any jerk limit and brought to a standstill.

Operating error with axis stop

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#8000		**Drive error, loss of "Drive ready"**		-
	16#0001	-	Acknowledge error with instruction "MC_Reset"; provide drive signal; restart command, if necessary	
16#8001		**Low SW limit switch has been tripped**		-
	16#000E	The position of the low SW limit switch was reached with the currently configured deceleration	Acknowledge the error with instruction "MC_Reset"; use a motion command to move the axis in the positive direction out of the range of the SW limit switch	
	16#000F	The position of the low SW limit switch was reached with the emergency stop deceleration		
	16#0010	The position of the low SW limit switch was exceeded with the emergency stop deceleration		Remove enable
16#8002		**High SW limit switch has been tripped**		-
	16#000E	The position of the high SW limit switch was reached with the currently configured deceleration	Acknowledge the error with instruction "MC_Reset"; use a motion command to move the axis in the negative direction out of the range of the SW limit switch	
	16#000F	The position of the high SW limit switch was reached with the emergency stop deceleration		
	16#0010	The position of the high SW limit switch was exceeded with the emergency stop deceleration		Remove enable

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#8003		**Low HW limit switch was reached**		For drive connection via PTO (Pulse Train Output): Stop with emergency stop ramp For drive connection via PROFIdrive/analog output: Remove enable
	16#000E	The low HW limit switch was reached. The axis was stopped with the emergency stop deceleration. (During an active homing procedure, the homing switch was not found)	Acknowledge the error for an enabled axis with instruction "MC_Reset"; use a motion command to move the axis in the positive direction out of the range of the HW limit switch.	
16#8004		**High HW limit switch was reached**		For drive connection via PTO (Pulse Train Output): Stop with emergency stop ramp For drive connection via PROFIdrive/analog output: Remove enable
	16#000E	The high HW limit switch has been reached. The axis was stopped with the emergency stop deceleration. (During an active homing procedure, the homing switch was not found)	Acknowledge the error for an enabled axis with instruction "MC_Reset"; use a motion command to move the axis in the negative direction out of the range of the HW limit switch.	
16#8005		**PTO/HSC are already being used by another axis**		-
	16#0001	-	**The axis was configured incorrectly:** Correct the configuration of the PTO (Pulse Train Output) / HSC (High Speed Counter) and download it to the controller	
			More than one axis is to run with one PTO: Another axis is using the PTO / HSC. If the current axis is to assume the control, the other axis must be disabled with "MC_Power" Enable = FALSE. (see also Using multiple axes with the same PTO (Page 6290))	
16#8006		**A communication error in the control panel has occurred**		Remove enable
	16#0012	A timeout has occurred	Check the cable connection and press the "Manual control" button again	
16#8007		**The axis cannot be enabled**		-
	16#0025	Restarting	Wait until the axis restart is complete.	
	16#0026	Executing loading process in RUN mode	Wait until the loading process is complete.	

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#8008		**Invalid direction of motion**		-
	16#002E	The selected motion direction is not allowed.	• Adjust the motion direction and restart the command. • Adjust the allowed direction of rotation in the technology object configuration under "Extended parameters > Mechanics". Restart the command.	
	16#002F	A reversing motion is not possible with the selected motion direction.		
16#8009		**Reference switch/encoder zero mark not found**		Stop with emergency stop ramp
	16#0033	Error in the configuration, hardware or installation of the encoder or at the homing switch.	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.	
16#800A		**Alarm message from encoder**		Remove enable
	16#0001	-	Check the device with regard to function, connections and I/Os.	
	16#0034	Hardware error at encoder		
	16#0035	Encoder dirty		
	16#0036	Error during reading of encoder absolute value	Compare the encoder type in the drive or encoder parameter P979 with the configuration data of the technology object.	
	16#0037	Zero mark monitoring of the encoder	Encoder reports error in zero mark monitoring (fault code 0x0002 in Gx_XIST2, see PROFIdrive profile). Check the plant for electromagnetic compatibility (EMC).	
	16#0038	Encoder is in "Parking" state	• Search for the cause of the error in the connected drive or encoder. • Check whether the error message was possibly triggered by a commissioning action at the drive or encoder.	
	16#0040	PROFIdrive: Encoder on bus failed (station failure).	Check the device with regard to function, connections and I/Os.	
	16#0041	PROFIdrive: Signs of life of encoder faulty.		
16#800B		**Range violation of the position**		Remove enable
	16#0039	Range violation in positive direction	Home the axis to a valid actual value range.	
	16#003A	Range violation in negative direction		
	16#003B	The change of the actual position in a position control clock cycle is greater than the modulo length.	Adjust the modulo length of the employed encoder.	

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#800C		**Alarm message from drive**		Remove enable
	16#0001	-	Check the device with regard to function, connections and I/Os. In the "Tuning" dialog box, select a smaller gain (Kv).	
	16#003C	PROFIdrive: Drive signal "Control requested" failed.		
	16#003D	PROFIdrive/analog drive connection: Drive has shut down.		
	16#003E	PROFIdrive: Drive on bus failed (station failure)		
	16#003F	PROFIdrive: Signs of life of drive faulty.	• Check the device with regard to function, connections and I/Os. • Compare the clock parameters of HW Config (PROFIBUS line, slave OM for drive or encoder) and the execution system. Tmapc and servo must be configured with the same clock cycle time.	
16#800D		**The permitted following error was exceeded**		Remove enable
	16#0001	-	• Check the configuration of the control loop. • Check the direction signal of the encoder. • Check the configuration of following error monitoring.	
16#800E		**Error at the hardware limit switch**		Remove enable
	16#0042	Illegal free travel direction with active hardware limit switch	The programmed direction of movement is disabled due to the active hardware limit switch. Retract the axis in the opposite direction.	
	16#0043	Hardware limit switch polarity is reversed, axis cannot be freed	Check the mechanical configuration of the hardware limit switch.	
	16#0044	Both hardware limit switches are active, axis cannot be freed		
16#800F		**Error in target range**		Remove enable
	16#0045	Target range not reached	Target range was not reached within the positioning tolerance time. • Check the configuration of the position monitoring. • Check the configuration of the control loop.	
	16#0046	Exit target range again	The target range was exited within the minimum dwell time. • Check the configuration of the position monitoring. • Check the configuration of the control loop.	

ErrorID	ErrorInfo	Description	Remedy	Error reaction
16#8010		**Position of the low SW limit switch is greater than that of the high SW limit switch when the axis is not a modulo axis**		Remove enable
	16#0001	-	Change the position of the software limit switches.	

Operating error without axis stop

ErrorID	ErrorInfo	Description	Remedy
16#8200		**Axis is not enabled**	
	16#0001	-	Enable the axis; restart the command
16#8201		**Axis has already been enabled by another "MC_Power" instance**	
	16#0001	-	Enable the axis with only one "MC_Power" instance
16#8202		**The maximum number of simultaneous motion control commands has been exceeded (max. 200 commands for drive connection via PTO (Pulse Train Output), max. 100 commands for drive connection via PROFIdrive/analog output)**	
	16#0001	-	Reduce the number of simultaneously active commands; restart the command A command is active if parameter "Busy" = TRUE in the Motion Control instruction.
16#8203		**Axis is currently operated in "Manual control" (axis control panel)**	
	16#0001	-	Exit "Manual control"; restart the command
16#8204		**Axis is not homed**	
	16#0001	-	Home the axis with instruction "MC_Home"; restart the command
16#8205		**The axis is currently controlled by the user program (the error is only displayed in the axis control panel)**	
	16#0013	The axis is enabled in the user program	Disable axis with instruction "MC_Power" and select "Manual control" again in the axis control panel
16#8206		**Technology object not activated yet**	
	16#0001	-	Enable the axis with instruction "MC_Power" Enable = TRUE or enable the axis in the axis control panel.
16#8207		**Command rejected**	

ErrorID	ErrorInfo	Description	Remedy
	16#0016	Active homing is running; another homing method cannot be started.	Wait for active homing to finish or abort the active homing with a motion command, for example, "MC_Halt".
	16#0018	The axis cannot be moved with a command table while it is being actively or passively homed.	Wait until direct or passive homing is complete.
	16#0019	The axis cannot be actively or passively homed while a command table is being processed.	Wait for command table to finish or abort the command table with a motion command, for example, "MC_Halt".
	16#0052	The specified position exceeds the numerical limit.	Enter a valid position value at the Motion Control instruction.
	16#0053	The axis is ramping up.	Wait until the axis is ready for operation.
	16#0054	Actual value is invalid	To execute a "MC_Home" command, the actual values must be valid. Check the status of the actual values. The tag of the technology object <Axis name>."StatusSensor.State" must show the value 2 (valid).
16#8208		**Difference between maximum and start/stop velocity is invalid**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#8209		**Invalid acceleration for technology object "Axis"**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#820A		**It is not possible to restart the axis**	
	16#0013	The axis is enabled in the user program	Disable the axis with the "MC_Power" instruction; restart again
	16#0027	The axis is currently being operated in "Manual control" (axis control panel)	Exit "Manual control"; restart again
	16#002C	The axis is not disabled.	Disable the axis; restart the command
	16#0047	The technology object is not ready for restart.	Download the project again.
	16#0048	Condition for restart of the technology object is not satisfied.	Disable the technology object.
16#820B		**It is not possible to execute the command table**	
	16#0026	Executing loading process in RUN mode	Wait until the loading process is complete.
16#820C		**No configuration available**	
	16#0001	-	Internal error Contact the hotline.

Block parameter error

ErrorID	ErrorInfo	Description	Remedy
16#8400		**Invalid value at parameter "Position" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	

ErrorID	ErrorInfo	Description	Remedy
16#8401		**Invalid value at parameter "Distance" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
16#8402		**Invalid value at parameter "Velocity" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0008	Value is greater than the configured maximum velocity	
	16#0009	Value is less than the configured start/stop velocity	
	16#0024	Value is less than 0	
16#8403		**Invalid value at parameter "Direction" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; restart the command
16#8404		**Invalid value at parameter "Mode" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; restart the command
	16#0015	Active/passive homing is not configured	Correct the configuration and download it to the controller; enable the axis and restart the command
	16#0017	The direction reversal is activated at the hardware limit switch, despite the fact that the hardware limit switches are disabled	• Activate the HW limit switch using the tag <Axis name>.PositionLimitsHW.Active = TRUE, restart the command • Correct the configuration and download it to the controller; enable the axis and restart the command
	16#0055	Invalid mode at incremental encoder	Start a homing process for an incremental encoder using parameter "Mode" = 0, 1, 2, 3.
	16#0056	Invalid mode at absolute encoder	Passive and active homing ("Mode" = 2, 3) are not possible for an absolute value encoder. Start a homing process for an absolute encoder using parameter "Mode" = 0, 1.
16#8405		**Invalid value at parameter "StopMode" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; enable the axis again
16#8406		**Simultaneous forward and backward jogging is not allowed**	
	16#0001	-	Take steps to ensure that parameters "JogForward" and "JogBackward" do not have signal status TRUE simultaneously; restart the command.
16#8407		**Switching to another axis with instruction "MC_Power" is only permitted after disabling the active axis.**	
	16#0001	-	Disable the active axis; it is then possible to switch to the other axis and enable it.
16#8408		**Invalid value at parameter "Axis" of the Motion Control instruction**	

ErrorID	ErrorInfo	Description	Remedy
	16#001A	The specified value does not match the required technology object version	Correct the value; restart the command
	16#001B	The specified value does not match the required technology object type	
	16#001C	The specified value is not a Motion Control technology data block	
16#8409		**Invalid value at parameter "CommandTable" of the Motion Control instruction**	
	16#001A	The specified value does not match the required technology object version	Correct the value; restart the command
	16#001B	The specified value does not match the required technology object type	
	16#001C	The specified value is not a Motion Control technology data block	
16#840A		**Invalid value at parameter "StartStep" of the Motion Control instruction**	
	16#000A	Value is less than or equal to 0.	Correct the value; restart the command
	16#001D	The start step is greater than the end step	
	16#001E	Value is greater than 32	
16#840B		**Invalid value at parameter "EndStep" of the Motion Control instruction**	
	16#000A	Value is less than or equal to 0.	Correct the value; restart the command
	16#001E	Value is greater than 32	
16#840C		**Invalid value at parameter "RampUpTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840D		**Invalid value at parameter "RampDownTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840E		**Invalid value at parameter "EmergencyRampTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840F		**Invalid value at parameter "JerkTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#8410		**Invalid value at parameter "Parameter" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000B	Address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0028	Data type of VARIANT pointer "Parameter" and "Value" do not match.	Use a suitable data type; restart command
	16#0029	VARIANT pointer "Parameter" does not point to a data block of the technology object.	Correct the VARIANT pointer; restart the command
	16#002A	The value at the VARIANT pointer "Parameter" cannot be read.	Correct the VARIANT pointer; restart the command
	16#002B	The value at the VARIANT pointer "Parameter" cannot be written.	Correct the VARIANT pointer or value; restart the command
	16#002C	The axis is not disabled.	Disable the axis; restart the command

ErrorID	ErrorInfo	Description	Remedy
16#8411		**Invalid value at parameter "Value" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command

Configuration error of the axis

ErrorID	ErrorInfo	Description	Remedy
16#8600		**Parameter assignment of pulse generator (PTO is invalid**	
	16#000B	The address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0014	The selected hardware is used by another application	
16#8601		**Parameter assignment of the high-speed counter (HSC) is invalid**	
	16#000B	The address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0014	The selected hardware is used by another application	
16#8602		**Invalid parameter assignment of "Enable output"**	
	16#000B	The address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8603		**Invalid parameter assignment of "Ready input"**	
	16#000B	The address is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8604		**Invalid "Pulses per motor revolution" value**	
	16#000A	Value is less than or equal to zero	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8605		**Invalid "Distance per revolution" value**	
	16#0002	The value has an invalid number format	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#000A	Value is less than or equal to zero	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8606		**Invalid "Start/stop velocity" value**	
	16#0002	The value has an invalid number format	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0007	The start/stop velocity is greater than the maximum velocity	
16#8607		**Invalid "maximum velocity" value**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8608		**Invalid "Acceleration" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8609		**Invalid "Deceleration" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#860A		**Invalid "Emergency stop deceleration" value**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#860B		**Value for position of the low SW limit switch is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
	16#0030	The position value of the low software limit switch is greater than that of the high software limit switch	
16#860C		**Value for position of the high SW limit switch is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#860D		**Invalid address of the low HW limit switch**	
	16#000B	Invalid address	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000C	The address of the falling edge is invalid	
	16#000D	The address of the rising edge is invalid	
16#860E		**Invalid address of the high HW limit switch**	
	16#000B	Invalid address	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000C	The address of the falling edge is invalid	
	16#000D	The address of the rising edge is invalid	
16#860F		**Invalid "home position offset" value**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8610		**Invalid "approach velocity" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0008	The velocity is greater than the maximum velocity	
	16#0009	The velocity is less than the start/stop velocity	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8611		**Invalid "Homing velocity" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0008	The velocity is greater than the maximum velocity	
	16#0009	The velocity is less than the start/stop velocity	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8612		**Invalid address of the homing switch**	
	16#000B	Invalid address	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000C	The address of the falling edge is invalid	
	16#000D	The address of the rising edge is invalid	
16#8613		**During active homing, direction reversal at the hardware limit switch is activated although the hardware limit switches are not configured**	

ErrorID	ErrorInfo	Description	Remedy
	16#0001	-	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8614		**Invalid "Jerk" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#001F	Value is greater than the maximum jerk	
	16#0020	Value is less than the minimum jerk	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8615		**Value for "Unit of measurement" is invalid**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8616		**Address of homing switch is invalid (passive homing as of V4)**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8617		**Value of tag <Axis name>.Sensor.Sensor[1].ActiveHoming.Mode is invalid**	
	16#0011	The selection value is invalid (Valid value: 2 = Homing via digital input)	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8618		**Value of tag <Axis name>.Sensor.Sensor[1].PassiveHoming.Mode is invalid**	

ErrorID	ErrorInfo	Description	Remedy
	16#0011	The selection value is invalid (Valid value: 2 = Homing via digital input)	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8619		**Value of tag <Axis name>.Actor.Type is invalid**	
	16#0011	The selection value is invalid (Valid value: 2 = Connection via pulse interface)	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#861A		**Value for "Permitted direction of rotation" is invalid**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#002D	"Both directions" not allowed when direction output is deactivated	
16#861B		**Faulty load gear factors**	
	16#0031	Valid is invalid.	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#861C		**Illegal combination of data for homing with incremental encoder**	
	16#0031	Valid is invalid.	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#861D		**The set encoder mounting type is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Mounting-Mode**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#861E		**The configuration of the measuring wheel circumference of the encoder is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.DistancePerRevolution**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary

ErrorID	ErrorInfo	Description	Remedy
16#861F		**The configuration for the resolution of the linear encoder is faulty. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.Resolution**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8620		**The set fine resolution for Gn_XIST1 is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.FineResolutionXist1**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8621		**The set fine resolution for Gn_XIST1 in <Axis name>.Sensor.Sensor[1].Parameter.FineResolutionXist1 is not consistent with the setting in PROFIdrive parameter P979**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8622		**Invalid value for the configuration date <Axis name>.Actor.Interface.AddressIn or <Axis name>.Actor.Interface.AddressOut**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8623		**The value set in the tag <Axis name>.Sensor.Sensor[1].Type is invalid.**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8624		**The set encoder system is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].System**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8625		**Parameter of position monitoring is faulty. Invalid value in <Axis name>.PositioningMonitoring.MinDwellTime**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary

ErrorID	ErrorInfo	Description	Remedy
16#8626		**Parameter of position monitoring is faulty. Invalid value in <Axis name>.PositioningMonitoring.Window**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8627		**The configuration of the PROFIdrive interface of the actual value is faulty. Invalid value in <Axis name>.Sensor.Sensor[1].Interface.AddressIn or <Axis name>.Sensor.Sensor[1].Interface.AddressOut**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8628		**Faulty controller factors**	
	16#0030	Value has an incorrect number format or is outside the valid number range	The value for the gain or the precontrol of the control loop is faulty. • Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary (<Axis name>.PositionControl.Kv, <Axis name>.PositionControl.Kpc)
16#8629		**Limit for standstill signal is faulty. Invalid value in <Axis name>.StandStillSignal.VelocityThreshold**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862A		**Parameter of position monitoring is faulty. Invalid value in <Axis name>.PositioningMonitoring.ToleranceTime**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862B		**Inconsistent PROFIBUS parameterization; the sum of Ti and To is greater than one DP cycle**	
	16#0030	Value has an incorrect number format or is outside the valid number range	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#862C		**Parameter of standstill monitoring is faulty. Invalid value in <Axis name>.StandStillSignal.MinDwellTime**	

ErrorID	ErrorInfo	Description	Remedy
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862D		**Parameter of following error monitoring is faulty. Invalid value in <Axis name>.FollowingError.MinValue**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862E		**Invalid value for the configuration date <Axis name>.Modulo.Length**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#862F		**Invalid value for the configuration date <Axis name>.Modulo.StartValue**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8630		**Invalid value for the configuration date <Axis name>.Actor.DriveParameter.ReferenceSpeed**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8631		**The set fine resolution for Gn_XIST2 is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.FineResolutionXist2**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8632		**The number of determinable encoder revolutions is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].Parameter.DeterminableRevolutions**	

ErrorID	ErrorInfo	Description	Remedy
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8633		**The specified approach direction of the homing switch for passive homing is invalid. Invalid value in <Axis name>.Sensor.Sensor[1].PassiveHoming.Direction**	
16#8634		**Parameter of the following error monitoring is faulty. Invalid value in <Axis name>.FollowingError.MaxValue**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8635		**Parameter of the following error monitoring is faulty. Invalid value in <Axis name>.FollowingError.MinVelocity**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8636		**Controller factor is incorrect. Invalid value of the precontrol factor <Axis name>.PositionControl.Kpc**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8637		**Invalid value for the configuration date <Axis name>.Sensor.Sensor[1].Interface.Type**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8638		**Invalid value for the configuration date <Axis name>.Sensor.Sensor[1].Interface.HSC**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8639		**Error at the drive**	
	16#0049	Configuration error at device	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#004A	The technology needs a smaller servo clock.	Internal system error. Check the project for consistency and reload it into the controller.
	16#004B	Device driver not initialized during ramp-up.	To enable a technology object, the actuator driver must be initialized. Execute the command again later.

ErrorID	ErrorInfo	Description	Remedy
16#863A		**Communication to the drive is faulty**	
	16#004C	Configuration error at device	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.
	16#004D	The device driver needs a smaller servo clock.	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.
	16#004E	Error in internal communication with the device	Check the project for consistency and reload it into the controller.
16#863B		**Error at encoder**	
	16#0049	Configuration error at device	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#004A	The technology needs a smaller servo clock.	Internal system error. Check the project for consistency and reload it into the controller.
	16#004B	Device driver not initialized during ramp-up.	To enable a technology object, the actuator driver must be initialized. Execute the command again later.
16#863C		**Communication with encoder is faulty**	
	16#004C	Configuration error at device	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.
	16#004D	The device driver needs a smaller servo clock.	• Connect a suitable device. • Check the device (I/Os). • Compare the configuration of HW Config and the technology object.
	16#004E	Error in internal communication with the device	Check the project for consistency and reload it into the controller.
16#863D		**Communication to the device (drive or encoder) is faulty**	
	16#0030	Value has an incorrect number format or is outside the valid number range	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0055	The requested logical address is invalid.	• Connect a suitable device. • Check the device (I/Os). • Check the topological configuration in HW Config. • Compare the configuration of HW Config and the technology object.
	16#0056	The requested logical output address is invalid.	
	16#0057	The requested logical output address is invalid.	

ErrorID	ErrorInfo	Description	Remedy
16#863E		**Value of tag "ControlPanel.Input.TimeOut" is invalid (axis control panel)**	
	16#0030	Value has an incorrect number format or is outside the valid number range	Correct the value in the tags of the technology object <Axis name>.ControlPanel.Input.TimeOut. The value is specified in milliseconds (ms).
16#863F		**Invalid value for the configuration date <Axis name>.Actor.DriveParameter.MaxSpeed**	
	16#0030	Value has an incorrect number format or is outside the valid number range	Correct the reference value in the drive and in the configuration of the technology object to Actuator.MaxSpeed/2. With analog drive connection, correct the reference value in the drive and in the configuration of the technology object to Actuator.MaxSpeed/1.17.

Configuration error of the command table

ErrorID	ErrorInfo	Description	Remedy
16#8700		**Value for "Command type" in the command table is invalid**	
	16#0001	-	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
16#8701		**Value for "Position / travel path" in the command table is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1E+12)	
	16#0006	Value is outside the number range (less than -1E+12)	
16#8702		**Value for "Velocity" in the command table is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0008	Value is greater than the configured maximum velocity	
	16#0009	Value is less than the configured start/stop velocity	
16#8703		**Value for "Duration" in the command table is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0021	Value is greater than 64800 s	
	16#0022	Value is less than 0.001 s	
16#8704		**Value for "Next step" in the command table is invalid**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0023	The command transition is not permitted for this command	

Internal errors

ErrorID	ErrorInfo	Description	Remedy
16#8FFF		**Internal error**	
	16#F0**	-	POWER OFF and POWER ON the CPU If this does not work, contact Customer Support. Have the following information ready: • ErrorID • ErrorInfo • Diagnostic buffer entries

See also

Error displays of the Motion Control statements (Page 6283)

Tags of the positioning axis technology object V4...5 (Page 6436)

ErrorIDs and ErrorInfos (Page 6386)

Using multiple axes with the same PTO (Page 6290)

Using multiple drives with the same PTO (Page 6293)

Tracking jobs from higher priority classes (execution levels) (Page 6294)

Special cases when using software limit switches for drive connection via PTO (Page 6296)

List of ErrorIDs and ErrorInfos (technology objects V2...3) (S7-1200)

The following table lists all ErrorIDs and ErrorInfos that can be indicated in Motion Control instructions. In addition to the cause of the error, remedies for eliminating the error are also listed:

Operating error with axis stop

ErrorID	ErrorInfo	Description	Remedy
16#8000		**Drive error, loss of "Drive ready"**	
	16#0001	-	Acknowledge error with instruction "MC_Reset"; provide drive signal; restart command, if necessary
16#8001		**Low SW limit switch has been tripped**	
	16#000E	The position of the low SW limit switch was reached with the currently configured deceleration	Acknowledge the error with instruction "MC_Reset"; use a motion command to move the axis in the positive direction out of the range of the SW limit switch
	16#000F	The position of the low SW limit switch was reached with the emergency stop deceleration	
	16#0010	The position of the low SW limit switch was exceeded with the emergency stop deceleration	
16#8002		**High SW limit switch has been tripped**	

ErrorID	ErrorInfo	Description	Remedy
	16#000E	The position of the high SW limit switch was reached with the currently configured deceleration	Acknowledge the error with instruction "MC_Reset"; use a motion command to move the axis in the negative direction out of the range of the SW limit switch
	16#000F	The position of the high SW limit switch was reached with the emergency stop deceleration	
	16#0010	The position of the high SW limit switch was exceeded with the emergency stop deceleration	
16#8003		**Low HW limit switch was reached**	
	16#000E	The low HW limit switch was reached. The axis was stopped with the emergency stop deceleration. (During an active homing procedure, the homing switch was not found)	Acknowledge the error for an enabled axis with instruction "MC_Reset"; use a motion command to move the axis in the positive direction out of the range of the HW limit switch.
16#8004		**High HW limit switch was reached**	
	16#000E	The high HW limit switch has been reached. The axis was stopped with the emergency stop deceleration. (During an active homing procedure, the homing switch was not found)	Acknowledge the error for an enabled axis with instruction "MC_Reset"; use a motion command to move the axis in the negative direction out of the range of the HW limit switch.
16#8005		**PTO/HSC are already being used by another axis**	
	16#0001	-	**The axis was configured incorrectly:** Correct the configuration of the PTO (Pulse Train Output) / HSC (High Speed Counter) and download it to the controller
			More than one axis is to run with one PTO: Another axis is using the PTO / HSC. If the current axis is to assume the control, the other axis must be disabled with "MC_Power" Enable = FALSE. (see also Using multiple axes with the same PTO (Page 6290))
16#8006		**A communication error in the control panel has occurred**	
	16#0012	A timeout has occurred	Check the cable connection and press the "Manual control" button again
16#8007		**The axis cannot be enabled**	
	16#0025	Restarting	Wait until the axis restart is complete.
	16#0026	Executing loading process in RUN mode	Wait until the loading process is complete.

Operating error without axis stop

ErrorID	ErrorInfo	Description	Remedy
16#8200		**Axis is not enabled**	
	16#0001	-	Enable the axis; restart the command
16#8201		**Axis has already been enabled by another "MC_Power" instance**	
	16#0001	-	Enable the axis with only one "MC_Power" instance
16#8202		**The maximum number of simultaneously active Motion Control commands has been exceeded (maximum of 200 commands for all motion control technology objects)**	

ErrorID	ErrorInfo	Description	Remedy
	16#0001	-	Reduce the number of simultaneously active commands; restart the command A command is active if parameter "Busy" = TRUE in the Motion Control instruction.
16#8203		**Axis is currently operated in "Manual control" (axis control panel)**	
	16#0001	-	Exit "Manual control"; restart the command
16#8204		**Axis is not homed**	
	16#0001	-	Home the axis with instruction "MC_Home"; restart the command
16#8205		**The axis is currently controlled by the user program (the error is only displayed in the axis control panel)**	
	16#0013	The axis is enabled in the user program	Disable axis with instruction "MC_Power" and select "Manual control" again in the axis control panel
16#8206		**Technology object not activated yet**	
	16#0001	-	Enable the axis with instruction "MC_Power" Enable = TRUE or enable the axis in the axis control panel.
16#8207		**Command rejected**	
	16#0016	Active homing is running; another homing method cannot be started.	Wait for active homing to finish or abort the active homing with a motion command, for example, "MC_Halt".
	16#0018	The axis cannot be moved with a command table while it is being actively or passively homed.	Wait until direct or passive homing is complete.
	16#0019	The axis cannot be actively or passively homed while a command table is being processed.	Wait for command table to finish or abort the command table with a motion command, for example, "MC_Halt".
16#8208		**Difference between maximum and start/stop velocity is invalid**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#8209		**Invalid acceleration for technology object "Axis"**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#820A		**It is not possible to restart the axis**	
	16#0013	The axis is enabled in the user program	Disable the axis with the "MC_Power" instruction; restart again
	16#0027	The axis is currently being operated in "Manual control" (axis control panel)	Exit "Manual control"; restart again
16#820B		**It is not possible to execute the command table**	
	16#0026	Executing loading process in RUN mode	Wait until the loading process is complete.

Block parameter error

ErrorID	ErrorInfo	Description	Remedy
16#8400		**Invalid value at parameter "Position" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0005	Value is outside the number range (greater than 1.0E+12)	
	16#0006	Value is outside the number range (less than -1.0E+12)	
16#8401		**Invalid value at parameter "Distance" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0005	Value is outside the number range (greater than 1.0E+12)	
	16#0006	Value is outside the number range (less than -1.0E+12)	
16#8402		**Invalid value at parameter "Velocity" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#0008	Value is greater than the configured maximum velocity	
	16#0009	Value is less than the configured start/stop velocity	
	16#0024	Value is less than 0	
16#8403		**Invalid value at parameter "Direction" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; restart the command
16#8404		**Invalid value at parameter "Mode" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; restart the command
	16#0015	Active/passive homing is not configured	Correct the configuration and download it to the controller; enable the axis and restart the command
	16#0017	The direction reversal is activated at the hardware limit switch, despite the fact that the hardware limit switches are disabled	• Activate the hardware limit switch using the tag <Axis>.Config.PositionLimits_HW.Active = TRUE, restart the command • Correct the configuration and download it to the controller; enable the axis and restart the command
16#8405		**Invalid value at parameter "StopMode" of the Motion Control instruction**	
	16#0011	The selection value is invalid	Correct the selection value; enable the axis again
16#8406		**Simultaneous forward and backward jogging is not allowed**	
	16#0001	-	Take steps to ensure that parameters "JogForward" and "JogBackward" do not have signal status TRUE simultaneously; restart the command.
16#8407		**Switching to another axis with instruction "MC_Power" is only permitted after disabling the active axis.**	
	16#0001	-	Disable the active axis; it is then possible to switch to the other axis and enable it.
16#8408		**Invalid value at parameter "Axis" of the Motion Control instruction**	

ErrorID	ErrorInfo	Description	Remedy
	16#001A	The specified value does not match the required technology object version	Correct the value; restart the command
	16#001B	The specified value does not match the required technology object type	
	16#001C	The specified value is not a Motion Control technology data block	
16#8409		**Invalid value at parameter "CommandTable" of the Motion Control instruction**	
	16#001A	The specified value does not match the required technology object version	Correct the value; restart the command
	16#001B	The specified value does not match the required technology object type	
	16#001C	The specified value is not a Motion Control technology data block	
16#840A		**Invalid value at parameter "StartStep" of the Motion Control instruction**	
	16#000A	Value is less than or equal to 0.	Correct the value; restart the command
	16#001D	The start step is greater than the end step	
	16#001E	Value is greater than 32	
16#840B		**Invalid value at parameter "EndStep" of the Motion Control instruction**	
	16#000A	Value is less than or equal to 0.	Correct the value; restart the command
	16#001E	Value is greater than 32	
16#840C		**Invalid value at parameter "RampUpTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840D		**Invalid value at parameter "RampDownTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840E		**Invalid value at parameter "EmergencyRampTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	
16#840F		**Invalid value at parameter "JerkTime" of the Motion Control instruction**	
	16#0002	The value has an invalid number format	Correct the value; restart the command
	16#000A	Value is less than or equal to 0.	

Configuration error of the axis

ErrorID	ErrorInfo	Description	Remedy
16#8600		**Parameter assignment of pulse generator (PTO is invalid**	
	16#000B	The address is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0014	The selected hardware is used by another application	
16#8601		**Parameter assignment of the high-speed counter (HSC) is invalid**	
	16#000B	The address is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0014	The selected hardware is used by another application	

ErrorID	ErrorInfo	Description	Remedy
16#8602		**Invalid parameter assignment of "Enable output"**	
	16#000B	The address is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8603		**Invalid parameter assignment of "Ready input"**	
	16#000B	The address is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8604		**Invalid "Pulses per motor revolution" value**	
	16#000A	Value is less than or equal to zero	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
16#8605		**Invalid "Distance per revolution" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0005	Value is outside the number range (greater than 1.0E+12)	
	16#000A	Value is less than or equal to zero	
16#8606		**Invalid "Start/stop velocity" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
	16#0007	The start/stop velocity is greater than the maximum velocity	
16#8607		**Invalid "maximum velocity" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
16#8608		**Invalid "Acceleration" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
16#8609		**Invalid "Deceleration" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
16#860A		**Invalid "Emergency stop deceleration" value**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value is higher than the high hardware limit	
	16#0004	Value is lower than the low hardware limit	
16#860B		**Value for position of the low SW limit switch is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1.0E+12)	
	16#0006	Value is outside the number range (less than -1.0E+12)	
	16#0007	The position value of the low software limit switch is greater than that of the high software limit switch	
16#860C		**Value for position of the high SW limit switch is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1.0E+12)	
	16#0006	Value is outside the number range (less than -1.0E+12)	
16#860D		**Invalid address of the low HW limit switch**	
	16#000C	The address of the falling edge is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000D	The address of the rising edge is invalid	
16#860E		**Invalid address of the high HW limit switch**	
	16#000C	The address of the falling edge is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000D	The address of the rising edge is invalid	
16#860F		**Invalid "home position offset" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1.0E+12)	
	16#0006	Value is outside the number range (less than -1.0E+12)	
16#8610		**Invalid "approach velocity" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0008	The velocity is greater than the maximum velocity	
	16#0009	The velocity is less than the start/stop velocity	
16#8611		**Invalid "Homing velocity" value**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0008	The velocity is greater than the maximum velocity	
	16#0009	The velocity is less than the start/stop velocity	
16#8612		**Invalid address of the homing switch**	
	16#000C	The address of the falling edge is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"
	16#000D	The address of the rising edge is invalid	
16#8613		**During active homing, direction reversal at the hardware limit switch is activated although the hardware limit switches are not configured**	
	16#0001	-	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
16#8614		**Invalid "Jerk" value**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#001F	Value is greater than the maximum jerk	
	16#0020	Value is less than the minimum jerk	
16#8615		**Value for "Unit of measurement" is invalid**	
	16#0011	The selection value is invalid	Download error-free configuration to the controller; enable the axis again with instruction "MC_Power"

Configuration error of the command table

ErrorID	ErrorInfo	Description	Remedy
16#8700		**Value for "Command type" in the command table is invalid**	
	16#0001	-	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
16#8701		**Value for "Position / travel path" in the command table is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0005	Value is outside the number range (greater than 1.0E+12)	
	16#0006	Value is outside the number range (less than -1.0E+12)	
16#8702		**Value for "Velocity" in the command table is invalid**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0008	Value is greater than the configured maximum velocity	
	16#0009	Value is less than the configured start/stop velocity	
16#8703		**Value for "Duration" in the command table is invalid**	
	16#0002	The value has an invalid number format	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0021	Value is greater than 64800 s	
	16#0022	Value is less than 0.001 s	
16#8704		**Value for "Next step" in the command table is invalid**	
	16#0011	The selection value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online and restart the command, if necessary
	16#0023	The command transition is not permitted for this command	

Internal errors

ErrorID	ErrorInfo	Description	Remedy
16#8FFF		**Internal error**	
	16#F0**	-	POWER OFF and POWER ON the CPU If this does not work, contact Customer Support. Have the following information ready: • ErrorID • ErrorInfo • Diagnostic buffer entries

See also

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

Using multiple axes with the same PTO (Page 6290)

List of ErrorIDs and ErrorInfos (technology objects V1) (S7-1200)

The following table lists all ErrorIDs and ErrorInfos that can be indicated in Motion Control instructions. In addition to the cause of the error, remedies for eliminating the error are also listed:

Operating error with axis stop

ErrorID	ErrorInfo	Description	Remedy
16#8000		**Drive error, loss of "Drive ready"**	
	16#0001	-	Acknowledge error with instruction "MC_Reset"; provide drive signal; restart command, if necessary
16#8001		**Lower SW limit switch has been tripped**	
	16#000E	The position of the lower SW limit switch was reached with the currently configured deceleration	Acknowledge the error with instruction "MC_Reset"; use a motion command to move the axis in the positive direction out of the range of the SW limit switch
	16#000F	The position of the lower SW limit switch was reached with the emergency stop deceleration	
	16#0010	The position of the lower SW limit switch was exceeded with the emergency stop deceleration	
16#8002		**Upper SW limit switch has been tripped**	
	16#000E	The position of the upper SW limit switch was reached with the currently configured deceleration	Acknowledge the error with instruction "MC_Reset"; use a motion command to move the axis in the negative direction out of the range of the SW limit switch
	16#000F	The position of the upper SW limit switch was reached with the emergency stop deceleration	
	16#0010	The position of the upper SW limit switch was exceeded with the emergency stop deceleration	
16#8003		**Lower HW limit switch was reached**	
	16#000E	The lower HW limit switch was reached. The axis was stopped with the emergency stop deceleration. (During an active homing procedure, the reference point switch was not found)	Acknowledge the error for an enabled axis with instruction "MC_Reset"; use a motion command to move the axis in the positive direction out of the range of the HW limit switch.
16#8004		**Upper HW limit switch was reached**	
	16#000E	The upper HW limit switch has been reached. The axis was stopped with the emergency stop deceleration. (During an active homing procedure, the reference point switch was not found)	Acknowledge the error for an enabled axis with instruction "MC_Reset"; use a motion command to move the axis in the negative direction out of the range of the HW limit switch.
16#8005		**PTO/HSC are already being used by another axis**	
	16#0001	-	**The axis was configured incorrectly:** Correct the configuration of the PTO (Pulse Train Output) / HSC (High Speed Counter) and download it to the controller
			More than one axis is to run with one PTO: Another axis is using the PTO / HSC. If the current axis is to assume the control, the other axis must be disabled with "MC_Power" Enable = FALSE. (see also Using multiple axes with the same PTO (Page 6290))

Operating error without axis stop

ErrorID	ErrorInfo	Description	Remedy
16#8200		**Axis is not enabled**	
	16#0001	-	Enable the axis; restart the command
16#8201		**Axis has already been enabled by another "MC_Power" instance**	
	16#0001	-	Enable the axis with only one "MC_Power" instruction
16#8202		**The maximum number of simultaneously active Motion Control commands was exceeded (maximum of 200 commands for all motion control technology objects)**	
	16#0001	-	Reduce the number of simultaneously active commands; restart the command A command is active if parameter "Busy" = TRUE in the Motion Control instruction.
16#8203		**Axis is currently operated in "Manual control" (axis control panel)**	
	16#0001	-	Exit "Manual control"; restart the command
16#8204		**Axis is not homed**	
	16#0001	-	Home the axis with instruction "MC_Home"; restart the command
16#8205		**The axis is currently controlled by the user program (the error is only displayed in the axis control panel)**	
	16#0001	-	Disable axis with instruction "MC_Power" and select "Manual control" again in the axis control panel
16#8206		**Technology object Axis not yet enabled**	
	16#0001	-	Enable the axis with instruction "MC_Power" Enable = TRUE or enable the axis in the axis control panel.
16#8207		**Command rejected**	
	16#0016	Active homing is running; another homing method cannot be started.	Wait for active homing to finish or abort the active homing with a motion command, for example, "MC_Halt". The other homing type can then be started.

Block parameter error

ErrorID	ErrorInfo	Description	Remedy
16#8400		**Invalid value at parameter "Position" of the Motion Control instruction**	
	16#0002	Number format of value is invalid	Correct the "position" value; restart the command
	16#0005	The value is outside the number range (greater than $1e^{12}$)	
	16#0006	The value is outside the number range (less than $-1e^{12}$)	
16#8401		**Invalid value at parameter "Distance" of the Motion Control instruction**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	Number format of value is invalid	Correct the "Distance" value; restart the command
	16#0005	The value is outside the number range (greater than $1e^{12}$)	
	16#0006	The value is outside the number range (less than $-1e^{12}$)	
16#8402		**Invalid value at parameter "Velocity" of the Motion Control instruction**	
	16#0002	Number format of value is invalid	Correct the "Velocity" value; restart the command
	16#0008	Velocity is greater than the maximum velocity	
	16#0009	Velocity is less than the start/stop velocity	
16#8403		**Invalid value at parameter "Direction" of the Motion Control instruction**	
	16#0011	Invalid selection value	Correct the selection value; restart the command
16#8404		**Invalid value at parameter "Mode" of the Motion Control instruction**	
	16#0011	Invalid selection value	Correct the selection value; restart the command
	16#0015	Active/passive homing is not configured	Correct the configuration and download it to the controller; enable the axis and restart the command
	16#0017	Axis reversal is activated at the HW limit switch, despite the fact that the hardware limit switches are disabled	• Activate the hardware limit switch using the tag <axis>.Config.PositionLimits_HW.Active = TRUE, restart the command • Correct the configuration and download it to the controller; enable the axis and restart the command
16#8405		**Invalid value at parameter "StopMode" of the Motion Control instruction**	
	16#0011	Invalid selection value	Correct the selection value; enable the axis again
16#8406		**Simultaneous forward and backward jogging is not allowed**	
	16#0001	-	Take steps to ensure that parameters "JogForward" and "JogBackward" do not have signal status TRUE simultaneously; restart the command.
16#8407		**Switching the axis with Motion Control instruction "MC_Power" is only permitted after disabling the axis.**	
	16#0001	-	Disable the active axis; it is then possible to switch to the other axis and enable it.

Configuration error

ErrorID	ErrorInfo	Description	Remedy
16#8600		**Parameter assignment of pulse generator (PTO is invalid**	
	16#000B	Address is invalid	Correct the configuration of the PTO (Pulse Train Output) and download it to the controller
16#8601		**Parameter assignment of the high-speed counter (HSC) is invalid**	
	16#000B	Address is invalid	Correct the configuration of the HSC (High Speed Counter) and download it to the controller
16#8602		**Invalid parameter assignment of "Enable output"**	
	16#000D	Address is invalid	Correct the configuration and download it to the controller
16#8603		**Invalid parameter assignment of "Ready input"**	

ErrorID	ErrorInfo	Description	Remedy
	16#000D	Address is invalid	Correct the configuration and download it to the controller
16#8604		**Invalid "Pulses per motor revolution" value**	
	16#000A	Value is less than or equal to zero	Correct the configuration and download it to the controller
16#8605		**Invalid "Distance per revolution" value**	
	16#0002	Number format of value is invalid	Correct the configuration and download it to the controller
	16#000A	Value is less than or equal to zero	
16#8606		**Invalid "Start/stop velocity" value**	
	16#0002	Number format of value is invalid	Correct the configuration and download it to the controller
	16#0003	Value exceeds the hardware limit	
	16#0004	Value is less than the hardware limit	
	16#0007	The start/stop velocity is greater than the maximum velocity	
16#8607		**Invalid "Maximum velocity" value**	
	16#0002	Number format of value is invalid	Correct the configuration and download it to the controller
	16#0003	Value exceeds the hardware limit	
	16#0004	Value is less than the hardware limit	
16#8608		**Invalid "Acceleration" value**	
	16#0002	Number format of value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value exceeds the hardware limit	
	16#0004	Value is less than the hardware limit	
16#8609		**Invalid "Deceleration" value**	
	16#0002	Number format of value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value exceeds the hardware limit	
	16#0004	Value is less than the hardware limit	
16#860A		**Invalid "Emergency stop deceleration" value**	
	16#0002	Number format of value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0003	Value exceeds the hardware limit	
	16#0004	Value is less than the hardware limit	
16#860B		**Value for position of the lower SW limit switch is invalid**	

ErrorID	ErrorInfo	Description	Remedy
	16#0002	Number format of value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	The value is outside the number range (greater than $1e^{12}$)	
	16#0006	The value is outside the number range (less than $-1e^{12}$)	
	16#0007	The position value of the lower SW limit switch is greater than that of the upper SW limit switch	
16#860C		**Value for position of the upper SW limit switch is invalid**	
	16#0002	Number format of value is invalid	• Download error-free configuration to the controller; enable the axis again with instruction "MC_Power" • Correct the incorrect value online; acknowledge error with instruction "MC_Reset" and restart the command, if necessary
	16#0005	The value is outside the number range (greater than $1e^{12}$)	
	16#0006	The value is outside the number range (less than $-1e^{12}$)	
16#860D		**Invalid address of the lower HW limit switch**	
	16#000C	Address of falling edge is invalid	Correct the configuration and download it to the controller
	16#000D	Address of rising edge is invalid	
16#860E		**Invalid address of the upper HW limit switch**	
	16#000C	Address of falling edge is invalid	Correct the configuration and download it to the controller
	16#000D	Address of rising edge is invalid	
16#860F		**Invalid "home position offset" value**	
	16#0002	Number format of value is invalid	Correct the configuration and download it to the controller
	16#0005	The value is outside the number range (greater than $1e^{12}$)	
	16#0006	The value is outside the number range (less than $-1e^{12}$)	
16#8610		**Invalid "approach velocity" value**	
	16#0002	Number format of value is invalid	Correct the configuration and download it to the controller
	16#0008	Velocity is greater than the maximum velocity	
	16#0009	Velocity is less than the start/stop velocity	
16#8611		**Invalid "Homing velocity" value**	
	16#0002	Number format of value is invalid	Correct the configuration and download it to the controller
	16#0008	Velocity is greater than the maximum velocity	
	16#0009	Velocity is less than the start/stop velocity	
16#8612		**Invalid address of the reference point switch**	
	16#000C	Address of falling edge is invalid	Correct the configuration and download it to the controller
	16#000D	Address of rising edge is invalid	
16#8613		**During active homing, direction reversal at the hardware limit switch is activated although the hardware limit switches are not configured**	
	16#0001	-	Correct the configuration and download it to the controller

Internal errors

ErrorID	ErrorInfo	Description	Remedy
16#8FFF		**Internal error**	
	16#F0**	-	POWER OFF and POWER ON the CPU If this does not work, contact Customer Support. Have the following information ready: • ErrorID • ErrorInfo • Diagnostic buffer entries

See also

List of ErrorIDs and ErrorInfos (technology objects V4...5) (Page 6386)

Using multiple axes with the same PTO (Page 6290)

Tag of the axis technology object V1...3 (S7-1200)

Config tags V1...3 (S7-1200)

Config.General tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.Config.General.PTO				
Tag cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWORD	DW#16#00000000	-	-	-

<Axis name>.Config.General.HSC				
Tag cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWORD	DW#16#00000000	-	-	-

<Axis name>.Config.General.LengthUnit ("Axis" technology object as of V2.0)				
The unit of measurement for the parameter selected in the configuration: • 1013 = "mm" • 1010 =: "m" • 1019 = "in" • 1018 = "ft" • 1005 = "°" (degrees) • -1 = "Pulse"				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Int	1013	R	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

Config.DriveInterface tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.Config.DriveInterface.EnableOutput				
Tags cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
-	-	-	-	-

<Axis name>.Config.DriveInterface.ReadyInput				
Tags cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
-	-	-	-	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

Config.Mechanics tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.Config.Mechanics.PulsesPerDriveRevolution				
Increments per motor revolution				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	L#1000	R	-	X

<Axis name>.Config.Mechanics.LeadScrew				
Distance per revolution (specified in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	1.0E+001	R	-	X

<Axis name>.Config.Mechanics.InverseDirection				
Invert direction				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

Config.DynamicLimits tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.Config.DynamicLimits.MinVelocity				
Start/stop velocity of axis (specified in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	1.0E+001	R	-	X

<Axis name>.Config.DynamicLimits.MaxVelocity				
Maximum velocity of axis (specified in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	2.5E+002	R	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

Config.DynamicDefaults tags V1...3 (S7-1200)

Legend

Data type	Data type of tag
Start value	Start value of tag The start value can be overwritten by the axis configuration.

Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
	1	When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE -> TRUE), disabled, or enabled
	2	When axis is enabled
	5	The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable or active MC_Home command is started (Mode = 3).
	6	When a MC_MoveJog command is stopped
HMI	The tag can be used in an HMI system.	

<Axis name>.Config.DynamicDefaults.Acceleration					
Acceleration of axis (specified in the configured dimension unit)					
Data type	**Start value**	**Access**	**Effective**		**HMI**
Real	4.8E+001	RW	5	CPU Firmware V1.0	X
			1, 5, 6	CPU firmware as of V2.0	

<Axis name>.Config.DynamicDefaults.Deceleration					
Deceleration of axis (specified in the configured dimension unit)					
Data type	**Start value**	**Access**	**Effective**		**HMI**
Real	4.8E+001	RW	5, 6	CPU Firmware V1.0	X
			1, 5, 6	CPU firmware as of V2.0	

<Axis name>.Config.DynamicDefaults.EmergencyDeceleration					
Emergency stop deceleration of axis (specified in the configured dimension unit)					
Data type	**Start value**	**Access**	**Effective**		**HMI**
Real	1.2E+002	RW	2, 5, 6	CPU Firmware V1.0	X
			1, 5, 6	CPU firmware as of V2.0	

<Axis name>.Config.DynamicDefaults.JerkActive ("Axis" technology object as of V2.0)				
TRUE = the jerk limit is activated				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 5	X

<Axis name>.Config.DynamicDefaults.Jerk ("Axis" technology object as of V2.0)				
Jerk during axis acceleration and deceleration ramp (specified in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	1.92E+002	RW	1, 5	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

Config.PositionLimits_SW tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
	1	When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE -> TRUE), disabled, or enabled
	4	Upon the next start of a Motion Control command after a standstill of the axis. The axis standstill can be checked with tag <Axis name>. StatusBits.Standstill.
	5	The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable or active MC_Home command is started (Mode = 3).
HMI	The tag can be used in an HMI system.	

<Axis name>.Config.PositionLimits_SW.Active					
TRUE = The software limit switches are activated					
Data type	**Start value**	**Access**	**Effective**		**HMI**
Bool	FALSE	RW	4	CPU Firmware V1.0	X
			1, 5, 6	CPU firmware as of V2.0	

<Axis name>.Config.PositionLimits_SW.MinPosition				
Position of lower software limit switch (specified in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**

<Axis name>.Config.PositionLimits_SW.MinPosition					
Real	-1.0E+004	RW	4	CPU Firmware V1.0	X
			1, 5, 6	CPU firmware as of V2.0	

<Axis name>.Config.PositionLimits_SW.MaxPosition					
Position of upper software limit switch (specified in the configured unit of measurement)					
Data type	**Start value**	**Access**	**Effective**		**HMI**
Real	1.0E+004	RW	4	CPU Firmware V1.0	X
			1, 5, 6	CPU firmware as of V2.0	

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

Config.PositionLimits_HW tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
	1	When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE -> TRUE), disabled, or enabled
	3	After axis enable (the axis must have previously been at a standstill). The axis standstill can be checked with tag <Axis name>. StatusBits.Standstill.
	4	Upon the next start of a Motion Control command after a standstill of the axis. The axis standstill can be checked with tag <Axis name>. StatusBits.Standstill.
	5	The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable or active MC_Home command is started (Mode = 3).
HMI	The tag can be used in an HMI system.	

<Axis name>.Config.PositionLimits_HW.Active				
TRUE = The hardware limit switches are active.				
Data type	**Start value**	**Access**	**Effective**	**HMI**

<Axis name>.Config.PositionLimits_HW.Active					
Bool	FALSE	RW	3, 4	CPU Firmware V1.0	X
			1, 5, 6	CPU firmware as of V2.0	

<Axis name>.Config.PositionLimits_HW.MinSwitchedLevel				
TRUE = 24 V at CPU input corresponds to lower hardware limit switch approached FALSE = 0 V at CPU input corresponds to lower hardware limit switch approached				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.Config.PositionLimits_HW.MinFallingEvent				
Tag cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	DW#16#00000000	-	-	-

<Axis name>.Config.PositionLimits_HW.MinRisingEvent				
Tag cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	DW#16#00000000	-	-	-

<Axis name>.Config.PositionLimits_HW.MaxSwitchedLevel				
TRUE = 24 V at CPU input corresponds to upper hardware limit switch approached FALSE = 0 V at CPU input corresponds to upper hardware limit switch approached				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.Config.PositionLimits_HW.MaxFallingEvent				
Tag cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	DW#16#00000000	-	-	-

<Axis name>.Config.PositionLimits_HW.MaxRisingEvent				
Tag cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	DW#16#00000000	-	-	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)
Tags of the positioning axis technology object V4...5 (Page 6436)

Config.Homing tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
	1	When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE -> TRUE), disabled, or enabled
	7	When a passive homing command is started
	8	When an active homing command is started
HMI	The tag can be used in an HMI system.	

<Axis name>.Config.Homing.AutoReversal

TRUE = Direction reversal at hardware limit switch enabled (active homing)
FALSE = Direction reversal at hardware limit switch disabled (active homing)

Data type	Start value	Access	Effective		HMI
Bool	TRUE	R	-	Technology object "Axis" V1.0	X
		RW	1, 8	Technology object "Axis" V2.0	

<Axis name>.Config.Homing.Direction

TRUE = Positive approach direction to search for reference point switch and positive homing direction (active homing)
FALSE = Negative approach direction to search for reference point switch and positive homing direction (active homing)

Data type	Start value	Access	Effective		HMI
Bool	TRUE	R	-	Technology object "Axis" V1.0	X
		RW	1, 8	Technology object "Axis" V2.0	

<Axis name>.Config.Homing.SideActiveHoming ("Axis" technology object as of V2.0)				
TRUE = Homing on high end of the reference point switch (active homing) TRUE = Homing on lower end of the reference point switch (active homing)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	RW	1, 8	X

<Axis name>.Config.Homing.SidePassiveHoming ("Axis" technology object as of V2.0)				
TRUE = Homing on high end of the reference point switch (passive homing) TRUE = Homing on lower end of the reference point switch (passive homing)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	RW	1, 7	X

<Axis name>.Config.Homing.RisingEdge (as of technology object "Axis" V1.0)				
TRUE = Homing with negative signal edge of the reference point switch (active homing) FALSE = Homing with positive signal edge of the reference point switch (active homing) For information on the effect of the tag on passive homing, refer to the description in "Configuration - Homing".				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.Config.Homing.Offset					
Home position offset /specified in the configured unit of measurement (active homing)					
Data type	**Start value**	**Access**	**Effective**		**HMI**
Real	0.0	R RW	-	Technology object "Axis" V1.0	X
			1, 8	Technology object "Axis" V2.0	

<Axis name>.Config.Homing.FastVelocity					
Approach velocity / specified in the configured unit of measurement (active homing)					
Data type	**Start value**	**Access**	**Effective**		**HMI**
Real	2.0E+002	R	-	Technology object "Axis" V1.0	X
		RW	1, 8	Technology object "Axis" V2.0	

<Axis name>.Config.Homing.SlowVelocity				
Homing velocity / specified in the configured unit of measurement (active homing)				
Data type	**Start value**	**Access**	**Effective**	**HMI**

<Axis name>.Config.Homing.SlowVelocity					
Real	4.0E+001	R	-	Technology object "Axis" V1.0	X
		RW	1, 8	Technology object "Axis" V2.0	

<Axis name>.Config.Homing.FallingEvent				
Tag cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	DW#16#00000000	-	-	-

<Axis name>.Config.Homing.RisingEvent				
Tag cannot be evaluated in the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	DW#16#00000000	-	-	

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

MotionStatus tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.MotionStatus.Position				
Current position of the axis (specified in the configured unit of measurement) If the axis is not homed, the tag indicates the position value relative to the enable position of the axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	R	-	X

<Axis name>.MotionStatus.Velocity				
Current velocity of the axis (specified in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	R	-	X

<Axis name>.MotionStatus.Distance				
Current distance to the target position of the axis (specified in the configured unit of measurement) The value of the tag is only valid during execution of a positioning command with "MC_MoveAbsolute" or "MC_MoveRelative" or of the axis command table.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	R	-	X

<Axis name>.MotionStatus.TargetPosition				
Target position of axis (specified in the configured unit of measurement) The value of the tag is only valid during execution of a positioning command with "MC_MoveAbsolute" or "MC_MoveRelative" or of the axis command table.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	R	-	X

See also

Motion status (Page 6288)

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

StatusBits tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.StatusBits.Activated				
TRUE = The axis is activated. It is connected to the assigned PTO (Pulse Train Output). The data of the technology data block will be updated cyclically.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.Enable				
TRUE = The axis is enabled and ready to take on Motion Control commands.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.HomingDone				
TRUE = The axis is homed and is capable of executing absolute positioning commands. The axis does not have to be homed for relative positioning. The status is FALSE during active homing. The status remains TRUE during passive homing if the axis has already been homed.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.Done				
TRUE = No Motion Control command is active on the axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.Error				
TRUE = An error occurred in the axis technology object. Detailed information about the error is available in automatic mode in the "ErrorID" and "ErrorInfo" parameters of the Motion Control instructions. In manual mode, the "Error message" box of the axis control panel displays detailed information about the cause of error.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.StandStill				
TRUE = The axis is at a standstill.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.PositioningCommand				
TRUE = The axis is executing a positioning command.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.SpeedCommand				
TRUE = The axis is executing a travel command at predefined velocity.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.Homing				
TRUE = The axis is executing a homing command of the "MC_Home" Motion Control instruction or axis command table.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.CommandTableActive				
TRUE = The axis is controlled by Motion Control instruction "MC_CommandTable".				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.ConstantVelocity				
TRUE = The axis travels at constant velocity.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.Acceleration				
TRUE = The axis accelerates.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.Deceleration				
TRUE = The axis decelerates (slows down).				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.ControlPanelActive				
TRUE = The "Manual control" mode has been enabled in the axis command table. The axis command table has control priority over the "Axis" technology object. The axis cannot be controlled from the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.DriveReady				
TRUE = The drive is ready.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.StatusBits.RestartRequired				
TRUE = Values were modified in the load memory. To download the values in the CPU RUN operating mode to the work memory, you need to restart the axis. Use the Motion Control instruction MC_Reset to do this.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

See also

Status and error bits (technology objects as of V4) (Page 6285)

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

ErrorBits tags V1...3 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.ErrorBits.SystemFault				
TRUE = Internal system error.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.ConfigFault				
TRUE = Incorrect configuration of axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.DriveFault				
TRUE = The drive has reported an error due to loss of its "Drive ready" signal.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.SwLimitMinReached				
TRUE = The lower software limit switch has been reached.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.SwLimitMinExceeded				
TRUE = The lower software limit switch has been exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.SwLimitMaxReached				
TRUE = The upper software limit switch has been reached.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.SwLimitMaxExceeded				
TRUE = The upper software limit switch has been exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.HwLimitMin				
TRUE = The lower hardware limit switch has been approached.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.HwLimitMax				
TRUE = The upper hardware limit switch has been approached.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.ErrorBits.HwUsed				
TRUE = A second axis is using the same PTO (Pulse Train Output) and is enabled with "MC_Power".				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

See also

Status and error bits (technology objects as of V4) (Page 6285)

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

Internal tags V1...3 (S7-1200)

The "Internal" tags contain no user-relevant data; these tags cannot be accessed in the user program.

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

ControlPanel tags V1...3 (S7-1200)

The "ControlPanel" tags contain no user-relevant data; these tags cannot be accessed in the user program.

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tags of the positioning axis technology object V4...5 (Page 6436)

Update of the technology object tags (S7-1200)

The status and error information of the axis indicated in the technology object tags is updated at each cycle control point.

The change in values of editable configuration tags does not take effect immediately. For information on the conditions under which a change takes effect, refer to the detailed description of the relevant tag.

Tags of the positioning axis technology object V4...5 (S7-1200)

Position tag V4...5 (S7-1200)

Legend

Data type	Data type of tag
Start value	Start value of tag

Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Position				
Setpoint position of the axis (indicated in the configured unit of measurement) If the axis is not homed, the tag indicates the position value relative to the enable position of the axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Motion status (Page 6288)

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Velocity tag V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Velocity				
Setpoint velocity of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Motion status (Page 6288)

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

ActualPosition tag V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.ActualPosition				
Actual position of the axis (indicated in the configured unit of measurement) If the axis is not homed, the tag indicates the position value relative to the enable position of the axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

ActualVelocity tag V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.ActualVelocity				
Actual velocity of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Actuator tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect:	
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Actor.Type				
Drive connection • Positioning axis technology object as of V5: 0 = The drive is connected via an analog output. All movements of the axis are position-controlled. 1 = The drive is connected via PROFIdrive telegrams. All movements of the axis are position-controlled. 2 = The drive is connected via a pulse interface. • Positioning axis technology object V4: The drive is connected via a pulse interface.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	2	R	-	X

<Axis name>.Actor.InverseDirection				
Invert direction FALSE = The direction is not inverted. TRUE = The direction is inverted.				
Data type	**Start value**	**Access**	**Effective**	**HMI**

<Axis name>.Actor.InverseDirection				
Bool	FALSE	R	-	X
		WP	2	-

<Axis name>.Actor.DirectionMode				
Permitted direction of rotation 0 = Both directions 1 = Positive direction 2 = Negative direction				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Int	0	R	-	X
		WP	2	-

<AxisName>.Actor.Interface.AddressIn (Positioning axis technology object as of V5)				
Input address for the PROFIdrive telegram (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<AxisName>.Actor.Interface.AddressOut (Positioning axis technology object as of V5)				
Output address for the PROFIdrive telegram (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Actor.Interface.EnableDriveOutput				
Enable output (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Actor.Interface.DriveReadyInput				
Ready input (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Actor.Interface.PTO				
Pulse output (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	0	-	-	-

<AxisName>.Actor.DriveParameter.ReferenceSpeed (Positioning axis technology object as of V5)				
Reference value (100%) for the setpoint speed of the drive (N-set) The setpoint speed is transmitted in the PROFIdrive telegram as a standardized value from -200% to 200% of the "ReferenceSpeed". For setpoint specification via an analog output, the analog output can be operated in the range from -117% to 117%, provided the drive permits this.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	3000.0	R	-	X
		WP	2	-

<AxisName>.Actor.DriveParameter.MaxSpeed (Positioning axis technology object as of V5)				
Maximum value for the setpoint speed of the drive (N-set) (PROFIdrive: MaxSpeed ≤ 2 × ReferenceSpeed Analog setpoint: MaxSpeed ≤ 1.17 × ReferenceSpeed)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	3000.0	R	-	X
		WP	2	-

<Axis name>.Actor.DriveParameter.PulsesPerDriveRevolution				
Pulses per motor revolution				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	1000.0	R	-	X
		WP	2	-

See also

Tag of the axis technology object V1...3 (Page 6420)

Tags of the positioning axis technology object as of V6 (Page 6328)

Sensor[1] tags (S7-1200)

Sensor[1] tags V4...5 (S7-1200)

Legend

Data type	Data type of tag
Start value	Start value of tag The start value can be overwritten by the axis configuration.

Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect:	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Sensor[1].Type				
Encoder type (internal parameter) 0 = incremental 1 = Absolute				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	0	-	-	-

<Axis name>.Sensor[1].InverseDirection				
Inversion of the actual value FALSE: Actual value is not inverted TRUE: Actual value is inverted				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.Sensor[1].System				
Encoder system 0 = Linear encoder 1 = Rotary encoder				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	1	R	-	X

<Axis name>.Sensor[1].MountingMode				
Type of encoder mounting 0 = Drive end 2 = External				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	0	R	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1].Interface tags V4...5 (S7-1200)

Legend

Data type	Data type of tag
Start value	Start value of tag The start value can be overwritten by the axis configuration.
Access	Access to the tag in the user program:
	- The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).

<Axis name>.Sensor[1].Interface.Type				
Encoder connection (internal parameter) 0 = PROFIdrive encoder on PROFINET 1 = Encoder on technology module (TM) 2 = Encoder on drive 4 = Encoder on high-speed counter				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	4	-	-	-

<Axis name>.Sensor[1].Interface.AddressIn				
Input address for the PROFIdrive telegram (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Sensor[1].Interface.AddressOut				
Output address for the PROFIdrive telegram (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Sensor[1].Interface.HSC				
High-speed counter to which the encoder transfers the actual value (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DWord	-	-	-	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1].Parameter tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Sensor[1].Parameter.Resolution				
Resolution of a linear encoder (offset between two encoder pulses)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.001	R	-	X

<Axis name>.Sensor[1].Parameter.StepsPerRevolution				
Increments per rotary encoder revolution				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	2048	R	-	X

<Axis name>.Sensor[1].Parameter.FineResolutionXist1				
Number of bits for fine resolution Gn_XIST1 (cyclic actual encoder value)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	11	R	-	X

<Axis name>.Sensor[1].Parameter.FineResolutionXist2				
Number of bits for fine resolution Gn_XIST2 (absolute value of the encoder)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	9	R	-	X

<Axis name>.Sensor[1].Parameter.DeterminableRevolutions				
Number of differentiable encoder revolutions for a multi-turn absolute value encoder (For a single-turn absolute value encoder = 1; for an incremental encoder = 0)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
UDInt	1	R	-	X

<Axis name>.Sensor[1].Parameter.DistancePerRevolution				
Load distance per revolution of an externally mounted encoder				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	100.0	R	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1].ActiveHoming tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program. The tag can be written with Motion Control instruction "MC_WriteParam".
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE → TRUE), disabled, or enabled
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
	8	For drive connection via PTO (Pulse Train Output): When an active homing command is started
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Sensor[1].ActiveHoming.Mode				
Active homing mode • Positioning axis technology object as of V5: 0 = Zero mark via PROFIdrive telegram (not PTO) 1 = Zero mark via PROFIdrive telegram and proximity switch (not PTO) 2 = Homing via digital input • Positioning axis technology object V4: 2 = Homing via digital input				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	2	R	-	X
		WP	2	-

<Axis name>.Sensor[1].ActiveHoming.SideInput				
End of the homing switch to which the axis is homed during active homing FALSE = Lower end TRUE = Upper end				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 8, 10	X

<Axis name>.Sensor[1].ActiveHoming.DigitalInputAddress				
Symbolic input address of the homing switch (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Sensor[1].ActiveHoming.HomePositionOffset				
Home position offset (active homing) (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RW	1, 8, 10	X

<Axis name>.Sensor[1].ActiveHoming.SwitchLevel				
Selection of signal level that is present at the CPU input when the homing switch is reached FALSE = Low level (Low-active) TRUE = High level (High-active)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	RW	1, 8, 10	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Sensor[1].PassiveHoming tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program. The tag can be written with Motion Control instruction "MC_WriteParam".
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE → TRUE), disabled, or enabled
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
	7	For drive connection via PTO (Pulse Train Output): When a passive homing command is started
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Sensor[1].PassiveHoming.Mode

Passive homing mode

- Positioning axis technology object as of V5:
  0 = Zero mark via PROFIdrive telegram (not PTO)
  1 = Zero mark via PROFIdrive telegram and proximity switch (not PTO)
  2 = Homing via digital input
- Positioning axis technology object V4:
  2 = Homing via digital input

Data type	Start value	Access	Effective	HMI
DInt	2	R	-	X
		WP	2	-

<Axis name>.Sensor[1].PassiveHoming.SideInput

End of the homing switch to which the axis is homed during passive homing

FALSE = Lower end

TRUE = Upper end

Data type	Start value	Access	Effective	HMI
Bool	FALSE	RW	1, 7, 10	X

<Axis name>.Sensor[1].PassiveHoming.DigitalInputAddress				
Symbolic input address of the homing switch (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.Sensor[1].PassiveHoming.SwitchLevel				
Selection of level that is present at the CPU input when the homing switch is reached FALSE = Low level (Low-active) TRUE = High level (High-active)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	RW	1, 7, 10	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Units tag V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Units.LengthUnit				
Configured unit of measurement of the parameter -1 = "Pulse" 1005 = "°" (degrees) 1013 = "mm" 1010 =: "m" 1018 = "ft" 1019 = "in"				
Data type	**Start value**	**Access**	**Effective**	**HMI**

<Axis name>.Units.LengthUnit				
Int	1013	R	-	X
		WP	2	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Mechanics tag V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Mechanics.LeadScrew				
Distance per revolution (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Modulo tags V4...5 (S7-1200)

Legend

Data type	Data type of tag
Start value	Start value of tag The start value can be overwritten by the axis configuration.

Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Modulo.Enable				
Enable modulo FALSE = Modulo conversion deactivated TRUE = Modulo conversion activated When modulo conversion is enabled, a check is made for modulo length > 0.0				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X

<Axis name>.Modulo.Length				
Modulo length				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	360.0	R	-	X

<Axis name>.Modulo.StartValue				
Modulo start value				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	R	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

DynamicLimits tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".

Effective	Specifies when a change to the tag takes effect:	
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.DynamicLimits.MaxVelocity				
Maximum velocity of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	250.0	R	-	X
		WP	2	-

<Axis name>.DynamicLimits.MinVelocity				
Start/stop velocity of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X
		WP	2	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

DynamicDefaults tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program. The tag can be written with Motion Control instruction "MC_WriteParam".

Effective	Specifies when a change to the tag takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE → TRUE), disabled, or enabled
	5	For drive connection via PTO (Pulse Train Output): The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable, or active MC_Home command is started (Mode = 3)
	6	For drive connection via PTO (Pulse Train Output): When a MC_MoveJog command is stopped
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.DynamicDefaults.Acceleration				
Acceleration of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	48.0	RW	1, 5, 6, 10	X

<Axis name>.DynamicDefaults.Deceleration				
Deceleration of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	48.0	RW	1, 5, 6, 10	X

<Axis name>.DynamicDefaults.EmergencyDeceleration				
Emergency stop deceleration of the axis (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	120.0	RW	1, 5, 6, 10	X

<Axis name>.DynamicDefaults.Jerk				
Jerk during acceleration and deceleration ramp of the axis (indicated in the configured unit of measurement) The jerk is activated if the configured jerk is greater than 0.00004 mm/s².				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	192.0	RW	1, 5, 10	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

PositionLimitsSW tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program. The tag can be written with Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE → TRUE), disabled, or enabled
	5	For drive connection via PTO (Pulse Train Output): The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable, or active MC_Home command is started (Mode = 3)
	6	For drive connection via PTO (Pulse Train Output): When a MC_MoveJog command is stopped
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.PositionLimitsSW.Active				
Activation of the software limit switches FALSE = The software limit switches are deactivated. TRUE = The software limit switches are activated.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 5, 6. 10	X

<Axis name>.PositionLimitsSW.MinPosition				
Position of the low software limit switch (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	-10000.0	RW	1, 5, 6, 10	X

<Axis name>.PositionLimitsSW.MaxPosition				
Position of the high software limit switch (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10000.0	RW	1, 5, 6, 10	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

PositionLimitsHW tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program. The tag can be written with Motion Control instruction "MC_WriteParam".
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE → TRUE), disabled, or enabled
	2	For drive connection via PTO (Pulse Train Output): When axis is enabled
	5	For drive connection via PTO (Pulse Train Output): The next time an MC_MoveAbsolute, MC_MoveRelative, MC_MoveVelocity, MC_MoveJog, MC_Halt, MC_CommandTable, or active MC_Home command is started (Mode = 3)
	6	For drive connection via PTO (Pulse Train Output): When a MC_MoveJog command is stopped
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.PositionLimitsHW.Active				
Activation of the hardware limit switches FALSE = The hardware limit switches are deactivated. TRUE = The hardware limit switches are activated.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 5, 6, 10	X

<Axis name>.PositionLimitsHW.MinSwitchLevel				
Selection of signal level that is present at the CPU input when the low hardware limit switch is reached FALSE = Low level (Low-active) TRUE = High level (High-active)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X
		WP	2	-

<Axis name>.PositionLimitsHW.MinSwitchAddress				
Symbolic input address of the low hardware limit switch (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

<Axis name>.PositionLimitsHW.MaxSwitchLevel				
Selection of signal level that is present at the CPU input when the high hardware limit switch is reached FALSE = Low level (Low-active) TRUE = High level (High-active)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	R	-	X
		WP	2	-

<Axis name>.PositionLimitsHW.MaxSwitchAddress				
Input address of the high hardware limit switch (internal parameter)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
VREF	-	-	-	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Homing tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program. The tag can be written with Motion Control instruction "MC_WriteParam".

Effective	Specifies when a change to the tag takes effect:	
	1	For drive connection via PTO (Pulse Train Output): When axis is activated (<Axis name>.StatusBits.Activated tag changes from FALSE → TRUE), disabled, or enabled
	8	For drive connection via PTO (Pulse Train Output): When an active homing command is started
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.Homing.AutoReversal				
Activation of auto reverse at the hardware limit switch during active homing FALSE = Auto reversal at the hardware limit switch is deactivated. TRUE = Auto reversal at the hardware limit switch is activated.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RW	1, 8, 10	X

<Axis name>.Homing.ApproachDirection				
Approach and homing direction of axis during active homing FALSE = Negative approach direction for finding the homing switch and negative homing direction TRUE = Positive approach direction for finding the homing switch and positive homing direction				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	RW	1, 8, 10	X

<Axis name>.Homing.ApproachVelocity				
Approach velocity of the axis during active homing (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	200.0	RW	1, 8, 10	X

<Axis name>.Homing.ReferencingVelocity				
Homing velocity of the axis during active homing (indicated in the configured unit of measurement)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	40.0	RW	1, 8, 10	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

PositionControl tags V5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.PositionControl.Kv				
Proportional gain in the position control ("Kv" > 0.0)				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X
		WP	10	-

<Axis name>.PositionControl.Kpc				
Velocity precontrol of the position control as a percentage				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	100.0	R	-	X
		WP	10	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

FollowingError tags V5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".

Effective	Specifies when a change to the tag takes effect:	
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.FollowingError.EnableMonitoring				
Enable following error monitoring FALSE = Following error monitoring disabled TRUE = Following error monitoring enabled				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	TRUE	R	-	X

<Axis name>.FollowingError.MinValue				
Permissible following error at velocities below the value of "MinVelocity".				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X
		WP	10	-

<Axis name>.FollowingError.MaxValue				
Maximum permissible following error which may be reached at the maximum velocity.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	100.0	R	-	X
		WP	10	-

<Axis name>.FollowingError.MinVelocity				
MinValue" is permissible below this velocity and is held constant.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	10.0	R	-	X
		WP	10	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)
Tag of the axis technology object V1...3 (Page 6420)

PositionMonitoring tags V5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.PositioningMonitoring.ToleranceTime				
Tolerance time Maximum permitted duration from reaching velocity setpoint zero until entrance into the positioning window.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	1.0	R	-	X
		WP	10	-

<Axis name>.PositioningMonitoring.MinDwellTime				
Minimum dwell time in positioning window				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.1	R	-	X
		WP	10	-

<Axis name>.PositioningMonitoring.Window				
Positioning window				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	1.0	R	-	X
		WP	10	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

StandstillSignal tags V5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	R	The tag can be read in the user program.
	WP	If the axis is disabled (MC_Power.Status = FALSE), the tag can be written with the Motion Control instruction "MC_WriteParam".
Effective	Specifies when a change to the tag takes effect:	
	10	For drive connection via PROFIdrive/analog output: With the next call of the MC-Servo [OB91]
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.StandstillSignal.VelocityThreshold				
Velocity threshold If velocity is below this threshold, then the minimum dwell time begins.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	5.0	R	-	X
		WP	10	-

<Axis name>.StandstillSignal..MinDwellTime				
Minimum dwell time				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.01	R	-	X
		WP	10	-

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

StatusPositioning tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.

Effective	Specifies when a change to the tag takes effect.
HMI	The tag can be used in an HMI system.

<Axis name>.StatusPositioning.Distance				
Current distance of axis from target position (indicated in the configured unit of measurement) The value of the tag is only valid during execution of a positioning command with "MC_MoveAbsolute", "MC_MoveRelative", or the axis control panel.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

<Axis name>.StatusPositioning.TargetPosition				
Target position of the axis (indicated in the configured unit of measurement) The value of the tag is only valid during execution of a positioning command with "MC_MoveAbsolute", "MC_MoveRelative", or the axis control panel.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

<AxisName>.StatusPositioning.FollowingError (Positioning axis technology object as of V5)				
Current following error of the axis (indicated in the configured unit of measurement) FollowingError = 0.0 for drive connection via PTO (Pulse Train Output).				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Motion status (Page 6288)

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

StatusDrive tags V5 (S7-1200)

Legend

Data type	Data type of tag
Start value	Start value of tag The start value can be overwritten by the axis configuration.

Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect:	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.StatusDrive.InOperation				
Operational status of the drive FALSE = Drive not ready. Setpoints will not be executed. TRUE = Drive ready. Setpoints can be executed.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusDrive.CommunicationOK				
Cyclic BUS communication between controller and drive FALSE = Communication not established TRUE = Communication established				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

StatusSensor tags V5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the axis configuration.	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect:	
HMI	The tag can be used in an HMI system (according to the access to the tag in the user program).	

<Axis name>.StatusSensor.State				
Status of the encoder value 0 = Invalid 1 = Wait for status valid 2 = Valid				
Data type	**Start value**	**Access**	**Effective**	**HMI**
DInt	0	RCCP, RP	-	X

<Axis name>.StatusSensor.CommunicationOK				
Cyclic BUS communication between controller and encoder FALSE = Communication not established TRUE = Communication established				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusSensor.AbsEncoderOffset				
Home point offset to the value of an absolute value encoder. The value will be retentively stored in the CPU.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RCCP, RP	-	X

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

StatusBits tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.StatusBits.Activated				
Activation of the axis FALSE = The axis is not activated. TRUE = The axis is activated. It is connected to the assigned PTO (Pulse Train Output). The data of the technology data block will be updated cyclically.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Enable				
Enable status of the axis FALSE = The axis is not enabled. TRUE = The axis is enabled and ready to accept Motion Control commands.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.HomingDone				
Homing status of the axis FALSE = The axis is not homed. TRUE = The axis is homed and is capable of executing absolute positioning commands. The axis does not have to be homed for relative positioning. The status is FALSE during active homing. The status remains TRUE during passive homing if the axis was already homed beforehand.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Done				
Command execution on the axis FALSE = A Motion Control command is active on the axis. TRUE = A Motion Control command is not active on the axis.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Error				
Error status on the axis FALSE = No error is active on the axis. TRUE = An error occurred on the axis. Additional information about the error is available in automatic control at the "ErrorID" and "ErrorInfo" parameters of the Motion Control instructions. In manual mode, the "Error message" box of the axis control panel displays detailed information about the cause of error.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.StandStill				
Standstill of the axis FALSE = The axis is in motion. TRUE = The axis is at a standstill.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.PositioningCommand				
Execution of a positioning command FALSE = A positioning command is not active on the axis. TRUE = The axis is executing a positioning command of the "MC_MoveRelative" or MC_MoveAbsolute Motion Control instruction.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.VelocityCommand				
Execution of a command with velocity specification FALSE = A command with velocity specification is not active on the axis. TRUE = The axis is executing a motion command with velocity specification of the "MC_MoveVelocity" or MC_MoveJog Motion Control instruction.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.HomingCommand				
Execution of a homing command FALSE = A homing command is not active on the axis. TRUE = The axis is executing a homing command of the "MC_Home" Motion Control instruction.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.CommandTableActive				
Execution of a command table FALSE = A command table is not active on the axis. TRUE = The axis is controlled by "MC_CommandTable" Motion Control instruction.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.ConstantVelocity				
Constant velocity FALSE = The axis is accelerating, decelerating, or at a standstill. TRUE = The setpoint velocity has been reached. The axis is moving at constant velocity.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Accelerating				
Acceleration process FALSE = The axis is decelerating, moving at constant velocity, or at a standstill. TRUE = The axis is accelerating.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.Decelerating				
Deceleration process FALSE = The axis is accelerating, moving at constant velocity, or at a standstill. TRUE = The axis is decelerating.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.ControlPanelActive				
Activation status of the axis control panel FALSE = "Automatic mode" is activated. The user program has control priority over the axis. TRUE = The "Manual control" mode was enabled in the axis control panel. The axis control panel has control priority over the axis. The axis cannot be controlled from the user program.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.DriveReady				
Operational status of the drive FALSE = The drive is not ready. Setpoints will not be executed. TRUE = The drive is ready. Setpoints can be executed.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.RestartRequired				
Restart of axis required FALSE = A restart of the axis is not required. TRUE = Values were modified in the load memory. To download the values to the work memory while the CPU is in RUN mode, the axis must be restarted. Use the MC_Reset Motion Control instruction to do this.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.SWLimitMinActive				
Status of the low software limit switch FALSE = The axis is kept within its configured work area. TRUE = The low software limit switch was reached or exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.SWLimitMaxActive				
Status of the high software limit switch FALSE = The axis is kept within its configured work area. TRUE = The high software limit switch was reached or exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.HWLimitMinActive				
Status of the low hardware limit switch FALSE = The axis is kept within its configured permitted traversing range. TRUE = The low hardware limit switch was reached or exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.StatusBits.HWLimitMaxActive				
Status of the high hardware limit switch FALSE = The axis is kept within its configured permitted traversing range. TRUE = The high hardware limit switch was reached or exceeded.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

See also

Status and error bits (technology objects as of V4) (Page 6285)
Tags of the positioning axis technology object as of V6 (Page 6328)
Tag of the axis technology object V1...3 (Page 6420)

ErrorBits tags V4...5 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag	
Access	Access to the tag in the user program:	
	RCCP	The tag can be read in the user program and is updated at each cycle control point.
	RP	The tag can be read with the Motion Control instruction "MC_ReadParam". The current value of the corresponding tags is determined at the start of the command.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Axis name>.ErrorBits.SystemFault				
Internal system error				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.ConfigFault				
Incorrect configuration of the axis				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.DriveFault				
Error in the drive. Loss of the "Drive ready" signal.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.SWLimit				
Software limit switch reached or exceeded				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.HWLimit				
Hardware limit switch reached or exceeded				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP	-	X

<Axis name>.ErrorBits.DirectionFault				
Impermissible motion direction				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP, RP	-	X

<Axis name>.ErrorBits.HWUsed				
Another axis is using the same PTO (Pulse Train Output) and is enabled with "MC_Power".				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<AxisName>.ErrorBits.SensorFault (Positioning axis technology object as of V5)				
Error in the encoder system				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.CommunicationFault (Positioning axis technology object as of V5)				
Communication error Error in communication with a connected device.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.FollowingErrorFault (Positioning axis technology object as of V5)				
Maximum permitted following error exceeded				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

<Axis name>.ErrorBits.PositioningFault (Positioning axis technology object as of V5)				
Positioning error The axis was not correctly positioned at the end of a positioning motion.				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Bool	FALSE	RCCP, RP	-	X

See also

Status and error bits (technology objects as of V4) (Page 6285)

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

ControlPanel tags V4...5 (S7-1200)

The "ControlPanel" tags do not contain any user-relevant data. These tags cannot be accessed in the user program.

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Internal tags V4...5 (S7-1200)

The "Internal" tags do not contain any user-relevant data. These tags cannot be accessed in the user program.

See also

Tags of the positioning axis technology object as of V6 (Page 6328)

Tag of the axis technology object V1...3 (Page 6420)

Update of the technology object tags (S7-1200)

The status and error information of the axis indicated in the technology object tags is updated at each cycle control point.

Changes to the values of configuration tags do not take effect immediately. For information on the conditions under which a change takes effect, refer to the detailed description of the relevant tag.

Tags of the command table technology object V1...3 (S7-1200)

Config.Command.Command[1 ... 32] tags V1...3 (S7-1200)

Legend

Data type	Data type of the tag
Start value	Start value of tag The start value can be overwritten by the configuration of the command table.

Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change in the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Command table>.Config.Command.Command[x].Type				
Command type • 0 = "Empty" command • 2 = "Hold" command • 5 = "Relative positioning" command • 6 = "Absolute positioning" command • 7 = "Velocity setpoint" command • 151 = "Wait" command				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Int	0	RW	-	X

<Command table>. Config.Command.Command[x].Position				
Command target position / travel path				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real		RW	-	X

<Command table>. Config.Command.Command[x].Velocity				
Command velocity				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RW	-	X

<Command table>. Config.Command.Command[x].Duration				
Command duration				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RW	-	X

<Command table>. Config.Command.Command[x].BufferMode				
Value for command "Next step" • 0 = "Complete command" • 1 = "Blend movement"				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Int	0	RW	-	X

<Command table>. Config.Command.Command[x].StepCode				
Command step code				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Word	0	RW	-	X

See also

Tag of the command table technology object V4...5 (Page 6472)

Tag of the command table technology object V4...5 (S7-1200)

Command[1...32] tags as of V4 (S7-1200)

Legend

Data type	Data type of tag	
Start value	Start value of tag The start value can be overwritten by the configuration of the command table.	
Access	Access to the tag in the user program:	
	RW	The tag can be read and written in the user program.
	R	The tag can be read in the user program.
	-	The tag cannot be used in the user program.
Effective	Specifies when a change to the tag takes effect.	
HMI	The tag can be used in an HMI system.	

<Command table>.Command[n].Type				
Command type • 0 = "Empty" command • 2 = "Hold" command • 5 = "Relative positioning" command • 6 = "Absolute positioning" command • 7 = "Velocity setpoint" command • 151 = "Wait" command				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Int	0	RW	-	X

<Command table>.Command[n].Position				
Target position/traversing distance of the command				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RW	-	X

<Axis name>.Command[n].Velocity				
Command velocity				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RW	-	X

<Command table>.Command[n].Duration				
Command duration				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Real	0.0	RW	-	X

<Command table>.Command[n].NextStep				
Mode for the transition to the next command • 0 = "Complete command" • 1 = "Blend movement"				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Int	0	RW	-	X

<Command table>.Command[n].StepCode				
Command step code				
Data type	**Start value**	**Access**	**Effective**	**HMI**
Word	0	RW	-	X

See also

Tags of the command table technology object V1...3 (Page 6470)

Using online and diagnostics functions 14

14.1 Displaying accessible devices

Accessible devices are all devices connected to an interface of the programming device / PC that are turned on. Devices that allow only restricted configuration using the currently installed products or that cannot be configured at all can also be displayed.

To speed up the display in large networks with many network devices, the PROFINET device names are displayed in a slightly adapted form for PROFINET devices. Unicode-specific characters in the PROFINET device name are converted into ASCII-compatible characters using the Punycode method (according to RFC 3492).

Displaying accessible devices in the project tree

To display accessible devices at an interface of the programming device / PC, follow these steps:

1. Open the "Online access" folder in the project tree.
2. Click on the arrow to the left of the interface to show all the objects arranged below the interface.
3. Double-click "Update accessible devices" below the interface.
   You can see the progress of the search in the status bar. If you have found the desired device before the search is completed, you can cancel the search. To do this, click on the cross to the right of the progress bar.
   The status of the search is indicated with an symbol at the interface.

	The interface is available. The search is performed.
	The interface is available. The search is cancelled.
	The interface is available. The search is completed.

   The network stations that have been found through the search are displayed below the interface.
4. Optional: Select the interface in the project tree. Then open the overview window in the detail view to display more details for the respective devices. More details include, for example, the device type, the MAC address or the configured device name.
   The detailed display of the devices in the overview window can take some time in networks with many devices because each individual device must submit the details.

Displaying accessible devices in a table

To display accessible devices in an easy to read table outside the project tree, follow these steps:

1. Select the "Accessible devices" command in the "Online" menu.
   The "Accessible devices" dialog is displayed.
2. Select the type of interface from the "Type of the PG/PC interface" drop-down list.
3. Select the required interface of the programming device / PC from the "PG/PC interface" drop-down list.
4. Click the "Start Search" button.
   The accessible devices at the selected interface of the programming device/PC are displayed in the table. The connection line between PG/PC and the network is closed and displayed in green in the graphic. If no devices are accessible at an interface, the connection line is displayed as a dashed line.
5. To go to a device in the project tree, select the device from the list of accessible devices and click the "Show" button.
   The selected device is highlighted in the project tree.

Displaying additional information about the devices in the project tree

To display additional information on the individual accessible devices in the project tree, follow these steps:

1. Click on the arrow to the left of one of the accessible devices in the project tree.
   All data available online, for example blocks and system data, is displayed for known devices. If additional editing options are available for a device with the shortcut menu, the device is shown in black text. You have only read access to the objects with a gray background, such as blocks.

See also

Changing the device configuration online (Page 6477)

Default setting online connection data (Page 6481)

Overview window (Page 262)

14.2 Changing the device configuration online

You can assign parameters to some devices, preferably in small hardware setups, directly online. You do not need to create a project or have offline data to do this. In this way, you can quickly and easily change the device configuration. You do not need to compile the hardware configuration or perform a download. Depending on the device, either all changes become active immediately or they are written to the device only after confirmation.

Requirement

- The device must support online parameter assignment. You can learn whether or not your specific devices support this function in the device manual.
- The device must be connected to the PG/PC and available in the list of accessible devices.

Procedure

To change the device configuration online, follow these steps:

1. Show the accessible devices on the interface via which the device is connected. To learn how to show the accessible devices, see the chapter "Showing accessible devices (Page 6475)".
2. Expand the device to display the lower-level elements.
3. Double-click the "Parametrize device" item.
   A configuration page for the device opens in the work area.
4. Make all required settings.
   With some devices, the new settings take effect immediately.
5. Optionally, depending on the device: Click on the "Upload to device" button.
   The settings are transferred to the device.

14.3 Connecting devices online

14.3.1 General information about online mode

Online mode

In online mode, there is an online connection between your programming device / PC and one or more devices.

An online connection between the programming device/PC and the device is required, for example, for the following tasks:

- Testing user programs
- Displaying and changing the operating mode of the CPU
- Displaying and setting the date and time of day of the CPU
- Displaying module information
- Comparing blocks
- Hardware diagnostics

Before you can establish an online connection, the programming device/PC and the device must be physically or remotely connected. As an alternative, some devices support a simulation mode. In this case, a connection to the device is simulated via the PLCSIM virtual interface.

After establishing a connection, you can use the Online and Diagnostics view or the "Online tools" task card to access the data on the device. The current online status of a device is indicated by an icon to the right of the device in the project tree. You will find the meaning of the individual status icons in the relevant tooltip.

Note

Some online functions depend on the scope of the installed software or whether a project is open.

Standby or hibernation of the programming device / PC

If the programming device / PC is changed to the standby or hibernation mode when there is an online connection, all online connections are terminated. When the programming device / PC wakes up from hibernation, the online connections are not automatically re-established.

Note that suddenly terminating an online connection can lead to loss of data or a connected device may interrupt program execution.

Performing an LED flash test

In many online dialogs you can perform an LED flash test, if the device connected online supports this feature. If you select the "Flash LED" check box, an LED flashes on the currently selected device. This feature is useful, for example, when you are not sure which device in the hardware configuration corresponds to the station currently selected in the software.

Read any additional information and learn about the possible limitations to the LED flash test in the respective device documentation.

See also

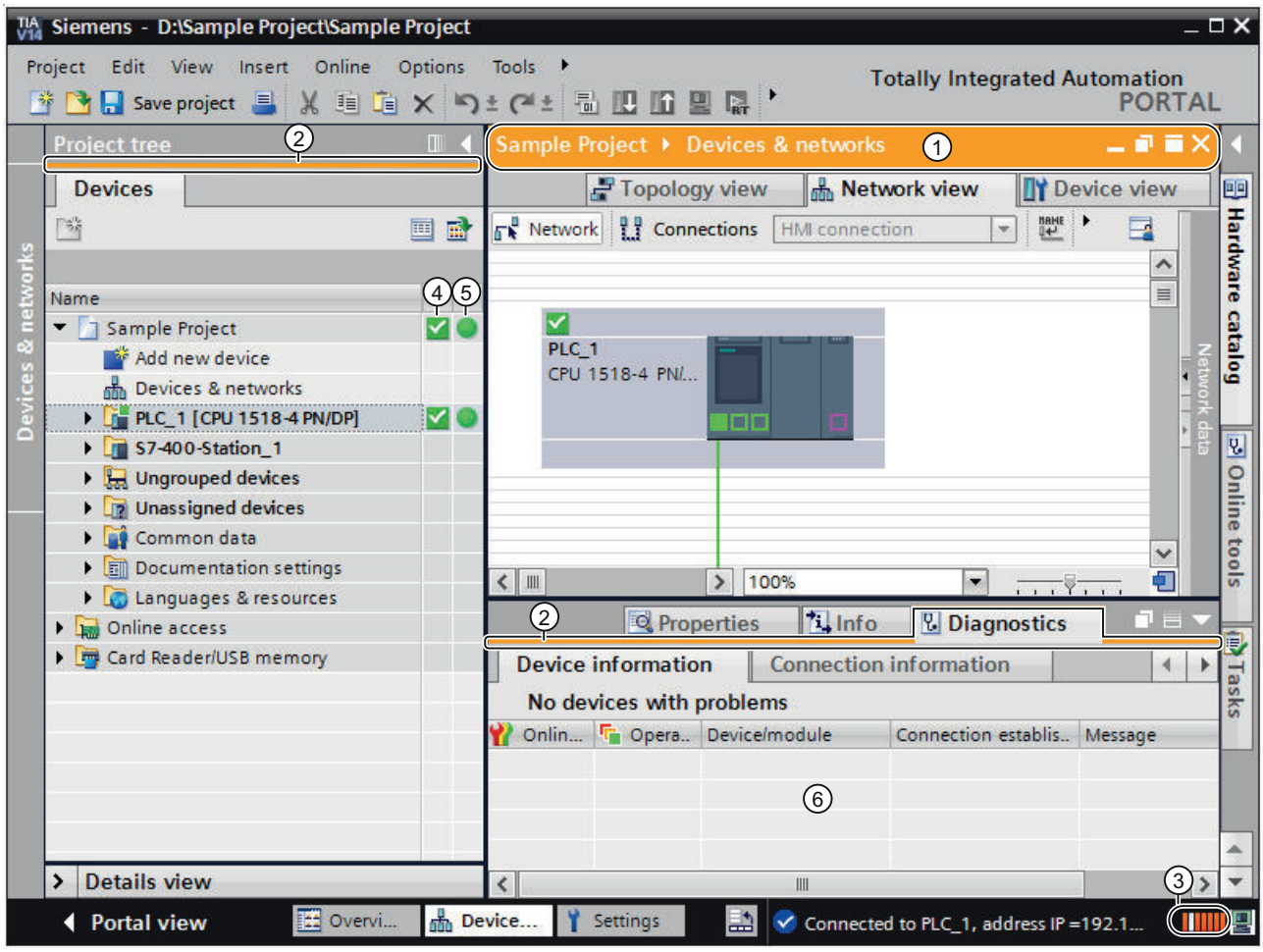
View in online mode (Page 6480)

Overview of the online and diagnostic settings (Page 235)

14.3.2 View in online mode

Online displays

After the online connection has been established successfully, the user interface changes. If a device is unavailable, this is indicated by a symbol. The following figure shows a device connected online and the corresponding user interface:



① The title bar of the active window gets an orange background as soon as at least one of the devices currently displayed in the editor has been successfully connected online. If one or more devices are unavailable, a symbol for a broken connection appears in the title bar of the editor.

② The title bars of inactive windows for the relevant station now have an orange line below them.

③ An orange, pulsing bar appears at the right-hand edge of the status bar. If the connection has been established but is functioning incorrectly, an icon for an interrupted connection is displayed instead of the bar. You will find more information on the error in "Diagnostics" in the Inspector window. In addition, the status line indicates if there is an online connection to the SIMATIC hardware via the TIA Portal Cloud Connector.

④ The left column in the project tree indicates the diagnostics status for hardware objects in online mode through symbols.

⑤ The right column in the project tree indicates the comparison status for software objects in online mode through symbols. A comparison of the online and offline state is performed automatically. Differences between online and offline objects are displayed in the form of symbols.

⑥ The "Diagnostics > Device information" area is brought to the foreground in the Inspector window.

Online connection abort

The online mode and its display are retained as long as at least one device is connected online. If the online connection of one or more devices breaks, the TIA Portal remains in online mode. The display of the TIA Portal changes to offline mode only when there is no longer an online connection for any device.

Online connection via the TIA Portal Cloud Connector

If you use the TIA Portal in a virtual environment, you have the opportunity to create an online connection via the TIA Portal Cloud Connector to SIMATIC hardware that is connected to a user device. If the online connection is made via the TIA Portal Cloud Connector, the following symbol is displayed in the status bar:



You can find instructions on how to use the TIA Portal in a virtual environment (private cloud) on the installation disk in the directory "Documents\Readme\<language directory>". You can open the PDF document "TIAPortalCloudConnectorHowTo<Sprachkennung>.pdf" here.

See also

General information about online mode (Page 6478)

Basics of project data comparison (Page 358)

14.3.3 Default setting online connection data

If you prefer to use a specific network interface of your programming device/PC to establish an online connection, you can preset this interface. The specified connection path is then already selected in the dialogs for online connection.

Procedure

To specify a default connection path, follow these steps:

1. Select the "Settings" command in the "Options" menu.
   The settings of the TIA Portal are opened.
2. Select the "Online & Diagnostics" entry in the area navigation.

3. In the "Preset connection path for online access" area, specify the type of the PG/PC interface as well as the interface itself.
4. Select the "Use preset connection path for online access" check box.

See also

Establishing or changing an online connection (Page 6482)

Displaying accessible devices (Page 6475)

Downloading project data to a device (Page 350)

Online access (Page 6495)

Overview of the online and diagnostic settings (Page 235)

14.3.4 Establishing or changing an online connection

To use the online functions of a device, specify the connection path the first time you establish the connection.

Requirement

At least one PG/PC interface is installed and is physically connected to a device, for example, with an Ethernet cable. As an alternative, it is also possible to establish virtual online connections using PLCSIM.

Procedure

To establish or change an online connection, follow these steps:

1. Select one or more devices for the online connection in the project tree.
2. Select the "Go online" command in the "Online" menu.
   - If the device has already been connected online, the online connection is automatically established using the previously specified connection path.
   - If there was no previous connection or if the device address is not located in the same subnet of the programming device/PC, the "Go online" dialog opens.

   If you want to change a specified connection path, select the "Extended go online" command instead in the "Online" menu. The "Go online" dialog box is opened again in this case.

3. Select the connection path:
   - Select the type of interface from the "Type of the PG/PC interface" drop-down list.
   - Select the interface of your programming device/PC from the "PG/PC interface" drop-down list.
     No selection is necessary if you have already defined a programming device/PC interface in the settings.
   - Select the interface or the subnet for the connection from the "Connection to interface/ subnet" drop-down list. Select a direct connection if no S7 gateway is connected in between. Select the matching subnet for the connection to the programming device or PC if the device can be accessed via an S7 gateway. Select the item "Try all interfaces" if the device has several interfaces and you do not know by which of the interfaces the device is connected to the programming device/PC.
     If you selected an MPI or PROFIBUS subnet, the bus parameters configured in the programming device/PC interface are applied at this point.
   - If the device is accessible via a gateway, select the gateway that connects the two subnets involved in the "1st gateway" drop-down list.
4. Optional: If no devices can be accessed by the selected connection path, a dashed connecting line is displayed between the programming device/PC and the device. This is the case, for example, when the device is connected with the network by a NAT router. To display all devices that are switched on and connected to an interface of the programming device/PC, select the entry "Show accessible devices" from the drop-down list and click "Start search" again.
5. Optional: You have two options to access the device despite a different subnet: Use the two options in administered networks only after consultation with the administrator.
   - Assign a static address to the programming device/PC interface under Windows.
   - During the creation of the online connection, an additional IP address in the same subnet as the device is temporarily assigned to the interface of the programming device/PC. Click "Go online" and confirm the next prompt.
6. Click the "Start search" button.
   Devices which can be reached by the set connection path are displayed in the table of the target devices. The connection line in the graphic on the left is displayed as solid.
7. Optional: Run a flash test. Select the "Flash LED" check box on the left in the graphic. You use the flash test to check that you have selected the correct device. The flash test is not supported by all devices.
8. Select the target device in the table and confirm the selection with "Go online".
   The online connection to the selected target device is established.

Result

After the online connection has been established, the title bars of the editors change to orange. An orange progress bar is also shown in the title bar of an editor and in the status bar. In the project tree, status symbols show the difference between online and offline objects.

The set connection path is stored for future connection attempts.

With automatic assignment of the programming device/PC address the dialog reappears when the computer is restarted. The connection must be re-established in this case.

Note
Undoing actions

You cannot undo actions once you have established an online connection.

Note
Using direct connections or subnets

An online connection between programming device/PC and connected devices is possible either via direct connection or via subnet.

Simultaneous connection of the devices over different subnets is not possible if these are physically connected through the same interface. Simultaneous connections to different subnets are only possible through routed connections.

Simultaneous connection of devices through a subnet and through direct connections using the same interface is not possible.

See also

Overview of the online and diagnostic settings (Page 235)

Default setting online connection data (Page 6481)

Canceling an online connection (Page 6484)

Connecting online with several devices (Page 6485)

View in online mode (Page 6480)

Assigning a temporary IP address (Page 6501)

Influence of user rights (Page 280)

14.3.5 Canceling an online connection

Procedure

To terminate the existing online connection, follow these steps:

1. Select the device for which you want to disconnect the online connection in the project tree.
2. Select the "Go offline" command in the "Online" menu.
   The online connection is disconnected.

Note

Undoing actions

You cannot undo actions once you have disconnected an online connection.

See also

Establishing or changing an online connection (Page 6482)

14.3.6 Connecting online with several devices

You can establish an online connection to several devices at the same time without needing to select individual devices previously in the network view.

Requirement

- No device must be selected
- At least one PG/PC interface is installed and is physically connected to a device, for example with an Ethernet cable. As an alternative, it is also possible to establish a virtual online connection using PLCSIM or a remote connection.

Procedure

To establish an online connection to several devices at the same time, follow these steps:

1. Select the project in the project tree.
2. Select the "Go online" command in the "Online" menu.
   The "Select device for opening the online connection" dialog opens with a table of all available devices.
3. Select the devices to which you want to establish an online connection in the "Go online" column.
4. Click the "Go online" button.

Result

Without any further prompt for confirmation, a connection is established to all selected devices if a connection was already established to the selected devices at least once. If there was no previous online connection, the "Go online" dialog opens. In this case, first configure the online connection as described in the section "Go online and Go offline (Page 6482)".

See also

Establishing or changing an online connection (Page 6482)

Assigning a temporary IP address (Page 6501)

14.3.7 Disconnecting online connections of multiple devices

You can disconnect the online connections to multiple devices at one time without needing to select individual devices beforehand in the network view.

Requirement

- No device is selected.
- There is currently an online connection to at least one device.

Procedure

To terminate the online connections to multiple devices at one time, follow these steps:

1. Select the "Go offline" command in the "Online" menu.
   The "Select devices" dialog opens with a table of all available devices.
2. Select the device for which you want to terminate the online connection in the "Go offline" column.
3. Click the "Go offline" button.

Result

The online connection to the all the selected devices is terminated.

14.4 Creating a backup of an S7 CPU

14.4.1 Backup options for S7 CPUs

You will make a number of changes to your automation system over time, for example, add new devices, replace existing devices or adapt the user program. If these changes were to result in undesirable behavior, you can restore the automation plant to an earlier version. To continue working without interruptions after replacing individual devices, you can accept existing programs and values. The CPUs offer different options for backup and restoration of the hardware configuration and software.

Backup options

The table below provides an overview of the backup and restoration options of S7 CPUs:

	Snapshot of the monitored values	Upload from device (software)	Upload device as new station (hardware and software)	Download backup from online device
Use case	Restoring a specific status of a data block. The actual values of data blocks including time stamp are accepted in the project.	Upload blocks on a CPU to the project.	Upload of hardware configuration and software from a device to the project.	Create a complete backup of a CPU as restore point. The backup copy is consistent and cannot be changed or opened.
Requirement	The CPU has already been created in a project. The data blocks must be identical online and offline.	The CPU is created in the project.	The device is available in the hardware catalog of TIA Portal. Any necessary HSPs or GSD files are installed.	-
Possible in mode	RUN, STOP	RUN, STOP	RUN, STOP	STOP
Possible for F-CPUs	Yes	Yes	No	Yes
Backup can be edited	Yes	Yes	Yes	No

Backup contents

The table below shows which data you can upload and back up with which options:

	Snapshot of the monitored values	Upload from device (software)	Upload device as new station (hardware and software)	Download backup from online device
Actual values of the data blocks	Snapshot is possible	Upload is possible	Upload is possible	Backup is possible
Software blocks	-	Upload is possible	Upload is possible	Backup is possible
PLC tags (tags and constants names)	-	Upload possible for S7-1200 and S7-1500 CPUs	Upload possible for S7-1200 and S7-1500 CPUs	Backup is possible
Technology objects	-	Upload is possible	Upload is possible	Backup is possible

	Snapshot of the monitored values	Upload from device (software)	Upload device as new station (hardware and software)	Download backup from online device
Hardware configuration	-	-	Upload is possible	Backup is possible
Monitoring tables (web server)	-	-	Upload is not possible	Backup is possible
Local data, bit memories, timers, counters and process image	Snapshot is not possible	Upload is not possible	Upload is not possible	Backup is possible
Archives and recipes (PLC)	-	-	-	Backup is possible
General data on the SIMATIC Memory Card, for example, help for program blocks or GSD files	-	-	-	Backup is possible

Special considerations during backup of actual values

The "Backup from online device" type of backup backs up the actual values of the tags that are set as retentive. To ensure consistency of the retentive data, disable all write access to retentive data during the backup.

Actual values of the non-retentive data are reset to their start values during a transition from STOP to RUN mode. When the CPU is backed up only the start values of non-retentive data are backed up.

See also

Creating a backup of a device (Page 6489)

Creating a backup of a device (Page 6491)

General information on loading (Page 348)

Downloading project data to a device (Page 350)

Downloading project data to a memory card (Page 352)

Uploading project data from a device (Page 354)

Loading project data from a memory card (Page 356)

14.4.2 Backing up S7-300 and S7-400 CPUs (S7-300, S7-400, PC)

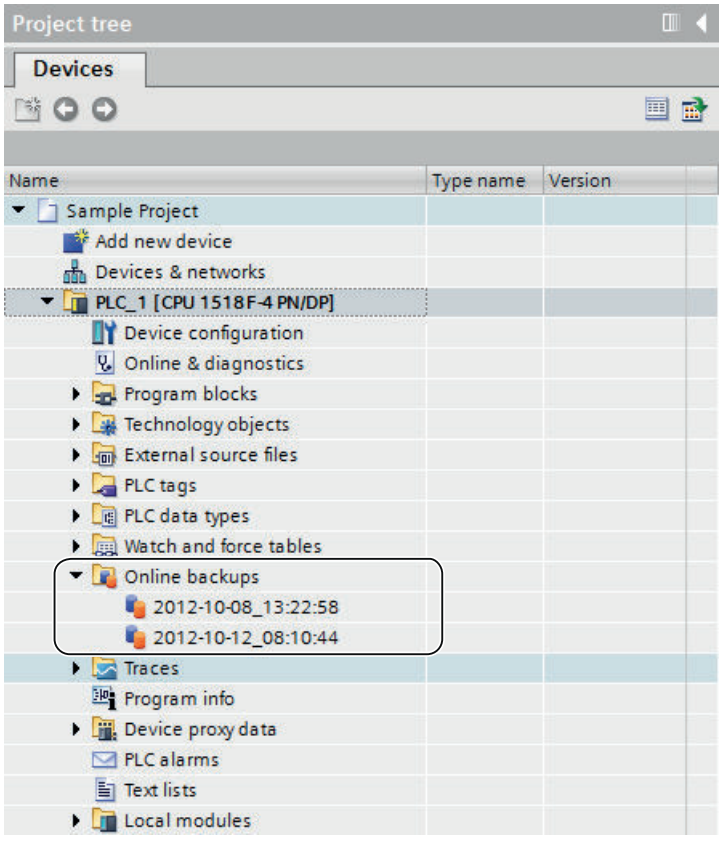
14.4.2.1 Creating a backup of a device (S7-300, S7-400, PC)

Backing up the software and hardware configuration of an S7-300/400 CPU

If you have already downloaded a configuration to an S7-300/400 CPU, it is advisable to make a backup. You may have modified the configuration and want to test the new configuration. Before you download the new configuration to the CPU, you can create a backup of the current device state and then restore the current configuration at a later date. The backup is performed with the current values of the CPU. In the case of S7-400 CPUs with fail-safe function, the initial values are backed up.

You can create as many backups as you want and store a variety of configurations for a CPU. The backups are named with the name of the CPU and the time and date of the backup. You can find the backup in the project tree under the CPU in the "Online backups" folder.

The following figure shows an S7-319 CPU for which two backups were created:



See also

Backup options for S7 CPUs (Page 6487)

14.4.2.2 Restoring the software and hardware configuration of a device (S7-300, S7-400, PC)

If you have backed up the configuration of a device at an earlier point in time, you can transfer the backup back to the device. The saved configuration is then restored on the device.

Requirement

You must have previously configure the device and stored a backup of the device in the project.

Procedure

To restore older software and hardware state on a device, follow these steps:

1. Open up the folder of the device in the project tree to display the lower-level objects.
2. Open the "Online backups" folder.
3. Select the backup you want to restore.
4. In the "Online" menu, select the "Download to device" command.
   - If you had previously established an online connection, the "Load preview" dialog opens. This dialog displays alarms and recommends actions needed for the loading operation.
   - If you had not previously established an online connection, the "Extended download to device" dialog opens, and you must first select the interfaces via which you want to establish the online connection to the device.
     See also: Establishing and terminating an online connection
5. Check the alarms in the "Load preview" dialog, and select the actions in the "Action" column, if necessary.

   Note

   Performing the proposed actions during operation of the plant can cause serious damage to property or injury to persons if there are functional disturbances or program errors.

6. As soon as downloading becomes possible, the "Load" button is enabled.
7. Click the "Load" button.
   The backup is transferred to the device and device is restored. The "Load results" dialog then opens. In this dialog, you can check whether or not the loading operation was successful and take any further action that may be necessary.
8. Click the "Finish" button.

14.4.2.3 Backing up a device configuration (S7-300, S7-400, PC)

You can back up the configuration of an S7-300/400 CPU in the TIA Portal. So you can download and test a new configuration to a device without any risk. If needed, you can restore the initial configuration of the CPU.

Requirement

- The CPU has already been created in the project.
- The CPU is online. If there is no online connection yet, an online connection is established during the backup.

Procedure

To create a backup of the current configuration of a CPU, follow these steps:

1. Select the CPU in the project tree.
2. Select the "Backup from online device" command in the "Online" menu.

Result:

A backup of the entire hardware configuration and software is created. The backup is stored in the project tree in the "Name of the CPU > Online backups" folder. The backup is assigned the name of the CPU with the time and date of the backup. You can rename the backup, but you cannot make any changes to the contents of the backup.

14.4.3 Backing up S7-1200 and S7-1500 CPUs (S7-1200, S7-1500, S7-1500, S7-1500)

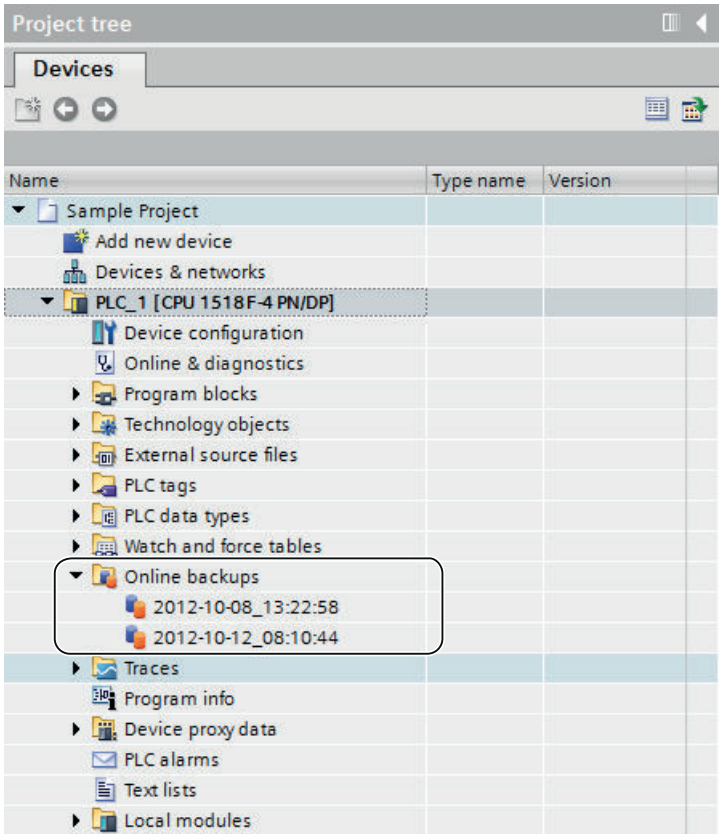
14.4.3.1 Creating a backup of a device (S7-1200, S7-1500, S7-1500, S7-1500)

Backing up the software and hardware configuration of a CPU

If you have already downloaded a configuration to an S7-1200/S7-1500 CPU, it may be useful to make a backup. You may have modified the configuration and want to test the new configuration. Before you download the new configuration to the CPU, create a backup of the current device version. You can restore the current configuration at a later time.

You can create as many backups as you want and store a variety of configurations for a CPU. The backups are named with the name of the CPU and the time and date of the backup. You can find the backup in the project tree under the CPU in the "Online backups" folder.

The following figure shows an S7-1500 CPU for which two backups have been created:



Scope of the backup

The backup includes all data that are needed to restore a particular configuration version of a CPU. The following data is backed up, for example:

- Contents of the memory card
- Retentive memory areas for example of data blocks, counters, and bit memory
- Other retentive memory contents, such as IP address parameters

The backup is performed with the current values of the CPU. Entries in the diagnostic buffer are not included in the backup. The current time of day is not backed up for an S7-1500 CPU.

See also

General information on loading (Page 348)

Backup options for S7 CPUs (Page 6487)

14.4.3.2 Backing up a device configuration (S7-1200, S7-1500, S7-1500, S7-1500)

You can back up the configuration of a CPU in the TIA Portal. So you can download and test a new configuration to a device without any risk. If needed, you can restore the initial configuration of the CPU.

The CPU goes to STOP while a backup is being created. If an access level is configured for the CPU, you need the password for read access to the CPU.

Requirement

- The CPU has already been created in the project.
- The device is connected to the programming device/PC directly via the PROFINET interface of the CPU. PROFIBUS interfaces of the CPU and interfaces of CMs/CPs are not supported.
- The CPU is online. If there is no online connection yet, an online connection is established during the backup.
- The CPU is in "STOP" mode.

Procedure

To create a backup of the current configuration of a CPU, follow these steps:

1. Select the CPU in the project tree.
2. Select the "Backup from online device" command in the "Online" menu.
   If necessary, you must enter the password for read access to the CPU and confirm that the CPU should enter "STOP" mode.

Result:

A backup of the entire hardware configuration and software is created. The backup is stored in the project tree in the "Name of the CPU > Online backups" folder. The backup is assigned the name of the CPU with the time and date of the backup. You can rename the backup, but you cannot make any changes to the contents of the backup.

An entry is created for each backup operation in the diagnostic buffer of the CPU.

See also

Restoring the configuration of a device (Page 6493)

14.4.3.3 Restoring the configuration of a device (S7-1200, S7-1500, S7-1500, S7-1500)

If you have backed up the configuration of a device at an earlier point in time, you can transfer the backup back to the device. The saved configuration is then restored on the device.

The CPU goes to STOP while a backup is being loaded to the CPU. If an access level is configured for the CPU, you need the password for write access to the CPU.

WARNING

Download backups with unknown content

If you activate the recommended actions during download during plant operation, you can cause serious damages or injuries in case of malfunctions or program errors.

Make sure that the backup contents do not include a configuration which results in unexpected plant behavior.

Requirement

- The device is configured and a backup of the device is stored in the project.
- The device is connected to the programming device/PC directly via the PROFINET interface of the CPU.
- The CPU is in "STOP" mode.
- You have the password for read access to the CPU, if an access level was configured.

Procedure

To restore older software and hardware state on a device, follow these steps:

1. Open up the folder of the device in the project tree to display the lower-level objects.
2. Open the "Online backups" folder.
3. Select the backup you want to restore.
4. In the "Online" menu, select the "Download to device" command.
   - If you had previously established an online connection, the "Load preview" dialog opens. This dialog displays alarms and recommends actions needed for the loading operation.
   - If you had not previously established an online connection, the "Extended download to device" dialog opens, and you must first select the interfaces via which you want to establish the online connection to the device.
     See also: AUTOHOTSPOT
5. Check the alarms in the "Load preview" dialog, and select the actions in the "Action" column, if necessary.
6. As soon as downloading becomes possible, the "Load" button is enabled.
7. Click the "Load" button.
   The backup is transferred to the device and device is restored. The "Load results" dialog then opens. In this dialog, you can check whether or not the loading operation was successful and take any further action that may be necessary.
8. Click the "Finish" button.
   If necessary, you must enter the password for read access to the CPU and confirm that the CPU should enter "STOP" mode.
   The contents of the backup are restored on the CPU. The CPU is then restarted.

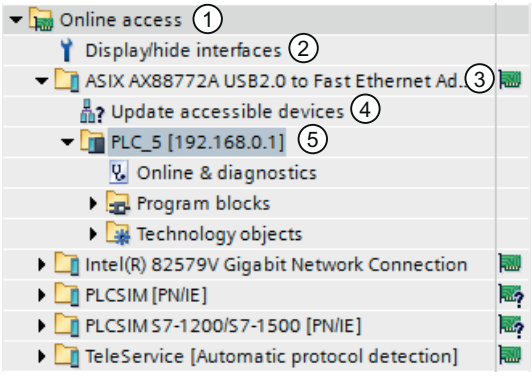
14.5 Configuring the PG/PC interface

14.5.1 Online access

Online access of the project

In the "Online access" folder of the project tree, you will find all active interfaces of your programming device/PC. Each interface icon provides you with information on the status. You can also display the accessible devices and display and edit the properties of an interface using the shortcut menu.

The following figure shows the "Online access" folder in the project tree:



① "Online access" folder in the project tree

All interfaces installed in the programming device/PC are displayed in the "Online access" folder.

② Show/hide interfaces

Use the function "Show/hide interfaces" to show or hide individual interfaces.

③ Status display for the interfaces

The current status of an interface is indicated by an icon to the right of the name. You can see the meaning of the icon in the tooltip.

④ Updating the list of accessible devices.

This function is available for each hardware interface of the programming device/PC. Software interfaces, such as a remote connection, do not offer this function.

⑤ Devices connected via the respective interface with the programming device/PC

The type of the device and its status are displayed by the preceding icon.

Displaying or updating accessible devices

You have the following options if you want to display all devices accessible online on your programming device/PC:

- Display of the accessible devices on a single interface of the programming device / PC in the project tree. In the project tree, you can also display additional information about the individual accessible devices.
- Display of the accessible devices of all interfaces in a list.

See also:AUTOHOTSPOT

Overview of icons for accessible devices

The accessible devices are identified with an icon according to their type and status. The following is an overview of all icons and their meaning.

	Icon for unidentified modules This icon is displayed whenever the identification of a module is not yet complete or when the identification of a module was not successful, for example, because the required online data could not be read.
	Icon for the following device types: • PLCs • SIMOCODE pro devices • IE/PB links • CPs of PC systems • SCALANCE head modules • S7-300 and S7-400 CPs • PROFINET IO devices and PROFINET CPs • SCALANCE modules and gateways that could not be identified
	PROFINET IO devices, encoders, switchgear, sensors and identification systems that were replaced by similar devices because these could not be identified
	Icon for the following device types: • HMI devices • PROFINET IO devices of the HMI type if these could not be identified and were therefore replaced by a similar device
	PROFINET IO devices of the drive type that could not be identified and were therefore replaced by a similar device
	PROFINET IO devices of the type development kit and network components that could not be identified and were therefore replaced by a similar device
	PROFINET IO devices of the type TeleService adapter that could not be identified and were therefore replaced by a similar device

See also

Displaying and modifying interface properties (Page 6498)

14.5.2 Basics of assigning parameters for the PG/PC interface

Options for connecting to target systems

If the devices of the project are connected via different subnets, you assign a suitable network access to each PG/PC interface to be able to establish online connections to the target systems. The following interfaces are automatically supported:

- MPI
- PROFIBUS
- Industrial Ethernet (ISO and TCP/IP)

You can make various settings for the interfaces. The following sections explain the parameter settings you can make.

Note

Note that changes to interface parameters have a direct influence on the operating system and the programming device / PC. Remember that some parameter settings can only be changed if you have adequate user rights.

See also

Setting parameters for the Industrial Ethernet interface (Page 6499)

Setting parameters for the MPI and PROFIBUS interfaces (Page 6503)

14.5.3 Showing or hiding interfaces

To improve clarity, it is possible to hide individual interfaces of the PG/PC in the project tree and to show them again when necessary.

Procedure

To show or hide individual interfaces, follow the steps below:

1. Open the "Online access" folder in the project tree.
2. Double click on the "Display/hide interfaces" icon.
   The "Display/hide interfaces" dialog opens.
3. Enable or disable the required interfaces in the "Display in the project tree" column.
4. Click the "Apply" button.
   The changes are applied. The view of the interfaces in the "Online access" folder is refreshed.

14.5.4 Displaying and modifying interface properties

Introduction

For each interface, you can display and, in some cases, modify properties, for example the network type, address, and status.

Procedure

To open the properties, follow these steps:

1. Right-click on the required interface below "Online access" in the project tree.
2. Select the "Properties" command from the shortcut menu.
   A dialog containing the properties of the interface opens. On the left of the dialog, you will see the area navigation. You can view the current parameter settings in the individual entries in the area navigation and, if necessary, change them.

14.5.5 Adding interfaces

You have the option of installing additional interfaces after installation of the TIA Portal.

Procedure

To install an interface at a later time and add it to the TIA Portal, follow these steps:

1. Install or update the drivers in the operating system once you have installed the interface hardware.
2. Close the TIA Portal if it is still open.
3. Open the Windows control panel.
4. Open the entry "Setting the PG/PC Interface" in the Control Panel.
   The "Setting the PG/PC Interface" dialog opens.
5. Make any necessary changes to the interface configuration and confirm them with "OK".
   You have to click "OK", even if you have not made any changes.
6. Restart the TIA Portal.

Result

The newly installed interface is now displayed in the project tree under the "Online access" folder.

14.5.6 Setting parameters for the Ethernet interface

14.5.6.1 Setting parameters for the Industrial Ethernet interface

Options in the parameter settings for the Industrial Ethernet interface

When setting parameters for the Industrial Ethernet interface, you have the following options:

- Parameters dependent on the operating system
  The Industrial Ethernet interface has parameters that are set in the operating system and are valid for all connected devices. These parameter settings are only displayed here, they can, however, be changed in the network settings of the operating system.
- Parameters that can be set in the software

Note

Note that changes to interface parameters have a direct influence on the operating system and the programming device / PC. Remember that some parameter settings can only be changed if you have adequate user rights.

Parameters for the Industrial Ethernet interface

The following table contains an overview of the parameters of the Industrial Ethernet interface that are set by the operating system and can be changed by the user.

Parameter settings that cannot be changed	Parameters that can be set
MAC address	Fast acknowledge at the IE-PG access and for TCP/IP
DHCP server activated/deactivated	Timeout at the IE-PG access and for TCP/IP
APIPA activated/deactivated	LLDP
IP address	Additional, dynamic IP addresses for the network adapter
Subnet mask	-
DNS addresses	-
DHCP addresses	-

See also

Basics of assigning parameters for the PG/PC interface (Page 6497)

Displaying operating system parameters (Page 6500)

Connecting the PG/PC interface to a subnet (Page 6500)

Setting parameters for the Ethernet interface (Page 6501)

Assigning a temporary IP address (Page 6501)

Managing temporary IP addresses (Page 6502)

Influence of user rights (Page 280)

14.5.6.2 Displaying operating system parameters

The Ethernet interface is part of the operating system. All parameters of the network adapter can therefore be adapted in the network settings of the operating system.

You can display the following parameters in the software:

- Physical address of the network adapter
- Assignment of the IP address by a DHCP server activated or deactivated
- Assignment of a private IP address by the operating system activated or deactivated
- Current static IP address
- Assigned subnet mask
- DNS addresses
- DHCP addresses

If you want to modify the parameter settings, please refer to the documentation of the operating system or the network adapter.

Displaying current parameters of the Ethernet interface

To display the current parameters of the Ethernet interface, follow these steps:

1. Select the Ethernet interface in the project tree in "Online access".
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Select "Configurations > Industrial Ethernet" in the area navigation.

See also

Setting parameters for the Ethernet interface (Page 6501)

14.5.6.3 Connecting the PG/PC interface to a subnet

If you have created several subnets, you can specify the subnet to which the Ethernet interface is connected.

Procedure

To select the subnet to which the Ethernet interface is connected, follow these steps:

1. Select the Ethernet interface in the project tree in "Online access".
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Go to "General > Assignment" and select the subnet to which you want to connect the Ethernet interface of the programming device / PC in the "Connection to subnet" drop-down list.
4. Close the dialog with "OK".

14.5.6.4 Setting parameters for the Ethernet interface

You can adapt some parameter settings relating to the network protocol directly in the software.

Requirement

You must have adequate user rights.

See also: Influence of user rights (Page 280).

Procedure

To change parameter settings relating to the network protocol, follow these steps:

1. Select the Ethernet interface in the project tree in "Online access".
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Select "Configurations > IE-PG access" to adapt the protocol settings relevant for network management.
   - Select the "Fast acknowledge" check box to achieve faster reaction times with smaller network packets.
   - From the "Timeout" drop-down list, select the maximum time that can elapse before a network node is detected.
4. To activate the LLDP protocol and discover the network topology more accurately, set the "LLDP active" check box in "Configurations > LLDP".
5. Select "Configurations > TCP/IP" to adapt the TCP/IP protocol for network traffic during runtime.
   - Select the "Fast acknowledge" check box to achieve faster reaction times with smaller network packets.
   - From the "Timeout" drop-down list, select the maximum time that can elapse before there is a timeout during communication with a network node.

See also

Influence of user rights (Page 280)

Displaying operating system parameters (Page 6500)

14.5.6.5 Assigning a temporary IP address

Adding a dynamic IP address

If the IP address of a device is located in a different subnet from the IP address of the network adapter, you will first need to assign an additional IP address with the same subnet address as the device. Only then is communication between the device and the programming device / PC possible.

The assignment of an additional temporary IP address is also proposed automatically if you want to perform an online action and the current IP address of the programming device/PC is not yet in the correct subnet.

A temporarily assigned IP address remains valid until the next time the programming device/PC is restarted or until you delete it manually.

Note

You require adequate permissions to be able to assign a temporary IP address.

See also: Influence of user rights (Page 280)

See also

Managing temporary IP addresses (Page 6502)

14.5.6.6 Managing temporary IP addresses

If the IP address of a device is located in a different subnet from the current permanently assigned IP address of the network adapter, a suitable IP address from the subnet of the device is temporarily assigned to the network adapter.

You can display all temporarily assigned addresses and delete them. Note that IP addresses that you manually assigned in the operating system are not displayed in the TIA Portal.

Requirement

To delete, you require adequate permissions.

Procedure

To display and delete temporarily assigned addresses, follow these steps:

1. Select the Ethernet interface in the project tree in "Online access".
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Select "Configurations > IE-PG access".
   A table with the assigned IP addresses is displayed.
4. Click the "Delete project-specific IP addresses" button to delete all the IP addresses at one time.

See also

Influence of user rights (Page 280)

14.5.6.7 Resetting the TCP/IP configuration

If you have changed the TCP/IP protocol settings, you can reset them to the defaults.

Procedure

To restore the TCP/IP configuration to the default settings, follow these steps:

1. Select the Ethernet interface in the project tree in "Online access".
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Select "Configurations > TCP/IP".
4. Click the "Standard" button to reset all the settings.

14.5.7 Setting parameters for the MPI and PROFIBUS interfaces

14.5.7.1 Setting parameters for the MPI and PROFIBUS interfaces

Possible parameter settings for the MPI and PROFIBUS interfaces

The following parameter settings can be made for the MPI and PROFIBUS interfaces:

- Automatic configuration: You can use automatic detection functions to find out whether a device is connected to the PG/PC interface over PROFIBUS or MPI.
- Selecting a default configuration for PROFIBUS or MPI that can be adapted later.

Device- and network-related settings for MPI and PROFIBUS

You can set device- and network-related parameters for MPI and PROFIBUS interfaces. Device-related parameters are local settings for the interface. Network-related parameters, on the other hand, must match up on all devices.

MPI interface parameters you can modify

You can adapt the following default parameters for the MPI interface:

Device-related parameters	Network-related parameters
Is only master	Highest address
Own address	Transmission rate
Timeout	

PROFIBUS interface parameters you can modify

You can adapt the following default parameters for the PROFIBUS interface:

Device-related parameters	Network-related parameters
Is only master	Highest address
Own address	Transmission rate
Timeout	Profile

Device-related parameters	Network-related parameters
	Bus parameters
	Number of masters on bus
	Number of slaves on bus

Note

Behavior when a subnet is assigned

If you permanently assign a subnet from the project to the interface, the subnet settings of the project take precedence. In this case, the settings for the respective interface cannot be changed.

See also

Basics of assigning parameters for the PG/PC interface (Page 6497)

14.5.7.2 Setting MPI or PROFIBUS interface parameters automatically

Setting up automatic bus parameter detection

If you select an interface with automatic detection of the bus parameters (for example CP 5611 (Auto)), you can connect the programming device or PC to MPI or PROFIBUS without needing to set bus parameters. At a transmission speed lower than 187.5 Kbps, you may, however, have waiting times of up to one minute.

Requirement

- Masters that distribute bus parameters cyclically are connected to the bus.
- In PROFIBUS networks, the cyclic distribution of bus parameters is enabled.
- The interface is not assigned to a subnet. If you assign a subnet in the properties of the interface, the settings for the subnet in the project take precedence. In this case, the settings for the automatic interface configuration cannot be changed.

Procedure

To enable automatic bus parameter detection, follow these steps:

1. Select the interface in the project tree.
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Go to "General > Configurations > Active configuration" and select the setting "Automatic protocol detection".
4. Go to "Configurations > Auto configuration > Local settings" and select the address of the PG/PC interface in the "Own address" drop-down list.
5. If you then want to display the current bus settings, click the "Network detection" button.

See also

Setting parameters for the MPI interface (Page 6505)

Setting parameters for the PROFIBUS interface (Page 6507)

14.5.7.3 Setting parameters for the MPI interface

Changing the parameter settings of the MPI interface

The network-related parameters and bus parameters for the MPI network can be adapted. You should first select a default setting and then adapt this to the specific situation.

Requirement

The interface is not assigned to a subnet. If you assign a subnet in the properties of the interface, the settings for the subnet in the project take precedence. In this case, the interface configuration cannot be changed.

Setting defaults for the MPI interface

To adapt the parameters of the MPI interface, follow these steps:

1. Select the interface in the project tree.
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Go to "General > Assignment" and select the subnet with which you want to connect the interface in the "Connection to subnet" drop-down list.
4. Under "General > Configuration", select a default for the device and network-related parameters. The defaults are suitable for most configurations. Select one of the following settings:
   - Automatic protocol detection
     You can connect the programming device to MPI or PROFIBUS without having to set bus parameters. At a transmission speed lower than 187.5 Kbps, you may, however, have waiting times of up to one minute. Prerequisite for the automatic detection is a connection to the bus master, which distributes the bus parameters cyclically. With PROFIBUS subnets, cyclic distribution of bus parameters may not be deactivated (default PROFIBUS network setting).
   - MPI
     The "MPI" transmission protocol is selected. Typical parameters are set that are adequate for most configurations. You can change the parameters to your needs, however.
   - PROFIBUS
     The "PROFIBUS" transmission protocol is selected. Typical parameters are set that are adequate for most configurations. You can change the parameters to your needs, however.

Changing the default parameter settings

To adapt the default settings to your requirements, change the parameter setting where necessary in "Configurations > MPI".

You can set the following device-related parameters:

- Is only master
  An additional verification function to prevent bus disruptions when connecting the PG/PC to the network is disabled because the programming device or PC is the only master on the bus.
  - Do not enable this option unless you have only connected slaves to your programming device or PC.
  - If the "Is only master" check box is enabled, it is not possible to identify the directly connected device in the "Accessible devices" window.
- Own address
  This setting relates to the programming device or PC on which you call up the parameter settings of the interface. Set the local device address of your programming device or PC here.
  - This address must be unique throughout the network.
  - The programming device or PC is addressed using this address in the MPI network.
- Check
  This enables an additional safety function to prevent bus disruptions when connecting the PG/PC to the network. The driver checks whether the local address is already being used by another station. Active as well as passive stations are taken into consideration in this case. The driver monitors this on the PROFIBUS. The connection of the PG/PC to the network will take longer with the automatic check. To use the check, the driver must support the function. Furthermore, the "Is only master" option must not be selected.
- Timeout
  Set a higher timeout value if, for example, you have problems with long response times on the network.

You can set the following network-related parameters:

- Highest address:
  Select the configured highest device address. Make sure that the same highest device address is set for all devices of a PROFIBUS or MPI network.
- Transfer rate:
  Here, you select the transmission speed to be used on the MPI network.

See also

Setting MPI or PROFIBUS interface parameters automatically (Page 6504)

14.5.7.4 Setting parameters for the PROFIBUS interface

Changing the parameter settings of the PROFIBUS interface

The network-related parameters and bus parameters for the PROFIBUS network can be adapted more precisely. You should first select a default setting and then adapt this to the specific situation.

Requirement

The interface is not assigned to a subnet. If you assign a subnet in the properties of the interface, the settings for the subnet in the project take precedence. In this case, the interface configuration cannot be changed.

Setting defaults for the PROFIBUS interface

To adapt the parameters of the PROFIBUS interface, follow these steps:

1. Select the interface in the project tree.
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Go to "General > Assignment" and select the subnet with which you want to connect the interface in the "Connection to subnet" drop-down list.
4. Under "General > Configuration", select a default for the device and network-related parameters. The defaults are suitable for most configurations. Select one of the following settings:
   - Automatic protocol detection
     You can connect the programming device to MPI or PROFIBUS without having to set bus parameters. At a transmission speed lower than 187.5 Kbps, you may, however, have waiting times of up to one minute. Prerequisite for the automatic detection is a connection to the bus master, which distributes the bus parameters cyclically. With PROFIBUS subnets, cyclic distribution of bus parameters may not be deactivated (default PROFIBUS network setting).
   - MPI
     The "MPI" transmission protocol is selected. Typical parameters are set that are adequate for most configurations. You can change the parameters to your needs, however.
   - PROFIBUS
     The "PROFIBUS" transmission protocol is selected. Typical parameters are set that are adequate for most configurations. You can change the parameters to your needs, however.

Changing the default parameter settings

To adapt the default settings to your requirements, change the parameter setting where necessary in "Configurations > PROFIBUS".

You can set the following device-related parameters:

- Is only master
  An additional verification function to prevent bus disruptions when connecting the PG/PC to the network is disabled because the programming device or PC is the only master on the bus.
  - Do not enable this option unless you have only connected slaves to your programming device or PC.
  - If the "Is only master" check box is enabled, it is not possible to identify the directly connected device in the "Accessible devices" window.
- Own address
  This setting relates to the programming device or PC on which you call up the parameter settings of the interface. Set the local device address of your programming device or PC here.
  - This address must be unique throughout the network.
  - The programming device or PC is addressed using this address in the PROFIBUS network.
- Check
  This enables an additional safety function to prevent bus disruptions when connecting the PG/PC to the network. The driver checks whether the local address is already being used by another station. Active as well as passive stations are taken into consideration in this case. The driver monitors this on the PROFIBUS. The connection of the PG/PC to the network will take longer with the automatic check. To use the check, the driver must support the function. Furthermore, the "Is only master" option must not be selected.
- Timeout
  Set a higher timeout value if, for example, you have problems with long response times on the network.

You can set the following network-related parameters:

- Highest address:
  Select the configured highest device address. Make sure that the same highest station address is set for all devices of a PROFIBUS network.
- Transfer rate:
  Here, you select the transmission speed to be used on the PROFIBUS network.
- Profile:
  You have a choice of four alternatives for the PROFIBUS settings. "DP", "Standard" and "Universal (DP/FMS)" are predefined settings that you cannot change. If you select "User-defined", you can adapt the bus parameters yourself.
  - If you have selected "User-defined", go to "Configurations > PROFIBUS > Bus parameters" in area navigation.
  - If you have selected one of the defaults (DP, Standard or Universal (DP/FMS)), you should select the "Include" check box in "Configurations > PROFIBUS > Bus parameters > Additional parameters". You can then set the number of masters and slaves on the bus. This allows a more precise calculation of the bus parameters and potential bus disruptions can be prevented. The option cannot be selected with a user-defined profile.

See also

Overview of the bus parameters for PROFIBUS (Page 6509)

Setting MPI or PROFIBUS interface parameters automatically (Page 6504)

14.5.7.5 Overview of the bus parameters for PROFIBUS

Introduction

The PROFIBUS subnet will only function problem-free if the parameters for the bus profile are matched to one another. You should therefore only change the default values if you are familiar with how to configure the bus profile for PROFIBUS.

It may be possible for the bus parameters to be adjusted depending on the bus profile. The offline values of the bus parameters are always shown even if you are online and linked to the target system.

The displayed parameters are valid for the entire PROFIBUS subnet.

Meaning of the individual parameters

- Tslot: Wait-to-receive time (slot time)
  The wait-to-receive time (slot time) defines the maximum time the sender will wait to receive a response from the addressed partner.
- Max. Tsdr: maximum protocol processing time (max. station delay responder)
  The maximum protocol processing time defines the time after which the responding device must have processed the protocol.
- Min. Tsdr: minimum protocol processing time (min. station delay responder)
  The minimum protocol processing time specifies the minimum time required by the responding device to process the protocol.
- Tset: Trigger time (setup time)
  The trigger time is the time that may lapse between the reception of a data frame frame and the reaction to it.
- Tqui: Quiet time for modulator
  The quiet time for modulator specifies the time required to change from sending to receiving.
- GAP factor: GAP update factor (GAP factor)
  The GAP update factor specifies the number of token rotations before a new device is included in the token ring.
- Retry limit: maximum number repeated calls (retry limit)
  This parameter defines the maximum number of attempts (message frame repeats) made to reach a node.
- Trdy: Ready time
  The ready time is the time for an acknowledgment or response.
- Tid1: Idle time 1
  Idle time 1 specifies the delay time after receiving a response.
- Tid2: Idle time 2
  Idle time 2 specifies the delay time after sending a call without a response.

- Ttr: Target rotation time
  The target rotation time is the maximum time made available for a token rotation. During this time, all active devices (masters) receive the token once. The difference between the desired token round-trip time and the actual token round-trip time decides how much time is left for masters to send data message frames to the slaves.
  As the minimum target rotation time (Ttr), select a value = 5000 times the HSA (Highest Station Address).
- Watchdog: Watchdog
  The watchdog time specifies the time after which a device must be addressed.
  As the minimum watchdog time, select a value = 6250 times the HSA.

Note

If you want to create a user-defined bus profile, note that the minimum target rotation time (Ttr) should be 5000 times the HSA (highest PROFIBUS address). The minimum watchdog time should also be 6250 times the HSA.

See also

Setting parameters for the PROFIBUS interface (Page 6507)

14.5.7.6 Resetting the MPI or PROFIBUS configuration

If you have changed the MPI or PROFIBUS protocol settings, you can reset them to the defaults.

Requirement

The interface is not assigned to a subnet. If you assign a subnet in the properties of the interface, the settings for the subnet in the project take precedence. In this case, the interface configuration cannot be reset to the default values.

Procedure

To restore the MPI or PROFIBUS configuration to the default settings, follow these steps:

1. Select the MPI/PROFIBUS interface in the project tree in "Online access".
2. Select the "Properties" command in the shortcut menu of the interface.
   The dialog for configuring the interface opens.
3. Select "Configurations > MPI" or "Configurations > PROFIBUS", depending on the interface properties you want to reset.
4. Click the "Standard" button to reset all the settings.

14.6 Using the trace and logic analyzer function

14.6.1 Preface

Purpose of the documentation

The diagnostics options available with the trace and logic analyzer function are described in this documentation. Depending on the device used, the recording options can vary.

Required basic knowledge

In order to understand this documentation, the following knowledge is required:

- General knowledge in the field of automation
- Knowledge about the use of Windows-based computers
- S7-1200/1500 CPUs, ET 200SP, ET 200Pro
  - Knowledge of the SIMATIC industrial automation system
  - Knowledge of working with STEP 7
- SINAMICS G120
  - Knowledge of working with the drive
- SINAMICS S120
  - Knowledge of working with the drive

Validity of the documentation

This documentation applies to all products of the S7-1200, S7-1500, SINAMICS G120 and SINAMICS S120 product family as of TIA Portal V14.

Conventions

This documentation contains pictures of the devices described. The pictures may differ slightly from the devices supplied.

Please also observe notes marked as follows:

Note

A note contains important information on the product described in the documentation, on the handling of the product and on the section of the documentation to which particular attention should be paid.

"mySupport"

With "mySupport", your personal working area, you make the most of your Industry Online Support.

In "mySupport" you can store filters, favorites and tags, request CAx data and put together your personal library in the Documentation area. Furthermore, your data is automatically filled into support requests and you always have an overview of your current requests.

You need to register once to use the full functionality of "mySupport".

You can find "mySupport" on the Internet (https://support.industry.siemens.com/My/ww/en/).

"mySupport" - Documentation

In the Documentation area of "mySupport", you have the possibility to combine complete manuals or parts of them to make your own manual.
You can export the manual in PDF format or in an editable format.

You can find "mySupport" - Documentation on the Internet (http://support.industry.siemens.com/My/ww/en/documentation).

Further support

- The range of technical documentation for the individual SIMATIC products and automation systems can be found on the Internet (http://www.siemens.com/simatic-tech-doku-portal).
- The online catalog and the online ordering system is available on the Internet (https://mall.industry.siemens.com).

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.

Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.

Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit (http://www.siemens.com/industrialsecurity).

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under (http://www.siemens.com/industrialsecurity).

14.6.2 Description

14.6.2.1 Supported hardware

If a device supports the trace and logic analyzer function, "Traces" is offered for selection in the project tree below the device.

The following devices (Page 6557) support the trace and logic analyzer function:

- SIMATIC S7-1200 CPUs (as of firmware version V4.0)
- SIMATIC S7-1500, ET 200SP and ET 200Pro CPUs
- SIMATIC S7-1500 software controller
- SINAMICS G120
- SINAMICS S120
- SIRIUS SIMOCODE pro (with Simocode ES)
- SIRIUS soft starter 3RW (with Soft Starter ES)

14.6.2.2 Recording of measured values with the trace function

Introduction

The trace and logic analyzer function can be called in the project navigator (Page 6517) by double-clicking an entry in the "Traces" system folder. The measurements on the memory card can also be read and displayed via the diagnostic interface of the Web server.

You record device tags and evaluate the recordings with the trace and logic analyzer function. Tags are, for example, drive parameters or system and user tags of a CPU. The number of installed traces is hardware-dependent.

The recordings are saved on the device and, when required, can be read out with the engineering system (ES) and saved permanently. The trace and logic analyzer function is

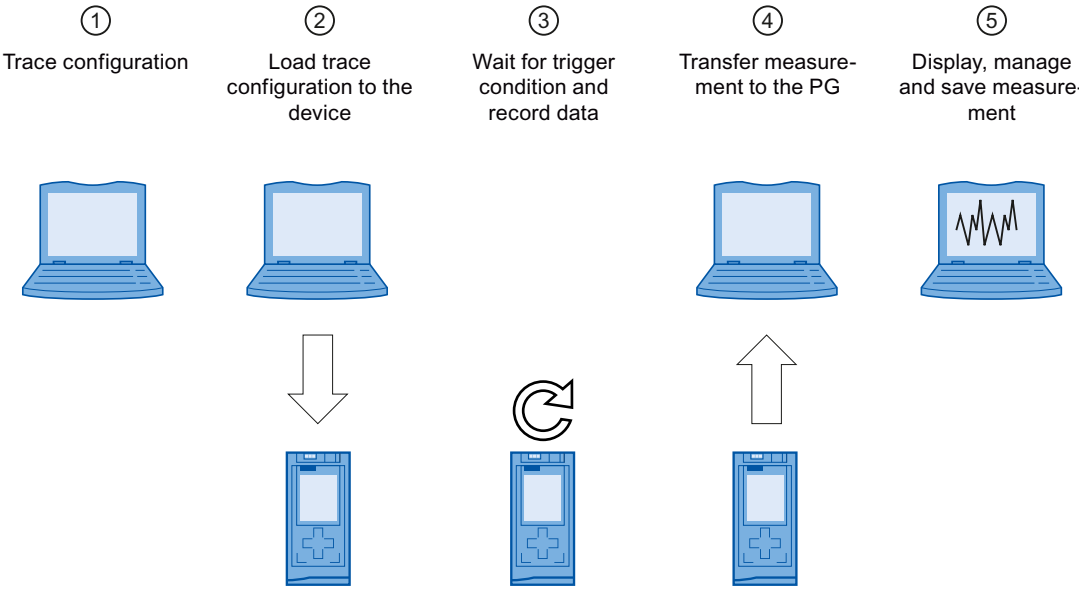
therefore suitable for monitoring highly dynamic processes. The recorded values are overwritten when the recording is activated again.

Active recordings from the axis control panel are displayed in the "Traces" system folder as installed traces. Recordings can be added to the measurements in the curve diagram of the axis control panel or the PID via a shortcut menu command.

Depending on the device (Page 6557) used, the recording options can vary.

A quick start (Page 6537) for working with the trace and logic analyzer function can be found in the Operation section.

The following figure shows the method of operation of the "Traces":



① Trace configuration on the programming device (PG) in the TIA Portal

You can specify the signals to be recorded, the duration of the recording and the trigger condition in the trace configuration. The trace configuration depends on the device and is described at the respective device (Page 6557).

② Transferring the trace configuration from the PG to the device

You can transfer the complete trace configuration (Page 6544) to the device when an online connection is established.

③ Waiting for the recording

If the installed trace configuration is activated (Page 6545), then the recording is performed independently of the PG. The recording is started as soon as the trigger condition is satisfied.

④ Transferring the measurement from the device to the PG

The saving of the measurement in the project (Page 6547) stores the measurement in the opened project of the TIA Portal. The measurement can be saved at any time after completing the recording, irrespective of the time of the measurement.

⑤ Evaluating, managing and saving the measurement

Numerous options are available for the evaluation of the measurement in the curve diagram and in the signal table (Page 6546). Various display types are possible, for example, a bit representation for binary signals.
Signal waves from different measurements can be put together as an overlay measurement and compared with each other.
Measurements can also be exported and imported as a file.
With the saving of the project (Page 6547) in the TIA Portal, the measurements transferred to the project are also saved.

Supported TIA Portal features

The following TIA Portal features are supported as of V14:

- Saving of trace configurations and measurements in libraries
- Multiuser support for trace configurations and measurements (offline data)

14.6.2.3 Trace configuration, recording, installed trace and measurement

This section explains the meaning and relationships of the terms: trace configuration, recording, installed trace and measurement.

Trace configuration

Implement the following settings in the trace configuration:

- Signals to be recorded
- Recording conditions
  - Sampling
  - Trigger
  - Installed measurements (memory card)

Recording

A recording is performed in the device. There is only one recording for each installed trace configuration. When a new recording is started, the old recording is overwritten.

An installed recording is not retentive (it is lost when the device is switched off/on) but can be saved permanently in the project as a measurement.

Installed trace

An installed trace consists of a trace configuration and optionally a recording. The maximum number of installed traces depends on the device.

The trace configuration is stored retentively on the device. The retentivity of the trace configuration may also be configurable depending on the device, e.g. with the S120.

Measurement

A measurement consists of a trace configuration and a recording, provided that recorded data is present. Each installed trace can be saved as a measurement in the project.

The recording of a measurement can be viewed offline.

The configuration for a measurement can be transferred to the “Traces” folder using drag & drop.

Installed measurements (memory card)

The folder contains measurements that are saved on the device (e.g. on the memory card). These measurements are retentive and can only be deleted by the user.

The installed measurements can be transferred to the Measurements folder using drag & drop and are then saved as measurements in the project.

Trace configuration with an installed trace of the same name

Usually, there is a trace configuration in the project with the same name for an installed trace. When there is an online connection, this trace is displayed with the icon in the project tree.

See also User interface - “Traces” project tree folder (Page 6519).

Overlay measurement

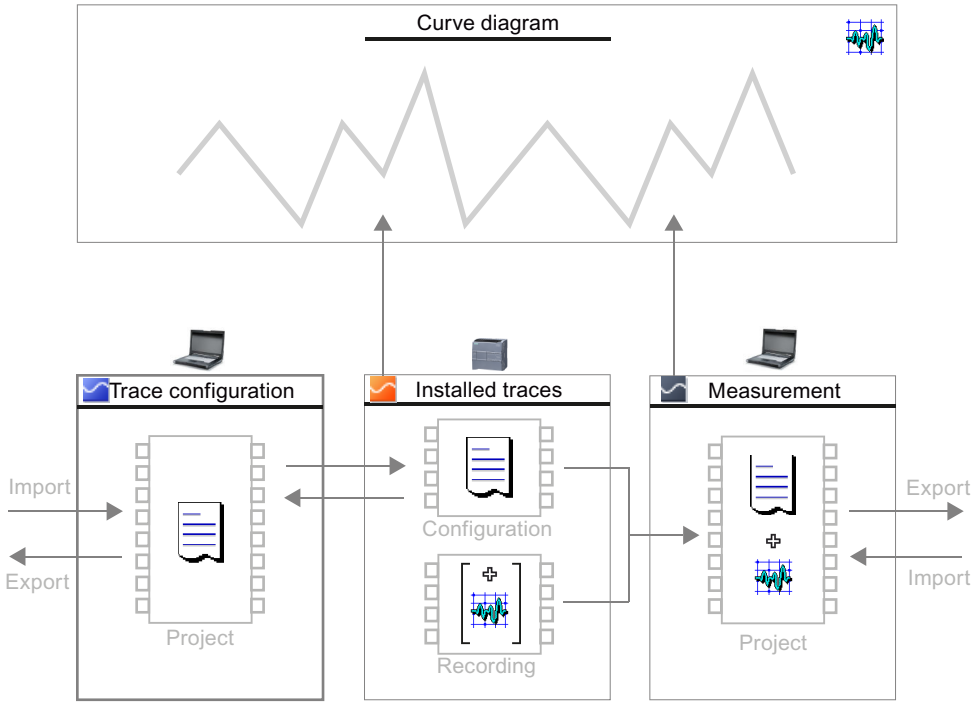
The overlay measurement allows a comparison and analysis of signals from different measurements with each other.

The measurements can be synchronized with each other and displayed as overlay measurements.

14.6.2.4 Data storage

The trace toolbar and the curve diagram also enable the transfer of the trace configuration and the viewing of the recording.

The following figure is a schematic diagram of the data storage:



Note

Saving the trace configuration and measurement

You save the trace configuration and measurement with the project in the TIA Portal.

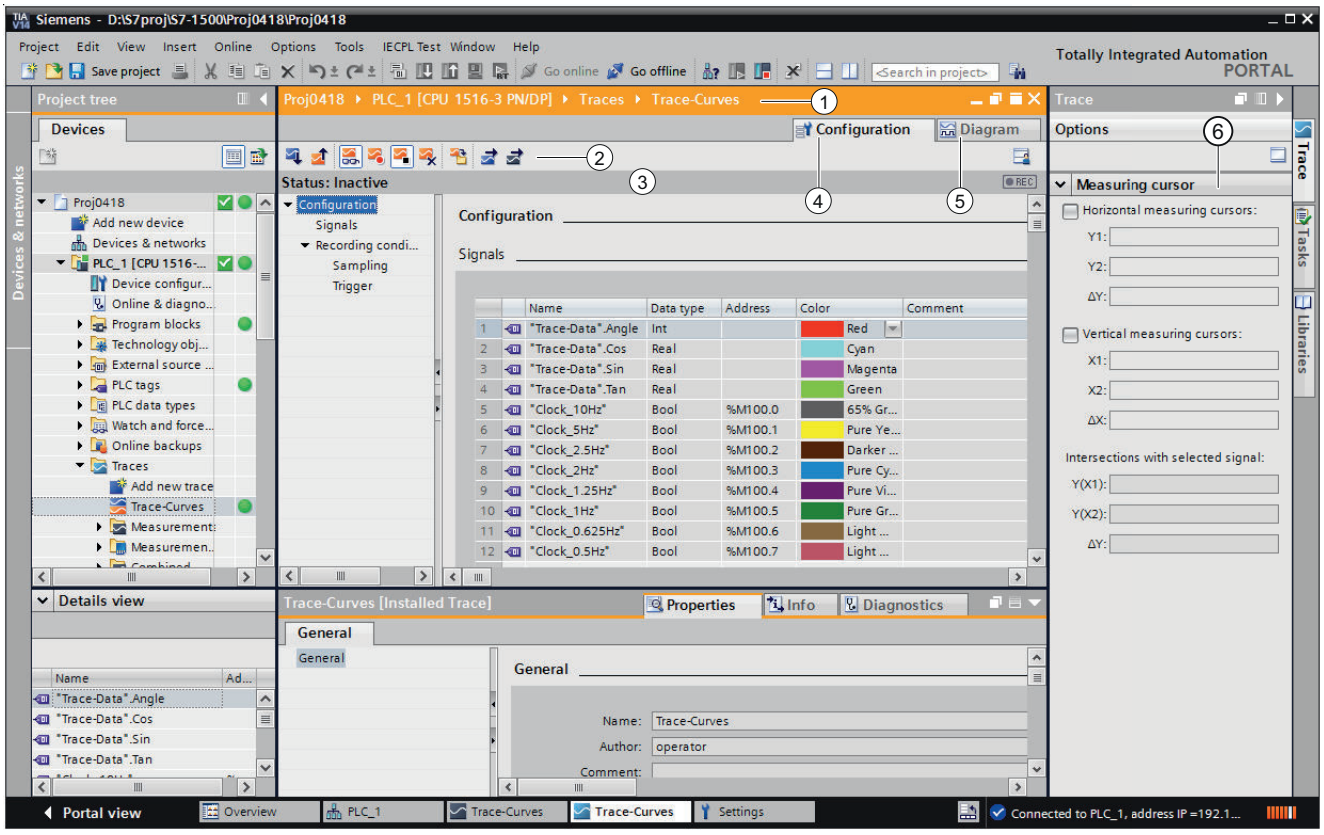
If you close the project without saving, the trace configurations and the measurements transferred to the project are discarded. The trace editor can be closed and reopened without loss of data until the project is closed.

14.6.3 Software user interface

Display areas

The user interface of the trace and logic analyzer function consists of several areas.

The example in the figure below shows the division of the user interface in the TIA Portal:



	Project navigator
Management and creation of the trace and measurements directly in the project tree and via context menu commands.	
	Working area

	① Title bar of the working area Shows the device to which the current display belongs.
	② Trace toolbar Buttons for managing the trace in the project and device: • Activation/deactivation of installed traces • Deletion of installed traces • Transfer of trace configurations and measurements between the device and the project • Export of trace configurations and measurements • Switchover between offline and online display
	③ Status display of the trace Display of the current status of the recording.
	④ Configuration tab Device-specific configuration of the recording duration, trigger condition and signal selection. See Device-specific descriptions (Page 6557).
	⑤ Diagram tab Display of the recorded values as a curve diagram and the signals from the displayed measurement.
	Signal selection tab Display of all signals that are contained in the overlay measurements.
"Trace" task card Display of the measurement cursor data ⑥ and snapshots.	
Inspector window Display of general information about the trace configuration	

14.6.3.1 Project navigator

User interface - “Traces” project tree folder

Trace configurations and installed traces are displayed in the "Traces" folder.

More information about the “Traces” sub-folder is provided in the following sections.

Double-click a trace to open the corresponding "Diagram" or "Configuration" tab in the working area.

Icons in the "Traces" folder

The following table explains the icons in the "Traces" folder:

Icon	Description
	Add trace configuration Double-click the icon to add a new trace configuration.
	Trace configuration (offline) Double-click the icon to open the "Configuration" tab.
	Installed trace (online) The icon is only displayed when there is no offline trace configuration of the same name for the installed trace. Double-click the icon to open the "Diagram" tab.
	Trace configuration with an installed trace of the same name If the button is deactivated, the trace configuration from the project is displayed. The trace corresponds to a trace configuration. If the button is activated, the trace configuration from the device is displayed. The trace corresponds to an installed trace. Double-click the icon to open the "Diagram" tab.

Status

When there is an online connection, the status is displayed in the right-hand column of the project tree. The status is also displayed as tooltip above the respective icon.

The following table shows the meaning of the icons:

Icon	Description
	Online and offline configuration are identical
	Online and offline configuration are different
	Configuration only exists online

Shortcut menu commands

The following table shows the shortcut menu commands for the "Traces" system folder:

Shortcut menu command	Description
“Add new group”	Inserts a new folder.
“Add new trace”	Inserts a new trace configuration and opens the configuration tab.
"Import trace configuration"	Imports a trace configuration from a file.

The following table shows the shortcut menu commands for trace configurations and installed traces / :

Shortcut menu command	Trace configuration	Installed trace	Description
"Copy"	x	-	Copies the trace configuration of the selected objects to the clipboard.
"Paste"	x	-	Inserts a trace configuration or measurement from the clipboard.
"Delete"	x	x	Deletes the selected objects from the project tree or from the device.
"Rename"	x	-	Switches the selected object to the editing mode.
"Export trace configuration"	x	-	Exports a trace configuration as a file with the extension "*.ttcfgx". For compatibility reasons, the "*.ttcfgx" file extension will be supported in V12, but does not contain any information about the device family.

The trace configuration can also be copied device-wide.

Several objects can be selected.

User interface - “Measurements” project tree folder

The “Measurements" folder shows the saved measurements.

Icons in the “Measurements" folder

The following table explains the icons in the “Measurements" folder:

Icon	Description
	Measurement (offline) Double-click the icon to open the "Diagram" tab. The configuration for a measurement can be transferred to the “Traces” folder using drag & drop.

Shortcut menu commands

The following table shows the shortcut menu commands for the "Measurements" system folder:

Shortcut menu command	Description
“Add new group”	Inserts a new folder.
"Import measurement"	Imports a measurement from a file with the "*.ttrecx" file extension. For compatibility reasons, the "*.ttrec" file extension will be supported in V12, but does not contain any information about the device family.

The following table shows the shortcut menu commands for measurements :

Shortcut menu command	Description
"Copy"	Copies the trace configuration of the selected objects to the clipboard.
"Paste"	Inserts a measurement from the clipboard.
"Delete"	Deletes the selected objects from the project tree or from the device.
"Rename"	Switches the selected object to the editing mode.
“Generate new overlay measure-ment”	Generates a new overlay measurement with the selected measurements.
"Export measure-ment"	Exports a measurement with the last saved standard view The measurement is saved with the extension "*.ttrecx" or "*.csv". For compati-bility reasons, the "*.ttrec" file extension will be supported in V12, but does not contain any information about the device family.

The measurements can also be copied device-wide.

Several objects can be selected.

User interface - “Installed measurements (memory card)” project tree folder

The folder shows all measurements present on the memory card. The folder is only displayed when there is an online connection to the device.

Drag folders or measurements contained here to the “Measurements” system folder using drag & drop. This transfers the measurements to the project.

Icons in the "Traces" folder

The following table explains the icons in the system folder :

Icon	Description
	Folders generated automatically with information on the recording start time The name of the folder cannot be changed.
	Installed measurement Double-click the icon to open the "Diagram" tab. The time stamp in the name shows the occurrence of the trigger event.

Shortcut menu commands

The following table shows the shortcut menu commands for the folder :

Shortcut menu command	Description
"Copy"	Copies the selected objects to the clipboard.
"Delete"	Deletes the selected objects from the project tree and from the device.

The following table shows the shortcut menu commands for measurements :

Shortcut menu command	Description
“Open”	Opens the measurement in the “Diagram” tab.
"Copy"	Copies the selected objects to the clipboard.
"Delete"	Deletes the selected objects from the project tree and from the device.
"Export measure-ment"	Exports a measurement as a file with the extension "*.ttrecx" or "*.csv". For compatibility reasons, the "*.ttrec" file extension will be supported in V12, but does not contain any information about the device family.
“Properties”	Displays the general properties of the measurement (Page 6536).

Several objects can be selected.

User interface - “Overlay measurements” project tree folder

The system folder “Overlay measurements" shows the configured overlay measurements.

Icons in the “Overlay measurements" folder

The following table explains the icons in the system folder “Overlay measurements”:

Icon	Description
	Add new overlay measurements Double-click the icon to add a new overlay measurement and open the “Diagram” tab.
	Overlay measurement Double-click the icon to open the "Diagram" tab.

Shortcut menu commands

The following table shows the shortcut menu commands for the system folder “Overlay measurements" or for a group folder contained within this:

Shortcut menu command	Description
“Add new group”	Inserts a new folder.
“Add new overlay measurement”	Inserts a new overlay measurement and opens the “Diagram” tab.

The following table shows the shortcut menu commands for overlay measurements:

Shortcut menu command	Description
“Open”	Opens the selected overlay measurements in the working area.
"Import measurement"	Imports a measurement from a file with the "*.ttrecx" file extension. For compatibility reasons, the "*.ttrec" file extension will be supported in V12, but does not contain any information about the device family.
"Copy"	Copies the selected objects to the clipboard.

Shortcut menu command	Description
"Paste"	Inserts measurements, measurements from installed traces or overlay measurements from the clipboard. Multiple objects can be inserted from the clipboard if they are all of the same type.
"Delete"	Deletes the selected objects from the project tree or from the device.
"Rename"	Switches the selected object to the editing mode.
“Properties”	Displays the general properties for the overlay measurements.

The overlay measurements can also be copied device-wide.

Several objects can be selected.

14.6.3.2 Working area

User interface - trace toolbar

Tools are available for handling the trace via buttons.

The following table shows the functions of the buttons:

Icon	Description
	Transfer the selected trace configuration to the device The selected trace configuration is transferred to the device.
	Transfer the selected trace configuration from the device The selected trace configuration is transferred from the device to the project.
	Observe on/off Change of the display between online and offline. **Note** Once monitor and automatic scaling are activated at the same time, no more actions can be undone using the “Undo” button. **Note** When an installed trace is first started the display in the curve diagram is set to automatic scaling by default. Make sure when the recording is restarted that any changes to the scaling settings are retained. Reactivate automatic scaling manually if necessary in order to monitor the recording.
	Activate recording If the recording of an installed trace is repeated, then the settings relevant for the display (curve diagram and signal table) are also retained for the new recording. **Note** When a recording is restarted, the previously recorded values are lost. To save the recorded values, save the measurement in the project (Page 6547) before you activate the recording again.
	Deactivate recording
	Delete installed trace Deletes the selected trace from the device.

Icon	Description
	Transfer the selected measurement from the device to the project The measurement is added to the "Measurements" system folder. **Note** In order to be able to save the installed trace data as a measurement, it must first have been displayed once in the curve diagram. The recording data is loaded from the device when displayed. If necessary wait until the display is updated, as only the displayed data is transferred to the project in all cases.
	Export trace configuration Exports a trace configuration as a file with the extension "*.ttcfgx". For compatibility reasons, the "*.ttcfgx" file extension will be supported in V12, but does not contain any information about the device family.
	Generate a trace configuration Generates a new trace configuration from the measurement.
	Export measurement with the settings from the current view Exports a measurement as a file with the extension "*.ttrecx" or "*.csv". For compatibility reasons, the "*.ttrec" file extension will be supported in V12, but does not contain any information about the device family.
	Import measurement (only with overlay measurements) Imports a measurement from a file with the "*.ttrecx" file extension. For compatibility reasons, the "*.ttrec" file extension will be supported in V12, but does not contain any information about the device family.
	Select a measurement (only with overlay measurements) The drop down list box contains the imported measurements. Select the desired measurement to display the configuration.

User interface - Configuration tab

User interface - Configuration

The trace configuration depends on the device and is described at the respective device (Page 6557).

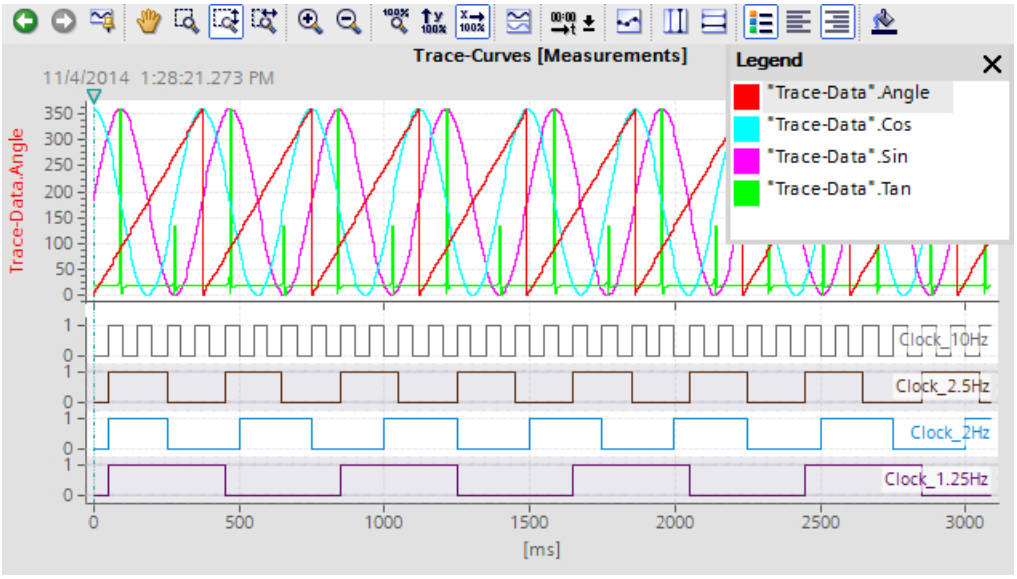
User interface - Diagram tab

User interface - curve diagram

The curve diagram displays the selected signals of a recording. Bits are shown in the lower diagram as a bit track. You can adjust the display of the signals in the signal table (Page 6529) and with the toolbar of the curve diagram.

Setting options and displays in the curve diagram

The following figure shows an example of the display in the TIA Portal:



The scale in the diagram applies to the selected (highlighted in gray) signal in the legend. The legend can be moved and its size can be adjusted with the mouse.

The ▽ icon shows the device trigger time with a vertical line.

Functions using the mouse wheel

The following table shows which functions are possible in the curve diagram using the mouse wheel:

Function of the mouse wheel	Description
Move the curve diagram vertically	Turning the mouse wheel moves the display in the upper curve diagram up or down. If the signals are arranged in traces the display for the group is shifted that is located below the cursor. The cursor must be positioned above the upper curve diagram.
Move the curve diagram horizontally	Turning the mouse wheel with the <Shift> button pressed down moves the display in the curve diagram to the left or the right. The cursor must be positioned above the curve diagram.
Zoom in and zoom out	Turning the mouse wheel with the <Ctrl> button pressed down zooms in or out of the display in the curve diagram. The cursor position is the starting point for zooming in or out. The value axis of the lower curve diagram (bit tracks) is not affected. The cursor must be positioned above the curve diagram.

Shortcut menu commands

The following table shows the shortcut menu commands in the curve diagram:

Shortcut menu command	Description
"Save diagram as image"	Exports the current display in graphic format, e.g. as a bitmap.
"Copy image to clipboard"	Copies the current display to the clipboard.
"Center measurement cursors"	Positions the activated measurement cursors at a central point in the current display.
"Add to measurements" (only axis control panel and PID)	Adds the displayed recording to the "Measurements" system folder.
“Automatic bit track height"	Automatically adjusts the height of the bit tracks and thereby determines the size of the lower curve diagram. The setting is automatically deactivated once you change the space allocation between the curve diagrams manually. **Note** You can change the vertical space allocation between the upper and lower curve diagram. To do this drag the time axis of the upper curve diagram up or down with the mouse.

Toolbar of the curve diagram

Tools are available for adapting the display via buttons.

The following table shows the functions of the buttons:

Icon	Function	Description
	Undo zoom	Undoes the zoom function executed last. If several zoom functions have been executed, they can be undone step-by-step.
	Redo zoom	Redoes the last undone zoom function. If several zoom functions have been undone, they can be redone step-by-step.
	Standard view	Uses the current view as standard for this recording. If the trace recording is shown again later, the standard view is restored.
	Move view	Moves the display with the mouse button pressed.
	Zoom selection	Selection of an arbitrary range with the mouse button pressed. The display is scaled to the range selection.
	Vertical zoom selection	Selection of a vertical range with the mouse button pressed. The display is scaled to the range selection.
	Horizontal zoom selection	Selection of a horizontal range with the mouse button pressed. The display is scaled to the range selection.
	Zoom in	Enlargement of the display. The ranges of the time axis and value axis are reduced every time the button is clicked. The curves are displayed larger.

Icon	Function	Description
	Zoom out	Reduction of the display. The ranges of the time axis and value axis are increased every time the button is clicked. The curves are displayed smaller.
	Display all	Scales the display of the available data so that the entire time range and all values are displayed.
	Automatic scaling of the value axis	Scaling of the display so that all values are displayed for the currently displayed time range. The relative scaling ratio between the signals may change. **Note** The automatic scaling of the value axis is stopped when the zoom function is activated for the value axis. This button reactivates the automatic adjustments to the minimum/maximum values.
	Show the overall time range	Scaling of the display so that the values in the value range currently displayed are displayed for the overall time range. The value range displayed only then changes if “Display all values”  is activated. **Note** The automatic scaling of the time axis is stopped when a zoom function is activated for the time axis. This button reactivates the automatic adjustments for the time range.
	Arrange in tracks	Activate or deactivate the trace arrangement. When the trace arrangement is activated the signals are arranged among themselves with the relevant value axes. Signal groups are displayed in the same trace. This setting does not affect the display for the bit tracks.
	Unit changeover of the time axis	Switching the unit of the time axis The following units are adjustable: • Measurement points • Time (relative time related to the trigger time) • Absolute time
	Display samples	The samples are displayed as small circles on the curves.
	Display vertical measurement cursors	Display of the vertical measurement cursors. The vertical position of the two measurement cursors can be moved with the mouse. The associated measured values and the difference of the measurement cursors corresponding to the position are shown in the signal table. Display the “Measurement cursor” pane (Page 6534) in the Trace task card in order to display further information.

Icon	Function	Description
	Display horizontal measurement cursors	Display of the horizontal measurement cursors. The horizontal position of the two measurement cursors can be moved with the mouse. Display the “Measurement cursor” pane (Page 6534) in the Trace task card in order to display the values or to reposition the measurement cursor through entering the position.
	Display chart legend	Showing or hiding of the legend in the curve diagram and the bit track labels.
	Align the chart legend to the left	Display of the legend and the bit track labels on the left side of the curve diagram.
	Align the chart legend to the right	Display of the legend and the bit track labels on the right side of the curve diagram.
	Change background color	Changeover between various background colors.

User interface - signal table

The signal table lists the signals of the selected measurement and provides setting options for some properties. If recording data of installed traces is displayed and the settings are changed in the signal table, these settings are retained until there is a change to the offline mode. If the installed trace is added to the measurements, the current settings of the signal table are saved in the measurement.

The signals can be sorted using drag-and-drop. The bits of a signal can be resorted within a signal.

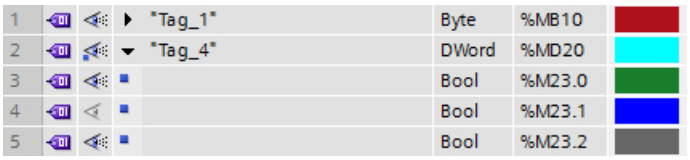
Setting options and displays in the signal table

The following figure shows an example of the display in the TIA Portal:

			Name	Data type	Address	Color	Scaling group	Min. Y scale	Max. Y scale	Y(t1)	Y(t2)	Del..	Unit	Comment
1			"Trace-Data".Angle	Int			Temp	-99.99987...	360	188	183	-5		
2			"Trace-Data".Cos	Real			Temp	-99.99987...	360	-99....	-99....	-0....		
3			"Trace-Data".Sin	Real				-99.99971...	99.999969...	-13....	-5.0...	8.6...		

The following table shows the settings and displays of the recorded signals:

Column	Description
	Static display of the signal icon
	Selection for the display in the curve diagram The point indicates that at least one bit has been selected for display as bit track for the signal in the bit selection.
"Name"	Display of the signal name A click on the name of a displayed signal updates the scale in the curve diagram.
“Measurement”	Display of the measurement (in the “Diagram” tab only with overlay measurements) Shows the name of the measurement to which the signal belongs.

Column		Description
		Open bit selection Individual bits can also be selected for the following data types for display as a bit track in the lower curve diagram. • Byte, Word, DWord, LWord • SInt, USInt, Int, UInt, DInt, UDInt, LInt, ULInt Example of an opened bit selection for the DWORD data type:  Select or deselect the relevant bit for display by clicking the icon.
"Data type"		Display of the data type
"Address"		Display of the address of the signal The field remains empty with optimized / type correct tags.
"Color"		Display and setting option for the color of the signal
"Signal group"		Display or input of the signal group name for one signal group The Y-scales are scaled identically for all signals of one signal group. Enter an identical signal group name for those signals that are to be scaled identically. Signals can be removed from the signal group by deleting the signal group name. The signal groups can be saved via the function "Use current view as standard" (button). **Note** Binary signals cannot be grouped.
	Gray field for chain icon	Move the cursor over the gray field or the chain icon (or) to add the signal to a signal group or delete the signal from the signal group. Clicking the chain icon adds the signal to a signal group or creates a new signal group. Clicking the chain icon removes the signal from the signal group. For a selected signal with signal group, the chain icon displays all signals of the same signal group.
	Input field	The input field displays the signal group name. As an alternative to the chain icon, you can assign or delete a group name via text input in this field.
"Min. Y-scale"		Display or input of the minimum value for the scaling of the signal
"Max. Y-scale"		Display or input of the maximum value for the scaling of the signal
"Y(t1)"		Display of the value at the position of the first measurement cursor
"Y(t2)"		Display of the value at the position of the second measurement cursor

Column	Description
"ΔY"	Display of the value difference between the first and the second measurement cursor
"Unit"	Display of the unit (e.g. for used-based values from technology objects)
"Comment"	Display and input option for a comment about the signal

Shortcut menu commands

The following table shows the shortcut menu commands of the signal table:

Shortcut menu command	Description
"Cut"	Cannot be selected.
"Copy"	Copies the contents of the selected lines to the clipboard.
"Paste"	Cannot be selected.
"Delete"	Cannot be selected.
"Rename"	Cannot be selected
“Display signal(s)”	Displays the selected signals in the curve diagram.
“Hide signal(s)”	Hides the selected signals in the curve diagram.

See also

Use of the signal table (Page 6552)

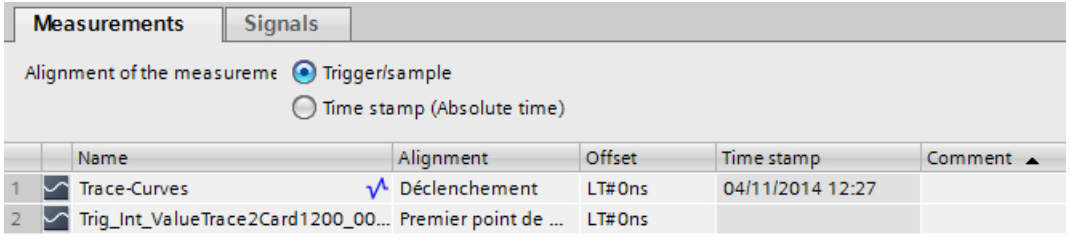
Using the signal group in the signal table (Page 6553)

User interface - Measurements (overlay measurements)

The Measurements tab displays the individual measurements and among other things provides the setting options for synchronization.

Setting options and displays in the Measurements tab

The following figure shows an example of the display in the TIA Portal:



The following table shows the settings and displays for the measurements:

Column		Description
Alignment of the measurements		
	“Trigger/ measure- ment point"	Alignment of the measurements in accordance with the trigger or measurement point The individual zero point for the measurement is predefined in the table under the “Alignment” column.
	“Time stamp (absolute time)”	Alignment of the measurements in accordance with their time stamp The signals are aligned in accordance with the time from the absolute time stamp.
Table columns		
		Static display of the measurement icon
	"Name"	Display and change options for the measurement name The name must be a unique one and can be changed.
	“Alignment”	Alignment of the measurement (only adjustable with the “Trigger/measurement point” check box activated) Determines the individual zero point for a measurement. All signals for the measurement are displayed in relation to this zero point. The following settings are possible: • Trigger • First measurement point following the trigger event • First measurement point • Last measurement point
	"Offset"	Offset related to the time axis Moves the measurement left or right by the offset stated on the time axis. The offset can also be transferred via the clipboard to the cell from the ΔX value of the measurement cursor. See Align measurements precisely (overlay measurements) (Page 6555).
	“Time stamp"	Display of the trigger time
	"Comment"	Display and input option for a comment about the signal

Shortcut menu commands

The following table shows the shortcut menu commands of the signal table:

Shortcut menu command	Description
"Cut"	Cannot be selected.
"Copy"	Copies the contents of the selected lines to the clipboard.
"Paste"	Cannot be selected.
"Delete"	Cannot be selected.
"Rename"	Switches the selected cell to the editing mode.

Shortcut menu command	Description
"Import measurement"	Imports a measurement from a file, e.g. with the "*.ttrecx" file extension. For compatibility reasons, the "*.ttrec" file extension will be supported in V12, but does not contain any information about the device family.
"Export measurement"	Exports a measurement as a file with the extension "*.ttrecx" or "*.csv". For compatibility reasons, the "*.ttrec" file extension will be supported in V12, but does not contain any information about the device family.

User interface - Signal selection tab (overlay measurements)

User interface - Signal selection (overlay measurements)

The Signal selection tab shows the signals for all measurements and allows signals that are presented in the signal table of the diagram to be preselected.

Setting options and displays in the Signal selection tab.

The following figure shows an example of the display in the TIA Portal:

		Availa...	Measurement	Name	Data type	Address	Comment
1		☑	Trace-Curves	"Trace-Data".Angle	Int		
2		☑	Trace-Curves	"Trace-Data".Cos	Real		
3		☑	Trace-Curves	"Trace-Data".Sin	Real		
4		☐	Trace-Curves	"Trace-Data".Tan	Real		
5		☑	Trace-Curves	"Clock_10Hz"	Bool	%M100.0	
6		☑	Trace-Curves	"Clock_5Hz"	Bool	%M100.1	

The following table shows the settings and displays for the table:

Column	Description
	Static display of the signal icon
"Available in the diagram"	Selection for the display in the curve diagram When the selection is activated the signal is transferred to the signal table for the curve diagram.
“Measurement”	Display of the measurement to which the signal belongs
“Name”	Display of the signal name
“Data type “	Display of the data type
"Address"	Display of the address (not for symbolic tags)
"Comment"	Display of a comment on the signal

You will find further information on the specific settings in User interface - signal table (Page 6529).

Shortcut menu commands

The following table shows the shortcut menu commands for the signal selection:

Shortcut menu command	Description
"Copy"	Copies the content of the selected lines to the clipboard.
“Display selection in the signal table”	The selected signals are displayed in the signal table and are available in the diagram.
“Remove selection from the signal table”	The removed signals are not available in the diagram.

Several objects can be selected.

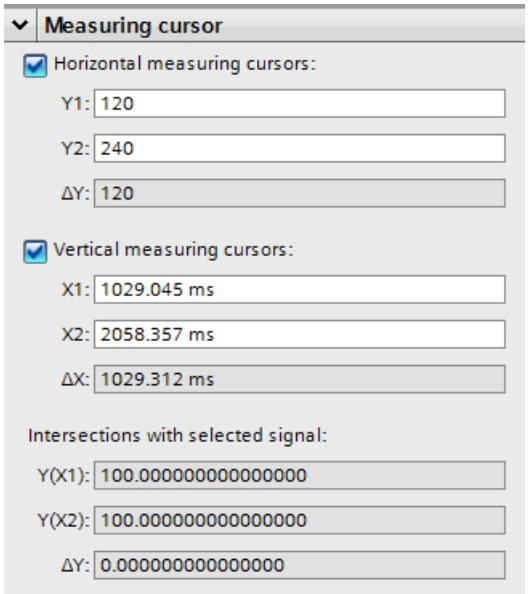
14.6.3.3 Trace task card

User interface - Measurement cursor pane

The “Measurement cursor” pane shows the position of the measurement cursor in the curve diagram and the values at the intersection points.

Setting options and displays of the "Measurement cursor” pane

The figure below shows the "Measurement cursor” pane:



The following table describes the settings and displays:

Setting/display		Description
Horizontal measurement cursor		
	Y1	Position of first measurement cursor The value states the position in relation to the scale of the signal currently selected. You also have the option of specifying a new position for the measurement cursor in this entry field for moving with the mouse.
	Y2	Position of the second measurement cursor The value states the position in relation to the scale of the signal currently selected. You also have the option of specifying a new position for the measurement cursor in this entry field for moving with the mouse.
	ΔY	Display of the position difference between the first and the second measurement cursor
Vertical measurement cursor		
	X1	Position of first measurement cursor You also have the option of specifying a new position for the measurement cursor in this entry field for moving with the mouse.
	X2	Position of the second measurement cursor You also have the option of specifying a new position for the measurement cursor in this entry field for moving with the mouse.
	ΔX	Display of the position difference between the first and the second measurement cursor
Intersection points with selected signal		
	Y(X1)	Display of the value at the position of the first measurement cursor
	Y(X2)	Display of the value at the position of the second measurement cursor
	ΔY	Display of the value difference between the first and the second measurement cursor

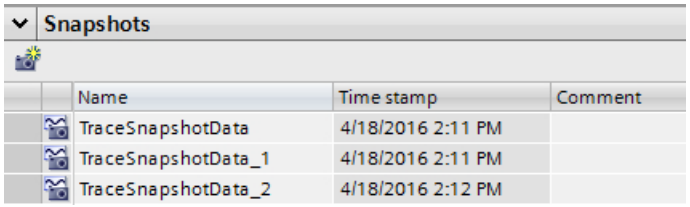
User interface - Snapshots pane

The “Snapshots” pane allows the user to save and restore different views for a measurement.

A snapshot is taken of the current view in the “Diagram” tab. The snapshots are saved in the measurement with the project.

Setting options and displays of the “Snapshots” pane

The figure below shows the “Snapshots” pane:



The following table shows the functions of the buttons:

Icon	Description
	Generate snapshot of the current view Saves the current view as a snapshot in the “Diagram” tab.

The following table shows the settings and displays:

Column	Description
	Static display of the snapshot icon
“Name”	Display and change options for the name
“Time stamp"	Display of the snapshot generation time
"Comment"	Display and input option for a comment

Several rows can be selected and deleted.

Double-clicking on a row opens the measurement with the saved view in the “Diagram” tab.

Shortcut menu commands

The following table shows the shortcut menu commands of the table:

Shortcut menu command	Description
“Restore snapshot”	Shows the measurement with the saved view in the “Diagram” tab.
"Delete"	Deletes the snapshot
"Rename"	Switches the cell to the editing mode

Several rows can be selected and deleted.

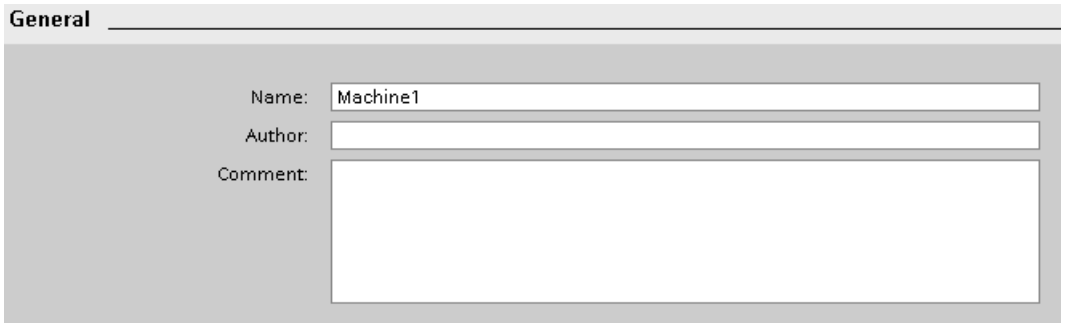
14.6.3.4 Inspector window

User interface - General

The "General" area shows the name of the trace configuration and input fields for the author and a comment.

Input options and displays in General

The following figure shows an example of the display in the TIA Portal:



The following table shows the input fields and displays:

Column	Icon	Description
"Name"	-	Name of the trace configuration.
"Author"	-	Author of the trace configuration
"Comment"	-	Input field for a comment. You can enter a line break with <Shift+Return>.

14.6.4 Operation

14.6.4.1 Quick start

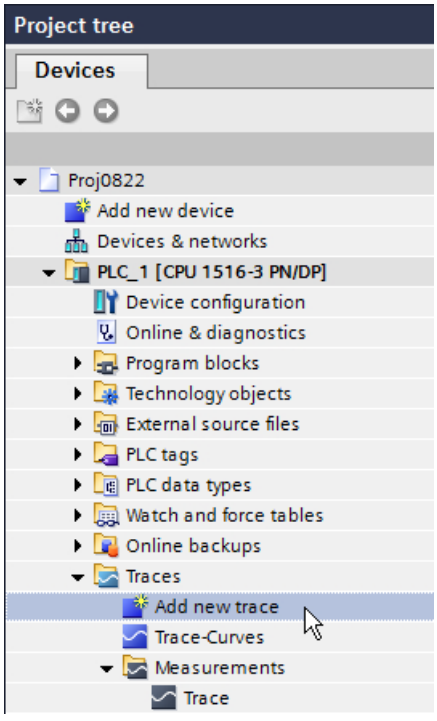
This description shows the steps for a recording of the S7-1500 CPU as an example. The displayed settings can differ depending on the device.

Requirement

A device is configured that supports the trace and logic analyzer function.

Creating a trace

The following figure shows the project tree with the "Traces" system folder below the device:

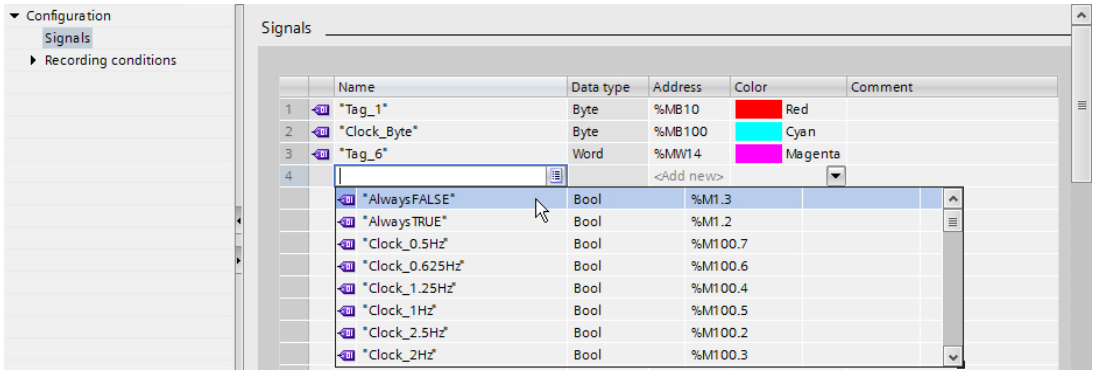


Procedure:

1. Double-click the "Add new trace" entry.
   A new trace configuration is created and the “Configuration" tab opens in the working area.
2. Adapt the name of the trace configuration by clicking the text.

Selecting signals

The following figure shows the configuration of the signals:

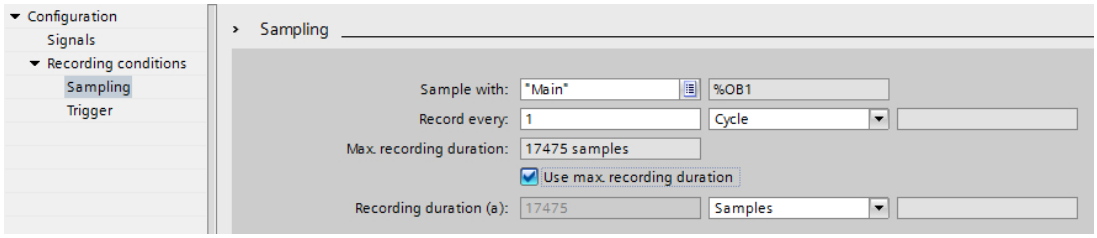


Procedure:

1. Select the signals to be recorded in the "Signals" area.
   Or:
2. Drag one or more signals, e.g. from a tag table, and drop them in the signal table.

Configuring the recording cycle

The following figure shows the configuration of the sampling:

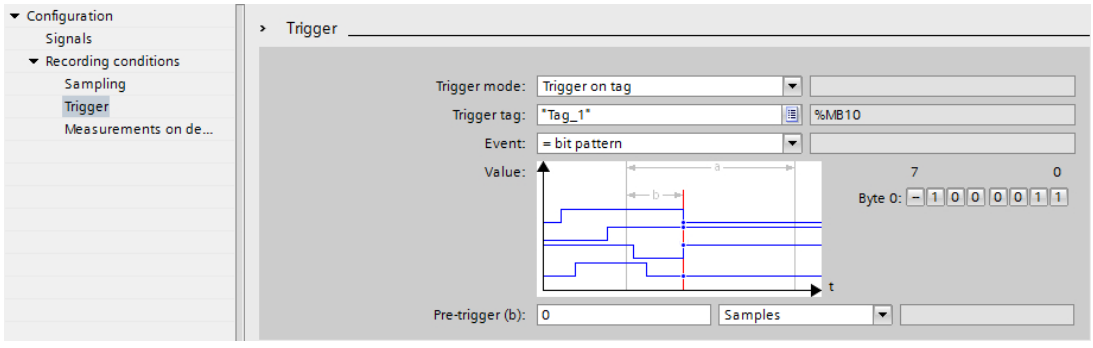


Procedure:

1. Configure the sampling.

Configuring the trigger

The following figure shows the configuration of the trigger:



Procedure:

1. Configure the trigger mode and the condition for the selected trigger.

Transferring the trace configuration to the device

Procedure:

1. Transfer the trace configuration to the device with the button.
   The following functions are executed:
   - An online connection is established to the device.
   - The trace configuration is transferred to the device.
   - The monitoring is activated.
   - The display switches to the "Diagram" tab.

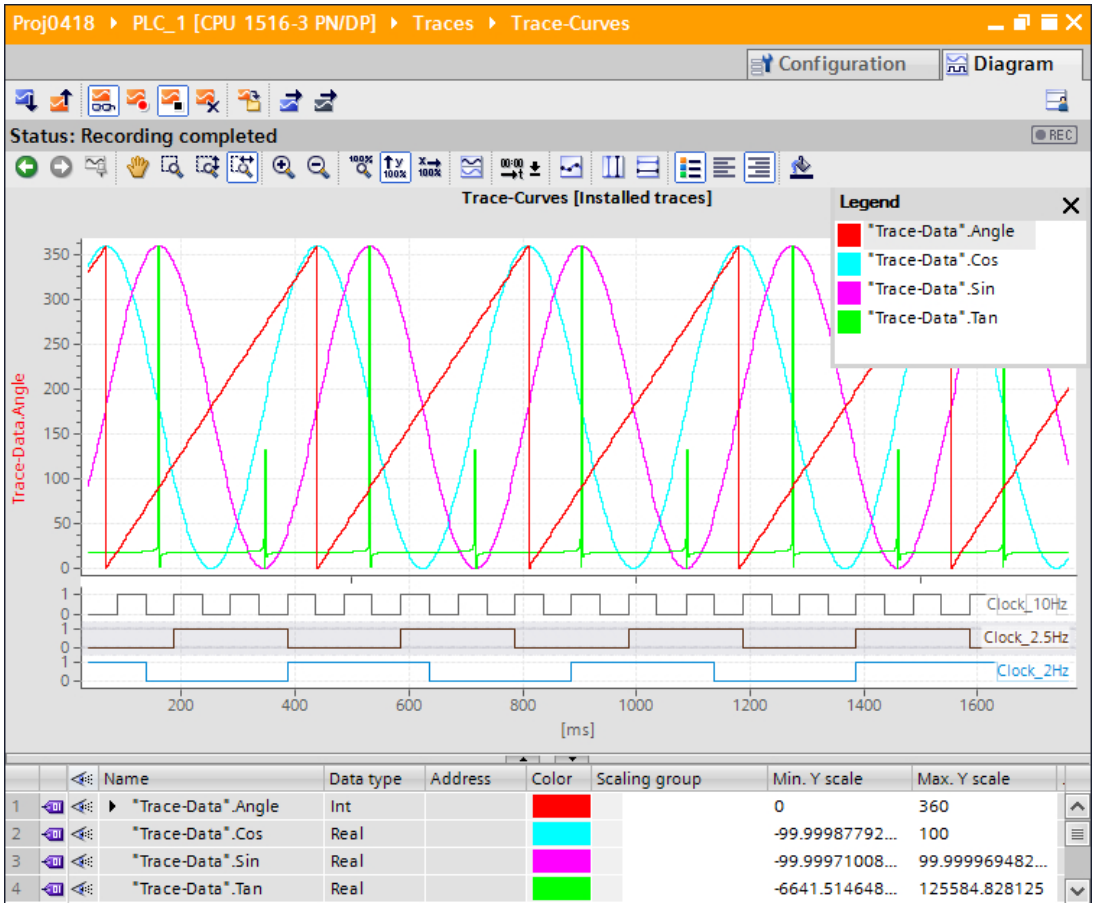
Activating a recording

Procedure:

1. Click the button.

Displaying the recording

The following figure shows the curve diagram with a recording:



Procedure:

1. Wait until the "Recording" or "Recording completed" status is displayed in the status display of the trace.
2. Switch to the "Diagram" tab.
3. Click the icon of a signal in the signal table.
   The individual bits of the signal are offered for display as a bit track.
4. In the signal table, select or deselect the individual signals and bits for display with the icon.

Saving the measurement in the project

Procedure:

1. Transfer the measurement to the project with the button.
   The measurement is displayed in the project tree under the "Measurements" system folder.

See also

User interface - trace toolbar (Page 6524)

14.6.4.2 Using the trace function - overview

Requirement

A device is configured in the TIA Portal that supports the trace and logic analyzer function and to which an online connection has been established.

Procedure

The following table shows a procedural overview with typical steps when working with the trace and logic analyzer function.

Step	Description
1	Creating a trace (Page 6542)
2	Configuring the trace (Page 6549)
3	Transferring the trace configuration to the device (Page 6544)
4	Activating/deactivating an installed trace (Page 6545)

Step	Description
5	Monitoring the recording (Page 6546)
6	Saving measurements in the project (Page 6547)
7	Displaying the recording (Page 6546)
8	Analyze an ongoing recording (Page 6546)
9	Compare records (overlay measurements) (Page 6554)

See also

Displaying a configuration (Page 6542)

14.6.4.3 Project tree

Creating a trace

Traces can be created in the form of trace configurations in the project navigator.

The following instructions describe how you can create a trace configuration under the "Traces" system folder

Procedure

To create a trace configuration, proceed as follows:

1. Double-click the "Add new trace" entry.

A new trace configuration is created.

Displaying a configuration

Requirement

A trace configuration, an installed trace, a measurement or an overlay measurement is available in the "Traces" system folder.

Procedure

To display a trace configuration, proceed as follows:

1. Double-click the appropriate icon of a trace configuration, an installed trace, a measurement or an overlay measurement in the project tree.
   The "Configuration" or "Diagram" tab opens in the working area.
2. If required, click the "Configuration" tab for the display.

Note

Write protection

The configuration data of an installed trace and in all measurements is displayed with write protection.

See also

User interface - “Traces” project tree folder (Page 6519)

Displaying a diagram

Requirement

An installed trace, a measurement or an overlay measurement is available in the "Traces" system folder.

Procedure

To display a diagram, proceed as follows:

1. Double-click the appropriate icon of an installed trace, a measurement or an overlay measurement in the project tree.
   The "Configuration" or "Diagram" tab opens in the working area.
2. If required, click the "Diagram" tab for the display.

See also

User interface - “Traces” project tree folder (Page 6519)

Apply overlay measurement

Overlay measurements can be applied in the project tree with a comparison function for different measurements.

The following instructions describe how you can create an overlay measurement under the “Overlay measurements" system folder

Requirement

A device is configured that supports the trace and logic analyzer function.

Procedure

To apply an overlay measurement, proceed as follows:

1. Select one or more measurements in the “Measurements" system folder.
2. Drag the measurements to the “Overlay measurements” system folder.

A new overlay measurement is created.

Configuring objects in groups

You can set up groups in system folders in the project navigator. Use this option to configure the view for multiple objects.

The following instructions use measurements as an example in order to describe how to consolidate the measurements into groups. The same functionality is also available for the Traces and Overlay measurements system folders.

Requirement

Measurements are available in the “Measurements" system folder.

Procedure

Proceed as follows to configure measurements into groups:

1. Select the shortcut menu command “Add new group” by right-clicking on the “Measurements" system folder.
   A new group folder is created.
2. Assign a meaningful name to the new group.
3. Repeat step 1 until all required groups have been created.
   (Sub-groups (groups within groups) can also be created.)
4. Drag & drop the corresponding measurements to the group folder that has been created.

14.6.4.4 Working area - general

Transferring the trace configuration to the device

Requirement

- A valid trace configuration is in the "Traces" system folder.
- The maximum number of installed traces has not been reached yet.

Procedure

To transfer a trace configuration to the device, proceed as follows:

1. Open a valid trace configuration in the working area.
2. Click the button.

Result

The trace configuration is transferred to the device.

Activating/deactivating an installed trace

Requirement

- There is an online connection to the device.
- There is a trace in the device.
- The installed trace is displayed in the working area.
- The button is activated for viewing the displayed trace.

Activating an installed trace

To activate the recording for an installed trace, proceed as follows:

1. Click the button.
   The installed trace is activated and starts the recording according to the configured trigger condition. The trigger condition is device-specific and described in Section "Configuration" below the respective device (Page 6557).
   The current status of the recording is displayed in the status display of the trace.

   Note

   When a recording is restarted, the previously recorded values are lost.

   To save the recorded values, save the measurement in the project (Page 6547) before you activate the recording again.

Deactivating an installed trace

To deactivate an activated installed trace, proceed as follows:

1. Click the button.
   The installed trace is deactivated.
   The status display of the trace changes to "Inactive".

Displaying the recording

Requirement

- There is an online connection to the device.
- There is a trace with recording in the device.

Or:

- A measurement is in the "Measurements" system folder.

Procedure

To display the recording, proceed as follows:

1. Select an installed trace.
2. Double-click the selected trace.
3. If required, activate the button for viewing.

Or:

1. Select a measurement in the "Measurements" system folder.
2. Double-click the selected measurement.

Result

The recording is displayed in the "Diagram" tab.

See also

User interface - “Traces” project tree folder (Page 6519)

Analyze an ongoing recording

Requirements

- An ongoing recording is displayed in the "Diagram" tab.

Save the data currently recorded as a measurement

To analyze a certain time range for an ongoing recording, follow these steps:

1. Click the button.
   The data recorded up to now is added to the measurements.
   The current recording is not affected by this and continues running uninterrupted.

Analyze the measurement

To display the saved measurement, follow these steps:

1. In the "Measurements" system folder double-click the measurement that has just been saved in order to open it
   The “Diagram” tab for the measurement opens in the working area.

See also

Displaying the recording (Page 6546)

Working area - Diagram tab (Page 6550)

Saving measurements in the project

Requirement

- There is an online connection to the device.
- There is a trace with recording in the device.
- The installed trace data must have been displayed at least once in the curve diagram. The recording data is loaded from the device for the display.

Procedure

To save a recording in the project, proceed as follows:

1. Open the installed trace with the recorded data.
2. If required, make sure that the current data is loaded from the device by activating the button.
3. After activating the button wait until all data has been loaded and displayed.
4. Click the button.
   The measurement is added to the "Measurements" system folder.
5. Save the project in the TIA Portal.

Note

Generate measurements

A measurement of an installed trace can be generated at any time.

Use this functionality e.g. to save the data recorded up until this point in a recording and to analyze it as a static measurement.

Exporting and importing measurements

Requirement

At least one measurement is in the "Measurements" system folder for export.

Exporting measurements

To export a measurement, proceed as follows:

1. Right-click a measurement in the "Measurements" system folder and select the shortcut menu command “Export measurement".
2. Select a folder, a file name and a data type to save the measurement.
3. Click the "Save" button.

Importing measurements

To import a measurement, proceed as follows:

1. Right-click in the "Measurements" system folder and select the shortcut menu command "Import measurement".
2. Select the file e.g. of the "*.ttrecx" file type with the measurement to be imported.
3. Click the "Open" button.
   The imported measurement is displayed with the file name in the "Measurements" system folder.

Note

Export and import trace configurations

The same functionality is available for exporting and importing trace configurations.

Transferring the trace configuration from the device to the project

Requirement

- There is an online connection to the device.
- There is a trace in the device.

Procedure

To transfer a trace configuration to the project, proceed as follows:

1. Open an installed trace.
2. If required, activate the button for viewing.
3. Click the button to transfer the trace configuration from the device.

Result

The configuration is taken over as new trace configuration in the "Traces" system folder.

A trace configuration of the same name is overwritten in the system folder.

Deleting installed traces

Requirement

- There is an online connection to the device.
- There is a trace in the device.

Procedure

To delete an installed trace, proceed as follows:

1. Open an installed trace.
2. If required, activate the button for viewing.
3. Click the button.
   A confirmation prompt opens.
4. Confirm the prompt for deletion.

Or

1. Select one or more installed traces / in the project tree.
2. Press <Del> to delete the installed traces.
   A confirmation prompt opens.
3. If required, select an option for deletion and confirm the prompt.

14.6.4.5 Working area - Configuration tab

Configuring the trace

Requirement

The "Configuration" tab is open in the working area.

Configuring the trace

In the configuration, you specify the recording and trigger conditions and select the signals to be recorded.

See Section "Configuration" below the respective device (Page 6557).

Note

Saving the trace configuration

You save the trace configuration with the project in the TIA Portal.

If you close the project without saving, the configuration is discarded.

See also

Displaying a configuration (Page 6542)

14.6.4.6 Working area - Diagram tab

Use of the curve diagram

The curve diagram shows the signals of a recording selected in the signal table.

The display area can be zoomed as required. Measurement cursors can be used to select individual values for display in the signal table.

The following operating instructions describe the use of the curve diagram and of the measurement cursors as examples.

Requirements

- An installed trace or a measurement has been selected for display.
- The  button is activated to monitor an installed trace.
- The "Diagram" tab is open in the working area.

Monitor an ongoing recording.

To display all of the data in an ongoing recording, proceed as follows:

1. Activate “Display all" via the  button.

The entire time range and all values for the ongoing recording are displayed.

To display a consistent time window in an ongoing recording, proceed as follows:

1. Activate “Display all" via the  button.
2. Select the desired time range via the  button.

The trend view is updated while the scaling of the time range remains the same.

Evaluation of a certain instant of a recording

To display the values for a specific sample, proceed as follows:

1. Display the vertical measurement cursors via the button.
2. Move a measurement cursor with the mouse to the required position in the recording. The values of the signals are displayed in the signal table and in the “Measurement cursor” pane of the “Trace” task card.

Evaluation of the difference between two samples

To display the difference, proceed as follows:

1. Display the vertical measurement cursors via the button.
2. Move both measurement cursors with the mouse to the required samples in the recording. The values of the signals and the difference are displayed in the signal table and in the “Measurement cursor” pane of the “Trace” task card.

Using horizontal measurement cursors

To check whether a certain value has been reached, proceed as follows:

1. Display the horizontal measurement cursors via the button.
2. Move a measurement cursor with the mouse to the required value of the recording. The values of the measurement cursors for the selected signal are displayed in the “Measurement cursor” pane of the “Trace” task card.

Moving the time range displayed

Proceed as follows to move the time range displayed:

1. Select a time range via the button.
2. Move the curve to the desired time range by turning the mouse wheel with the <Shift> key pressed down.

Bringing a signal into the foreground

1. Display the legend via the button.
2. Click a signal in the legend.

Or:

1. Click a signal in the curve diagram.

The signal is displayed in the foreground and is highlighted/selected in the signal table. The value axis is updated for the selected signal.

See also

Displaying a diagram (Page 6543)

User interface - curve diagram (Page 6525)

User interface - signal table (Page 6529)

Use of the signal table

The signal table shows the signals of an installed trace or a measurement. The preselected signals in the signal selection are displayed with an overlay measurement. You can show or hide individual signals for the display in the table and adapt the properties for the display. Individual bits can be selected for some data types and displayed as a bit track.

The following operating instructions describe the operation of the signal table.

Requirements

- An installed trace or a measurement has been opened in the "Diagram" tab.
- The  button is activated to monitor an installed trace.
- For the display of individual bits as a bit track:
  at least one recorded signal supports the display as a bit track.

Display or hide individual signals and change the color

To adapt the display to suit your requirements, proceed as follows:

1. Click the icon of the respective signal in the  column to select or deselect it for the display.
2. Click in the "Color" column for the respective signal and select a color.
   The default color for the signal changes.

Bringing a signal into the foreground

1. In the signal table, select the line of the signal.
   The Y-scale of the signal is displayed.
   The signal curve is brought into the foreground in the curve diagram.

Selecting individual bits for display as a bit track

To display individual bits as a bit track in the lower curve diagram, proceed as follows:

1. Click the icon of a signal in the signal table.
2. Click the  icon in the open bit selection of the signal.
   The bits are selected or deselected for display.

See also

Displaying the recording (Page 6546)

Using the signal group in the signal table

Individual signals can be scaled identically in a signal group, which makes it easier to compare the curve characteristics.

Binary signals cannot be grouped.

The following operating instructions describe how to work with the signal group.

Note

Saving signal groups

The signal groups can be saved individually for each measurement via the "Use current view as standard" function (button).

If the signal groups and the project are not saved then the signal groups created will be lost when the "Diagram" tab is closed.

Requirements

- An installed trace or a measurement is displayed.
- The button is activated to monitor an installed trace.
- The "Diagram" tab is open in the working area.
- There are at least two signals in the signal table that are not of the BOOL type.

Assigning signals to a signal group

To apply a signal group and assign signals to this group, proceed as follows:

1. In the signal table, select the line or cell of the required signal.
2. Click the gray field in the "Signal group" column.
   The sequence icon is displayed in the gray field and the name of the signal group is pre-assigned: Group
3. Click the gray fields of further signals that are to be assigned to this signal group.

Or:

1. Click in the text field of the "Signal group" column for a signal to be grouped.
2. Enter a name for the group.
3. Enter the same group name in the respective text fields for further signals or select the group name via the drop-down list.

The Y-scales of the grouped signals are scaled with the values of the signal that was selected first. Changes to a scale value always affect the entire group.

Removing signals from a signal group

To delete the assignment of a signal to a signal group, proceed as follows:

1. Click the sequence icon for the required signal in the "Signal group" column.

Or:

1. Click the text field for the required signal in the "Signal group" column.
2. Press the <Del> key.

Or:

1. Select the respective text field in the "Signal group" column for several signals using the <Shift> and <Ctrl> keys.
2. Press the <Del> key.

The signals are removed from the signal group or the signal group is deleted.

Compare records (overlay measurements)

Requirement

- An overlay measurement is created or is created implicitly by dragging the measurements to the system folder “Overlay measurements”.
  See also AUTOHOTSPOT.

Adding measurements for comparison

To compare measurements, insert the measurements to be compared to the overlay measurements. Proceed as follows for this:

1. In the project tree drag one or more measurements from the system folder “Measurements" to the icon for the overlay measurement.

Or:

1. Import saved measurements via the “Import measurement” shortcut menu command.

A copy of the measurements is added to the overlay measurement.

Note

Changes to the settings for measurements within the overlay measurement have no impact on the original measurements. The original measurements remain unchanged.

Select signals of the measurements for the signal table

Proceed as follows to select the signals for the signal table in the “Diagram” tab:

1. Double-click on the icon for the overlay measurement in the project tree.
   The tabs for the overlay measurement will be displayed in the working area.
2. Click the “Signal selection" tab in the working area.
   The signals for all measurements are displayed in the table.
3. Activate or deactivate the signal check box for those signals that should be visible or should not be visible in the signal table.

The activated signals are displayed in the signal table of the “Diagram” tab.

Use of the signal table

Proceed as follows to open and use the signal tables:

1. Click the “Diagram" tab in the working area.
2. Click on the “Signals” tab within the “Diagram” tab.
3. Use the signal tables as described under AUTOHOTSPOT.

Align measurements

Proceed as follows to align the time axis for the measurements for the comparison:

1. Click on the “Measurements” tab within the “Diagram” tab.
2. Select the alignment for the measurements via the check box.
3. Adjust the alignment and if necessary set an offset for the alignment of the individual measurements.

The measurements are aligned with each other accordingly on the time axis.

(The precise alignment of two measurements is described in the next section.)

Use of the curve diagram

Proceed as follows to open and use the curve diagram:

1. Click the “Diagram" tab in the working area.
2. Use the curve diagram as described under AUTOHOTSPOT.

Align measurements precisely (overlay measurements)

Requirement

- An overlay measurement is applied.
- Measurements for comparison are added to the overlay measurement.

- Signals of the measurements for the signal table are selected.
- The “Diagram” tab for the overlay measurement opens in the working area.

Align measurements precisely with position difference ΔX

Proceed as follows to align the time axis for two measurements precisely:

1. Display the vertical measurement cursors via the button.
2. Zoom into the time range, e.g. with the button until you are able to position the first measurement cursor precisely on the desired reference point for the first measurement.
3. Move the first measurement cursor with the mouse to the required position.
4. Search for the reference point for the second measurement, e.g. by switching to “Display all” with the button.
5. Zoom into the time range, e.g. with the button until you are able to position the second measurement cursor precisely on the desired reference point for the second measurement.
6. Move the second measurement cursor with the mouse to the required position.
7. Open the “Trace" task card.
8. In the “Measurement cursor” pane select the position difference value ΔX.
9. Copy the value to the clipboard.
10. Insert the value from the clipboard into the Offset cell of the first or second measurement.

Both measurements are precisely aligned with each other at the desired measurement points.

Note

When inserting the position difference as the offset make sure that you also adjust the leading character as necessary.

Printing a recording

The curve diagram supports the saving of the display as a graphic and the copying of the display to the clipboard. Also use these functions (Page 6525) for printing.

14.6.5 Devices

14.6.5.1 S7-1200/1500 CPUs

Recordable variables

Device-dependent recording of tags

The following list shows the operand areas from which tags can be recorded:

- Process image input
- Process image output
- Bit memory
- Data blocks

Data types

Elementary data types can be recorded. The availability of the individual data types depends on the device used:

The following table lists the elementary data types:

Data types	Note
Binary numbers	
BOOL	-
Bit strings	
BYTE	-
WORD	-
DWORD	-
LWORD [1)]	Symbolic name required
Integers	
SINT	-
USINT	-
INT	-
UINT	-
DINT	-
UDINT	-
LINT [1)]	Symbolic name required
ULINT [1)]	Symbolic name required
Floating-point numbers	
REAL	-
LREAL	Symbolic name required

1) Not supported by S7-1200.

Lifetime of the installed trace configuration and recorded values

Installed trace configurations are retained after POWER OFF. The recording is activated again after the restart of the CPU.

Recorded values are lost during the restart.

Note

Downloading a configuration to the device in the "STOP" operating state

Note that after downloading a configuration in the "STOP" operating state, you must check the installed traces and, if required, reactivate them or transfer them again.

Note

If trigger tags that affect the address are changed, the trace configuration must also be transferred to the device again.

This is the case for example, when a data block is shortened or extended or the data type is changed.

Recording levels

The following list shows the execution levels that can be selected for the recording cycle:

- Program cycle - OB 1
- Time-of-day interrupt - OB 1x
- Time-delay interrupt - OB 2x
- Watchdog interrupt - OB 3x
- Synchronized processing cycles - OB 6x, not OB 60
- MC-PreServo - OB 67
- MC-Servo - OB 91
- MC-Interpolator - OB 92
- MC-PostServo - OB 95

Note

The measured values are recorded at the end of the OB after the processing of the user program.

Quantity structure

The following table shows the maximum quantity structure that can be recorded with the trace and logic analyzer function:

Device	Maximum number of installed traces	Maximum number of signals per trace configuration
S7-1200 (as of firmware version V4.0)	2	16
S7-1500	At least 4 (depending on the CPU type)	16

Example of CPU 1516-4 PN/DP

- Maximum of 3854 samples for 16 signals from PLC tags of the DWORD data type
- Maximum of 21844 samples for 16 signals from PLC tags of the BOOL data type
- Maximum of 58250 samples for 1 signal from a PLC tag of the BOOL data type

Further information can also be found in the form of FAQs under the ID 102781176 (http://support.automation.siemens.com/WW/view/en/102781176).

CPU load through trace recording

A trace recording increases the runtime of the respective recording level that can result in an execution level overflow with high utilization of the CPU.

Remedy for execution level overflow:

- **Change the trace configuration**
  1) Configure fewer tags and signals.
  2) Then increase the number of tags and signals up to the maximum number of signals step-by-step without an execution level overflow.
- **Select a slower recording level**

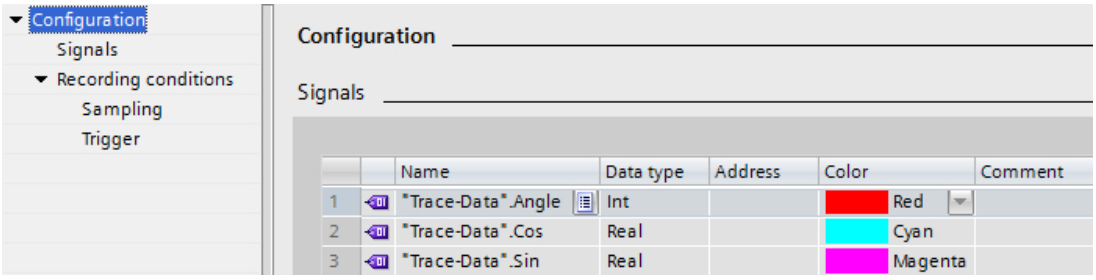
Software user interface of the configuration

Structure of the user interface

Display areas in the "Configuration" tab of the working area

The settings options differ depending on the configured device.

The following figure shows an example of the display in the TIA Portal:



The area navigation provides the following entries for selection:

- Configuration
  - Signals (Page 6560)
  - Recording conditions (Page 6561)

Displaying and changing properties of a trace configuration

A trace is selected in the project tree and displayed in the "Configuration" tab.

The trace configuration can only be changed offline and is displayed online with the write protection.

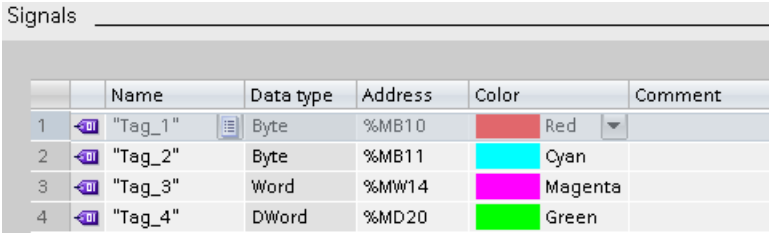
User interface - Signals

The "Signals" area shows a table in which the signals to be recorded are configured for the selected trace configuration.

Signals can also be inserted in the table using drag-and-drop.
The signals can be sorted using drag-and-drop.

Setting options and displays in "Signals"

The following figure shows an example of the display in the TIA Portal:



The following table shows the settings and displays:

Column	Icon	Description
		Display of the signal icon for a selected signal.
"Name"	-	Input field for the name or address of the signal. **Examples:** • “Data_block_1”.pressure • M0.0 • DB1.DBW3
-		Button to open the signal selection table. The button is displayed when the table line is selected. Clicking the icon opens a table which offers possible signals for selection. The selected signal is displayed in the input field.
"Data type"	-	Text field with display of the data type for the signal.
"Address"	-	Input field for the address of the signal. The field remains empty with optimized / type correct tags.
"Color"	-	Text field for display and selection of the color. Click the signal color to open the color selection dialog.
"Comment"		Input field for a comment on the signal.

Shortcut menu commands

The following table shows the shortcut menu commands of the table:

Shortcut menu command	Description
"Cut"	Cannot be selected.
"Copy"	Copies the contents of the selected lines to the clipboard.
"Paste"	Pastes the contents of the clipboard to the selected line. The existing contents are overwritten.
"Delete"	Deletes the selected lines from the table or deletes the content of the selected cell.
"Rename"	Switches the selected cell to the editing mode.

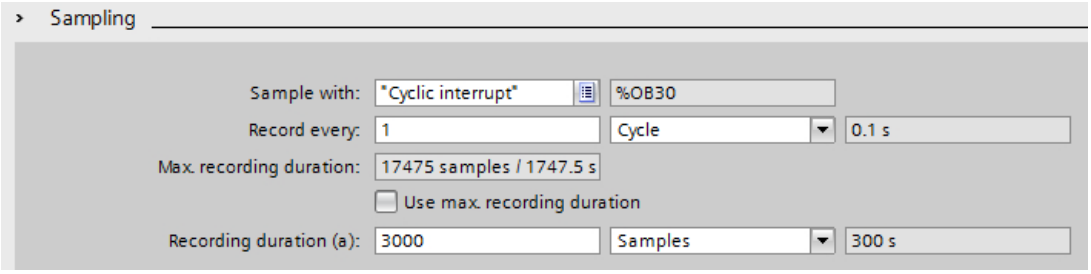
Recording conditions

User interface - Recording conditions

The "Recording conditions" area shows the trigger condition for the selected trace configuration and in which cycle, how fast and how long the recording is made. Configuration is possible when the trace configuration is displayed in offline mode or online mode with deactivated viewing .

Setting options and displays in "Recording conditions"

The following figure shows an example of the settings for the sampling in the TIA Portal:

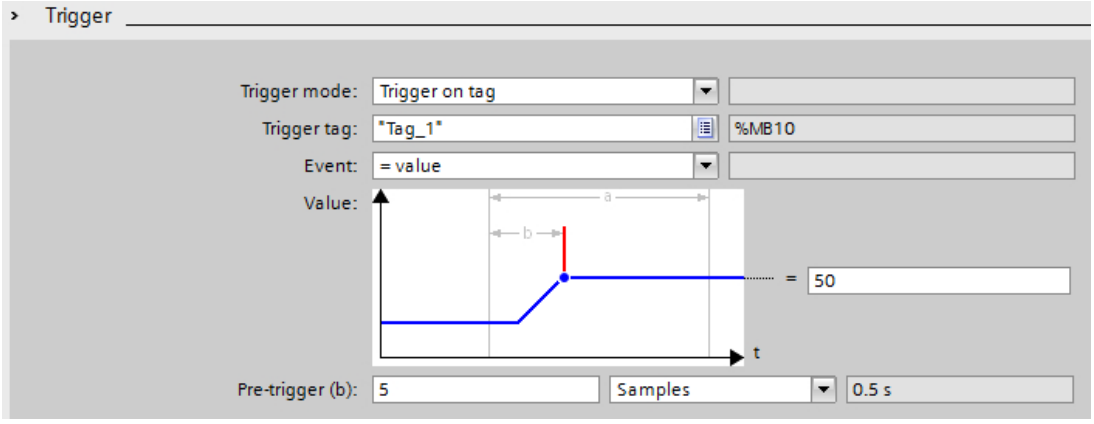


The following table explains the settings and displays:

Setting/display		Description
"Recording time"		
	Recording level entry field	Selection of the recording time. See Recording levels (Page 6558)
	Address of the OB text field	Detailed information on the selected recording time.
"Record every"		
	Reduction entry field	Input of the reduction in relation to the reduction ratio and the unit.
	Reduction ratio drop-down list	Selection of the reduction ratio unit The following settings are possible: • "Cycle" • "s" (setting option depends on the recording level selected under “recording time”)
	Sampling time text field	Display of the sampling time, taking into account the configured reduction and the selected unit (only for constant bus cycle time OBs).
"Max. recording duration"		
	Max. recording duration text field	Displays the calculated maximum recording duration. The "Max. recording duration" depends on how many signals are recorded and the data type of these signals.
	"Use max. recording duration"	Set the recording duration to the maximum value. When the checkbox is activated, the recording duration is set to the maximum possible recording duration. The set reduction in the "Record every" input field is taken into account. The recording duration is also adapted when additional signals are added. Further information can also be found in the form of FAQs under the ID 102781176 (http://support.automation.siemens.com/WW/view/en/102781176).
"Recording duration"		
	Recording duration entry field	Input of the recording duration in relation to the selected unit. If the "Recording duration = max. recording duration" checkbox is activated, entries are overwritten by the value displayed in "Max. recording duration".

Setting/display		Description
	Unit drop-down list	Selection of the unit for the recording duration. The following settings are possible: • "Samples" The maximum number of samples recorded is the number for which parameters are assigned under recording duration. • "s" (setting option depends on the recording level selected under “recording time”)
	Calculated recording duration text field	Display of the calculated recording duration (only for constant bus cycle time OBs)

The following figure shows an example of the settings for the trigger in the TIA Portal:

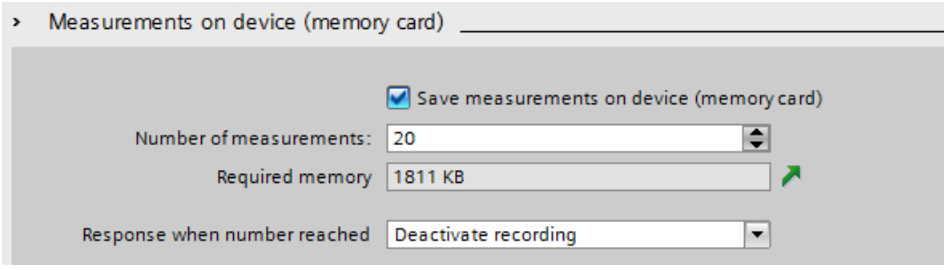


The following table explains the settings and displays:

Setting/display		Description
"Trigger mode"		Selection of the trigger mode.
	Trigger mode drop-down list	The following settings are possible: • "Record immediately" The recording is made immediately after the device is activated. • "Trigger on tag" The recording is made as soon as the installed trace is activated and the configured trigger condition is fulfilled.
	Text field	-
"Trigger tag"		The "Trigger tag" specifies a signal that triggers the recording.
	Trigger tag entry field	Enter a signal. **Examples:** • "DataBlock_1".Temperature • M0.0 • DB1.DBW3 See also Data types for trigger tags (Page 6566).

Setting/display			Description
			Opens the signal selection table. Clicking the icon opens a table which offers possible signals for selection as trigger tag. The selected signal is displayed in the input field.
	Trigger tag address text field		Display of the trigger tag address. With purely symbolic signals the field remains empty.
"Event"			The events that can be used on this trigger tag are offered for selection according to the data type of the trigger tag. The event can only be configured when a valid signal has been entered as trigger tag.
	Trigger events drop-down list		Event selection for which the trigger tag is checked. The entries in the drop-down list are described in Section Trigger event (Page 6566).
	Text field		-
"Value"			Configuration of the selected event. The configuration options differ depending on the format of the trigger tag and the selected event. SeeTrigger event (Page 6566).
"Pre-trigger"			"Pre-trigger" defines the number of samples that are already recorded before the actual trigger condition is fulfilled. If the trigger event occurs immediately or shortly after the activation of the recording, this may result in a shorter recording duration. Examples of "Recording duration (a)" = 20 samples and "Pre-trigger (b)" = 5 samples: • Case 1: Trigger event occurs 50 samples after activation of the recording Actual recording duration (a) = 20 samples • Case 2: Trigger event occurs 2 samples after activation of the recording Actual recording duration (a) = 17 samples
	Duration entry field		Input of the duration in relation to the selection in the drop-down list.
	Unit drop-down list		Selection of the unit The following settings are possible: • "Samples" • "s" (setting option depends on the recording level selected under “recording time”)
	Resulting pre-trigger duration text field		Display of the calculated "Pre-trigger" duration (only for recording in constant bus cycle time OBs)

The following figure shows an example of the settings for the saving of installed measurements:



Note

Available memory in the device (memory card)

The memory in the device (memory card) is partly used by system-relevant functions or reserved for that purpose.

Thus it is not possible for the entire memory to be used for saving measurements.

For further information please refer to the Function Manual Structure and Use of the CPU Memory (https://support.industry.siemens.com/cs/us/en/view/59193101).

Note

Memory requirements upon restart

Following a device reboot the maximum number of measurements saved in the device is the number configured under “Number of measurements”.

With repeated restarts note that the measurements already saved are not overwritten and the “Number of measurements” configured in the device is saved once again.

The following table explains the settings and displays:

Setting/display	Description
“Saving of installed measurements (memory card)”	Repeat measurement automatically and store in the device retentively This setting is only possible with “Trigger on tag” trigger mode. The measurements are stored on the “primary” memory card. **Note** Only completed measurements are saved on the device. Any recording deactivated by the user is not saved on the device. **This function is available with the following firmware versions:** • S7-1200 as of V4.2 • S7-1500 as of V2.0
“Number of measurements”	Input of the number of measurements to be saved on the card.
“Memory requirements”	Display of the expected memory requirement for all measurements

Setting/display	Description
	Displaying memory usage Shows the tab with the memory usage
“Behavior if number reached”	Selection for the behavior once “Number of measurements” is reached The following settings are possible: • “Deactivating a recording” The measurements are repeated until the “Number of measurements on the card” is reached. • “Overwrite oldest recording” The measurements are saved in a ring buffer and repeated until the user deactivates the recording. Once the number of measurements exceeds the “Number of measurements on the card” the oldest measurement on the card is overwritten in each case. **Note** Note that write processes that are repeated frequently may damage the card.

Data types for trigger tags

The following table shows the supported data types for the trigger tag:

Memory requirement and format of the number	Data type
1 byte	BOOL
8-bit integers	SINT, USINT, BYTE
16-bit integers	INT, UINT, WORD
32-bit integers	DINT, UDINT, DWORD
64-bit integers	LINT, ULINT, LWORD (not S7-1200)
32-bit floating-point numbers	REAL
64-bit floating-point numbers	LREAL

Trigger event

Depending on the selection in the drop-down list, the further settings differ for the "event".

The individual events are described below.

"=TRUE"

Supported data types: Bit (Page 6566)

The recording starts when the state of the trigger is TRUE.

"=FALSE"

Supported data types: Bit (Page 6566)

The recording starts when the state of the trigger is FALSE.

"Rising edge"

Supported data types: Bit (Page 6566)

The recording is started when the trigger state changes from FALSE to TRUE.
After activation of the installed trace, at least two cycles are required to identify the edge.

"Rising signal"

Supported data types: Integers and floating-point numbers (Page 6566)

The recording is started when the rising value of the trigger reaches or exceeds the value configured for this event.
After activation of the installed trace, at least two cycles are required to identify the edge.

"Falling edge"

Supported data types: Bit (Page 6566)

The recording is started when the trigger state changes from TRUE to FALSE.
After activation of the installed trace, at least two cycles are required to identify the edge.

"Falling signal"

Supported data types: Integers and floating-point numbers (Page 6566)

The recording is started when the falling value of the trigger reaches or falls below the value configured for this event.
After activation of the installed trace, at least two cycles are required to identify the edge.

"In the range"

Supported data types: Integers and floating-point numbers (Page 6566)

The recording starts as soon as the value of the trigger is in the value range configured for this event.

"Outside of the range"

Supported data types: Integers and floating-point numbers (Page 6566)

The recording starts as soon as the value of the trigger is outside the value range configured for this event.

"Value change"

All data types are supported.

The value is checked for change when the recording is activated. The recording starts when the value of the trigger changes.

This trigger event is supported as of V13 SP1. Older versions of the TIA Portal cannot interpret the trigger. Note that no explicit information is output in this case. This can occur, for example,

when the trace is transferred from a CPU to a TIA Portal less than V13 SP1 or a trace configuration is imported.

"= value"

Supported data types: Integers (Page 6566)

The recording starts when the value of the trigger is equal to the value configured for this event.

"<> "<> value"

Supported data types: Integers (Page 6566)

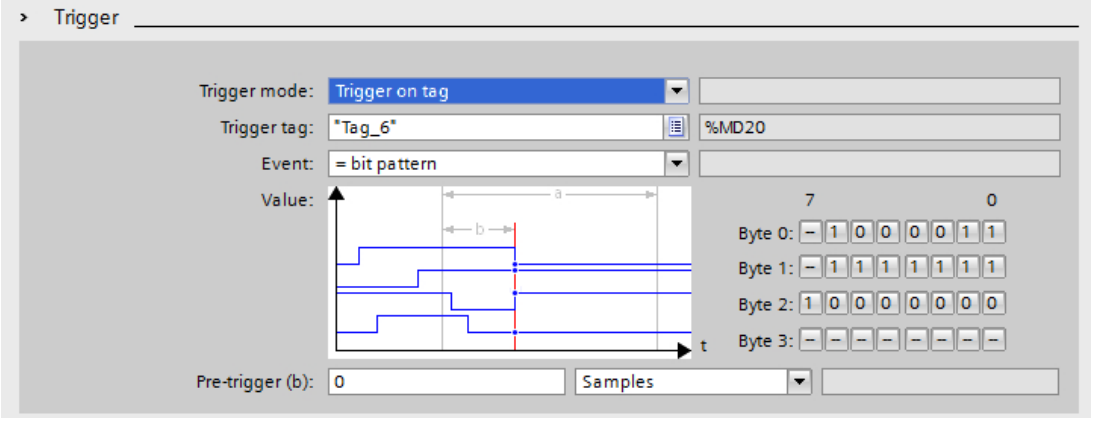
The recording starts when the value of the trigger is not equal to the value configured for this event.

"= bit pattern"

Supported data types: Integers (Page 6566)

The recording starts when the value of the trigger matches the bit pattern configured for this event.

The following figure shows the setting options for a "bit pattern":



It is possible to switch between the icons by clicking the respective button.

The following table shows the icons:

Icon	Description
–	Bit is not evaluated
0	Bit is checked for FALSE
1	Bit is checked for TRUE

"<> bit pattern"

Supported data types: Integers (Page 6566)

The recording starts when the value of the trigger does not match the bit pattern configured for this event.

See also

Configuring the trigger conditions (Page 6570)

Configuration

Trace configuration - overview

The configuration of the recording conditions and the signals to be recorded is device-specific.

Requirement

A trace configuration has been created and opened in the working area of the "Configuration" tab.

Procedure

The following table shows the procedure for configuring.

Step	Description
1	Documentation of the configuration (optional) Enter a comment and an author for the configuration in the Inspector window.
2	Selecting signals (Page 6569) Select the signals to be recorded in the "Signals" area.
3	AUTOHOTSPOT Select a recording time, a cycle and the duration in the "Recording conditions" area.
4	Configuring the trigger conditions (Page 6570) In the "Recording conditions" area, select whether the recording is to be performed immediately or depending on a trigger condition.
5	Configure installed measurements (memory card) (Page 6571) In the "Recording conditions" area, select whether the recording is to be saved on the device (memory card).

Selecting signals

Requirement

- A trace configuration has been created and opened.
- The "Signals" area is open in the "Configuration" tab.

Procedure

To configure the signals to be recorded, proceed as follows:

1. Select a signal. The following options are available:
   - In the "Name" column, click the button and select a tag.
   - Enter the symbolic tag name in the cell in the "Name" column.
   - Enter the address directly in the "Address" column.
   - Drag a signal to the table using drag-and-drop.
2. Click in the "Color" column and select a color for the display of the signal.
3. Click in the "Comment" column and enter a comment for the signal.
4. Repeat the procedure from step 1 until all the signals to be recorded have been entered in the table.

Configuring the recording cycle and duration

Requirement

- A trace configuration has been created and opened.
- The "Recording conditions" area is open in the "Configuration" tab.

Procedure

To configure the cycle and the duration of a recording, proceed as follows:

1. Click the button for the recording time.
2. Select an OB for the recording time (Page 6558).
3. Select a unit for the reduction factor in the drop-down list for "Record every".
4. Enter the factor for the reduction in the input field for "Record every".
5. Select a unit in the drop-down list for "Recording duration".
6. Specify the recording duration.
   The following options are available:
   - Enter a value for the duration in the input field for "Recording duration".
   - Activate the "Use max. recording duration" checkbox.

Configuring the trigger conditions

Requirement

- A trace configuration has been created and opened.
- The "Recording conditions" area is open in the "Configuration" tab.

"Record immediately" trigger condition

To start the recording immediately, proceed as follows:

1. Select the "Record immediately" entry in the drop-down list for "Trigger mode".
   The input fields for the trigger tag are hidden.

"Trigger on tag" trigger condition

To start the recording depending on a condition, proceed as follows:

1. Select the "Trigger on tag" entry in the drop-down list for "Trigger mode".
2. Select a trigger tag. The following options are available:
   - Click the button for the trigger tag and select a tag.
   - Enter the address or the symbolic name of the tag directly in the input field for the trigger tag.

   A drop-down list with events and input fields is displayed. The display depends on the data type of the tag.
3. Configure the event.
4. Select a unit for the pre-trigger in the drop-down list for "Pre-trigger".
5. In order to record a period before the trigger event, enter a value greater than 0 in the input field for the pre-trigger.

Note

Cyclic test of the trigger condition

The trigger condition is checked in every cycle irrespective of the setting in "Record every". To reliably identify the trigger, the trigger signal must be present for at least one full cycle.

Note

No evaluation of the trigger during saving

No new trigger can be evaluated provided that the recording is saved.

Configure installed measurements (memory card)

Requirement

- A trace configuration has been created and opened.
- The "Recording conditions" area is open in the "Configuration" tab.
- The “Trigger on tag” trigger mode is set.
- The firmware on the device supports the recording of an installed measurement.

Procedure

Proceed as follows to save the installed measurement (on the memory card):

1. Select the "Save measurements on device (memory card)" check box.
2. Enter the number of measurements that ought to be saved on the card in the “Number of measurements” entry field.
3. Set the desired behavior once the “Number of measurements” has been reached in the “Behavior if number reached” drop-down list.

14.7 Establishing a remote connection with TeleService

14.7.1 Basics of working with TeleService

14.7.1.1 Introduction to TeleService

Introduction

TeleService gives your controller telecommunication capability. You can manage, control and monitor distributed plants centrally by means of remote connections.

Scope of functions

TeleService allows you to use the range of TIA portal functions via a telephone network or an Internet connection by establishing a remote connection to a remote system. The online connection allows you to edit a remote system in the usual way with the TIA Portal.

Advantages

Using TeleService offers the following advantages:

- You can easily access even remote sections of plants and include them in a complete system.
- You can offer rapid help and support in the event of faults in a remote system without having to go there yourself.
- You can employ your resources effectively.
- It significantly reduces costs.
- It can significantly reduce plant downtimes.
- It improves the efficiency of your plant.

See also

TeleService functionality (Page 6574)

14.7.1.2 TeleService functionality

TeleService fields of application

TeleService offers the following the fields of application:

- **Access to remote systems (remote maintenance):**
  You can manage, control, and monitor remote systems centrally by means of remote connections.
  This is possible with an S7-300/400 CPU, an S7-1200 CPU and an S7-1500 CPU and a TS Adapter MPI or a TS Adapter IE.
- **Establishing connections from and to remote systems (PG-AS remote link):**
  You can use PRODAVE MPI V5.0 and newer to establish a remote connection to a remote system, and the communications instruction "PG_DIAL" to establish a remote connection from a remote system.
  This is possible with an S7-300/400 CPU and a TS Adapter MPI.
- **Data exchange between systems (AS-AS remote link):**
  The communications instruction "AS_DIAL" allows two automation systems to exchange process data over the telephone network.
  This is possible with an S7-300/400 CPU and a TS Adapter MPI.
- **Sending a text message from a system:**
  An automation system can send a message (SMS) via a GSM wireless modem using the communications instruction "SMS_SEND" ".
  This is possible with an S7-300/400 CPU and a TS Adapter MPI.
- **Sending an email from a system**
  An automation system can also send an email with the following communications instructions and a TS Adapter IE .
  - S7-300/400 CPUs (CPU S7-31x-2PN/DP or CPU 41x-3PN/D) use the instruction "AS_MAIL"
  - S7-1200 CPUs use the instruction "TM_MAIL"
  - S7-1500 CPUs use the instruction "TMAIL_C"

14.7.1.3 Telephone book at TeleService

Introduction

By double clicking the "phone book" folder in the project tree you open the phone book editor displaying the TeleService phone book.

Each version of the TIA Portal has its own "global phone book". If a global phone book from an earlier version of the TIA Portal is found in a more recent version of the TIA Portal, you will be asked once if you want to import this phone book.

This gives you the advantage of being able to import the system data from the previous version into the more recent version of the TIA Portal.

Global phone book properties

The global phone book is used in TeleService to manage specific system data that are required to establish a remote connection.

When you open the phone book for the first time, you will see an empty phone book with all available columns; in all other cases, the last phone book edited is displayed.

You can enter any number of systems in a phone book. Systems contain the data required for establishing a remote connection, for example, the name and location of the device and the phone number to be dialed, along with all country-specific details. With VPN connections, you can enter an IP address or a DNS name instead of the phone number.

The TS adapters used for establishing the connection are distinguished by color, depending on whether a TS Adapter MPI or a TS Adapter IE is used for establishing the connection.

14.7.2 Working with the phone book

14.7.2.1 Basics on working with the phone book

Working with the phone book

You have the following options when working with a phone book:

- Open phone book
- Save phone book
- Import phone book data
- Export phone book data
- Printing the phone book
- Use phone book data to establish a remote connection

You can implement these functions simply and easily by using the buttons in the phone book toolbar.

Note

Access to phone books

The phonebook is user specific in TeleService. However, it is not possible to access the global phone book with more than one instance of the TIA portal at the same time.

See also

Open phone book (Page 6577)

Saving phone book (Page 6578)

Exporting phone book data (Page 6580)

Printing the phone book (Page 6581)

Structure of the phone book (Page 6576)

14.7.2.2 Structure of the phone book

Introduction

A global phone book in TeleService is used to manage the data you require for establishing a remote connection. Once you have created the connection data and saved it in the phone book, you can access it each time you want to establish a remote connection.

Structure of the phone book

The integrated global phone book in TeleService contains the following columns:

Column name	Description
Plant name	Enter a name for your system.
Adapter type	Select the TS Adapter type used in the drop-down list: TS Adapter MPI or TS Adapter IE.
Connection type	Select the required connection type: Dial-up connection or VPN connection.
Area code	Enter the required area code. This column is only active with dial-up connections. This column cannot be edited for VPN connections.
Phone number/remote address	Enter the required connection data for establishing the remote connection. For dial-up connections, this can be a phone number, and for VPN connections a DNS name or an IP address.
Fingerprint	Enter the corresponding fingerprint for establishing a VPN connection to the TS Adapter IE Advanced.
Country	Enter the country code. This column is only active with dial-up connections. This column cannot be edited for VPN connections.
User name	Enter the user name you have logged on under.
Password	Enter the password for this user name.
Group	Enter the group if you have carried out grouping.
Company	Enter the company to be contacted.
Department	Enter the relevant department.
Street	Enter the street.
Town/City	Enter the town or city to which the remote connection is to be established.
Comment	Enter a comment if required.

Showing or hiding columns

You can show or hide the individual columns. To do so, select the required column header and open the shortcut menu with the right mouse button.

14.7.2.3 Symbols in the phone book

Meaning of the TeleService icons

The following table shows the meaning of the TeleService icons:

Symbol	Meaning
	Open the global phone book
	Imports a phone book
	Exports a phone book
	Establishes a remote connection
	Terminates the active remote connection
	Establishes a remote connection or terminates it
	Displays the connection to a TS Adapter IE in the phone book
	Displays the connection to a TS Adapter MPI in the phone book
	Adds a new row to the phone book
	Inserts a new row in the phone book

14.7.2.4 Manage phone book

Open phone book

Opening phone books

To open the phone book, follow these steps:

1. In the project tree, double-click on the "Phone book" folder under "Online access" > "TeleService".
2. The phone book opens so that you can enter or edit the required system data.

Inserting rows in the phone book

Inserting rows in the phone book

To insert a new row in the phone book, follow these steps:

1. Select the row in front of which you want to insert a new row.
2. Click the "Insert row" button in the toolbar.

Result

A new row is inserted in the phone book above the selected row.

Showing and hiding columns in the phone book

Showing and hiding columns

To show or hide columns in the phone book, follow these steps:

1. Click a column header.
2. In the shortcut menu, select the "Show/hide columns" command.
   The selection of available columns is displayed.
3. To show a column, select the column's check box.
4. To hide a column, clear the column's check box.

Result

The respective columns are shown or hidden when displaying the phone book.

Saving phone book

Saving phone books

When you exit the phone book editor or the TIA Portal, you are asked whether you want to save the global phone book.

Click "Yes" to save the phone book.

Importing phone book data

Introduction

It is possible to import the phone book data from an external file or from an older version of the TIA Portal.

Requirement

You have already created an import-capable phone book file.

You have already created a phone book with an older version of the TIA Portal.

Importing phone book data from a phone book file

To import phone book data from a phone book file, follow these steps:

1. Open the "TeleService" folder under "Online access" in the project tree.
2. Double-click on the "Phone book" folder.
3. Click the "Import" button in the toolbar.
4. Confirm the prompt asking if you want to save the current state of the phone book with "Yes", if applicable, and specify the location for storing the phone book in the dialog that follows.
5. If you do not want to save the current state of the phone book, answer the prompt with "No". In the subsequent dialog, select the phone book file to which the current phone book should be stored.
6. Close the dialog box with "OK".

Result

The imported phone book data is displayed in the global phone book.

Note

Defining dialing rules

Dialing rules must be defined before the "area code" and "country" columns in the imported phone book can be edited for dial-up connections.

To define dialing rules, follow the link in the history.

Importing phone book data from an older version of the TIA Portal

To import phone book data from an older version of the TIA portal, follow these steps:

1. Open the "TeleService" folder under "Online access" in the project tree.
2. Double-click on the "Phone book" folder.
3. Click the "Import" button in the toolbar.
4. Confirm the prompt asking if you want to save the current state of the phone book with "Yes", if applicable, and specify the location for storing the phone book in the dialog that follows.
5. If you do not want to save the current state of the phone book, answer the prompt with "No".
6. Enter the following path in the dialog that follows: "%appdata%\siemens\automation \TeleService\GlobalTeleServicePhoneBook.tel".
7. Close the dialog box with "OK".

Result

The phone book data imported from the older version of the TIA Portal is displayed in the global phone book.

Note

Defining dialing rules

Dialing rules must be defined before the "area code" and "country" columns in the imported phone book can be edited for dial-up connections.

To define dialing rules, follow the link in the history.

See also

Defining dialing rules (Page 6581)

Exporting phone book data

Introduction

It is possible to export phone book data to an external file.

Requirement

You have already created a phone book under TeleService with the corresponding system data.

Procedure

To export the phone book data, follow these steps:

1. Open the "TeleService" folder in project tree.
2. Double-click on the "Phone book" folder.
3. Click the "Export" button in the toolbar.
4. In the next dialog box, select where the current phone book is to be exported.
5. Close the dialog box with "OK".

Result

The exported phone book data are saved in the specified export file.

Printing the phone book

Printing phone books

You can print out all or some of the data in a phone book.

Proceed as follows:

1. Open the phone book.
2. Select the **Project > Print** menu command or click on the appropriate button in the toolbar. The "Print" dialog will open.
3. Specify whether you wish to print the complete phone book or just part of it and set all other options.
4. Start the print job with "OK".

Result

The phone book data is printed on the default printer. If the printout is more than one page long, an identifier is printed after the page number at the bottom right corner of the page to indicate that there is another page. The last page does not have this symbol, indicating that no more pages are to follow.

Defining dialing rules

Procedure

To define location-specific dialing rules, follow these steps:

1. Open the "Online access" folder in the TIA Portal and select the "TeleService" folder.
2. Use the shortcut menu to access the "TeleService" properties.
3. In the dialog that follows, open the "Advanced settings" tab and activate the "Use dialing rules" option.
4. Click the "Adapt" button.
5. In the dialog that follows, enter the required location information for connection establishment.
   - For new locations, enter the country or region and the corresponding area code.
   - If required, set the correct destination code and pre-dial line for local/long-distance calls.
   - Select your chosen dialing mode for the location.
6. Exit the dialog by clicking "OK".
7. In the next dialog, select the location from which you wish to dial, and click "OK".

8. In the TeleService dialog that follows, check that the correct location appears under "Location".
9. Confirm your entries by clicking "OK".

Note

Modem operation on a local loop or extension

You do not need to specify a pre-dial line if you operate your modem on a local loop (main telephone line). The fields for the pre-dial lines for local calls and long-distance calls must be empty.
If you operate your modem on an extension, you should enter the pre-dial lines which need to be called to access a local loop.

Example of the use of dialing rules

The following example illustrates the use of location-specific dialing rules for the city of Karlsruhe in Germany.

- In the TeleService phone book, you enter the phone number "1234567", without the area code and without the country code.
  Result:
  The phone number "1234567" is always dialed, irrespective of the location of the caller.
- In addition to the phone number "1234567", you enter the location-specific dial parameters for the Karlsruhe area code "0721" and the country code "+49" for Germany in the TeleService phone book.
  Result:
  "1234567" is dialed from Karlsruhe.
  "07211234567" is dialed from other towns in Germany.
  "00497211234567" is dialed from other countries.

Note

Display in the phone book

The "area code" and "country" columns in the TeleService phone book can only be edited once you have defined the dialing rules as detailed above.

14.7.3 Remote connections as dial-up connections

14.7.3.1 Basics for establishing a dial-up connection

Using a TS Adapter for dial-up connections

A TS Adapter is needed for establishing a remote dial-up connection using TeleService.

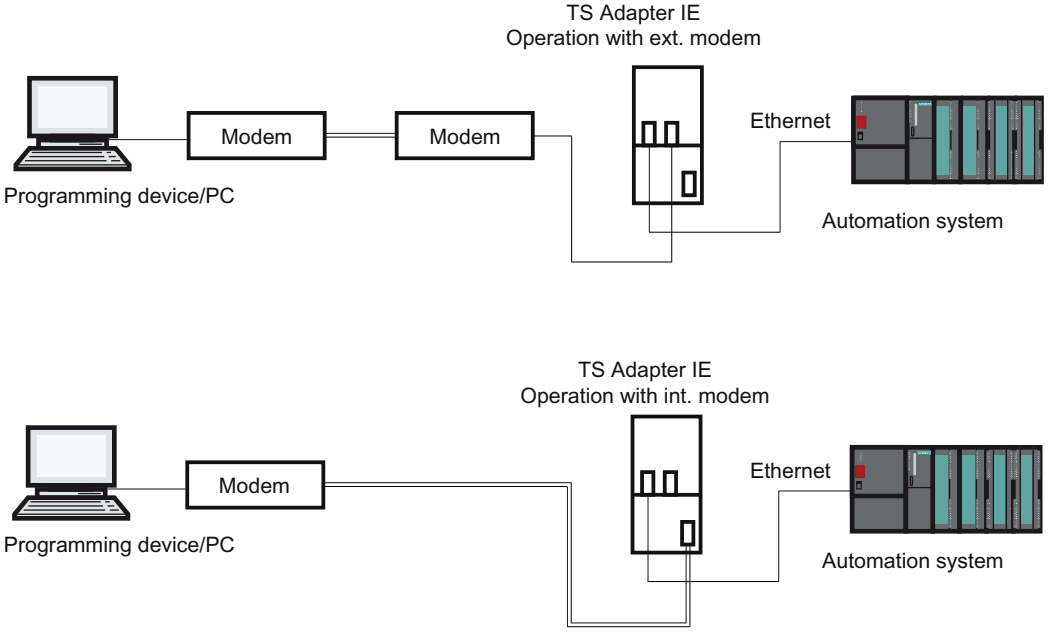
The TS Adapter is used to prepare an automation system for use with TeleService by connecting it to a telephone network via a modem. The TS Adapter has an integrated parameter memory in which a parameter set for TeleService operation is stored.

With the function "export adapter parameter", different parameter sets can be saved in external files, and reloaded in the TS Adapter via the function "import adapter parameter".

Establishing a remote dial-up connection

You can choose between a number of different TS Adapters, each of which offers a different functionality and different connection options.

The figure below shows two possible configurations for establishing a dial-up connection to a system with the TS Adapter IE.



Overview of possible TS Adapter:

The TS Adapter comes in the following versions:

- TS Adapter II (also designated "TS Adapter MPI")
- TS Adapter IE Standard (also designated "TS Adapter IE")
- TS Adapter IE Basic (also designated "TS Adapter IE")

Designation "TS Adapter"

In the following pages, the designation "TS Adapter" stands for all versions. The relevant product designation is listed beside information which only applies to a specific version, for example, "TS Adapter II" or "TS Adapter IE Standard".

Note

For more detailed information on your TS Adapter, please refer to the documentation supplied with the TS Adapter.

See also

Short description of the TS adapter MPI (Page 6592)

Short description of the TS adapter IE (Page 6599)

Exporting adapter parameters (Page 6597)

Importing adapter parameters (Page 6598)

14.7.3.2 Telephone networks and modems

Supported telephone networks and modems

Telephone networks which can be used

TeleService can be used with digital networks (ISDN), analog networks and wireless networks (with GSM technology). This version supports a remote connection to a TS Adapter.

Modem support

TeleService has been implemented to be independent of the modem. This means that all standard modems which can be installed in the Windows Control Panel and are visible as modems can also be used by TeleService.

The choice of modem type is determined primarily by the existing hardware of the programming device/PC and the telephone network to be used.

The following modem types/media are supported:

- Modems (external modems at the COM interface, internal modems and PCMCIA cards)
- External ISDN adapter at the COM interface or USB interface
- Internal ISDN adapter with virtual COM interface (for example AVM CAPI port)
- External ISDN modems (ISDN adapter with integrated analog modem functionality) at the COM interface or USB interface
- Radio network modems with GSM technology, PCMCIA adapter card or data cable and mobile phone

Gateways

Gateways between the various telephone networks are in principle possible. Remote connections from an ISDN adapter to an analog modem and vice versa only function with special ISDN telephone adapters.

Performance in telephone networks

The data throughput of a remote connection depends on the modem and telephone network used and the quality of the telephone line.

Installing the local modem

Introduction

If you have already installed a modem for data transfer in your operating system, this modem can also be used for TeleService.

If a modem has not yet been installed for your operating system, one must be installed before you can establish a remote connection with TeleService.

Procedure

Proceed as follows:

1. Make sure your programming device/PC and the modem are switched off.
2. Physically connect the external modem to a COM or USB interface on your programming device/PC. You can also install an internal modem or a PCMCIA card in accordance with the manufacturer's specifications.
3. Now switch on the modem and then the programming device or the PC.

Result

Plug-and-play modems are recognized and installed automatically by the operating system. Dialogs will take you through the installation procedure.

Note

Modems without plug-and-play

If your modem is not recognized automatically when switched on, you will have to install it yourself using the Control Panel.

Please refer to the information in the documentation supplied with your modem.

Connecting and configuring the remote modem

Introduction

A modem must also be connected to the remote system before you can work with TeleService. This modem is designated the "remote modem".

Configuring the remote modem

The modem receives all parameters required for operation from the TS Adapter connected. These include data for initializing the modem and settings for serial transmission between the TS Adapter and the modem.

The data required for the remote modem is specified during the configuration of the TS Adapter.

The modem may be internal or external depending on the TS Adapter used.

How to connect a TS Adapter with an internal modem

1. Switch off the TS Adapter.
2. Connect the TS Adapter to the automation system.
3. Connect the TS Adapter to the telephone line.
4. Switch on the TS Adapter.

How to connect a TS Adapter with an external modem

1. Switch off the modem.
2. Connect the TS Adapter to the automation system.
3. Connect the TS Adapter to the modem using a modem cable.
4. Connect the modem to the telephone line.
5. Switch on the modem.
6. Switch on the TS Adapter.

Note

Please note the following information on configuring the remote modem:

- The default parameters for the modem and the serial port set in the TS Adapter should in most cases ensure successful operation; changes in the parameter assignment will only be needed in rare cases.
- You need only change the parameter assignment of the TS Adapter if a modem connection is not established or if factory settings are to be adapted or optimized.
- TS Adapter parameter assignment can be changed via either a direct or a remote connection.

14.7.3.3 Access protection for dial-up connections

Access protection information

Introduction

When you assign the parameters for your TS adapter, you can restrict access to the parameters of the TS adapter and access to remote systems.

Scope of access protection

Access protection only exists for remote connections; TS adapter parameter assignment can be accessed at any time in direct connection mode.

Access protection also exists in direct connection for the TS adapter IE.

Access protection information

The TS Adapter MPI is not delivered with access protection activated. There is a default password for the TS adapter IE.

The first user who assigns the parameters for this adapter can therefore activate access protection by defining the password for a user and/or a callback number.

This is a multi-level access protection with several users, each with or without administrator rights. For the TS adapter MPI there is only one adminstrator and no more than two users.

For a modem connection, only an administrator can define the two users and change and, if necessary, delete their settings. Those logged in as users can only change their own passwords and their own callback numbers. However, with the TS adapter MP you can access the process of assigning parameters of the TS adapter in direct connection, without restriction.

Advantages

Access protection offers the following advantages:

- Unauthorized access by persons outside the system is almost impossible.
- The plant operator bears most of the telephone costs.

TeleService callback options

Callback variants

The costs of a telephone connection are normally borne by the caller who establishes the dial-up connection.

TeleService can, however, be used so that after a short initial connection the modem connection is established again in the opposite direction, in other words initiated by the TS Adapter (callback). In this case, the plant operator bears the costs of the callback.

There are two callback variants in TeleService:

1. Callback to a number specified during connection establishment.
2. Callback to a number stored on the TS Adapter.

Levels of protection

Introduction

You can set up one of two possible levels of access protection for TeleService access to the TS Adapter. Different options are available with each protection level.

Access protection options

Access protection level 1:

The TS Adapter is protected by the user name and password. You can access the TS Adapter via any telephone line and specify any callback number during connection establishment.

Access protection level 2:

The TS Adapter is protected by the user name, password, and the callback number. You can only access the TS Adapter from one telephone connection per user.

The table below sets out the above conditions for the various protection levels:

Level of access protection	Administrator/User password	Callback number
1	enter	do not enter
2	enter	enter

Logging on to TS Adapter

When you log on to the TS Adapter and after you have set up access protection, enter your user name, the corresponding password and, if desired, a callback number:

Level of access protection	Administrator/User password	Callback number
1	enter	do not enter or enter any callback number
2	enter	do not enter

If you have entered a callback number during connection establishment (access protection level 1) or have stored a callback number in the TS Adapter (access protection level 2), the modem connection is disconnected and the TS Adapter calls the specified number back.

Setting up access protection and callback number for the TS adapter

Introduction

During the parameter assignment for the TS adapter MPI in TeleService, you can set up access protection and a callback number for the parameter assignment of the adapter and connection to the remote system. The following describes the parameter assignment for a TS adapter MPI. The parameter assignment of a TS adapter IE is carried out in analog. The specific method is described in the web help of this adapter.

Requirement

A TS Adapter MPI is connected to your computer and is displayed in the project tree under "Accessible nodes".

Procedure

To set up access protection for the TS adapter, proceed as follows:

1. Click on the command "Assign TS Adapter MPI parameters" in the project tree.
2. Open the "Access security" tab.
3. Enter a password for your user name and/or number that you want the modem to call back following logon.
   - If you are an administrator, you can change all the settings for administrators and users, and delete or create users.
   - If you are logged on as a user, you can only change your own settings (password and callback number).
4. Confirm all entries before exiting the dialog with "OK".
5. Click the "Yes" button to confirm the following query.

Result

The parameter assignment for the access protection and the callback number is saved in the non-volatile memory of the TS adapter MPI.

Note

Important points to note when setting up access protection:

- The settings in the "Modem" tab must correspond to the conditions at the plant if callback functionality is to be guaranteed.
- Entering an incorrect callback number in the role of "ADMIN" user will mean you are no longer able to access the TS Adapter MPI over a remote connection!
- Test the callback number before you enter it as the "ADMIN" user by calling the given callback number during connection establishment (access protection level 1).

Complete a callback in TeleService

Callback options

Two different callback variants can be set up in TeleService.

The following callback options are available:

- Callback to a number specified during connection establishment.
- Callback to a number stored on the TS Adapter

Callback to a number specified during connection establishment

1. In the project tree of the TIA Portal, click the "Online access" folder.
2. Click on the "TeleService" folder it contains.
3. Double-click the "Set up/close remote connection" entry. The "Set up remote connection to the remote system" dialog opens.
4. In the "Adapter type" drop-down list, select the adapter type used.
5. Select the "dial-up connection" under "Connection type" if it is not already selected.
6. Select the modem you are using under "Local Settings".
7. Enter the phone number to be dialed in the appropriate box or open the phone book by clicking on the button behind it and take the desired phone number from the phone book.
8. Enter your user name and associated password of the TS adapter.
9. If you want a "Connection setup with callback", select the appropriate option button.

10. Click the "Connect" button to establish the required remote connection. This button only becomes active when you have entered all the parameters needed to establish a remote connection. Any remote connection is displayed under "Status".
11. Enter the desired callback number in the dialog that follows.

Result

The remote connection to the desired system is made with callback.

The connected system is shown with the corresponding icon in the project tree.

Note

This procedure is useful if the costs of the modem connection are to be borne by the plant operator and if the actual callback number is not fixed, which means callback is not always to the same receiver. It is particularly useful for mobile users.

Callback to a number stored on the TS Adapter

1. Assign the parameters for the desired callback number in the TS adapter.
2. Establish a connection to the TS adapter as described above, and observe the following features:
   - Enter the user name and password for which the callback number parameters are assigned in the TS adapter.
   - The check box "Establish a connection with callback" does not have to be selected, since the callback number is already known by the TS adapter.

Result

Callback to a number stored on the TS adapter has been established. If a remote connection is established, the callback occurs from the remote system.

Note

This procedure offers the highest level of access protection. However, it does pose a risk: if the callback number stored on the TS Adapter is not correct, it will no longer be possible to access the TS Adapter over a modem connection. The device can in such a case only be put back into operation by changing the parameter settings on site.

14.7.3.4 TS adapter MPI

Short description of the TS adapter MPI

TS Adapter MPI:

The designation "TS Adapter MPI" is a collective term for allTS Adapter with an MPI/DP interface.

The TS Adapter MPI comes in the following versions:

- As TS Adapter I (parameters cannot be assigned via the TIA Portal)
- as TS Adapter II

The table below provides a short description of the functionalities. For detailed information on your TS Adapter, please refer to the documentation supplied with your TS Adapter.

TS Adapter II:	
Direct connection	Connection via the Universal Serial Bus (USB). The firmware can be replaced. The modem is integrated or can also be connected as external modem. The TS Adapter II switches automatically between the modems. As long as no external modem is connected, the adapter will use the internal modem.
There are two variants	• With internal analog modem. An external modem can also be connected to the RS232 port. • With internal ISDN adapter. An external modem can also be connected to the RS232 port.

Use of the designation "TS Adapter"

For TeleService the designation "TS Adapter" is the generalization for all versions. The relevant product designation is listed beside information which only applies to a specific version of a TS Adapter, for example, "TS Adapter II", "TS Adapter IE Standard" or "TS Adapter IE Basic".

How the TS adapter MPI works

How the TS Adapter MPI Works

In line with the configuration, the TS Adapter MPI connects the serial port or USB port of your programming device/personal computer (direct connection) or the serial port of a modem (modem connection) to the MPI/PROFIBUS network of your automation system.

The TS Adapter MPI has a non-volatile memory. Parameters for the following functions are stored in this memory:

- The MPI/PROFIBUS network (network parameters)
- The mode of the modem used

- The serial port to the modem
- Access protection

Default parameter assignment

The TS Adapter comes with default parameter assignment. The parameters can be set and saved to the non-volatile memory of the TS adapter in a parameter assignment session.

When "Direct connection" is configured, the TS Adapter will only use the network parameters for access to the MPI/PROFIBUS network.

In the "Modem connection" configuration, all the parameters stored on the TS Adapter will be activated.

Note

For more detailed information on the configuration of your TS Adapter, please refer to the documentation supplied with it.

Operating a TS adapter MPI in direct connection mode

Direct connection with TS Adapter MPI

The direct connection is used to assign the parameters of the TS Adapter MPI. The same configuration also allows you to go online in the TIA Portal and thereby check the assigned MPI/PROFIBUS parameters for bus compatibility. This means that (as with a PC adapter) SIMATIC S7/C7 systems can be accessed via the MPI/DP interface without an MPI/ PROFIBUS module occupying a slot for a programming device/PC.

Access protection for the TS Adapter is not active in direct connection configuration. This means that the parameter assignment of the TS Adapter can be changed without any problems, for example by importing adapter parameters.

Note

Display of the TS Adapter MPI in the TIA Portal

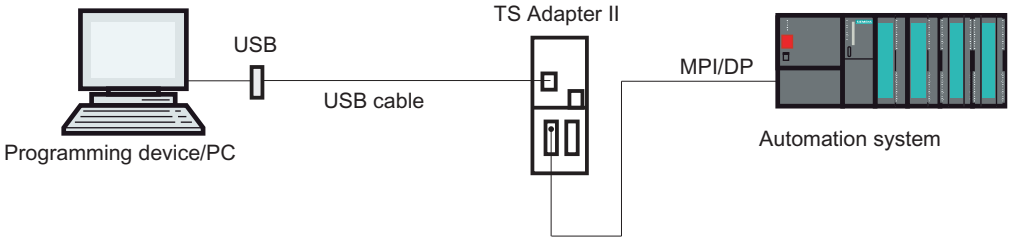
As soon as you have connected a TS Adapter MPI to the programming device/PC via the USB interface, the "TS Adapter" folder is displayed in the project tree in the TIA Portal.

When you open the folder, you can assign the parameters of the connected TS Adapter MPI via the following dialog.

Establishing the direct connection for TS Adapter MPI

Direct connection mode means there is a direct connection via the TS Adapter MPI between the programming device/personal computer on which TeleService is installed and the automation system. No modem is required.

The figure below shows the configuration of the TS Adapter MPI with a direct connection.



Operating a TS adapter MPI in modem connection mode

Introduction to the modem connection with TS Adapter MPI

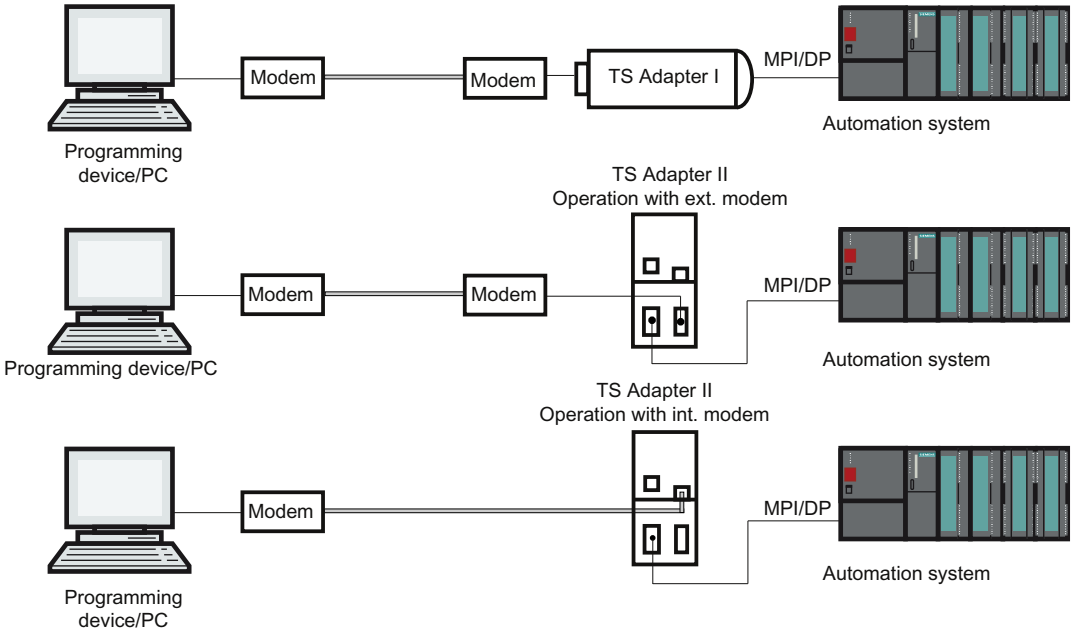
This configuration allows you to dial into a remote system. To do this, you establish a remote connection to a remote system using TeleService on a telephone network. Via the established modem connection you can then process the selected plant as usual with the TIA Portal.

Establishing a modem connection with TS Adapter MPI

This connection between the programming device or PC on which TeleService is installed and the automation system in which the TS Adapter MPI is inserted in the MPI/DP interface is made through a modem route.

The configuration therefore includes the programming device or PC via the telephone network and the TS Adapter MPI on the MPI/DP interface of the automation system.

The figure below shows the structure of the modem connection.



Note

Parallel operation between direct and modem connection

The TS Adapter II has two connections for communication with PG/PC, both of which can be connected at the same time. At the same time, connect the USB interface with the PG/PC and the modem interface with the telephone network.

In this configuration you can either use the direct or the modem connection.

A parallel operation is **not** possible!

TS adapter MPI configuration options

Useful information on configuring the TS Adapter MPI

The TS Adapter MPI can be configured in both direct connection mode and via an existing remote connection.

The following parameter assignment options are available:

- Reconfiguration (Page 6596)
- Restoring default parameter assignment (Page 6597)
- Importing adapter parameters (Page 6598)
- Exporting adapter parameters (Page 6597)
- Setting up access protection (Page 6589)

Parameter assignment

Configure your TS Adapter in accordance with the documentation supplied with the TS Adapter. It will detail the exact procedure for parameter assignment.

Note

Please note the following when configuring the TS Adapter MPI

- If you change the current parameter settings when there is an established remote connection, there is a risk it will not subsequently be possible to establish a modem connection with the modified parameters. The TS Adapter MPI can in this case only be configured in direct connection mode.
- This means that either parameter assignment must be carried out with a programming device/personal computer at the plant location or the TS Adapter MPI must be brought to the location of the local programming device/personal computer in order to be configured.

Positive acknowledgement

During parameter assignment, the data is written to the non-volatile memory of the TS Adapter MPI. The parameter assignment process is not acknowledged positively until all precautions have been taken to ensure that parameter changes have been carried out correctly and will thus survive a power failure.

Changes become effective for the TS Adapter MPI as follows:

- The serial parameters, the modem parameters and the parameters for access protection are activated once the remote connection has been terminated.
- The modified network parameters are activated immediately.

Configuring TS adapter MPI

Introduction

The TS Adapter MPI can be configured in both direct connection mode and via an existing remote connection in modem connection mode.

The following describes the method for assigning parameters.

Requirement

A TS Adapter MPI is connected to your computer and the folder "TS Adapter" is displayed in the project tree under "Online access".

Procedure

To assign the parameters for the TS Adapter MPI im Direktanschluss , follow these steps:

1. Double-click the "Online access" folder in the project tree.
2. Open the required folder:
   - The "TS Adapter" folder for a direct connection.
   - For an existing remote connection, the "TeleService" folder followed by the folder with the required system name.
3. Select the command "Assign TS Adapter MPI parameters". The "Assign TS Adapter MPI parameters" dialog opens.
4. Set the required parameters in the individual tabs of the dialog.
5. Confirm your entries with "OK".

Result

The configured parameters are saved in the non-volatile memory of the TS Adapter MPI. Parameter assignment is then complete.

Restoring default parameter assignment for TS adapter MPI

Introduction

You can restore the default, factory state parameters of the TS Adapter MPI.

Requirement

A TS Adapter MPI is connected to your computer and is displayed in the project tree under "Online access" in the "TeleService" folder.

Procedure

To restore the default parameter assignment for the TS Adapter MPI, follow the steps below:

1. Open the "TeleService" folder in project tree.
2. Double-click the "TS Adapter MPI" folder.
3. Select the command "Assign TS Adapter MPI parameters". The "Assign TS Adapter MPI parameters" dialog opens.
4. Click the "Reset" button under "General".
5. Confirm your entries with "OK".

Result

The TS Adapter MPI default parameters set on delivery are restored.

See also

TS adapter MPI configuration options (Page 6595)

Exporting adapter parameters

Introduction

You can export the configuration of a TS Adapter MPI to an external file. The configuration saved in this file can be imported in turn into any number of TS Adapter MPI.

This can for example be useful if you want to assign identical parameters to multipleTS Adapter MPI or if you want save, document or distribute the parameter set.

Requirement

A TS Adapter MPI is connected to your computer and is displayed in the project tree under "Online access" in the "TeleService" folder.

The parameter is stored on the TS adapter.

Procedure

To export the adapter parameters of a TS adapter MPI, follow the steps below:

1. Open the "TeleService" folder in project tree.
2. Double-click the "TS Adapter MPI" folder.
3. Select the command "Assign TS Adapter MPI parameters". The "Assign TS Adapter MPI parameters" dialog opens.
4. Click the "Export" button.
5. A window will open in which you can select the file to which you wish to export the configuration of the TS Adapter MPI.
6. Confirm with "Save".

Result

The parameters of the TS Adapter MPI are saved in the specified file (*.tap). The export of the adapter parameters is now complete.

Importing adapter parameters

Introduction

You can import the configuration of a TS Adapter MPI from a previously created export file (*.tap).

The configuration saved in this file can be imported into any number of TS Adapter. This can for example be useful if you want to assign identical parameters to multiple TS Adapter MPI.

You can import parameters locally in direct connection mode or via an existing remote connection in modem connection mode.

Requirement

A TS Adapter MPI is connected to your computer and is displayed in the project tree under "Online access" in the "TeleService" folder.

Procedure

To import the adapter parameters of a TS Adapter MPI:

1. Open the "TeleService" folder in project tree.
2. Double-click the "TS Adapter MPI" folder.
3. Select the command "Assign TS Adapter MPI parameters". The "Assign TS Adapter MPI parameters" dialog opens.
4. Click the "Import" button.

5. A dialog will open in which you can select the file to which you wish to import the configuration of the TS Adapter MPI.
6. Confirm the next dialog with "Yes".

Result

The parameters selected are saved in the non-volatile memory of the TS Adapter MPI. Adapter parameter import is then complete.

14.7.3.5 TS adapter IE

Short description of the TS adapter IE

TS Adapter IE

The designation "TS Adapter IE" is a collective term for allTS Adapter with an Ethernet port.

The TS Adapter IE comes in the following versions:

- as TS Adapter IE Standard
- as TS Adapter IE Basic

The tables below provide a short description of the functionalities. For detailed information on your TS Adapter, please refer to the documentation supplied with it.

TS Adapter IE Standard:
Direct connection by means of Industrial Ethernet (IE). Firmware update possible. Modem integrated or external. The TS Adapter IE cannot automatically switch between modems like the TS Adapter II. Parameters are assigned via a Web interface.
There are 2 variants:
• With internal analog modem. An external modem can also be connected to the RS232 port.
• With internal ISDN adapter. An external modem can also be connected to the RS232 port.

TS Adapter IE Basic:
Direct connection by means of Industrial Ethernet (IE). Firmware update possible. Plug-in modules. Parameters are assigned via a Web interface.
There are 4 variants:
• TS Adapter IE Basic MODEM: Basic device TS Adapter IE Basic with TS Module MODEM for operation on the analog telephone network.
• TS Adapter IE Basic ISDN: Basic device TS Adapter IE Basic with TS Module ISDN for operation on ISDN telephone systems.
• TS Adapter IE Basic GSM: Basic device TS Adapter IE Basic with TS Module GSM for operation on the GSM radio network.
• TS Adapter IE Basic RS232: Basic device TS Adapter IE Basic with TS Module RS232 for connecting an external modem.

Use of the designation "TS Adapter"

"TS Adapter" is used in the TeleService online help as a general designation for all versions. The relevant product designation is listed beside information which only applies to a specific version of a TS Adapter, e.g. "TS Adapter I", "TS Adapter II", "TS Adapter IE Standard" or "TS Adapter IE Basic".

How the TS adapter IE works

How the TS Adapter IE works

The TS Adapter IE connects the telephone network or the serial port of a modem with the Industrial Ethernet of your automation system.

The TS Adapter IE has a non-volatile memory. Parameters for the following functions are stored in this memory:

- The mode of the modem used
- The serial port to the modem
- Access protection

Default parameter assignment

The TS Adapter IE comes with default parameter assignment. The parameters can be set and saved to the non-volatile memory of the TS adapter in a parameter assignment session.

Note

For more detailed information on the configuration of your TS Adapter, please refer to the documentation supplied with it.

Connection Types

Connection types of the TS Adapter IE Basic

The following diagrams show the connection types possible with the TS Adapter IE Basic.

Direct connection

In the direct connection to the PG/PC, you can set the TS Adapter IE Basic through Ethernet.

Note

The operation of the TS Adapter IE Basic without a TS module is not permitted.

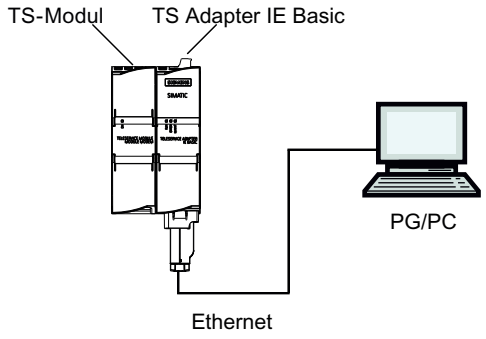


Figure 14-1 Direct connection

Connection to the telephone network

In order to have a direct connection to the telephone network, you must connect the TS Adapter IE Basic together with one of the following TS modules:

- TS Module Modem
- TS Module ISDN

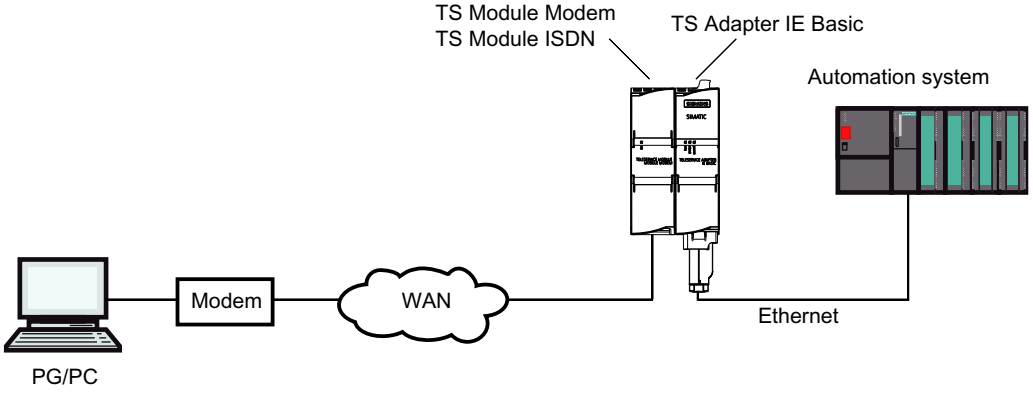


Figure 14-2 Direct connection to the telephone network

More information about the TS modules can be found in the *TS Adapter modular* manual.

Connection to the GSM network

In order to connect to the GSM network, you must operate the TS Adapter IE Basic together with this TS module:

- TS Module GSM

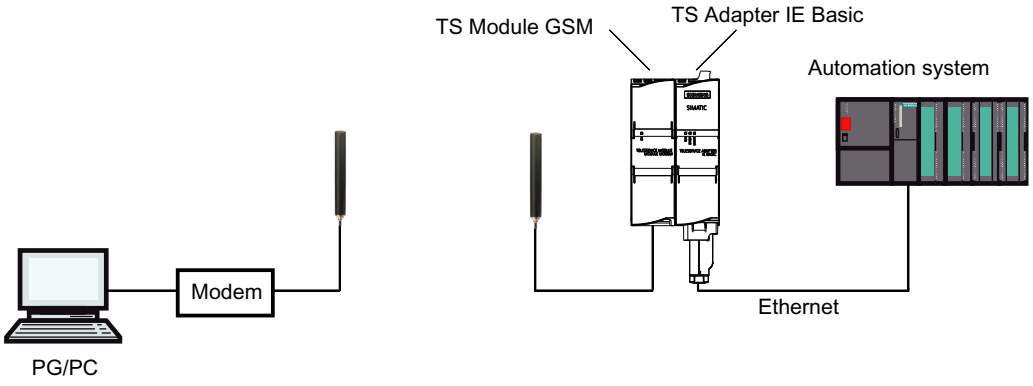


Figure 14-3 Connection to the GSM network

More information about the TS modules can be found in the *TS Adapter modular* manual.

Connection to the telephone network through an external modem

For the connection to an external modem, you must operate the TS Adapter IE Basic together with this module:

- TS Module RS232

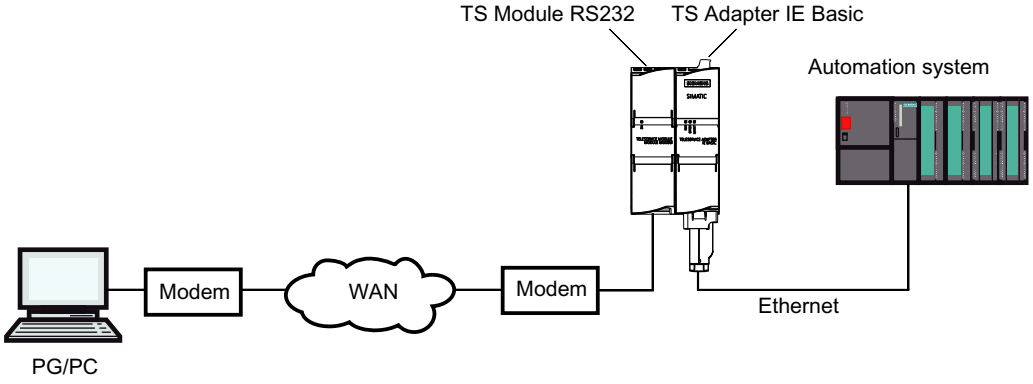


Figure 14-4 Connection to an external modem

More information about the TS modules can be found in the *TS Adapter modular* manual.

TS Adapter IE parameter assignment options

Useful information for configuring the TS Adapter IE

The TS Adapter IE is configured via a Web interface.

Web help associated with the parameter assignment interface is made available for the configuring the TS Adapter IE.

The following parameter assignment options are available:

- Reconfiguration
- Restoring default parameter assignment
- Importing adapter parameters
- Exporting adapter parameters

Note

Parameter assignment

Configure your TS Adapter in accordance with the documentation supplied with the TS Adapter. It will detail the exact procedure for parameter assignments.

Parameter assignment for TS Adapter IE

Introduction

The TS Adapter IE can be configured in both direct connection mode and via an existing remote connection in modem connection mode.

Both parameter assignment options are described below.

Specific details for assigning parameters of the TS Adapter IE can be obtained from the TS Adapter IE documentation.

Parameter assignment of the TS Adapter IE in direct connection

Requirement

There is a LAN connection to your TS Adapter IE.

The TS Adapter IE Basic is connected to the power supply.

Procedure

To assign the parameters for the TS Adapter IE , follow these steps:

1. In the project tree of the TIA Portal, open the "Online access" folder.
2. Double-click on the Ethernet port of your computer.
3. Double-click on the "Display accessible nodes" command. The TS Adapter IE is then displayed.

4. Double-click on the <TS Adapter IE> folder and then on "Online and diagnostic", and assign the desired IP address to the TS Adapter IE in the following dialogs. Please note that the IP address of the PG/PC interface card is located in the same subnet as the IP address that you issue for the TS Adapter IE.
5. Update the view in the project tree for the "Accessible nodes", so that the TS Adapter IE is displayed with the newly allocated IP address.
6. Open the folder <TS Adapter IE> in the device list.
7. Double-click the command "Assign TS Adapter IE parameters". The allocated web interface opens for assigning the parameters of the TS Adapter IE.
8. Complete the "logon" for the web interface.
9. Set the required parameters in the individual tabs of the dialog.
10. Confirm your entries with "Save settings".

Result

The configured parameters are saved in the non-volatile memory of the TS Adapter IE. Parameter assignment is then complete.

Parameter assignment of the TS Adapter IE using a remote connection

Requirement

There is an established remote connection to a TS Adapter IE .

Procedure

1. In the project tree of the TIA Portal, open the "Online access" folder.
2. Open the "TeleService" folder followed by the required system folder.
3. Double-click the command "Assign TS Adapter IE parameters". The associated web interface for assignment of the TS Adapter IE parameters opens. The "logon" for the web interface takes place automatically with the login data of the remote connection.
4. Set the required parameters in the individual tabs of the dialog.
5. Confirm your entries with "Save settings".

Result

The configured parameters are saved in the non-volatile memory of the TS Adapter IE. Parameter assignment is then complete.

14.7.3.6 Establishing a dial-up connection to a remote system

Establishing a dial-up connection

Introduction to establishing a remote dial-up connection

A dial-up connection is established when you use TeleService to dial into a remote system via a telephone network. The programming device/personal computer is connected to the telephone network with TeleService via a modem. At the other end, the automation system is connected to the telephone line via a configured TS Adapter and a modem.

Requirements

A local modem is installed and configured.

The TS Adapter is located in the remote system.

A remote modem is installed and parameters have been assigned.

Proceed as follows:

1. In the project tree of the TIA Portal, click the "Online access" folder.
2. Click on the "TeleService" folder it contains.
3. Double-click the "Set up/close remote connection" entry. The "Set up remote connection to the remote system" dialog opens.
4. In the "Adapter type" drop-down list, select the adapter type used.
5. Under "Connection type", select "Dial-up connection".
6. Select the modem you are using under "Local settings".
7. Enter the phone number to be dialed in the appropriate box or open the phone book by clicking on the button behind it and take the desired phone number from the phone book.
8. Enter your user name and the corresponding password.
9. If you want a "Connection setup with callback", select the appropriate option button.
10. Click the "Connect" button to establish the required remote connection. This button only becomes active when you have entered all the parameters needed to establish a remote connection. Any remote connection is displayed under "Status".

Result

The dial-up connection to the desired system is established.

The dialog closes as soon as the dial-up connection is established. The following message appears in the status bar of the TIA Portal: "Remote connection is up". You can now use the remote connection with the TIA Portal and communicate with the automation system.

Connection cannot be established

If the connection cannot be established, try to find the cause using the "Notes on troubleshooting".

Terminating the connection

Once you have finished editing the remote system, exit the remote connection in the project tree by double-clicking on the entry "Establish/disconnect remote connection".

By exiting the TIA Portal, you are also terminating the remote connection.

Terminating a dial-up connection

Terminating an active dial-up connection

Note

Terminating the dial-up connection

You should go offline in the TIA Portal before you terminate the remote connection.

Proceed as follows:

1. Double-click the "Set up/close remote connection" entry in the TIA Portal.
2. Confirm the prompt in the dialog that follows with "Yes".

Result

The connection is terminated.

14.7.4 Remote VPN connections

14.7.4.1 Basics for establishing a VPN connection

Using a TS Adapter IE Advanced for VPN connections

A TS Adapter IE Advanced is required to establish a VPN connection using TeleService.

VPN definition

VPN stands for "Virtual Private Network". This is a private network of computers based on a public network infrastructure.

VPN is used to create a connection that is as secure as possible between devices in a private network at different locations and the gateway of another network.

A VPN device has direct access to the gateway in the other network over an encrypted connection.

VPN connections

When a VPN connection is established with TeleService, a CA certificate and a unique fingerprint generated from it are used for the unique identification of the TS Adapter IE Advanced. The VPN connection allows a communication that is secure from tapping and manipulation between the remote PC and the TS Adapter.

If you want to establish a VPN connection to multiple remote networks, you will require a separate TS Adapter IE Advanced as an interface for each network. The networks connected by the TS Adapter can used identical IP addresses internally. Only the external IP addresses (WAN) of the TS Adapters need to be different.

Only one VPN connection to a TS Adapter is possible at any one time.

Note

Authentication

Please not that the CA certificate exclusively authenticates the TS Adapter IE Advanced.

As in the case of dial-up connections, users are authenticated via the login (user name and password).

Therefore, use only a secure password for the login.

See also

- Establishing VPN connections (Page 6613)
- Terminating VPN connections (Page 6614)

14.7.4.2 Basics of CA certificates

Introduction

To establish a secure VPN connection with TeleService, you need to generate a CA certificate with a unique fingerprint when you configure the TS Adapter IE Advanced.

You can install this CA certificate on every PC that should have access to the TS adapter IE advanced.

- Using the automatic download in the TeleService connection dialog, by entering the fingerprint for the CA certificate
- Manually by using the Microsoft® Management Console.

Definition of CA certificate

A CA certificate is a digital certificate issued by a "certificate authority" or "certification authority", hereinafter referred to as "CA". In the case of the TS Adapter IE Advanced, self-signed certificates are used; the certification authority in this case is the TS Adapter IE Advanced itself.

Certificates for "SSTP" (Secure Socket Tunneling Protocol) and "HTTPS" (Hypertext Transfer Protocol Secure) are derived from the CA certificate.

CA certificates contain a "key" and additional information used for authentication and for encrypting and decrypting confidential data. Additional information includes the validity period, references to certificate revocation lists, etc. which are included in the certificate by the CA.

Using CA certificates with TeleService

A CA certificate with a unique fingerprint is generated by the TS Adapter to uniquely identify the TS Adapter IE Advanced as connection partner for the remote PC.

This CA certificate must be saved in the Windows certificate store of your programming device/ PC before you can establish a VPN connection. When you access the Web server using a direct connection, a security warning will appear if no CA certificate exists. In such cases, however, you can choose to ignore the warning and allow the connection.

Note

Handling CA certificates

Specialist knowledge of the operating system is required for handling CA certificates. CA certificates should only be managed by trained personnel!

You require administrator rights to manage CA certificates.

Definition of fingerprint

The fingerprint is a 20-byte hexadecimal expression. It provides a unique value for a CA certificate and is used to identify a specific CA certificate.

A fingerprint is calculated dynamically using the SHA-1 algorithm and is not physically contained in the CA certificate.

Use of fingerprints with TeleService

The CA certificate is used to uniquely identify the TS Adapter IE Advanced as connection partner. A unique 20-byte fingerprint of this certificate is generated each time a CA certificate is generated by the TS Adapter IE Advanced. It is calculated automatically by the TS Adapter IE Advanced when the certificate is generated. Each certificate has a specific, unique fingerprint. This fingerprint must be transferred securely to your computer, for example by phone or in an encrypted e-mail. You need to enter this fingerprint in the connection dialog when establishing a VPN connection with TeleService, unless the CA certificate is already saved on your PC in the Windows certificate store.

You will find the fingerprint for your TS Adapter IE Advanced in the web interface for the TS Adapter IE Advanced. To open the web interface, double-click the command "Assign TS Adapter IE parameters" in the device list in the TIA Portal. Perform the Web login to view the fingerprint in the "Security > Certificates" tab.

CA certificate download during connection establishment

When the connection is established, TeleService checks whether a suitable CA certificate is installed in the Windows certificate store of your programming device/PC. If an appropriate certificate is found, the VPN connection is established as an SSTP connection (Secure Socket Tunneling Protocol).

If no appropriate CA certificate is found, the CA certificate is first loaded by the corresponding TS Adapter IE Advanced. The TS Adapter IE Advanced is called by the remote address entered in the connection dialog. If this download of the CA certificate is successful, the fingerprint of the downloaded CA certificate is calculated and compared with the fingerprint entered in the connection dialog. If the two fingerprints match, a dialog opens and you are prompted to save the CA certificate in the Windows certificate store of your programming device/PC. You require administrator rights to save the CA certificate.

The VPN connection is then established.

See also

- Installing CA certificates for VPN connections (Page 6609)
- Deleting CA certificates for VPN connections (Page 6612)

14.7.4.3 Installing CA certificates for VPN connections

Installing CA certificates

To establish a secure VPN connection between your programming device/PC and a remote system with TeleService, you require a valid CA certificate generated by the TS Adapter IE Advanced. This must be saved in the Windows certificate store of your programming device/PC.

A CA certificate can be installed manually or using an automatic download.

CA certificates are managed with the Microsoft® Management Console.

Note

Handling CA certificates

Specialist knowledge of the operating system is required for handling CA certificates. CA certificates should only be managed by trained personnel!

You require administrator rights to manage CA certificates!

Requirement

A CA certificate has yet to be installed in the Windows certificate store on your computer.

Installing CA certificates using the automatic download

Proceed as follows:

1. Log on to the system as administrator.
2. Transfer the fingerprint for the CA certificate from the TS Adapter IE Advanced to your computer using a "secure route", for example by phone or in an encrypted e-mail.
   You will find the fingerprint for your TS Adapter IE Advanced in the web interface for the TS Adapter IE Advanced. To open the web interface, double-click the command "Assign TS Adapter IE parameters" in the device list in the TIA Portal. Perform the Web login to view the fingerprint in the "Security > Certificates" tab.
3. In the project tree of the TIA Portal, click the "Online access" folder.
4. Click on the "TeleService" folder it contains.
5. Double-click the "Set up/close remote connection" entry. The "Set up remote connection to the remote system" dialog opens.
6. Select the "TS Adapter IE" as Adapter type from the drop-down list.
7. Under "Connection type", select "VPN".
8. Enter the IP address / DNS name for the TS Adapter IE Advanced to be contacted in the relevant field. Alternatively, you can apply any existing data from the phone book by clicking the corresponding button.
9. Enter your user name and the corresponding password.
10. Copy the fingerprint displayed in the web interface for the TS Adapter IE Advanced to the "Fingerprint" column.
11. Click the "Connect" button to establish the required remote connection. This button only becomes active when you have entered all the parameters needed to establish a remote connection.

12. No valid CA certificate is found, because no CA certificate has been installed on your PC yet.
    A "normal" connection (not a VPN connection) is therefore established and the certificate required is downloaded from the TS Adapter IE Advanced to the working memory of your computer. The fingerprint is then calculated (SHA-1 algorithm) and compared with the fingerprint entered in the connection dialog. If the two fingerprints match, a dialog opens and you are prompted to save the CA certificate in the Windows certificate store of your programming device/PC.
13. Confirm that you want to save the CA certificate.

Note

Restart of STEP 7 required after renewal of CA certificate

STEP 7 must be restarted after renewal of the CA certificate on the TS Adapter IE Advanced before the remote connection can be established with the help of the new fingerprint. Otherwise, it may not be possible to store the CA certificate in the Windows certificate memory and the remote connection cannot be established.

Result

The VPN connection to the required TS Adapter IE Advanced is established. The dialog closes once the connection is established.

Manual installation of CA certificates

Proceed as follows:

1. Log on to the system as administrator.
2. Open the Windows Certificate Manager on your programming device/PC with the Microsoft® Management Console.
   Click "Start", enter "mmc" in the search field and press ENTER.
   The console will open.
3. Open the "File" menu and click "Add/remove snap-in...".
   The "Add/remove snap-in..." dialog will open.
4. Double-click "Certificates" in the "Snap-in" list and select "Computer account" in the dialog that opens.
5. Select "Local computer" in the next dialog and click "Finish" and "OK".
   The console root opens and the "Certificates (local computer)" folder appears.
6. Open the displayed "Certificates (local computer)" folder and click "Trusted certification authorities".
7. Click the "Certificates " folder and use the shortcut menu to access the command "All tasks" > "Import...".
8. Note the information that appears in the "Certificate - Import wizard" dialog and click "Next".
9. In the dialog that follows, click "Browse ..." and select the required CA certificate.
10. Then click "Next" twice and subsequently "Finish" to install the CA certificate.

Result

The selected CA certificate is installed at the specified location in the Windows certificate memory.

Note

Additional information ...

... on installing CA certificates can be found using the "F1" button in the online help of your operating system.

14.7.4.4 Deleting CA certificates for VPN connections

Deleting CA certificates

Proceed as follows:

1. Log on to the system as administrator.
2. Open the Windows Certificate Manager on your programming device/PC with the Microsoft® Management Console.
   Click "Start", enter "mmc" in the search field and press the ENTER KEY.
   The console will open.
3. Open the "File" menu and click "Add/remove snap-in...".
   The snap-in selection dialog will open.
4. Double-click "Certificates" in the "Snap-in" list and select "Computer account" in the dialog that opens.
5. Select "Local computer" in the next dialog and click "Finish" and "OK".
   The console root opens and the "Certificates (local computer)" folder appears.
6. Open the displayed "Certificates (local computer)" folder and click "Trusted certification authorities".
7. Open the "Certificates" folder, select the required CA certificate and click "Delete" in the shortcut menu.
8. Confirm the next prompt with "Yes".

Result

The selected CA certificate is deleted from the list of available certificates.

14.7.4.5 Establishing a VPN connection to a remote system

Establishing VPN connections

Introduction to establishing VPN connections

A VPN connection is established when you use TeleService to connect to a remote system over the Internet.

Your programming device/PC with installed TIA Portal is connected to the Internet at one end for this. At the other end, the automation system is connected to the Internet via the WAN (Wide Area Network) interface of the configured TS Adapter IE Advanced.

Requirement

Your programming device/PC is connected to the Internet.

There is a TS Adapter IE Advanced in the remote system.

The TS Adapter IE Advanced has been configured and is connected to the Internet.

The CA certificate required for identifying the TS Adapter has been generated and installed in the Windows certificate store of your programming device/PC.

Proceed as follows:

1. In the project tree of the TIA Portal, click the "Online access" folder.
2. Click on the "TeleService" folder it contains.
3. Double-click the "Set up/close remote connection" entry. The "Set up remote connection to the remote system" dialog opens.
4. Select the "TS Adapter IE Advanced" as "Adapter type" from the drop-down list.
5. Enter "VPN" as "connection type".
6. Enter the IP address / DNS name for the TS Adapter IE Advanced to be contacted in the relevant field. Alternatively, you can apply any existing data from the phone book by clicking the corresponding button.
7. Enter your user name and the corresponding password.
8. Click the "Establish" button to establish the required VPN connection. This button only becomes active once you have entered all the parameters needed for establishing the remote connection.

Result

The VPN connection to the required system is established. "Status" indicates the progress in establishing the connection. The dialog closes once the VPN connection is established. The following message appears in the status bar of the TIA Portal: "Remote connection is up". You can now use the remote connection with the TIA Portal and communicate with the automation system.

Connection cannot be established

If the connection cannot be established, try to find the cause using the "Notes on troubleshooting".

Note

Rules for IP addresses

- Only use IP addresses which have not yet been assigned in the plant network.
- If the IP address configured for the TS Adapter IE Advanced has already been assigned in the system network, the TS Adapter IE Advanced can only be addressed by its MAC address.

See also

Installing CA certificates for VPN connections (Page 6609)

Terminating VPN connections

Terminating an active VPN connection

Note

Terminating the VPN connection

You should go offline in the TIA Portal before you terminate the VPN connection.

Proceed as follows:

1. Double-click the "Set up/close remote connection" entry in the TIA Portal.
2. Confirm the prompt in the dialog that follows with "Yes".

Result

The VPN connection is terminated.

14.7.4.6 TS Adapter IE Advanced

Short description of the TS Adapter IE Advanced

TS Adapter IE Advanced

The TS Adapter IE Advanced has the following properties:

- Direct connection over Industrial Ethernet (IE), 2 ports
- WAN (Wide Area Network) interface for VPN connections
- Firmware update supported
- Plug-in modules
- Parameters are assigned via a Web interface.

Note

Further information on the TS Adapter IE Advanced

For detailed information on your TS Adapter, please refer to the documentation supplied with your TS Adapter.

Connection types

TS Adapter IE Advanced connection types

The following diagrams show the types of connection possible with the TS Adapter IE Advanced.

Direct connection

In the direct connection to the programming device/PC, you can set the TS Adapter IE Advanced parameters over the Ethernet.

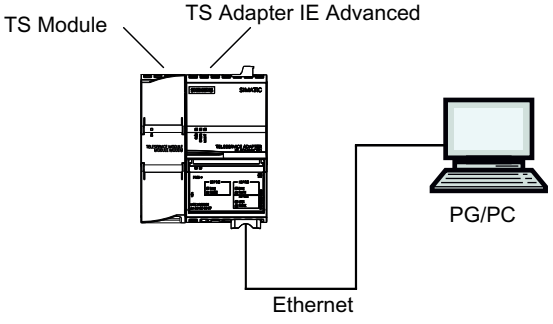


Figure 14-5 TS Adapter IE Advanced - direct connection

Connection to the Internet (DSL modem/router)

In order to connect to the Internet, you must operate the DSL modem/router at the WAN connection of the TS Adapter IE Advanced.

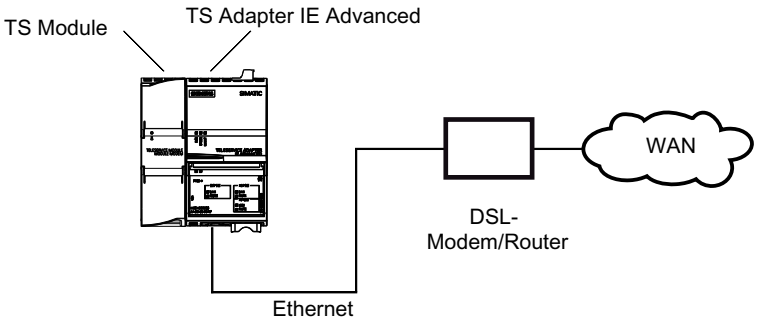


Figure 14-6 TS Adapter IE Advanced - connection to the Internet

Connection to the company network (Intranet)

In order to connect to the Intranet, you must operate plant network (Ethernet) at one of the two LAN connections of the TS Adapter IE Advanced.

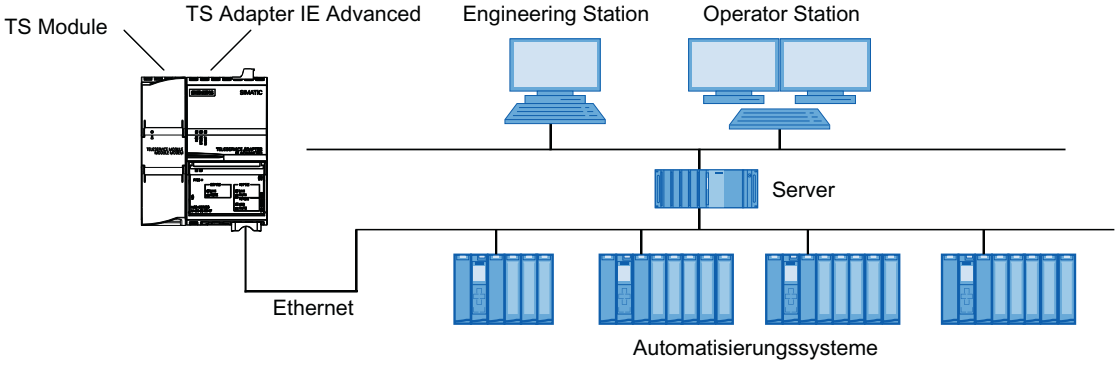


Figure 14-7 TS Adapter IE Advanced - connection to company network (Intranet)

Additional information

For more detailed information on the TS modules, please refer to the documentation supplied with your TS Adapter IE Advanced.

TS Adapter IE Advanced parameter assignment options

Basics on configuring the TS Adapter IE

The TS Adapter IE Advanced is configured via a Web interface.

Web help in the Web interface is available for assigning TS Adapter IE Advanced parameters.

Parameter assignment options include:

- Reconfiguration
- Restoring default parameter assignment
- Importing adapter parameters
- Exporting adapter parameters

Note

Parameter assignment

Configure your TS Adapter IE Advanced in accordance with the documentation supplied. This details the exact procedure for parameter assignment.

Parameter assignment for TS Adapter IE Advanced

Introduction

The TS Adapter IE Advanced can be configured in both direct connection mode and via an existing remote connection.

Both parameter assignment options are described below.

Specific details on assigning parameters for the TS Adapter IE Advanced are available in the TS Adapter IE Advanced documentation.

Assigning parameters for the TS Adapter IE Advanced in direct connection

Requirement

There is a LAN connection to your TS Adapter IE Advanced.

Procedure

To assign parameters for the TS Adapter IE Advanced, follow the steps below:

1. In the project tree of the TIA Portal, open the "Online access" folder.
2. Double-click on the Ethernet port of your computer.
3. Double-click on the "Display accessible nodes" command. The TS Adapter IE Advanced is then displayed.
4. Double-click on the "TS Adapter IE Advanced" folder and then on "Online and diagnostics", and assign the desired IP address to the TS Adapter in the dialogs that follow. Please note that the IP address of the programming device/PC interface card must be located in the same subnet as the IP address that you assign for the TS Adapter IE Advanced.

5. Update the view in the project tree for the "Accessible nodes", so that the TS Adapter IE Advanced is displayed with the newly assigned IP address.
6. Open the folder "TS Adapter IE Advanced" in the device list.
7. Double-click the command "Assign TS Adapter IE parameters". The corresponding web interface opens for assigning TS Adapter parameters.
8. Complete the "logon" for the web interface.
9. Set the required parameters in the individual tabs of the dialog.
10. Confirm your entries with "Save settings".

Result

The configured parameters are saved in the non-volatile memory of the TS Adapter IE Advanced. Parameter assignment is then complete.

Assigning parameters for the TS Adapter IE Advanced using a remote connection

Requirement

There is an established remote connection to a TS Adapter IE Advanced..

Procedure

1. In the project tree of the TIA Portal, open the "Online access" folder.
2. Open the "TeleService" folder followed by the required plant folder.
3. Double-click the command "Assign TS Adapter IE parameters". The associated web interface for assignment of the TS Adapter IE parameters opens. The "logon" for the web interface takes place automatically with the login data of the remote connection.
4. Set the required parameters in the individual tabs of the dialog.
5. Confirm your entries with "Save settings".

Result

The configured parameters are saved in the non-volatile memory of the TS Adapter IE Advanced. Parameter assignment is then complete.

14.7.5 CPU controlled TeleService remote connections

14.7.5.1 Overview of CPU controlled remote connections

Introduction

TeleService offers a range of options for establishing remote connections; these differ according to the CPU used. The initiative for establishing a connection starts from the CPU. The communication instructions given below are used for the individual connection options.

Connection establishment with S7-300/400 CPUs

The following communication instructions are available:

- Communication instruction "PG_DIAL": Establish remote connection to programming device/PC
- Communication instruction "SMS_SEND": Send text message (SMS)
- Communication instruction "AS_DIAL": Establish remote connection to AS
- Communication instruction "AS_MAIL": Transfer email

Connection establishment with S7-1200 CPUs

The following communication instruction is available:

- Communication instruction "TM_MAIL": Transfer email

Connection establishment with S7-1500 CPUs

The following communication instruction is available:

- Communication instruction "TMAIL_C": Transfer email

Note

Description of individual communication instructions

More detailed information on the available communication instructions can be found in the information system of the TIA Portal in the "References > Communication > TeleService" directory.

See also

TS Adapter IE parameter assignment options (Page 6602)

TM_MAIL: Transfer email (Page 4676)

14.7.5.2 Establishing a connection from and to remote systens (PG-AS-remote coupling)

Remote plant access to a programming device/personal computer

Introduction

You can establish a remote connection to and communicate with a remote system using the application TeleService and a TS Adapter MPI. The initiative for establishing the remote connection comes from the programming device/personal computer.

However, events which require rapid intervention often occur at a remote system. In such cases, the automation system can initiate a remote connection to a programming device/ personal computer if an asynchronous event occurs.

The graphic below shows the components which are required for establishing a connection from a plant to a programming device/personal computer.

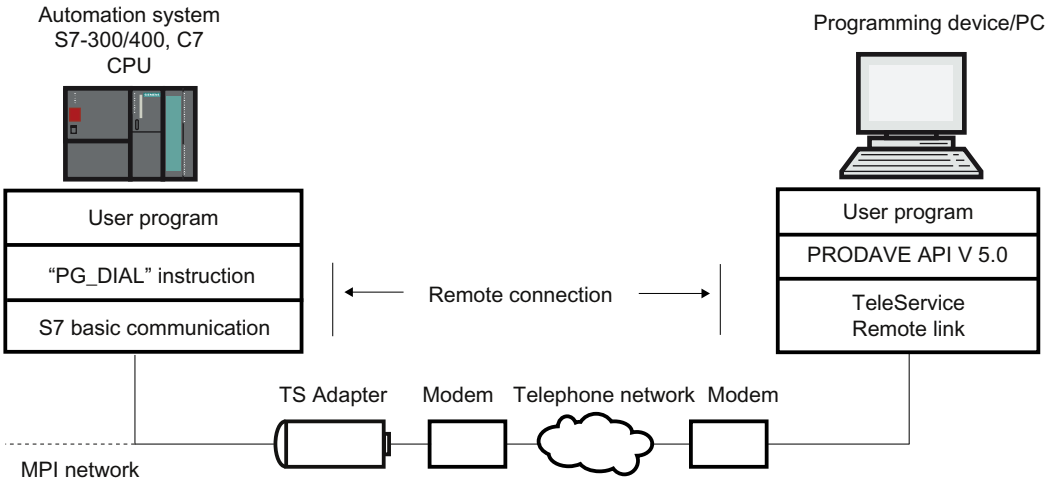


Figure 14-8 How the "PG_DIAL" communications instruction works

Requirements for establishing a connection

Introduction

Certain hardware and software requirements must be fulfilled if a remote system is to establish a remote connection to a programming device/personal computer. These requirements are detailed below.

Hardware requirements:

The only hardware required for establishing a remote connection from a remote system to a programming device/personal computer is that needed for accessing the remote system from the programming device/personal computer.

Your user program calls the "PG_DIAL" communication instruction to establish the connection. This can only be executed on an S7-300 or S7-400 CPU on which S7 basic communication is implemented.

A TS Adapter I , version 5.0 or later, or a TS Adapter II must be used.

Software requirements on the system side:

The "PG_DIAL" communication instruction is included in the scope of delivery of TeleService and is installed when the TIA Portal is installed. You will find the installed communications instructions in the task card of the block editor in the folder "Communication > TeleService".

If a remote system is to establish a remote connection to a programming device/PC, the user program of the system must call the "PG_DIAL" communication instruction.

Software requirements on the programming device/PC side:

You require a software product on the programming device/PC which, with TeleService, waits for a call from a remote system, recognizes this call and informs your user program.

14.7.5.3 Data exchange between remote systems (AS-AS-remote coupling)

AS-AS remote link basics

Introduction

The AS-AS remote link allows two automation systems to exchange process data via the telephone network.

Requirement

Communication instruction "AS_DIAL" is available if you use a CPU from the S7-300/400 family.

Definition: Local and remote automation system

- The automation system from which the initiative to establish the remote connection originates is described as **local**.
- The automation system to which the remote connection is to be established is described as **remote**.

Data exchange over the AS-AS remote link

Data exchange is carried out using specific communication instructions for non-configured S7 connections. Use the communication instruction "AS_DIAL" to establish a remote connection to the automation system.

More detailed information on establishing the connection can be found in the information system in the directory "References > Communication > TeleService".

The following graphic shows the components required for establishing a connection from a local to a remote automation system.

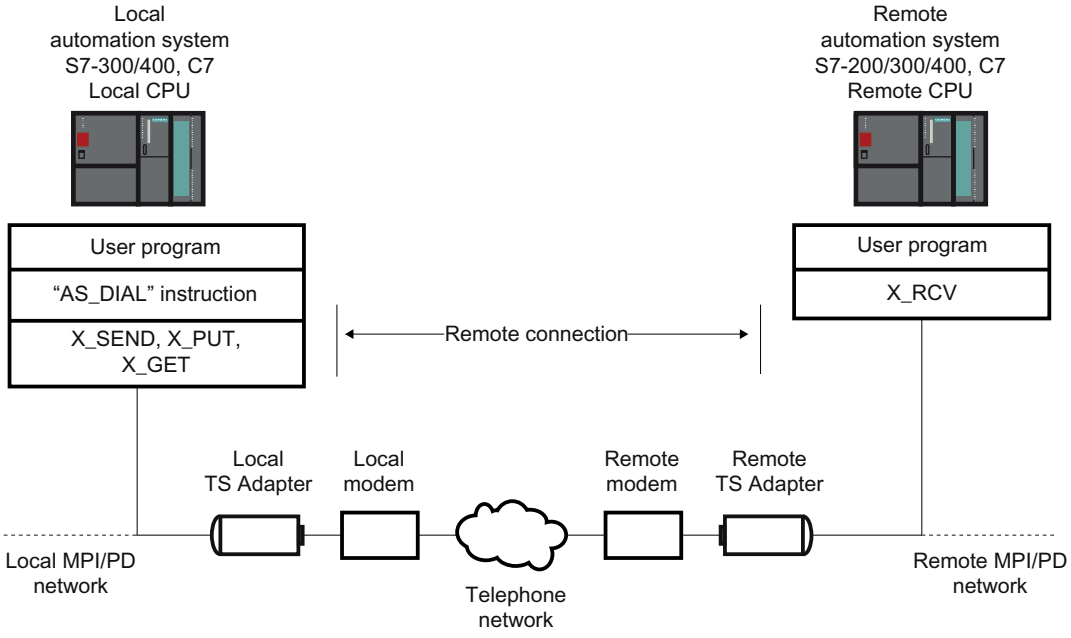


Figure 14-9 Data exchange over the AS-AS remote link

Hardware and software requirements for AS-AS remote link

Introduction

Certain hardware and software requirements must be fulfilled before a local automation system can establish a remote connection to a remote automation system. These requirements are detailed below.

Hardware requirements

The only hardware you need for transferring process data from a local to a remote automation system is that also needed for accessing the respective automation system from the programming device/personal computer.

To establish and terminate the remote connection, the user program of the TIA Portal of the local CPU calls a communication instruction. This communication instruction can be executed on an S7-300/400 CPU or a C7 CPU. The communication instruction requires S7 basic communication to be implemented on the CPU. The remote CPU must also support S7 basic communication.

A TS Adapter I, version V5.1 or later, or a TS Adapter II must be used.

Software requirements

The "AS_DIAL" communication instruction is included in the scope of delivery of TeleService. When installed, the instruction is integrated in the library of the TIA Portal in the task card of the Communication instructions folder under TeleService. In order to establish and terminate a remote connection to a remote automation system from a local automation system, the "AS_DIAL" communication instruction must be called in the user program of the TIA Portal on the local CPU.

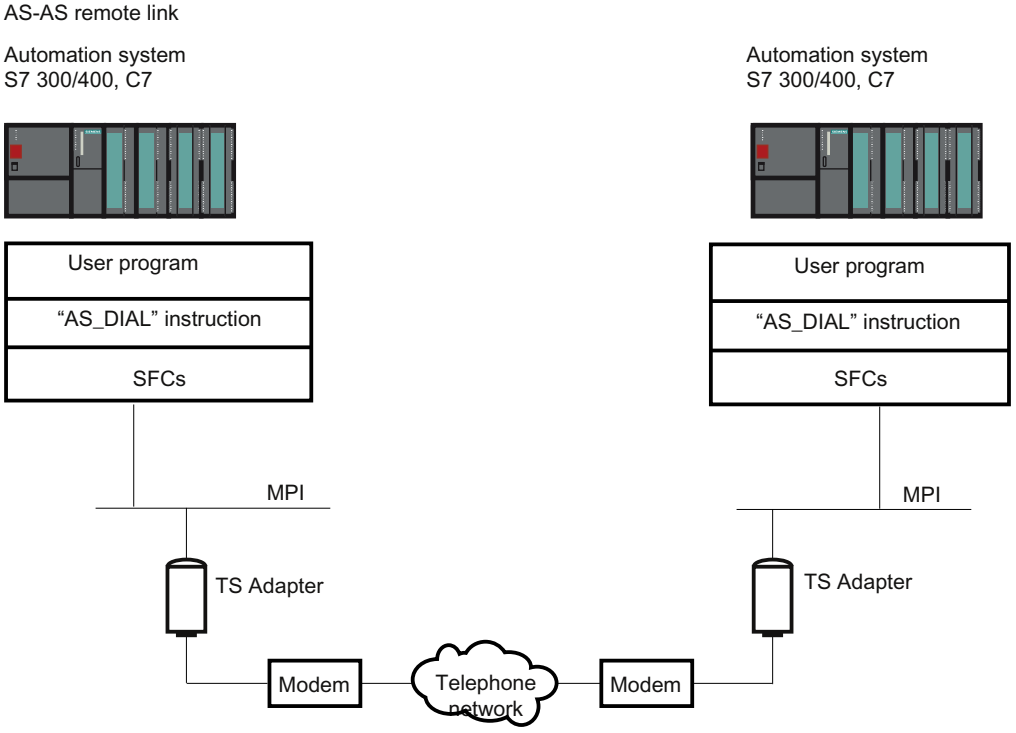


Figure 14-10 Hardware and software requirements for AS-AS remote link

14.7.5.4 Send SMS from a system

Requirements for sending an SMS

Introduction

Certain hardware and software requirements must be fulfilled before a system can send an SMS. These requirements are detailed below.

Hardware requirements

To send an SMS from a system, you will require a GSM wireless modem and a TS Adapter MPI.

A TS Adapter I, version V5.2 or later, or a TS Adapter II must be used.

Software requirements on the system side

The "SMS_SEND" communication instruction is included in the scope of delivery of TeleService. When installed, the instruction is integrated in the library of the TIA Portal in the task card of the Communication instructions folder under TeleService. If a system is to send an SMS, the user program of the system must call the "SMS_SEND" communication instruction.

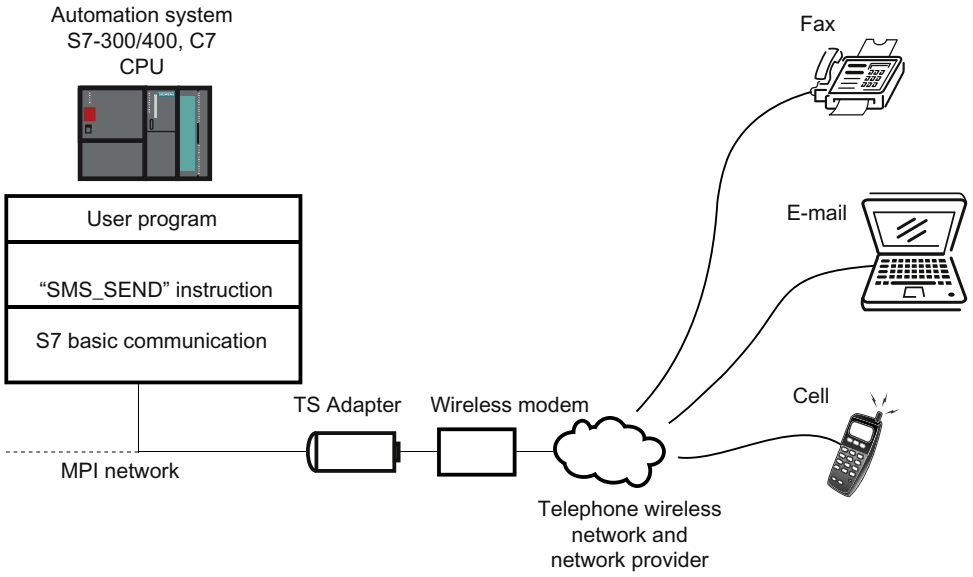


Figure 14-11 How the "SMS_SEND" communication instruction works

Note

You may be able to send an SMS not only to a cell phone but also to an email address or a fax device by using additional services offered by the cell phone service provider.

14.7.5.5 Send an email from a system

Requirements for sending e-mails

Introduction

The following hardware and software requirements must be fulfilled if a system is to send an email:

Hardware requirements

You need a TS Adapter IE to send an email from a system and one of the CPUs listed below:

- a CPU 31x2 PN/DP as of firmware version V2.5
- a CPU 41x-3 PN/DP
- a CPU of the S7-1200 series
- a CPU of the S7-1500 series

Software requirements on the system side

Various communication instructions are included in the scope of delivery of TeleService depending on the CPU. When installed, they are integrated in the library of the TIA Portal in the task card of the Communication instructions folder under TeleService.

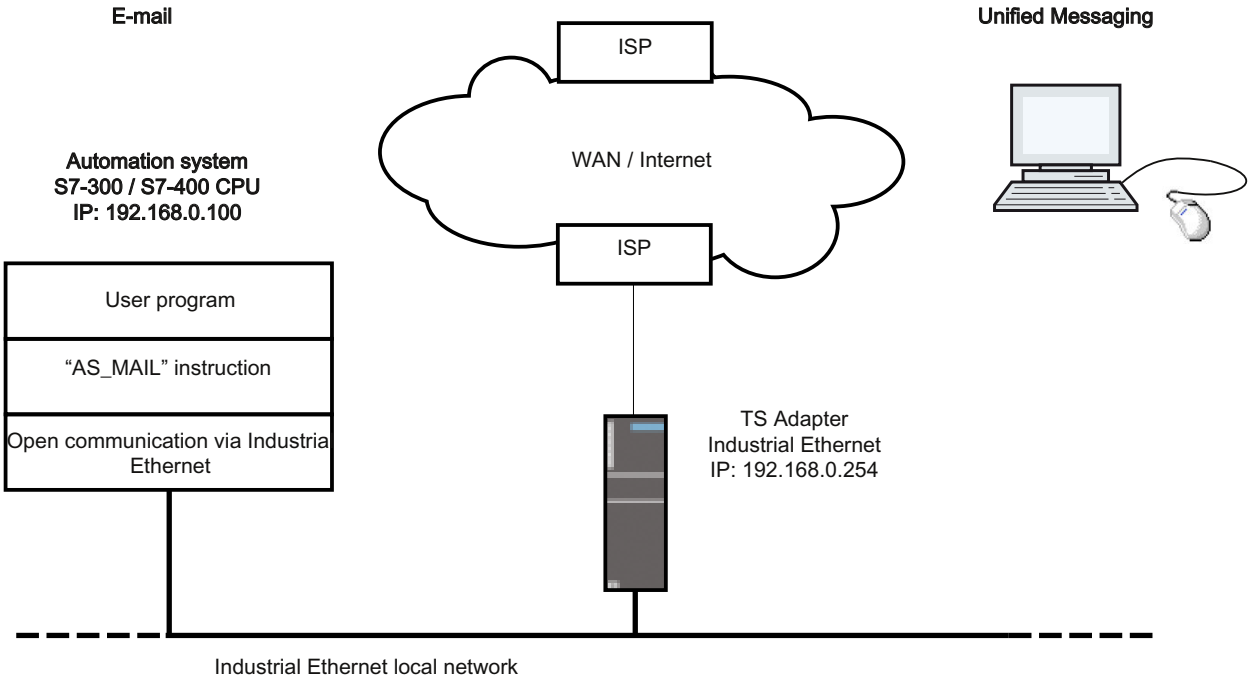
If a system is to send an email, the user program of the system must call the corresponding CPU-dependent communication instruction.

The following communication instructions are available for sending an email:

- S7-300/400 CPUs: Uses the communication instruction "AS_MAIL": Transfer email
- S7-1200 CPU V2.x and V3.x: Uses the communication instruction "TM_MAIL": Transfer email
- S7-1200 CPU >V4.0: Uses the communication instruction "TMAIL_C": Transfer email
- S7-1500 CPUs: Uses the communication instruction "TMAIL_C": Transfer email

The respective communication instruction transfers an email from a CPU to a mail server by means of the Simple Mail Transfer Protocol (SMTP) with the "LOGIN" authentication method. The data is transferred unencrypted with this SMTP process.

The figure below shows an example with the "AS_MAIL" communication instruction:



The "Use gateway/router" property must also be set for the Ethernet interface in the configuration of the CPU on which the "AS_MAIL" communication instruction runs. (available in the device configuration under Ethernet addresses and there under IP protocol)The IP address of the Ethernet interface of the TS Adapter IE is to be specified as "Address".

Note

You can find further information in the task card in the "Communication instructions" folder under TeleService.

14.7.6 Notes on troubleshooting

14.7.6.1 General information on troubleshooting for modem problems

Introduction

The information below should help you establish and eliminate the causes of any modem problems.

1. Enable "Record log file" for data traffic between PG/PC and modem. The entries in this file can provide valuable information for determining the cause of errors.
2. Switch on the loudspeaker on your local modem. Select a volume loud enough to be clearly heard.
   You can then hear whether:
   - there is a dial tone at the connection
   - the modem called is busy
   - the modem called accepts the call.

Common modem problems

Modem connection problems are among the most common modem problems:

- Modem connection is not established
- Modem connection is interrupted

The topics below contain tables detailing possible causes and providing information on eliminating the error in question.

See also

Dial-up connection to the TS Adapter is not established (Page 6628)

Dial-up connection from the TS Adapter is not established (Page 6630)

Modem connection is interrupted (Page 6631)

Modem alarms (Page 6632)

Recording a log file for the modem (Page 6627)

14.7.6.2 Recording a log file for the modem

Introduction

It is advisable to record a log file as this makes it easier to find the causes of faults in a modem.

Procedure:

Proceed as follows:

1. Activate the properties dialog of the modem used via the "Phone and modem options" option in the Control Panel.
2. Check the settings of the "Log" option in the "Diagnostics" tab and, if necessary, change the log file settings so that the file is recorded.

Result:

Activity between the programming device/personal computer and the modem are entered in the log file. If there are problems establishing the connection, you can evaluate the data recorded in the log file to determine the cause of the error.

14.7.6.3 Dial-up connection to the TS Adapter is not established

Dial-up connection to TS Adapter is not established

The table below sets out possible causes and how to eliminate them if no remote connection to the TS Adapter can be established.

Possible cause	Check/Remedy
Cabling faulty	• Are all the connecting cables connected correctly? • Are the connectors loose?
Dial parameters for main telephone line and extension incorrectly set	• Do the properties and dial parameters of your modem match the phone connection to main phone line or extension? • Do not specify a dial-out code in the "Dial parameters" dialog if you operate your modem on a local loop (main telephone line). The fields for the dial-out code for local calls and long-distance calls must be empty.
Dialing mode incorrectly set	• Is the correct dialing mode (tone/pulse) set in the dialog for the dial parameters of your modem? • Use a connected telephone to check the connection on which you want to operate the modem. You should hear crackling noises on the telephone during pulse dialing and tones of varying pitches during tone dialing. Set the corresponding dialing mode in the modem dial parameters.

Possible cause	Check/Remedy
Dial disable active	• The dial disable function is a country-specific modem property which, depending on the modem, becomes effective after one or more unsuccessful attempts to establish a connection. • If your modem still does not respond after several attempts to dial, the dial disable function may be active. Characters are still sent to the modem after the dial command but the modem does not start the dialing process. The driver receives a general error message. • Refer to the modem documentation for information on how the dial disable function is implemented for your modem. • Create a log file (Page 6627) (modemlog.txt) in which the activities between the programming device/personal computer and modem are recorded. Then check whether the file contains an entry caused by dial disable (e.g. DELAYED).
Phone connection defective or busy	• Connect a phone and check whether a dial tone can be heard on this connection. • Any analog phone connected on the same connection must be hung up. You cannot establish an additional modem connection on this connection if there is an existing phone connection.
Serial parameters set incorrectly	• Are the correct values entered in the "Settings" tab for modem properties (8 data bits, no parity, 1 stop bit)? Is the correct COM interface set in the "General" tab for the modem properties?
Initialization string of the TS Adapter is not suitable for the modem.	• Familiarize yourself with the modem initialization string requirements and set the string accordingly. Procedure for configuring the TS Adapter IE (Page 6600)
Settings for error correction between the modem at the TS Adapter and the modem at the PC/programming device are not compatible.	• Adapt the modem settings. Useful information on configuring the TS Adapter MPI (Page 6595) Restoring the default parameter assignment of a TS Adapter MPI (Page 6597) Procedure for configuring the TS Adapter IE (Page 6596)

14.7.6.4 Dial-up connection from the TS Adapter is not established

No callback from TS Adapter

The table below sets out possible causes and how to eliminate them if there is no callback from the TS Adapter.

Possible cause	Check/Remedy
Errors in the location or call settings in the TS Adapter	Check the TS Adapter parameter assignment: • Are the dialing mode and dial-out code set correctly for your phone connection? • Does the modem at the TS Adapter support the characters configured for the dial-out code? • Is "Wait for dial tone before dialing" deactivated for an extension?
Initialization of modem insufficient	Check the string for modem initialization: • The modem may require a further initialization in order to establish a remote connection. • Properties of the modem initialization string for the TS Adapter MPI
Callback number is incorrect	Check the configuration of the callback number you assigned.

No call from TS Adapter MPI

The table below sets out possible causes and and how to remedy them if there is no call from the TS Adapter MPI.

Possible cause	Check/Remedy
Phone number is incorrect	Is the required number being transferred to the communication instruction "PG_DIAL" ?
TS Adapter MPI parameter assignment incorrect	Check the TS Adapter MPI parameter assignment: • Are the dialing mode and dial-out code set correctly for your phone connection? • Does the modem at the TS Adapter MPI support the characters configured for the dial-out code? • Is "Wait for dial tone before dialing" deactivated for an extension?

14.7.6.5 Modem connection is interrupted

Modem connection is interrupted

The table below sets out possible causes and and how to remedy them if the modem connection is interrupted.

Possible cause:	Check / Remedy:
Metering pulses in the line	Metering pulses will be generated if you have applied to the phone company for a metering clock. This may mean that the modem no longer recognizes the carrier signal and switches off. • Set a longer waiting or switch-off time at the modem. • Have the metering pulse deactivated by the phone company.
Shielding	• Are the connection cables used shielded sufficiently? • Make sure that the modem cables do not run next to power cables and that they are as far as possible from power supply units and monitors.
Protocol timeout	• Set fixed monitoring times.
Automatic connection termination	• Deactivate the option that terminates an existing connection automatically after a specified time without data transfer ("Terminate after idle of ...").
Data flow control deactivated	• Click on the "Extended" button in the "Settings" tab of the modem properties and activate the following options in the dialog displayed (if available and not yet set): – Data flow control – Hardware (RTS/CTS) – Data compression – Error control
Initialization string of the TS Adapter is not suitable for the modem	• Set the modem initialization string in accordance with the following requirements. See also: TS Adapter IE parameter assignment options (Page 6602)

See also

TS adapter MPI configuration options (Page 6595)

14.7.6.6 Checklist for troubleshooting the modem

Introduction

The following list should help you establish the potential cause of any problems with the modem. The help topics below set out how and in which dialogs you define the relevant settings.

Modem connection cannot be established:

- Check the cabling and the connections.
- Check whether the correct dialing mode (tone/pulse) is set.

- If your modem does not react after several attempts to dial, a dial disable function may be active. Familiarize yourself with dial disable on your modem.
- Are you operating your modem on a main telephone line or on an extension line? Configure the properties and dialing parameters of the modem accordingly.
- Enable the log file option in the advanced properties. The next attempt to establish a connection will then be recorded in a file in the Windows directory.
- Ensure that the ISDN TAs used work with the same B and D channel protocol.

The modem connection is terminated:

- Metering pulses can have a negative affect on a connection. Have the pulses deactivated by your telephone company.
- Set fixed monitoring times.
- Deactivate the option that terminates an existing connection automatically after a specified time without data transfer (idle).
- Make sure that you have activated RTS/CTS for data flow control.

14.7.6.7 Modem alarms

Information in the log file

The modem alarms are entered in a log file if you have activated the recording function.

The log file contains the following information:

Alarm:	Possible cause:	Remedy:
NO DIALTONE	A phone call may currently be being carried out on this line.	• Repeat the process once the phone call is over.
NO CARRIER	The device dialed is not ready, is not a modem or cannot establish a connection in the set operating mode.	• Check the numbers and the settings.
BUSY	The device dialed is busy.	• Try again later.
DELAYED: ...	Dial disable	• Refer to the modem documentation for information on how the dial disable function is implemented for your modem and if necessary remove it.

14.7.6.8 Possible error messages with VPN connections

VPN connection to TS Adapter IE Advanced is not established

The table below sets out possible errors and how to eliminate them if no VPN connection to the TS Adapter IE Advanced can be established.

Possible errors	Check/Remedy
The error message "The webpage cannot be displayed" appears when the Web interface is accessed.	• Note: This problem occurs when the Windows certificate store contains a CA certificate with the same name as the CA certificate which has just been generated. • If a CA certificate for this TS Adapter is already installed in the Windows certificate store and you have generated a new CA certificate on the TS Adapter, you need to remove the old CA certificate from the Windows certificate store and install the newly generated CA certificate. • See also: Installing CA certificates for VPN connections (Page 6609) • See also: Deleting CA certificates for VPN connections (Page 6612)
Error message: "Unexpected error in application".	• Check whether other users have been saved in the TS Adapter IE Advanced user database. • The user "Administrator" cannot establish a remote connection.
Error message: "No corresponding CA certificate was found in the Windows certificate store".	• Use the automatic certificate download. Enter the fingerprint in the corresponding field in the connection dialog. • See also: Establishing VPN connections (Page 6613) Or • Export the CA certificate from the Web interface of the TS Adapter IE Advanced and install it manually in the Windows certificate store. • See also: Installing CA certificates for VPN connections (Page 6609)
Error message: "The remote address specified does not correspond to the remote address of the TS Adapter".	• You must use the remote address which you specified in the TS Adapter IE Advanced to establish the connection. • You cannot use the IP address if you have entered the DNS name (and vice versa).
Error message: "The signature of the certificate could not be verified".	• If you have generated a new CA certificate on the TS Adapter IE Advanced, you need to delete the old CA certificate from the Windows certificate store and install the new one. • See also: Deleting CA certificates for VPN connections (Page 6612) • See also: Installing CA certificates for VPN connections (Page 6609)
Error message: "Protocol error"	• Check that the IP address of the service PC is set in the configuration of the TS Adapter IE Advanced (Parameters > Plant network)
Error message: "Connection setup error."	• Check to make sure that the LAN IP-address of the TS Adapter IE Advanced and the IP address of your network cards are in the same subnet. If this is the case, deactivate this network card before you establish the remote connection.
An e-mail cannot be sent.	• Check whether there is a network/Internet connection at the WAN port. • Check whether the mail server is accessible. • Check whether the outgoing connections in the TS Adapter IE Advanced are valid. • Check whether the SMTP port in the firewall is open for outgoing connections.

14.8 Simulating devices with S7-PLCSIM

14.8.1 Simulation of devices

Introduction

You can use the TIA Portal to run and test the hardware and software of the project in a simulated environment. The simulation is performed directly on the programming device or PC. No additional hardware is required.

The simulation software provides a graphical user interface for monitoring and changing the configuration. It differs according to the currently selected device.

Integration in the TIA Portal

The simulation software is fully integrated in the TIA Portal but is only supported by certain devices. Therefore, the button for calling the simulation software is only active if the selected device supports simulation.

The simulation software for some devices requires its own virtual interface to communicate with the simulated devices. The virtual interface can be found in the project tree under the "Online access" entry next to the physical interfaces of the programming device/PC.

Once you have opened the software, additional help on the simulation software is available via a separate link.

See also

Starting the simulation (Page 6634)

14.8.2 Starting the simulation

Some devices can be simulated with additional software. You therefore do not have to have the actual devices to perform comprehensive testing of your project.

Procedure

To start the simulation software, follow these steps:

1. Select the device you want to simulate, for example, in the project tree.
2. Select the "Simulation > Start" command in the "Online" menu.
   This calls the simulation software.

See also

Simulation of devices (Page 6634)

Using Team Engineering 15

15.1 Shared commissioning of projects

15.1.1 Basics for shared commissioning

Introduction

As part of Team Engineering , you have the option to perform shared commissioning of projects. Several editors can access one CPU at the same time with up to five Engineering Systems (ES) in this process.

A special advantage is that parts of a master project can be edited independently and offline at the same time during the commissioning phase. The changes of the other editors are displayed in the dialog "Software synchronization" during loading and synchronized automatically, if possible.

Depending on the firmware version of the CPU in use, certain online functions can also be executed at the same time from several Engineering Systems on the shared CPU, for example:

- Monitoring blocks on the CPU
- Modifying and forcing blocks on the CPU
- Trace functions

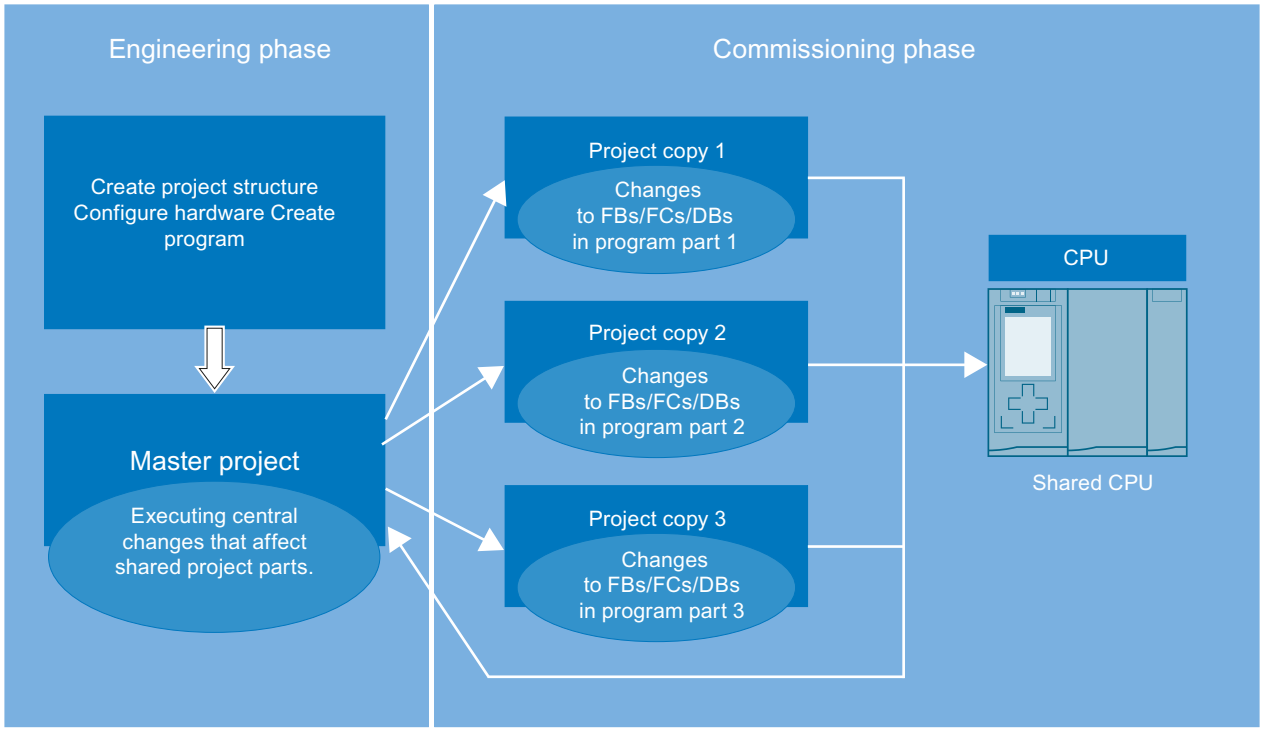
The following online functions cannot be executed at the same time:

- Loading: Only one ES can load to the CPU.

You can also find additional information on the topic of Team Engineering at Service and support in Siemens Industry online support (http://support.automation.siemens.com/WW/view/en/82142829).

Creating a master project

A "master project" structured according to the specified rules (Page 6642), which already contains the complete configured hardware configuration with all required tags and blocks, is used as the basis for the shared commissioning. This project is loaded in the jointly used CPU and then distributed as "master project" by means of project copies to up to five participating Engineering Systems.



Working with project copies

Each Engineering System only processes the parts it was assigned within the specific project copy. This is especially important to avoid conflicts during loading and unintentional overwriting of blocks that were already modified by other Engineering Systems.

Each ES loads its parts to the shared CPU after editing. The TIA Portal helps you during download to the CPU. The changes made since the last download are displayed in the dialog "Software synchronization before loading to a device" (hereafter referred to as "synchronization dialog"), by means of an online-offline comparison. You are provided recommendations during download to the CPU about how you can synchronize the offline data you have edited with the existing online data.

If there are "competing changes", you may have to synchronize them manually.

Competing changes are:

- when different editors edit the same block in different project copies at the same time.
- when PLC tags (I, Q, M, T, C) are added, changed or deleted in the project copies.

- when blocks that have relations to the hardware configuration are edited in the project copies.
- when F blocks are edited in the project copies.

Each editor loads their changes only after all conflicts have been removed.

The procedure for editing and loading of project copies can be repeated as often as necessary by the individual Engineering Systems until the shared commissioning is complete.

Integrating project copies into the master project once again

The individual project copies must once again be integrated into the master project at the end of commissioning at the same time. This way you also back up the data that only exist in the offline projects and have not been downloaded to the CPU. These include, for example, the project languages and text lists created in the project copies.

To do this, open the master project and one project copy as reference object. You determine the differences between the two projects by performing an offline/offline comparison using the comparison editor. Copy the edited objects from the respective project copy to the master project. Repeat this step for all project copies.

When the shared commissioning is completed, you have an executable master project with all created project data.

The integration of the project data into the master project is required even if you determine during the commissioning phase that the shared central elements need to be expanded or corrected, for example, the PLC tag table needs to be expanded, or a new hardware component needs to be added.

Notes on compatibility

The following compatibility rules apply to shared commissioning:

- If an Engineering System with TIA Portal V13 or later is online on the CPU and performs a download, a second ES cannot establish an online connection, regardless of the version.
- If an Engineering System with TIA Portal < V13 is online on the CPU, a second ES cannot establish an online connection, regardless of the version.

Team Engineering

As part of Team Engineering , you have the option to perform shared commissioning of projects. Several editors can access one CPU in parallel and at the same time with multiple Engineering Systems (ES) in this process.

Shared commissioning

In shared commissioning, multiple Engineering Systems work together and in parallel on one CPU as part of team engineering.

Master project

The master project is a basic project for shared commissioning. It is a project structured according to the specified rules which already contains the fully configured hardware configuration with all required tags and blocks. This project is loaded into the jointly used CPU and then distributed as "master project" by means of project copies to the participating Engineering Systems. Each Engineering System only processes the parts it was assigned within the specific project copy. The project copies are then integrated into the master project once again.

Project copy

Project copies are created from structured master projects when working with team engineering and distributed to the participating Engineering Systems for editing. Each Engineering System only processes the parts it was assigned within the specific project copy as part of shared commissioning. The individual project copies are once again integrated into the master project at the end of shared commissioning.

See also

Requirements for shared commissioning (Page 6638)

Procedure for shared commissioning (Page 6639)

15.1.2 Requirements for shared commissioning

Software and hardware requirements

For shared commissioning of projects, the minimum software and hardware requirements for the installation of TIA Portal V13 or later must be met.

The following requirements also apply:

Software:

- You have installed TIA Portal V13 or later and the "SIMATIC STEP 7 Professional" software package on the involved Engineering Systems.
- The same software version must be installed on all Engineering Systems.
- The master project must have been created with a project of V13 or later, which has been updated to the version of the TIA Portal used in the engineering system.
- After switching from a lower to a higher TIA Portal version, all involved engineering systems must use this version.
- After switching from a lower to a higher TIA Portal version, the master project must centrally adapted and new project copies must be transferred to the engineering systems involved.

Hardware:

- You have access to a fully configured CPU S7-1500 as of firmware version V1.5.
- The involved Engineering Systems can establish an online connection to this CPU.

Requirements for shared, parallel working on the project

The following requirements are in effect for shared commissioning of projects:

- You have created a project with complete hardware configuration and fully programmed user program that can be commissioned.
- The project was downloaded to the CPU and defined as "master project".
- Multiple Engineering Systems have access to this CPU and can establish an online connection to this CPU.
- You are familiar with the defined rules and procedures for working together on one CPU.

Note on compatibility mode

The functions for shared commissioning are not available in compatibility mode.

See also

Basics for shared commissioning (Page 6635)

Procedure for shared commissioning (Page 6639)

Rules for shared commissioning (Page 6642)

15.1.3 Procedure for shared commissioning

Introduction

If you are performing shared commissioning of a project as part of team engineering, it is very important that all editors of the project observe a defined procedure.

Only when the specified procedure is observed are changes and corrections in the project automatically synchronized and applied, and changes of individual editors not unintentionally overwritten during loading or even lost in case of competing changes.

Procedure for creating the master project

The master project is created in the following steps:

1. Create a master project that includes the entire project structure.
2. Fully configure the hardware for the master project.
3. Define a language as project language, which must also be used exclusively by all participating engineering systems.
4. Create all tags and blocks that are required in the master project.

5. Create your own folders and groups for the blocks to be edited by the individual engineering systems.
6. Create a fully programmed and executable user program.
7. Download the master project to the shared CPU.
8. Save the master project following every download.

For this, also observe the Rules for shared commissioning (Page 6642).

Procedure for shared commissioning

Shared commissioning takes place with the following steps:

1. Download the master project to the shared CPU.
2. Create copies of the master project and distribute them to the editors involved in the project on the associated engineering systems.
3. Inform all editors who is to edit the specific parts of the project copies and who may download to the CPU.
4. Have the project copies edited in the individual engineering systems.
5. After editing, the individual engineering systems download the changes one after the other to the CPU.
6. All changes are automatically detected during loading by means of an online-offline comparison. The displayed synchronization dialog offer you recommendations for synchronizing the modified data when possible. You may first have to download blocks that were edited by other editors to your ES before you can download your changes to the CPU. This may be needed to update the corrections of the other editors that have already been downloaded to the CPU in your project copy.
   The following synchronization options are available:
   - Download block: There are edited blocks on the CPU that must be updated in your project copy.
   - Download block: There are new blocks on the CPU that must be downloaded to your project copy.
   - Blocks with competing changes: System-supported synchronization is not possible in this case; the conflict must be resolved manually.
7. Manually resolve any conflicts that may occur due to competing changes or changes to central objects.
8. Download your project copy to the CPU when all conflicts have been removed.
9. Save your project copy following every download.
10. Repeat editing of project copies and download to the CPU as often as necessary until shared commissioning is complete.
11. Integrate the completely edited project copies into the master project once again in order to also back up the project data that only exist offline.

You should also observe the Rules for shared commissioning (Page 6642).

Procedure for manual synchronization of competing changes

Manual synchronization of competing changes is performed in the following steps:

1. Start the comparison editor to manually resolve a conflict displayed in the synchronization dialog during download to the CPU. Select the shared CPU in the project tree and select the "Compare > Offline/Online" command in the shortcut menu. You can define certain actions depending on the status of the objects. Note, however, that you can only perform actions in one direction during synchronizing.
2. First, select the action "Upload from device" for all blocks that were changed by other editors on the CPU and which you want to apply.
3. If necessary, perform a detailed comparison of the blocks to identify differences between your offline blocks and the online blocks loaded on the CPU.
4. Manually correct the competing changes to the blocks.
5. Then download the affected blocks to the CPU using the "Continue without synchronization" command.
6. Save your project or project copy following every download.

For this, also observe the Rules for shared commissioning (Page 6642).

Note

To prevent online changes occurring again while an engineering system is being manually synchronized, no downloading to the shared CPU should be performed by another participating engineering system during the manual synchronizing.

Procedure for integrating project copies into the master project

You integrate project copies into the master project with the following steps:

1. Open the master project and the project copy to be integrated as a reference object.
2. Copy the program parts you have edited from the respective project copy to the master project and confirm overwriting the existing objects. You can also use the comparison editor to apply your program parts to the master project.
3. Save the master project and download it to the shared CPU with the "Continue without synchronization" command.
4. Save the master project following every download.

For this, also observe the Rules for shared commissioning (Page 6642).

Procedure for modifying central objects within the master project

Modification of central objects that have an effect on all program parts is performed with the following steps:

1. Stop further processing of the project copies.
2. Integrate all existing project copies one after the other into the master project as described above.
3. Download the master project to the shared CPU with the "Continue without synchronization" command.
4. Make the required modification on the shared objects such as in the hardware configuration, for example, or in the PLC tag table. The procedure is the same for modification of other central objects such as technology objects, F blocks, OBs, etc.
5. Download the master project into the CPU once again when modification is complete.
6. Save the master project following every download.
7. Distribute the updated project copies to the respective engineering systems for further processing.

For this, also observe the Rules for shared commissioning (Page 6642).

See also

Basics for shared commissioning (Page 6635)

Requirements for shared commissioning (Page 6638)

Rules for shared commissioning (Page 6642)

15.1.4 Rules for shared commissioning

Introduction

If you are performing shared commissioning of a project as part of team engineering, it is very important that all editors of the project observe predefined rules for successful cooperation.

Rules for the master project

The following rules are to be observed:

- Create a master project that is suitable for shared editing.
- Divide the user program into program parts that are independent of each other.
- Use "Groups" to visually separate the program parts from each other.
- Use a main OB and a central FC for each program part which calls the functions of the part program.

- If possible, create a separate PLC tag table for each part.
- In the master project, specify a project language which cannot be changed in the project copies.
- If you want to exchange data between the program parts, use the FC and FB interface parameters (IN, OUT, INOUT) or global data blocks.
- Use global data blocks to save the data for the individual program parts, no bit memories.
- Do not assign different names for blocks with identical addresses.
- Do not assign identical names for blocks with different addresses.

Example of the program structure in the master project

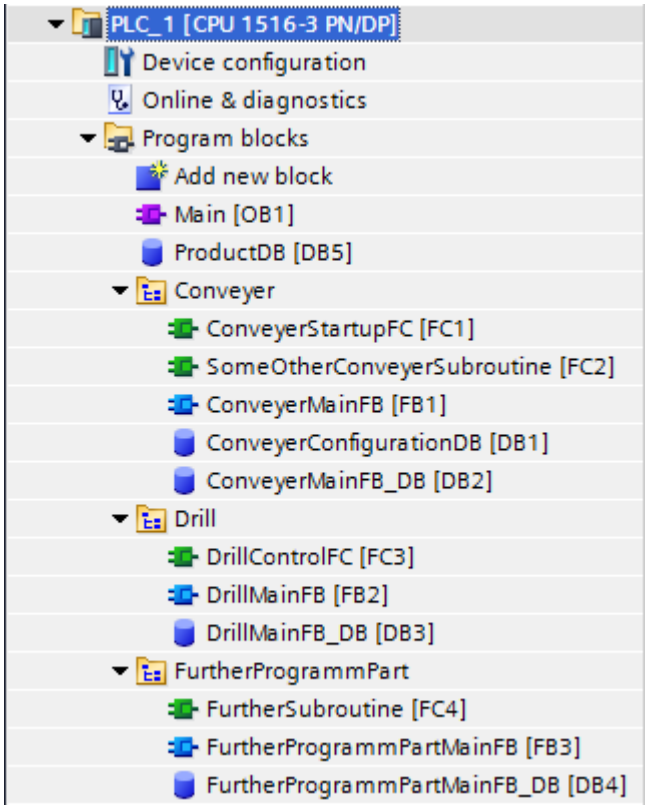
Here is an example of a team-capable, structured master project that is suitable for parallel editing during shared commissioning.

The program parts to be edited by the individual Engineering Systems are:

- Program part 1: "Conveyer"
- Program part 2: "Drill"
- Program part 3: "FurtherProgramPart"

Each program part contains a "main FB", which calls the specific lower-lever functions for this program part.

Example: In the "Conveyer" program part, the "ConveyerMainFB" calls the "ConveyerStartupFC" function.



In team engineering, this subdivision within the project structure allows the parallel editing of individual program parts and the automatic synchronization of changes during the shared commissioning phase.

Rules for shared working on one CPU

Observe the following rules for shared working:

- Editor only edits the blocks they have been assigned in the project copy in the assigned groups.
- Only OBs, FBs, DBs, FCs and UDTs may be changed in the project copies, which means the following program elements must **not** be edited in the project copies:
  - Hardware configurations
  - PLC tag table
  - Technology objects
  - Text lists and project language
  - F blocks
- Global data blocks should be used in programming instead of bit memories.
- IEC timers and counters should be used during programming instead of SIMATIC timers and counters.
- The project language specified in the master project may not be changed in the project copies.

Rules for editing shared, central objects

Observe the following rules for shared central objects:

- Changes to shared central objects always have to be made by means of the master project. These objects include:
  - Hardware configurations
  - PLC tag table
  - Technology objects
  - Text lists and project language
  - F blocks
- Before the shared central objects can be modified in the master project, the various working versions within the project copies must first be integrated into the master project once again.
- The modification can then be made within the master project, and the master project can then be downloaded to the shared CPU.
- Once modification is complete, new project copies can be created and distributed to the Engineering Systems for further processing.

Rules for working with organization blocks

Observe the following rules for editing OBs:

- OBs can only be created in the master project. If you create new OBs in a project copy, they **cannot** be synchronized.
- OBs that already exist in the master project and that have already been downloaded to the CPU, can be modified in the respective project copies. These OBs are automatically synchronized during downloading.

Rules for downloading to the CPU

Observe the following rules for downloading:

- Editors can download the project copies one after the other to the CPU; only one Engineering System can download at any given time.
- The online connection of the involved Engineering Systems can be maintained when downloading the blocks (hardware may only be downloaded in the master project).
- Perform the synchronization that is offered automatically in the synchronization dialog during download. This ensures that changes from other editors are not unintentionally overwritten.
- Correct the name conflicts that are shown in the synchronization dialog, which you rename in your offline block before downloading to the CPU. A name conflict is indicated if you want to download to the CPU a block that already exists with identical name on the CPU.

- So-called "competing changes" must be corrected manually. Competing changes come about when two editors work on the same block on two Engineering Systems at the same time. The first editor can still download his or her changes to the CPU without any problems. However, if the second editor wants to download the same block with the changes the editor has made, the synchronization dialog displays a conflict that cannot be removed automatically. The previous changes made to this block would be undone by overwriting during download. The editors have to decide in this case which change is to be applied and which one is to be discarded. Alternatively, you can manually merge the changes with the help of the detailed comparison of the blocks in the comparison editor.
  Such changes should always be avoided by implementing a good project structure and corresponding agreements between the editors.
- Changes to central objects are displayed in the synchronization dialog but cannot be synchronized automatically. Changes to central objects and to the hardware configuration always have to be made by means of the master project.
- Save your project or project copy following every download.

Note

Creating and subsequent deleting of objects in the project copies

Please note that a software synchronization is also required before downloading when you create a new object in a project copy and then delete this again immediately afterwards.

The creation of an object causes a change in the internal data management which cannot be undone by the subsequent deleting of the object. As a result it is necessary to perform a synchronization before the next loading if the newly created object was deleted immediately thereafter and no change in the existing object is visible within the project copy.

Rules for online functions

The following rules are to be observed when using online functions:

Loading:

- Only one Engineering System can download data to the CPU at any given time.
- Other Engineering Systems can use additional online functions during the download, such as monitoring and controlling.

Monitoring and controlling:

- Up to five Engineering Systems can be monitored and controlled simultaneously on the CPU blocks.
- A specific code block can only be monitored and controlled by one Engineering System at a time.
- However, additional Engineering Systems can simultaneously monitor and control other code blocks.

NOTICE
Danger due to modifying an identical operand in parallel with different modify values in more than one watch table When working with more than one watch table, avoid modifying identical operands permanently multiple times with different modify values. If an identical operand is modified permanently with different modify values at the same time in different watch tables, all watch tables will display the last modified value, because the modify value assigned last will be used in this case.

Force

- Forcing can only be implemented on the CPU from one Engineering System at a time.
- The Engineering Systems that starts the forcing has taken over the force job exclusively. Although the other Engineering Systems receive the information that a force job is running, they cannot access this job to make changes. Use the command "Update forced operands" to update all operands and values currently being forced on the CPU in the open force table. All forced operands in the open force table are updated with the relevant values. A red "F" is displayed in the first column indicating which operands are being forced.
- As soon as the Engineering System to which the force job belongs ends the online connection, the force job can be applied by another ES that establishes an online connection to the CPU and has executed the "Update forced operands" command. This enables the "Force all" and "Stop forcing" buttons and you can select these functions.

Trace functions:

- Up to four Engineering Systems can perform trace functions simultaneously.
- The Engineering Systems that starts the tracing has taken over the trace job exclusively. Although the other Engineering Systems can see this job in the trace editor, they cannot access it.
- As soon as the Engineering System to which the trace job belongs closes the trace editor, the trace job can be taken over by another ES that re-opens the trace editor.

See also

Basics for shared commissioning (Page 6635)

Requirements for shared commissioning (Page 6638)

Procedure for shared commissioning (Page 6639)

15.2 Exchanging data with Inter Project Engineering (IPE)

15.2.1 Basics of Inter Project Engineering (IPE)

Introduction to Inter Project Engineering (IPE)

The functionality of Inter Project Engineering, hereinafter referred to as IPE, is used to exchange the controller data in a source project between different projects with the help of a device proxy.

You can then transfer this data to other projects, for visualization in HMI, for example, and use it there for further configuration.

Inter Project Engineering gives you a simple way of providing controller data from a PLC programming across multiple projects for an HMI configuration.

Using the object "Device Proxy Data", you exchange controller data across multiple projects consistently and easily without redundant configuration overheads.

This makes it possible for HMI project engineers to work on a project without the hardware configuration having to be present in their project. This means they can work on the PLC project and on the HMI project at the same time.

You can exchange the following controller data between projects via the device proxy:

- Program blocks
- Technological objects
- PLC tags
- PLC alarms

The following data is automatically included in the exchange:

- Controller interfaces
- Configured communication processors and communication modules

This automatic data exchange ensures that the interfaces and the configured communication processors and modules are transferred along with the data you selected. This data is required to allow you to continue working consistently and without problems in your target project.

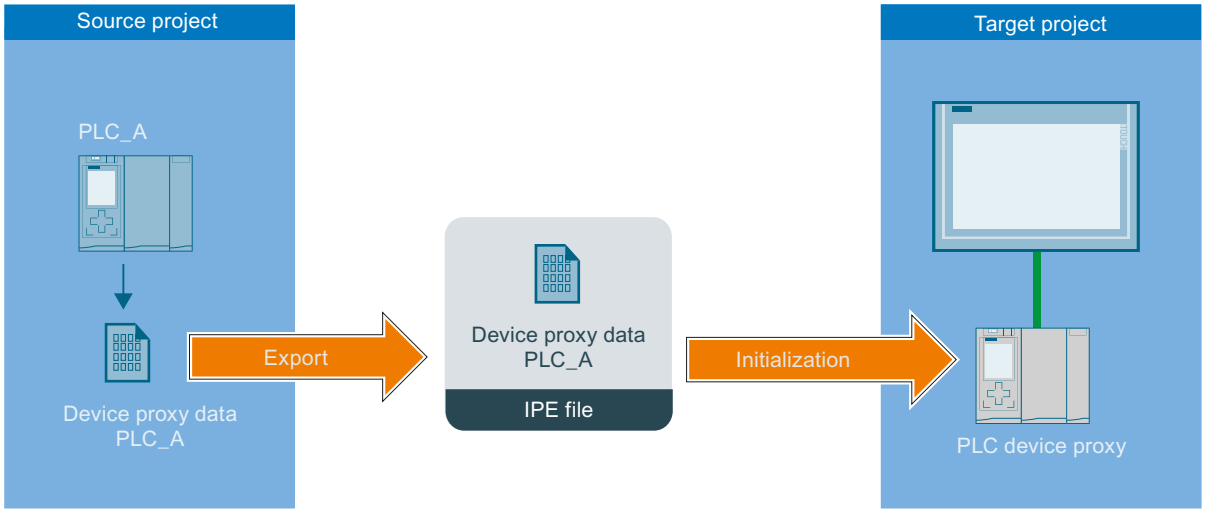
Cross-project data exchange

The following options are available for cross-project data exchange with Inter Project Engineering:

- Exchanging controller data via IPE file
- Exchanging controller data via project file

Exchanging controller data via IPE file

The figure below shows cross-project data exchange via an IPE file.

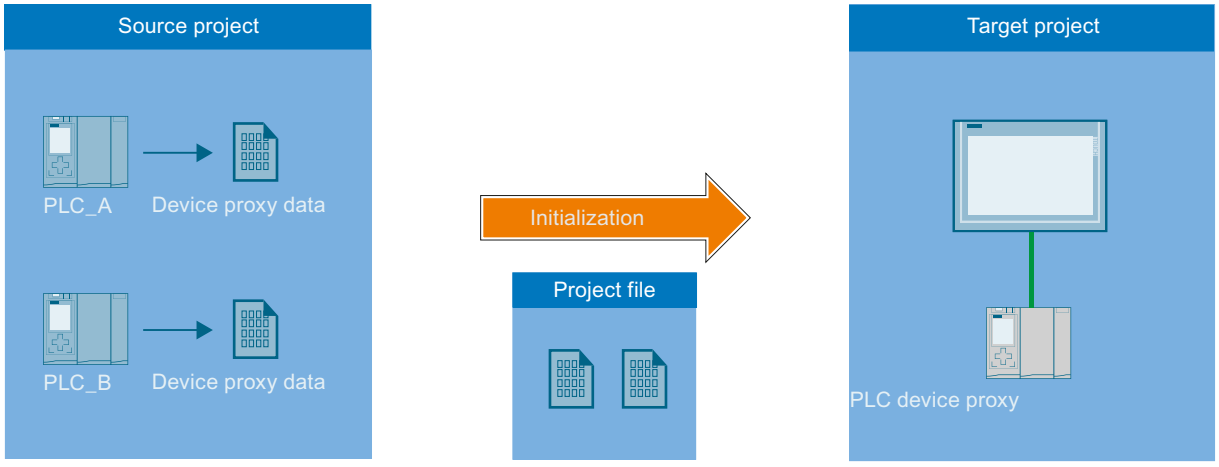


The PLC controller data from the source project is transferred to the PLC in the target project via the object "Device Proxy Data".

PLC data that is exchanged via an IPE file must originate from TIA Portal projects of V13 or higher.

Exchanging controller data via a project file

The figure below shows cross-project data exchange via a project file.



The PLC controller data from the source project is transferred to the PLC in the target project via a project file.

You can easily transfer and continue to use controller data from older TIA Portal projects as well as STEP 7 Classic projects that are not integrated in TIA Portal in your current TIA Portal projects.

Inter Project Engineering (IPE)

The functionality of Inter Project Engineering, also referred to as "IPE", is used to exchange the controller data in a source project between different projects with the help of the "Device proxy data" object.

IPE

The functionality of Inter Project Engineering, also referred to as "IPE", is used to exchange the controller data in a source project between different projects with the help of the "Device proxy data" object.

IPE file

The IPE file includes CPU controller data from a source project which is transferred with the "Device proxy data" object to the CPU in the target project.

Project file

The project file transfers CPU controller data from a source project to the CPU in the target project. You can conveniently transfer and continue to use controller data from older TIA Portal projects as well as STEP 7 Classic projects that are not integrated in TIA Portal in your current TIA Portal projects.

Device proxy data

The "Device proxy data" object supports consistent data exchange of CPU controller data across projects between a source project and a target project without redundant configuration.

Data exchange across projects with IPE

The Inter Project Engineering (IPE) functionality gives you the following options for consistent data exchange across projects:

- Exchanging PLC data via an IPE file
- Exchanging controller data via a project file

Both files transfer CPU controller data from a source project to a CPU in the target project.

See also

Requirements for Inter Project Engineering (IPE) (Page 6651)

Overview for working with Inter Project Engineering (IPE) (Page 6651)

15.2.2 Requirements for Inter Project Engineering (IPE)

Software and hardware requirements

To use the functionality of Inter Project Engineering, the minimum software and hardware requirements for the installation of TIA Portal V13 or higher must be met.

The following requirements also apply:

- You have installed TIA Portal V13 or higher and the "SIMATIC STEP 7 Professional" software package.
- You are using a CPU of the S7-1200/1500 or S7-300/400 series.
- The same software version must be installed on all involved controllers.

Requirements for IPE

The requirements for accepting controller data from projects are as follows:

- You have created a project with hardware configuration and PLC data that can be transferred to another project.
- You have made sure that the project is consistent before you transfer the controller data by means of the "Device proxy data" object.
- You are familiar with the defined procedures for Inter Project Engineering.

Notes on compatibility mode

The functions for Inter Project Engineering are not available in compatibility mode.

See also

Basics of Inter Project Engineering (IPE) (Page 6648)

Overview for working with Inter Project Engineering (IPE) (Page 6651)

15.2.3 Overview for working with Inter Project Engineering (IPE)

Exchanging controller data across projects

You can use the functionality of Inter Project Engineering to exchange existing controller data between different projects with the help of a "Device proxy" and to continue using this data in another project. The "Device proxy" provides you with a consistent and convenient way of exchanging data between projects without redundant configuration work.

Procedure for exchanging controller data via an IPE file

To exchange data via an IPE file, follow these steps:

1. Using TIA Portal, create a project with all required controller data and an executable user program.
2. Configure the hardware required for the project.
3. Compile the project to ensure consistency in your project.
4. In your project, create the object "Device Proxy Data" below the required CPU by clicking the corresponding command "Add device proxy data" in the "Device Proxy Data" folder.
   Result: The "Device Proxy Data" object has been created.
5. Select the required device proxy data in the project tree and double-click the object to open it in the editor.
6. Enter the required data for the device proxy data in the editor under "General" or apply the default setting. You can change the name and enter a comment for export of the controller data.
   In the "Define content" area of the editor select which of the existing controller data you wish to export as IPE file and click "Export device proxy data".
7. In the subsequent dialog, enter the name and storage location for the IPE file to be created and click "Save". You will be notified as soon as the export has been completed successfully.
   Result: The selected controller data has been exported as an IPE file.
8. In your target project, create a device proxy to be able to import the controller data contained in the IPE file. To do so, click the command "Add new device" and select the required device proxy in the dialog that opens.
9. Click the newly created device proxy and select the "Initialize device proxy" command from the shortcut menu.
10. Select the previously created IPE file to be used for the initialization, and confirm the selection in the following dialog with "OK". This starts the initialization.
    Result: The controller data from the source project contained in the IPE file is transferred to the device proxy in the target project.

After the successful exchange of controller data, you can continue with the configuration in your target project and, for example, connect PLC tags with HMI tags. The controller data can be updated as often as required when it is changed in the source project.

Procedure for exchanging controller data via a project file

When exchanging controller data via a project file, device proxy data can either be predefined in the source project or it can be defined and selected from the target project.

Follow these steps to exchange data via a project file:

1. Using TIA Portal V13 or higher, create a project with all required controller data and an executable user program.
2. Configure the hardware required for the project.
3. Compile the project to ensure consistency in your project.

4. Open the TIA Portal project from which you want to import the controller data into the target project.
5. In your project, create the object "Device Proxy Data" below the required CPU by clicking the corresponding command "Add device proxy data" in the "Device Proxy Data" folder.
   Result: The "Device Proxy Data" object has been created.
6. Enter the required data for the device proxy data in the editor under "General" or apply the default setting. You can change the name and enter a comment for the device proxy data.
7. In the "Define content" area of the editor select which of the existing controller data you want to provide by means of the device proxy data.
8. Click "Save" to save the changes in the project.
   Result: You have successfully saved the selected controller data in your device proxy data.
9. In your target project, create a device proxy to import the controller data contained in the project file. To do so, click the command "Add new device" and select the required device proxy in the dialog that opens.
10. Click the newly created device proxy and select the "Initialize device proxy" command from the shortcut menu.
11. Select the previously created project file with the prepared device proxy data to be used for the initialization, and confirm the selection in the following dialog with "OK". This starts the initialization.
    Result: The controller data from the source project contained in the project file is transferred to the device proxy in the target project.

After the successful exchange of controller data, you can continue with the configuration in your target project and, for example, connect PLC tags with HMI tags. The controller data can be updated as often as required when it is changed in the source project.

Procedure for updating already transferred controller data

System requirements for the update:

The following requirements must be met for updating already transferred controller data with the help of the already created "Device Proxy Data" object:

- You are using the same project for the update as for the previous transfer of the controller data.
- You are using the already created "Device Proxy Data" for transfer of the controller data.
- You have not made any changes to the hardware configuration or the communication interfaces in the meantime. You can add new hardware components.

Updating procedure:

1. Open the TIA Portal project from which you want to update the controller data in the target project.
2. Double-click the existing "Device Proxy Data" object and select under content which controller data you want to make available for the update.
3. Save the project or export the IPE file once again.
4. Open the required target project in the TIA Portal.

5. Click the existing device proxy in the target project and select the "Update device proxy" command from the shortcut menu.
6. Select the source which is to be used for the update, and confirm the selection.
7. In the following dialog, select under "Device" the CPU that was previously used as source, from which the controller data is to be re-imported.
8. Under "Defined device proxy data", select the required object and exit the dialog with "OK".
9. The update is then started and the selected device proxy data is transferred from the source project to the device proxy in the target project.

After the successful updating of the controller data, you can continue with the configuration in your target object and, for example, connect PLC tags with HMI tags. The controller data can be updated as often as required when it is changed in the source project.

See also

Basics of Inter Project Engineering (IPE) (Page 6648)

Requirements for Inter Project Engineering (IPE) (Page 6651)

15.2.4 Creating "Device proxy data" in the source project

Introduction

In order to exchange controller data across projects you require the "Device proxy data" object that can transfer the controller data of the associated CPU as an IPE file in the source project.

Creating a "Device proxy data" object in the source project

Proceed as follows to create a new "Device proxy data" object:

1. Open the "Device proxy data" folder below the CPU to which you want to create a "Device proxy data".
2. Click the command "Add new device proxy data".

Result

The "Device proxy data" object has been created and is displayed in the project navigation below the associated CPU.

The created "Device proxy data" contains the controller data of the associated CPU.

15.2.5 Creating an IPE file by means of the "Device proxy data"

Opening the "Device proxy data" and defining the content for an IPE file

Proceed as follows to create an IPE file by means of a "Device proxy data":

1. Open the required "Device Proxy Data" with a double-click in the "Device Proxy Data" folder in the project tree below the CPU.
2. Enter the required settings for the "Device proxy data" object in the editor under "General" or apply the default setting. You can change the name and enter a comment for export of the controller data.
3. In the "Define content" area of the editor select which of the existing controller data of the CU you wish to export as an IPE file and click "Export device proxy data".
4. In the subsequent dialog, enter the name and storage location for the IPE file to be created and click "Save". You will be notified as soon as the export has been completed successfully.

Result

The selected controller data have been exported as an IPE file.

15.2.6 Using controller data from other projects with IPE

15.2.6.1 Using controller data from other projects in an HMI device

Basics on controller data

Introduction

In the source project of the PLC, you can select which controller data is to be stored as device-proxy data.

To this end, you create a device proxy data from a configured PLC.

Which controller data can be exchanged?

The following controller data can be exchanged via IPE:

- Data blocks
- Technology objects
- PLC tags
- PLC supervisions and PLC alarms

The following information is exchanged automatically:

- Interfaces of the PLC
- Configured communication processors and communication modules at the PLC

Initializing a device proxy via an IPE file

Initializing device proxy via an IPE file

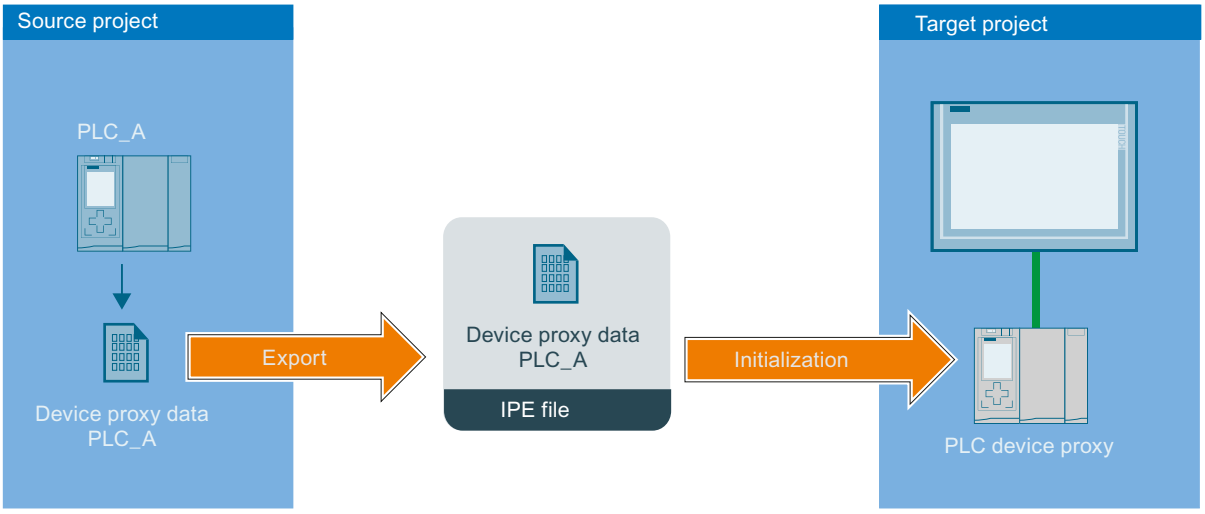
How does initialization of a device proxy via an IPE file work?

You can exchange controller data between two or more projects using IPE files.

Due to their small data volume, you can send IPE files by e-mail, for example.

You can create multiple IPE files with different device proxy data for each PLC.

A device proxy can only be filled with one device proxy data record in the target project.



- An IPE file is exported from the source project.
  The IPE file contains device proxy data of the PLC "PLC_A".
- A device proxy is created in the target project.
- In the target project, the device proxy is then initialized with the device proxy data from the IPE file.
- Following initialization, all device proxy data from the IPE file is contained in the device proxy.
- If changes are made in the source project, the device proxy of PLC_A in the target project can be updated via the new IPE file.

Initializing a device proxy via an IPE file

Introduction

You initialize device proxy data using a dialog on the device proxy in the TIA Portal.

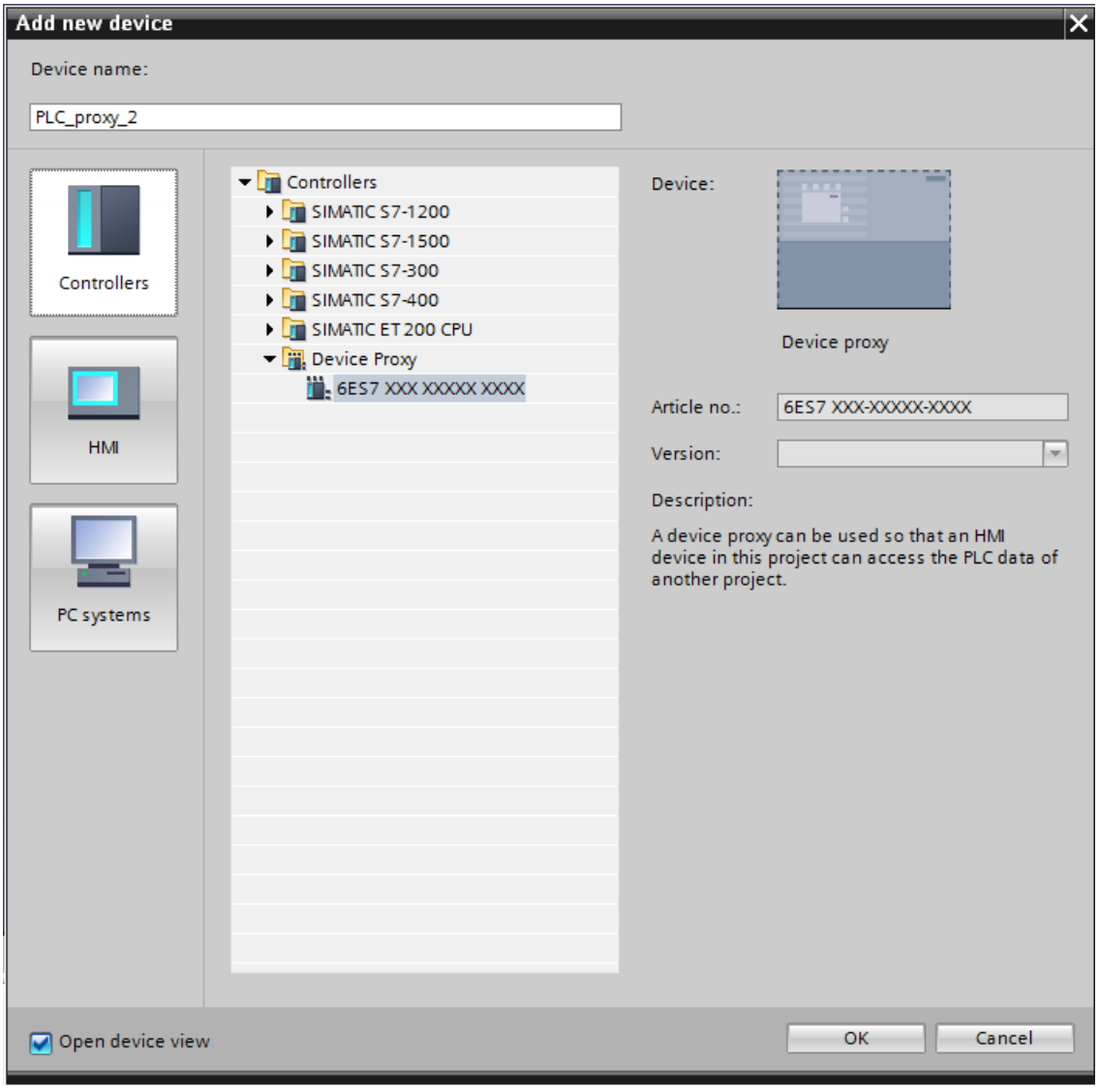
The IPE file contains the device proxy data from the source project.

Requirement

IPE file is available.

Procedure

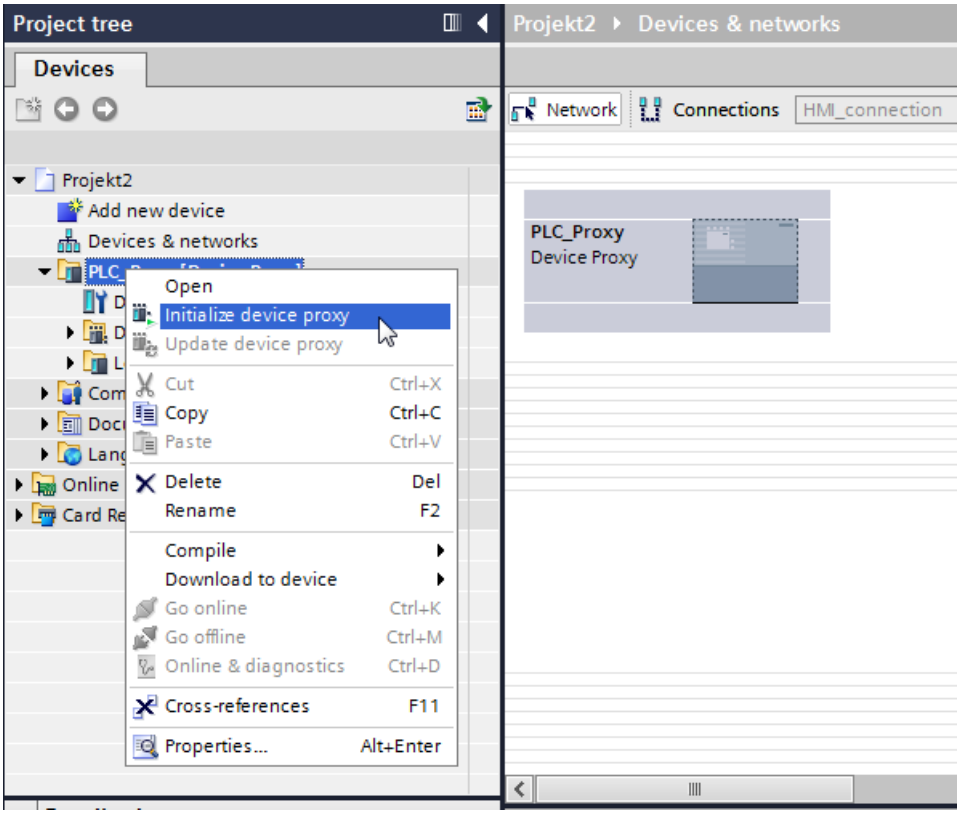
1. Double-click "Add new device" in the project tree.
2. Select the device proxy under "Controller".



A new device is created in the "Devices & Networks" editor.

3. Select the device proxy in the project tree.

4. Click "Initialize device proxy" in the shortcut menu.



5. Select the IPE file and click "Open".
   The "Initialize device proxy" dialog opens.

Note

You cannot select the device proxy data before initialization in the target project.

All device proxy data included in the IPE file is transferred.

Initialize the device proxy via project file if you want to select specific device proxy data before initialization: Initializing a device proxy via a project file (Page 6662)

6. Click "OK".
   Initialization of the device proxy is started.

Result

Following initialization, all device proxy data from the IPE file is contained in the device proxy.

You can now configure an HMI connection with the device proxy and connect PLC tags from the device proxy with HMI tags, for example.

Updating a device proxy via an IPE file

Introduction

If changes were made to the device proxy data in the device proxy source project, the device proxy data can be updated in the target project.

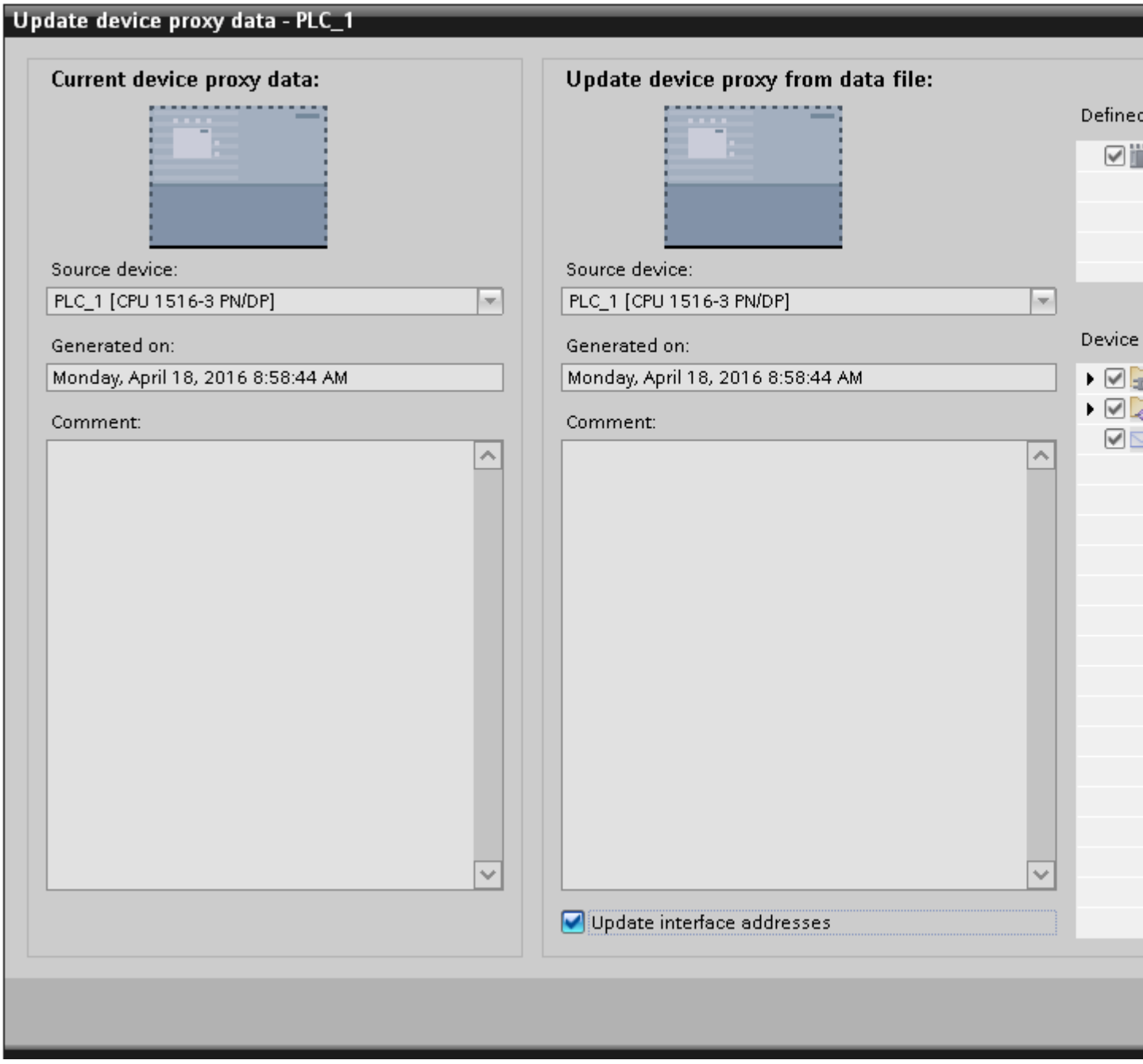
Requirement

- New IPE file contains the device proxy data from the source project.
- A device proxy has already been initialized using an IPE file in the target project.

Procedure

1. Select the device proxy in the project tree.
2. Click "Update device proxy" in the shortcut menu.

3. Select the IPE file.



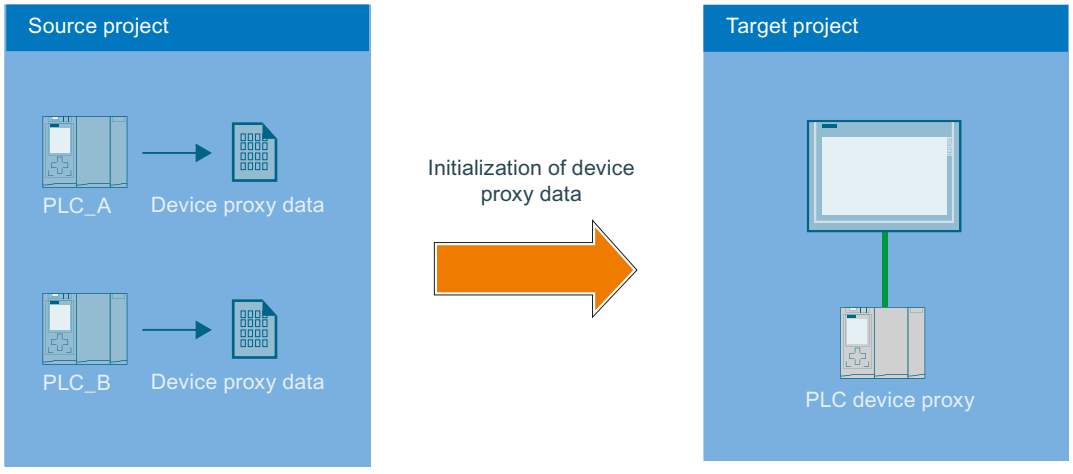
4. Select whether you wish to update the PROFINET or PROFIBUS addresses.
5. Click "OK".

Initializing a device proxy via a project file

Initializing a device proxy via a project file

How does an initialization via project file work?

You can transfer controller data using a project file in your TIA Portal project:



- There are two PLCs in the source project.
- Device proxy data objects of the two PLCs are created in the source project.
- The source project is saved.
- A device proxy is created in the target project.
- The source project (*.ap13) is first selected using the command for the initialization of device proxy in the device shortcut menu.
- The "PLC_A" and "PLC_B" controllers are available for selection and their device proxy data have already been created.
- After selection of device proxy data of "PLC_A", the data is stored in the PLC_Proxy of the target project.
- Changes from the source project can be transferred by an update via the project file.

Initializing a device proxy via a project file

Introduction

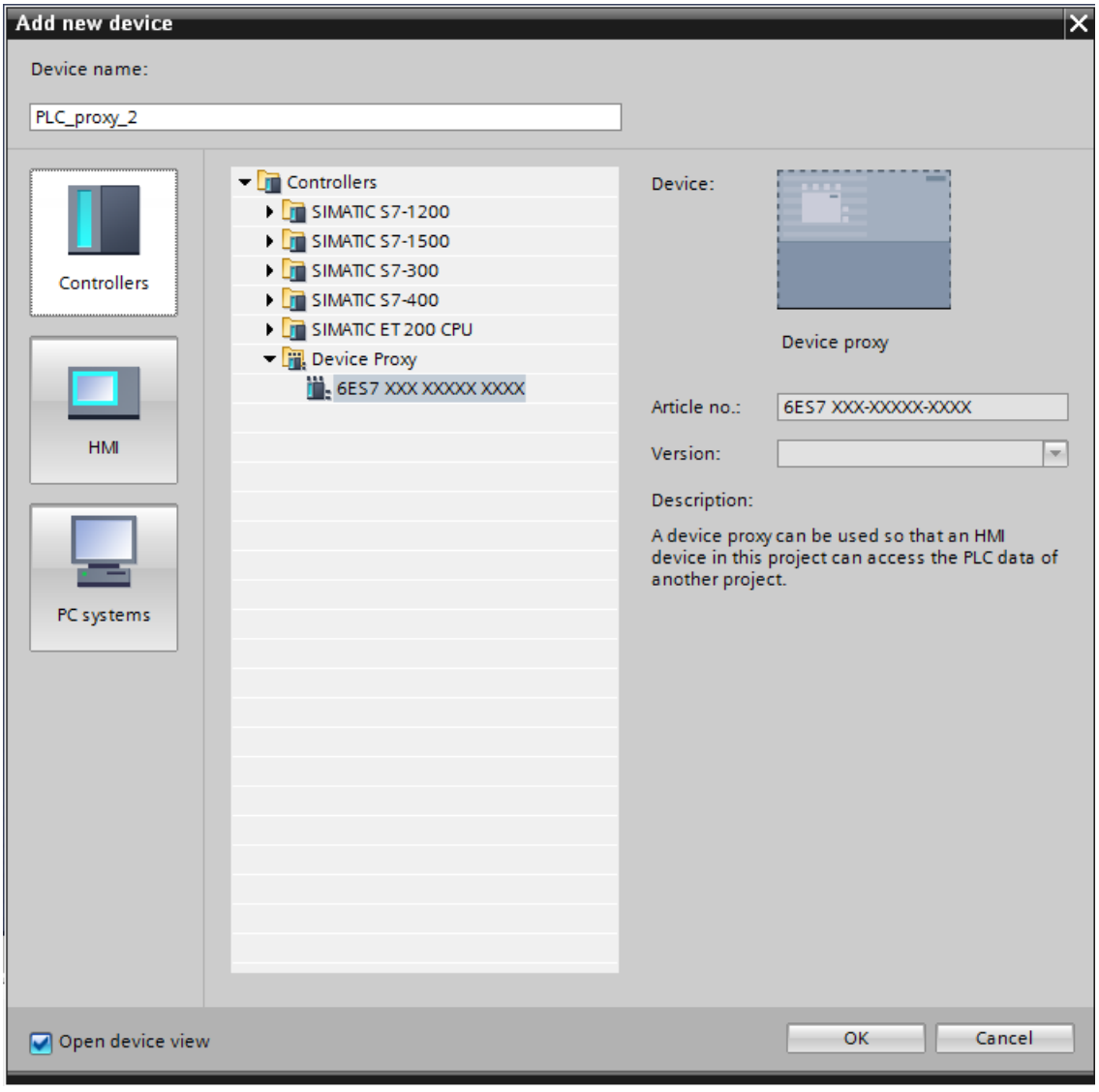
You initialize the device proxy using a project file.

Requirement

Project file (*.ap*) is available.

Procedure

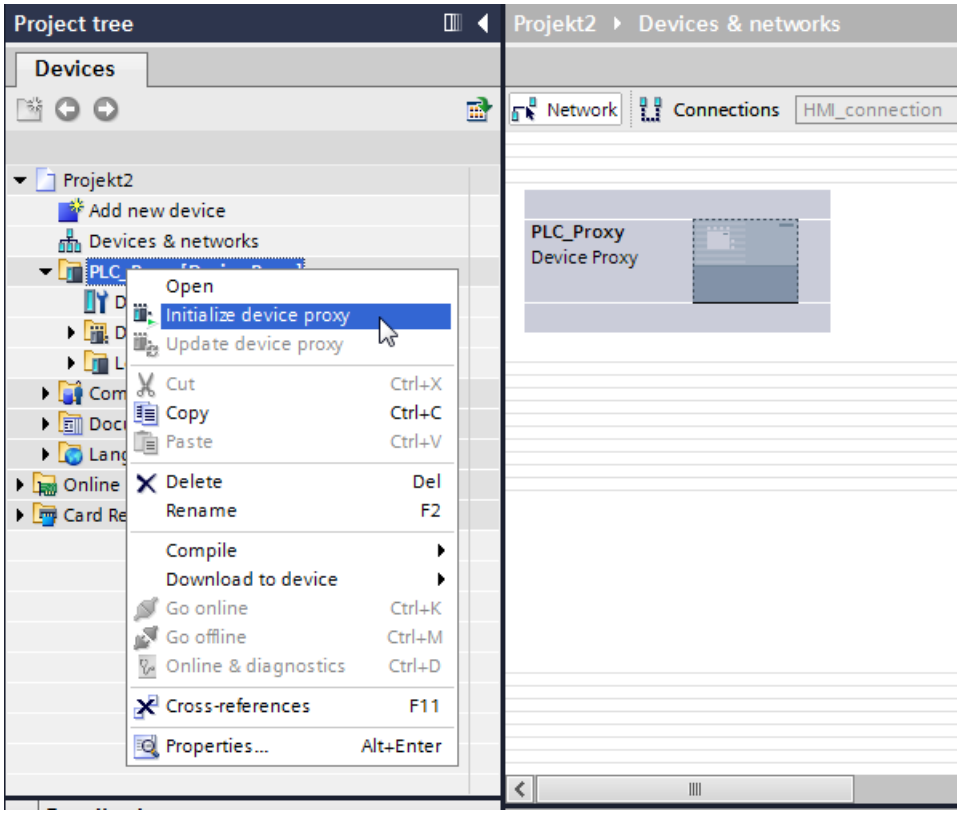
1. Double-click "Add new device" in the project tree.
2. Select the device proxy under "Controller".



A new device is created in the "Devices & Networks" editor.

3. Select the device proxy in the project tree.

4. Click "Initialize device proxy" in the shortcut menu.



5. Select the following entry in the "Open device proxy data source" dialog:
   "TIA Portal projects (*.ap*)"
6. Select a project file and click "Open".
   The "Initialize device proxy" dialog opens.
7. Select a device proxy data object already created from an available PLC to initialize the device proxy.
8. Click "OK".

Note

During initialization of the device proxy a check is automatically carried out whether the selected data are consistent and can be imported into the target project. If the source project contains inconsistent data, compile your source project.

Result

Following initialization, the controller data from the project file in the selected device proxy data object is stored in the device proxy.

You can now configure an HMI connection with the device proxy and connect PLC tags from the device proxy with HMI tags, for example.

Initializing via STEP 7 V5.5 projects

You can find more information about importing control data from projects prior to TIA Portal V11 in the section Integrated configuration with WinCC and Simatic Manager (Page 6674).

You can find additional information in the FAQ with entry ID: 73502293 (http://support.automation.siemens.com/WW/view/en/73502293)

See also

Integrated configuring with WinCC and SIMATIC Manager (Page 6674)

Updating a device proxy via a project file

Introduction

If changes have been made to the controller data in the source project of the device proxy, you can update the device proxy in your TIA Portal project.

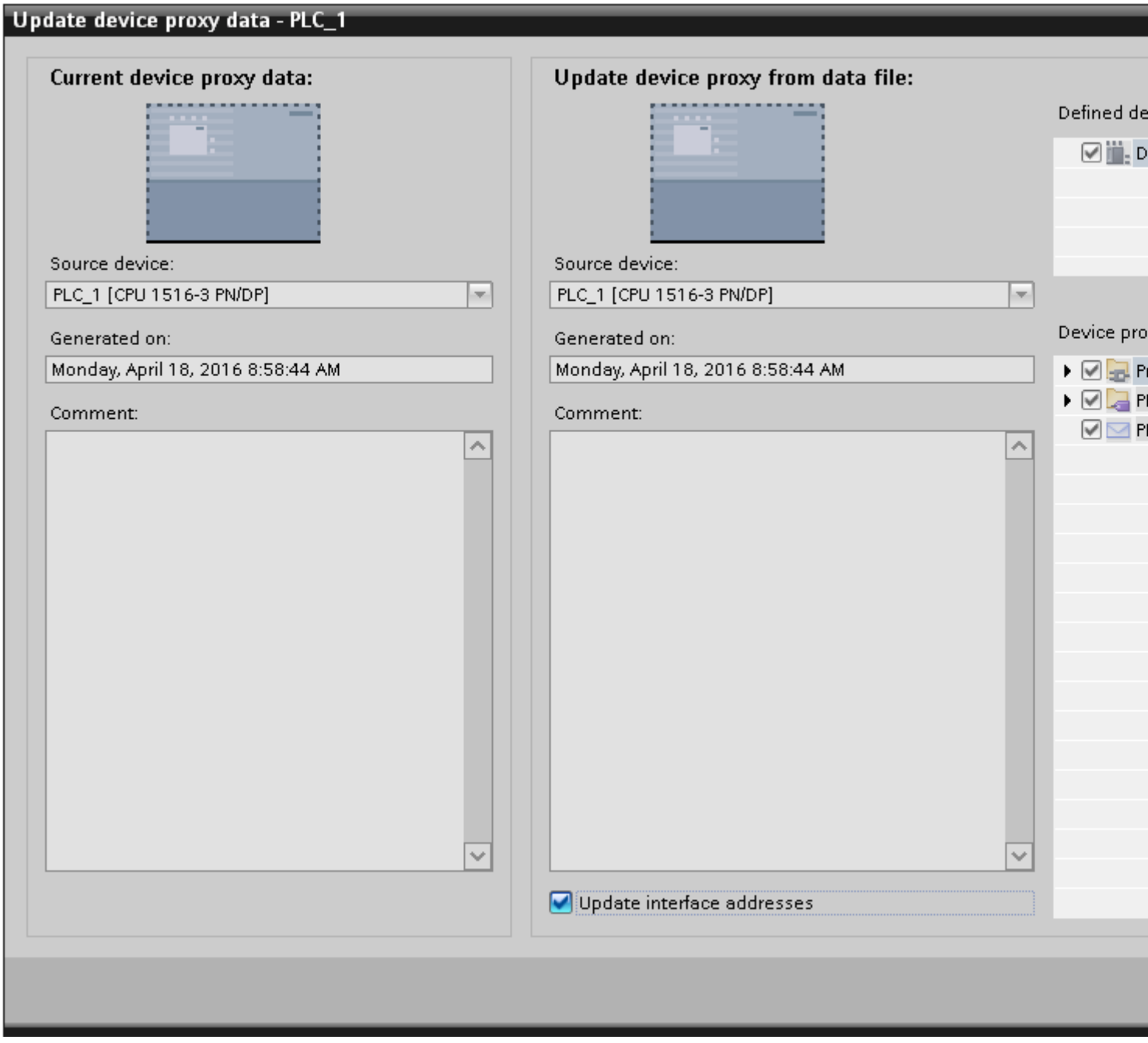
Requirement

- Project file has been generated from the source project of the device proxy.
- A device proxy that has already been initialized is available in the target project.

Procedure

1. Select the device proxy in the project tree.
2. Click "Update device proxy" in the shortcut menu.

3. Select the project file.
   The "Update device proxy" dialog opens.



4. Select a device.

   Note

   If you have created several device proxy data objects in the source project, you need to select one device proxy data object.

   Note

   Already initialized device proxies cannot be overwritten with controller data from a different PLC.

5. Select whether you wish to update the PROFINET or PROFIBUS addresses.
6. Click "OK".

15.2.6.2 Communication with device proxies

Basics of communication with device proxies

Introduction

Data exchange between the HMI device in your target project and existing controller from the source project is enabled in the TIA Portal using device proxies. You initialize a device proxy with device proxy data from the source project and link the HMI device to the device proxy PLC in the target project.

The HMI device and the device proxy PLCs can communicate with each other directly via a subnet or across multiple subnets. For communication across multiple subnets, you need to configure a device proxy-PLC as a router that connects the HMI device and the target PLC.

The communication between connection partners is possible via the following connections:

- PROFINET
- PROFIBUS
- MPI

Setting network parameters

In the TIA Portal, you connect the HMI device and the device proxy PLC in the "Devices & Networks" editor.

Once you have connected an HMI device with the device proxy PLC, adapt the properties of the network parameters. Note the

network parameters from the source project and enter them in the properties of the network connection between the device proxy PLC and the HMI device.

Note

IPE does not transfer data to the network configuration. You therefore configure the required networks in the target project manually so that the network configuration in the target project corresponds to the configuration in the source project. The subnet IDs of the devices in the target project in particular must correspond to each other and to the IDs of the devices in the source project. Otherwise, communication will not be possible between the source device and the HMI devices in runtime

Depending on the connection, provide the following parameters:

Connection	Parameters
Ethernet	S7 subnet ID
MPI PROFIBUS	S7 subnet ID Highest address Transmission rate Bus parameters

Communication via S7 routing

To establish a routed connection to these connection partners, a router must be interposed. Configure both the router and the target PLC as device proxies.

The router and the target PLC can be initialized with an IPE file, TIA Portal project file or a STEP 7 V5.5 project file.

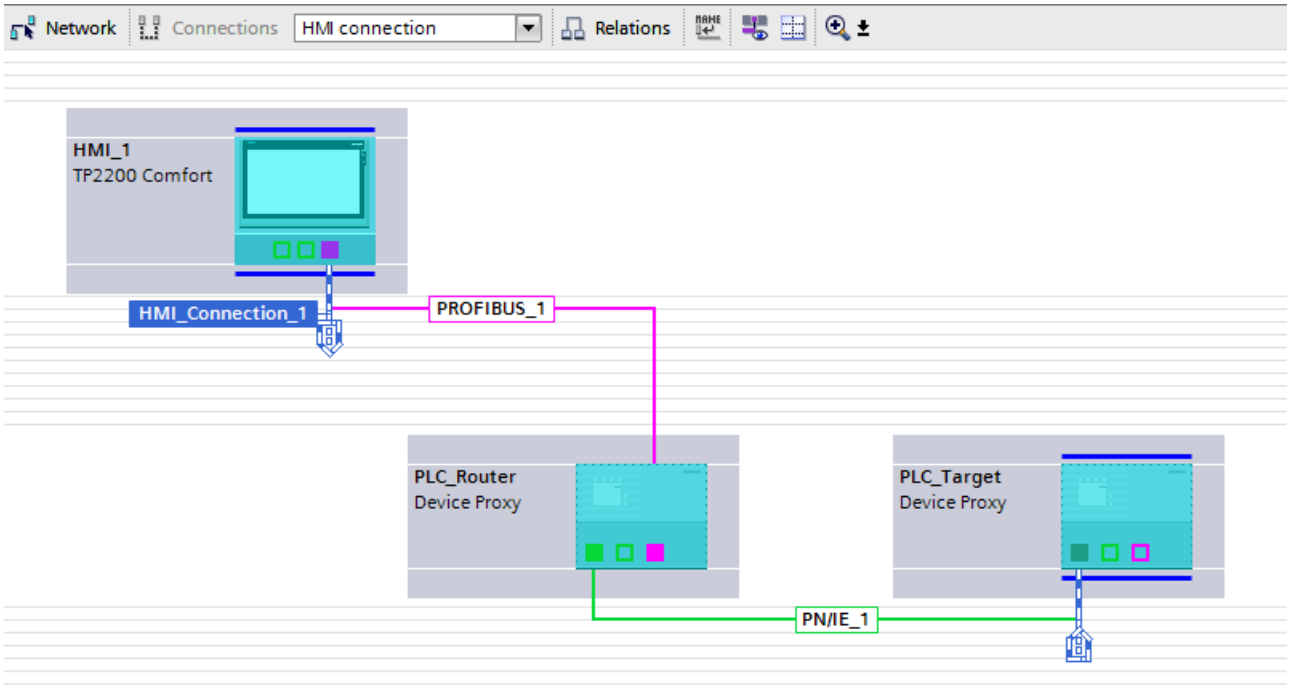
Modules with communication capability (CPUs or CPs) used to establish gateways between the subnetworks must have routing capability.

The S7 routing settings can be edited in the relevant interface properties.

You can configure the S7 routing connection in the "Devices & Networks" editor.

Note

A routed connection must be configured in the source project for a routed connection to a connection partner that was loaded with the source project data.



The routing path is determined in Runtime by the system and cannot be influenced by the user. During configuration, it is not possible to generate information about a faulty connection. All relevant routing paths have to be mapped in the TIA Portal exactly as they were configured in STEP 7 source project.

Configuring a direct connection

Introduction

In the target project, configure a direct connection between a device proxy PLC and an HMI device in "Devices & Networks" editor.
The following configuration describes a network for the following communication partners:

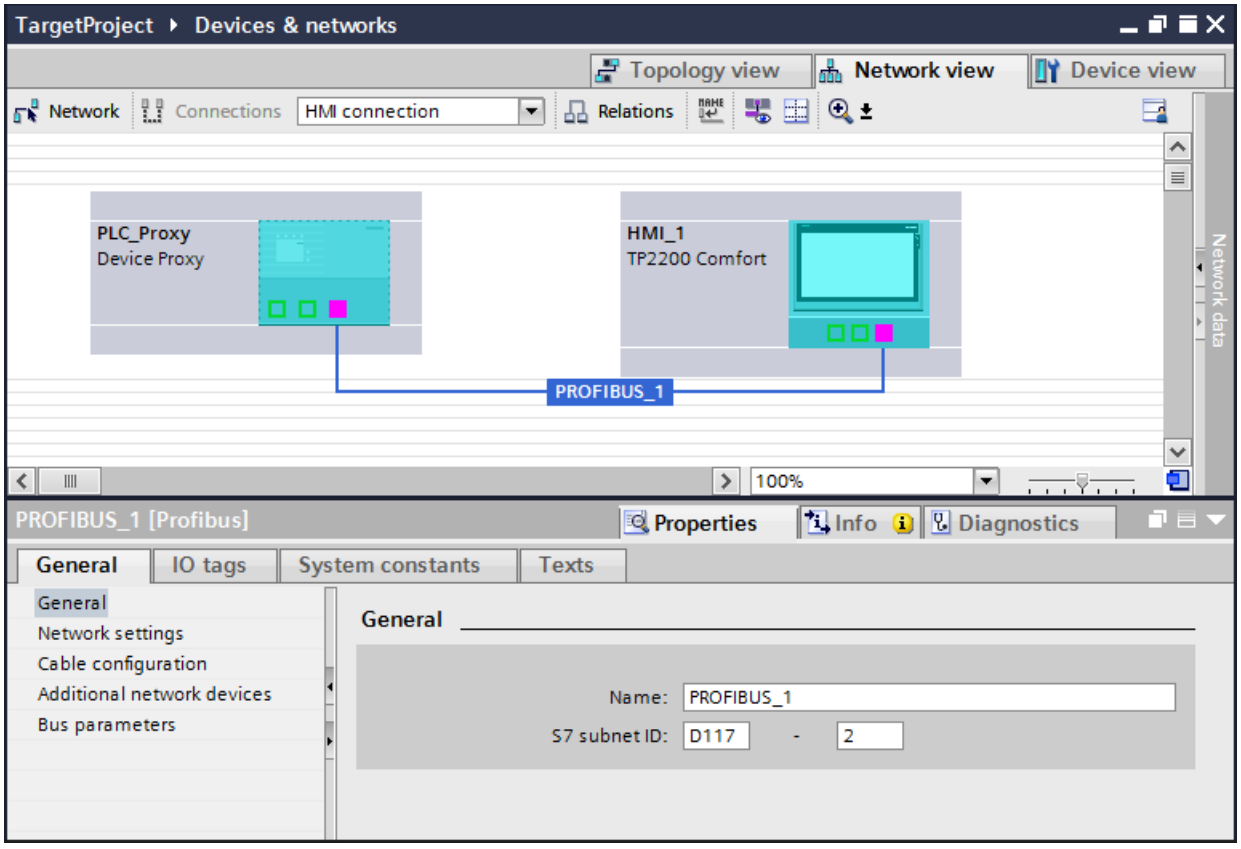
- HMI device
- Device proxy PLC

Requirements

- Device proxy data were exported from the source project
- An HMI device has been created in the target project
- A proxy device has been created

Procedure

1. Note the subnet ID of the connection in the source project.
   For an MPI or PROFIBUS connection, also note also the highest address, the transmission speed and the bus parameters.
2. Initialize the device proxy in the target project with the device proxy data from your source project.
3. Connect the communication interfaces of the HMI device and of the device proxy PLC.
   The corresponding S7 subnet is automatically generated.



4. Transfer the parameters that you noted from the source project into the parameters of the TIA project:
   - Enter the S7 subnet ID under "Properties > General".
   - Enter the highest address and the transmission rate under "Properties > Network Settings".
   - Enter the bus parameters under "Properties > Bus parameters".

Configuring a routed connection

Introduction

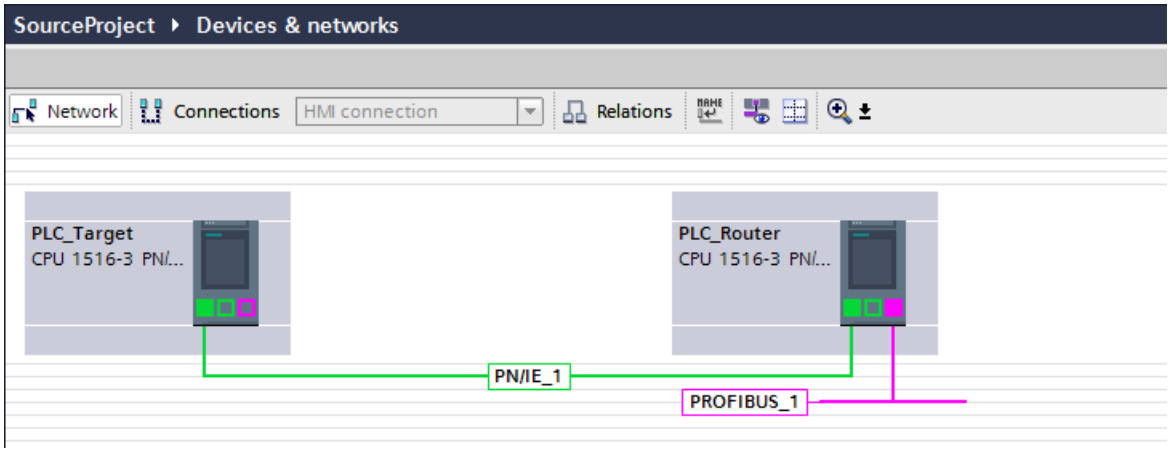
The HMI device and the device proxy PLCs can communicate with each other across multiple subnets. For a routed connection, configure a device proxy PLC as a router between the HMI device and the target PLC.

Note

For a routed connection to the connection partner from the source project, a routed connection must be configured in the source project.

The following configuration describes a network for the following communication partners:

- HMI device
- Device proxy PLC as router
- Device proxy PLC as target PLC
  All subnets that are involved in the routing between the HMI device and the target PLC must be configured in the source project. They must be identically configured in the target project.



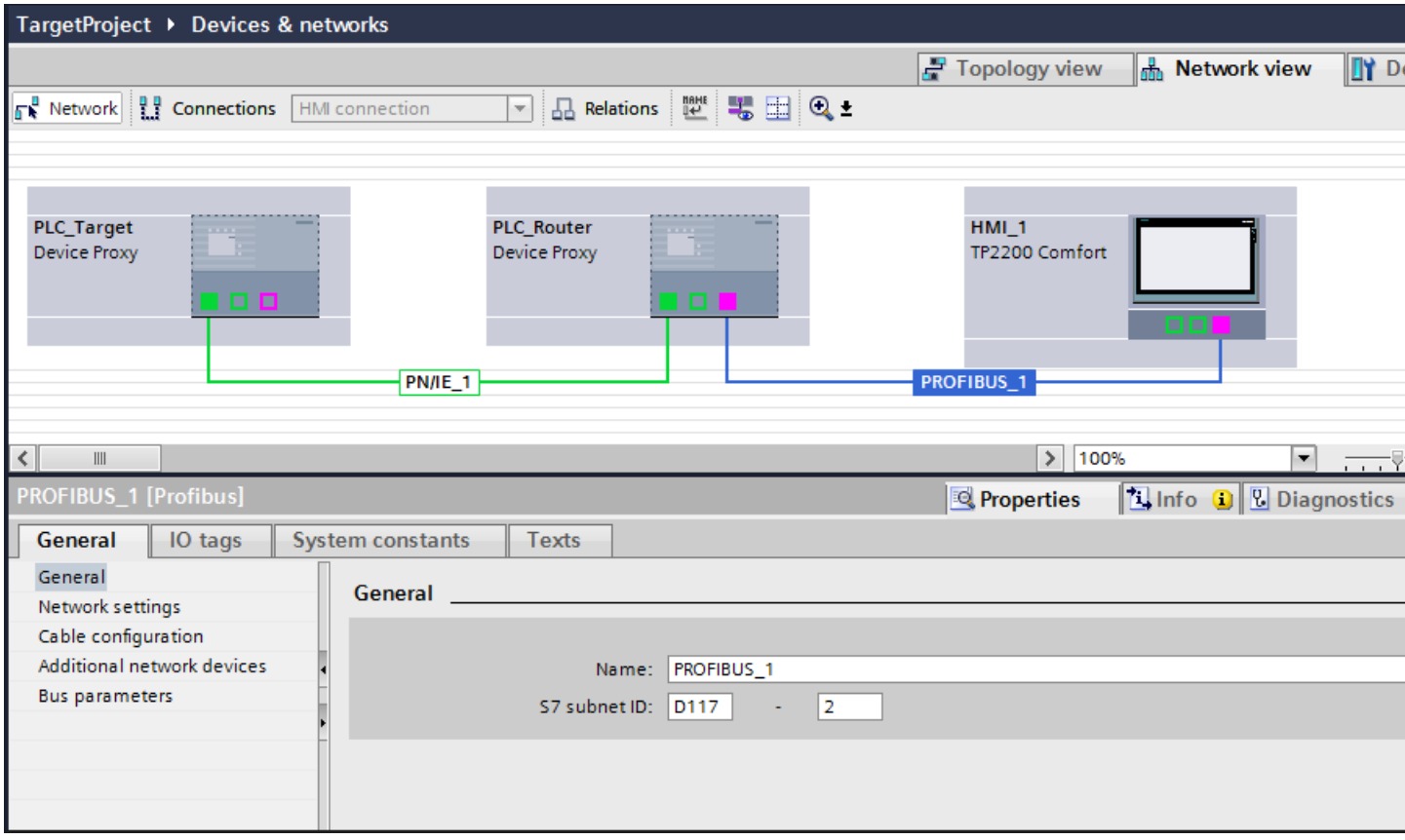
Requirements

- Device proxy data were exported from the source project
- An HMI device has been created
- Two device proxies have been added

Procedure

1. Note the subnet IDs of the connections in the source project.
   For a PROFIBUS connection, also note also the highest address, the transmission speed and the bus parameters.
2. Initialize the source proxy in the target project with the device proxy data from your source project.
3. Create the network connection between the HMI device and the router as well as between the router and the target PLC as it was configured in the source project.
4. Click the "Connections" button.
5. Connect the communication interfaces of the HMI device and of the router.
   Router and target PLC are connected automatically.
6. Create an HMI connection between the HMI device and the target PLC using drag-and-drop operation.
   The "Connect with subnet" dialog opens.

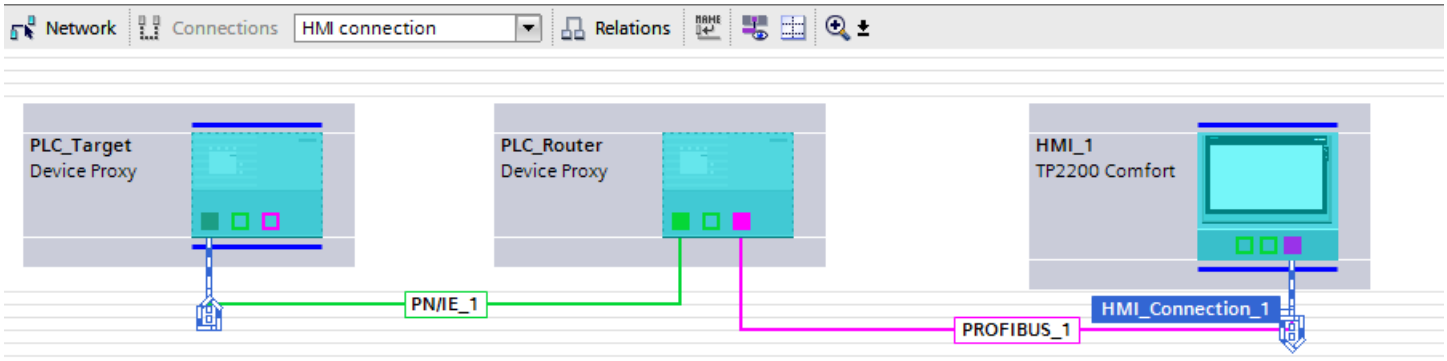
7. Select "Add S7 routed connection".
   The routing connection is created.



8. Transfer the parameters that you noted from the source project into the parameters of the TIA project:
   - Enter the S7 subnet ID under "Properties > General".
   - Enter the highest address and the transmission rate under "Properties > Network Settings".
   - Enter the bus parameters under "Properties > Bus parameters".

Result

The created connection is displayed in the "HMI connections" editor.



15.2.6.3 Integrated configuring with WinCC and SIMATIC Manager

Basic principles for configuring with WinCC and SIMATIC Manager

Integrated configuring with WinCC and SIMATIC Manager

The TIA Portal provides the option to create PLC programs and the HMI configurations using a uniform user interface.

In some cases it is still necessary, for technical or customer-specific reasons, to create the PLC program with the "STEP 7 V5.x" software and the HMI configuration with the WinCC software (TIA Portal).

You can directly access the PLCs configured in the SIMATIC Manager and their current data via the device proxy PLC and integrate this into WinCC (TIA Portal).

Requirement

- WinCC with version V13 or higher
- SIMATIC Manager STEP 7 with version V5.4 SP3 or higher
- In SIMATIC Manager all required software option packages that are used in the STEP 7 project must be installed.
- The project must be consistent in the SIMATIC Manager.

Supported PLCs

- S7-300 PLC
- S7-300F PLC
- S7-400 PLC
- S7-400F PLC

- S7-300 T-PLC
- ET200 PLC (IM 151-x CPU)

Supported alarm procedures

- Discrete alarm types
- Analog alarm types
- Signal system fault (SFM)
- Alarm_S, Alarm_SQ, Alarm_D, Alarm_DQ

Initializing a device proxy via a STEP 7 project file

Introduction

A connection to a CPU within the SIMATIC Manager is established through the initialization. After successful initialization the tags and messages contained in the SIMATIC Manager are also available in the WinCC project.

Initializing a device proxy PLC

1. Initialize the device proxy via the project file (*.s7p) of your STEP 7 project.
   In the "Initialize device proxy" dialog, select the CPU, program blocks, symbols and PLC messages for the device proxy.
2. After successful initialization the selected data are displayed in the program folders in the project tree.

Note

The data are not processed in WinCC (TIA Portal). Only reading access to the tags is available. Changes to the STEP 7 program are always carried out in the SIMATIC Manager.

Note

Support of H-PLCs

IPE functionality in the TIA Portal also supports the use of H-PLCs from STEP 7 as of V5.5. Only the master H-PLC is relevant during the initialization of the device proxy.

Display of the original name of the source device

1. Open the device configuration of the device proxy PLC.
2. Select "Device view"".
3. Open "Properties > General > General > Device proxy information".
   The original name is displayed under "Proxy source".

Creating a network connection between HMI and PLC

As when using a "Standard PLC", you configure a network connection between the used HMI device and the device proxy PLC.

1. In the device configuration select "Network view".
2. Mark the device proxy PLC and enter the same connection properties under "Properties" that the PLC program has in the SIMATIC Manager.
3. Network and connect the two devices.

Note

Network connection and addresses

Irrespective of whether you have created an Ethernet connection or a PROFIBUS connection, check the respective addresses and bus parameters of the modules used and if necessary adapt these. If you use PROFIBUS, observe the additional information under Adapting network parameters (Page 6676).

Adapting network parameters

Introduction

If you have networked an HMI device with the device proxy PLC, you have to adapt the properties of the parameters in the STEP 7 5.x project in the following cases:

- You have a STEP 7 V5.x project. The HMI devices in this project created with WinCC flexible were migrated to WinCC (TIA Portal) and networked with the device proxy PLC.
- You have a STEP 7 V5.x project. The HMI devices were configured only with WinCC (TIA Portal) and are networked with the device proxy PLC.

Depending on the connection, provide the following parameters:

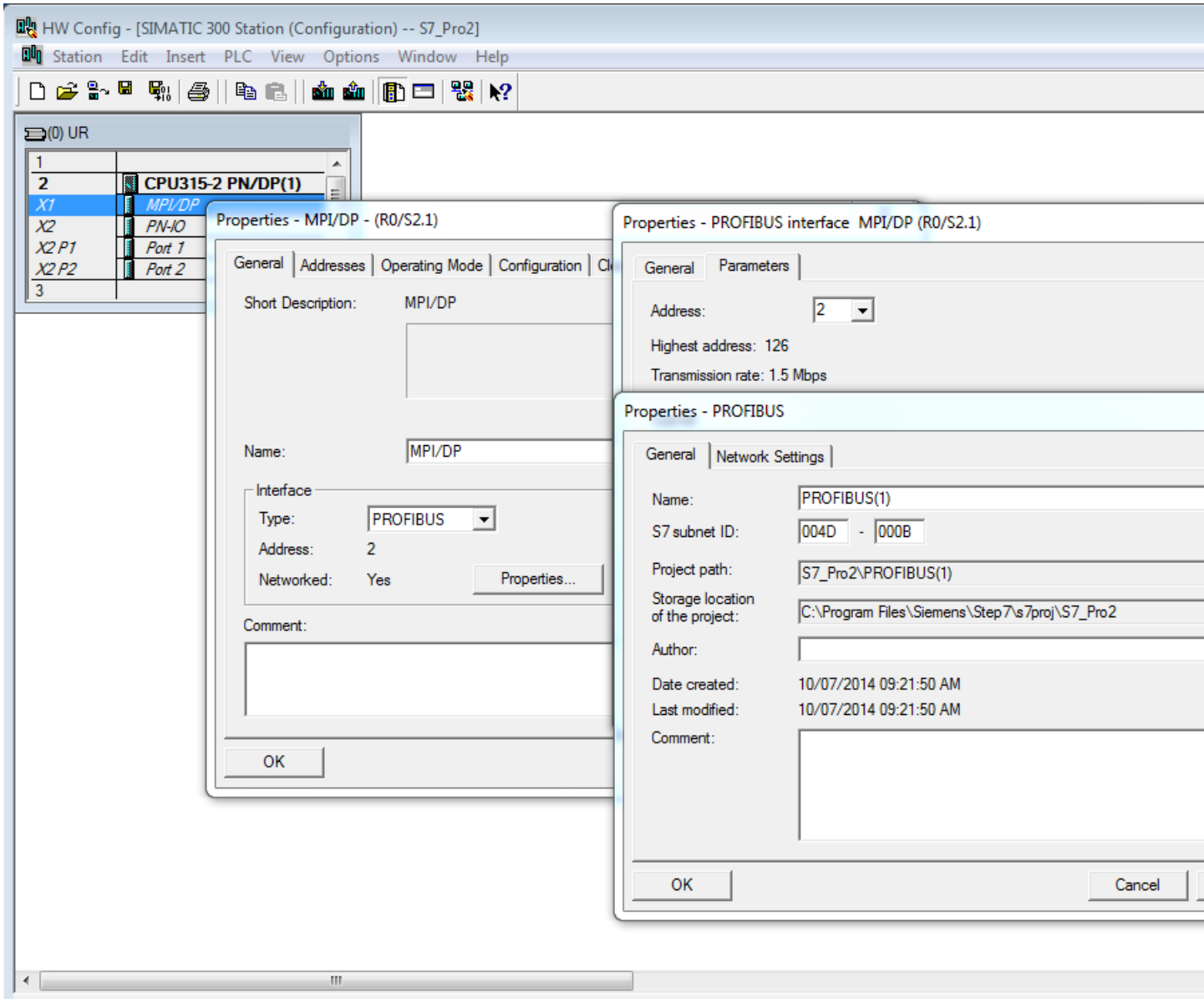
Connection	Parameters
Ethernet	S7 subnet ID
MPI PROFIBUS	S7 subnet ID Highest address Transmission rate Bus parameters

Record the parameters in the STEP 7 project and enter these in the TIA Portal at the properties of the network connection.

Adapting network parameters

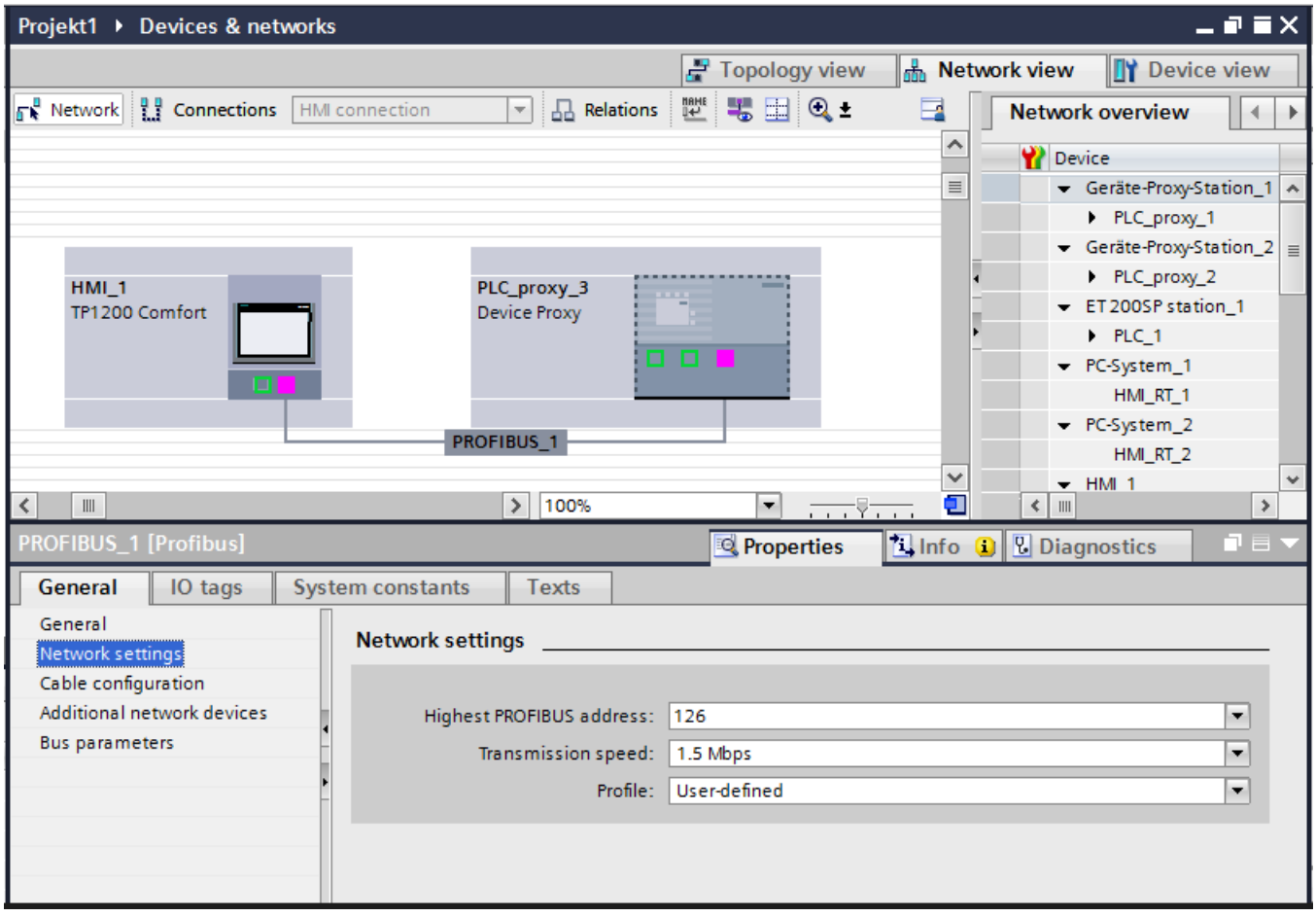
1. Open the hardware configuration of the CPU in the SIMATIC Manager.
2. Open the object properties of the PROFIBUS interface of the CPU.
3. Select the "Properties..." button under "General" in the "MPI/DP properties" dialog. The "Properties – PROFIBUS interface" dialog is opens.

4. Select the "Properties..." button under "Parameters" in the "Properties - PROFIBUS interface" dialog.
   The "Properties - PROFIBUS" dialog is opens.
5. Note the S7 subnet ID.

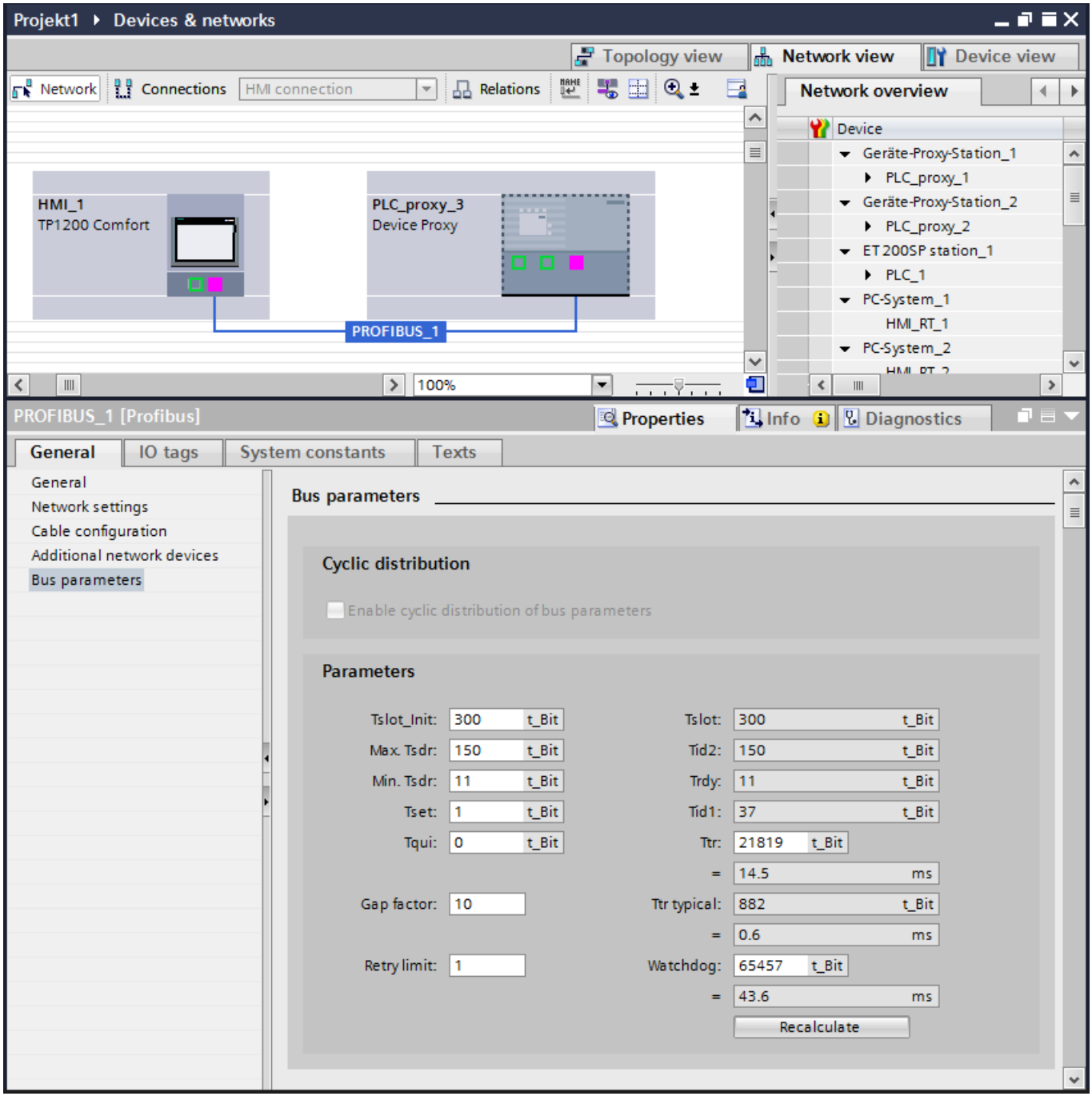


6. If you adapt the PROFIBUS connection, record the "Highest address" and "Transmission rate" parameters in the "Network settings" dialog box.
7. If you adapt the PROFIBUS connection, select the "Bus Parameters" button in the "Properties - PROFIBUS" window under "Network settings".
   Note the bus parameters listed there.
8. In "Devices & Networks" editor of the TIA Portal, select the network connection between the device proxy PLC and the HMI device.

9. Select the "User-defined" profile under "Properties > General> Network Settings".



10. Transfer the parameters that you noted from the STEP 7 configuration into the parameters of the TIA project:
    - Enter the S7 subnet ID under "Properties > General".
    - Enter the highest address and the transmission rate under "Properties > Network Settings".
    - Enter the bus parameters under "Properties > Bus parameters".


Projekt1 ▸ Devices & networks
Topology view
Network view
Device view
Network
Connections
HMI connection
Relations
Network overview
Device
Geräte-Proxy-Station_1
PLC_proxy_1
Geräte-Proxy-Station_2
PLC_proxy_2
ET 200SP station_1
PLC_1
PC-System_1
HMI_RT_1
PC-System_2
HMI_1
TP1200 Comfort
PLC_proxy_3
Device Proxy
PROFIBUS_1
100%
PROFIBUS_1 [Profibus]
Properties
Info
Diagnostics
General
IO tags
System constants
Texts
General
Network settings
Cable configuration
Additional network devices
Bus parameters
Bus parameters
Cyclic distribution
Enable cyclic distribution of bus parameters
Parameters
Tslot_Init: 300 t_Bit
Max. Tsdr: 150 t_Bit
Min. Tsdr: 11 t_Bit
Tset: 1 t_Bit
Tqui: 0 t_Bit
Gap factor: 10
Retry limit: 1
Tslot: 300 t_Bit
Tid2: 150 t_Bit
Trdy: 11 t_Bit
Tid1: 37 t_Bit
Ttr: 21819 t_Bit
= 14.5 ms
Ttr typical: 882 t_Bit
= 0.6 ms
Watchdog: 65457 t_Bit
= 43.6 ms
Recalculate

Checking inconsistent data

Introduction

During initialization of a device proxy a check is automatically carried out whether the selected data are consistent in the source project and can be imported into the target project. If the source project contains inconsistent data, repair them in the source project in the SIMATIC Manager.

The following tools are available for determining and repairing the inconsistent data in STEP 7 V5.x:

- "Check block consistency" for checking data blocks and alarms
- "Symbol Editor" for checking the symbols

Note

You can find additional information on "Check block consistency" and "Symbol editor" in the STEP 7 V5.x documentation.

Checking data blocks and messages

1. Open the source project in the S7 SIMATIC Manager.
2. Mark the block folder in the project window.
3. Select the "Check block consistency..." entry from the shortcut menu.
   The "Check block consistency" editor opens.
4. Select the "Program > Compile" command from the menu.
   Inconsistencies in the blocks are corrected automatically and objects are compiled.
5. If an inconsistency cannot be corrected automatically, an error message is output. By double-clicking the error message you jump to the faulty object. Correct the inconsistencies manually and save the project afterwards.

Checking symbols

1. Open the source project in the S7 SIMATIC Manager.
2. Start the Symbol Editor by double-clicking the symbol table.
   The Symbol Editor is opened.
3. Invalid global symbols in the table are marked in red.
4. Repair or delete the invalid symbols.

Adapting tags and connections

Introduction

Adapt the connection names and synchronize the tags for establishing the connection after the migration.

When the HMI device is networked with the device proxy PLC the system creates a connection and assigns a connection name. If an HMI connection already existed, adapt the new connection name.

Adapting the connection name

1. Copy the previous connection name.
2. Delete the existing connection.
3. Replace the new connection name by the previous name.

Symbolic connection of the tags

1. Call up the tag editor.
2. Select all the tags.
3. Click "Synchronize to compare with PLC tag".
   The "Options for synchronizing WinCC tags" window opens.
4. Select the options "Data type and absolute address agree" and "Replace WinCC tag name by the PLC tag name" in the window.
5. Click "Synchronize".
   Synchronization of the tags is carried out.

Updating a device proxy via a STEP 7 project file

Introduction

The data of the device proxy PLC have to be updated whenever changes are made in the SIMATIC Manager that affect the HMI device. This is, for example, the case when address area in the data blocks have been changed or extended.

Updating device proxy data

1. Update the data of the device proxy via the project file (*.s7p) of your STEP 7 project.
2. In the update dialog. select the same CPU that you also used during the initialization of the device proxy PLC as the data source.
   An update with a different hardware configuration is not allowed.

3. Through the dialog you have the possibility to select newly created data blocks from the SIMATIC Manager additionally.
   Only selected objects are imported in the TIA Portal project.

   Note

   All the data that were selected during the previous initialization are automatically selected again as well. You should only change this default setting if you consciously want to remove data from the device proxy PLC. If, for example, the DB1 was included in the TIA Portal project through the initialization, and you now deactivate the DB1, the DB1 is deleted from the TIA Portal project during updating.

4. Confirm the selection using the "OK" button.
   Updating of the tags is started.
5. The addresses of the tags were adapted automatically through the symbolic connection of the tags.
   This completes updating of the tags.

Including tags from the SIMATIC Manager

Introduction

Through the initialization of the device proxy PLC you have included the tags from the SIMATIC Manager into the WinCC project.

In WinCC, you insert tags of the device proxy PLC directly into an HMI screen or a screen object.

Inserting a tag into the HMI configuration

1. Open the folder with the program blocks of the device proxy PLC in the project tree.
2. Select the block that contains the tags.
3. All the tags of the selected block are displayed in the detail view.
4. Drag&drop the tag from the detail view into the working area.
   An I/O field with the selected tag is created. Carry out further settings in the Inspector window.

Alternatively configure an I/O field in the HMI screen and drag&drop the tag into the I/O field.

Result

The tag from a block of the device proxy PLC is inserted into the HMI project and connected with the I/O field.

Configuring the "Direct keys" system function

Introduction

If you want to integrate HMI devices that can only be configured with WinCC (TIA Portal), for example SIMATIC HMI Comfort Panels, into the hardware configuration of the SIMATIC Manager, you require the corresponding matching GSD/GSDML files.

The required GSD/GSDM files for WinCC are located in FAQs mutual configuration with WinCC (TIA Portal) and STEP 7 V5.x (http://support.automation.siemens.com/WW/llisapi.dll?aktprim=4&lang=en&referer=%2fWW%2f&func=cslib.csinfo&siteid=csius&groupid=4000002&extranet=standard&viewreg=WW&nodeid4=20229806&objaction=csopen)

Installing GSD files

1. Open the hardware configuration in the SIMATIC Manager.
2. Select the "Options > Install New GSD Files" menu command.
   The "Install GSD files" dialog is displayed.
3. Use the "Browse" button to select the file folder that contains the GSD files.

   Note

   PROFINET / PROFIBUS storage paths in the SIMATIC Manager

   The GSDML files for the HMI devices are located in the hardware catalog under „PROFINET IO > HMI > SIMATIC HMI > GSD > KP/x“.

   The GSD files for the HMI devices are located in the hardware catalog under „PROFIBUS DP > Weitere FELDGERÄTE > MMI > SIMATIC_HMI > HMI CP_x“.

4. Mark all the files and click the "Install" command button.

PROFINET connection: PROFINET IO direct keys

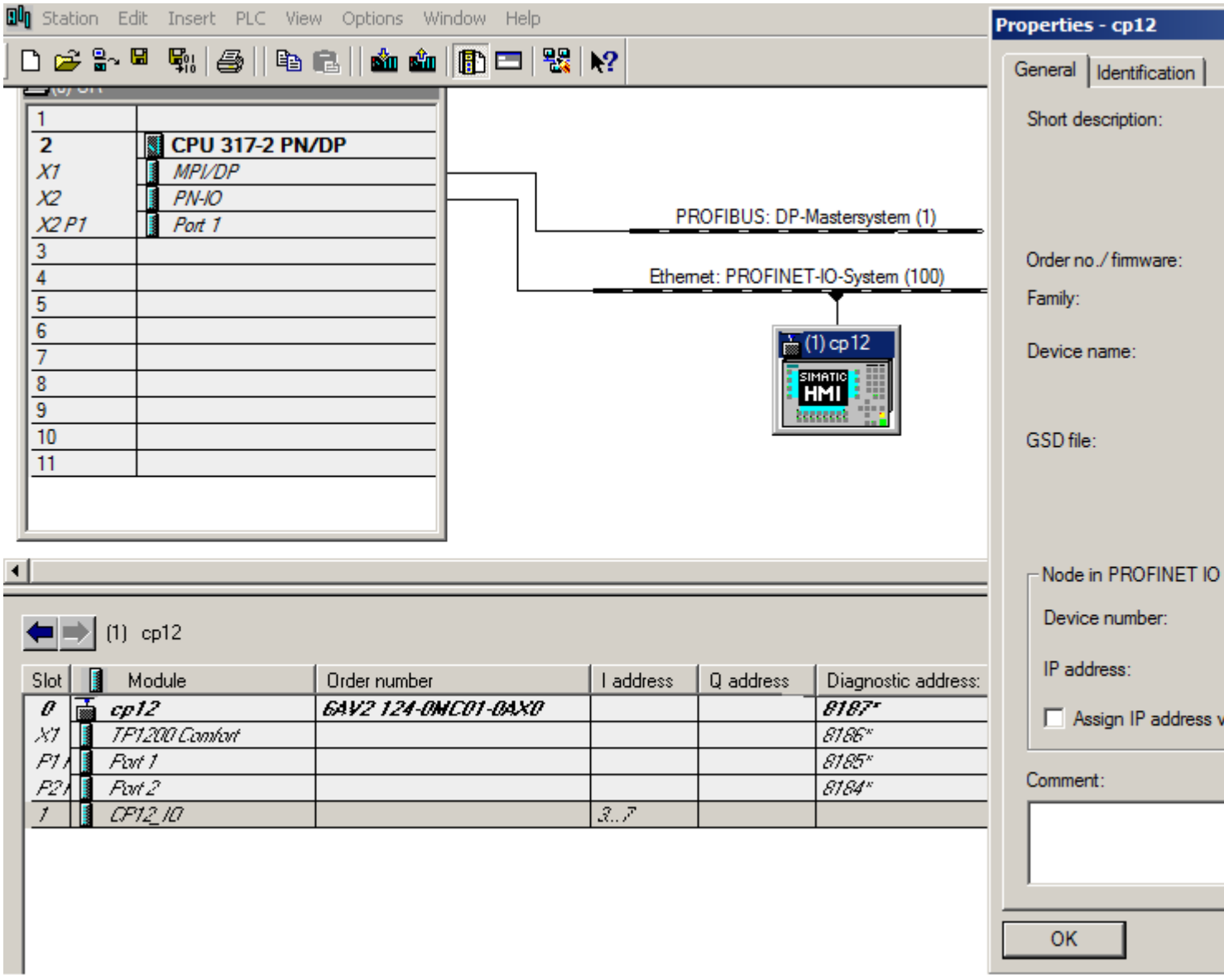
1. Note the properties used for the HMI device to be replaced.
   Take particular note of I/O addresses, device name, device number and diagnostic addresses if these are evaluated.
2. Remove the existing GSD file.
3. Drag&Drop the configured HMI device type from the hardware catalog onto the displayed PROFINET IO bus.
4. Adapt the properties as noted.

   Note

   Take the rules for the names of PROFINET IO devices into consideration for the device name. Use only lower-case characters and no special characters for the device name.

5. Confirm the entries with OK.

6. Save and compile the configuration and transfer the hardware configuration to the controller.



7. Open the device configuration of the TP1200 Comfort and change to the "Device view".
8. Enter the PROFINET device name in the properties of the configured HMI device in WinCC under "Properties > General > PROFINET interface X1 > Ethernet addresses".
   To assign the PROFINET device name, deselect the option "Generate PROFINET device name automatically".

   Note

   The PROFINET name in the WinCC configuration must correspond to the PROFINET name stored in the SIMATIC Manager.

PROFIBUS connection: PROFIBUS DP direct keys

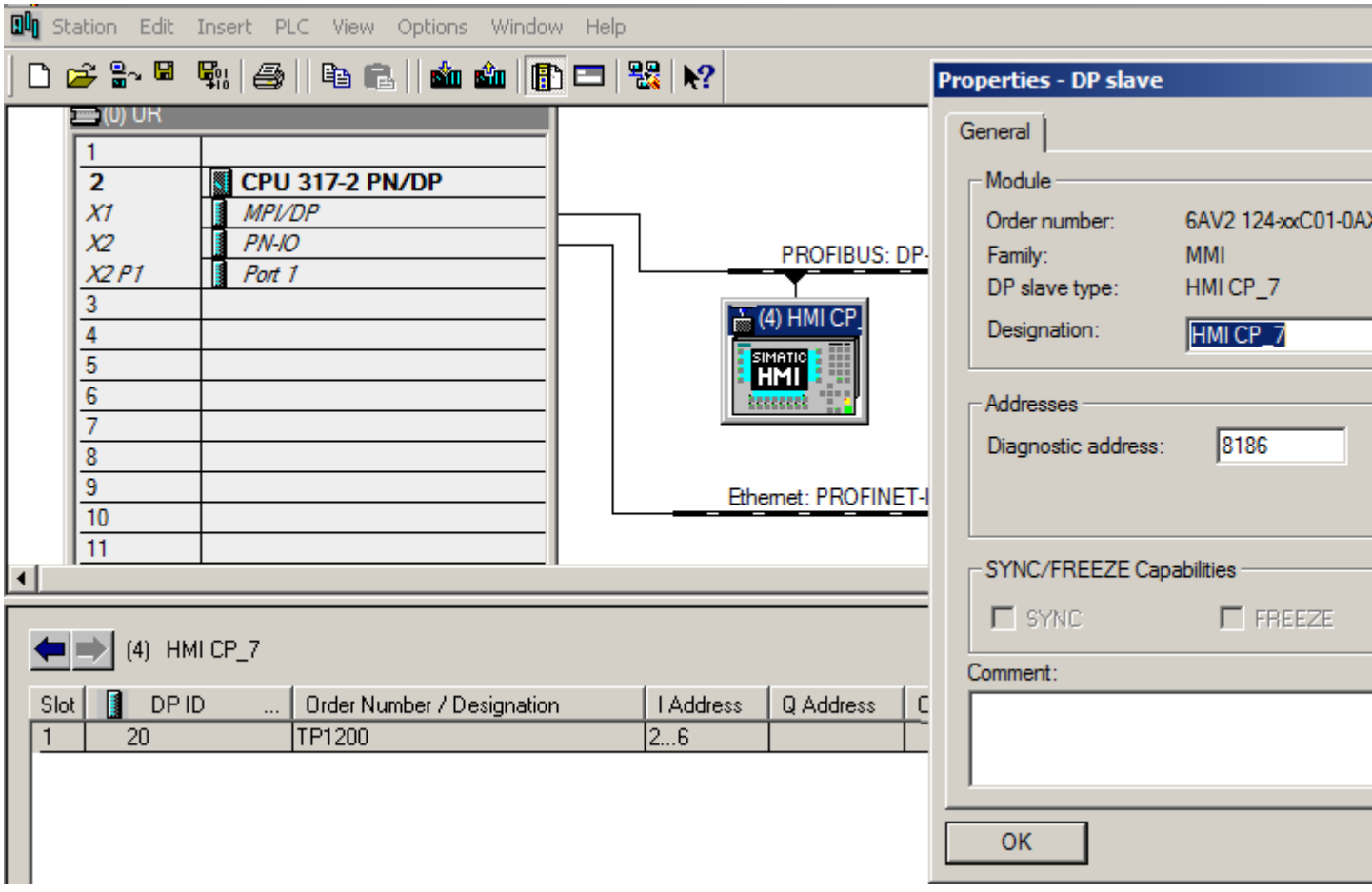
Please note that the direct keys are configured in the SIMATIC Manager.

Note

If direct keys to several CPUs are configured from one panel, the direct keys will only function on one CPU.

The further CPUs display a bus / group error.

1. Note the properties used for the HMI device to be replaced.
   Take particular note of I/O addresses, device name and diagnostic addresses if these are evaluated.
2. Remove the existing GSD file.
3. Drag-and-drop the GSD module that contains the configured HMI device type from the hardware catalog to the PROFIBUS DP bus displayed.
4. Adapt the properties I/O address, device name, PROFIBUS address and diagnostic address as previously noted.
5. Confirm the entries with OK.
6. Save and compile the configuration and transfer the hardware configuration to the controller.



7. Open the device configuration of the HMI device that you want to use and change to the "Device view".
8. Network the PROFIBUS interface in the properties of the configured HMI device under "Properties > General > MPI/DP interface X2 > PROFIBUS address > Network interface with".
9. Specify the PROFIBUS address at "Parameters".

Note

The PROFIBUS address in the WinCC configuration must correspond to the PROFIBUS address stored in the SIMATIC Manager.

15.3 Using Multiuser Engineering

15.3.1 Introduction to Multiuser Engineering

Introduction

In the TIA Portal V14, you have the option of working with Multiuser Engineering .

The advantage of this is that you can work with multiple users simultaneously in multiuser projects. This approach significantly reduces the configuring times for your projects. You can commission your plant faster and start production sooner.

In Multiuser Engineering, you can also use other functionalities of Team Engineering , such as working together on a CPU and exchanging data with Inter Project Engineering (IPE).

This is how Multiuser Engineering works in TIA Portal:

1. Project management is located either on a local or external server.
2. Different users work in local sessions, based on the projects managed by the server.
3. Users work independently from each other in local sessions.
4. The changes from the local sessions are transferred to the server project on check-in.
5. Checked-in changes from other editors are displayed and can be conveniently applied.

Note

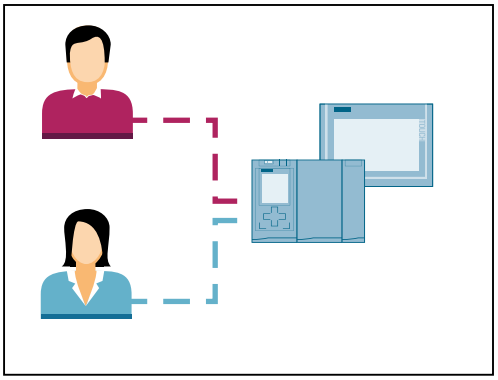
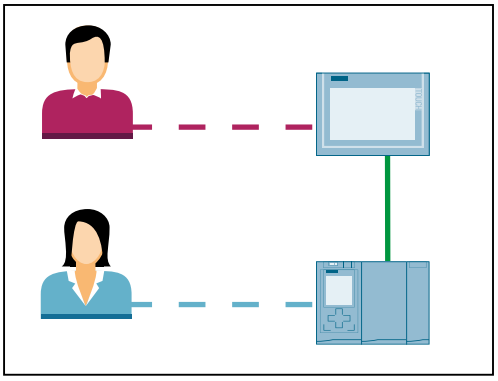
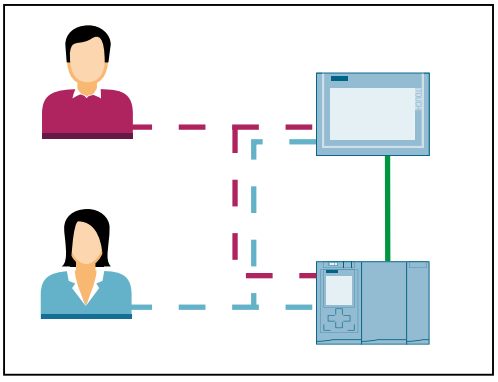
Working efficiently with Multiuser Engineering

When working in parallel in multiple local sessions, you should correct errors that occur at an early stage so that the distributed work on a shared project stays efficient.

Therefore, it is recommended to first compile pending changes before check-in in the multiuser editor. Several compiling options are available for this. If compiling errors are indicated, you should correct them remove before check-in.

Compiling pending changed before check-in ensures that you are checking in and transferring an error-free project to other editors.

Different options are available for working simultaneously on a multiuser project, as can be seen in the table below:

Application scenarios for Multiuser Engineering	
Scenario 1: Several configuration engineers work *across multiple devices* on different objects. In this scenario, each configuration engineer works on the objects assigned to them on different devices.	
Scenario 2: Several configuration engineers work *device-oriented* on different objects. In this scenario, each configuration engineer works on the objects assigned to them on only one device.	
Scenario 3: Several configuration engineers work *technology-based* on different objects. In this scenario, each configuration engineer works on defined technological objects (e.g. Motor_1 or Motor_2) on all existing devices.	

By working on different objects, such as different tags, blocks, PLC tags, interrupts, screens, text and graphic lists, technology objects etc., on one or multiple devices within a multiuser project at the same time, you can make your configuration times more effective and reduce them significantly.

See also

Overview of the mode of operation with Multiuser Engineering (Page 6689)

Basics of the multiuser server concept (Page 6693)

Requirements for working with Multiuser Engineering (Page 6696)

Supported objects for Multiuser Engineering (Page 6699)

15.3.2 Overview of the mode of operation with Multiuser Engineering

Introduction

If you want to work with multiple users simultaneously in multiuser projects, specific procedures should be observed.

Below you will find an overview of the general mode of operation with Multiuser Engineering.

Application example

You can find an application example for working with Multiuser Engineering at:

https://support.industry.siemens.com/cs/ww/en/view/109740141 (https://support.industry.siemens.com/cs/ww/en/view/109740141)

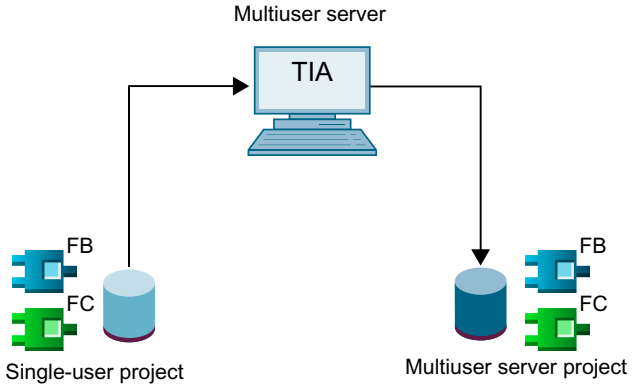
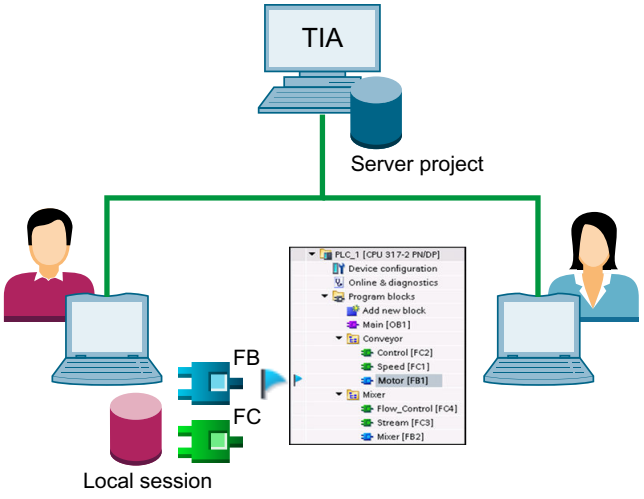
Requirements for working with Multiuser Engineering

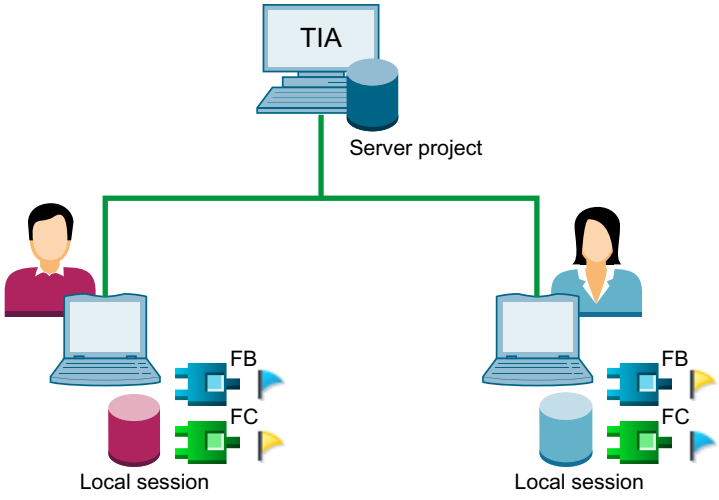
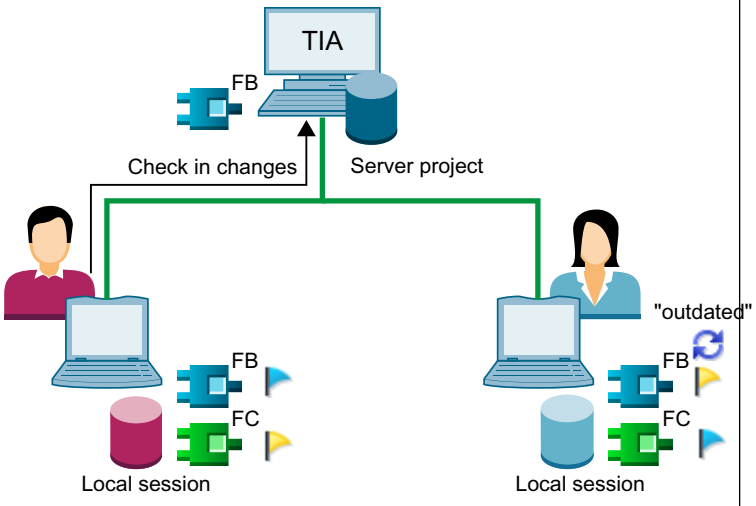
Please observe the following rules for working in multiuser projects:

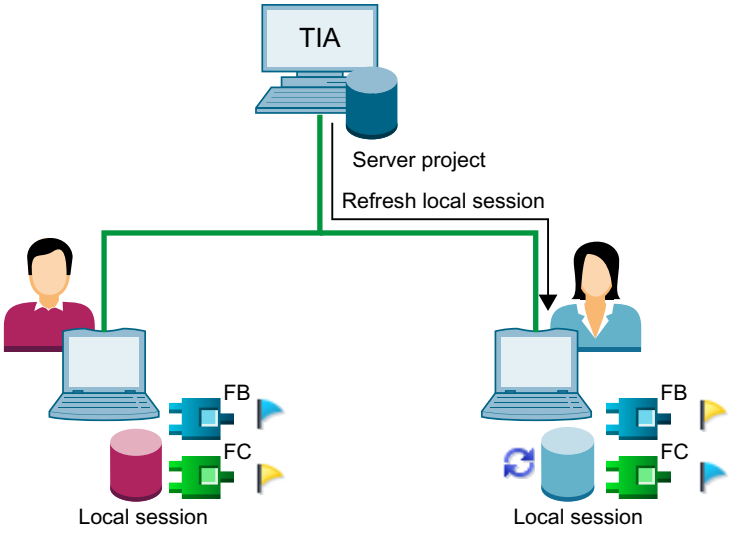
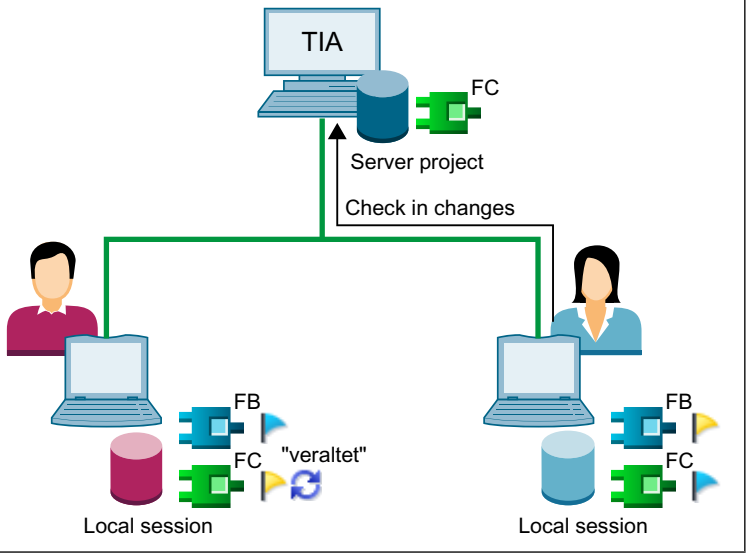
- Before they can be used as multiuser projects, the projects to be edited should already contain the entire hardware configuration, the required blocks, connections and global settings.
  For more information on this topic, refer to: Requirements for multiuser projects (Page 6703)
- Avoid working on the same objects simultaneously by consulting beforehand with the other editors.
- All global settings, such as changes within the device configuration, must be made in the server project view (i.e. in the central server project).
- All objects of a multiuser project can be edited in the server project view. You can also edit objects that do not support the Multiuser Engineering functionality.

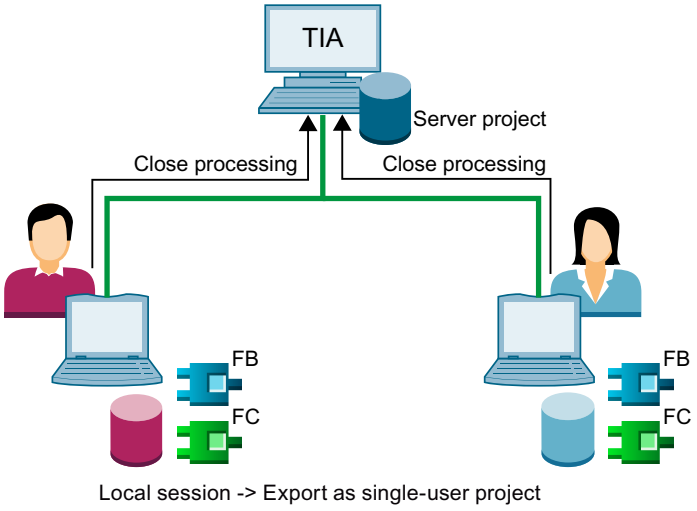
Overview of the mode of operation with Multiuser Engineering

To use the Multiuser Engineering functionality, carry out the following steps:

Action	Example
1. Create multiuser project on the server You install a multiuser server and upload a single-user project to the multiuser server that is being used. • The project should be structured so that the requirements for multiuser projects are fulfilled. **Result:** By uploading a project to the multiuser server, you generate a multiuser server project out of the single-user project.	
2. Create local session for "Editor 1" You create a "local session" as copy of the server project on your computer. • Each editor marks the object they want edit in their local session. • Various objects can be edited at the same time by several users in the various local sessions. **Result:** Objects marked in their own local session are identified with a blue flag. The flags are also displayed in the project tree and the editors of the TIA Portal.	

Action	Example
3. Create local session for "Editor 2" You create a second local session so that the second editor can edit their task in their own local session. • The second editor also marks the object they want to edit in their local session. **Result:** Objects marked in their own local session are identified with a blue flag. Objects marked in another local session are identified with a yellow flag. Therefore, each editor also sees which objects they are working on or which other editors are working on.	
4. Checking in the changes by "Editor 1". As soon as the work in the local session has reached a certain working version, check the changes into the multiuser server project. • The changes to marked objects are applied in the multiuser server project by the check-in. • After the check-in, "Editor 2" is shown in their local session that a object is "outdated". **Result:** The changes of "Editor 1" are transferred to the server project. "Editor 2" can refresh their local session with the new server data, if required.	

Action	Example
5. "Editor 2" updates their local session "Editor 2" is shown in their local session that an object is available in a newer version in the server project and updates their local session. • The newer version is applied by the refresh and inserted in their own local session. • All unmarked objects are overwritten with the current server version, all marked objects are retained. **Result:** The changes from the server project are transferred to the local session of "Editor 2".	
6. "Editor 2" checks in changes to server project "Editor 2" has completed their changes in the local session and checks their blue-marked objects into the multiuser server project. • The changes to marked objects are applied in the multiuser server project by the check-in. • After the check-in, "Editor 1" is shown in their local session that an object is "outdated". **Result:** The changes of "Editor 2" are transferred to the server project. "Editor 1" can refresh their local session with the new server data, if required.	

Action	Example
7. Processing is completed The processing of the various configuration tasks in the local sessions has been completed. **Result:** All changes are checked in to server project. The local sessions and the server project are synchronized. There are no more markings in the local sessions. You can save the multiuser server project as single-user project and continue to use it in TIA Portal as usual. **Note**: If required, you can export the local sessions as single-user project and continue to work on them as usual in the TIA Portal.	

See also

Requirements for working with Multiuser Engineering (Page 6696)

Basics of the multiuser server concept (Page 6693)

Supported objects for Multiuser Engineering (Page 6699)

Requirements for multiuser projects (Page 6703)

Exporting a local session as single-user project (Page 6787)

15.3.3 Basics of the multiuser server concept

Introduction

To work with Multiuser Engineering , you need a multiuser server that manages your multiuser projects and the local sessions.

With the help of the multiuser server you can:

- Create new multiuser server projects in which multiple users can work at the same time.
- Display and manage existing server projects.
- Create and manage local sessions.

Requirement

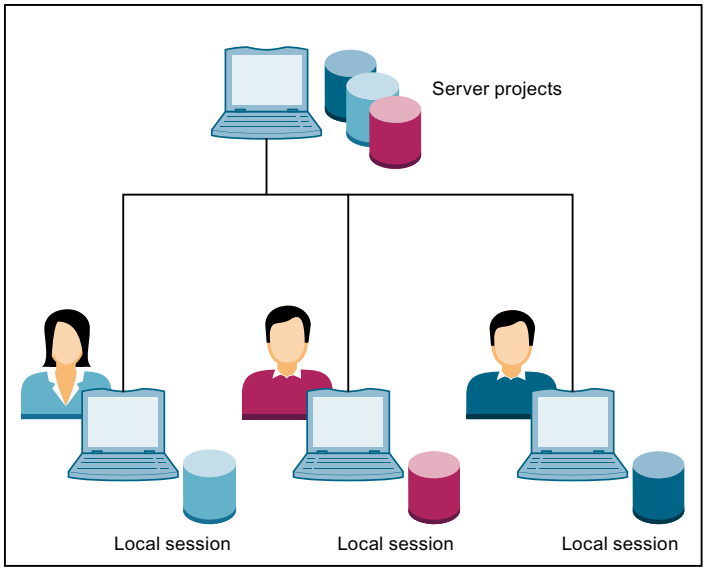
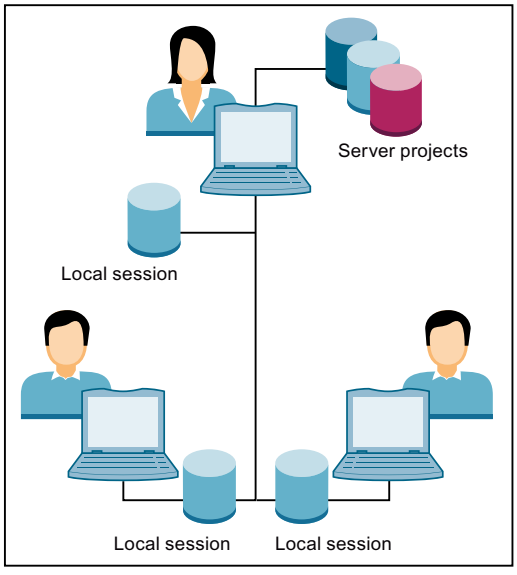
To use the functionality of the multiuser server, you must have installed the multiuser server software suitable for your version of the TIA Portal (e.g. TIA Portal V14 and multiuser server V14) or use the local multiuser server that was installed by default.

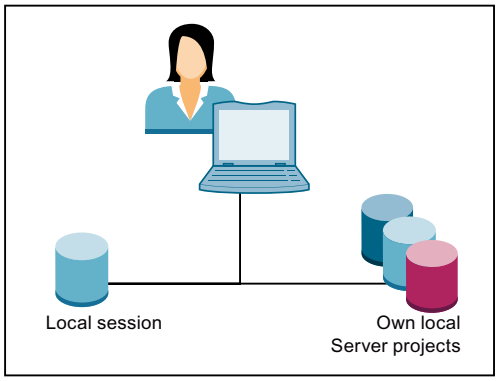
Possible server constellations

Different server constellations for the Multiuser Engineering functionality are available in TIA Portal V14 or higher:

1. The "local" multiuser server that is automatically installed in the TIA Portal.
2. The so-called "temporary multiuser server" in which the server functionality is implemented on a workstation.
3. The multiuser server that is implemented as dedicated external server for long-term collaboration.

The table below provides an overview of working with the possible server constellations.

Overview of possible server constellations for Multiuser Engineering	
1. Multiuser server as dedicated server • The multiuser server can be installed with the TIA Portal or as a standalone product. • This configuration is usually the one that is used as "office network configuration". • The multiuser server runs on a separate computer. • This server constellation can be reached at all times and supports integrated working "around the clock" by default. • The multiuser server must be installed and configured.	
2. Temporary multiuser server: • Editors of the multiuser project can apply the server function to their computer. • This configuration is also a possible "office configuration". • The temporary multiuser server must be installed and configured.	

Overview of possible server constellations for Multiuser Engineering	
3. Local multiuser server: • The local multiuser server runs on the computer of the respective editor. • It is available immediately without additional installation and configuration because it is installed with the TIA Portal. • There is no multiuser license request for the local multiuser server. • The local multiuser server cannot be accessed from other computers over a network. • The local multiuser server is suitable for learning and testing the Multiuser Engineering functionality because it does not require server installation or configuration. • The local multiuser server can also be used for separate project management with "rollback" function.	

See also

Overview of the mode of operation with Multiuser Engineering (Page 6689)

Requirements for working with Multiuser Engineering (Page 6696)

Supported objects for Multiuser Engineering (Page 6699)

15.3.4 Requirements for working with Multiuser Engineering

System requirements

The same hardware and software requirements as for working with TIA Portal V14 apply to working with Multiuser Engineering in V14 .

For information about the regulations in place for the products you have installed, go to "Installation" > "System requirements for installation".

Note

Identical software installation as requirement for working with Multiuser Engineering

To work with Multiuser Engineering without any restrictions, it is important that the same software products of the TIA Portal are installed with identical product versions on all Engineering Systems used.

Note

Access to the Multiuser server

Users can only receive access to the Multiuser server if they are **not** included in the local administrator group of the Multiuser server.

Note

Access to the Multiuser server project

Users can only receive access to the Multiuser server project if they are included at least as a "Member" in the "User Management" of the Multiuser server.

Multiuser server

To work with the full functionality of Multiuser Engineering, you need an installed multiuser server.

You need administrator rights in part for installing, configuring and administering the multiuser server.

The multiuser server is configured and administered with the supplied multiuser tools.

Two alternative procedures are available for this purpose:

- Using the graphical user interface
- Using the multiuser administrative tools and power tools which are executed with the command line.

You can configure and administer the multiuser server using a convenient user interface or with command line commands. Both procedures lead to an identical result.

A detailed description of the procedure is available under "Installing the multiuser server (Page 6711)" and under "Configuring and administering the multiuser server (Page 6714)".

Multiuser network profiles

You have the option of setting the required performance capability for the network profile you are using in the settings for the TIA Portal.

To do so, select the required profile from the drop-down list in the TIA Portal under "Options" > "Settings" > "Multiuser" > "Multiuser network profiles".

You will find a detailed description of the network profiles at Setting the multiuser network profiles (Page 6767)

Licensing for Multiuser Engineering

For working with Multiuser Engineering , you only need a license for a specific version (Floating License) and an associated License Key when you are working with a multiuser server.

As long as you are only using the local multiuser server with limited functionality, you do not need an additional license for the installed TIA Portal.

You can either include the License Key in the installation of the TIA Portal or transfer it using the Automation License Manager after the installation. Multiuser Engineering checks whether a valid license key exists as soon as you open a local session to edit objects.

If a valid license key is found, the local session is opened and you can work in the local session without restrictions.

If a valid license key is not found, you will receive an error message.

You can temporarily activate a "Trial" license to work in a local session.

Once the "Trial" license has expired, you need a valid multiuser license to work in local sessions.

A license is not required for the following actions in the context of Multiuser Engineering:

- Configuring and starting the multiuser server
- Opening the multiuser project management
- Adding projects to the multiuser server
- Creating local sessions
- Deleting local sessions
- Exporting local sessions as single-user projects

Note

Handling licenses

Please observe the general information on handling licenses in the TIA Portal. This information is available in the "Installation" section of the respective products in the information system.

Archiving and retrieving projects

The archiving and dearchiving of local sessions is not supported by Multiuser Engineering.

Archiving and retrieving of multiuser projects results in invalid local sessions, because the connection to the multiuser server is interrupted by these actions.

Opening a local session as reference project

It is not possible to open a local session as reference project when working with Multiuser Engineering. This prevents a local session from being opened multiple times simultaneously and therefore becoming invalid.

Length of the path and file names in Multiuser Engineering

If the path and file names have more than 259 characters in total, an error message is displayed when you are working with Multiuser Engineering. You are notified that the used names are too long and that the requested action cannot be performed for this reason. Shorten the path names or file names in this case so that the required action can be performed.

See also

Setting the multiuser network profiles (Page 6767)

15.3.5 Supported objects for Multiuser Engineering

Introduction

The Multiuser Engineering functionality is available for almost all objects in TIA Portal V14 or higher.

Objects that do not support this functionality yet can be edited in the local session but cannot be marked.

This means they are **not** applied to the server project during check-in. The changes to such objects are again overwritten with the contents of the server project and lost after check-in and after a refresh in the local session.

You can edit unmarked objects in the server project view.

You should make version changes to instructions in the server project view.

In Multiuser Engineering, it is also possible to work with the Team Engineering functionality, such as working on a shared CPU and data exchange via Inter Project Engineering (IPE).

Objects supported by STEP 7 for Multiuser Engineering

The objects listed below for the STEP 7 product are supported:

- Blocks:
  OB, FB, FC, global DBs and instance DBs
- PLC data types and constants:
  With the exception of system data types generated automatically by the system.
- Tags:
  With the exception of system tags that are created and deleted by the system.
- Interrupts:
  With the exception of system interrupts.
  Program interrupts can only be marked for editing together with the associated FB or DB.
- Interrupt classes:
  With the exception of system interrupt classes
- Text lists:
  With the exception of system text lists
- Watch and force tables
- Library elements of the project libraries

- Technology objects:
  With the exception of technology objects for Standard Motion Control and Advanced Motion Control.
- Trace functionality:
  With the exception of online configurations

Objects supported by WinCC for Multiuser Engineering

The objects listed below are supported for the WinCC product:

- Screens:
  With the exception of global screen, overview screen, slide-in screens and global filtering of the alarm view.
- Scripts:
  With the exception of the header file "GlobalDefinitions.h".
- Tags:
  With the exception of system tags, recipe tags, archive tags and raw data tags generated by the system.
- Alarms:
  With the exception of system alarms, controller alarms, predefined system alarm classes (including "Safety Warnings" for KTP700F, KTP900F), system class groups and system alarm groups as well as message blocks.
- Text lists, graphic lists and C text list entries

NOTICE
Editing unsupported objects in the local session Objects that are not supported can be edited in a local session but cannot be marked. Changes to unsupported objects are **not** applied to the server project during check-in. Changes to such objects are lost after check-in and after a refresh in the local session, because they are once again overwritten with the contents of the server project. You can, however, edit such objects that are not supported in the usual way in the server project view.

15.3.6 Settings for working with Multiuser Engineering

Introduction

For working with Multiuser Engineering you can make the following multiuser-specific settings in the

TIA Portal under "Options > Settings" on the "Multiuser" tab:

- General
- Multiuser server connection

- Multiuser network profiles
- Searching in the project
- Storage settings
- Compilation settings

The individual setting options are described below.

General

In the "General" settings, you can select whether the editors opened in the local session are to be hidden when the server project view is opened.

This option is disabled by default.

Multiuser server connection

This dialog is used to add new server connections for a selected multiuser server.

- To create this connection, click the "Add server connection" button.
- In the following dialog box, enter the server name and the associated host and port under "Server Name".
- Select the required transmission protocol for the server connection. "http" is selected by default as transmission protocol.
- Click "Add".

The new multiuser server connection is added and displayed.

See also Introduction to the multiuser server (Page 6766).

Multiuser network profiles

You have the option to select from the drop-down list the required network profile that matches the performance capability of the network you are using

The selected multiuser network profile determines the time until a timeout message is generated.

You will find a detailed description of the individual network profiles at Setting the multiuser network profiles (Page 6767).

Searching in the project

The data for the search must be indexed in order for the global search to also work in local sessions. This option is enabled by default so that the indexing for the search is always executed during check-in.

This is the recommended setting.

If you disable this option, the indexing is not executed during check-in but instead at a different time.

The indexing data is then transferred first as file to the multiuser server. However, a search in the local session is only possible after indexing has been successfully completed. If no indexing was performed during check-in, this can result in somewhat longer waiting times when searching and prolong the check-in times of other editors that have activated this option.

Storage settings

Here you can define where your local sessions and your multiuser server projects will be saved.

- Storage location currently used
  This option is enabled by default.
- Defining a storage location by default
  Select the respective required storage location here by navigating to the required location. Click "Browse" for this purpose. Your settings are applied as default.

Compilation settings

Define here the required compilation settings for your local sessions. The settings made here are applied as defaults in the multiuser editor and can no longer be changed there.

Select the required settings.

- Compile before check-in
  This option is enabled by default and means that a compiling process is executed before the check-in.
- Compilation settings:
  This setting defines the extent to which the software changes are compiled. You have the following options.
  - Marked objects (changes only)
    The changes made to the marked objects for the selected device are compiled together with the device-specific objects.
  - Device software (changes only)
    The changes made to the marked objects are compiled together with the software version on the assigned devices.
    Example: You have marked and changed several objects that belong to more than one device (for example to CPU_1 and to CPU_2). These marked objects are compiled together with the software versions of both CPUs during compiling.
- Disable compilation settings in the multiuser editor
  If you activate the option, you can no longer change the compilation settings in the multiuser editor.
  The settings you made here then apply as default.

See also

Setting the multiuser network profiles (Page 6767)

Introduction to the multiuser server (Page 6766)

15.3.7 Requirements for multiuser projects

Introduction

To work with Multiuser Engineering simultaneously and in parallel, it is very important that the multiuser server project that serves as the work basis is clearly structured.

Requirements for multiuser server projects

To create multiuser capable server projects, start with a single-user project that fulfills the following requirements:

- The project should already include the hardware configuration with all required connections.
- Divide the user program into program parts that are independent of each other.
- Create a useful, technology-oriented project structure with folders and groups for the objects to be edited by the individual users.
- When structuring the project, make sure that multiple users can work with the different project directories simultaneously and exclusively.
  Example: One user edits the blocks in group 1 while another user simultaneously edits the blocks in group 2.
- Use a main OB and a central FB or FC for each program part which calls the functions of the subprogram.
- If possible, create a separate PLC tag table for each group.
- You should define in advance all needed project languages that are to be used by the engineering systems involved.
- If possible, create all connections that are to be edited in the multiuser project.
- Define the screen resolution and specify the size of the HMI screens.
- Use global data blocks to save the data for the individual program parts, no bit memories.
- When you load a single-user project created according to these rules to a multiuser server, it becomes a multiuser project.
- Edit the defined objects of the multiuser project simultaneously with multiple users in the local sessions. When you create new objects in the local sessions, make sure that you use different symbolic names.

Example of a server project for multiple users

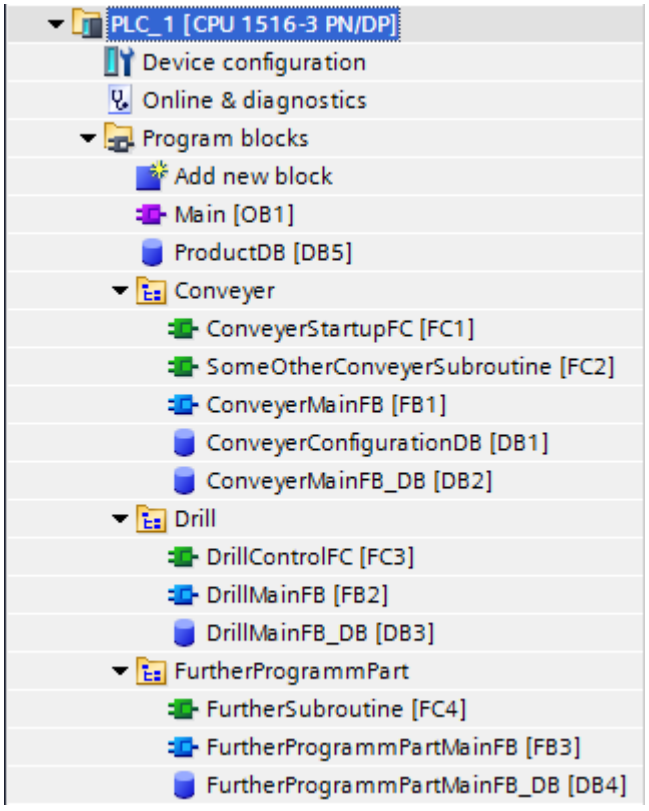
The figure below shows an example of a server project for multiple users with a clear structure that permits parallel editing of individual objects in the context of Multiuser Engineering .

Each program part contains a "main FB", which calls the specific lower-lever functions for this program part.

The program parts edited by the individual Engineering Systems are:

- Program part 1: "Conveyer"
- Program part 2: "Drill"
- Program part 3: "FurtherProgramPart"

In the "Conveyer" program part, the "ConveyerMainFB" calls the "ConveyerStartupFC" function.



15.3.8 Operator controls in Multiuser Engineering

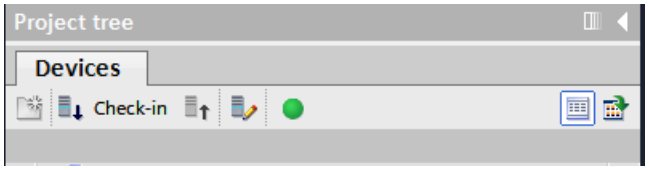
15.3.8.1 Multiuser icons in the user interface

Introduction

When you are working with Multiuser Engineering and a local session is open, you can see multiuser-specific icons in the TIA Portal user interface.

Multiuser icons in the toolbar

The Multiuser Engineering user interface shows the following icons in the toolbar when a local session is open:



Icon	Meaning
	**Check-in** When you click this icon, the multiuser editor opens in the change view. In the modification view, all objects marked for check-in are displayed and you can select and check in the desired objects.
	**Refresh** When you click this icon, the multiuser editor opens in the "Refresh" view. In this view, you can, if you wish, refresh your local session with the new contents of the server project.
	**Open/close server project** When you click this icon, the multiuser server project associated with your local session is opened for editing. You can then make your changes directly in the server project.
	**Server status "connected"** This icon indicates that the associated multiuser server is available.
	**Server status "locked"** This icon indicates that the associated multiuser server is currently being used by a user and is therefore locked. The server is also displayed as locked when a user is working the server project view. If you place the mouse pointer on this icon, you will receive additional information about the lock.
	**Server status "not connected"** This icon indicates that the associated multiuser server is not connected.
	This overlay icon shows that a newer version of the server project is available on the multiuser server.

Icons for display of multiuser projects

The following icons are displayed for multiuser projects and local sessions in the user interface:

Icon	Meaning
	**Multiuser server** This icon identifies a multiuser server.
	**Multiuser project** This icon identifies a multiuser project associated with a multiuser server.
	**Local session** This icon identifies a local session that is associated with a multiuser project that is assigned to a multiuser server.

15.3.8.2 Multiuser icons in the user interface

Introduction

When working with Multiuser Engineering , multiuser-specific icons are displayed in the TIA Portal user interface which make it easier for multiple users to work at the same time. A unique, user-specific visualization is intended to make collaboration within multiuser projects and in local sessions more effective and transparent.

Marking objects

To edit an object using Multiuser Engineering, it should first be "marked". Only objects marked for check-in can be transferred into the server project after editing.

Marking can only be added for "marked objects". Objects that cannot be edited in the local session also cannot be marked.

Such objects can, however, be edited in the server project view.

The icons described below show you which objects were marked by you or other editors.

Icons for marking in Multiuser Engineering

The following icons are used for marking in Multiuser Engineering :

Icon	Meaning
	**Markable object** This object can be marked for check-in by clicking it.
	**Marked object** The object was marked in your own local session for check-in.
	**Marked object** The object was marked for check-in in a different session or in the server project view.
	**Marked object with conflict** The object was marked for check-in in multiple local sessions or in a local session and in the server project view and causes a conflict. An object should only be marked for check-in once within the entire multiuser working area (with the local session and within the server project view). An object marked in red can still be checked in. However, the check-in may cause unwanted overwriting in the server project because only the last checked in version is applied in the server project.
	**Object is outdated** If one of the icons mentioned above Is also identified with this overlay, the object does no longer correspond to the latest version of the server project and should definitely be updated. If the object is checked in without refresh, the editing of the other user is overwritten and thus cancelled. The refresh ensures that all editors of a multiuser project have a consistent version.

Note

Objects without flags

Note that objects without flag can only be edited in the server project view and not in the local session.

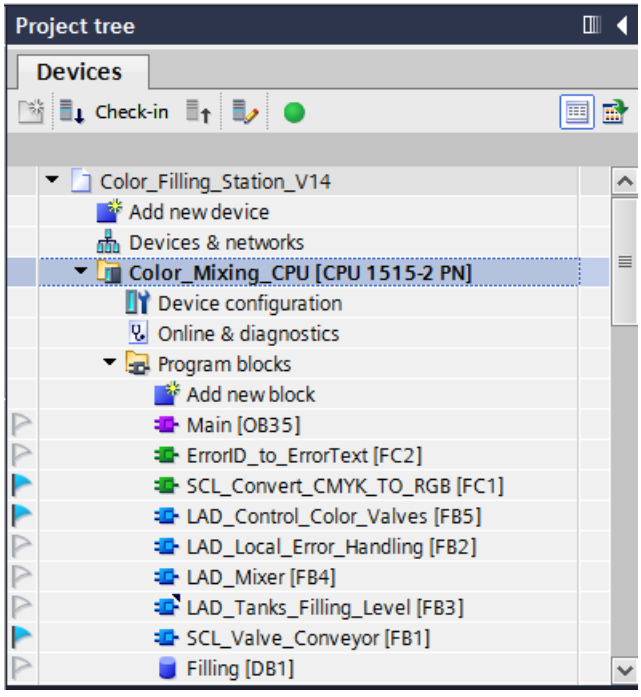
Display of multiuser markings in the TIA Portal

When you are working with Multiuser Engineering in the local session or in the server project, the multiuser-specific markings are displayed in the following views of the TIA Portal in all editors:

- In the project tree
- In the detail view
- In the overview window
- In the library views

An additional first table column is available in all table displays within the project tree when working with Multiuser Engineering. The first table column is reserved for the multiuser-specific markings described above.

The following figure shows an example of multiple selected objects in the project tree:



15.3.8.3 Multiuser banner in the user interface

Introduction

When working with Multiuser Engineering , you are supported by the display of multiuser-specific banner texts in editors.

Banner texts include notes and warnings on the further procedure which make your work easier.

Banner texts can be expanded and collapsed by clicking the yellow circle with the exclamation point.

Example of a multiuser banner text



Functionality of the multiuser banner texts

Banner texts are displayed both when working into the local session and when working in the server project view.

As soon as you need additional information to work with Multiuser Engineering , a yellow banner text appears.

The multiuser banner texts have the following function:

- Information about how to edit the opened object:
  - Banner texts point out, for example, that the opened object has not been selected for check-in yet and your changes therefore cannot be applied to the server project.
  - Or they make you aware that this specific object cannot be edited in the local session but only in the server project view.
- Links make for easier execution of the further sequences required.
  - You can use the links, for example, to conveniently mark the open object or open the server project view in which you will continue editing.

As soon as you have executed the required sequences in the individual editors, the banner texts are partially deleted in the editor once again and you can continue working as usual.

15.3.8.4 Functionality of the multiuser editor

Introduction

You can call the multiuser editor with the "Check-in" and "Refresh local session" buttons in the Multiuser Engineering toolbar.

The multiuser editor has two different views, depending on the button with which the editor was started in the multiuser toolbar:

- "Check-in" view: Opened by clicking the "Check-in" button in the toolbar of the project tree.
- "Refresh" view: Opened by clicking the "Refresh local session" button in the toolbar of the project tree.
  The "Update" view can always be opened even if the local session does not contain any objects that can be updated.

The structure of the multiuser editor is the same in both views, but each view offers different buttons and options.

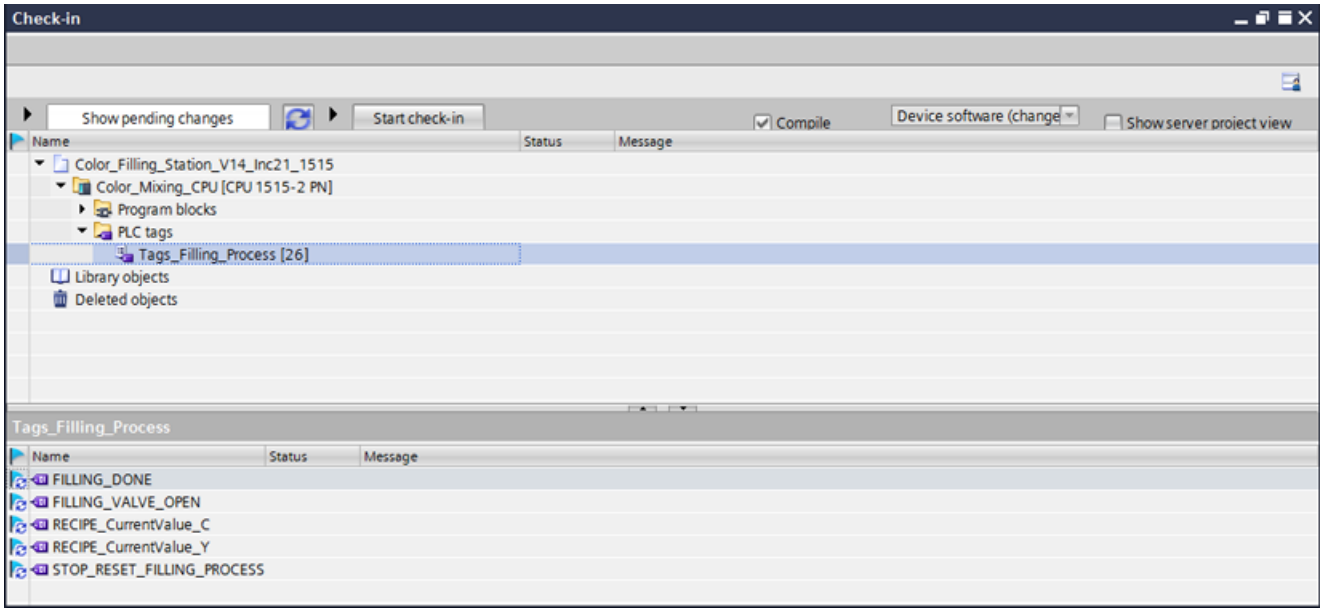
Note

Displaying in the multiuser editor

If you have opened the multiuser editor and simultaneously make changes in your local session, these changes are only visible in the open editor after a "Refresh". To do this, click the "Refresh" button (circular arrow) in the open editor.

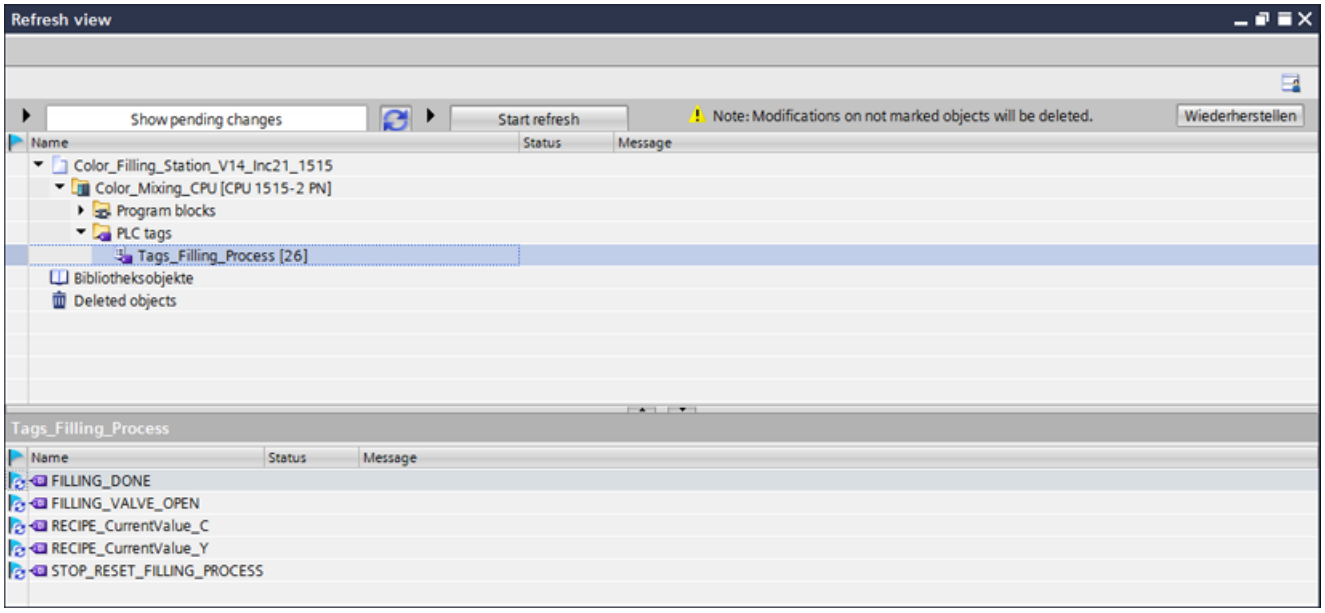
Multiuser editor in the "Check-in" view

The figure below shows an example of the multiuser editor in the "Check-in" view:



Multiuser editor in the "Refresh" view

The figure below shows an example of the multiuser editor in the "Refresh" view:



Structure of the multiuser editor

The multiuser editor has the following structure:

1. Toolbar:

Shows the icon for saving the layout settings.

2. Button area:

Shows the multiuser-specific buttons for the check-in and refresh.

3. Working area:

Shows the change view prior to check-in or refresh.

- Shows all objects marked in the local session for check-in and refresh as these are also displayed in the project tree.
- Shows the status messages with the results for the check-in and the refresh.
- The "PLC_Tags" folder can be opened.
  Clicking on the tag table in the bottom window in the detail view shows the tags selected for check-in.
- The "Library objects" folder is only displayed when library objects were marked for check-in.
  If the folder is selected, the selected library objects are displayed in the detailed view.
- The "Deleted objects" folder is only displayed if objects were deleted in the local session.
  If the folder is selected, all objects that were deleted in the local session are displayed in the detailed view.

4. Detailed view:

Shows information on lower-level elements (in this case, the selected PLC tags) and status messages.

Note

Procedure for working with the multiuser editor

The procedures for working with the multiuser editor are described in detail in the sections below.

15.3.9 Installing and uninstalling the Multiuser server

15.3.9.1 Notes on the installation of the multiuser server

Introduction

Installation of the multiuser server gives you access to the Multiuser Engineering functionality that is available with TIA Portal V14.

You can install the multiuser server as follows:

- Together with the installation of the TIA Portal.
- As standalone installation by means of a separate installation process without TIA Portal.

Note

Administrator rights are required for the server installation.

Installation of the multiuser server with the TIA Portal

The multiuser server is installed together with the following TIA Portal products:

- SIMATIC STEP 7 Basic V14.0
- SIMATIC STEP 7 Professional V14.0
- SIMATIC WinCC Basic V14.0
- SIMATIC WinCC Comfort/Advanced V14.0
- SIMATIC WinCC Professional V14.0

The multiuser server is usually activated in the product selection by default and is then installed with the TIA Portal. Check the settings during installation of the TIA Portal. You can find the specific default settings for the multiuser server during setup when selecting the product in the "Tools" directory. By selecting or clearing the options, you can specify if the multiuser server is to be installed along with the TIA Portal or not.

Installation of the multiuser server as standalone installation

The installation of the multiuser server can also be a standalone installation.

A description of automated installation is also available on the product DVD in the directory "Documents\Readme\<language directory>".

Identical TIA Portal versions during installation

When installing different TIA Portal products on various clients, make sure that all participants use the same versions of software products for the installation.

For example, if you have installed STEP 7 Professional, WinCC Advanced and Safety as well as the multiuser server with version V14.0, all other Team Engineering participants must also have installed this software in identical versions.

The service packs and updates must be installed for all products at the same time.

Installation path

Do not use any UNICODE characters (for example, Chinese characters) in the installation path.

Antivirus programs

During the installation, read and write access to installed files is necessary. Some antivirus programs block this access. We therefore recommend that you disable antivirus programs during the installation of the multiuser server and enable them again afterwards.

15.3.9.2 Installing the multiuser server

Introduction

The multiuser server V14 can be installed as part of the TIA Portal V14 installation or as standalone product.

A description of automated installation is also available on the product DVD in the directory "Documents\Readme\<language directory>".

Installation rules

Multiple versions of the multiuser server can be installed on a computer, but only one version of the multiuser server can be run with the corresponding version of the TIA Portal.

Example: The multiuser server V14.0 can only be run with TIA Portal V14.0.

Requirements for installation

The same requirements apply to the installation of the multiuser server V14.0 as for the installation of the TIA Portal V14.0.

For more details, see: AUTOHOTSPOT

Installing the multiuser server with the TIA Portal

Follow these steps for installation:

1. Note the requirements for the installation of the selected software package.
2. Start the setup program of the required product by inserting the installation medium into the respective drive.
3. Select the preferred settings and click the "Install" button.
4. Make sure that the multiuser server is selected in the product selection and observe the notes in the setup dialogs.

Result

The multiuser server is installed on your computer together with the TIA Portal.

Installing the multiuser server as standalone product

Follow these steps for installation:

1. Note the requirements for the installation of the multiuser server.
2. Start setup for the standalone installation of the multiuser server with a double-click on the self-extracting .exe file "TIA_Portal_Multiuser_Server_V14.exe". You can find these files in the directory "Support" on the corresponding product DVD.
3. Select the preferred settings and click the "Install" button.
4. Follow the instructions in the setup dialogs.

Result

The multiuser server is installed on your computer.

15.3.9.3 Uninstalling the multiuser server

Introduction

You can use the Control Panel to remove the multiuser server.

Requirements for removal

There is a multiuser server installed on your computer.

Uninstalling the multiuser server

Proceed as follows for removal:

1. Open the Control Panel.
2. Click on "Uninstall a program" in the Control Panel.
   The "Uninstall or change a program" dialog opens.
3. Right-click the entry for the "Multiuser server" in the "Uninstall or change a program" dialog, and click "Uninstall".
   A security prompt appears.
4. Click "Yes" to confirm this prompt.

Result

The multiuser server is removed from your computer.

15.3.10 Configuring and managing the multiuser server

15.3.10.1 Options for configuring and administering the multiuser server

Introduction

The following options are also available for configuring and administering the multiuser server:

- Using the graphic tools (Configuration Tool and Administration Tool).
- Using the command line tools (Multiuser Administrative Tools and Multiuser Power Tools).

You can configure and manage multiuser server with both methods.

The graphical tools are easier to operate using the graphical user interface. The command line tools have a somewhat wider range of functions with respect to the error log. You can compress and send the error logs as a zip file.

You have the option of using the procedure you prefer.

Administrator rights are required for configuring the multiuser server irrespective of the selected procedure.

You need administrator rights for some for administrative tasks (for "User Management") and users' rights.

Note

Available languages

The tools for configuring and administering the multiuser server are available in English only.

Configuration and administration with the graphic tools

The graphic tool provides you with a very convenient way of configuring and administering the multiuser server.

You see at a glance all setting options and can activate these by selecting the corresponding options.

You can find the exact procedure in: Using the graphic tool (Page 6715)

Configuration and administration with the command line tools

The configuration of the multiuser server is performed with the administrative tools.

The power tools are used for administering the multiuser server.

The power tools for performing administration tasks on the multiuser server are also available to you within the TIA Portal.

You can find the exact procedure in: Using the command line tools (Page 6734)

Note

Working with the command line tools

For working with the command line tools, you should have prior experience handling inputs in the command line.

If you lack this experience, you should use the more user-friendly graphic user interface for configuring and administering the multiuser server.

See also

Using the command line tools (Page 6734)

Using the graphic tool (Page 6715)

15.3.10.2 Using the graphic tool

Introduction to the graphic multiuser tools

Introduction

The multiuser server can be easily configured and administered with the help of the graphic tools.

You use the "ConfigurationTool" to configure the multiuser server.

You use the "AdministrationTool" to administer the multiuser server.

You need administrator rights in some areas to configure the multiuser server.

Note

Configuration tool and Administration tool

The user interfaces for both tools are available in English only.

Requirement

The multiuser server is installed.

Steps for configuring the multiuser server with the "Configuration Tool"

To configure the multiuser server, the following steps are required:

- Open the graphic tool for the multiuser server configuration with administrator rights.
- To open the start menu, click the command "Start" > "All Programs" > "Siemens Automation".
- Click on the "Multiuser Server V14 Configuration" entry to open the Configuration Tool.
- Enter the required information and click "Install service".

You can find the exact procedure in: Configuring a multiuser server with the graphic tool (Page 6717)

Steps for administering the multiuser server with the "Administration Tool"

To administer the multiuser server, the following steps are required:

- Open the graphic tool for the multiuser server administration.
- To open the start menu, click the command "Start" > "All Programs" > "Siemens Automation".
- Click on the "Multiuser Server V14 Administration" to open the Administration Tool.
- Insert a connection to the desired multiuser server and enter the data for the administration of the server.
- Assign the desired roles and rights for working with the newly created multiuser server connection under "User Management".
- Administer your local sessions under "Sessions".
- Under "History", you will find the complete history with all activities for your projects.

You can find the exact procedure in: Administering a multiuser server with the graphic tool (Page 6721)

See also

Configuring a multiuser server with the graphic tool (Page 6717)

Administering a multiuser server with the graphic tool (Page 6721)

Configuring a multiuser server with the graphic tool

Introduction

You use the graphic "Configuration Tool" to configure the multiuser server.

Note
Configuration Tool

- The user interface for the Configuration tool is available in English only.
- You need administrator rights to work with the Configuration Tool.

The following functions can be executed with the "Configuration Tool":

"Server status" area

- Contains the above-displayed contents the first time the Configuration tool is opened. This area is not editable.
- Once the configuration has been successfully completed, the server can be uninstalled, started and stopped in this area.

"Connection settings" area

- Allows you to select the connection data.

"Security" area

- Offers the option to create a new certificate or to use an existing certificate for a selected https connection.

"Storage" area

- Allows you to select the storage location for multiuser projects.
- Specifies how many project versions are available as "rollback".

Note
Changes to existing inputs

To apply changes to existing inputs in the Configuration tool, you must click the "Apply" button.

Note

Memory space on the multiuser server

For the project management, you need to have sufficient free space on the multiuser server, depending on the project size and the number of project revisions.

Basically, the provided space should be at least five times the expected size of the project.

When using more than three project revisions, the free memory space must be increased accordingly. You set the number of project revisions when configuring the multiuser server.

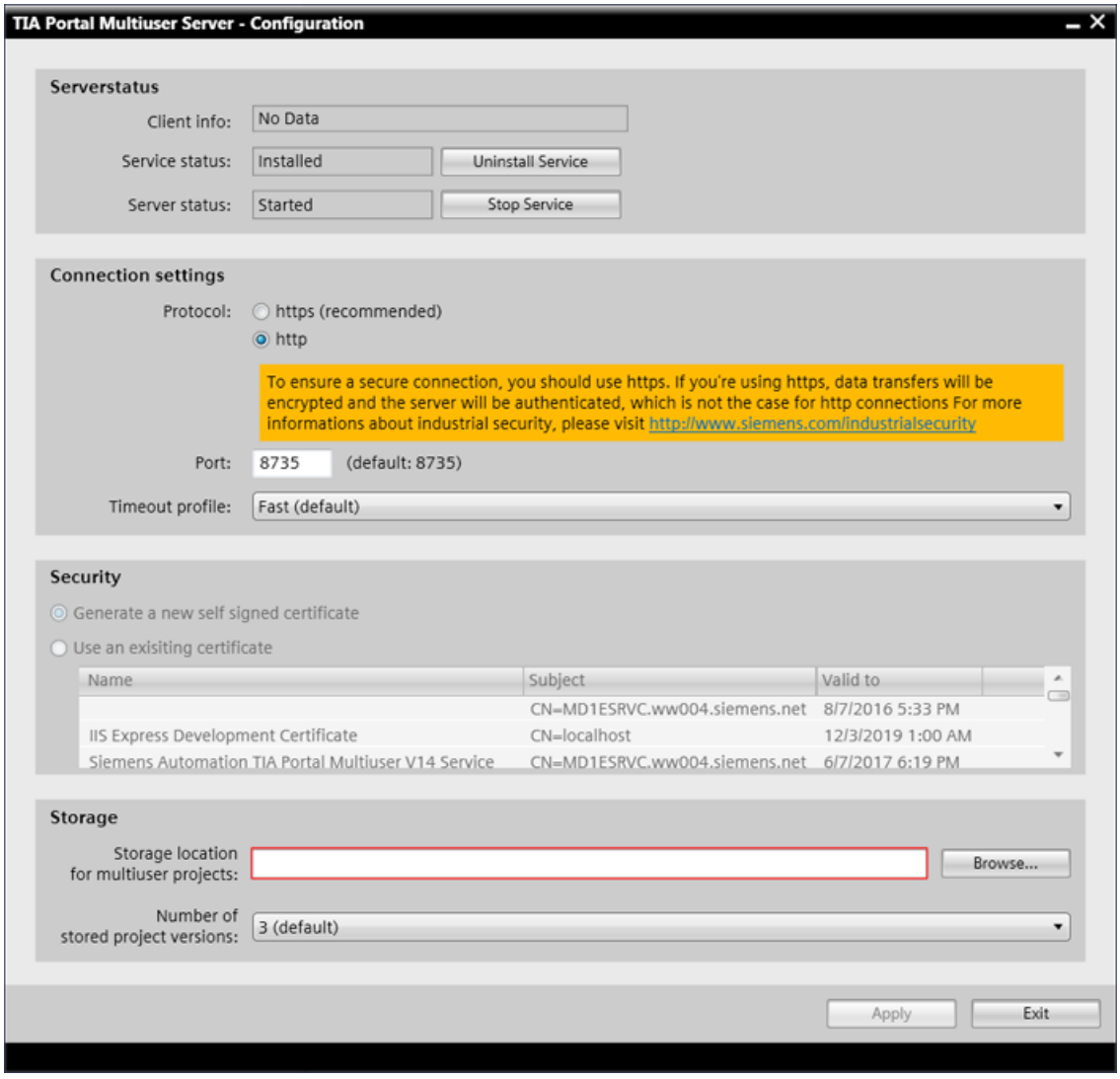
Generally, you should provide sufficient storage capacity and also take into account or evaluate the messages of the operating system as the memory space decreases.

Requirement

The multiuser server is installed.

User interface of the Configuration tool

The user interface of the Configuration tool contains the following the first time it is opened:



Configuring the multiuser server with the "Configuration Tool"

To configure the multiuser server, follow these steps:

1. Open the graphic tool for the multiuser server configuration with administrator rights.
   To open the start menu, click the command "Start" > "All Programs" > "Siemens Automation".
   Click on the "Multiuser Server V14 Configuration" entry.
   The user interface of the Configuration Tool then opens.
2. In the "Connection settings" area, select the desired protocol.
   Default selection is "http". The text with yellow background is hidden as soon as you click on "https".
   With "https", communication is encrypted and a certificate is requested. This means communication has a higher security level than the unencrypted "http" communication.
3. Enter the desired port for the server connection if you do not want to use the default setting.
4. Select the desired "Timeout profile" for the server from the drop-down list.
   This setting defines the timeout behavior for the server.
   You have the option of selecting between three profiles: "Fast", "Middle" and "Slow", where the default setting is "Fast".
5. "Security" area: If you have selected "https" as the protocol, the "Security" area is activated.
   Select the desired option for the certificate.
   You have the option of creating a new certificate or using an existing certificate on your computer.
   If you activate the first option, a new certificate is created after successful completion of the configuration.
   If you activate the second option, you can click on the desired certificate in the displayed list of certificates to select it.
6. Under "Storage", specify the desired storage location for multiuser projects. You can enter the path directly or navigate to the desired location via "Browse".
7. Select the number of project versions to be stored from the subsequent drop-down list.
   Three project versions are saved by default.
   You can choose to store between "1" and "10" project versions.
8. Click "Install service" to complete the configuration.

Result

A message informs you that the configuration of the multiuser server was successful.



In the "Server status" area, you will find the URL for the multiuser server and the fingerprint for the generated certificate under "Client info".

Example: https://XY12345.ww00X.siemens.net:8735/

C35F81B179062BA92A3EAE4DBF18B0DEA9F50D8A

If you access the server from the client side, you should always check the certificate in order to ensure secure communication with the multiuser server. The configuration of the multiuser server is now complete.

Error correction in the Configuration tool

If warnings or errors occur, you will find specific information about these in ServerConfiguration.log.



Proceed as follows:

1. Click on "View Log" in the yellow highlighted area in order to open ServerConfiguration.log.
2. Correct the errors and warnings displayed in the log file.
3. Restart the Configuration tool if it is no longer open.
4. Click the "Install Service" button.

Result

A message informs you that the configuration of the multiuser server was successful.

The configuration of the multiuser server is now complete.

Administering a multiuser server with the graphic tool

Introduction

You use the graphic "Administration Tool" to manage the multiuser server.

Note

Administration Tool

- The user interface for the Administration tool is available in English only.
- To work with the Administration Tool, you need administrator rights for "User Management".
- The Administration Tool is also available outside the multiuser server.

The following functions can be performed with the "Administration Tool":

1. Add and delete server connections ("Add server" and "Remove server")
2. Define "User Management" for the server connections ("Manager", "Contributor", "Member")
3. Manage existing multiuser projects and sessions
   - Specify user management for projects and sessions: ("Manager", "Contributor", "Member")
   - Projects: Lock, unlock, and delete ("Lock/Unlock project", "Delete project")
   - Sessions: Display and delete ("Delete selected session")
4. Show "History": Project history with all data and display of the stored project revisions ("Rollback to selected revision" and "Export selected revision")

Requirement

The multiuser server is installed.

The multiuser server has already been configured.

You have already created projects as multiuser projects and created local sessions in the TIA Portal.

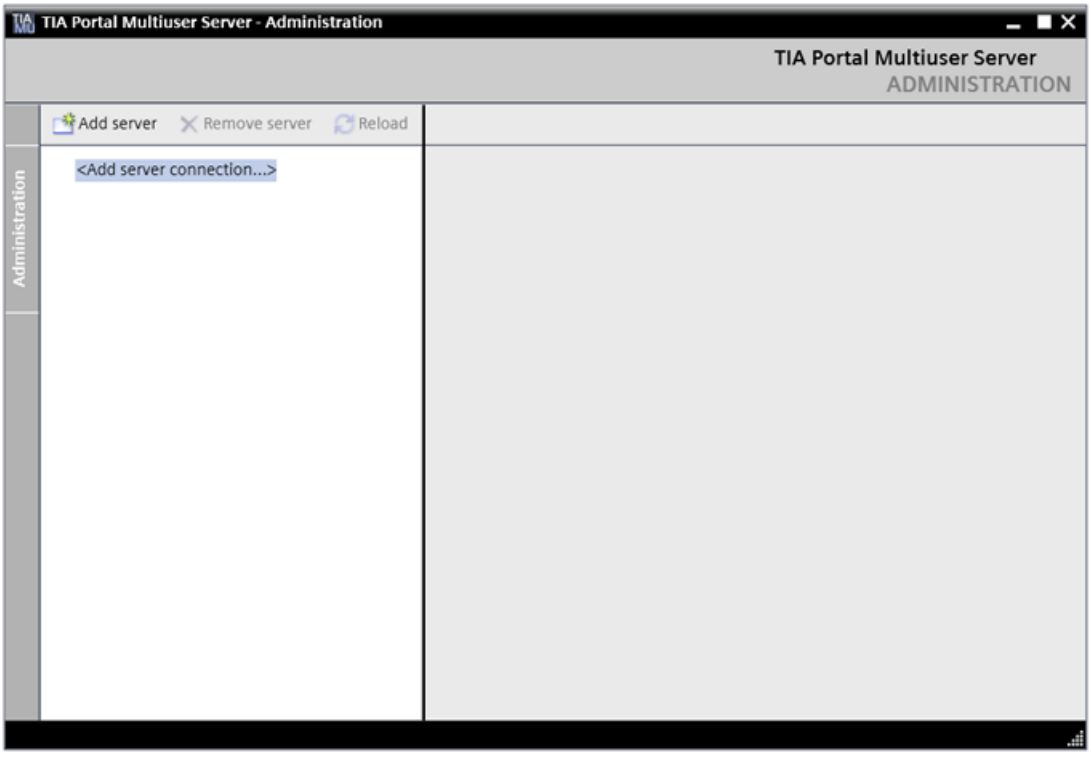
Note

Multiuser projects and local sessions

If you have not yet created multiuser projects and local sessions on the multiuser server in the TIA Portal, no information regarding these will be displayed when you administer the multiuser server.

User interface of the Administration tool

The user interface of the Administration tool has the following appearance the first time it is opened:



Structure of the Administration Tool

The left side of the tool shows you a list of configured server connections once you have established a connection to the multiuser server via "Add server connection".

On the right side, you receive additional information about the objects selected on the left in a detail view.

Both pages contain their own toolbar for executing the displayed administration functions.

The view can be expanded and collapsed by clicking the "Administration" button.

Note

"Reload" button

Clicking this button updates the display for the selected object in the Administration Tool.

The exact procedures for working with the Administration tool are described in the following.

See also

Configuring a multiuser server with the graphic tool (Page 6717)

Adding and deleting a new server connection

Introduction

You can add and delete new multiuser server connections with the graphic "Administration" tool.

These commands can be executed using the commands in the toolbar or from the shortcut menu.

Note

Administration Tool

- The user interface for the Administration tool is available in English only.
- You need administrator rights to work with the Administration tool.

Requirement

The multiuser server is installed.

The multiuser server has already been configured.

You have already created projects as multiuser projects and created local sessions in the TIA Portal.

Note

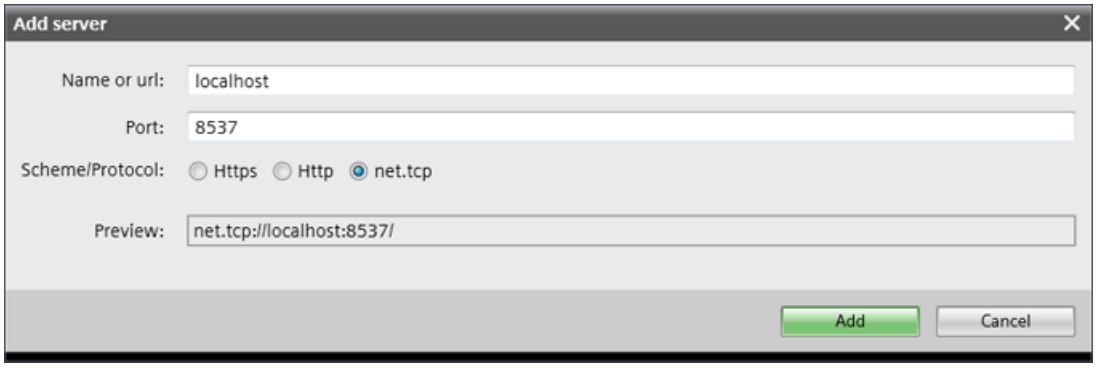
Multiuser projects and local sessions

If you have not yet created multiuser projects and local sessions on the multiuser server in the TIA Portal, no information regarding these will be displayed when you administer the multiuser server.

Adding a new server connection

Proceed as follows:

1. Open the graphic tool for the multiuser server administration.
   To open the start menu, click the command "Start" > "All Programs" > "Siemens Automation".
   Click on the "Multiuser Server V14 Configuration" entry to open the Configuration Tool.
2. Click "Add server connection".
3. In the dialog that follows, enter the desired connection data for your server connection as shown by way of example in the graphic:



Note

Connection information for the local multiuser server

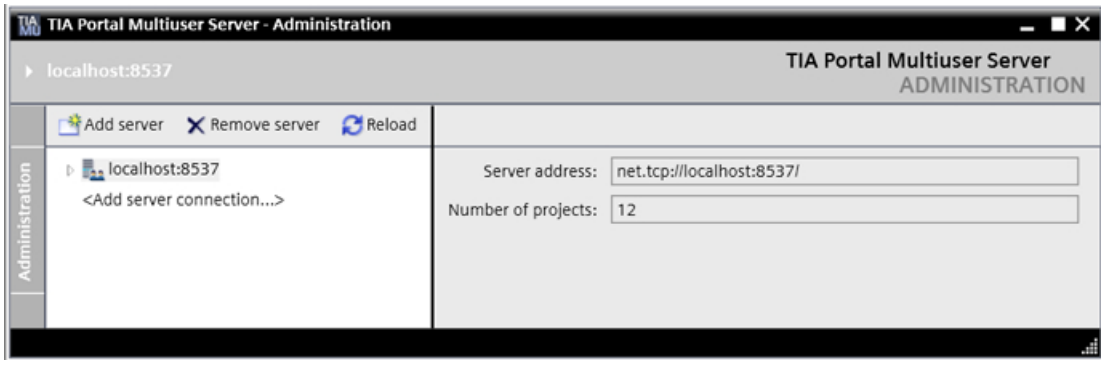
If you want to connect to the local multiuser server, you must start it first.

The local multiuser server is started automatically when you open a local session in the TIA Portal.

Alternatively, you can restart the local multiuser server by double-clicking on the file "Siemens.Automation.Portal.LocalServer.exe" under "bin" in the installation directory of the TIA Portal.

You can see the connection information for the local multiuser server after opening the displayed command window.

4. Click on "Add" to add the server connection. Servers in



5. Click on the triangle of the new server connection to activate the connection and view the project list.
6. If you use an "https" connection, you are prompted to confirm the validity of the displayed certificate.
   You have already specified the utilized connection protocol when configuring the multiuser server under "Connection settings".
   Compare the fingerprint of the displayed certificate with the certificate generated during configuration.
   Click "OK" when both fingerprints match.
7. If the multiuser server does not recognize the login data you used, you are prompted to enter alternative connection data for access to the server.
   Enter alternative login data that the multiuser server recognizes in the displayed "Sign in" dialog under "Sign in as another user".

Result

The new server connection is displayed in the left part of the Administration tool.

You see the number of existing multiuser projects on the right.

If you expand the new server connection in the left area, you see a list of all existing projects and other setting options.

Delete an existing server connection

Proceed as follows:

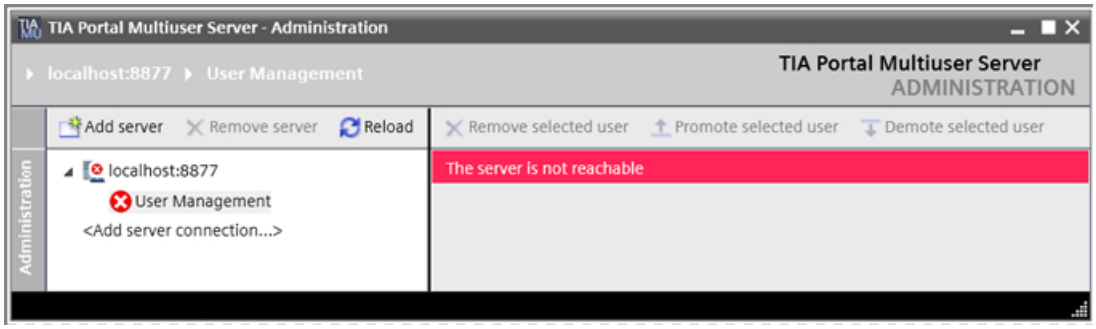
1. Select the desired server connection on the left side of the Administration tool.
2. Click on the "Remove server" button or select the command "Remove" from the shortcut menu.
3. Confirm the next prompt with "Yes".

Result

The selected server connection is deleted.

Accessibility of the multiuser server

If the multiuser server is not accessible, you receive the following error message:



Proceed as follows:

1. Check the displayed connection data.
2. Ensure that the selected server is started.
3. Delete the invalid server connection by clicking on it and then clicking "Remove server".
4. Confirm the next prompt with "Yes".
5. Click "Add server" and create a new server connection to a started multiuser server with valid connection data.

Result

The new server connection is displayed with valid connection information on the left.

Define "User Management" for the multiuser servers and projects

Introduction

With the graphical "Administration Tool", you specify the rights for working with multiuser server connections, multiuser projects and local sessions. The rights concept is based on the Windows access rights for folders and files.

You can find the "User Management" entry in the Adminstration Tool below each server connection as well as below each project and each local session. Depending on where you call the "User Management", you specify the corresponding authorizations.

You use the "User Management" to specify explicit personal authorizations for the following roles:

- Manager (full access)
- Contributor (partial access, no deleting)
- Member (read-only authorization)

The following functions can be executed for each role using the buttons:

- "Add new": Adds new users with the corresponding authorizations.
- "Remove selected user: Removes the selected user
- "Promote selected user": Moves the selected user up to the next higher authorization group.
- "Demote selected user": Moves the selected user down to the next lower authorization group.

The rights that are specified for the server connections are passed on to the authorizations for projects and local sessions.

Note

Administration Tool

- The user interface for the Administration tool is available in English only.
- You need administrator rights to work with the "User management".
- After changing the permissions, the project must be restarted.

Requirement

The multiuser server is installed.

The multiuser server has already been configured.

You have already created projects as multiuser projects and created local sessions in the TIA Portal.

Note

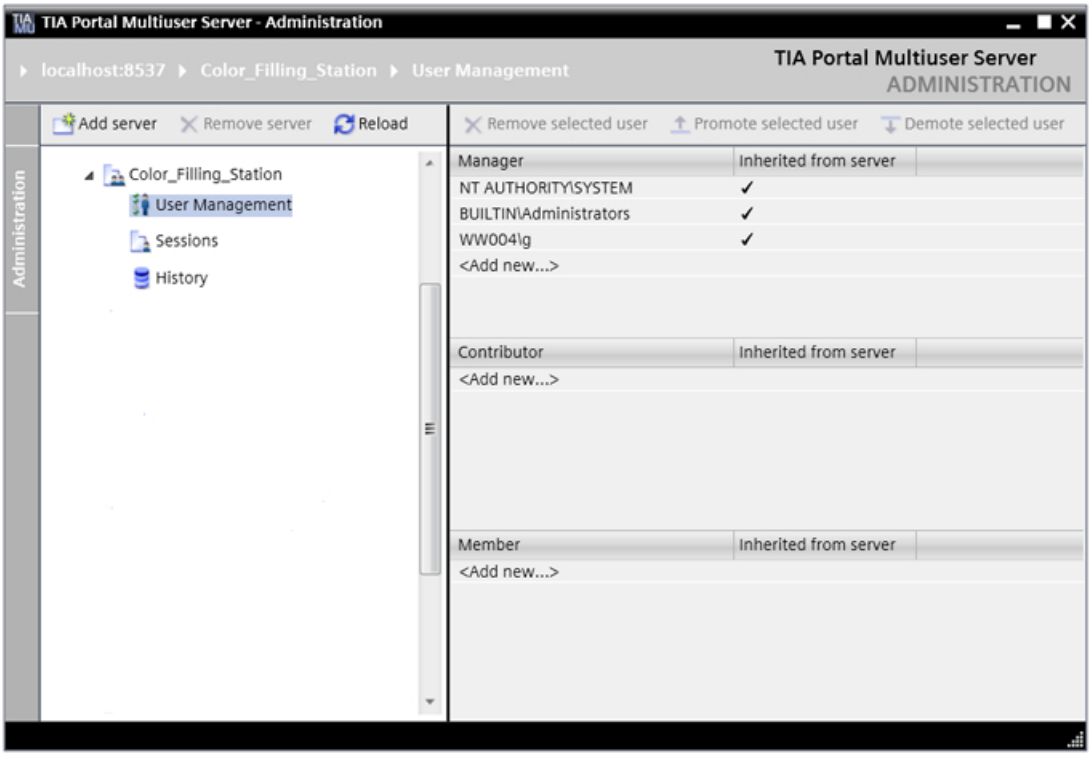
Multiuser projects and local sessions

If you have not yet created multiuser projects and local sessions on the multiuser server in the TIA Portal, no information regarding these will be displayed when you administer the multiuser server.

Assigning access rights

Proceed as follows:

1. Open the graphic tool for the multiuser server administration.
   - To open the start menu, click the command "Start" > "All Programs" > "Siemens Automation".
   - Click on the "Multiuser Server V14 Administration" to open the Administration Tool.
2. Double-click "User Management" below the desired entry on the left side.
   - When you click on "User Management" below a server, you assign the rights for working with the server.
   - When you double-click "User Management" below a project, you assign the rights for working with the project.
3. "User Management" opens and displays the possible settings on the right side. You already see which rights have been assigned for the associated server connection. These rights are inherited for the projects.



4. Click on "Add new" to add new users for the displayed roles.
5. Enter the required data in the subsequent Windows dialog and click "OK".
6. You can use the "Demote selected user" and "Promote selected user" buttons to move users up and down to other permission groups.
7. You can use the "Remove selected user" button to remove the selected user again.
8. Assign the desired rights and roles via "User Management" for both the server as well as for the lower-level projects and sessions.

Managing multiuser projects and local sessions

Introduction

You can manage the sessions created for a project with the graphical "Administration Tool".

Under "Sessions", you can see the local sessions for the selected multiuser project.

You can find the "Sessions" entry in the "Administration Tool" below each project.

Note
Administration Tool

The user interface for the Administration tool is available in English only.

Requirement

The multiuser server is installed.

The multiuser server has already been configured.

You have already created projects as multiuser projects and created local sessions in the TIA Portal.

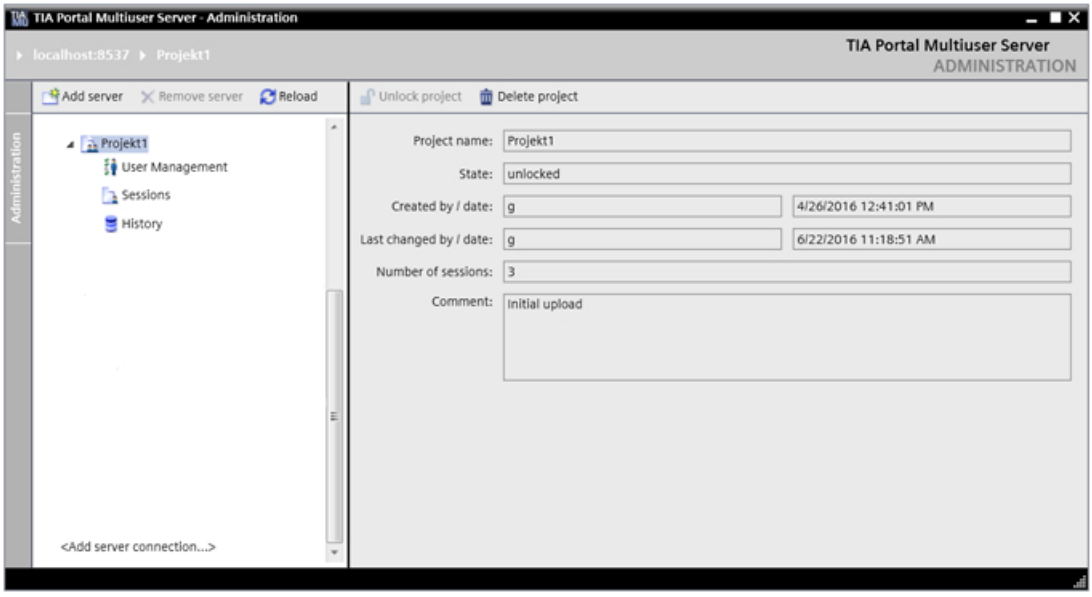
Note
Multiuser projects and local sessions

If you have not yet created multiuser projects and local sessions on the multiuser server in the TIA Portal, no information regarding these will be displayed when you administer the multiuser server.

Display project details and manage projects

Proceed as follows:

1. Open the graphic tool for the multiuser server administration.
   - To open the start menu, click the command "Start" > "All Programs" > "Siemens Automation".
   - Click on the "Multiuser Server V14 Administration" to open the Administration Tool.
2. Click on the desired project so that the project details are displayed on the right side.
   The following information about the projects is displayed:
   "Project name": Displays the project name
   "State": Displays the project status ("locked" or "unlocked").
   "Created by / date": Editor who created the project and the creation date for the project.
   "Last changed by / date": Editor who last amended the project and the change date for the project.
   "Number of sessions": Number of local sessions that belong to the project.
   "Comment": Display of project comments, if available.

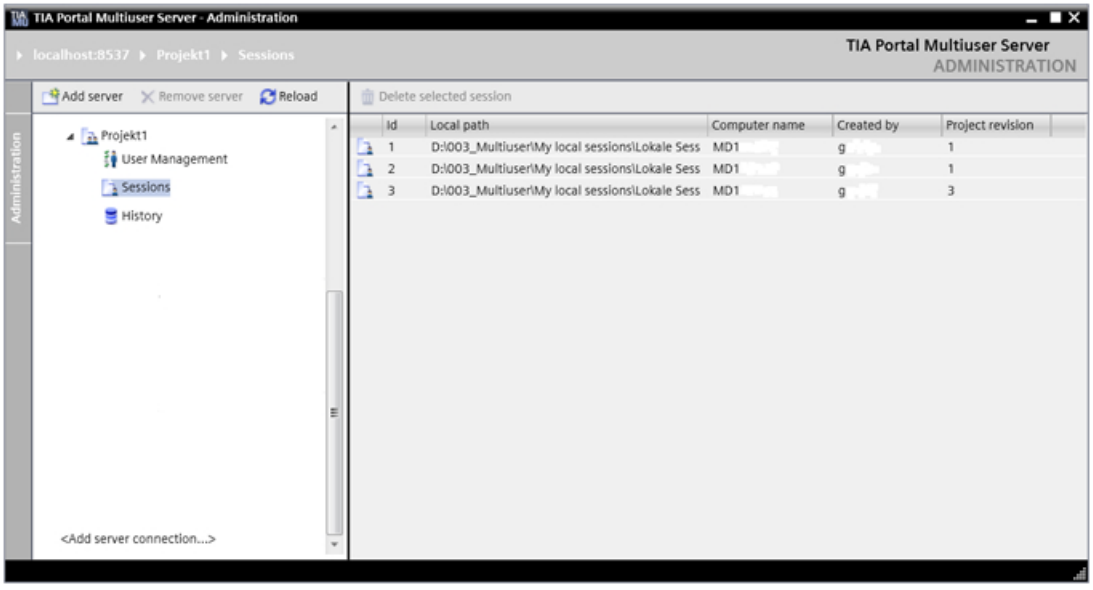


3. To delete a project, select the respective project and then click on the "Delete project" button. Projects can only be deleted if there are no local sessions for them.
4. Confirm the next prompt to delete the project.
5. To unlock a locked project, select the respective project and then click on the "Unlock project" button.
6. Confirm the next prompt to unlock the project.

Display and delete local sessions

Proceed as follows:

1. Open the graphic tool for the multiuser server administration.
   - To open the start menu, click the command "Start" > "All Programs" > "Siemens Automation".
   - Click on the "Multiuser Server V14 Administration" to open the Administration Tool.
2. Click on "Sessions" on the left below the desired multiuser project.
3. The following information on the existing sessions is displayed on the right in the "Administration Tool".
   "Id": Increments the existing sessions.
   "Local path": Displays the set local disk space for the sessions.
   "Computer name": Displays the name of the computer where the session is located.
   "Created by": Displays the name of the creator.
   "Project revision": Displays the project version to which each session belongs.



4. Select the desired session and click on the "Delete selected session" button to delete it.
5. In the following dialog, click "OK" to delete the selected session.

Note

Number of project revisions

You set the number of project revisions when configuring the multiuser server.

For more details, see: Configuring a multiuser server with the graphic tool (Page 6717)

See also

Configuring a multiuser server with the graphic tool (Page 6717)

Show "History" for projects

Introduction

You can view the history of the created projects with the graphical "Administration Tool".

You can also perform a "rollback" to a desired revision and export a project as "single-user project".

You can find the "History" entry in the "Administration Tool" below each project.

Note

Administration Tool

The user interface for the Administration tool is available in English only.

Requirement

The multiuser server is installed.

The multiuser server has already been configured.

You have already created projects as multiuser projects and created local sessions in the TIA Portal.

Note

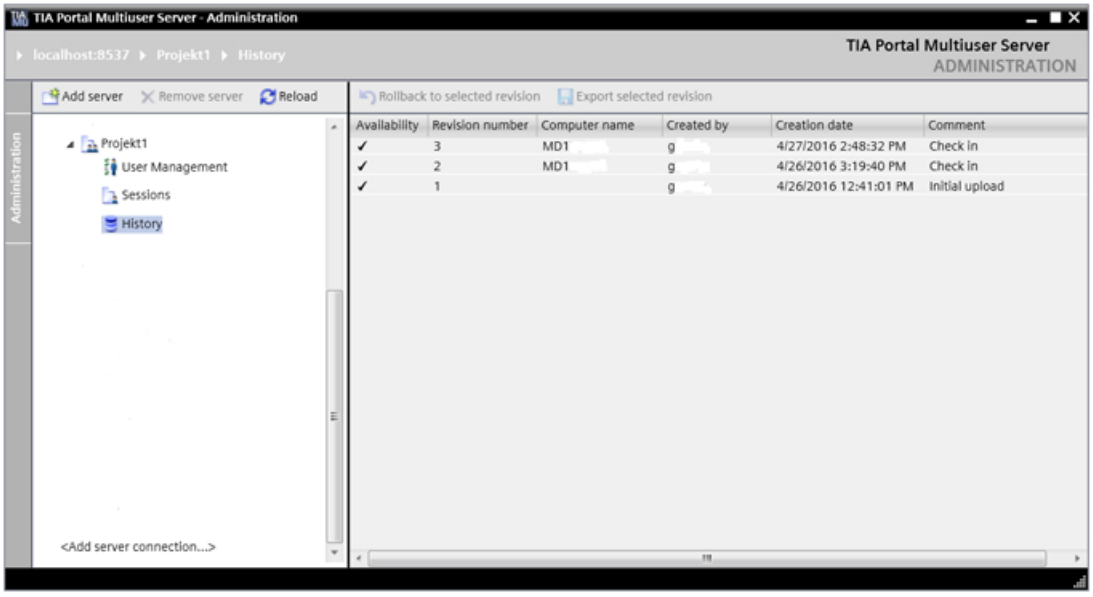
Multiuser projects and local sessions

If you have not yet created multiuser projects and local sessions on the multiuser server in the TIA Portal, no information regarding these will be displayed when you administer the multiuser server.

Show "History"

Proceed as follows:

1. Open the graphic tool for the multiuser server administration.
   - To open the start menu, click the command "Start" > "All Programs" > "Siemens Automation".
   - Click on the "Multiuser Server V14 Administration" to open the Administration Tool.
2. Click on "History" on the left below the desired multiuser project.
   The history displays all existing project revisions.
   The following information on the existing sessions is displayed on the right in the "Administration Tool".
   "Availability": Indicates whether the session on the server is still available.
   "Revision number": Displays the revision number.
   "Computer name": Displays the name of the computer on which the revision was created.
   "Created by": Displays the name of the creator.
   "Creation date": Displays the creation date of the revision.
   "Comment": Displays the comment for the revision, if any.



3. You can use the "Rollback to selected revision" button to roll back a selected session to the selected revision.
4. A new line is then created showing the new revision with the comment "Rollback to revision <number>".
5. To export the revision as a single-user project, click on the "Export selected revision" button.
6. Select the desired storage path in the subsequent dialog and click "OK".

15.3.10.3 Using the command line tools

Introduction to the multiuser command line tools

Introduction

The multiuser server can be configured and managed with the help of the Multiuser Administrative Tools and the Multiuser Power Tools.

Administrator rights are required for the configuration of the multiuser server with the Administrative Tools.

User rights are required for administrative tasks with the Power Tools.

The Multiuser Power Tools are also available in the TIA Portal for administrative tasks on the multiuser server.

Requirement

The multiuser server is installed.

Steps for the configuration of the multiuser server with the Administrative Tools

The server is installed and managed using a command line command.

To configure the multiuser server, the following steps are required:

- Call the multiuser server Administrative Tools with administrator rights.
- The complete installation is executed with the command:
  `musrv.exe install -r d:\TIA\SRV -p 5080 -b https`
- The project directory (repository) is created automatically.
- The definition of the protocol and the port address is already contained in this command, together with the integrated certificate administration for secure HTTPS communication.
- Windows tools are used to create new certificates and select existing certificates.
- You can find additional commands for configuring the multiuser server here: Commands for configuring the multiuser server (Page 6738)

Note

Enabling the firewall port

The specified port is automatically enabled in the firewall for access to the multiuser server.

Managing the multiuser server with the power tools

To manage the multiuser server, the following steps are required after installation of the server:

- To access the multiuser server, a port may have to be enabled in the firewall.
- To manage the multiuser server, the multiuser management or the multiuser server power tools must be called.
- You can find additional commands for managing the multiuser server here: Commands for managing the multiuser server (Page 6741)

User and rights management

The user and rights management for the multiuser server is based on the user management for the Windows user accounts.

- Project-specific rights assignment is possible.
- The use of group guidelines enables simple, fast and clear assignment of rights.

Project management

Project management enables a fast overview of existing projects and associated local sessions.

Multiuser projects and local servers can be conveniently managed.

The project management can be automated by scripts (batch files).

Error messages

If an action cannot be executed successfully, a corresponding error message is displayed.

You can find the error descriptions in the corresponding tools by calling the help for the tools:

Calling help in the administrative tools:

- Open the Administrative Tools as follows:
- Copy the command "musrv" to the command line and press "Enter".

Calling help in the power tools:

- Open the Power Tools as follows:
- Copy the command "mupt" to the command line and press "Enter".

See also

Commands for configuring the multiuser server (Page 6738)

Commands for managing the multiuser server (Page 6741)

Configuring the multiuser server with the administrative tools

Introduction

Before you can use the installed external multiuser server as part of Multiuser Engineering , the server must be configured.

When you configure the multiuser server using the Multiuser Administrative Tools , you use the commands described below, which you execute in the command line.

The following actions are automatically executed when you input the "Install" command described below:

- The Windows service for the multiuser service is installed.
- A firewall activation for the specified port is added via a TCP connection.
- Performance monitoring installed to monitor the multiuser server.
- The storage directory, called "Repository", is created if it does not already exist.
- The "binding" is defined, whether you use "https" (recommended) or "http" as transfer protocol.
- The "fingerprint" to confirm a valid certificate is transferred to the server.

The multiuser server can then be used.

Note

You need administrator rights to configure the multiuser server.

Requirement

The multiuser server is installed.

Note

The commands as well as the associated help and information texts within the command line are only available in English.

Procedure

To configure the multiuser server, follow these steps:

1. Open the tool for the multiuser server configuration with administrator rights. Under "bin" in the installation directory of the TIA Portal, double-click on the file "start-musrv.bat" to do this.
2. Copy the following command for server configuration with the required parameters to the command line and press "Enter".
   Command:
   ```
   => musrv install -r <repository-path> -p <port> -b <binding> -h
   <certificate-hash>
   => musrv install
   ```
   Example of the entry:
   ```
   musrv install -r c:\TIA\SERVER -p 5080 -b https -h <certificate-
   hash>
   ```
3. If the command "musrv install" could not be executed successfully, you receive an error message for which you can view an explanation.
   Once these commands have been executed successfully, the multiuser server is configured.

Configuration example

If you do not enter any parameters, the system will prompt you interactively for the required parameters.

Command	Description
Define "Repository-path" Used in the above example: `-r c:\TIA\SERVER`	Enter the name and the path of the directory (Repository) in which the server is to save the files. If you enter a new directory that does not exist yet, the directory is automatically created by the system. **Important note:** This directory and the data storage is retained even after deleting the server. If you reinstall the multiuser server after deleting it and enter an existing directory as "Repository", the data is also available on the new server without any restrictions.
Define "Port" Used in the above example: `-p 5080`	Enter the port you want to use for communication with the multiuser server.
Define "Binding" Used in the above example: `-b https`	Define whether you want to use "https" (recommended) or "http" as transfer protocol. With "https", communication is encrypted and a certificate is requested. This means communication has a higher security level than the unencrypted "http" communication.

Command	Description
"Certificate hash" = fingerprint for the certificate used	For secure server communication, enter the fingerprint for the certificate you created or the certificate used. For more information on creating a certificate, refer to the Help for the respective Windows operating system.
"Run service" = start service Example: `musrv start`	Use it to start the Windows service for the multiuser server.

You can find additional commands for configuring the multiuser server here: Commands for configuring the multiuser server (Page 6738)

See also

Commands for configuring the multiuser server (Page 6738)

Commands for configuring the multiuser server

Introduction

The commands for configuring the multiuser server are available in the Multiuser Administrative Tools .

Observe the basic procedures for configuring the multiuser server: Configuring the multiuser server with the administrative tools (Page 6736)

Note

The commands as well as the associated help and information texts within the command line are only available in English.

Commands for configuring the multiuser server

The commands listed below are available:

Command	Description	Possible entries
"Install"	Installs the multiuser server and enables communication with the server. A configuration example is available at: Configuring the multiuser server with the administrative tools (Page 6736)	**Parameters:** • -r, --repository: Directory in which the server data is to be stored. • -p, --port: Port number for the server access • -b, --binding: Specifies the protocol for the communication (http or https) • -c, --certhash: Fingerprint of the certificate which is to be used for secure data transfer (is only required for https). **Example**: `musrv install -r <repository-path> -p <port> -b <binding> -h <certificate-hash>`
"Config"	Displays the current configuration of the multiuser server and offers optional dynamic configuration parameters.	**Parameters:** • -t, --timeout: (optional): Definition of the utilized timeout profile ("fast", "middle" or "slow"). • -d, --debug: (optional): Defines whether runtime debug information is displayed by the server. Note: If you use the option "-d", no debug logs are displayed. **Example:** `musrv config -d`

"Start/Stop"	Starts or stops the multiuser server.	**Parameters:**
		• [-f], [--force]: (optional): Activates the command without further query
		Example:
		`musrv start`
		`musrv stop`

"Uninstall"	Uninstalls the multiuser server. Uninstallation can also take place with Windows mechanisms.	**Parameters:** • - [-f], [--force]: (optional): Activates the command without further query **Example:** `musrv uninstall`

See also

Configuring the multiuser server with the administrative tools (Page 6736)

Managing the multiuser server with the power tools

Introduction

To manage server projects and local sessions on the multiuser server as part of Team Engineering , use the commands described below that can be executed with the help of the Power Tools in the command line.

The multiuser server can be managed via the Power Tools from any computer on which TIA Portal and Power Tools are installed.

Your entries are stored in a "Session memory" and are then available the next time the Power Tools are opened.

Note

You need user rights to manage the multiuser server.

Requirement

The multiuser server is installed and a port is enabled in the Windows firewall so that the multiuser server is accessible.

Procedure

To manage the multiuser server, follow these steps:

1. Open the Power Tool for the multiuser server administration with administrator rights. Under "bin" in the installation directory of the TIA Portal, double-click on the file "start-mupt.bat" to do this.
2. Copy the command "mupt" to the command line and press "Enter".
   The help for the multiuser power tools is displayed.
3. Enter the required command for server management in the command line and press "Enter".
   The available commands are described under: Commands for managing the multiuser server (Page 6741)
4. If the command could not be executed successfully, you receive an error message.

See also

Commands for managing the multiuser server (Page 6741)

Commands for managing the multiuser server

Introduction

The commands for managing the multiuser server are available in the Multiuser Power Tools .

Note

The commands as well as the associated help and information texts within the command line are only available in English.

"User names" and "Role names"

"User names":

All "User names" that are used must take the following format into account:

- <domain-or-machine-name>\<account>, e.g. ww004\adgrfl0 or MD17E4MC\adgrfl0.
- If the <domain-or-machine-name> is not assigned, Windows will automatically propose a corresponding domain for the account.

"Role names":

The following "Role names" with the corresponding access rights are available:

- Member (only read access)
- Contributor (read & write access)
- Manager (read and write access and role management).

Commands for managing the multiuser server

The commands listed below are available:

Command	Description	Possible entries
"Display project"	Lists all projects on the current multiuser server.	**Parameters:** • -s, --server: Multiuser server URI address **Example:** `%InstallDirectory% > mupt project show -s http://192.168.0.100:8057/` **Help:** • "show-projects -?"
"Delete project"	Deletes specific projects on the multiuser server	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Definition of the project to be deleted • -f --force: (optional): Activates the command without further query **Example:** `%InstallDirectory% > project remove -s http://192.168.0.100:8057/ -p MyProject` **Help:** • "delete-project -?"
"Display Sessions"	Lists all local sessions that are part of the current server project.	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Selection of the project **Example:** `%InstallDirectory% > mupt session show -s http://192.168.0.100:8057/ -p MyProject` **Help:** • "show-sessions -?"
"Delete Session"	Deletes the selected local session on the multiuser server	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Selection of the project • -i, --id: Session identification number • -f, --force: (optional): Activates the command without further query **Example:** `%InstallDirectory% > mupt session remove -s http://192.168.0.100:8057/ -p MyProject -i 15` **Help:** • "delete-session -?"

Command	Description	Possible entries
"Remove project lock"	Removes the lock for the multiuser project	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Selection of the project • -f, --force: (optional): Activates the command without further query **Example:** `%InstallDirectory% > mupt project unlock -s http://192.168.0.100:8057/ -p MyProject` **Help:** • "remove-projectlock -?"
"Show check-in history"	Displays the check-in history for the selected server and the selected local session.	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Selection of the project • -i, --id: Session identification number • -u, --user: User name by which to filter. • -d, --date: Date by which to filter. • -r, --revision: Project revision by which to filter. • -f, --force: (optional): Activates the command without further query **Example:** `%InstallDirectory% > mupt checkin show -s http://192.168.0.100:8057/ -p MyProject` **Help:** • "show-checkinhistory -?"
"Show server access rights"	Displays the defined user rights for the selected server.	**Parameters:** • -s, --server: Multiuser server URI address **Example**: `%InstallDirectory% > mupt user show -s http://192.168.0.100:8057/` **Help:** • "show-serveraccessrights -?"
"Modify server access rights"	Modifies the access rights for the selected server.	**Parameters:** • -s, --server: Multiuser server URI address • -r, --role: Selecting the role • -u, --user: User name by which to filter. • -f, --force: (optional): Activates the command without further query **Example:** `%InstallDirectory% > mupt user add -u user1 -r Manager -s http://192.168.0.100:8057/` **Help:** • "modify-serveraccessrights -?"

Command	Description	Possible entries
"Remove server access rights"	Removes the access rights for the selected server.	**Parameters:** • -s, --server: Multiuser server URI address • -u, --user: User name by which to filter. • -f, --force: (optional): Activates the command without further query **Example:** `%InstallDirectory% > mupt user remove -u user1 -s http://192.168.0.100:8057/` **Help:** • "modify-serveraccessrights -?"
"Show project access rights"	Displays the defined user rights for the selected project.	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Project name for which the user rights are to be listed. **Example**: `%InstallDirectory% > mupt user show -s http://192.169.0.100:8057/ -p MYProject` **Help:** • "show-projectaccessrights -?"
"Modify project access rights"	Modifies the user rights for the selected project.	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Project name for which the user rights are to be listed. • -r, --role: Selecting the role • -u, --user: User name by which to filter. • -f, --force: (optional): Activates the command without further query **Example**: `%InstallDirectory% > mupt user add -u user2, user3 -r Member -p MyProject -s http://192.168.0.100:8057/` **Help:** • "modify-projectaccessrights -?"

Command	Description	Possible entries
"Remove project access rights"	Removes the access rights of the selected user for the selected project.	**Parameters:** • -s, --server: Multiuser server URI address • -u, --user: User name by which to filter. • -p, --project: Project name for which the user rights are to be listed. • -f, --force: (optional): Activates the command without further query **Example**: `%InstallDirectory% > mupt user remove -u user3 -p MyProject -s http://192.168.0.100:8057/` **Help:** • "remove-serveraccessrights -?"
"Export project to single user"	Exports the multiuser project as single-user project to the specified path.	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Project name for which the user rights are to be listed. • -t, --path: Selection of the storage path **Example**: `%InstallDirectory% > mupt project export -s http://192.168.0.100:8057/ -p MyProject -t D:\MyDirectory\` **Help:** • "export-projectassingleuser -?"

Command to reset the server to an older revision

The command listed below is available. This command can only be executed if the server is not locked.

"Project rollback"	Resets the server to an existing older revision.	**Parameters:** • -s, --server: Multiuser server URI address • -p, --project: Project name for which the user rights are to be listed. • -v, --revision: Revision number of the project to which it is to be reset. • -f, --force: (optional): Activates the command without further query **Example**: `%InstallDirectory% > mupt project rollback -s http://192.168.0.100:8057/ -p MyProject -v 132`

15.3.10.4 Confirm the certificate for a multiuser server

Introduction

When working with Multiuser Engineering , you must first specify the SSL Server certificate in order to establish a secure connection to an installed multiuser server. As soon as you start working in a local session for the first time after installation of the server, you are prompted to confirm the displayed certificate.

When an existing certificate has expired, you are prompted to confirm it again.

The mechanisms provided by Windows are used for managing the SSL server certificates.

If you do not confirm the SSL server certificate, a connection to the multiuser server cannot be established.

Requirement

You have installed TIA Portal V14 or higher and the required multiuser server software.

The multiuser server must be configured and started with the help of the multiuser tools.

You are working in the local session for the first time.

Confirm the certificate for a multiuser server

To confirm the certificate for the multiuser server, follow these steps:

1. When you are working with a connected multiuser server for the first time, for example, when opening a local session, the operating system automatically prompts you to confirm the server certificate for the multiuser server.
2. Check whether the displayed certificate is trustworthy. If so, click "OK" in the dialog that appears to confirm the certificate for the "TIA_Multiuser_Server".

Result

The dialog is closed. The certificate for the multiuser server is considered trustworthy and saved permanently.

15.3.10.5 Enter access data for the multiuser server

Introduction

When you are working with Multiuser Engineering and have installed a multiuser server, you must log on to the multiuser server if an authentication is required.

In this case the multiuser server prompts you to log on. A dialog is displayed in which you can enter the access data.

You have two options for logging on:

- Logon with your current Windows-user name und password
- Logon with the access data of a different user

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

The multiuser server must be configured and started with the help of the Power Tools.

Enter access data for the multiuser server

To enter the user data, follow these steps:

1. Select the required option in the dialog by clicking it.
   – For logon with your current login, the user name is already entered and cannot be edited.
   – Alternatively, you can log on with the access data of a different user.
2. To do so, select the second option and enter the respective data for "User name" and "Password".
3. Click "OK".

Result

If authentication was successful, the dialog is closed and the connection to the multiuser server is being established.

15.3.10.6 Transferring server data to other multiuser servers

Introduction

If you work with Multiuser Engineering and have installed a multiuser server, you can transfer existing server data to another multiuser server without data loss.

The data can be transferred to the same computer or to a another computer.

Requirement

You have installed TIA Portal as of V14 and at least one multiuser server.

The multiuser server must be configured and started.

Transfer multiuser server data to other multiuser servers

To transfer the server data of a multiuser server to another multiuser server, follow these steps:

1. Stop the active multiuser server.
   - Open the graphic ConfigurationTool for the multiuser server configuration with administrator rights.
   - Open the start menu by clicking the command "Start" > "All Programs" > "Siemens Automation".
   - Click on the "Multiuser Server V14 Configuration" entry. This will open the user interface of the Configuration Tool.
   - In the "Server status" area, click on the "Stop service" button.
2. Uninstall the "old" multiuser server.
   For more details, see: Installing and uninstalling the Multiuser server (Page 6711)
3. Copy the entire repository with the existing server data to the new storage location, for example, to another computer.
4. Install the multiuser server on the new computer.
   For more details, see: Installing and uninstalling the Multiuser server (Page 6711)
5. Configure the new storage directory for the new multiuser server.
   For more details, see: Configuring a multiuser server with the graphic tool (Page 6717)
6. Start the new multiuser server. In the Configuration Tool, click on "Start service" in the "Server status" area.

Result

The server data were transferred without data loss to the new multiuser server.

See also

Configuring a multiuser server with the graphic tool (Page 6717)

Installing and uninstalling the Multiuser server (Page 6711)

15.3.10.7 Reading out error log for multiuser server

Introduction

To check the multiuser server functionality, you can read out the error logs. When server errors occur, the error logs provide you with additional information on the error messages displayed in the user interface.

A separate log file is created by the server for each day. The log files are not deleted automatically.

Reading out the error log of a multiuser server

To read out the error log, follow these steps:

1. Open the directory for the respective multiuser server in your archiving.
2. Select the desired log file under the path <Repository>\logs.
3. Open the requested log file to check the content

Reading out error logs for a multiuser project

To read out the error log, follow these steps:

1. Open the directory for the respective multiuser server in your archiving.
2. Select the desired log file under the path <Repository>\proj\<Project name>\logs.
3. Open the requested log file to check the content.

Note

Reading out the error log

Keep in mind that the error log cannot be read out when you are working in debug mode.

Check and/or change the server configuration in the command line via the administrative tools using the command "musrv config".

Make sure that the option "-d" is not selected.

Reading out Windows event view

The event view is a tool with which detailed information about important events can be displayed on the computer. The event view can be helpful for dealing with problems and errors.

The following security relevant events relating to the multiuser server also receive an entry in the Windows event view:

- If a user wants to the start the multiuser server and the multiuser server does not "know" this user; i.e. the user is not logged in on the server.
- If a user with an incorrect password want to log onto the multiuser server.

To read out the event view, follow these steps:

1. Click "Start" > "Control Panel" > "System and Security" > "Administration". Alternatively, click "Start" and enter "Event view" in the search field.
2. Double-click to open the Event view.
3. Under "Windows protocols" click "Security" to display the logged security relevant events related to the multiuser server.

Result

The logged security relevant events for the multiuser server are shown in a list.

15.3.10.8 Error log for multiuser server packed as Zip file

Introduction

To check the multiuser server functionality, you can read out the error logs. When server errors occur, the error logs provide you with additional information on the error messages displayed in the user interface.

You have the option of packing all existing error logs into a Zip file to forward these, if necessary, to Customer Support for example.

To do so you must log yourself in as Administrator directly on the server. It is not possible to execute this action via the TIA Portal or via the administrative and power tools.

Packing error logs as Zip file

To create a Zip file for error logs, follow these steps:

1. Log onto the multiuser server as administrator.
2. Execute the command "musrv ziplogs".
3. Alternatively, you can execute the command with the following options:
   - -n, --name: Entering the storage path for the Zip file
   - -f, --force: Overwrites any already existing files without prompt.

Result

The Zip file with all existing error logs for the multiuser server is created.

15.3.10.9 Information on response of the multiuser server to errors

Introduction

The response of the multiuser server after the occurrence of an error is described below.

If an error occurs, you receive a corresponding error message.

Behavior of the server after the occurrence of an error

The multiuser server exhibits the following behavior after the occurrence of an error:

Project lock:

- Information on which project is locked may get lost after a server error.
- This information is no longer available after a restart.

Errors that occur during "Upload from device" or "Download to device":

- The refresh of the local session cannot be performed. The action is aborted with an error message.
- The check-in cannot be executed. The action is aborted with an error message.
- Any already loaded data are removed again from the server, the server has the same status as before the start of the faulty action.

Note

Manual restart of the server

The server has to be manually restarted after the occurrence of an error.

An automatic start of the server only occurs after the next restart of the computer on which the server is running.

15.3.10.10 Supported network configurations for the multiuser server

Requirements for successful client-server communication

Introduction

Below is an overview of the requirements for successful client-server communication in the context of Multiuser Engineering with:

- Network communication
- Certificate management
- Authentication

Requirements for network communication

The network used must allow HTTPS communication between client and server.

There are two ways of doing this:

- Direct TCP communication between client and multiuser server via the enabled server port.
  - No outbound firewall settings blocks communication on the workstations.
  - No incoming firewall settings blocks communication on the server.
  - Both the multiuser server and all workstations are in the same IP network (without routing).
  - If a router is installed in between, the router should not block communication via the selected server port.
- Indirect communication via an HTTPS proxy server.
  - An HTTPS proxy server is available.
  - The HTTPS proxy server is configured at the client end in the Windows system and enabled for communication.

Requirements for certificate management

To establish secure communication, you should use "HTTPS". With "HTTPS", communication is encrypted and a certificate is requested. This means communication has a higher security level than the unencrypted "http" communication.

To establish HTTPS communication, the multiuser server must be configured in such a way that it uses a valid HTTPS certificate for data encryption.

- The certificate in the Windows certificate store is used on the multiuser server.
- The certificate must be currently valid and registered for use with "http.sys".
- The fingerprint of the certificate must be handed over to the client.

Requirements for authentication

Windows mechanisms are used for the authentication of the communication with the multiuser server.

Multiuser Engineering supports both authentication through domain accounts and authentication through local Windows accounts.

- Authentication with domain account
  - Both the client and the multiuser server are members of the same Windows domain.
  - A valid user account for this domain is used.
  - Both the client and the multiuser server can establish network communication to the domain server.
  - The requirements for using a secure https communication must be met.

- Authentication with local account
  - A local account on the multiuser server must be used for authentication.
  - The requirements for using a secure HTTPS communication must be met.

Note

NTLM authentication on the multiuser server

NTLM is used for authentication on the multiuser server. For security reasons, we recommend you activate NTLM V2, which has been used as standard since Windows 2000, and not NTLM V1.

NTLM V2 offers a logon method more secure than NTLM V1.

Note

In the following sections, you can find more information about supported network configurations for the multiuser server.

See also

Network with dedicated multiuser server (Page 6753)

Network with temporary multiuser server (Page 6755)

Network in a Windows domain (Page 6757)

Network in a Windows domain with Internet access via VPN (Page 6759)

Network in a Windows domain with Internet access via proxy (Page 6762)

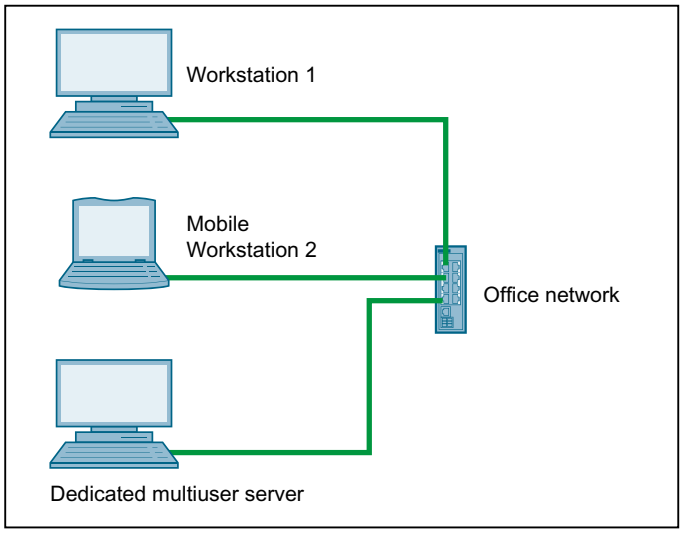
Network with dedicated multiuser server

Introduction

Below, you can find information on a server configuration with a dedicated multiuser server in an office network.

The graph shows the possible structure of such a configuration:

Office network with dedicated multiuser server



Description of the network configuration with a dedicated multiuser server

The network configuration should meet the following requirements.

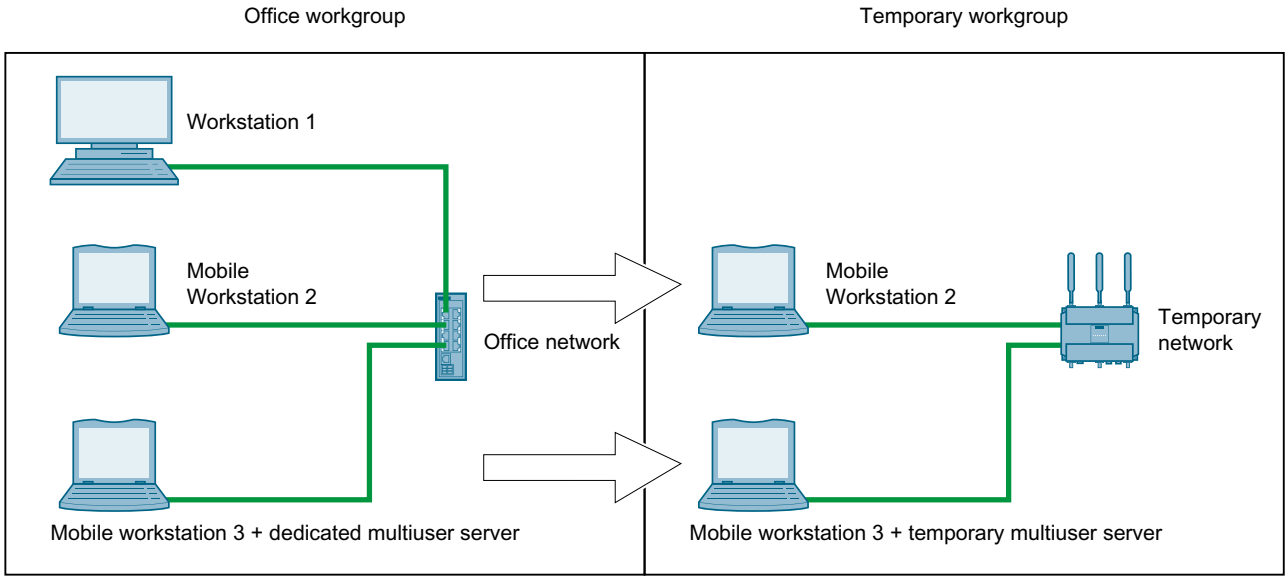
Configuration	Description
Network environment	**Configuration:** • Various Windows workstations on which the TIA Portal is installed. • A dedicated Windows server, which is installed as a multiuser server. • All workstations and servers are located in the same network.
Network	**Requirements:** • There is a direct TCP connection between the workstations and the multiuser server via the selected port. • No outbound firewall settings blocks communication on the workstations. • No incoming firewall settings blocks communication on the server. • Both the multiuser server and all workstations are in the same IP network (without routing). • If a router is installed in between, the router should not block communication via the selected server port. • Either no proxy server is configured, or there is a local bypass. The "Bypass proxy server for local addresses" option should be selected in the Windows settings for a local area network.
Authentication	**User accounts and Windows workgroups:** • These must be created locally on the multiuser server. • This must be configured in accordance with the multiuser rights and roles for servers and projects. • Name and password for authentication are queried on all workstations. • Only local accounts of the server system can be used for authentication. **Note**: The Windows authentication can be used to query name and password. You have to create local accounts, which are identical to the accounts that exist on the multiuser server. This procedure allows the server to automatically authenticate a local account with identical logon information for an existing server account. **Restriction:** User accounts and groups cannot be managed centrally; local users and groups are required for this. This configuration is therefore unsuitable for enterprise networks.

Network with temporary multiuser server

Introduction

Below, you can find information on a server configuration with a temporary multiuser server in an office network.

The graph shows the possible structure of such a configuration:


Office workgroup
Temporary workgroup
Workstation 1
Mobile
Workstation 2
Office network
Mobile
Workstation 2
Temporary
network
Mobile workstation 3 + dedicated multiuser server
Mobile workstation 3 + temporary multiuser server

Description of the network configuration with a temporary multiuser server

The network configuration should meet the following requirements.

Configuration	Description
Network environment	Configuration: • Various Windows workstations on which the TIA Portal is installed. • A temporary Windows server, which is installed as a multiuser server at one of the workstations. The office environment may be left, for example, to perform commissioning in a factory. • All workstations and temporary multiuser servers are located in the same network. • With the temporary multiuser server, the office environment can be left and work can still continue.
Network	Requirements: • There is a direct TCP connection between the workstations and the multiuser server via the selected port. • No outbound firewall settings blocks communication on the workstations. • No incoming firewall settings blocks communication on the server. • Both the multiuser server and all workstations are in the same IP network (without routing). • If a router is installed in between, the router should not block communication via the selected server port. • Either no proxy server is configured, or there is a local bypass. The "Bypass proxy server for local addresses" option should be selected in the Windows settings for a local area network.
Authentication	User accounts and Windows workgroups: • These must be created locally on the multiuser server. • This must be configured in accordance with the multiuser rights and roles for servers and projects. • Name and password for authentication are queried on all workstations. • Only local accounts of the server system can be used for authentication. Note: The Windows authentication can be used to query name and password. You have to create local accounts, which are identical to the accounts that exist on the multiuser server. This procedure allows the server to automatically authenticate a local account with identical logon information for an existing server account. Restriction: User accounts and groups cannot be managed centrally; local groups are required for this. This configuration is therefore unsuitable for enterprise networks.

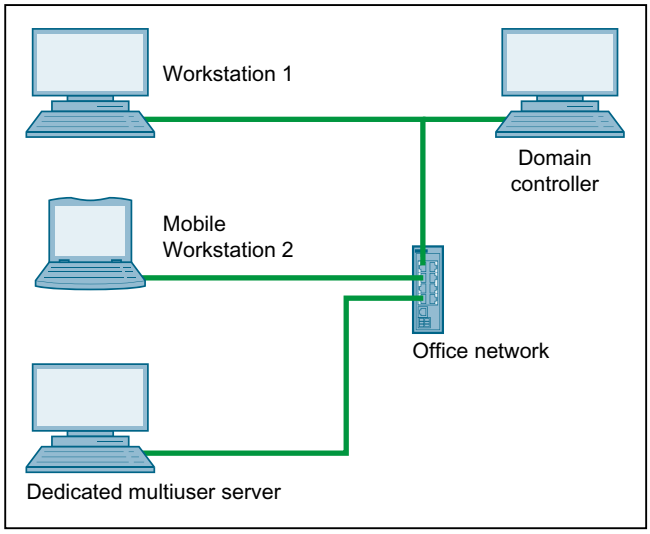
Network in a Windows domain

Introduction

Below, you can find information on a server configuration with a dedicated multiuser server in a Windows domain.

The graph shows the possible structure of such a configuration:

Network within a Windows domain



Description of the network configuration in a Windows domain

The network configuration should meet the following requirements.

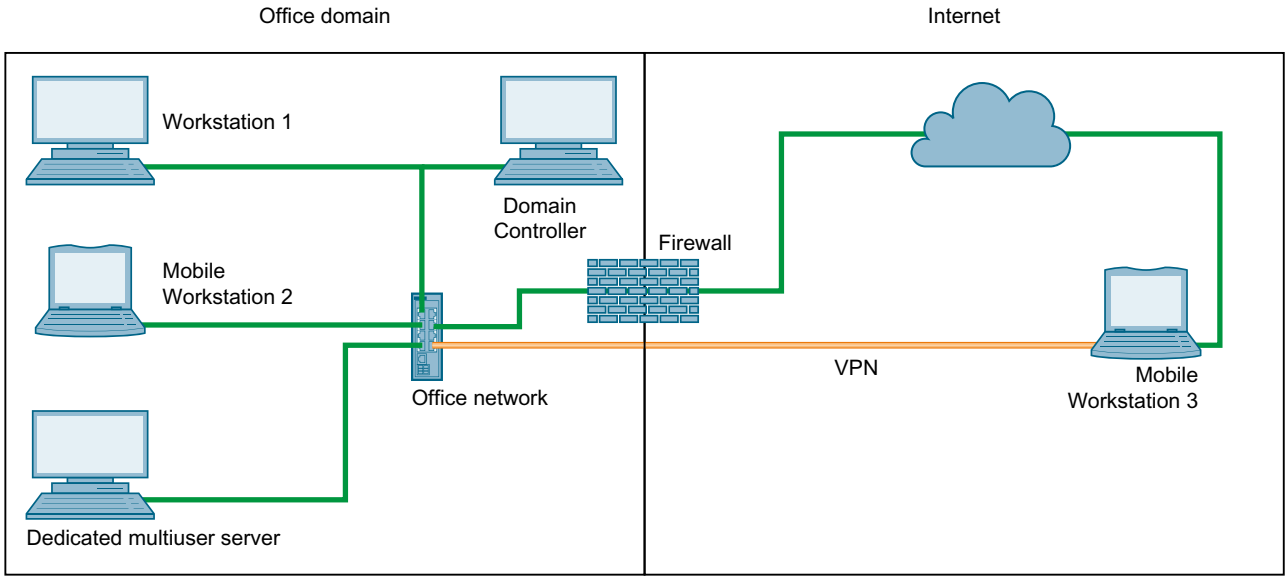
Configuration	Description
Network environment	**Configuration:** • All Windows workstations on which the TIA Portal is installed are members of the same domain. • A dedicated Windows server as a multiuser server is also located in the same domain. • All workstations and multiuser servers use the same LAN (Local Area Network). • All workstations and multiuser servers can connect to the domain controller.
Network	**Requirements:** • There is a direct TCP connection between the workstations and the multiuser server via the selected port. • No outbound firewall settings blocks communication on the workstations. • No incoming firewall settings blocks communication on the server. • Both the multiuser server and all workstations are in the same IP network (without routing). • If a router is installed in between, the router should not block communication via the selected server port. • Either no proxy server is configured, or there is a local bypass. The "Bypass proxy server for local addresses" option should be selected in the Windows settings for a local area network.
Authentication	**User accounts and Windows workgroups:** • These must be created on the domain controller. • This must be configured in accordance with the multiuser rights and roles for servers and projects. • Users at workstations on which the TIA Portal is running are automatically authenticated with their domain logon information. **Note**: You can also log on with other valid logon data on the multiuser server. **Restriction:** This configuration is not suitable for temporary server configurations, since they also need access to the domain controller.

Network in a Windows domain with Internet access via VPN

Introduction

Below, you can find information on a server configuration with a dedicated multiuser server in a Windows domain with VPN (Virtual Private Network).

The graph shows the possible structure of such a configuration:


Office domain
Internet
Workstation 1
Domain Controller
Firewall
Mobile Workstation 2
Office network
VPN
Mobile Workstation 3
Dedicated multiuser server

Description of the network configuration in a Windows domain with VPN

The network configuration should meet the following requirements.

Configuration	Description
Network environment	**Configuration:** • All Windows workstations on which the TIA Portal is installed are members of the same domain. • A dedicated Windows server as a multiuser server is also located in the same domain. • All workstations and multiuser servers use the same LAN (Local Area Network). • Both workstations and the multiuser server have access to the domain controller. • Workstations can also be connected to the LAN via VPN.
Network	**Requirements:** • There is a direct TCP connection via the selected port between the workstations and the multiuser server. • No outbound firewall settings blocks communication on the workstations. • No incoming firewall settings blocks communication on the server. • Both the multiuser server and all workstations are in the same IP network (without routing). • If a router is installed in between, the router should not block communication via the selected server port. • Either no proxy server is configured, or there is a local bypass. The "Bypass proxy server for local addresses" option should be selected in the Windows settings for a local area network. • For security reasons, we recommend using VPN solutions, such as "Routing and Remote Access Service". • For security reasons, it is strongly recommended to set up a firewall. It is not necessary to open a port in the firewall for multiuser server when using VPN.

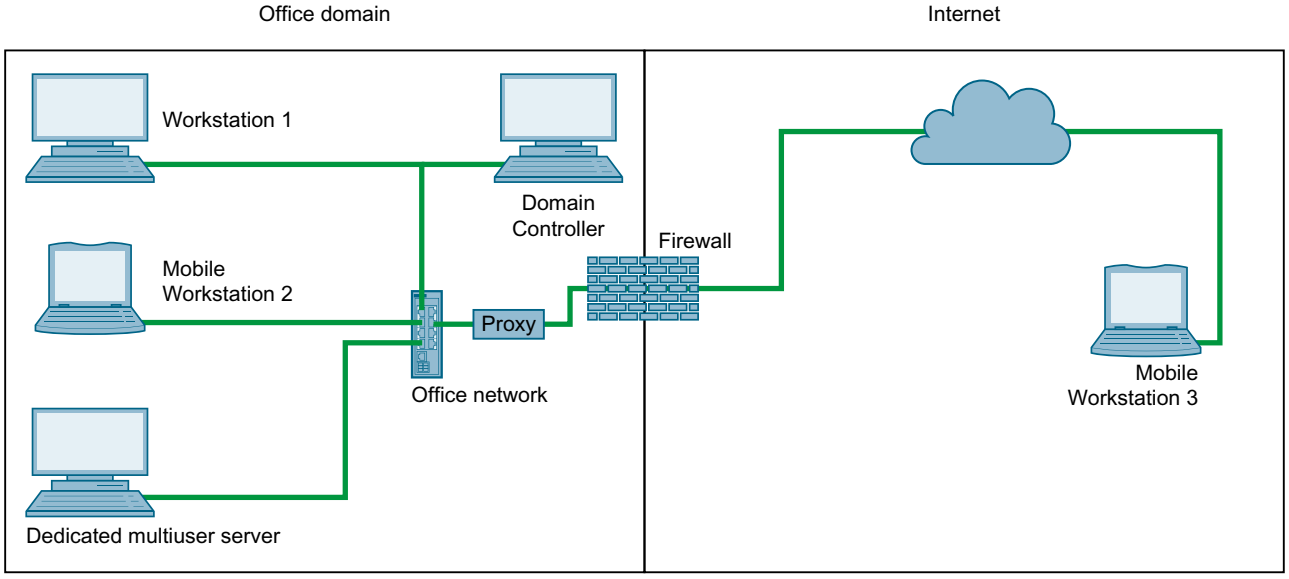
Configuration	Description
Authentication	Authentication is performed based on the current configuration selected. **User accounts and Windows workgroups when configured as an office workgroup:** • These must be created locally on the multiuser server. • This must be configured in accordance with the multiuser rights and roles for servers and projects. • Name and password for authentication are queried on all workstations. • Only local accounts of the server system can be used for authentication. **Note**: The Windows authentication can be used to query name and password. You have to create local accounts, which are identical to the accounts that exist on the multiuser server. This procedure allows the server to automatically authenticate a local account with identical logon information for an existing server account. **Restriction:** User accounts and groups cannot be managed centrally; local users and groups are required for this. This configuration is therefore unsuitable for enterprise networks. **User accounts and Windows workgroups with configuration as Windows domain:** • These must be created on the domain controller. • This must be configured in accordance with the multiuser rights and roles for servers and projects. • Users at workstations on which the TIA Portal is running are automatically authenticated with their domain logon information. **Note**: You can also log on with other valid logon data on the multiuser server. **Restriction:** This configuration is not suitable for temporary server configurations, since they also need access to the domain controller. Communication with a remote computer through a VPN demonstrates typical performance limitations for all client-server applications. The multiuser server is also affected.

Network in a Windows domain with Internet access via proxy

Introduction

Below, you can find information on a server configuration with a dedicated multiuser server in a Windows domain with intranet/Internet access via a proxy.

The graph shows the possible structure of such a configuration:


Office domain
Internet
Workstation 1
Domain
Controller
Firewall
Mobile
Workstation 2
Proxy
Office network
Mobile
Workstation 3
Dedicated multiuser server

Description of the network configuration in a Windows domain with proxy

The network configuration should meet the following requirements.

Network environment	**Configuration:** • All Windows workstations on which the TIA Portal is installed are members of the same domain. • A dedicated Windows server as a multiuser server is also located in the same domain. • All workstations and multiuser servers use the same LAN (Local Area Network). • Both workstations and the multiuser server have access to the domain controller. • Workstations outside the LAN can also be connected to the LAN via a proxy server.
Network	**Requirements:** • There is a direct TCP connection between the workstations and the multiuser server via the selected port. • No outbound firewall settings blocks communication on the workstations. • No incoming firewall settings blocks communication on the server. • Both the multiuser server and all workstations are in the same IP network (without routing). • If a router is installed in between, the router should not block communication via the selected server port. • Either no proxy server is configured, or there is a local bypass. The "Bypass proxy server for local addresses" option should be selected in the Windows settings for a local area network. • For security reasons, it is not recommended to make the multiuser server available on the Internet. • For security reasons, it is strongly recommended to set up a firewall to accept the incoming connections to the selected port. • The proxy server must accept HTTPS communication and forward it to the multiuser server on the selected port.

Authentication	Authentication is performed based on the current configuration selected. **User accounts and Windows workgroups when configured as an office workgroup:** • These must be created locally on the multiuser server. • This must be configured in accordance with the multiuser rights and roles for servers and projects. • Name and password for authentication are queried on all workstations. • Only local accounts of the server system can be used for authentication. **Note**: The Windows authentication can be used to query name and password. You have to create local accounts, which are identical to the accounts that exist on the multiuser server. This procedure allows the server to automatically authenticate a local account with identical logon information for an existing server account. **Restriction:** User accounts and groups cannot be managed centrally; local users and groups are required for this. This configuration is therefore unsuitable for enterprise networks. **User accounts and Windows workgroups with configuration as Windows domain:** • These must be created on the domain controller. • This must be configured in accordance with the multiuser rights and roles for servers and projects. • Users at workstations on which the TIA Portal is running are automatically authenticated with their domain logon information. **Note**: You can also log on with other valid logon data on the multiuser server. **Restriction:** This configuration is not suitable for temporary server configurations, since they also need access to the domain controller. Communication with a remote computer through a proxy demonstrates typical performance limitations for all client-server applications. The multiuser server is also affected.
Port	**Note**: Sometimes it is only possible to use port "443" for an https connection on a proxy server. A port other than "443" for an https connection cannot be set on some proxy servers. In this case, you need to install the multiuser server on port "443" as well.

15.3.11 Working with the multiuser server

15.3.11.1 Introduction to the multiuser server

Introduction

To be able to work with the full functionality of Multiuser Engineering , you need a multiuser server, which manages your multiuser projects.

The multiuser server must be installed first and then configured and started with the help of the multiuser tools.

The connections to the multiuser server in the TIA Portal can then be configured and administered using the "Options" > "Settings" > "Multiuser" command.

In the TIA Portal, you can execute the following actions for the server connections:

- Add new multiuser server connections.
- Create, edit, and delete existing multiuser server connections

Properties of the multiuser server

The multiuser server has the following properties:

- You need administrator rights to install and configure the multiuser server.
- The installation of the server requires settings in the Windows firewall and may require a certificate.
- The multiuser server is configured and administered by an administrator with the supplied multiuser tools.
- The server is available to all editors of the multiuser project and it can also be accessed from the outside.
- The multiuser server can be started and administered outside the TIA Portal.
- The "Alias" assigned as the name for the multiuser server is used as an identifier and may not be changed at a later point in time.

Quantity structure for the multiuser server

For efficient work with Multiuser Engineering, you can create up to 100 multiuser server connections.

When you have reached this number, you receive a message informing you that you cannot create any new server connections. Once you have deleted any server connections that are no longer required, you can once again create new connections up to the maximum number.

The following configuration limits are available for a multiuser server depending on the respective hardware:

Number of projects on the multiuser server	**Standard multiuser server (application as a temporary multi-user server)** **Hardware:** **Field PG M5 Advanced (or equivalent)** **Intel® Core™ i5-6440EQ up to 3.4 GHz·,16 GB RAM (32 GB for large projects)** **SSD with 50 GB of available disk space** **Network 1 GB** **15.6" Full HD Display (1920 x 1080)**
Maximum number of server projects per server:	100
Maximum number of local sessions per server:	200
Maximum number of local sessions per server project:	25

Note

Server hardware

When more powerful server hardware is used, even higher quantities are possible.

See also

Displaying the multiuser server connection (Page 6769)

Editing the multiuser server connection (Page 6770)

Deleting the multiuser server connection (Page 6772)

15.3.11.2 Setting the multiuser network profiles

Introduction

When working with Multiuser Engineering you have the option of selecting between three different network profiles, depending on the performance capability of your own network.

The selected network profile determines the time until a timeout message is generated.

Available multiuser network profiles

The following network profiles are available for working with the multiuser server:

Network profile	Area of application
"Fast"	• Office environment • Gigabit/Ethernet • Stable and mostly interference-free network environment
"Medium"	• Office environment, small workgroup with fixed location • Fast Ethernet, WLAN, VPN connections • Good network environment, rare interference
"Slow"	• Commissioning, non-stationary workstations • WLAN connection • Interference possible (machines, electrical installations)

Setting the multiuser network profiles

To set the network profiles in the TIA Portal, follow these steps:

1. In the TIA Portal, select the command "Options" > "Settings" > "Multiuser".
   The available options for the network profiles are displayed in the following dialog.
2. From the drop-down list for "Multiuser network profiles", select the desired profile.

Result

The selected connection profile is activated.

Note

The timeout behavior of the current network profile is not affected at the changeover.

15.3.11.3 Adding a new multiuser server connection

Introduction

If you work with Multiuser Engineering and have installed a multiuser server, you can add up to 100 server connections to the multiuser server in the TIA Portal.

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

The multiuser server must be configured and started with the help of the power tools.

Adding a new multiuser server connection

To add a new multiuser server connection, follow these steps:

1. In the TIA Portal, select the command "Options > Settings > Multiuser".
   The "Multiuser server connections" tab is open and all existing server connections are displayed.
   If no multiuser server connection has been configured yet, only the entry for the "Local multiuser server" is available.
2. In the following editor, click the "Add server connection" command in the "Server name" column.
   If the maximum number of 100 server connections has already been reached, you receive a message informing you that you first have to delete one of the existing server connections before you can add a new one.
   If it is possible to add a new server connection, the "Add new Multiuser server connection" dialog box opens.
3. Enter the name for the multiuser server in the "Server name" field.
   You can use a maximum of 256 characters for the name and the path. The name can include letters, numbers and commas.
   You will be notified if the entered name is already being used or if the path is longer than 256 characters in total.
4. Select the required transmission protocol (https or http) for the multiuser server. The "http" protocol is selected by default.
5. Enter the required host and port for the server connection.
   You will be notified if the entered host and/or port is already in use.
6. Click "Add".

Result

The new multiuser server connection is added and displayed under "Options > Settings > Multiuser".

15.3.11.4 Displaying the multiuser server connection

Introduction

When you are working with Multiuser Engineering you can display the existing server connections to the multiuser server in the settings of the TIA Portal.

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

The multiuser server must be configured and started with the help of the multiuser tools.

Displaying the multiuser server connection

To display the existing multiuser server connections, follow these steps:

1. In the TIA Portal, select the command "Options > Settings > Multiuser".
   The existing multiuser server connections are displayed in the "Server name" column in the next editor.
2. If you are looking for a specific server connection, click the "Server name" or "Host" column headers to sort the content.

Result

The existing multiuser servers are displayed in the editor under "Options > Settings > Multiuser"

15.3.11.5 Editing the multiuser server connection

Introduction

When you are working with Multiuser Engineering you can edit the existing server connections to the multiuser server in the settings of the TIA Portal.

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

The multiuser server must be configured and started with the help of the multiuser tools.

Editing the multiuser server connection

1. In the TIA Portal, select the command "Options > Settings > Multiuser".
   All existing multiuser server connections are listed in the next dialog in the "Server name" column.

   Note

   It is not permitted to edit a multiuser server connection that is used in a currently open local session.

2. If necessary, click the "Server name" or "Host" column headers to sort the contents.
3. Double-click the server connection that you wish to edit.
   Result: The "Edit multiuser server connections" dialog box opens.
4. You can enter the desired changes here.
   - The "Server name" field cannot be edited.
   - The entry in the "Host" field can be edited.
     You can navigate to the required host by clicking the button with the dots.
   - The entry in the "Port" field can be edited.

5. If possible, select "https" as the transmission protocol to ensure secure data exchange with encryption and certificate.
6. Click "OK" for the changes to become effective.

Result

Your changes for the selected multiuser server connection are applied and displayed under "Options > Settings > Multiuser".

15.3.11.6 Restore multiuser server connection

Introduction

When you are working with Multiuser Engineering the existing server connection to the multiuser server can be interrupted.

If this is the case, you will receive a corresponding error message after about 30 seconds.

You have different options, depending on whether you are working in the local session or in the server project view.

You have the following options when you are working in the local session:

- Restore the server connection
- Save and close the local session
- Close the local session without saving

You have the following options when you are working in the server project view:

- Restore the server connection
- Discard changes

Possible procedure in the local session

To restore a multiuser server connection, follow these steps:

1. Click "Restore" in the displayed dialog.
   If the server connection can be restored, the dialog closes and you can continue working without restrictions.
2. Click "Save and close the local session", if you are not continuing to work at the moment but want to save your changes.
3. Click "Close local session", if you want to exit the local session without storing it.

Possible procedure in the server project view

To restore a multiuser server connection, follow these steps:

1. Click "Restore" in the displayed dialog.
   If the server connection can be restored, the dialog closes and you can continue working without restrictions.
2. Click "Discard changes", if you exit the server project view without saving it and want to discard any changes you may have made.
   The server project view closes. The associated local session does not open, because there is currently no connection to the multiuser server.

15.3.11.7 Deleting the multiuser server connection

Introduction

When you are working with Multiuser Engineering you can delete the existing server connections to the multiuser server in the settings of the TIA Portal.

You can create, edit and delete up to 100 multiuser server connections in the TIA Portal.

Note

Keep in mind that you can only delete a server connection if all existing local sessions for this server connection were deleted beforehand.

Keep in mind that a connection from a local session to a deleted multiuser server connection is no longer possible.

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

The multiuser server must be configured and started with the help of the multiuser tools.

Deleting the multiuser server connection

To delete an existing multiuser server connection, follow these steps:

1. In the TIA Portal, select the command "Options > Settings > Multiuser".
   All existing multiuser server connections are listed in the "Server name" column in the next editor.
2. If necessary, click the "Server name" or "Host" column headers to sort the contents.
3. Select one or more server connections that you wish to delete and click "Delete" in the shortcut menu.
   The "Delete" dialog box opens.
4. Click "Yes" to confirm the next message and to delete the selected server connection permanently.

Result

The selected multiuser server connection is deleted and is no longer displayed under "Options > Settings > Multiuser".

15.3.12 Working with the local multiuser server

15.3.12.1 Introduction to the local multiuser server

Introduction

To work with Multiuser Engineering , you need a multiuser server that manages your multiuser projects.

A local multiuser server is always installed automatically together with the installation of the TIA Portal and cannot be deleted. It does not require a separate software installation.

The local multiuser server is displayed in the list of Multiuser servers as the first entry under the non-editable "Local Multiuser Server" name and can be called with the " Options" > "Settings" > "Multiuser" command.

Advantages of the local multiuser server

The local multiuser server offers the following advantages:

- You can try out Multiuser Engineering without having to install a dedicated multiuser server first.
- You can buffer your work progress on the local multiuser server.

Requirement

To use the functionality of the local multiuser server, TIA Portal V14 or higher must be installed.

Properties of the local multiuser server

Unlike a dedicated multiuser server, the local multiuser server offers only limited functionality.

The local multiuser server has the following properties:

- It is only available to one user on their computer and cannot be deleted.
- It can only be started within the TIA Portal.
- It can only be called from the computer on which the local session is running that is assigned to this local multiuser server.
- It does not support management tasks.
- Its settings do not have to be adapted for the firewall and it does not need certificates.
- Only the port can be edited; the server name and host cannot be changed.

See also

Configuring the local multiuser server (Page 6774)
Displaying the local multiuser server (Page 6775)
Starting the local multiuser server (Page 6775)
Exiting the local multiuser server (Page 6777)

15.3.12.2 Configuring the local multiuser server

Introduction

A local multiuser server is installed by default during installation of the TIA Portal by default.

It is displayed in the list of multiuser servers as the first entry and can be called with the command " "Options > Settings > Multiuser server".

Requirement

You have installed TIA Portal V14 or higher.

Configuring the local multiuser server

To configure the local multiuser server, follow these steps:

1. In the TIA Portal, select the command "Options > Settings > Multiuser server".
   All existing multiuser servers are displayed in the next dialog in the "Server name" column.
2. If necessary, click the "Server name" or "Host" column headers to sort the contents.
3. Double-click the entry "Local multiuser server".
   The "Server connection" dialog opens.
4. Enter your changes.
   - The "Server name" and "Host" fields cannot be edited for the local multiuser server.
   - The entry in the "Port" field can be edited. Enter the required port here.
5. Click "OK" for the changes to become effective.
   Result: Your changes to the local multiuser server are applied and displayed under "Options > Settings > Multiuser server".

Note

It is not permitted to configure the local multiuser server while it is connected to an open local session.

If a local session has already been created and the local multiuser server is only reconfigured afterwards, the server must first be stopped and then restarted to prevent the existing local session from becoming invalid.

See also

Exiting the local multiuser server (Page 6777)

15.3.12.3 Displaying the local multiuser server

Introduction

A local multiuser server is installed by default during the installation of the TIA Portal.

Requirement

You have installed TIA Portal V14 or higher.

Displaying the multiuser server

To display the local multiuser server, follow these steps:

1. In the TIA Portal, select the command "Options > Settings > Multiuser server".
   In the next editor, the local multiuser server is displayed as the first entry in the "Server name" column.
   The dedicated multiuser servers you have created are listed below this.

Note

The local multiuser server cannot be deleted.

The local multiuser server is displayed with the uneditable server name "Local Multiuser Server" in all languages.

15.3.12.4 Starting the local multiuser server

Introduction

A local multiuser server is installed by default during the installation of the TIA Portal.

It is displayed in the list of multiuser servers as the first entry and can be called with the command " "Options > Settings > Multiuser server".

Starting the local multiuser server

To start the local multiuser server, follow these steps:

1. In the TIA Portal, select the command "Project > Manage multiuser projects".
   The "Manage multiuser projects" dialog box opens.
2. In the "Select server" box, select the local multiuser server from the drop-down list. It is displayed under the name "Local Multiuser Server".
   - Any existing server projects are listed here. The individual columns include detailed information on the respective project and on which user has made which changes and when. If required, you can change the arrangement of the columns.
   - You will be informed if a connection to the selected server is not possible.
   - If the local multiuser server has not yet been started, you will be asked if you want to start it now.
3. Confirm the next query as to whether the local multiuser server is to be started on your computer with "Yes".

Result

The connection to the local multiuser server is being established. A command window opens and shows that the server is now activated.

Note

Although the connection to the local multiuser server is terminated by closing the TIA Portal, this server is not stopped automatically.

To stop the local multiuser server, you must close the displayed command line.

Opening the local session assigned to the local multiuser server

As soon as you open a local session that is assigned to the local multiuser server and when the local multiuser server has not ben started, you are queried if you want to start the local multiuser server.

If you answer this query with "Yes", the local multiuser server is started and the local session is opened for editing.

You cancel the process with "No".

Note

Keep in mind that you can only delete a server connection if all existing local sessions for this server connection were deleted beforehand.

Keep in mind that a connection from an existing local session to a deleted multiuser server project is no longer possible.

15.3.12.5 Exiting the local multiuser server

Introduction

When you open a local session that is assigned to the local multiuser server , a prompt appears asking whether the local multiuser server is to be started. When the local session is ended or the TIA Portal is closed, the local multiuser server is not automatically ended.

The local multiuser server must be explicitly ended.

Requirement

The local multiuser server was started.

Procedure

To end the local multiuser server, follow these steps:

1. Terminate the local multiuser server by closing the displayed command line.

15.3.13 Creating and managing multiuser server projects

15.3.13.1 Introduction to working in multiuser server projects

Introduction

Once the multiuser server has been installed and then configured and started with the help of the multiuser tools , you can manage the multiuser server projects in the TIA Portal.

Managing the multiuser server projects

When you execute the command "Project > Manage multiuser server projects", all existing server projects for the existing server connections are displayed in the next dialog box.

The local multiuser server is displayed first in the list of existing server connections, because it always exists (by default).

If you have not added a multiuser server project yet, the list is empty.

If server projects already exist, they are displayed. From the drop-down list, select the required server connection for which all existing server projects are then listed.

You can execute the following actions with the individual buttons:

- Add and delete server connections.
- Add new server projects to the selected server connection.
- Delete existing server projects.

- Create new local sessions for a server project.
- Open existing local sessions.

Note
Displaying server projects

Server projects can only be displayed when a connection to the multiuser server exists and it has been started.

You will be notified if a connection could not be established.

Note
Deleting server projects

To prevent inconsistencies in the project management, server projects must not be deleted "manually" with the Explorer in the storage directory.

In the context of Multiuser Engineering, server projects must always be deleted with the Multiuser Tools.

All associated local sessions must be deleted before a server project can be deleted.

15.3.13.2 Adding a new multiuser server project

Introduction

When working with Multiuser Engineering, you must first create a multiuser server project. A multiuser server project is created by adding a standard project (single-user project) to the previously installed multiuser server and then be able to use this project as a multiuser server project.

Note
Multiuser server projects

Observe the requirements described under Requirements for multiuser projects (Page 6703), which a multiuser project has to fulfill.

Requirement

- The TIA Portal V14 is installed on your computer and executable.
- The multiuser server V14 is installed and can be accessed.

Procedure

To create a multiuser server project, follow these steps:

1. In the TIA Portal, click the menu command "Project > Manage multiuser server projects".
2. Select the required sever connection from the drop-down list in the next dialog box.
3. In the last row, click the "Add project to the server" button.
4. Set the required path to the single-user project or navigate to the required directory via the "..." button.
5. Select the project which you want to add as multiuser server project.
6. Enter the necessary information under "Details".
   The server project name is always preset and cannot be edited.
7. The "Create local session" button is enabled by default. Disable this button if you do not want to create a local session for the newly added multiuser server project.
8. Click "Add".
   If the selected project already exists as multiuser server project or is currently opened by another user, you will be notified.

Result

The selected single-user project is added as new multiuser server project and displayed under "Project > Manage multiuser server projects".

If the option "Create local multiuser session" was selected, a new local session is created and opened.

See also

Requirements for multiuser projects (Page 6703)

15.3.14 Working in the local session

15.3.14.1 Creating and managing a local session

Creating a local session

Introduction

To work with Multiuser Engineering , you must create a local session. The local session is always assigned to a specific multiuser server project.

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

You have created an executable multiuser server project.

Procedure

To create a local session, follow these steps:

1. In the TIA Portal, select the command "Project > Manage multiuser projects".
   The "Manage multiuser projects" dialog box opens.
2. Select the required multiuser server in the drop-down list.
3. Select the required project from the displayed list and click "Create new local session".
   The "Create local session" dialog box opens.
4. In the "Create local session" dialog box, check your selection regarding the multiuser server and the multiuser server project.
   The displayed selection cannot be modified in this dialog box.
   If the displayed selection is not want you wanted, click "Cancel" and start again with step 1.
5. Add the detailed information to the local session:
   - Enter a valid path for storage of the local session or use the corresponding button to navigate to the required location.
   - Assign a name for the directory in which the local session is to be saved. The directory is created implicitly if it does not exist yet.
   - If required, enter a comment for the local session.

     Note

     Keep in mind that the author for the local session was already entered by default in this dialog box and cannot be edited.

   - The "Open local session" button is selected by default and causes the new local session to open as soon as it has been created. If desired, deactivate this button.
6. Click the "Create" button to execute this action.
   A message indicates that the local session is being created.

Result

The newly created local session is opened and displayed.

See also

Opening a local session (Page 6781)

Saving a local session (Page 6783)

Closing a local session (Page 6784)

Deleting a local session (Page 6785)

Displaying all local sessions associated with a server project (Page 6786)

Exporting a local session as single-user project (Page 6787)

Opening a local session

Introduction

You have different options for opening a local session:

- In the TIA Portal with "Recently used projects"
- In the TIA Portal with the associated Multiuser server project
- With the Windows Explorer when the TIA Portal is closed

As soon as you open a local session that is assigned to the local multiuser server, a prompt appears asking if you want to start the local multiuser server, if it has not yet been started. Click "Yes" to start the local multiuser server and to open the selected local session.

Note

Instructions for opening local sessions:

- A renamed or deleted local session can no longer be opened.
- A local session cannot be opened when the associated server is not available.
- A local session cannot be opened as a reference project.

When opening a local session that meets one of the criteria listed above, you will see a note informing you that the session cannot be opened. In this case a prompt asks whether you want to export this local session as single-user project.

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

You have created an executable multiuser server project and a valid local session.

Opening a local session

To open a recently used local session, follow these steps:

1. In the TIA Portal, select the command "Project > Open".
   The "Open project" dialog box opens with a list of recently used projects and sessions. The icons in the first column help you distinguish between single-user projects and multiuser projects.
2. Select the preferred local session and click "Open".

Result

The selected local session is opened.

Opening a local session by means of the associated server project

To open a local session via the associated server project, follow these steps:

1. In the TIA Portal, select the command "Project > Manage multiuser projects".
   The "Manage multiuser projects" dialog box opens.
2. Select the required multiuser server in the drop-down list.
3. Select the desired local session from the displayed list and click the "Show local session" button.
   The "Open local session" dialog box opens and displays all existing local sessions for the selected server.
   The entry in the first row is selected by default.
4. Select the preferred local session and click "Open".

Result

All opened dialog boxes are closed and the selected local session is opened.

Opening a local session with the Windows Explorer

To open a local session with the Windows Explorer, follow these steps:

1. While the TIA Portal is closed, navigate to the directory in which your local sessions are stored in Windows Explorer.
2. Select the preferred local session with the file extension "*.als14" and double-click it.

Result

The TIA Portal is started and the selected local session is opened.

Note

You close a local session as you would a single-user project with the command "Project > Close".

Opening a local session from a different storage location

To open a local session that has been moved to another storage location in the meantime, follow these steps:

1. In the TIA Portal, select the command "Project > Open".
   The "Open project" dialog box opens with a list of recently used projects and sessions.
2. Select the preferred local session and click "Open".
   You are notified that you are trying to open a local session from a different storage location.
3. Confirm this query with "OK" if you want to execute the action.

Result

The selected local session is opened from the new storage location.

The information about the new storage location is saved in the associated server.

See also

Creating a local session (Page 6779)

Saving a local session (Page 6783)

Closing a local session (Page 6784)

Deleting a local session (Page 6785)

Displaying all local sessions associated with a server project (Page 6786)

Exporting a local session as single-user project (Page 6787)

Saving a local session

Introduction

You have different options for saving a local session:

- As "local session" with the file extension "*.als14"
- After an export as "single-user project" with the file extension "*.ap14".

When you export and save a local session as single-user project, you can continue to edit this project in the familiar way without multiuser functionality.

While working in the local session, you save this as usual as local session in order to continue to work in the context of Multiuser Engineering .

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

You have created an executable multiuser server project and a valid local session.

Procedure

To save a local session, follow these steps:

1. Open the local session:
2. Make your required changes.
3. In the TIA Portal, select the command "Project > Save".

Result

The local session is saved on the associated multiuser server with the file extension "*.als14".

See also

Creating a local session (Page 6779)

Opening a local session (Page 6781)

Closing a local session (Page 6784)

Deleting a local session (Page 6785)

Displaying all local sessions associated with a server project (Page 6786)

Exporting a local session as single-user project (Page 6787)

Closing a local session

Introduction

You close a local session as you would a single-user project with the command "Project > Close".

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

You have created an executable multiuser server project and a valid local session that is open.

Procedure

To close a local session, follow these steps:

1. In the TIA Portal, select the command "Project > Close".
2. Confirm the next prompt with "Yes" if you want to save your changes.

Result

The local session is closed.

See also

Creating a local session (Page 6779)

Opening a local session (Page 6781)

Saving a local session (Page 6783)

Displaying all local sessions associated with a server project (Page 6786)

Exporting a local session as single-user project (Page 6787)

Deleting a local session

Introduction

To avoid inconsistencies in multiuser projects, you **cannot** delete a local session in the usual way with the command "Project > Delete project...".

Local sessions can only be deleted in the context of the multiuser functionality and depending on the associated multiuser server project.

Note

If you have manually deleted a local session with the Windows Explorer from the storage directory, it is still displayed in the "Local sessions" dialog box but can no longer be opened.

Delete this local session again explicitly in the multiuser context as described in the procedure below.

Local sessions that existing in the multiuser server can also be deleted with the Multiuser Tools in the server management. For more details, see:Commands for managing the multiuser server (Page 6741)

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

You have created an executable multiuser server project and a valid local session.

The local session you wish to delete must not be open.

Procedure

To delete a local session in the TIA Portal, follow these steps:

1. In the TIA Portal, select the command "Project > Manage multiuser projects...".
   The "Manage multiuser projects" dialog box opens with a list of sessions available for the selected server.
2. Select the respective server from the drop-down list.
3. Select the relevant server project from the list and click the "Show local session" button.
4. In the next dialog box "Open local session", select the respective session and click the "Delete" button.

   Note

   A local session cannot be deleted if it is open; the delete command is inactive in this case.

   In this case, close the local session first and click the "Delete" button once again.

5. Confirm the next query regarding deletion with "Yes".

Result

The selected local session is deleted and it is also deleted from the list of existing local sessions.

See also

Creating a local session (Page 6779)

Opening a local session (Page 6781)

Saving a local session (Page 6783)

Closing a local session (Page 6784)

Displaying all local sessions associated with a server project (Page 6786)

Exporting a local session as single-user project (Page 6787)

Commands for managing the multiuser server (Page 6741)

Displaying all local sessions associated with a server project

Introduction

You have the option of displaying all local sessions associated with a multiuser server project in the TIA Portal.

Requirement

You have installed TIA Portal V14 or higher and a multiuser server.

You have created an executable multiuser server project and a valid local session.

Displaying all local sessions

To display all local sessions belonging to a server project, follow these steps:

1. In the TIA Portal, select the command "Project > Manage multiuser projects".
   The "Manage multiuser projects" dialog box opens and displays all existing server connections for the multiuser server.
2. Select the required server connection.
   All server projects associated with this server connection are listed.
3. Select the relevant server project and click the "Show local sessions" button.
4. The "Open local session" dialog box opens and displays all existing local sessions for the selected server project.
   The entry in the first row is selected by default.
   If you want to open a local session, select the respective session and click "Open".

Result

All existing local sessions for the selected server project are displayed.

See also

Creating a local session (Page 6779)

Opening a local session (Page 6781)

Saving a local session (Page 6783)

Closing a local session (Page 6784)

Deleting a local session (Page 6785)

Exporting a local session as single-user project (Page 6787)

Exporting a local session as single-user project

Introduction

When working with Multiuser Engineering, you have the option of exporting a local session as single-user project.

This is beneficial when a local session can currently not be opened, for example, because the multiuser server is not available.

However, a local session may also become invalid and can therefore no longer be opened.

A local session becomes invalid for the following reasons:

- When the local session was renamed.
- When the local session has been moved to a different storage location.
- When the settings for the associated server have been changed and the server is no longer available for this reason.

If a local session is detected to be invalid when opened, you will receive a query as to whether you want to export the session as a single-user project.

This means you can save the work results from the local session in a single-user project and still use them.

You can apply the changes from the exported local session into your multiuser server project once again by copying them.

You also have the option of exporting a local session with the multiuser Power Tools.

For more details, see: Commands for managing the multiuser server (Page 6741)

Exporting a local session as single-user project

To export a local session as single-user project, follow these steps:

1. Click the command "Project > Manage multiuser projects".
2. Select the respective server from the drop-down list in the displayed dialog box.
3. Select the relevant server project and click the "Show local sessions" button.
4. Select the local session and click "Export as single-user project" in the shortcut menu. In the next dialog box, the name of the local session and the file extension ".ap14" are already preset.

5. Under "Export to" enter the preferred storage location or navigate to the preferred storage location and click "Export".
   Alternatively, you will be notified by the system that the local session is invalid when you open it.
6. Confirm the subsequent query whether you want to export the invalid local session as single-user project with "Yes".
   The "Export as single-user project" dialog box opens. The name of the local session and the name of the export file are preset and cannot be edited.
7. Enter the preferred export path or navigate to the preferred storage location and click "Export".

Result

The local session is exported as single-user project and saved under the set storage path.

The export dialog box is closed and the server project list is displayed once again.

See also

Creating a local session (Page 6779)

Opening a local session (Page 6781)

Saving a local session (Page 6783)

Closing a local session (Page 6784)

Deleting a local session (Page 6785)

Displaying all local sessions associated with a server project (Page 6786)

Commands for managing the multiuser server (Page 6741)

15.3.14.2 Editing objects in the local session

Introduction to working in local sessions

Introduction

Within the context of Multiuser Engineering , to work simultaneously with multiple users in a multiuser project, each editor must create their own local session.

The respective editor can make changes in one or more local sessions, check them in to the multiuser server project and thus publish them.

After check-in, the changes from the local session are once again available to all editors in the server project. They can be reused and modified once again in subsequent local sessions.

Procedure when working in local sessions

The following steps are necessary to work as multiuser in the engineering tasks that have been assigned to you in a local session:

- You have installed TIA Portal V14 or higher and a multiuser server.
- Alternatively, you can use the local multiuser server installed by default.
- You have created an executable project for working with Multiuser Engineering.
- You have loaded the prepared project to the multiuser server and thus created a multiuser project.
- You have coordinated the task and the configuration limits for working in the local sessions with the other editors.
- You have created a local session for each engineering task.

Rules for working in local sessions

The following rules apply to working in local sessions:

- A local session is always assigned to a defined multiuser server and a defined multiuser server project.
- Each local session includes different tasks for different editors who are assigned to a multiuser server projects.
- A local session receives the file extension "*.als14" when it is saved and must not be renamed in Windows Explorer.
- While a local session is open, the associated multiuser server connection and the associated multiuser server project must not be renamed, moved or deleted.
- Local sessions can no longer be opened and edited when the associated multiuser server or the associated multiuser server project have been deleted.
- A local session cannot be opened without a server connection.
- A local session cannot be opened as reference project.
- A local session can be exported as single-user project and edited further without multiuser functionality.
- A local session cannot be archived or retrieved. These actions are possible as soon as the local session is once again edited as single-user project.
- Working with external tools is not possible in the local session. Changes by external tools in local sessions are lost after the local session is checked in and refreshed. External tools that make changes to the file structure of a multiuser project may only be used in the server project view.
- Changing of user-defined documentation is not possible in the local session. Changes to existing user-defined documentation are lost after the local session is checked in and refreshed. Changing of user-defined documentation is only possible in the server project view.

Note

Renaming objects in local sessions

Objects should be renamed only in the server project view, since this does **not** cause any inconsistencies in the server project.

When renaming objects in the local session, you must ensure that all objects affected by the renaming and their places of use are also marked and checked in the server project. Otherwise, you get compiling errors and an inconsistent server project due to partial renaming.

Before checking the places of use of the renamed object, you should be careful not to create conflicts (red flags) when marking and you should not mark any outdated objects; otherwise, you will overwrite the work of another user when checking in.

Note

Editing library objects in local sessions

Library objects should be edited only in the server project view, since this does **not** cause any inconsistencies in the server project.

With version changes to library objects in the local session, for example, when you create new versions of type instance objects, you must ensure that all relevant objects have been marked with a version change and checked into the server project.

Otherwise, you get compiling errors and an inconsistent server project from creating new versions.

Note

You can find detailed instructions on working with local sessions in the sections below.

See also

Displaying usage info for marked objects (Page 6794)

Unmarking objects (Page 6793)

Check-in objects to the server project (Page 6797)

Possible conflicts during check-in (Page 6800)

Possible error messages during check-in (Page 6803)

Marking objects for editing

Introduction

For objects to be edited in Multiuser Engineering at the same time, each editor must mark the objects they want to edit in their local session.

You can mark objects within the local session as follows:

- With the shortcut menu command "Mark objects for check-in".
- By clicking the displayed flags in the project tree and in the individual editors of the TIA Portal.

In this version of the TIA Portal, edited objects are only marked automatically when objects are deleted.

For marking of objects, multiple selections are also possible via the shortcut menu. A prerequisite for the marking is that the selection includes at least one object that can be marked. If the multiple selection does not include an object that can be marked, the shortcut menu commands are possibly not active.

Rules for marking objects

The following rules apply to marking:

- The markings and the marking column in the project tree and in the individual editors are only displayed in the context of Multiuser Engineering.
- Objects that can be marked are identified by a gray flag in the project tree and in the editors.
- You can only mark objects that can be edited with Multiuser Engineering.
- If an object does **not** have a gray flag in front of it, this object cannot be marked and it should not be edited in the local session as the changes cannot be checked into the server project. Such objects that cannot be marked should be edited in the server project view.
- Some higher-level directories cannot be marked themselves, but the individual objects in the folder can be marked.
  Example: "PLC tag table": The PLC tag table itself cannot be marked, but the individual tags in the open tag table can be marked.
- You mark them by clicking the gray flag or with the shortcut menu. The flag is shown in color as soon as an object has been marked.
- The color of the flags provides additional information about the marking:

Icon	Meaning
	Object can be marked for check-in.
	Object is marked in own local session.
	Object is marked in a different local session that is part of the same multiuser server project.
	Conflict: The object was marked in multiple sessions simultaneously.

Note

Changes to objects that are not marked

Changes to objects that are not marked in the local session are possible but **cannot** be applied to the server project.

These changes are lost after update of the local session, because they are once again overwritten with the contents of the server project.

As soon as you want to edit an object that is not marked, a banner text is displayed in the open editor. This banner text contains information on the further procedure.

Procedure for marking objects

To mark objects in a local session, follow these steps:

1. Open the required local session and click on the displayed gray flag to mark an object for check-in.
2. Alternatively, you can select the required object and mark it with the shortcut menu command "Mark object for check-in".
3. Mark all objects in your local session that you currently want to edit. You can make multiple selections for marking. After marking, you can edit and save the objects as usual.

Note

Multiple selections

Once > = 50 objects are selected in the local session, both shortcut menu commands are active simultaneously: the command for marking the objects as well as the command for unmarking.

Result

The marked objects are displayed with a blue or red flag.

If a blue flag is displayed, the object can be edited in the local session.

If a red flag is displayed, you have a conflict that has to be resolved through coordination with the other editors.

Note

Markings with conflict (red flag)

A marking conflict must be resolved by agreement of the editors, and a sequence for editing the marked objects must be specified. To solve the conflict, one editor must unmark the objects.

See also

Introduction to working in local sessions (Page 6788)

Unmarking objects (Page 6793)

Displaying usage info for marked objects (Page 6794)

Unmarking objects

Introduction

For objects to be edited in Multiuser Engineering at the same time, each editor must mark the objects they want to edit in their local session. The objects marked for check-in can also be unmarked.

This is necessary primarily in a conflict (indicated by a red flag) in which multiple editors have marked the same object for check-in in their respective local session.

Note

Markings with conflict (red flag)

A marking conflict must be resolved by agreement of the editors, and a sequence for editing the marked objects must be specified. To solve the conflict, one editor must unmark the objects.

Procedure

To unmark objects in a local session, follow these steps:

1. Open the local session and click on the displayed blue or red flag to unmark an object.
2. Alternatively, you can also select the required object and delete the marking with the shortcut menu command "Unmark".

Result

The objects that are no longer marked and objects marked as conflict are displayed as follows:

Icon before	Action	Icon after
	Unmark	
	Unmark	

See also

Introduction to working in local sessions (Page 6788)

Marking objects for editing (Page 6791)

Displaying usage info for marked objects (Page 6794)

Displaying usage info for marked objects

Introduction

For objects to be edited in Multiuser Engineering at the same time, each editor must mark the objects they want to edit in their local session.

To avoid multiple marking of objects and any marking conflicts caused by it, you receive the following information for marked objects:

- User: Indicates who has marked the object
- Computer: Indicates on which computer the object was marked.
- Storage location: Indicates where the object was marked: in a local session or in the server project view.

This information helps you coordinate the further procedure for marked objects or objects that are marked with a conflict with the other users.

Show usage info

To display the usage info for marked objects, follow these steps:

1. Select a yellow or red marked object in your local session or in the server project view.
2. Call the "Usage info" command from the shortcut menu.

Result

The "Usage info" dialog box opens and displays the above-mentioned information for the marked object.

Note

Markings with conflict (red flag)

A marking conflict must be resolved by agreement of the editors, and a sequence for editing the marked objects must be specified. To solve the conflict, one editor must unmark the objects.

See also

Multiuser icons in the user interface (Page 6706)

Introduction to working in local sessions (Page 6788)

Marking objects for editing (Page 6791)

Unmarking objects (Page 6793)

Refresh local session

Introduction

For objects to be edited in Multiuser Engineering at the same time, each editor must mark the objects they want to edit in their local session. Objects are labeled as "outdated" when their version is newer in the server project than in the local session.

You can refresh these objects in the local session by clicking the "Refresh local session" icon in the multiuser toolbar.

Other users may have created new objects or deleted objects in the server project that are not yet displayed in your local session.

These new objects are also displayed in your local session after a refresh.

The contents of "outdated" objects in the local session are overwritten with the more recent contents of the server project during the refresh.

Note

Refresh local session

- The button to refresh the local session opens the view "Refresh", even if there are no objects to be refreshed in your local session. In this case, the button "Start refresh" is disabled in the editor.
- During the refresh, all changes to **unmarked objects** in your local session are overwritten with the contents of the server project. The marked objects remain unchanged.
- You can undo the refresh of the local session if required by clicking the "Restore" button. Your local session will then be restored with the contents prior to the refresh.

Refresh procedure

To refresh objects in a local session, follow these steps:

1. Open the local session that contains the objects that are to be updated.
2. Click the "Refresh local session" button in the multiuser toolbar.
   The multiuser editor opens in the refresh view and shows all objects marked in your local session. The objects you have marked are not overwritten during the refresh. Only unmarked objects are overwritten with the contents of the server project.
3. Click the "Start refresh" button to start the refresh.
   You are kept up-to-date on the progress of the refresh. You will receive an error message if an error occurs.
4. You receive the message that the refresh was completed successfully. Click "OK".

Result

All unmarked objects in your local session were refreshed.

The objects marked in your local session were **not** synchronized with the server version.

Note

Online/offline differences after the refresh

After downloading to the device, the online information on the loaded objects is stored in the local session in the TIA Portal. This enables a comparison between the online and offline version. However, because this online information is not checked in on the server, this information is no longer available in the local session after a refresh. If you then work online on the device after the refresh, the TIA Portal cannot perform a comparison because of missing information. The objects are displayed with online/offline differences although the contents of the objects are identical.

To correct this, perform a new complete download to the device after a refresh of the local session.

Restoring the local session after the refresh

Introduction

Unmarked objects which have a newer version in the server project can be updated in the local session. The unmarked objects in the local session are overwritten with the more recent contents of the server project during the refresh.

You can undo the refresh of the local session if required. To do this, click the "Restore" button in the multiuser editor after the refresh. Your local session will then be restored with the contents prior to the refresh.

Note

"Restore" button

The "Restore" button is only active after a refresh has been performed as long as a restore of the contents of the local session with the last version is possible.

The "Restore" button is inactive if no restore of the local session can be performed, for example because no refresh was performed.

Closing and opening the editor has no impact on the "Restore" button. If a refresh is available that can be undone, the button is still active even if you close the editor and open it again.

Procedure to restore the local session

To restore the local session after the refresh, follow these steps:

1. Open the local session for which you have performed a refresh, if you have closed this in the meantime.
   Start with step 3 if the refresh view is still open.
2. Click the "Refresh local session" button in the multiuser toolbar to open the multiuser editor in the refresh view.
3. Click the "Restore" button to restore the version of the local session before the last refresh.
4. In the next dialog box, click "Yes" to answer the prompt.
5. When errors occur, you receive an error message that you have to acknowledge.
6. When the restore is successfully completed, you receive a message. Click "OK".

Result

Your local session was successfully restored with the version before the last refresh.

See also

Introduction to working in local sessions (Page 6788)

Check-in objects to the server project

Introduction

After editing marked objects in the local session, you can check the changes into the server project.

The following rules apply to check-in:

- Only changes to marked objects are applied during check-in to the server project.
- If you have edited unmarked objects or objects that cannot be marked in the local session, these changes are **not** applied during check-in.
- During the refresh of your local session after check-in, changes to unmarked objects and objects that cannot be marked are overwritten with the contents of the server project.
- When deleting unmarked objects in the local session, these objects are once again present in your local session after the refresh. Because these objects still exist in the server project, they are once again added to the local session.
- You will receive an error message with additional information if an error occurs during check-in. Correct the errors that occurred as specified, either in your local session or in the server project view. Then start the check-in process again.
- During check-in and editing of objects in the server project view, the respective user works exclusively on the multiuser server. The server is therefore locked for all other users. The server status is shown as "not available" in the toolbar during exclusive editing. You can nevertheless create a new local session and refresh local sessions.

Options for the check-in

Compile

You have the option to compile the changes made in the local session before check-in.

Note

Options displayed for the compilation during check-in

The options displayed for the compilation during the check-in depend on the settings made in the TIA Portal under "Options > Settings > Multiuser" for "Compilation settings" .

The compiling options may be deactivated in the multiuser editor based on these settings.

Compile mode

The compile mode defines the extent to which the software changes are compiled. You can select the following options from the drop-down list:

- Marked objects (changes only)
  The changes made to the marked objects for the selected device are compiled together with the device-specific objects.
- Device software (changes only)
  The changes made to the marked objects are compiled together with the software version on the assigned devices.
  Example: You have marked and changed several objects that belong to more than one device (for example to CPU_1 and to CPU_2). These marked objects are compiled together with the software versions of both CPUs during compiling.

Display server project view

When you enable this option, you have the possibility before check-in to again display your changes to the marked objects in the server project view together with the current contents of the server project.

You can mark and edit all objects in the server project view and unmark them. After this, you can either save your changes in the server project or discard your changes and exit the server project view without further action.

Requirement

The connection to the multiuser server exists and the server is available.

You have marked the objects edited in the local session for check-in.

Procedure for check-in of changes

To check-in the changes from the local session to the server project, follow these steps:

1. Open the local session that you want to check in.
2. Click the "Check-in" button in the multiuser toolbar.
   The multiuser editor opens in the "check-in" view and shows all objects marked in the local session that are applied to the server project during check-in.
3. Open the displayed folders and check the objects displayed and marked for check-in.

4. Select the preferred options for check-in.
   - "Compile": Select the required setting from the drop-down list.
   - "Open server project view": When you click this options, the server project view is opened in the project tree prior to check-in.
5. Click "Start check-in" after reviewing to check the displayed objects into the multiuser server project.
   - If you have opened the server project view, you can check in your changes with the "Save changes" button. If you click "Discard changes", your changes are not applied in the server project.
   - If you want to keep the marking in the local session even after the check-in, you must select the option "Keep markings for objects".
   - If you have marked and checked in only some of your changes, you should perform **no refresh** to retain your changes to the unmarked objects.
6. If check-in was successful, you receive a message to this effect.
7. If check-in was unsuccessful and aborted due to errors, you receive a corresponding error message. Correct the specified errors in your local session or in the server project view. Then start the check-in process again.

Result

Your local session is refreshed and opened with the contents of the server project after successful check-in.

- The local session will include all changes after the refresh that have been checked in in the server project in the meantime.
- If new objects were added to the server project, these objects are now also visible in your local session.
- If objects were deleted in the server project, these objects now no longer exist in your local session.
- Your markings in the local session are automatically deleted if you have not selected the option "Keep marking for objects".
- The server connection is released once again after check-in. The server status changes in the display within the toolbar from "busy" to "available".

See also

Possible conflicts during check-in (Page 6800)

Possible error messages during check-in (Page 6803)

Introduction to working in local sessions (Page 6788)

Introduction to working in the server project view (Page 6806)

Editing objects in the server project view (Page 6809)

Possible conflicts during check-in

Introduction

Errors can occur during check-in of changes from the local session to the associated multiuser server project that may prevent successful check-in.

The error constellations listed below can be easily avoided through coordination between editors and by observing the recommended procedures.

Possible naming conflicts during check-in

Naming conflicts during check-in arise when identical names are used for different objects in different local sessions.

Example:

- Editor 1 creates the tag "Motor_1" in their local session.
- Editor 2 also creates a tag "Motor_1" in their local session.

Result:

Editor 1 can check in their changes without errors.

Editor 2 **cannot** check in their changes. When they want to check in their changes they receive an error message due to a naming conflict and the check-in is aborted. The error must be removed in the local session.

Possible conflicts when creating devices in the local session

If an editor adds new devices with marked objects in a local session, for example, a new CPU with blocks, the result will be a conflict during check-in.

Example:

- Editor 1 creates a new CPU with two function blocks in their local session.
- Editor 1 marks the blocks appropriately before editing them.

Result:

Editor 1 receives an error message during check-in because the newly created CPU cannot be found in the server project. The transfer of devices is currently not supported in Multiuser Engineering. Check-in is aborted in this case and the server project view opens. Editor 1 can copy their newly created objects directly to the server project. This means the editing work is not lost.

Possible conflicts when renaming objects in the local session

Objects should be renamed only in the server project view, since this does **not** cause any inconsistencies in the server project.

When renaming objects in the local session, you must ensure that all objects affected by the renaming and their places of use are also marked and checked in the server project. Otherwise, you get compiling errors and an inconsistent server project due to partial renaming.

Before checking the places of use of the renamed object, you should be careful not to create conflicts (red flags) when marking and you should not mark any outdated objects; otherwise, you will overwrite the work of another user when checking in.

Possible conflicts when changing the device type in the local session

If a device type is changed in a local session, e.g. a "CPU_1" of the type S7-15xx is replaced or renamed, the objects marked in the "old" CPU may no longer be displayed in the local session.

Possible conflicts when deleting devices in the server project

If devices with included objects are deleted in the server project, it may result in a transfer error.

Example:

- Editor 1 works in their local session on marked objects that are part of "CPU_1".
- In the meantime, editor 2 deletes "CPU_1" in their server project view and saves the changes in the server project.

Result:

Editor 1 receives an error message during check-in because "CPU_1" no longer exists in the server project. In this case, editor 1 can add their edited objects in the server project view. The editor either copies "CPU_1" completely to the server project or they add the edited blocks to an existing CPU in the server project.

Possible scenarios during check-in of objects that are deleted in the server project

When you check in objects that are already deleted in the server project, these objects are recreated.

Example:

- User 1 edits the marked tag "Tag_1" in their local session that is part of a group that includes a tag table with multiple tags.
- User 2 deletes this group with its entire tag table in the server project view and saves the changes in the server project.

Result:

User 1 ignores the display that "Tag_1" is outdated, continues working on it and checks in the changes. The group with the tag table is created once again in the server project, but it now only includes the tag "Tag_1".

Renaming objects

You should avoid renaming objects in the local session as it results in naming conflicts during check-in. If objects must be renamed, they should be renamed directly in the server project view. Any local session with marked objects that is associated with this server project must not be present at the same time.

Possible name conflicts when checking in deleted but not marked objects

In the change view, you can also adapt the markings for the displayed objects. If the marking of a deleted object is removed in the change view and this object still has a name conflict with another object in the server project, the check-in cannot be performed.

Without the marking, the information about the deletion of the object cannot be transferred to the server. Therefore, an identical name for an object created or renamed in the local session is still occupied by the previous object. During check-in, this results in a critical transfer error that must be corrected in the local session.

Example:

- User 1 deletes the text list "TL_1" in his local session. The deleted text list is automatically marked and thereby transferred at the next check-in on the server.
- User 1 accidentally deletes the marking for deleted text list "TL_1" before check-in in the change editor. Removing the marking means that the information about the deletion of the text list "TL_1" in the local session is not transferred to the server project. The text list "TL_1" still exists in the server project.
- User 1 now creates a new text list, also named "TL_1", in the local session. The user marks this newly created text list and wants to check it into the server project.

Result:

- Since the original text list "TL_1" is still present in the server project, however, the newly created text list "TL_1" in the local session causes a critical transfer error during check-in.
- The error must be corrected in the local session by renaming the newly created "TL_1", for example, to "TL_1_new".
- After this, the check-in can be performed by updating the local session. After the update, both text lists are again available in the local session and User 1 can delete the unneeded text list "TL_1" and check in the change.

Different project languages in server project and local session

If a project language is changed in a local session, these changes are lost in the local session after check-in and after refresh.

If a project language is subsequently deleted in the server project view, this project language no longer exists in the server project.

Changes to the project languages should always be made only directly in the server project view. Note that no local sessions with marking to this server project were edited or opened.

See also

Introduction to working in local sessions (Page 6788)

Opening and closing the server project view (Page 6808)

Possible error messages during check-in

Introduction

The following described error messages can occur during check-in of marked changes from the local session to the multiuser server project :

- Transfer error
- Compilation error

The procedure for dealing with these error messages is described below.

Transfer error

The following actions are automatically executed when transfer errors occur during check-in:

- An error message will inform you about the type of error that has occurred and the next steps. After acknowledgment, this message is closed.
- In the case of a critical transfer error, the status prior to check-in is restored in the local session as well as in the server project. Changes are not lost. Correct the error in your local session.
- If the transfer error can be corrected in the server project view, the server project view is opened. Correct the error in the server project view.
- The faulty objects are displayed and identified in the multiuser editor in the "Check-in" view.
- The multiuser server locked for check-in is released once again if the errors are corrected in the local session.
- The multiuser server remains locked in the server project view during error correction. Note that no other user can check in their changes during this time period.

Correcting transfer errors

After occurrence of a transfer error, you have the following options for the action to take:

1. Correct the errors in your local session or in the server project view.
2. Then start the check-in process again so that an error-free project can be checked in.

Compilation error

If the option "Compile" is selected during check-in, a compilation process takes place after transfer of the marked changes to the server project with the selected option. The following compile modes are available:

- Marked objects (changes only)
  The changes made to the marked objects for the selected device are compiled together with the device-local objects.
- Device software (changes only)
  The changes made to the marked objects are compiled together with the software version on the assigned devices.

Note

Options displayed for the compilation during check-in

The options for compiling that are displayed during check-in are dependent on the settings made in the TIA Portal under "Options > Settings > Multiuser" in "Compilation settings" .

The compiling options may be deactivated in the multiuser editor due to these settings.

If the compiling process is completed successfully, the changes from the local session are automatically transferred to the server project.

The following actions are taken when a compilation error occurs:

- The progress indicator for check-in is closed, and the check-in is stopped.
- The faulty objects are displayed and identified in the multiuser editor in the "Check-in" view.
- The multiuser server remains locked in the server project view during error correction. Note that no other user can check in their changes during this time period.

Correcting compilation errors

After occurrence of a compilation error, you have the following options for the action to take:

1. Correct the indicated errors in your local session or in the server project view.
2. Then start the check-in with compilation again so that an error-free project can be checked in.

Ignoring compilation errors

You also have the option of ignoring the indicated compilation errors and checking in a project with errors.

NOTICE
Check-in with compilation errors This action is possible but not recommended! To enable efficient work to continue for all editors involved, you should check in only error-free projects on the multiuser server whenever possible.

See also

Introduction to working in local sessions (Page 6788)

Deleting objects in the local session

Introduction

When working with Multiuser Engineering , you can delete objects in the local session.

The following rules apply to deleting objects:

- Deleted objects are no longer displayed in the local session.
- The "Undo" function can be use to undo the "Delete" action in the local session.
- Deleted objects are automatically marked by the system and displayed in the "Deleted objects" folder.
- If you unmark a deleted an object in the multiuser editor, the deleted object is **not** checked-in in the server project. This means that the deleted object is **not** deleted in the server project. The now deleted but not checked-in object is only displayed after the next refresh.
- When deleting unmarked objects in the local session, these objects are once again present in your local session after the refresh. Only marked objects are transferred during check-in. Since the deleted objects could not be marked, the deletion is not applied in the server project. During the next refresh of the local session, these objects are once again added to the local session.

Requirement

The connection to the multiuser server exists and the server is available.

Procedure for deleting objects

To delete object in the local session, follow these steps:

1. Open the required local session.
2. Delete the requested marked object or delete an unmarked but markable object.
3. Click the "Check-in" button in the multiuser toolbar.
   The multiuser editor opens in the "check-in" view and shows all objects marked in the local session that are applied to the server project during check-in. The objects deleted in the local session are displayed in the "Deleted objects" folder.
4. Check the objects displayed and marked for deletion. You can unmark any objects that are already marked if you do not want to delete them after all.

5. Select the preferred options for check-in:
   - "Compile": When you click this option, your changes are compiled with the option "Software, changes only" prior to check-in. This means that the software changes you have made are compiled with all their dependent objects in the server project.
   - "Open server project view": When you click this options, the server project view is opened in the project tree prior to check-in. The server project view shows you in advance the content of the current server project with your changes marked for check-in. You can mark and edit all objects in the server project view and unmark them.
6. Click the "Check-in" button to check in the displayed objects into the multiuser server project. If you want to keep the markings in the local session even after check-in, you must select the option "Keep markings".
7. If check-in was unsuccessful and aborted due to errors, you will receive the corresponding error messages. Correct the errors in your local session or in the server project view. Then start the check-in process again.

Result

Your local session is refreshed and opened with the contents of the server project after successful check-in.

- The local session will include all changes after the refresh that have been checked in in the server project in the meantime.
- If new objects were added to the server project, these objects are now also visible in your local session.
- The objects deleted in the server project, now no longer exist in your local session.
- Your markings in the local session are automatically deleted if you have not selected the option "Keep objects marked".
- The server connection is released once again after check-in. The server status changes in the display within the toolbar from "busy" to "available".

15.3.15 Working in the server project view

15.3.15.1 Introduction to working in the server project view

Introduction

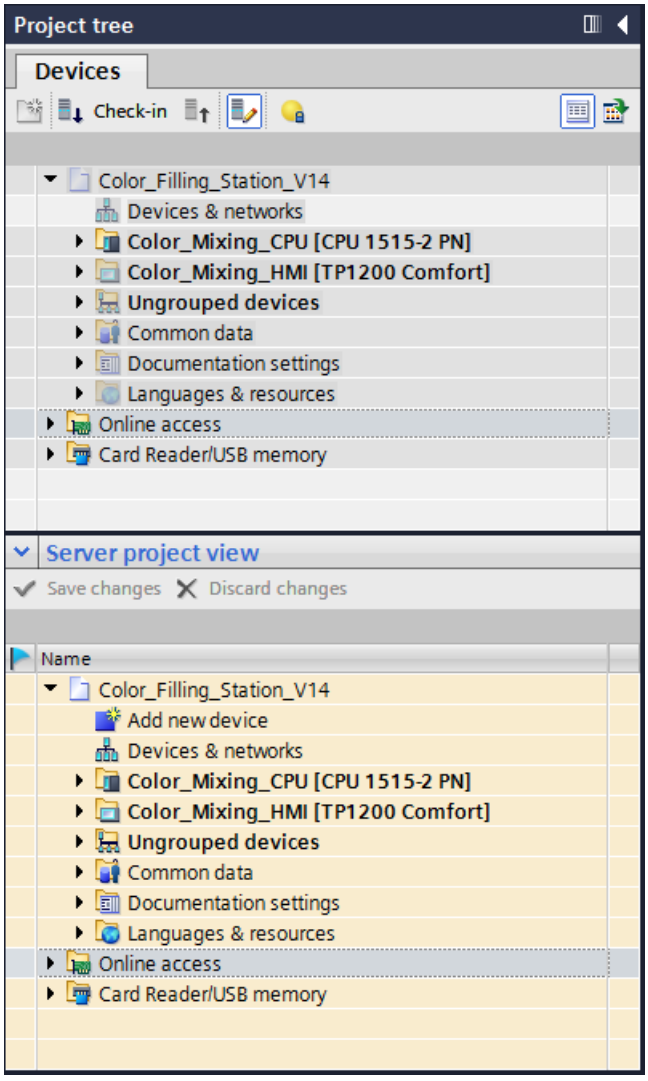
In Multiuser Engineering , you have the option of working directly in the server project view in addition to working in local sessions.

The server project view can be opened for editing with the icon in the multiuser toolbar.

Structure of the server project view

The server project view shows the current content of the server project in the project tree below the local session against a yellow background.

The figure below shows an example of a multiuser project opened in the server project view:



Actions in the server project view

You can perform the following actions in the server project view:

- The server project view lets you work directly in the server project.
- All existing objects can be edited in the server project view, even those that could not be edited in the local session.
- In the server project view you can work with the functionality of Inter Project Engineering (IPE) and exchange data between CPU and HMI via a PLC proxy.
- Shared working on one CPU is also supported during shared commissioning.

- Blocks or larger program parts from a CPU should only be uploaded in the server project view.
- You can also make changes to the configuration with the help of external tools in the server project view. This function is not supported when working in the local sessions.
- Changes to user-defined documentation, such as adding new user texts, should only be made in the server project view.
- The objects that are to be edited must also be marked for editing in the server project view so that other editors see that these objects are being changed.
- You can save or discard the changes made in the server project view with the "Save changes" and "Discard changes" buttons.
- Any changes you have made will be saved directly in the server project with the save step.

Note

Restrictions when working in the server project view

The following conditions apply as soon as the server project view is open:

- The local session is locked to prevent editing.
- The server is locked for the other editors.
- The icon for the server status is shown in yellow in the toolbar.

See also

Opening and closing the server project view (Page 6808)

Editing objects in the server project view (Page 6809)

15.3.15.2 Opening and closing the server project view

Introduction

The server project view lets you edit all objects that are available in the server project.

You have the following options to open the server project view:

- Using the icon in the multiuser toolbar
- Using the link displayed in the banner texts
- Using the button in the multiuser editor in the "Check-in" view

You will receive a message if the server project view cannot be opened because the server is currently in use by another editor and locked.

Displayed contents in the server project view

The server project view always shows the latest version of the multiuser server project.

If the server project view is opened in the "Check-in" view in the multiuser editor, it also shows the changes made in the local session in addition to the current content of the server project. This means it offers a preview of how the server project will look after check-in of the changes made in the local session.

Opening the server project view

To open the server project view, follow these steps:

1. Open the required local session.
2. In the multiuser toolbar, click the "Open/close server project view" icon.
   Or you can click "Check-in" in the multiuser toolbar and then click "Open server project view before check-in" in the multiuser editor that opens.
   Alternatively, you can also click on the link in a displayed banner text.

Closing the server project view

To close the server project view, follow these steps:

1. In the multiuser toolbar, click the "Open/close server project view" icon once again.
2. If you have made changes in the server project view, you will be prompted as to whether you want to save or discard the changes.
3. Click "Yes" to save your changes or "No" to discard them.
   Alternatively, you can click on the corresponding button in the server project view before you close it to save or discard your changes.

Result

The server project view is closed.

The local session is opened once again for editing.

The server is released again; the server status is displayed in green.

See also

Introduction to working in the server project view (Page 6806)

Editing objects in the server project view (Page 6809)

15.3.15.3 Editing objects in the server project view

Introduction

In the context of Multiuser Engineering you can also edit objects directly in the server project view .

The server project view lets you edit all objects in the server project view, even those that must not be edited in the local session.

Working in the server project view is an exclusive working directly in the multiuser server project. The multiuser server is locked for other editors while you are working in the server project view. The associated local session cannot be edited during the working in the server project view.

You also have to mark the objects you want to edit in the server project view. This enables you to ensure that other editors in your local session can see that you are currently editing these objects. Marking conflicts can hereby be avoided.

Note

Marking of identical objects in the local session and in the server project view

Keep in mind that a marking conflict is displayed with red flag if you mark an object more than once, even if both markings belong to the same editor.

Editing objects in the server project view

You can, for example, perform the following actions in the server project view:

- Adding new objects that did not exist previously in the server project.
- Add or change hardware configuration and connections.
- Perform version changes on the instructions used.
- Working with Inter Project Engineering (IPE).
- Editing all objects regardless of whether these are marked or not.
- Renaming, moving and deleting objects.
- Establishing an online connection to a CPU and downloading or uploading objects.
- Executing all functions of the TIA Portal supported by Multiuser Engineering.

Procedure for editing objects in the server project view

To edit objects in the server project view, follow these steps:

1. Open the required local session.
2. In the multiuser toolbar, click the "Open/close server project view" icon.
   The server project view is then shown against a yellow background below the local session in the project tree.
   The server project view is closed again the next time you click this icon.
3. Mark the objects you want to edit in the server project view by clicking on the flag in front of them.
   Alternatively, you can also edit unmarked objects in the server project view.
4. Make your required changes.
5. Save your changes by clicking the "Save changes" button.
   If you do not want to apply your changes to the server project, click "Discard changes".

Result

The changes made in the server project view are saved in the server project.

- The server project view is closed and the local session is once again released for editing.
- The markings are removed in the server project view.
- The local session is **not** automatically refreshed with the new contents of the server project.

See also

Introduction to working in the server project view (Page 6806)

Opening and closing the server project view (Page 6808)

15.3.15.4 Renaming objects in the server project view

Introduction

When you work with Multiuser Engineering , object can be renamed.

Rules for renaming objects

The following rules apply to renaming objects:

- You should avoid renaming objects in the local session as it can lead to naming conflicts during check-in.
- The renaming of objects should always be performed in the server project view.
- When objects are renamed in the server project view, the objects to be renamed must have no markings within local sessions.
- Name conflicts may occur due to the renaming. During renaming in the project, make sure that you do not again use an already used name.
- To avoid unintended references of objects, conflicts that occur during renaming are not automatically corrected.
- The renaming of devices does not create any conflicts for objects within a device, as devices are not identified by their name but by their ID.
- However the renaming of devices may create conflicts with any existing references between devices (PLC and HMI).
- After objects have been renamed, the other editors should refresh their local sessions.

Requirement

The connection to the multiuser server exists and the server is available.

No other local session to this server project is open.

Renaming objects

To rename objects in the server project view:

1. Open the required local session.
2. In the multiuser toolbar, click the "Open/close server project view" icon.
   The server project view is then shown against a yellow background below the local session in the project tree.
   The server project view is closed again the next time you click this icon.
3. Mark the required objects in the server project view and rename these.
4. Click the "Save changes" button to apply the changes in the multiuser server project.

Result

The changes made in the server project view are saved in the server project.

- The server project view is closed and the local session is once again released for editing.
- The markings are removed in the server project view.
- The local session is **not** automatically refreshed with the new contents of the server project.

Index

-

-, 4993

"

"Change color reference" dialog, 5903
"WWW" instruction, 1073

#

#, 4773

&

&, 5000, 5001

(

(), 5001

*

*, 4993
**, 4993
*.bmp, 5313
*.cer, 838
*.dat, 838
*.emf, 5313
*.gif, 5313
*.ico, 5313
*.jpeg, 5313
*.jpg, 5313
*.p12, 838
*.tif, 5313
*.wmf, 5313

.

.Net service packs, 66
.Net versions, 66

/

/, 4993

:

:=, 5001, 5002
:P, 971

+

+, 4993

<

<, 4995
<=, 4995
<>, 4995, 5001

=

=, 4995
==, 5001

>

>, 4995
>=, 4995

3

32-bit down counter, 1919
3DES, 810

A

AB, 4841
Ability to read back connection parameters, 638
ABS, 2707, 3033, 3272
Absolute Addressing
    of a tag, 5425
Absolute value, 2707, 3033, 3272
Access control, 1430
    Automatic learning, 1430

Access point
Overlapping channels, 1584
Overview, 1579
Access protection, 5638
Access protection level 1, 6588
Access protection level 2, 6588
Advantages, 6588
Alarm view, 5403
Configuring, 5642
Factory state, 6587
Parameter assignment, 1133
User administration, 5637
Validity, 6587
Access to operands, 2147, 2148, 2150, 2152, 2153, 2154, 2155, 2276, 2281, 2285, 2292
Accessible devices, 1798
Accuracy, 5005
ACK
Key, 5941
Acknowledge, 5486, 5776
Key, 5941
AcknowledgeAlarm, 5722
Acknowledgment, 5489
Configuring, 5512, 5513, 5514
Acknowledgment concept
Alarm with simple acknowledgment, 5491
Alarm without acknowledgment, 5491
Acknowledgment group
Text, 5908
Acknowledgment model, 5491
ACL, 1430
ACOS, 2724, 3049, 3286
Acquisition cycle
Tag, 5446, 5453, 5468
Acquisition mode
Tag, 5446
ACT_TINT, 3819
Actions
Basics of redoing actions, 316
Basics of undoing, 316
Redoing, 319
Undoing, 318
activate, 5769
Project language,
ActivateCleanScreen, 5743
ActivatePreviousScreen, 5673
ActivateScreen, 5671
ActivateScreenByNumber, 5672
Activating
Font size adjustment, 5274
Active bus module (ET 200M), 2003
Active nodes, 817
AD, 4841
Adapting
Connection name, 6681
Adapting a project
For a different HMI device, 5957
Add, 2698, 3023
Graphic to graphics library, 5862
Objects to the group, 5320
Add empty line, 6248
Add separator line, 6248
Adding
Adding timers with T_ADD, 3458
Additional field devices (PROFIBUS and PROFINET), 1761, 1808, 1809
address conversion
Instructions for, 4073
Address multiplexing
with absolute addresses, 5454
with symbolic addresses, 5455
Address overview, 577
Address packing, 1751
Address range, 777
Changing, 1046
Address register, 2169
Addresses
Assigning, 1048
Determining a module with GEO_LOG, 4086
Interrupt with packed, 1757
Pack, 1751
Read station address with GetStationInfo, 3957
Reading out the MAC address with GetStationInfo, 3957
Unpack, 1751
Addressing, 971
Addressing tags indirectly, 5450, 5451
Changing, 1046
General, 1045
Multiplexing, 5450
Addressing operands, 2147, 2148, 2150, 2152, 2153, 2154, 2155, 2158, 2164, 2168, 2169, 2276, 2281, 2285, 2292
AdjustContrast, 5749
Administrator, 736
Advanced Encryption Algorithm, 918
Advanced Encryption Standard (AES), 810
Advanced mode
Global firewall rules, 748
Advanced offline/online comparison
Automatic device assignment, 719
Advanced recipe view, 5578
AES, 810, 918
AES-128, 802

Aggressive mode, 809
Aging time, 1434, 1436
Alarm
  Components, 5492
  Configuring, 5498, 5505
  Displaying, 5399
  Exporting, 5824
  Importing, 5826
  In Runtime, 5525
  System function, 5532
Alarm buffer, 5521
  In Runtime, 5526
  Memory size, 5521
Alarm buffer overflow, 5521, 5776
Alarm class, 5492
  In Runtime, 5525
  Layout, 5527, 5529
Alarm classes, 5488
  Colors, 5405, 5908
  Custom, 5488
  Name change through migration, 192
  Predefined, 5488
  Use, 5488
Alarm event
  Acknowledge, 5486
  Incoming, 5486
  Outgoing, 5486
Alarm group, 5491, 5493
  Configuring, 5496
  Creating, 5496
  Migration, 191
Alarm indicator, 5408, 5508, 5530
  Alarm classes, 5409
  Configuring, 5510
  Events, 5409
  In Runtime, 5526
  Layout, 5530
  Operation, 5530
Alarm line, 5402
Alarm log, 5515, 5561
  Configuring, 5518
  Creating, 5515, 5518
  Naming conventions, 5564
Alarm number, 5492
Alarm status, 5493
  Acknowledged, 5486
  Incoming, 5486
  Outgoing, 5486
Alarm system, 5483
Alarm text, 5493
  Formatting, 5502
  Removing format settings, 5502
Alarm types, 5484
Alarm view, 5399, 5406, 5508, 5528
  ~ configuring for logged alarms, 5520
  Access protection, 5403
  Alarm text window, 5528
  Application, 5528
  Column, 5401
  Column headers, 5405
  Column sequence, 5404
  Configuring, 5509
  Configuring the layout, 5509
  Define columns, 5404
  Define display area, 5405
  Filter alarms, 5404
  Identify alarm classes, 5404
  Layout, 5400, 5527, 5529
  Operation, 5528, 5529
  Operator control, 5401, 5529
  Operator controls, 5403
  Select alarm classes, 5403
  Sorting, 5404
Alarm window, 5508, 5528
  Application, 5528
  Configuring, 5510
  In Runtime, 5526
  Operation, 5529
  Operator control, 5529
Alarm with simple acknowledgment, 5491
Alarm without acknowledgment, 5491
Alarms
  Output of a tag value, 5500
AlarmViewAcknowledgeAlarm, 5721
AlarmViewEditAlarm, 5720
AlarmViewShowOperatorNotes, 5721
Align
  Object flush, 5293
Allen-Bradley DF1
  Migrating data types, 204
Allen-Bradley Ethernet IP
  Migrating data types, 205
Alphanumeric key assignment, 5941
ALT
  Key, 5941
Analog alarm, 5485
  Configuring, 5498, 5506
  Limit value exceeded, 5444
Analog alarm types, 5485
Analog alarms
  Configuring, 5493
Analog module
  Resetting to factory settings, 2068
Analog value, 2810, 2812, 3135, 3138, 3371, 3373

Analog value processing, 2810, 2812, 3135, 3138, 3371, 3373
AND, 2848, 2895, 2896, 3172, 5000, 5001
Animation
  Configuring, 5349, 5577
  Diagonal movement, 5353
  Direct movement, 5353
  Green arrow in Overview, 5349
  Horizontal movement, 5352
  Multiple selection, 5357
  Object group, 5356
  Overview, 5349
  Vertical movement, 5353
Antivirus programs, 123, 134
ANY, 2305
Appearance
  Dynamization of an object, 5350
Applet, 738
Applicant, 920
Application, 1812
  Alarm view, 5528
  Alarm window, 5528
  Recipe view, 5602
  Simple user view, 5385
Arccosine, 2724, 3049, 3286
Archiving projects, 341
Arcsine, 2723, 3048, 3285
Arctangent, 3287
Arctangent value, 2726, 3051
Area of unplugged modules, 535
Area pointer
  Data record, 5573
ARP, 888
  Diagnostics, 888
  Table, 888
Arrange
  Object in the screen, 5286
Arrangement of byte sequence, 3096, 3322
ARRAY, 2164, 2215
  Addressing, 2153, 2292
  Creating,
  Declaration in global data blocks, 4803
  Declaration in PLC data types, 4871
  Declaration in the block interface, 4775
  Example, 2298
  Format, 2286
  Indirect addressing,
  see ARRAY,
ARRAY data block, 2091, 2092, 2102, 2150, 2153, 2772, 2774, 2777, 2780, 3097, 3100, 3102, 3105, 3323, 3325, 3327, 3329, 4716, 4795, 4798, 4831
ARRAY DB, 2092
Array elements, 5464
ARRAY limits, 5013
Array tag, 5464
Article number, 545, 1280, 1559
AS interface, 1764
ASCII code table, 641
ASCII TSAP, 641
ASI_CTRL, 3721
Asian, 66
Asian characters
  Input on the HMI device, 5872
  Interpretation, 5872
  Memory requirements, 5872
Asian languages
  Configuration, 5872
  Font size, 5872
  Text field length, 5872
Asian operating system, 5853
ASIN, 2723, 3048, 3285
Assembling manuals, 300
Assign
  a graphic to a function key, 5368
  Object of a layer, 5375
Assigning
  a function to a function key, 5364
  Function key, 5359, 5361, 5363
Assigning an IP address
  Basics, 2059
  from the project context, 2061
  Using "Accessible devices", 2060
Assigning interfaces, 1139, 1141
  Reference lines, 1139
Assigning symbol, 1048
Assigning tag, 1048
Assignment, 2578, 2902, 5002
  Negate, 2579, 2903
Assignment list
  Defining filter,
  Delete filter,
  Displaying,
  Enabling the display of retentive bit memories, 5129
  Example for displaying bit memory, 5122
  Example for displaying inputs and outputs, 5122
  Filter options,
  Filtering,
  Introduction,
  Meaning of symbols,
  Setting view options,
  Structure, 5122
Asymmetric encryption, 918

Asynchronous error event
Delaying with DIS_AIRT, 3869
Disabling with DIS_IRT, 3861
Enabling with EN_AIRT, 3870
Enabling with EN_IRT, 3863
AT, 2155
ATAN, 2726, 3051, 3287
ATH, 3526
ATTACH, 978, 3795
Attacks, 1013
ATTR_DB, 4059
Audit log, 833, 835
Authentication, 1354, 1503, 1629
Authentication methods, 806, 808
Authorization
Assignment, 5622, 5644
Changing the name, 5627
Configuring, 5636
Creating, 5620, 5642
Deleting, 5627
Managing, 5625
Autocompletion
Function, 4706
Insert tag, 4706, 4707
Inserting an instruction, 4707
Automatic commissioning, 1798
Automation system
Local, 6621
Remote, 6621
Autonegotiation, 1797
Availability
Object for Basic Panels, 5382
Availability for specific devices
Screen, 5255
Available system functions, 1229, 1516, 1521, 1524
Basic Panels, 5659, 5665
AW, 4841
AWP command, 1059, 1061
AWP_In_Variable, 1064
AWP_Out_Variable, 1062
AX, 4841
Axis and command table technology object
List of ErrorIDs and ErrorInfo, 6304, 6386

B

Background color
Dynamization, 5350
Backing up the table layout, 266
Backing up the user interface, 266
Backing up the layout in editors, 266
BACKSPACE key, 5940
Backup
Data of the HMI device, 5951
Deleting, 5899
Rename, 5899
Backup from online device, 6491, 6493
Bar, 5383
Color transition, 5384
Display limit lines, 5384
Bar segment
Defining, 5384
Barcode, 5947
Barcode scanner, 5946
BA-Send, 1947
BA-Send 1xFC, 1947
Basic mode, 5197
Basic Panel
Display and operating object, 5382
Runtime start, 5925
Basic Panels
Available system functions, 5659, 5665
Basics
Migration, 181
BCDCPL, 2893, 3217, 3453
Behavior
Simple recipe view, 5596
Benefits of using TeleService, 6573
Bit (0, 1)
Graphics list, 5343
Text list, 5334
Bit field
Reset, 2907
Resetting, 2582
Set, 2906
Setting, 2581
Bit logic operation
AND, 2895, 2896
EXCLUSIVE OR, 2899
Insert input, 2900
OR, 2896, 2898
Bit mask, 2884, 3209, 3445
Bit memories
Enabling the display of retentive bit memories, 5129
Bit number (0 - 31)
Graphics list,
Text list,
Bit selection
Design analysis, 5907
Text and graphics lists, 5907
Bit string, 2244, 2245
64-bit, 2246

Bits
Count, 2894, 3454
Counting, 3218
BITSUM, 2894, 3218, 3454
BLKMOV, 2794, 3119, 3355
Block,
Block, 3119
Changing passwords for know-how protected blocks, 4761
Closing, 4735
Comparing, 5088
Comparing code blocks, 5085
Comparison, 5083
Compiling, 5061
Compiling in the program editor, 5064
Compiling in the project tree, 5063
Consistency check, 5061, 5134, 5139, 5140
Consistency check in the call structure, 5135
Copying, 4719, 4722
Deleting offline, 4736
Deleting online, 4737
Displaying properties, 4732
Download to device, 5074
Downloading to a memory card, 5080
Downloading to device, 5066
Downloading to device in RUN operating mode, 5070
Editing properties, 4732
Entering a comment, 4724
Entering a title, 4723
exporting to an external source file, 4741
Fill, 2764, 2799, 3089, 3124, 3315, 3360
Fill uninterruptible, 2767, 3092, 3319
Find and open, 4734
Inserting, 4719
Know-how protection, 4746
Leave, 3396
Loading blocks from device, 5079
Move, 3069, 3297, 3355
Move uninterruptible, 2761, 2797, 3086, 3122, 3313, 3358
Moving, 2743, 2794
Opening, 4733
Opening know-how protected blocks, 4759
Optimized access, 2102, 2104
Pasting, 4723
Printing know-how protected blocks, 4760
Properties, 4726
Remove write protection, 4756
Removing copy protection, 4754
Renaming, 4736
Saving, 4735
Setting up copy protection, 4754
Setting up write protection, 4756
Time stamp, 4730
Types, 2088
Upload from device, 5066
Uploading blocks from a memory card, 5081
Using a library, 4718
Write to memory card, 5080
Block access
Data block, 4809
Block call
"Call by reference" or "Call by value", 2126
Basics, 2107
Calling as single instance or multi-instance, 2110
Changing, 4902, 4959
Correcting the call type, 4902, 4959
Inserting, 4899, 4956, 5040, 5042, 5043, 5044, 5045
Multi-instance, 2112
Nesting depth, 2108
Single instance, 2111
Transfer parameter as copy or as pointer, 2126
Updating, 4900, 4957, 5047
Block comment
Hiding, 4694
Showing, 4694
Block comparison
Basics, 5083
Comparing code blocks, 5085
Detailed comparison, 5090, 5093, 5095, 5099, 5102, 5105
Execute action, 375
Navigation, 5111
Synchronize scrolling, 5112
Updating comparison results, 372, 5114
Block consistency
Checking, 5135
Checking in the dependency structure, 5140
Block folder, 4714
Block interface
Declare STRING, 4778
Declaring ARRAY, 4775
Declaring PLC data type, 4778
Declaring STRUCT, 4777
Declaring tags, 4772, 4773, 4779
Hiding, 4694
Importing and exporting tags, 4791
Multi-instance, 4780
Purpose of variable declaration, 4764
Retentivity, 4769
Showing, 4694
Structure, 4771

Tag properties, 4784, 4786
Updating, 4791
Valid data types, 4767, 4769
Block parameters, 2122, 2123, 2126, 2129, 2131, 2132, 2133
Basics, 1039
Block interface, 4771
Block property
Displaying, 4732
Editing, 4732
Function, 4725
Overview, 4726
BLOCK_DB_TO_WORD, 2567
Bookmarks
Deleting, 5022
Function, 5020
Navigating, 5021
Setting, 5020
BOOL, 2243, 2334, 2382, 2468, 2487, 2534, 2545
BOOL_TO_, 2382, 2487, 2545
Borders
Placeholder for document information, 393
Specifying the print area, 392
Boundaries, 1798
Boundary reached, 5776
Branch, 3385, 3386
Closing, 4918
Definition, 4916, 4973
Deleting, 4919, 4975
Inserting, 4918, 4974
Rules, 4917, 4974
BRCV, 4410
Bridge priority, 1556
Broadcast, 778, 786, 1437
Brute force, 1013
Brute-force attack, 1013
Brute-force attacks, 1013
BSEND, 4407
Buffer overflow, 5908
Button, 5414
Configuring, 5642
Configuring access protection, 5638
Define hotkey, 5415
Graphic, 5415
Mode, 5415
Text, 5415
Buttons and Switches
Library, 5798
BY, 3388
BYTE, 2244, 2335, 2384, 2469, 2488, 2535, 2546
BYTE_TO_, 2384, 2488, 2546
Bytes
Swap, 2770, 3096, 3322

C

CA certificate, 728, 732, 6608
CA group certificate, 732
Cabling rules (PROFINET), 1797
CALC, 2695, 3020
CALCULATE, 4903, 4960
CalibrateTouchScreen, 5750
Calibrating, 2066
Overview, 2065
Call environment
Basics, 5178
Setting for blocks, 5179
Call hierarchy, 2108
Call structure, 5129
Displaying, 5133
Introduction, 5129
Meaning of symbols, 5131
Setting view options, 5133
Structure, 5132
Callback to a number specified during connection establishment, 6590
Calling FAQs for a module, 549
Calling manuals for a module, 549
Calling user-defined documentation, 435
CAN_DINT, 3832
CAN_TINT, 3818
Canceling a calibration, 2068
Canceling printing, 398
Card type, (See Memory card)
Cascaded counting function, 1919
CASE, 3386
Catalog, (see Hardware catalog)
CD, 2661, 2987
CEIL, 2806, 3131, 3366
CER, 938
Certificate, 729, 807
Exporting, 729
Importing, 729
Renewing, 731
Replace, 732
Replacing, 732
self-signed, 731
signed by certificate authority, 731
Certificate authorities, 920
Certificate authority, 728, 729
Certificate check, 5948
Certificate holder, 920
Certificate manager, 729

Certificate manager (global), 936
Certificates
Deleting (global), 942
Displaying, 941
Exporting (global), 938
Importing, 938
Renewing, 940
Replacing, 941
Change, 5769
ChangeConnection, 5716
Changeover contact, 2004
Changeover contact sensor type, 2004
Changing
Colors, 5903
Displayed name of user group, 5626
Logoff time in runtime, 5633
Name of the user, 5633
Object size, 5290
Password, 5625
Recipe data record in runtime, 5606
User group in runtime, 5633
Changing a port interconnection
Graphic view, 713
Changing and displaying operating mode (example), 5655
Changing the configuration online, 6477
Changing the device configuration online, 6477
Changing the name
Authorization, 5627
User group, 5626
Users, 5624
CHAR, 2269, 2376, 2456, 2483, 2484, 2526, 2527, 2544, 2565
CHAR_TO_, 2456, 2526, 2527, 2565
Character, 2269
Character string, 91, 2270, 2273
Combining character strings with CONCAT, 3533
Comparing string tags with S_COMP, 3492
Converting from a hexadecimal number with HTA, 3529
Converting number string with STRG_VAL, 3497
Converting number to character string with VAL_STRG, 3501
Converting to a hexadecimal number with ATH, 3526
Converting with S_CONV, 3494
Copying characters to character string with Chars_TO_Strg, 3509
Deleting characters with DELETE, 3539
Determine maximum length with MAX_LEN, 3512
Determining length with LEN, 3531
Finding a character with FIND, 3546
Inserting characters with INSERT, 3541
Moving with S_MOVE, 3490
Reading out left character with LEFT, 3534
Reading out middle character with MID, 3537
Reading out right character with RIGHT, 3536
Replacing character with REPLACE, 3543
Characters, 66
Chars_TO_Strg, 3509
Checking
Device version, 5897
Checking the connection, 4538
Checklist for troubleshooting the modem, 6631
Checksum
Reading out with GetChecksum, 3963
Circle, 5396
Radius, 5396
Circular log, 5516
Select size, 5474, 5524
Class of Service, 1390
ClearAlarmBuffer, 5718
ClearAlarmBufferProTool, 5719
ClearAlarmBufferProtoolLegacy, 5719
ClearDataRecord, 5731
ClearDataRecordMemory, 5732
cleared, 5769
ClearLog, 5683
Click, 5774
Click when flashing, 5775
Client
Overview, 1585
Client Supplicant, 1722
Clipping, 1968
Clock memory, 982, 1006
CloseAllLogs, 5685
Closed loop control, 6220
CMP <, 2677, 3002
CMP <=, 2673, 2998
CMP <>, 2667, 2992
CMP ==, 2662, 2988
CMP >, 2675, 3000
CMP >=, 2671, 2997
Coil, 2578
Negate, 2579
Color
Diagnostics of Ethernet cables, 2030
Diagnostics of ports, 2030
Color transition
Bar, 5384
Colors
Alarm classes, 5405, 5908
Changing, 5903
Find and replace, 5902

Column
Alarm view, 5401
Column headers
Alarm view, 5405
Recipe view, 5413
Column sequence
Alarm view, 5404
Column width
User view, 5388
Command table technology object
Add new object, 6242
Basic parameters, 6244
Command table configuration, 6245
Command.Command[1...32] tags, 6472
Configuration window icons, 6243
Configuring activate warnings, 6245
Configuring duration, 6247
Configuring position / travel path, 6246
Configuring the command type, 6245
Configuring the next step, 6247
Configuring the step code, 6247
Configuring use axis parameters of, 6245
Configuring velocity, 6246
Extended parameters, 6257
General configuration, 6244
Tools, 6241
Usage, 6241
Comments
Inserting in SCL program, 5050
Comments in Runtime, 5907
Communication, 1012, 1141
Interfaces, 1141
Number of connections (PROFINET/ PROFIBUS), 682
Routing, 6667
S7 routing, 6671
Communication diagnostics, 2044
Communication error, 1152
Communication instruction
"PG_DIAL", 6621
"SMS_SEND", 6623
Communication load, 982
Communication via PUT/GET instructions
Basic information on PUT/GET instructions, 666
Creating and assigning parameters to a connection, 671
Deleting an interconnection, 672
Overview of connection configuration, 667
Requirements, 666
Starting connection parameter assignment, 670
Communications
Cycle load, 989
Communications instruction
"AS_DIAL", 6621, 6622
"AS_MAIL", 6624
"PG_DIAL", 6620
Communications load, 989
Communications module (CM), 4607
Communications modules, 998
Properties, 998
Communications options, 1721
Communications port
Configuring, 998
Communications protocol
Defining, 1001
Overview, 1001
Comparator operations
Time tags, 3455
Compare
Bit mask, 2884, 3209, 3445
Character strings with S_COMP, 3492
Equal, 2662, 2988
Greater or equal, 2671, 2997
Greater than, 2675, 3000
Less or equal, 2673, 2998
Less than, 2677, 3002
Not equal, 2667, 2992
Compare editor
Changing the view, 376
Device comparison, 583
Filtering the view, 370
Overview, 363
Showing and hiding columns, 369
Specifying actions, 374
Compare offline/online
Automatic device assignment, 717
Compare scan matrix, 2886, 3211, 3447
Comparing library elements, 67
Comparing library objects, 440
Compatibility, 75, 90
Backward compatibility of projects, 333
Projects from older program versions, 333
S7-1200 CPU firmware versions, 1043
WinCC version, 5892
Compatibility check, 5907
Project ID, 5907
Compatibility mode, 5441
Compile
Block, 5063
Compiling
Blocks, 5061
Consistency check, 5061
Correcting compilation errors, 5065

Migrated project, 185
Project, 5915
Components
Alarm, 5492
CONCAT, 3533
Conditional spontaneous, 1196
Configuration
Advanced configuration, 1134
DP slave, 1745
DP slave, simple, 1737
Hardware, 555
Host interface, 1134
Loading to device, 1815
Loading to PG/PC, 1816
PC systems, 1134
Tag, 5437
Configuration control, 1085, 1094, 1934, 1939, 1981, 1991, 2013
Configuration control with ET 200pro, 2013
Configuration control with ET 200AL, 1981
Configuration control with ET 200MP, 1991
Configuration control with ET 200SP, 1934
Configuration file, 1111, 1112
Configuration mode, 1327, 1600
Configuration rights, 738
Configure a PROFIBUS subnet
Matching parameters to one another, 596
Setting bus parameters, 596
Configuring, 5498, 5507
Access protection, 5642
Alarm acknowledgment, 5512, 5513, 5514
Alarm group, 5496
Alarm log, 5518
Alarm view, 5509
Alarm view for logged alarms, 5520
Analog alarm, 5498, 5506
Analog alarms, 5493
Animation, 5577
Authorization, 5636
Controller alarms, 5493
Direct connection, 6669
Discrete alarm, 5496, 5505
Discrete alarms, 5493
Event, 5576, 5581
Event-driven tasks, 5500
HART variable, 1911
Integrated, 6674
I-slave, 1747
Layout of the alarm view, 5509
Logging tag, 5479
Multiple tags, 5438
PC station, offline, 1112
Recipe view, 5591, 5593
Rectangle, 5327
System events, 5493
Trend view for values from the PLC, 5480
Configuring a CPU
Loading a project, 1108
Configuring a PROFIBUS subnet
Meaning of the bus parameters, 597
Configuring HMI array tags, 5467
Configuring internal subnets manually, 879, 915
Configuring IP network nodes manually, 879
Configuring MAC network nodes manually, 879
Configuring the network with Ethernet, 601
Creating private subnets, 603
Linking networks, 603
Relationship between IP address and subnet mask, 602
Setting the IP address, 602
Setting the subnet mask, 602
Configuring the remote modem, 6586
Connecting a TS Adapter with an external modem, 6586
Connecting a TS Adapter with an internal modem, 6586
Connection, 643, 653
Address details, 2082
Configuring a connection when there is no or no clear network assignment, 658
Creating, 616
Deleting, 619
integrated connection, 5425
Connection configuration
Connection parameters, 612
General, 608
Overview, 610
Starting, 615
Connection description
Changing parameter values, 638
Data block, 622, 625, 627, 628, 629, 631
Structure, 623, 625, 627, 629, 630, 631
Connection details, 2082
Connection diagnostics
Detailed, 2080
Overview, 2078
Connection information, 2080
Connection mechanisms, 1012
Connection name
Adapting, 6681
Connection parameter assignment of PUT/GET instructions, 670
Connection parameters of PUT/GET instructions, 669

Connection resource, 644, 654
Connection resources, 677
Data record routing, 690
Display in STEP 7, 692
Display in the Web server, 695
HMI communication, 690
Module-specific, 683, 693
occupying, 691
Online, 2080, 2081
Overview, 686
S7 routing, 690
Station-specific, 683, 693
Connection rules, 900, 913
Connection status
Displaying using icons, 2079
Connections
Number of connections (PROFINET/ PROFIBUS), 682
Consistency
Slot rules, 563
Consistency check, 875, 5134, 5135, 5139, 5140
Introduction, 5134, 5139
local, 727
project-wide, 727
Consistency of data, 699
Constant
Data types, 2144
Definition, 2142
Global constant, 2143, 2144, 2146, 4852, 4853, 4854, 4857
Local constant, 2143, 2144, 2146, 4771, 4772, 4784
Non-typed constant, 2142
Symbolic constant, 2143
Typed constant, 2142
Constants, 5010
Context filter, 547
CONTINUE, 3393
Control data record, 1085, 1939, 1981, 2013
Control data record 196, 1934, 1939, 1991, 2013
Control element
Simple alarm view, 5528
Control unit, 1151
Controller alarms
Configuring, 5493
Controller data, 6655
Initializing, 5905, 6657
Control-timer alarm, 2877, 2880, 3202, 3206, 3439, 3441
Conventions for user-defined documentation, 432
Conversion, 2330, 2465, 2531
Explicit, 2382, 2384, 2387, 2390, 2394, 2398, 2402, 2405, 2409, 2412, 2416, 2419, 2423, 2426, 2429, 2432, 2434, 2437, 2440, 2443, 2446, 2449, 2451, 2453, 2456, 2458, 2460, 2463, 2487, 2488, 2490, 2493, 2497, 2500, 2503, 2506, 2509, 2512, 2515, 2518, 2521, 2523, 2524, 2525, 2526, 2527, 2528, 2530, 2545, 2546, 2548, 2550, 2553, 2556, 2559, 2560, 2561, 2563, 2564, 2565, 2566, 2567
Implicit, 2334, 2335, 2338, 2339, 2341, 2342, 2344, 2345, 2347, 2349, 2351, 2353, 2355, 2357, 2358, 2360, 2362, 2364, 2366, 2368, 2370, 2372, 2374, 2376, 2377, 2379, 2380, 2468, 2469, 2470, 2471, 2472, 2473, 2474, 2475, 2476, 2477, 2478, 2479, 2480, 2481, 2482, 2483, 2484, 2485, 2486, 2534, 2535, 2537, 2538, 2539, 2540, 2541, 2542, 2543, 2544
Conversion operations
Converting timers with T_CONV, 3457
Conversions, 94
Convert, 2803, 3128, 3364
HW identifier to slot, 4073
IO address to HW identifier, 4073
Slot to HW identifier, 4073
Convert character string, 91
Converted name (PROFINET), 1774
Converting
A character string to a hexadecimal number with ATH, 3526
Character string in array with Strg_TO_Chars, 3506
Character strings with S_CONV, 3494
Converting number string to number with STRG_VAL, 3497
Converting number to number string with VAL_STRG, 3501
Converting timers with T_CONV, 3457
Copying characters to numerical character string with Chars_TO_Strg, 3509
Hexadecimal number to character string with HTA, 3529
Converting an unspecified CPU, 215
Cookie, 1065
Coordinated transfer, 5573
With PLC, 5573
Copy
Hardware component, 570
Recipe data record in runtime, 5606
Screen, 5260
Tag, 5436
Template, 5266
Copy protection, 4754

Copying
Adjusting screen size, 5886
Color, 5885
Excel format, 5820
Font, 5885
Function key, 5886
Invalid object, 5885
Linked objects, 5887, 5888
Principle, 5884
Screen, 5886
Style sheet, 5281
User-defined folders, 5885
Corporate libraries, 440
COS, 2720, 3045, 3284
Queue,
CoS (Class of Service), 1245, 1539
Cosine, 2720, 3045, 3284
Count
Down, 2647, 2655, 2661, 2973, 2980, 2987, 3259, 3266
Up, 2645, 2653, 2660, 2970, 2979, 2985, 3257, 3264
Up and down, 2650, 2657, 2975, 3261, 3267
Count (8 DI NAMUR), 1917
COUNTER, 2325, 2657, 3267
Counter input, 993
Counter mode, 993
Counter, high-speed, 992
Counters
Control high-speed counter (extended), 4196
Control high-speed counters, 4194
High-speed counters, 4194, 4196
CountOfElements, 2786, 3111, 3352
Cover page
Placeholder for document information, 393
CP 1613, 69
CP 1623, 69
CP 343-2, 77
CP 5512, 69
C-PLUG, 1645
Formatting, 1378, 1647
Saving the configuration, 1647
CPU
Displaying the current LED status, 2041
Fill level of all types of memory, 2041, 2042
Inserting a signal board, 1050
Number of communication connections, 682
Properties, 980
Reading date and time-of-day with RD_SYS_T, 3469
Reading out a diagnostics buffer, 2046
Selecting from the hardware catalog, 559
Setting the time-of-day time-of-day with WR_SYS_T, 3467
Switching operating mode, 2051
CPU control panel
Display area, 2041
CPU data block
Definition, 2101
Deleting, 4738
CPU firmware, 1043
CPU load through trace, 6559
CPU memory area
Displaying, 5144
CPU properties, 982
Create
Global library, 5803
Group, 5318
Template, 5265
Users, 5622
Users in runtime, 5630
CREATE_DB, 4045
Creating, 5585
Alarm group, 5496
Alarm log, 5515, 5518
Array, 5466
Data log, 5471
External tag, 5430
Infotext, 5497, 5500
Internal tag, 5433
Network connection, 6676
Recipe, 5585
Recipe data record on the HMI device, 5599, 5600, 5605
Screen, 5259
Style sheet, 5278
User group, 5621, 5644
Users, 5646
Creating a CHM file, 438
Creating a homepage for user-defined documentation, 431
Creating a new type
Style, 5818
Style sheet, 5818
Creating a print preview, 394
Creating a route, 870
Creating a type
Screen, 5817
Script, 5816
Creating a watch table, 5200
Creating custom documentation, 300
Creating labeling strips, 399
Creating user-defined documentation, 436
CRL, 938

Crossing
Definition, 4919
Deleting, 4921
Inserting, 4920
rearranging, 4920
Cross-reference list, (See cross-references)
Adding new objects, 5169
Benefits, 5146
Displaying, 5153
Introduction, 5146
Overview, 5153
Settings, 5149
Sorting columns, 5147
Source object, 5147
Structure, 5147
Uses, 5146
Cross-references
Add new object, 5169
Displayed objects, 5146
Displaying, 5153
Introduction, 5844
Linking, 5845
Replace, 5845
Uses, 5844
CRT, 938
CSV file, 5820, 5822, 5831
CTD, 2647, 2973, 3259
CTRL
Key, 5941
CTRL_HSC, 4194
CTRL_HSC_EXT, 4196
CTRL_PTO, 3993
CTRL_PWM, 3991
CTU, 2645, 2970, 3257
CTUD, 2650, 2975, 3261
CU, 2660, 2985
Cu10 sensor, 1965
Cursor key, 5939
Custom alarm classes, 5488
Cycle load, 989
Cycle monitoring time, 2835, 3159, 3405
Cycle time, 982, 988
display configured, 2034
display measured, 2040
Cyclic
Continuous, 5447
In operation, 5447
Cyclic interrupt, 1029
Cyclic interrupt OB
Assigning parameters, 1040
Description, 1029
Querying parameters with QRY_CINT, 3807
Setting parameters with SET_CINT, 3805
Cyclic program execution
Options for interrupting, 1022
Programming, 1022

D

D_ACT_DP, 3673
Data backup, 142
HMI device, 5951
Data bit (DBX), 967
Data block
Adjusting data values during commissioning, 4830, 4831, 4832, 4833, 4835
ARRAY data block, 2091, 2102, 2150, 2153, 4716, 4795, 4798, 4831
Based on a PLC data type, 4806
CPU data block, 2101
Creating, 4716, 4798
Creating with CREATE_DB, 4045
Declaration table, 4797
Declaring ARRAY, 4803
Declaring STRUCT, 4804
Default value, 4807
Deleting with DELETE_DB, 4063
Downloading changes without reinitialization, 4801
Global data block, 2091, 4795
Importing and exporting tags, 4820
Instance data block, 2100, 4795
Monitoring and modifying tags, 4822, 4823, 4824, 4825, 4826, 4828
Optimized access, 2102, 2104
Programming, 4795, 4802
Reading attributes with ATTR_DB, 4059
Reading from load memory with READ_DBL, 4050
Retentive behavior, 4809, 4810
Snapshot, 4825, 4826, 4828
Start value, 4807, 4808
Tag properties, 4811, 4815, 4816
Updating, 4800
Using setpoints, 4830, 4831, 4832, 4833, 4835
Writing to load memory with WRIT_DBL, 4054
Data block 196, 1934, 1991, 2013
Data block 197, 1934, 1991, 2013
Data byte (DBB), 967
Data consistency, 699, 4383
Data double word (DBD), 967
Data Encryption Standard (DES), 810

Data exchange
DP slave, 1737
I-slave - DP master, 1738
Data exchange over the AS-AS remote link
Communication instruction "AS_DIAL", 6621
Data flow control, 1000
Data log, 1055, 5561
Closing with DataLogClose, 4031
Creating, 5471
Creating with DataLogCreate, 4016
Creating with DataLogNewFile, 4034
Creating with DataLogTypedNewFile, 4036
Deleting with DataLogDelete, 4033
Empty with DataLogClear, 4027
Naming conventions, 5564
Opening with DataLogOpen, 4023
Opening with DataLogTypedOpen, 4025
Tags, 5470, 5475
Tolerance band, 5475
Writing with DataLogWrite, 4029
Data logging, 5469
Application, 5469
Data mailbox
For recipes, 5573
Data point configuration, 1171
Data point type, 1177
Data record
Asynchronously reading data record of a module with RD_DPARA, 3789
Exporting, 5582
Importing, 5582
Making available on I-device with PRVREC, 3706
read, 5608
Reading, 5600
Reading from configured system data with RD_DPARM, 3791
Reading of a module with RD_DPAR, 3783
Reading with RD_REC, 3694
Reading with RDREC, 3578
Receiving on I-device with RCVREC, 3704
Transfer, 5601, 5608
Transferring with WRREC, 3596
Writing and reading data records, 3782
Writing predefined parameters with WR_DPARM, 3793
Writing with WR_REC, 3699
Data record list, 5574
Data type
ANY, 2305
ARRAY, 2286
BOOL, 2243, 2334, 2382, 2468, 2487, 2534, 2545
BYTE, 2244, 2335, 2384, 2469, 2488, 2535, 2546
CHAR, 2269, 2376, 2456, 2483, 2484, 2526, 2527, 2544, 2565
Conversion, 2330, 2338, 2465, 2471, 2480, 2481, 2482, 2531, 2537, 2540, 2542, 2543
DATE, 2264, 2370, 2449, 2481, 2523, 2542, 2562
DINT, 2252, 2347, 2412, 2476, 2509, 2539, 2556
DT, 2265, 2364, 2440, 2543, 2564
DTL, 2267, 2368, 2446, 2483, 2525
DWORD, 2245, 2338, 2390, 2471, 2493, 2537, 2550
Explicit conversion, 2382, 2384, 2387, 2390, 2394, 2398, 2402, 2405, 2409, 2412, 2416, 2419, 2423, 2426, 2429, 2432, 2434, 2437, 2440, 2443, 2446, 2449, 2451, 2453, 2456, 2458, 2460, 2463, 2487, 2488, 2490, 2493, 2497, 2500, 2503, 2506, 2509, 2512, 2515, 2518, 2521, 2523, 2524, 2525, 2526, 2527, 2528, 2530, 2545, 2546, 2548, 2550, 2553, 2556, 2559, 2560, 2561, 2562, 2563, 2564, 2565, 2566, 2803, 3128, 3364
Implicit conversion, 2334, 2335, 2336, 2339, 2341, 2342, 2344, 2345, 2347, 2349, 2351, 2353, 2355, 2357, 2358, 2360, 2362, 2364, 2366, 2368, 2370, 2372, 2374, 2376, 2377, 2379, 2380, 2468, 2469, 2470, 2472, 2473, 2474, 2475, 2476, 2477, 2478, 2479, 2483, 2484, 2485, 2486, 2534, 2535, 2536, 2538, 2539, 2541, 2543, 2544
INT, 2250, 2344, 2405, 2474, 2503, 2538, 2553
Internal tag, 5428
LDT, 2266, 2366, 2443
LINT, 2254, 2351, 2419
LREAL, 2258, 2357, 2429, 2479, 2518
LTIME, 2263, 2362, 2437
LTIME_OF_DAY, 2265
LTOD, 2374, 2453
LWORD, 2246, 2339, 2394
PLC data type, 2277
POINTER, 2302
REAL, 2257, 2355, 2426, 2478, 2515, 2539, 2559
S5TIME, 2261, 2358, 2432, 2541, 2561
SCL instruction, 5034, 5035
SINT, 2248, 2341, 2398, 2472, 2497
STRING, 2270, 2379, 2460, 2485, 2486, 2528, 2530, 2544, 2566
STRUCT, 2282
TIME, 2262, 2360, 2434, 2480, 2521, 2540, 2560
TIME_OF_DAY, 2264
TOD, 2372, 2451, 2482, 2524, 2543, 2563
UDINT, 2253, 2349, 2416, 2477, 2512

UINT, 2251, 2345, 2409, 2475, 2506
ULINT, 2255, 2353, 2423
USINT, 2249, 2342, 2402, 2473, 2500
Validity, 2239
VARIANT, 2308
WCHAR, 2269, 2377, 2458
WORD, 2245, 2336, 2387, 2470, 2490, 2536, 2548
WSTRING, 2273, 2380, 2463
Data types, 2211
Migration, 204
Data word (DBW), 967
DataLogClear, 4027
DataLogClose, 4031
DataLogCreate, 4016
DataLogDelete, 4033
DataLogNewFile, 4034
DataLogOpen, 4023
DataLogTypedNewFile, 4036
DataLogTypedOpen, 4025
DataLogWrite, 4029
Date, 2264, 2265, 2266, 2267
Date/time field, 5388
Display system time, 5388
Format, 5389
Using tags, 5389
DATE_AND_LTIME, 2266
DATE_AND_TIME, 2265
DATE_TO_, 2449, 2523, 2562
DB, 4716, 4798
DB variable
Definition, 2140
DB_ANY_TO_VARIANT, 3378
DCAT, 2877, 3202, 3439
DCP server, 1327, 1599
Deactivate, 94, 5772
Dead peer detection (DPD), 821
Deadband
Setting, 5500
DEC, 2706, 3032
Decimal places, 67, 2727, 3052, 3288
DECO, 2853, 3177, 3415
Decode, 2853, 3177, 3415
DecreaseFocusedValue, 5680
DecreaseTag, 5697
Decrement, 2706, 3032
Default font, 5869, 5908
Default routes
IPv6 routes, 1605
Default style, 5907
Defining, 5273
Default template, 5907
Defective devices, 2023
Defective license, 131, 143
Define
Reference object, 5292
Define columns
Alarm view, 5404
Define display area
Alarm view, 5405
Define hotkey, 5415
Defining, 5522
For the assignment list, 5126
DEL key, 5940
Delay time
Setting, 5499
Delete,
Template, 5266
DELETE_DB, 4063
Deleting
Authorization, 5627
Filter in the assignment list, 5126
Hardware component, 575
Object, 5288
Recipe data record in runtime, 5606
Screen, 5260
Style sheet, 5280
Tag, 5436
User group, 5626
Users, 5625
Deleting a connection, 220
Deleting CA certificates, 6612
Demultiplex, 2860, 3185, 3422
DEMUX, 92, 2860, 3185, 3422
Dependencies of rights, 741
Dependency on HMI device
Logging, 5563
Dependency structure, 5135
Displaying,
Introduction, 5135
Meaning of symbols, 5137
Setting view options, 5139
Structure, 5137
DER, 938
DES, 802, 810
Deserialize, 2735, 3060, 3290
Design analysis
Bit selection, 5907
Designation
TS Adapter, 6584, 6592, 6600
Designing a background, 5298
Designing a border, 5298
Destruction of license keys, 130
DETACH, 3799

Detailed comparison
Performing, 371
Starting, 5090, 5093
Visualization of the comparison result for GRAPH, 5105
Visualization of the comparison result for LAD/FBD, 5095
Visualization of the comparison result for SCL, 5102
Visualization of the comparison result for STL, 5099
Details view, 260
Determine diagnostics status, 2043
Determining time of day, 2052
Device
Adding to a hardware configuration, 556
Copying, 570
Defective devices, 2023
Deleting, 575
Moving, 572
Multiple selection, 567
Rename, 573
Renaming, 716
Device address, 1045
Device comparison, 583
Device information, 384, 2023
Device name, 1767, 1774
Device name, automatic commissioning (PROFINET), 1770
Device number, 1769
Device overview
Address range, 1045
Device proxy
Initializing, 6662
Updating, 6660, 6665
Device replacement, 572
Device Tool, 1813, 1814
Device version, 5918
Checking, 5897
Switching, 5896
Updating, 5953
Device view
Area of unplugged modules, 535
Edit parameters, 576
Edit properties, 576
Hardware and network editor, 530
Inserting a signal board, 1049
Inserting module, 564
Racks, 561
Device wizard, 5900
Device-dependent
Font size, 5275
DeviceStates, 3969
DHCP
Client, 1345, 1622
Server configuration, 873
DHCP server, 874
DHCPv4 Relay Agent
General settings, 1482
DHCPv6
Client, 1476
PD Sub Client, 1477
Prefix delegation, 1476, 1477
Diagnose repeater, 2075
Diagnostic error interrupt, 1033
Diagnostic interrupt OB, 1033
Diagnostics, 825, 1928
DP slave, reading diagnostic data with DPNRM_DG, 3715
Online & diagnostics in TIA Portal, 1107
PC System V2.0, 1107
PROFINET IO components, 2045
Diagnostics buffer, 1012
Basics, 971, 2070
Organization, 2070
Diagnostics data
Reading out with GET_DIAG, 3982
Diagnostics status
Determining and displaying online, 2024
Diagnostics user, 736
Diagonal movement
Animation, 5353
Dialing rules in phone books, 6581
Dial-up connection
Establishing, 6605
Terminating, 6606
Diffie-Hellman key exchange, 809
Digital certificates, 920
DINT, 2252, 2347, 2412, 2476, 2509, 2539, 2556
DINT_TO_, 2412, 2509, 2556
Direct access to the I/O, 971
Direct connection
Configuring, 6669
Direction output and travel direction relation, 6147
DirectKey
Configuring, 5370
Directories for user-defined documentation, 433
DIS_AIRT, 3869
DIS_IRT, 3861
Disabling
Project language, 5854
Discrete alarm, 5485, 5486
Configuring, 5496, 5505

Discrete alarm types, 5485
Discrete alarms
Configuring, 5493
display
HMI backup,
Infotext, 5941, 5945
Program name,
Display number
Recipe view, 5412
Display object
Availability for Basic Panels, 5382
Display PDF files, 126
Display system time, 5388
Display Welcome Tour, 126
Displaying
Assignment list,
Call structure, 5133
CPU memory area, 5144
Cross-references, 5153
Dependency structure,
Limit lines on the bar, 5384
Load memory, 5145
Log contents, 5517
Maximum load memory available, 5145
Program information, 5120
Displaying a force table, 5229
Displaying memory types of a CPU, 2041, 2042
Displaying monitor values for structures in the Inspector window, 5184
Displaying or hiding tag information, 4915, 4972, 5036
Displaying support for a module, 549
Displaying the call log, 436
Displaying the monitor values for structures in the Inspector window, 5184
Displaying the online status, 6483
Displaying types for an instance, 487
Displaying values
As a trend, 5480
Distribute
Objects evenly, 5294
Distributed I/O, 1735, 1926, 1947, 1978
DIV, 2702, 3027
Divide, 2702, 3027
DMSK_FLT, 3847
DNS client, 1332
DO, 3388, 3391
Documentation editor
Layout, 389
Documentation settings
Using frames and cover pages, 387
Double word, 2245
double-click, 5772
Download, (See Loading to device)
Error messages, 5956
Project, 5920
Downloading
Blocks to the device, 5070
Downloading blocks to a memory card, 5079
Downloading blocks to device, 5075, 5076
Downloading to a memory card, 352
in RUN operating mode, 5070
Uploading blocks from a memory card, 5081
Writing to a memory card, 352
DP interface, 1739
DP master, 1912
Add DP master system, 1742
Devices and modules, 1738
Disconnection from DP master system, 1742
Display on DP slave, 1740
DP interface, 1739
DP master system
Add DP slave, 1743
Creating, 1735, 1738, 1745
Determining topology with DP_TOPOL, 3718
Disconnect DP slave, 1744
Disconnection from subnet, 1741
Edit properties, 1742
Highlight, 1741
Node disconnection, 1741
DP slave
Activating and deactivating with D_ACT_DP, 3673
Add to DP master system, 1743
Assign DP master system, 1744
Configuring, 1737
Data exchange, 1737
Disconnect from DP master system, 1744
DP master display, 1740
Hardware catalog, 1736
Intelligent, (See I-slave)
Networking, 1745
Reading diagnostic data with DPNRM_DG, 3715
Slot rules, 1743
Synchronizing groups with DP_SYC_FR, 3709
Types, 1743
With preprocessing, (See I-slave)
DP standard slave,
DP standard slaves
Reading a portion of the inputs with GETIO_PART, 3624
Reading all inputs with GETIO, 3615
Reading consistent data with DPRD_DAT, 3700

Writing a part of the outputs with SETIO_PART, 3626
Writing all outputs with SETIO, 3616
Writing consistent data with DPWR_DAT, 3702
DP/DP coupler, 1736
DP_TOPOL, 3718
DPNRM_DG, 3715
DPRD_DAT, 3700
DPSYC_FR, 3709
DPV1
Configuring an ET 200S, 1756
DPWR_DAT, 3702
Drive connection S7-1200 Motion Control, 6151
DRUM, 2871, 3196, 3432
DSCP, 1390
DST
Daylight saving time, 1356, 1631, 1632
DT, 2265, 2364, 2440, 2543, 2564
DT_TO_, 2440, 2564
DTL, 2267, 2368, 2446, 2483, 2525
DTL_TO_, 2446, 2525
Duration, 2261
DWORD, 2245, 2338, 2390, 2471, 2493, 2537, 2550
DWORD_TO_, 2390, 2493, 2550
Dynamic reference temperature, 1998
Dynamization
Color of an object, 5350
Flashing, 5350
Object, 5347
Dynamize
Appearance of an object, 5350
Direct movement, 5353
Green arrow in Overview, 5349
Movement of an object, 5352

E

Eastern characters
Input on the HMI device, 5872
EB, 4841
EC31-HMI/RTX
Visualization, 1144
EC31-RTX, 77
ED, 4841
Edge
Negative, 2588, 2591, 2593, 2913, 2916, 2918, 3220
Positive, 2586, 2589, 2592, 2911, 2914, 2917, 3219
Edit
Object within a group, 5321
Edit networking
Disconnecting from a network, 593
EditAlarm, 5718
Editing, (Tags)
Folder link, 5314
Function list, 5654
Style sheet, 5280
System event, 5502
Editing a version of a type in testing, 489
Editing language, 407, 5851
Selecting, 5855
Editing networking
Copying a subnet, 593
Copying subnets and devices, 594
Editor
Graphics, 5862
Element list, 5574
Ellipse, 5391
Horizontal radius, 5392
Vertical radius, 5392
ELSE, 3385, 3386
ELSIF, 3385
E-mail
Transferring e-mail with TM_MAIL, 4676
Transferring e-mail with TMAIL_C, 4453
Embedded controller, 1145
Load:Download, 1145
Empty box
Inserting a LAD element, 4893
Inserting an FBD element, 4950
EN/ENO mechanism
Basics, 2177
FBD example, 2186
LAD example, 2182
SCL example, 2194
STL example, 2189
EN_AIRT, 3870
EN_IRT, 3863
Enable autonegation, 1796
Enabling the firewall
SCALANCE S, 861, 863
Enabling tunneled communication
CP 343-1 Adv. / 443-1 Adv., 893
SCALANCE S, 861, 863
ENCO, 2854, 3179, 3417
Encode, 2854, 3179, 3417
Encoder connection S7-1200 Motion Control, 6151
Encryption, 722
Encryption software, 134, 138
END key, 5940
End of detection of accessible devices, 1798
End of sync domain, 1798

End of topology discovery, 1798
END_CASE, 3386
END_FOR, 3388
END_IF, 3385
END_REPEAT, 3392
END_WHILE, 3391
End-entity certificate, 923
Endian, 3341, 3343, 3345, 3347
ENDIS_PW, 2829, 3153, 3398
Enhanced Write Filter (EWF), 1145
ENO, 94
ENTER key, 5940
Enumeration types, 1067
Enumerations
  Web server, 1069
EQ_ElemType, 2688, 3013
EQ_Type, 2684, 3009
Equal, 2692, 3016
EQUAL ARRAY, 2694, 3019, 3271
Error analysis with RET_VAL, 2198
Error handling
  GetError, 2837, 3161, 3406
  GetErrorID, 2840, 3164, 3409
Error information, 2198, 2837, 2840, 3161, 3164, 3406, 3409
Error messages
  Download, 5956
Error status, 1284, 1563
ESC key, 5940
ESP protocol, 810
Establishing a connection from a remote system ("PG_DIAL"), 6620
Establishing a remote connection, 6605
  Procedure, 6605
ET 200AL, 1909, 1939
ET 200eco, 1909
ET 200eco PN, 1909, 1926
ET 200iSP, 1908
ET 200iSP distributed I/O station
  Definition, 1912
ET 200L, 1908
ET 200M, 1908, 2003
  Definition, 2002
ET 200MP, 1908, 1989
  Pulse width modulation, 2000
ET 200pro, 1909
ET 200R, 1909
ET 200S, 1908
  DPV1 mode, 1756
  Option handling, 1753, 1755
  Reference junctions, 1749
  Slot rules, 1748
ET 200S COMPACT, 1908
ET 200SP, 1908, 1932, 1939
  Application area, 1932
ET 200AL, 1947, 1978
  Area of application, 1975
  Configuration example, 1976
ET 200M distributed IO device, 2002
ET 200SP, 1947
ET 200SP with ET 200AL mixed mode, 1947, 1978
ET-Connection, 1947
ET-Connection rack, 1947
Ethernet, 601, 603
  Number of communication connections, 682
Ethernet interface
  Displaying parameters, 6500
Ethernet interfaces
  EC31-HMI/RTX, 1143
  EC31-RTX, 1143
  EC31-RTX F, 1143
Ethernet module
  Removal/insertion, 75
Ethernet non IP frames, 747
Ethernet Statistics
  Frame length, 1294, 1568
  Frame type, 1295, 1569
  Interface statistics, 1294, 1567
  Packet Errors, 1296, 1297, 1570
Evaluation unit, 1152
Event
  Acknowledge, 5776
  activate, 5769
  Alarm buffer overflow, 5776
  Assigning organization block (OB) with ATTACH, 3795
  Boundary reached, 5776
  Canceling assignment to organization block (OB) with DETACH, 3799
  Change, 5769
  cleared, 5769
  Click, 5774
  Click when flashing, 5775
  Configuring, 5576, 5581
  Deactivate, 5772
  double-click, 5772
  Execute, 5770
  For function lists, 5452
  Free space critically low, 5779
  Incoming, 5774
  Input finished, 5773
  Loop-In-Alarm, 5775
  Low free storage space, 5778
  On exceeding, 5770

On falling below, 5771
Outgoing, 5774
Overflow, 5477, 5778
Press, 5773
Press ESC twice, 5773
Press key, 5777
release, 5775
Release key, 5777
Runtime Stop, 5777
Screen change, 5772
Selection changed, 5770
Switch OFF, 5778
Switch ON, 5778
Tags, 5452
Time expired, 5779
User change, 5771
Value change, 5779
When dialog is closed, 5771
When dialog is opened, 5771
Event name, 1041
Event trigger, 5784, 5787
Event-driven tasks
Configuring:event-driven tasks, 5498
Events
Configuration, 1340
Log table, 1282, 1561
Severity filter, 1343, 1620
EW, 4841
EX, 4841
Example
Application for alarm classes, 5488
Calculating an equation, 4931, 4986
Changing and displaying operating mode, 5655
Configuration ET 200AL, 1976
Controlling a conveyor belt, 4924, 5053
Controlling room temperature, 4933, 4988
Detecting the direction of a conveyor belt, 4926, 4980, 5055
Detecting the fill level of a storage area, 4928, 4982, 5057
for displaying bit memory in the assignment list, 5122
for displaying inputs and outputs in the assignment list, 5122
For entering force values in the force table, 5232
Modify values in the watch table, 5204
System alarm, 5485
Example of homepage for user-defined documentation, 432
Example of user-defined documentation, 437
Example:
Discrete alarm, 5486
Examples
Controlling a conveyor belt, 4978
Exchangeable medium, 1798
Exchanging, 6655
EXCLUSIVE OR, 2898, 2899
Bit logic operation, 2898
Execute, 5770
Exit,
User program, 2836, 3160, 3405
EXP, 2718, 3043, 3282
Expand ET 200SP, 1947
Expanded mode, 5197
Expanding and collapsing sections of code, 5018
Expanding ET 200SP with ET 200AL modules, 1947
Exponential value, 2718, 3043
Exponentiate, 2728, 3053
Export
Alarm, 5824
Excel format, 5820
Project texts, 5841, 5859
Recipe, 5725, 5820
Tag, 5831
Text list, 5837
Export of labeling strips, 399
ExportDataRecords, 5723
Exporting
Recipe, 5582
Recipe data record, 5582
Exporting an NTP server, 800
Exporting labeling data as XML, 402
Expression
Arithmetic expression, 4993
Basics, 4992
Logical expression, 5000
Relational expressions, 4995
EXPT, 2728, 3053
Extended status information, 5013
External graphic
Edit folder, 5313
Link folder, 5314
Removing the folder link, 5313
Rename folder, 5313
External image file
add to graphics library, 5864
Managing, 5286
Storing in the image browser, 5315
External source file
Basics, 4738
Editing, 4744
Exporting blocks, 4741
inserting, 4743
Linking file types to an editor, 4744

Opening, 4744
Rules for programming, 4740

F

F_TRIG, 2595, 2920, 3220
Facility, 783
Factory settings
Resetting to, 2053, 2054, 2055, 5954
Fail-safe modules
Selecting from the hardware catalog, 561
FALSE, 2594, 2595, 2919, 2920, 3219, 3220
Fault monitoring
Connection status change, 1373, 1641
Power supply, 1372, 1640
Redundancy, 1373
Favorites
Adding, 4897, 4954, 5037
Hiding, 4694
Removing, 4898, 4955, 5038
Showing, 4694
Using, 4897, 4954, 5038
FB, 2090
FBD, 4937, 4978, 4980, 4982
Detailed comparison, 5095
FBD element
Copying, 4964
Cutting, 4965
Deleting, 4969
Inserting, 4949, 4950
Inserting an operand, 4970
pasting from the clipboard, 4966
Replacing, 4966
Rules for inserting, 4948
Selecting, 4964
FBD network
Branch, 4973
Deleting a branch, 4975
Inserting a block call, 4899, 4956
Inserting a branch, 4974
Program status display, 5188
Rules for branches, 4974
FC, 2089
F-CM AS-i Safety ST, 77
F-CPU, 561
Feedback data record, 1939, 1981
FETCH/WRITE, 1012
Field device, 1761, 1808, 1809
FieldRead, 2790, 3115
FieldWrite, 2792, 3117
FIFO, 2320
FIFOQueue, 2320
File Browser, 1055
File format
CER, 938
CRL, 938
CRT, 938
DER, 938
P12, 938
PEM, 938
Fill,
Block, 2764, 2799, 3089, 3124, 3315, 3360
Block uninterruptible, 2767, 3092, 3319
FILL_BLK, 2764, 3089, 3315
Filter
Defining for the assignment list, 5126
Delete, 5126
Hardware catalog, 547
In the assignment list, 5125
Selecting, 5127
Filter alarms
Alarm view, 5404
Filters
Assignment list,
FIND, 3546
Find and replace
Colors, 5902
Firewall
Creating service groups, 766, 769
Defining ICMP services, 765
Firewall rules, 747
Managing service groups, 767, 769
Firmware, 1043
Updating, 5953
Firmware card, 964
Firmware update
Basics, 2057
from the project context, 2059
from the project tree, 2058
Firmware update memory card, 965
Firmware versions, 70
Fixed reference temperature, 1998
Flash test, 2074
Flashing, 5350
Flip
Object, 5286
Flip-flop
Reset/set, 2585, 2910
Set/reset, 2584, 2908
Floating-point number, 2257, 2258
Check invalidity, 2683, 3008
Check validity, 2682, 3007
Floating-point numbers, 2259
Invalid, 2259

FLOOR, 2807, 3133, 3368
Flutter monitoring, 2005
Folder link
  Editing, 5314
  Removing, 5314
  Renaming, 5314
Following error monitoring, 6219
Font settings
  Migrate, 189
Font size
  Asian languages, 5872
  Device-dependent, 5275
Font size adjustment
  Activating, 5274
FOR, 3388
Force
  Force all, 5248
  Stop forcing, 5249, 5250
Force job on SD card, 68
Force table
  "Monitor once and now" command for tags, 5240
  Basic mode, 5227
  Changed display formats, 85
  Display, 5229
  Expanded mode, 5227
  Functionality, 5224
  Layout, 5226
  Meaning of the columns, 5226
  Meaning of the icons, 5228
  Monitoring and modifying modes, 5212
  Opening, 5229
  Overview of the display formats, 5234
  Overview of the test options, 5224
  Permitted operands, 5231
  Permitted operands for force values, 5232
  Saving, 5230
  Switching between basic mode and expanded mode, 5227
  Syntax check, 5230
  Test options, 5224
Force value
  Permitted operands, 5232
Forcing tags
  Safety precautions, 5225, 5243
Foreground color
  Dynamization, 5350
Form exponential value, 3282
Format, 5389
Formatting
  Alarm text, 5502
FRAC, 2727, 3052, 3288
Fragment, 1070, 1076
Free space critically low, 5779
Free-form comments
  Deleting, 4906, 4963
  Editing, 4905, 4962
  Inserting, 4905, 4962
  Introduction, 4904, 4961
Frequency meter, 1923, 1926
FTP, 738, 740, 1012
FTPS certificates, 729
Full-screen mode, 5907
Function
  Assigning to a function key, 5364
Function (FC)
  Creating, 4716
  Definition, 2089
  exporting to an external source file, 4741
Function block (FB)
  Creating, 4716
  Definition, 2090
  exporting to an external source file, 4741
  Instance data block, 2090
Function Block Diagram, 4937
Function key, 5359
  Assigning a function, 5364
  Assigning a graphic, 5368
  Global assignment, 5361, 5942
  Global screen, 5264
  Local assignment, 5264, 5363, 5942
  protect with password, 5366
  Use global assignment, 5264
  used for screen navigation, 5372
Function keys, 5907
Function list, 5452, 5651, 5783
  Asynchronous execution, 5655
  Execution, 5652
  Execution in Runtime, 5655
  Synchronous execution, 5655
  System function, 5652
  User-defined functions, 5652
Function lists
  Editing, 5654
Function value, 92
Functional scope
  ProSave, 5950
Functionality of the force table, 5224
Functions
  Updating tag value, 5425

G

GADR_LGC, 4091
Gateway, 601

GBIT interface (CPU 1518-4 PN/DP), 1796
GEN_DIAG, 3979
Gen_UsrMsg, 3892
General information on troubleshooting, 6627
Generating diagnostics information, 3979
Generation of packed addresses, 1751
GEO_LOG, 4086
GEO2LOG, 4074
Geographic coordinates, 1330, 1603
GET, 4388
Get_AlarmState, 3888
GET_DIAG, 3982
Get_IM_Data, 3947
GET_NAME, 3953
GetBlockName, 3560
GetChecksum, 3963
GetDataRecordFromPLC, 5728
GetDataRecordTagsFromPLC, 5730
GetError, 2837, 3161, 3406
GetErrorID, 2840, 3164, 3409
GetGroupNumber, 5689
GetInstanceName, 3554
GetInstancePath, 3556
GETIO, 3615
GETIO_PART, 3624
GetPassword, 5690
GetPLCMode, 5711
GetSMCinfo, 3966
GetStationInfo, 3957
GetSymbolName, 3547
GetSymbolPath, 3550
GetUserName, 5689
Gigabit PROFINET interface, 1796
Global assignment
  Of a function key, 5359, 5361
Global data block, 2091
  Retentive behavior, 4810
Global firewall rules, 748
  Assigning, 750
Global libraries, 440
  Archiving, 466
  Backward compatibility with older product versions, 459
  Corporate libraries, 440
  Retrieving, 467
  System libraries, 440
  User libraries, 440
Global library, 5793
  Adding types, 496
  Archiving, 467
  Cleaning up, 519
  Closing, 464
  Create, 5803
  Creating, 458
  Creating folders, 515
  Deleting, 465
  Displaying logs, 463, 5805
  Migrating, 196
  Open, 5805
  Opening, 460
  Save, 5804
  Saving, 463
  Showing properties, 462
  Updating the project, 499
  Upgrade, 5895
  Using filter view, 474
  Using types, 498
Global packet filter rules, 750
Global screen, 5264
  Function key, 5264
  Template, 5265
Global softkey, 5361
Global tag, (See PLC tag)
GMP, 5908
GMRP, 1435
Go online, 2072, 6482
  connecting several devices, 6485, 6486
  Multiple TIA Portal instances, 75
Going offline, 2074
GOTO, 3395
GoToEnd, 5749
GoToHome, 5748
GRAPH, 2196
  Detailed comparison, 5105
  Program status display, 5192
Graphic
  Adapting, 5393
  Add to graphics library, 5862
  Adjust, 5395
  Alignment, 5304
  Assigning to a function key, 5368
  Button, 5415
  Distance, 5304
  Graphic view, 5392
  Inserting, 5286
  managing, 5313
  Size adjustment, 5304
  Using from the graphic browser, 5312
  With transparent background, 5313
Graphic browser, 5312, 5864, 5907
Graphic I/O field, 5394
  Output graphics list, 5346
Graphic lists
  Bit selection, 5907

Graphic view, 5392
transparent color, 5393
Graphics, 5315
Editor, 5862
Graphics list
Application, 5338
Bit (0, 1), 5343
Bit number (0 - 31),
Creating, 5339
Graphic I/O field, 5346
Outputting configuration data, 5345
Range (0 - 31),
Range (... - ...), 5342
group
Adding objects,
Cancel, 5319
Creating,
Edit, 5318
Editing,
Removing an object,
Ungroup, 5319
Group alarm, 1013
Group name, 764, 767
Group properties, 808
Groups, 1498, 1703
GSD files
Configuring devices (PROFIBUS), 1761
Delete, 1759
Delete unused GSD data, 1760, 1810
GSD revisions (PROFIBUS), 1758
Installation, 1758
GSD files (PROFINET), 1806
Changing revision, 1811
Delete, 1809
Installation, 1808
GSDML, (See GSD files (PROFINET))

H

Handshake Protocol, 919
Handshaking, 1000
Hardware
Configuring and assigning parameters, 555
Detection, 78
Edit parameters, 576
Edit properties, 576
Hardware and network editor
Device view, 530
Network view, 528
Topology view, 537
Hardware and software limit switches
Function, 6164
Hardware catalog
Adding device, 557
Browsing, 547
DP slave, 1736
DP/DP coupler, 1736
I slave, 1736
Selecting fail-safe modules, 561
Selecting the hardware component, 559
Selection of the hardware components, 1104
Task card, 545
Hardware configuration
Adding device, 556
Adding module, 564
Overview of the settings, 527
Hardware configuration for Motion Control
S7-1200, 6141
Hardware data types, 2328
Hardware detection, 558
Hardware diagnostics, 2022
Hardware editor
Components, 526
Function, 525
Hardware catalog, 545
Inspector window, 539
Hardware identifier, 1045, 1936, 1986, 1993, 2017, 4073, 4853
Determining with LOG2MOD, 4079
Hardware interrupt, 1030
How it works, 978
Hardware interrupt OB
Description, 1030
Parameter assignment, 1041
Hardware requirements, 131, 132
Communication instruction "AS_MAIL", 6625
Communication instruction "PG_DIAL", 6620
Communication instruction "SMS_SEND", 6623
Hardware requirements for AS-AS remote link
Communication instruction "AS_DIAL", 6622
Hardware support package
Installing, 157
Hardware version, 1280, 1559
Hardware-controlled data flow control, 1000
Harmonizing
Object size, 5291
HART, 1914
HART variables
Configuring, 1911
Structure, 1911
Help
Browsing, 292
Call topic from favorites, 296
Components of the information system, 283

Delete topic from favorites, 296
Full-text search, 292
Help on messages, 287
Identification of help topics, 285
Navigation view, 290
Open tooltip cascades automatically, 299
Opening, 288, 289
Preparing your own help contents, 426
Printing help topics, 298
Roll-out, 286
Save topics in favorites, 295
Settings for user-defined documentation, 429, 430
Tooltip, 286
Help indicator, 5395
Hidden input, 5390
Hidden parameters, 4914, 4971
Hierarchy levels
User rights, 5908
High-speed analog modules
Isochronous mode, 1963
High-speed counter
Configuring, 995
General, 992
How it works, 992
HMI backup
Deleting, 5899
Display, 5898
Rename, 5899
HMI connection, 216, 643, 1012
HMI device, 140
Changing the device type, 5957
Configuration, 5896
Data backup, 5949, 5951, 5952
Image, 5949
Load, 5949
Performance features, 5962, 5966
Reloading the operating system, 5954
Reset to factory settings, 5955
Restoring data, 5951, 5952
Software, 5949
Style, 5907
System limits, 5962, 5966
HMI device configuration, 5896
HMI device licensing
Non-PC-based, 140
HMI device signature, 5948
HMI device type
Changing, 5957
HMI device version, 5896, 5897, (Device version)
HMI device wizard, 5900
HMI style, 5818
HMI style sheet, 5818
HMI tags
Connect PLC array tags, 5466
HOME key, 5940
Homing
Homing modes, 6166
Horizontal radius, 5392
How the frequency meter works, 1923
HSC, 992
HSP, (See support package)
HTA, 3529
HTMLBrowerZoomIn, 5754
HTMLBrowserBack, 5758
HTMLBrowserForward, 5757
HTMLBrowserHome, 5757
HTMLBrowserPageDown, 5756
HTMLBrowserPageUp, 5757
HTMLBrowserRefresh, 5754
HTMLBrowserScrollDown, 5753
HTMLBrowserScrollLeft, 5755
HTMLBrowserScrollRight, 5756
HTMLBrowserScrollUp, 5754
HTMLBrowserStop, 5753
HTMLBrowserZoomOut, 5755
HTTP, 765
Load/save, 1335, 1612
HTTPS, 862, 1054
HTTPS server only, 1326
HW ID
See Hardware identifier, 1045

I

I address, 1045
I slave
Hardware catalog, 1736
I&M data
downloading to PROFINET IO devices, 2069
I&M data: Read with Get_IM_Data, 3947
I/O
Direct access to, 971
I/O access error, 970
I/O address, 971, 1045
I/O communication, 1776
I/O field, 5389
Data format, 5390
Hidden input, 5390
Mode, 5390
I/O input, 967
I/O output, 967
I/Os, 2148
IB, 4841

ICMP, 771
Icon
For comparison, 2028
For comparison status, 2079
for connection status, 2079
For hardware diagnostics, 2027
for operating state, 2029
For software diagnostics, 2028
Overlay icon, 2029
Icons
in TeleService, 6577
Icons in the force table, 5228
Icons in the watch table, 5199
ID, 4841
Identify alarm classes
Alarm view, 5404
I-device, 1763, 1779
Proxy device, 1823
IE/AS-i link PN IO, 1764
IE/PB Link, 78, 1800
IE/PB Link PN IO, 78
IEC Check,
Setting, 2334, 2534
IEC timer, 2262, 2263
IEEE 802.11n, 1542, 1663
Channel bonding, 1544
Frame aggregation, 1544
Guard interval, 1544
Maximum ratio combining, 1543
MIMO, 1543
Spatial multiplexing, 1543
IEEE 802.3, 747
IEEE tag, 1913
IF, 3385
iFeatures
iREF, 1550
IGMP, 1434
IGMP querier, 1434
IKE settings, 808
IM0_Data, 3947
Image, 5896, 5917, 5918, 5953
HMI device, 5949
Image file
Storing in the image browser, 5315, 5864
Image memory, 1186
IMC, 2884, 3209, 3445
Import
Alarm, 5826
Analog alarm structure, 5827
Project texts, 5843, 5861
Recipe, 5822
Structure of discrete alarms, 5830
Structure of recipe data, 5824
Tag, 5833
Text list, 5839
ImportDataRecords, 5726
Importing
Recipe, 5582
Recipe data record, 5582
Importing a recipe
Structure for the import, 5824
Importing analog alarms
Structure for the import, 5827
Importing discrete alarms
Structure for the import, 5830
Importing NTP servers, 800
in the screen
Arrange object, 5295
IN_RANGE, 2679, 3004
INC, 2705, 3031
Incoming, 5486, 5774
IncreaseFocusedValue, 5674
IncreaseTag, 5692
Increment, 2705, 3031
Indenting and outdenting
Lines, 5017
Index, 1135
Index tag, 5450, 5451
Indirect addressing, 2158, 2164, 2168, 2169, 2212, 2215, 5450
Industrial PC, (see IPC)
Information
ARP table, 1281, 1560
Ethernet Statistics, 1294, 1568
Event log, 1282, 1561
Frame type, 1295, 1569
Groups, 1325, 1577
Inter AP blocking, 1579
IPv6 Neighbor Table, 1282, 1559
LLDP, 1300
Packet Errors, 1296, 1297, 1570
Role, 1326, 1578
Security, 1323, 1325, 1574, 1577
Spanning Tree, 1285, 1564
Versions, 1279, 1558
WLAN authentication log, 1562
Information system
Components of the information system, 283
Help on messages, 287
Roll-out, 286
Tooltip, 286
Infotext, 5493
Creating, 5497, 5500

display, 5941, 5945
Key, 5941
INIT_RD, 2843, 3167, 3412
Initialization
IPE file, 6656
Project file, 6662
Initialization (user-defined web pages), 1057
Initialize all retain data, 2843, 3167, 3412
Initializing
Controller data, 5905
IPE file, 6657
Project file, 6662
Input
Insert, 2900
Inserting, 4910
remove, 4911, 4968
Input (I), 967
Input byte (IB), 967
Input field
Character mode, 5940
Standard mode, 5940
Input finished, 5773
Input on the HMI device
By means of function key, 5942
Input word (IW), 967
INSERT, 3541
Insert a comment section, 3397
Insert empty line, 6248
Insert input, 4967
Insert separator line, 6248
Inserting
Graphic, 5286
Library object, 5815
Object, 5286
Rectangle, 5326
Inspector window, 576
Diagnostics tab, 2023
Hardware and network editor, 539
Layout, 255
Reducing automatically, 249
Install update, 157
Installation
Displaying software, 148
Licenses, 125
Log, 145
Modifying products, 149
Repairing products, 151
run automatically, 123
Starting, 146
Support package, 157
System requirements, 126
Target directory, 123
Updates, 157
Updating products, 149
Virtual environment (private cloud), 123
Installation requirement
PC Station, 1105
Installing CA certificate, 6609
Installing license keys, 129
Installing support packages, 157
Installing the local modem, 6585
Instance
Creating an instance, 486
Displaying associated type, 487
Instance data block
Creating, 2100
Definition, 2100
Modifying the data types of IEC timers and IEC counters, 5035
Retentive behavior, 4809
Instruction
Copying, 5052
Cutting, 5052
Deleting, 5052
Inserting, 5052
Rules, 5023
Search, 4697
Specify data type, 4894, 4896, 4951, 4953
Versions, 4697
Instruction profile
Activating and deactivating, 4704
Basics, 4701
Creating, 4702
Deleting, 4705
Editing, 4703
Opening, 4703
Instructions for configuring a remote modem, 6587
INT, 2250, 2344, 2405, 2474, 2503, 2538, 2553
INT_TO_, 2405, 2503, 2553
Integer, 2809, 3134, 3369
16-bit, 2250, 2251
32-bit, 2252, 2253
64-bit, 2254, 2255
8-bit, 2248, 2249
Integrated
Configuring, 6674
Integrated connection, 5425
Integrated project
Converting an unspecified CPU, 215
Creating an integrated HMI connection, 216
Deleting an unspecified connection, 220
Linking HMI tags, 218
Migrating, 200, 211
Post-editing, 213

Intelligent DP slave, (See I-slave)
Inter AP blocking
Allowed addresses, 1728
Basic, 1727
Configuration, 1727
Information, 1579
Inter Project Engineering
Creating "Device proxy data" in the source project, 6654
Exchanging controller data across projects, 6651
Exporting an IPE file by means of the "Device proxy data", 6655
Procedure for exchanging controller data via a project file, 6652
Procedure for exchanging controller data via an IPE file, 6652
Procedure for updating already transferred controller data, 6653
Requirements, 6651
Software and hardware requirements, 6651
Inter Project Engineering (IPE)
Exchanging controller data via IPE file, 6649
Exchanging controller data via project file, 6649
Introduction, 6648
Interconnecting, 603
Interconnecting of ports, 1788
Interconnecting ports
Graphic view, 713, 715
Table view, 714, 716
Interface, (See block interface)
Adding, 6498
Displaying, 2035
Renaming, 716
Interface properties, 2035
Interference frequency suppression, 1996
Internal network nodes
Configuring, 817
Diagnostics, 831
Internal reference junction, 1998
Internet Key Exchange (IKE), 809
Interpolator OB, 1038, 6160, 6162
Inter-Project_Engineering
IPE, 5905
Interrupt event
Delaying with DIS_AIRT, 3869
Disabling with DIS_IRT, 3861
Enabling with EN_AIRT, 3870
Enabling with EN_IRT, 3863
Interrupts
Activating time-of-day interrupt with ACT_TINT, 3819
Canceling time-delay interrupt with CAN_DINT, 3832
Canceling time-of-day interrupt with CAN_TINT, 3818
Querying time-delay interrupt with QRY_DINT, 3833
Querying time-of-day interrupt with QRY_TINT, 3820
Receiving from I/O module with RALRM, 3636
Setting time-of-day interrupt with SET_TINT, 3813
Setting time-of-day interrupt with SET_TINTL, 3816
Starting time-delay interrupt with SRT_DINT, 3830
With packed addresses, 1757
Intrusion Detection System (IDS), 134, 138
Invalid cross-references, 5849
Invalid license
With a time zone change, 131, 143
InverseLinearScaling, 5693
INVERT, 2852, 3176
InvertBit, 5698
InvertBitInTag, 5700
InvertLinearScaling, 5693
IO address, 4073
IO controller
Devices and modules, 1781
Display in the IO device, 1783
PN interface, 1782
IO device
Display of the IO controller, 1783
Networking, 1786
Update time, 1789
Watchdog time, 1791
IO system, 1785
Creating, 1786
IO2MOD, 4080
IO-Link, 1814
IP access control list, 739
IP address, 69, 601, 603, 1767
IP address parameters, 1769, 1779
IP configuration
Change parameters from the user program, 4547
IP mapping, 1589
IP packet filter rules, 771
IP parameters, 2036
IP rule sets, 748
IP services, 764
IP subnet, 603
IPC, 1102

iPCF, 1547
Configuration, 1596, 1597, 1729
How it works, 1547
PNIO communication, 1547
Restriction, 1549
iPCF-MC
PNIO communication, 1547
IPE, 5905
Creating "Device proxy data" in the source project, 6654
Exchanging controller data across projects, 6651
Exporting controller data of the "Device proxy data" as an IPE file, 6655
File, 6656, 6657
Introduction, 6648
Software and hardware requirements, 6651
IPE data
Updating, 6660, 6665
IPE file, 6656
Initialization, 6656
Initializing, 6657
iPRP
Configuration, 1733
IPsec settings, 808
IPv4
OSPF, 1263
VRRP, 1262
VRRPv3, 1262
IPv4 addresses, 777
IPv4 routing
NAT Translations, 1313
IPv4 routing over backplane bus (CM/CP 154x-1), 2020
IPv6
E-mail, 2021
FETCH/WRITE, 2021
FTP client, 2021
FTP server, 2021
Notation, 1240, 1537, 2020
OSPFv3, 1267
SNMP, 2021
Use with the CP 1543-1, 2021
VRRPv3, 1262
IPv6 addresses, 778
IPv6 routing
Default routes, 1605
OSPFv3 AS-Scope LSDB, 1318, 1319
OSPFv3 Interfaces, 1315
OSPFv3 Link-Scope LSDB, 1321
OSPFv3 Neighbors, 1316
OSPFv3 virtual neighbors, 1317
Routing table, 1314, 1321, 1572
Static routes, 1477
IQ sense, 2006
iREF, 1550
Configuration, 1734
IRT
Setup recommendations, 1803
IS_ARRAY, 2694, 3019, 3271
IS_NULL, 2692, 3016
ISAKMP, 822
I-slave, 1737, 1746
Configuring, 1747
Data access, 1748
Data exchange, 1738
Proxy device, 1821
ISO protocol, 877
Isochronous mode, 1763, 1815, 1821
Oversampling, 1963
Isochronous real-time communication (IRT), 1763
ISO-on-TCP
Characteristics, 621
TSAP, 639
IW, 4841
IX, 4841

J

Jerk limit
Function, 6165
JMP, 2819, 3145
JMP_LIST, 2823, 3148
JMPN, 2820, 3146
JOIN, 3513
Jump, 2819, 2820, 2821, 2823, 2824, 2827, 3145, 3146, 3147, 3148, 3149, 3152
Jump distributor, 2824, 3149
Jump label, 2821, 3147, 3395
Jump list, 2823, 3148

K

K parameter, 92
Key
Backspace, 5940
Cursor, 5939
Delete, 5940
END, 5940
ENTER, 5940
ESC, 5940
HOME, 5940
Infotext, 5941

Scrolling back, 5940
Scrolling up, 5940
TAB, 5939
Key operation, 5939
Key pair for password protection, 5912
Keyboard layout, 5907
Keyboard operation, 553
Basic functions of the TIA Portal, 270
Customizing editors, 275
Editing objects, 276
Editing texts, 277
Navigation in the TIA Portal, 273
Online functions, 279
Project editing, 272
Selecting objects, 276
Tables, 278
Window, 272
Keyboard shortcut
Move object, 5295
Keyboard shortcuts, 5907
KEY-PLUG, 1376, 1378
Formatting, 1378, 1647
iFeatures, 1648
Keyword highlighting, 5013
Keywords, 2135
Know-how protection
Changing a password, 4761
Introduction, 4746
Opening blocks, 4759
Printing block, 4760
remove, 4762
Setting up, 4758

L

Label,
Recipe view, 5412
LACP, 1424
LAD, 4878, 4926
Detailed comparison, 5095
LAD element
Copying, 4908
Cutting, 4908
Deleting, 4912
Inserting, 4893
Inserting an operand, 4913
pasting from the clipboard, 4909
Replacing, 4910
Rules for inserting, 4890
Selecting, 4907
LAD network
Branch, 4916
Closing a branch, 4918
Crossing, 4919
Deleting a branch, 4919
Deleting a crossing, 4921
Insert crossing, 4920
Inserting a block call, 4899, 4956
Inserting a branch, 4918
Program status display, 5188
Prohibited interconnections, 4892
Rearranging a crossing, 4920
Rules for simultaneous branches, 4917
Ladder Logic, 4878
Language
Activate project language, 5854
Asian languages, 5853
Asian operating system, 5853
Disabling the project language, 5854
Editing language, 5855
Language support, 5853
Language-dependent format, 5852
Language-specific graphic, 5862
Log, 5473, 5521, 5871
multilingual project, 5856
Reference language, 5855
Regional format of the date, time, currency, and numbers, 5852
Language abbreviation, 1080
Language and font, 5908
Language switching, 1081, 5866
Log, 5473, 5521, 5871
Order, 5908
Runtime language, 5868
Languages
Migrating, 194
Layer
Assigning objects to a layer, 5375
Layer 2, 747
MLD (IPv6), 1436
Layer 3, 747, 1378
Layer 3 (IPv6), 1474
Configuration, 1474
Layout
Alarm indicator, 5530
Alarm view, 5399, 5400, 5407, 5527, 5529
Bar, 5384
Button, 5414
Circle, 5396
Date/time field, 5388
Ellipse, 5391
Graphic I/O field, 5394
Graphic view, 5392
I/O field, 5389

Line, 5399
Recipe view, 5411
Rectangle, 5409
Simple recipe view, 5595
Simple user view, 5386
Switch, 5413
Symbolic I/O field, 5416
Text field, 5419
Trend view, 5397
User view, 5387
Layout of the force table, 5226
LDT, 2266, 2366, 2443
LDT_TO_, 2443
Lead and lag algorithm, 2888, 3213, 3449
LEAD_LAG, 2888, 3213, 3449
Learning functionality, 818
Learning mode, 877
LED, 3945
LED status
Reading out with LED, 3945
LEDs, 1137
LEDs of the hardware platform, 1137
LEFT, 3534
LEN, 3531
LGC_GADR, 4092
Library, 67, 5793
Adding a master copy, 473
Adding and using a block, 4718
Basics, 439
Buttons and Switches, 5798
Cleaning up, 519
Comparing library elements, 520
Copying a library object, 5797
Copying types to the clipboard, 516
Creating folders, 515
Cutting library elements, 516
Cutting master copies, 516
Cutting types, 516
Deleting instances, 517
Deleting types and versions, 517
Harmonizing names and path structure, 518
Master copies, 439
Master copy, 5797
Moving library elements, 517
Open, 5805
Opening a released type version, 479
Pasting master copies from the clipboard, 517
Pasting types from the clipboard, 516
PLC data types generated by the system, 83
Released type version, 478
Save, 5804
Storing an object, 5814
Style, 5818
Style sheet, 5818
Type, 5797
Type version in test, 478
Types, 439
Using filter view, 474
Using master copies, 472, 475
Using types, 476
Versioning of types, 476
Library management
Displaying cross references of an instance, 456
Displaying instances of type versions, 456
Displaying relations between library objects, 457
Filtering, 5802
Filtering by non-released types, 455
Opening, 454, 5801
Overview, 451, 5800
Library object, 5793, 5794
Inserting, 5815
Library view
Exit, 450
Opening, 450
Overview, 447, 5795
License
Defective, 131, 143
Defective license, 131, 143
Starting without a valid license, 140
Starting without valid license, 128
License key, 130, 141
Handling license keys, 130
Working with license keys, 141
License Manager Panel Plugin, 141
Licenses, 125
Life of certificates, 805
Light emitting diode
Acknowledge, 5941
Toggle, 5941
LIMIT, 2712, 3037, 3277
Login attempts,
Limit and enable password legitimation, 2829, 3153, 3398
Limit range
Tag, 5443
Limit value, 1954, 2712, 3037, 3277, 5493
Tag, 5444
Limit value exceeded
Analog alarm, 5444
Limits
Display, 5907
Line, 5399
Design, 5298

Line end, 5298, 5399
Line start, 5399
Line break, 1929
Line end
Line, 5399
Line start
Line, 5399
Linear programming, 2086
LinearScaling, 5694
Link folder
External graphic, 5314
Linked objects
Copying, 5887, 5888
Linking, 5845
Linking HMI tags, 218
LINT, 2254, 2351, 2419
LINT_TO_, 2419
LLDP, 739, 740, 1300, 1798
LLDP (Link Layer Discovery Protocol), 1798
LN, 2717, 3042, 3281
Load
HMI device, 5949
Recipe data record in runtime,
Load memory, 966, 5141
Displaying, 5145
Load names, 5907
Load window layout
Loading additional window layouts, 267
Loading via quick access, 267
LoadDataRecord, 5728
Loading
Configurations with PROFIBUS, 1820
Configurations with PROFINET, 1822
Configurations with Web server, 1820
Device configurations, 1818
Distributed I/O, 1819
Downloading project data to the device, 350
from a device, 75, 82
General information, 348
Going online, 1826
GSDML, 1822
HMI devices, 1825
I-device, 1823
IE/PB-Link, 1824
I-slave, 1821
Isochronous applications, 1815, 1821
Module comments, 77
Proxy device, 1821, 1823
Shared Devices, 1825
Subnets, 1819
to PG/PC, 1816
to the device, 75, 1815
Loading a project
from external storage medium, 5918
with connected HMI device, 5917
Without connected HMI device, 5918
Loading with S7 routing, 5917
Local assignment
Of a function key, 5359, 5363
Local data bit (L), 967
Local data byte (LB), 967
Local data double word (LD), 967
Local data word (LW), 967
Local logging, 825
Local time, 982
Calculating with SET_TIMEZONE, 3476
Reading out with RD_LOC_T, 3471
Writing with WR_LOC_T, 3473
Local users, 1495, 1699
Local variable, 4764
Log, 5908
Alarm log, 5561
Automatic entries, 5562
Data log, 5561
Language switching, 5473, 5521, 5871
Level-dependent management, 5474, 5523
Runtime language, 5473, 5521, 5871
Log behavior
Control with system function, 5477
Level-dependent management, 5474, 5523
Managing upon system start, 5473, 5522
Log contents
Displaying, 5517
Log entries, 5562
Log file
Modem alarms, 6632
Log language, 5473, 5521, 5871
Log table
Event log, 1282, 1561
WLAN authentication log, 1562
Log type, 5516
Log with level-dependent execution of system functions, 5517
Log with level-dependent system alarm, 5517
LOG_GEO, 4088
LOG2GEO, 4077
LOG2MOD, 4079
Logarithm, 2717, 3042, 3281
Logging, 825
CP 343-1 Adv. / 443-1 Adv., 893
Log type, 5516
SCALANCE S, 861, 863
Tag value, 5469
Tags, 5470, 5475

Tolerance band, 5475
Within/outside the limit values, 5475
Logging cycle, 5475
Logging language, 5907
Logging tag
Configuring, 5479
Logic analyzer function, 6513
Logic operation
AND, 2848, 3172
EXCLUSIVE OR, 2850, 3175
OR, 2849, 3173
Logic path
Deleting, 4977
Inserting, 4977
Use, 4976
Logical address
Determining a module with GADR_LGC, 4091
Determining a module with RD_LGADR, 4089
Determining associated slot with LGC_GADR, 4092
Determining associated slot with LOG_GEO, 4088
Login attempts
Limit, 5908
Logoff, 5688
Logoff time, 5908
Changing, 5625
Changing in runtime, 5633
Logon, 5688
Logon failed, 5636
Reporting, 5639
User, 5634
Logon dialog
Configuring access protection, 5638
Logon for web server, 1052
Logon web server, 1052
Logout
Automatic, 1639
Logs
Deleting logs, 330
Displaying logs, 329
LogTag, 5682
Long word, 2246
Loop, 3385, 3388, 3391, 3392, 3393, 3394
Loop monitoring with S7-1200/1500, 5182
Loop monitoring with S7-300/400, 5182
Loop-In-Alarm, 5775
Configured,
Low free storage space, 5778
LREAL, 2258, 2357, 2429, 2479, 2518, 5005
LREAL_TO_, 2429, 2518
LTIME, 2263, 2362, 2437
LTIME_TO_, 2437
LTOD, 2265, 2374, 2453
LTOD_TO_, 2453
LWORD, 2246, 2339, 2394
LWORD_TO_, 2394

M

MAC packet filter rules, 775
MAC rule sets, 748
MAC services, 767
Machine tailoring, (See Reconfiguring IO systems)
Main, 1022
Main mode, 809
Making characters available for
WString/WChar, 5463
Managing
Graphic,
User group, 5625
Users, 5624
Users in runtime, 5632
Man-in-the-middle attack, 920
Manual fragment, 1076
Master copy
Library, 5797
Math functions
CALCULATE, 4903, 4960
Mathematical functions
CALCULATE, 4931, 4986
MAX, 2710, 3036, 3275
MAX_LEN, 3512
Maximum, 2710, 3036, 3275
Maximum cycle time, 963, 988
Maximum load memory available
Displaying, 5145
MB, 4841
MB_CLIENT, 4634, 4653
MB_COMM_LOAD, 4617
MB_MASTER, 4620
MB_SERVER, 4643, 4665
MB_SLAVE, 4628
MC_ChangeDynamic
Instruction, 4124, 4159
Parameter, 4124, 4159
MC_CommandTable
Instruction, 4122, 4157
Parameter, 4123, 4158
MC_Halt
Function chart, 4107, 4142
Instruction, 4105, 4140
Parameter, 4174
Parameters, 4106, 4141

MC_Home
Instruction, 4100, 4136
Parameter, 4102, 4137
MC_MoveAbsolute
Function chart, 4111, 4145
Instruction, 4108, 4143
Parameter, 4108, 4144
MC_MoveJog
Function chart, 4121, 4156
Instruction, 4119, 4154
Parameter, 4120, 4155
MC_MoveRelative
Function chart, 4114, 4149
Instruction, 4112, 4146
Parameter, 4113, 4147
MC_MoveVelocity
Function chart, 4118, 4153
Instruction, 4115, 4150
MC_Power
Function chart, 4098, 4133
Instruction, 4094, 4129
Parameters, 4094, 4130
MC_ReadParam
Instruction, 4126, 4161
Parameter, 4127, 4162
MC_Reset, 4099, 4134
MC_WriteParam
Instruction, 4128, 4163
Parameter, 4128, 4164
MCAT, 2880, 3206, 3441
MC-Interpolator OB, 1038, 6160, 6162
MC-PostServo OB, 1037
MC-PreServo OB, 1037
MC-Servo
Assigning parameters, 1043
MC-Servo OB, 1036, 6160, 6162
MD, 4841
MD5, 802
MDM, 300
Meaning of the columns in the force table, 5226
Measure program runtime, 2846, 3170, 3413
Measurements
Project navigator, 6521
Measuring range, 1968
Measuring window (GATE), 1926
Media converter, 1788
Media redundancy (MRP), 1763
Memory area
Load memory, 966
Retentive memory areas, 968
Work memory, 967
Memory bit (M), 967
Memory byte (MB), 967
Memory card, 68, 964, (See Memory Card)
Accessing, 422
Adding a card reader, 422
Formatting, 2056
Introduction, 421
Reading out information with GetSMCinfo, 3966
Removal/insertion, 75
Showing properties, 423
Memory double word (MD), 967
Memory requirements
Recipe, 5961
Memory reset, 964
Making, 2052
Memory size
Alarm buffer, 5521
Memory test, 1131
Memory word (MW), 967
Menu command
Simple recipe view, 5596
Message
Sending, 1003
Specifying the end, 1005
Specifying the start, 1004
MIB, 739
MID, 3537
Migrated project
Compiling, 185
Migrating projects
Procedure, 179
Requirements, 178
Migration
Adaptations beforehand, 186
Alarm groups, 191
Allen-Bradley DF1 data types, 204
Allen-Bradley Ethernet IP data types, 205
Basics, 181
Data types Modicon Modbus, 206
Data types Modicon Modbus TCP/IP, 207
Data types Omron Hostlink/Multilink, 207
Data types SIMATIC S7 200, 208
Data types SIMATIC S7 300/400, 208
Displaying a log file, 180
Displaying history, 180
Font settings, 189
Global library, 196
Including the hardware configuration, 174
Integrated project, 211
Introduction, 181
Introduction to migration, 173
Migrating an integrated project, 200, 211
Migrating projects, 184, 202

Mitsubishi FX data types, 206
of external tags, 204
of languages, 194
of project text, 194
Project library, 196
Recipe data, 197
Runtime data, 197
Supported HMI devices, 186
Supported products, 173
Tags, 190
Text, 194
The migration process, 173
User administration, 197
Migration of IP access protection lists when activating Security, 79
Migration of projects with Ethernet CPs and Security, 79
Migration tool, 183
Creating a migration file, 178
Distribution and sources, 164
Including the hardware configuration, 176
Removing, 165
System requirements, 164
Using the migration tool, 175
Migration: Check migration readiness of the hardware, 175
MIN, 2709, 3034, 3273
Minimizing projects, 341
Minimum, 2709, 3034, 3273
Minimum cycle time, 963, 988
Mirroring, 1182, 1403, 1406
Missing supply voltage, 1995
Mitsubishi FX
Migrating data types, 206
Mixed mode, 816
MLD, 1436
MOD, 2703, 3028, 4993
Modbus communication
as a slave with MB_SLAVE, 4628
as Modbus master with MB_MASTER, 4620
Configuring a port with MB_COMM_LOAD, 4617
Mode, 5415
Graphic I/O field, 5394
I/O field, 5390
Modem
Local, 6585
Problem, 6627
Remote, 6586
Modem alarms
Log file, 6632
Modem connection cannot be established, 6631
Modem connection is closed, 6632
Modem connection is interrupted, 6631
Causes and solutions, 6631
Modems without plug-and-play, 6586
Modicon Modbus
Migrating data types, 206
Modicon Modbus TCP/IP
Migrating data types, 207
Modify
High-speed counters, 4194, 4196
Modify value
Permitted operands, 5204
Modifying mode, 5212
Module
Addressing, 1045
Asynchronously reading data record with RD_DPARA, 3789
Communication Status, 830
Copying, 570
Deleting, 575
Determine diagnostics status, 2043
Determining logical address of the slot with LOG_GEO, 4088
Determining logical address with GADR_LGC, 4091
Determining logical addresses with RD_LGADR, 4089
Determining slot of a logical address with LGC_GADR, 4092
Determining start address with GEO_LOG, 4086
Inserting, 75, 564, 566
Moving, 572
Multiple selection, 567
Reading data record with RD_DPAR, 3783
Removing, 75
Rename, 573
Replacing, 572
Select, 563
Time of day on a module, 2052
Module addressing, 971
Module arrangement, 564
Module comments, 77
Module replacement, 215
Module rights, 738
Module titles, 562
Module version
Updating, 580
ModuleStates, 3974
Monitor, 1793
Monitor GRAPH programs, 5192
Monitor SCL programs, 5191
Monitor STL programs, 5189

Monitoring
"Monitor all" command, 5214, 5240
"Monitor now" command, 5240
"Monitor once and now" command, 5215
Monitoring and modifying tags in the DB editor, 4822, 4823, 4824
Monitoring and modifying modes, 5212
Monitoring FBD programs, 5188
Monitoring GRAPH programs, 5192
Monitoring LAD programs, 5188
Monitoring loops, 5182
Monitoring mode, 5212
Monitoring of the partner port..., 1788
Monitoring SCL programs, 5191
Monitoring STL programs, 5189
Monitoring structures, 5184
Monitoring structures (S7-1200/1500), 5184
Monitoring window, 2005
Mono-master system, 1737
Motion Control CPU S7-1200
Guidelines, 6168
Motion Control S7-1200
Drive and encoder connections, 6151
PROFIdrive, 6151
Telegram, 6151
Move,
Block, 2743, 2794, 3069, 3119, 3297, 3355
Block uninterruptible, 2761, 2797, 3086, 3122, 3313, 3358
Character string in array with Strg_TO_Chars, 3506
Screen, 5260
Value, 2730, 3055
Move block, 2757, 3082, 3309
Move object
Keyboard shortcut, 5295
Move view
Keyboard operation, 554
Overview navigation, 529, 531, 538
MOVE_BLK, 2743, 3069, 3297
MOVE_BLK_VARIANT, 2316, 2757, 3082, 3309
MoveVelocity
Parameter, 4116, 4151
Moving
Hardware components, 572
Moving columns
User view, 5388
MRP domain, 2039
MRP domain properties, 2038
MSK_FLT, 3846
MSTP, 1419, 1691
MUL, 2700, 3025
Multicast, 786, 1433
Multi-instance, 2152
Declaring, 4780
Definition, 2112
Multiple instance
Changing the call type, 4902, 4959
Multiple selection, 563, 5291
Hardware component, 567
Multiple Spanning Tree, 1419, 1691
Multiplex, 2857, 3182, 3419
Multiplex/synchronous mode, 2011
Multiplexing, 5450
Address multiplexing, 5454
with absolute addresses, 5454
with symbolic addresses, 5455
Multiply, 2700, 3025
Multiuser Engineering
Compiling and loading,
Compiling in the local session,
Compiling in the server project view,
Installation of the multiuser server as standalone installation, 166, 6712
Installation of the multiuser server with the TIA Portal, 166, 6711
Introduction, 166, 6687, 6689, 6711
Rules and requirement for installation, 167, 6712
Rules for compiling and loading,
Starting the installation of the multiuser server, 167, 6712
Uninstalling the multiuser server, 168, 6713
MUX, 92, 2857, 3182, 3419
MW, 4841
MX, 4841
My Documentation Manager, 300

N

N, 2588, 2591, 2913, 2916
N_TRIG, 2593, 2918
Names of alarm classes
Change through migration, 192
Naming conventions
Alarm log, 5564
Data log, 5564
NAPT, 1546
Configuration, 1696
Configuring, 1473
NAT, 1546
Configuration, 1695
Configuring, 1471, 1472
NAT Translations, 1313

NAT/NAPT
Routing, 785
NE_ElemType, 2690, 3015
NE_Type, 2686, 3011
NEG, 2704, 3029
Negate, 2704, 3029
Negotiation, 1366, 1650
Nesting depth, 2087, 2108
Network, 81
Closing, 4886, 4944
Copying, 4885, 4943
Deleting, 4885, 4943
Entering a comment, 4888, 4946
Insert title, 4887, 4945
Inserting, 4883, 4885, 4941, 4943
Navigating blocks, 4889, 4947
Opening, 4886, 4944
Selecting, 4884, 4942
Using, 4882, 4940
Network access
Opening the properties, 6498
Network comments
Hiding, 4694
Showing, 4694
Network connection
Creating, 6676
Network drive, 66
Network editor
Components, 526
Function, 525
Hardware catalog, 545
Inspector window, 539
Network ID, 870
Network overview
Basic functions, 589
Basic functions for editing the network overview table, 590
Network Syslog, 825
Network view
Adding device, 557
Hardware and network editor, 528
I/O communication, 1776
Networking devices
Basics of configuring networks, 585
Editing interface addresses, 592
Editing interface parameters, 591
Editing network parameters, 591
Networking several interfaces at the same time, 587
Networking with an existing subnet, 588
Networking without an existing subnet, 587
Networks within a project, 585
Requirements, 591
Types of communication, 585
Networking in the device view
Networking options, 590
Procedure, 590
Networking mode, 587
New data types, 2211
New key pair
Generation, 5913
No supply voltage, 1954
Non-licensed mode
Engineering System, 140
ES, 128
HMI devices, 141
non-typed, 5010
NORM_X, 2812, 3138, 3373
Normalize, 2812, 3138, 3373
Normally closed contact, 2576
Normally open contact, 2574
NOT, 2577, 5000, 5001
Not equal, 2693, 3017
NOT_NULL, 2693, 3017
NOT_OK, 2683, 3008
NTP, 1433
NTP (secure), 796
NTP server, 796
NULL, 2302
Number of lines
User view, 5387
Numerical key assignment, 5941

NOT DEFINED

add external graphic, 5864

O

O, 6478
OB
Events and OBs, 973
Overview, 973
OB 1, 1022
OB 80, 1032
OB 82, 1033
OB for manufacturer-specific interrupt, 1027
OB for profile-specific interrupt, 1027
OB servo, 6162
OB 83, 1034
object
Arrange,

Assigning to a layer,
Availability for Basic Panels,
Creating a new OLE object,
Creating an OLE object from a file,
Deleting,
Design,
Designing the background color,
Designing the fill pattern,
Dynamization,
Dynamization of the appearance,
Dynamize direct movement,
editing within a group,
Evenly distributing,
Flip,
Flush alignment,
Group, 5318
Grouping,
Inserting,
Inserting of the same type,
Inserting several times,
Multiplying,
outside the area,
Paste, 5310
Repositioning and resizing multiple objects,
Repositioning objects,
Resize,
Rotate, 5297
Select multiple objects,
Selecting multiple objects,
Storing in a library,
Tab sequence,
Transparency,
Object group
Animation, 5356
Editing an object within a group, 5321
Removing an object, 5320
Object list
Tag, 5432
Object names
Uniqueness, 182
Object size
Harmonizing, 5291
Obtaining user rights
Logging on to the operating system with administrator privileges, 281
With Windows user account control, 281
Occupation of connection resources, 691
OF, 3386
Off delay, 2602, 2614, 2633, 2643, 2927, 2958, 2968, 3231, 3250
Offline, 721
OK, 2682, 3007
OLE object
create from a file, 5286
Recreate, 5286
Save in Graphics, 5315
Storing in the image browser, 5864
Omron Hostlink/Multilink
Migrating data types, 207
On delay, 2599, 2611, 2628, 2639, 2924, 2953, 2964, 3228, 3245
Retentive, 2630, 2641, 2956, 2966, 3248
On demand, 5447
On exceeding, 5770
On falling below, 5771
Ones complement, 2852, 3176
Online, 721
Change connection path, 6482
Display of the online mode, 6483
Establish connection, 6482
Go offline, 6484
Hardware detection, 558
Show accessible devices, 6475
Online access, 6495
Online and Diagnostics view, 2022
Online assignment (PROFINET device names), 1776
Online connection, 6478
Multiple TIA Portal instances, 75
Specifying a default connection path, 6481
Online device, 2023
Online diagnostics, 827
Online displays
Orange color, 2031
Online mode, 6478
Display of the online mode, 6480
Go online, 6482, 6484
Standby or hibernation of the programming device / PC, 6478
Online operation, 75
Online Tools, 2022, 2032
On-screen keyboard, 5907
OPC server
Central communication service provider, 1100
OPC Unified Architecture Server, 5908
Open
Global library, 5805
Selection list, 5940
Open force job, 68
Open log file
Packet filter events, 833, 835
Security events, 833, 835
System events, 832, 834

Open user communication
Ability to read back,
Changing IP configuration parameters with T_CONFIG,
Changing parameter values,
Connection configuration,
Connection description, 627, 628, 629
Connection establishment,
Connection parameters,
Creating a connection,
Deleting an interconnection,
Establishing a communication connection with TCON,
Establishing a connection and reading data with TRCV_C,
Establishing a connection and sending data with TSEND_C,
General,
Instructions,
Port numbers, 637
Protocols used,
Receiving data via UDP with TURCV,
Receiving data with TRCV,
Sending data via UDP with TUSEND,
Sending data with TSEND,
Starting connection parameter assignment,
TCON_FDL,
TCON_IP_RFC,
TCON_IP_v4,
TCON_IP_V4_SEC,
TCON_Param,
TCON_QDN, 629
TCON_QDN_SEC, 630
Terminating a communications connection with TDISCON,
TSAP,
OpenAllLogs, 5684
OpenControlPanelDialog, 5675
Opening a force table, 5229
Opening a project, 323
Opening a watch table, 5201
Opening phone books, 6577
Opening Siemens Industry Online Support, 549
Operand, 2134, 2135, 2140, 2142, 2143, 2144, 2146, 2147, 2148, 2150, 2152, 2153, 2154, 2155, 2276, 2281, 2285, 2292
Inserting, 4913, 4970
Operand area, 967
Operating behavior
Enabling system memory, 1006
Using clock memory, 1006
Using time-of-day functions, 990
Operating element
Availability for Basic Panels, 5382
Recipe view, 5603
Operating hours counters
Handling with RTM, 3487
Operating mode, 892
Introduction, 958
RUN, 963
STARTUP, 960
STOP, 963
Transitions, 959
Operating object
operation, 5940
Operating recipes
Copying a recipe data record, 5606
Creating recipe data records, 5599, 5600, 5605
Deleting a recipe data record, 5606
Load recipe data record, 5606
Modifying a recipe data record, 5606
Read recipe data record, 5600
Reading data records, 5608
Transfer data record, 5601
Transferring a data record, 5608
Operating system, 131, 2085
Asian language setting, 5852
Setting to Western, 5852
Updating, 5955
Operation
Alarm indicator, 5530
Alarm view, 5529
Alarm window, 5529
Key,
Operating object,
Recipe view, 5602
Simple alarm view, 5528
Simple recipe view, 5596
Operation with the keyboard
Recipe view, 5604
Operation with the mouse
Recipe view, 5604
Operator control
Alarm view, 5401, 5407, 5529
Alarm window, 5529
Trend view, 5398
Operator controls
Alarm view, 5403
Recipe view, 5412
Optical handheld reader, 5946
Optimize position controller, 6264
Optimized access, 2102, 2104
Optimized block access, 2210

Option handling, 1753, 1755, (See Configuration control (ET 200MP)), (See Configuration control (ET 200SP))
Option handling (see Configuration control), 1085, 1939, 1981, 2013
OR, 2849, 2896, 2898, 3173, 5001
Order
  Language switching, 5908
Order number, (See Article number)
Organization block (OB)
  Assigning event with ATTACH, 3795
  Canceling event assignment with DETACH, 3799
  Creating, 4715
  Cyclic interrupt OB, querying parameters with QRY_CINT, 3807
  Cyclic interrupt OB, setting parameters with SET_CINT, 3805
  exporting to an external source file, 4741
  Function, 2088
  Measuring runtime with RT_INFO, 3940
  Reading start information with RD_SINFO, 3930
  Start information, 2088
OSPF
  Area range, 1464
  Areas, 1263, 1463
  Interfaces, 1465
  Link State Advertisement, 1263
  OSPFv2 interfaces, 1307
  OSPFv2 LSDB (information), 1311
  OSPFv2 neighbors, 1308
  OSPFv2 virtual neighbors, 1310
  Router, 1263
  Router status, 1263
  Virtual Links, 1467
OSPF (IPv6)
  Area Range, 1490
  Areas, 1489
  OSPFv3 AS-Scope LSDB, 1318, 1319
  OSPFv3 Interfaces, 1315
  OSPFv3 Link-Scope LSDB, 1321
  OSPFv3 Neighbors, 1316
  OSPFv3 virtual neighbors, 1317
  Virtual Links, 1492
OSPFv2
  Configuration, 1462
OSPFv3
  Configuration, 1488
OUC, (See Open User Communication)
OUT_RANGE, 2681, 3006
Outgoing, 5486, 5774
Output
  Inserting, 4910, 4967
  remove, 4911, 4968
Output (O), 967
Output byte (QB), 967
Output double word (QD), 967
Output pulse sequence
  with preset frequency, 3993
Output rate (high-speed analog modules), 1964
Output word (QW), 967
Overflow, 1930, 1954, 5778
Overlay icon, 2029
Overlaying
  Tag, 2155
Overlaying tags, 4779
Oversampling, 1963
Overview
  Access point, 1579
  Clients, 1585
  Overlap APs, 1584
  Overlapping channels, 1584
  using the display formats in the force table, 5234
  via the display formats in the watch table, 5206
Overview navigation, 529, 531, 538
Overview of the cross-reference list, 5153
Overview of the mode of operation with Multiuser Engineering, 6687, 6689
Overview of TS Adapters that can be used, 6584
Overview window, 262
  Comparing objects, 264
  Display all blocks, 265
  Showing and hiding columns, 265
  Sorting details view, 265

P

P, 2586, 2589, 2911, 2914
P_TRIG, 2592, 2917
P12, 938
Packet filter log, 833, 835
PageDown, 5748
PageUp, 5747
Pane mode, 259
Panning, (See move view)
Parameter
  For CPU, 982
  Hidden parameters, 4914, 4971
  Predefined - writing with WR_DPARM, 3793
Parameter assignment
  Hardware, 539
Parameter assignment parameters
  Hardware, 555

Parameter passing, 2122
Parameter transfer, 2123, 2126, 2129, 2131, 2132, 2133
  Transfer parameter as copy or as pointer, 2126
Parameter types, 2325
Partner port
  Information on monitoring, 1788
Password, 5908
  Changing, 5625
  For CPU access protection, 982
  for function key, 5366
  Hierarchy level, 5618
  Options, 1500, 1698
  Password aging, 5619
  Password complexity, 5619
Password for transport
  Creating, 5913
  Reset, 5913
Password legitimation, 2829, 3153, 3398
Password protection, 982, 1007, 1008, 1052
  Protection concept, 378
  Revoking access rights, 378
Password protection in runtime, 5912
Paste
  Tag, 6682
Pasting
  Adjusting screen size, 5886
  Color, 5885
  Font, 5885
  Function key, 5886
  Invalid object, 5885
  Principle, 5884
PC CP, 720
PC station
  Common features of PC Station V2.0 and V1.0,
  Configuring offline, 1111, 1112
  Improvements from PC Station V2.0 to V2.1,
  Improvements from PC Station V2.1 to V2.2,
  Improvements of PC Station V2.0 compared to V1.0,
  Installation requirement,
  Selecting the version,
PC system
  Configuring, 1104
  Selection from hardware catalog, 1104
PE_CMD, 3735
PE_DS3_Write_ET200S, 3740
PE_End_RSP, 3771
PE_Error_RSP, 3769
PE_Get_Mode_RSP, 3774
PE_I_DEV, 3765
PE_List_Modes_RSP, 3773
PE_Measurement_List_RSP, 3779
PE_Measurement_Value_RSP, 3780
PE_PEM_Status_RSP, 3776
PE_Start_End, 3730
PE_Start_RSP, 3770
PE_WOL, 3741
PEEK, 3332
  Read memory address, 3332
PEEK_BOOL, 3334
  Read memory bit, 3334
PEM, 938
Perfect Forward Secrecy, 811
Performance, 2209
Performance features
  HMI device, 5962, 5966
Permanent connection, 1214
Permit access with PUT/GET communication from remote partners, 1012
Permit overwriting of device name, 1798
Permitted file names, 434
Permitted operands for the force table, 5231
Permitted operands for the watch table, 5203
PG/PC interface
  Go online, 6484
  Modifiable MPI interface parameters, 6503
  Modifiable PROFIBUS interface parameters, 6503
PID_3Step
  In/out parameters, 4268
  Input parameters, 4264, 4295
  Instruction, 4255, 4287
  Output parameters, 4266, 4297
  Static tags, 4298
PID_Compact
  In/out parameters, 4210
  Input parameters, 4208, 4238
  Instruction, 4235
  Output parameters, 4209, 4239
  Static tags, 4211, 4240
PID_Temp
  ActivateRecoverMode tag, 4375
  Cascade, 4333
  Cascading, 6126
  ErrorBits parameter, 4373
  In/out parameters, 4333
  Input parameters, 4329
  Mode, 4333
  Multi-zone applications, 6133
  Operating principle, 4323
  Output parameters, 4330
  PID_Temp state and mode parameters, 4364
  PwmPeriode, 4379

Static tags, 4334
Tag Warning, 4378
Ping, 1380, 1649
PLC
Tag, 5450
PLC data, 6655
PLC data type, 2229
Addressing, 2281
Change the number, 4869
Comparing, 5118
Creating, 4868
Declaration in data blocks, 4806
Declaration in the block interface, 4778
Declaration table for PLC data types, 4867
Declaring, 4873
Declaring ARRAY, 4871
Declaring STRUCT, 4872
Definition, 2277
Deleting, 4869
PLC data types generated by the system in libraries, 83
Programming the structure, 4870
Tag properties, 4874, 4875
PLC data types, 2211, 5429
PLC prefix, 5441
PLC tag
Automatically filling in cells, 4861
Comparing, 5115
Comparing a PLC tag table, 5086
Comparison, 5083
Copying, 4859
Declaring, 4846, 4847, 4849, 4851
Definition, 2140
Importing and exporting, 4862, 4863, 4864, 4865
Monitor, 4858
Offline/offline comparison, 5086
Permissible addresses and data types, 4843
PLC tag table, 4837, 4838, 4844, 4845
Properties, 4855, 4856
Reconnecting, 5442
Retentive behavior, 4848
Rules, 4843
Sorting rows, 4860
Updating comparison results, 372
PLC tags, 1061, 1062, 1064
PLC UDTs
WinCC, 5429
PLUG, 1378
C-PLUG, 1645, (C-PLUG)
KEY-PLUG, 1648, (KEY-PLUG)
PN/PN couplers
Grouping, 1812
Linking Ethernet subnets, 1812
Pointer,
ANY, 2305
POINTER, 2302
VARIANT, 2308
point-to-point, 1251, 1556
Point-to-point communication (PtP), 997
Freeport protocol, 997
Point-to-point connection
Configuring communication parameters with PORT_CFG, 4584
Configuring receive parameters with RCV_CFG, 4589
Configuring serial transmission parameters with SEND_CFG, 4587
Deleting receive buffer with RCV_RST, 4601
Enable receiving with RCV_PTP, 4599
Initiating data transfer with SEND_PTP, 4597
Querying the signal state with SGN_GET, 4602
Setting output signals with SGN_SET, 4603
POKE, 3335
Write memory address, 3335
POKE_BLK, 3339
Write memory area, 3339
POKE_BOOL, 3337
Write memory bit, 3337
Port, 603
102 (S7 protocol - TCP), 765
123 (NTP), 786, 796
161 (SNMP), 786
20/21 (FTP), 765
443 (HTTPS), 786
4500 (IPsec), 786
500 (IPsec), 786
500 (ISAKMP - UDP), 822
514 (Syslog), 786
80 (HTTP), 765
8448 (security diagnostics), 905, 908
Configuration, 1368, 1651
Interconnecting, 1788
Port configuration, 1365, 1372, 1649, 1652
Renaming, 716
Port configuration, 1372, 1652
Port diagnostics
SFP diagnostics, 1384
Port numbers, 637
Port options, 1797
Enable autonegation, 1796
Monitor, 1793
Transmission rate/duplex, 1793

PORT_CFG, 4584
Portal view, 239
Position
  Editing multiple objects, 5312
  Of objects, 5288
Position control, 6220
Position monitoring, 6218
Positioning axis technology object
  Active homing, 6215
  Actor tags, 6439
  Actor variables, 6331
  Add new object, 6182
  Axis name configuration, 6184, 6377
  Basic parameters, 6184, 6376
  Changing the configuration parameters for dynamics in the user program, 6209
  Changing the configuration parameters for homing in the user program, 6217
  Commissioning overview, 6181
  Configuration overview, 6181
  Configuration window icons, 6183
  Configuring acceleration, 6204
  Configuring active homing, 6211, 6381
  Configuring approach/homing direction, 6212, 6382
  Configuring deceleration, 6204
  Configuring distance per motor revolution, 6196, 6379
  Configuring end of homing switch, 6213, 6214
  Configuring end of reference point switch, 6381, 6383
  Configuring home position offset, 6213, 6383
  Configuring homing switch input, 6212, 6214
  Configuring invert direction signal, 6196, 6380
  Configuring jerk limiter, 6205
  Configuring maximum velocity / start/stop velocity, 6204
  Configuring passive homing, 6213, 6380
  Configuring permit auto reverse at the hardware limit switch, 6382
  Configuring pulses per motor revolution, 6196, 6379
  Configuring ramp-down time, 6204
  Configuring ramp-up time, 6204
  Configuring reference point position, 6213, 6215, 6381, 6383
  Configuring reference point switch input, 6381, 6382
  Configuring smoothing time, 6206
  Configuring the homing speed, 6213, 6383
  Configuring the velocity limiting unit configuration, 6204
  Diagnostics overview, 6181
  Drive enable configuration, 6379
  Drive ready configuration, 6379
  Drive signal configuration, 6188, 6379
  DynamicDefaults tags, 6344, 6452
  DynamicLimits tags, 6343, 6451
  Emergency stop deceleration configuration, 6207
  ErrorBits tags, 6362, 6468
  Extended parameters, 6379
  FollowingError tags, 6350, 6458
  General dynamics configuration, 6203
  Hardware and software components, 6177
  Hardware interface configuration, 6186, 6377
  Homing tags, 6348, 6456
  Mechanics configuration, 6196, 6197, 6379
  Mechanics tag, 6342, 6449
  Modulo tags, 6342, 6450
  Passive homing, 6216
  Position tag, 6328, 6437
  PositionControl tags, 6350, 6457
  PositionLimitsHW tags, 6454
  PositionLimitsHW variables, 6347
  PositionLimitsSW tags, 6453
  PositionLimitsSW variables, 6346
  PositionMonitoring tags, 6352, 6459
  PTO and HSC configuration, 6187, 6377
  Response when jerk limiter is activated, 6208
  Sensor[1] tags, 6442
  Sensor[1] variables, 6334
  Sensor[1].ActiveHoming tags, 6446
  Sensor[1].ActiveHoming variables, 6338
  Sensor[1].Interface tags, 6443
  Sensor[1].Interface variables, 6335
  Sensor[1].Parameter tags, 6444
  Sensor[1].Parameter variables, 6336
  Sensor[1].PassiveHoming tags, 6447
  Sensor[1].PassiveHoming variables, 6340
  Simulation tag, 6353, 6354
  StandstillSignal tags, 6353, 6460
  StatusBits tags, 6464
  StatusBits variables, 6357
  StatusDrive tags, 6355, 6462
  StatusPositioning tags, 6354, 6461
  StatusSensor tags, 6356, 6463
  Tag ActualPosition, 6330, 6438
  Tag ActualVelocity, 6330, 6439
  Tools, 6180
  Unit of measurement configuration, 6185
  Units tag, 6341, 6448, 6449
  Updating the tags, 6364, 6470
  User unit configuration, 6378
  Velocity tag, 6329, 6437

PostServo OB, 1037
Power budget, 1990
Pragmas in SCL, 5049
Predefined alarm classes, 5488, 5489
Predefined firewall rules
CP 1543-1, 905, 908
CP 1628, 896
CP 343-1 Adv. / 443-1 Adv., 894
SCALANCE S, 861, 863
Predefined styles, 5309
Prerequisites for establishing a VPN connection, 6613
PreServo OB, 1037
Pre-shared keys, 807
Press, 5773
Press ESC twice, 5773
Press key, 5777
PressButton, 5676
Print scope, 381
Printable contents, 381
Printing
Changing the settings, 382
Creating a cover page, 387
Creating frames, 386
Documentation settings, 380
Elements in the library, 385
Elements in the project tree, 385
Non-printable contents, 381
Non-printable objects, 382
Print contents, 384
Printing device view, 550
Printing network view, 550
Process a cover page, 388
Process borders, 388
Specify layout, 383
Specify print layout, 383
Specifying the print area, 392
Structure of printout, 380
Use borders, 385
Use borders and cover pages, 380
Use cover page, 383
Printing a project, 397
Printing labeling strips, 401
Determining the correction value for the print shift, 406
Printing of alarms
Configuring:printing of alarms, 5507
Prioritization, 1391
Prioritized startup, 1763
Priority, 1391
Private Cloud, 70
Private Key, 916
Procedure for monitoring structures, 5185
Procedure when working with local sessions, 6789
Process data area
Module addressing, 971
Module start address, 971
Process image, 971
Basics, 969
I/O access error, 970
Reading inputs with GETIO, 3615
Reading the process image area with GETIO_PART, 3624
Synchronizing the inputs with SYNC_PI, 3568
Synchronizing the outputs with SYNC_PO, 3569
Transferring the process image area with SETIO_PART, 3626
Updating, 970
Updating inputs with UPDAT_PI, 3562
Updating the outputs with UPDAT_PO, 3565
Writing outputs with SETIO, 3616
PROFIBUS, 598
Connection with PROFINET, 1764
ET 200S in DPV1 mode, 1756
Number of communication connections, 682
PROFIBUS cable configuration
Optical ring, 600
PROFIBUS DP, 1745
PROFIBUS profile, 598
Different profiles on the same subnet, 598
Effect on the transmission rate, 598
Meaning of profiles, 599
PROFIBUS DP, 1739
PROFIdrive, 6151
PROFIenergy, 1763
Control PROFIenergy commands in the I-Device, 3765
Description, 3728
Generate answer to command at end of pause, 3771
Generate answer to command at start of pause, 3770
Generate list of supported measured values as answer, 3779
Generate negative answer to command, 3769
Generate PEM status as answer, 3776
Generate queried energy data as answer, 3774
Generate queried energy savings modes as answer, 3773
Generate queried measured values as answer, 3780
Generate supported PROFIenergy commands as answer, 3777
PE_CMD, 3735

PE_DS3_Write_ET200S, 3740
PE_End_RSP, 3771
PE_Error_RSP, 3769
PE_Get_Mode_RSP, 3774
PE_I_DEV, 3765
PE_Identify_RSP, 3777
PE_List_Modes_RSP, 3773
PE_Measurement_List_RSP, 3779
PE_Measurement_Value_RSP, 3780
PE_PEM_Status_RSP, 3776
PE_Start_End, 3730
PE_Start_RSP, 3770
Reading out status information, 3735
Set power module switching behavior, 3740
Start and exit energy-saving mode, 3730, 3735
Wake on LAN, 3741
Profile, (See bus profile)
Profiles, 1027
PROFINET, 603, 877, 1242, 1545, 1643
Assigning a device name, 1776
Connection with PROFIBUS, 1764
Device replacement without exchangeable medium, 1798
Number of communication connections, 682
Optimizing with IRT, 1803
Optimizing with RT, 1801
PROFINET device name, 1774
PROFINET interface, 982, 1779, 2035
PROFINET IO, 1242, 1545, 1762, 1763, 1767, 1769, 1782, 1786
Port options, 1793
PROFINET IO device
Assigning a name, 2061
Name assignment in the dialog, 2064
Name assignment in the online and diagnostics view, 2062, 2063
PROFINET IO devices
Reading a portion of the inputs with GETIO_PART, 3624
Reading consistent data with DPRD_DAT, 3700
Writing a part of the outputs with SETIO_PART, 3626
Writing all outputs with SETIO, 3616
Writing consistent data with DPWR_DAT, 3702
PROFINET IO system, 1785
Creating, 1781
Highlighting, 1784
PROFINET RT class, 1763
Program alarm
generating with Program_Alarm, 3882
Program card, 964
Program control operations, 2836, 3160, 3405
Program cycle OB
Description, 1022
Program editor
"Instructions" task card, 4686
Block interface, 4686
Enlarging the programming window, 4693
Favorites, 4686
Function, 4684
General settings, 230, 232, 233, 235, 4708
Programming window, 4686
Structure, 4685
Testing task card, 4687
Toolbar, 4686
Using the keyboard, 4688
Program execution
Cyclic, 1022
Program information
Displaying, 5120
In the assignment list, 5121
In the call structure, 5129
In the dependency structure, 5135
in the tab, 5141
Views, 5120
Program name
Displaying, 6675
Program resources, 4714
Program status
Call environment, 5178, 5179
Editing blocks, 5185
FBD, 5188
Function, 5176
GRAPH, 5192
LAD, 5188
Modify tag, 5186
SCL, 5191
STL, 5189
Switching off, 5180
Switching on, 5180
Program_Alarm, 3882
Programming
Linear, 2086
Structured, 2086
Programming language
Changeover, 4713
FBD, 4937
LAD, 4878
Rules for changing, 4713
Programming recommendations, 2213
Programming window
Customizing, 5016
SCL, 5015

Project
Carrying out an online/offline comparison, 360
Carrying out offline/offline comparisons, 361
Closing, 326
Comparing data, 358
compile (overview), 5914
Compiling, 5915
Compiling project data, 348
Creating, 322
Deleting, 327
Detailed comparison, 371
Download, 5920
download (overview), 5914
Harmonizing, 518
Load, 1108
Migrate, 184, 202
multilingual, 5856
Removing, 326
Saving, 325
Showing properties, 329
Simulation with a tag simulator, 5928
Upgrade, 5894
Project archives, 341
Project data
Compiling, 348
Downloading to a memory card, 352
Export alarm, 5824
Export text list, 5837
Exporting a recipe, 5820
Exporting tag, 5831
Importing a tag, 5832
Importing alarms, 5825
Information on compiling, 347
Information on loading, 348
Loading, 350
Uploading from a device, 354
Writing to a memory card, 352
Project file
Initialization, 6662
Initializing, 6662
Project ID, 5907
Project language, 407, 5851
Activate, 5854
Disabling, 5854
Project languages
Changing the editing language, 410
Specifying project languages, 409
System texts, 5857
Use, 407
User texts, 5857
Project library, 439, 5793
Adding types, 481
Assigning a common version to types, 500
Cleaning up, 519
Creating folders, 515
Discarding all versions, 491
Discarding the type version, 491
Duplicating types, 484
Migrating, 196
Performing a consistency check for a type version, 490
Release all versions, 493
Release the type versions, 492
Updating the project, 494
Using filter view, 474
Using types, 486
Project size
Reducing, 5932
Project text
Exporting, 5841, 5859
Project texts
Application example, 415
Changing the text of selected objects, 411
Displaying reference text, 411, 5858
Exporting all texts, 414
Exporting individual texts, 413
Exporting texts of a device, 413
Importing, 415, 5843, 5861
Migrating, 194
Translating individual texts, 5857
Translating project texts, 410
Translation into project languages, 407, 5856
Project tree
Adding device, 556
Function, 243
Hiding, 4693
Reducing automatically, 249
Showing, 4693
Showing additional columns, 247
Project version
Upgrade, 5894
Project view, 240
Properties (CPU), 982
Properties of the VPN group, 808
Properties window, (See Inspector window)
ProSave, 5950
Protect
Function key with password, 5366
Protection concept, 1009, 1012
Introduction, 378
Revoking access rights, 378
Protection level, 1007, 1008
Revoking access rights, 378
Protocol, 764

Proxy
Routing, 6667
Proxy ARP, 888
Proxy device, 1821, 1823
PRVREC, 3706
PT, 2945
PTC resistor, 1965
Public Key, 916
Pull/plug interrupt (ET 200M), 2003
Pull/plug interrupt OB, 1034
Pulse, 2597, 2608, 2622, 2635, 2922, 2947, 2961, 3225, 3240
Extended, 2625, 2637, 2950, 2963, 3243
Pulse extension, 1953
Pulse generator
Enabling/disabling with CTRL_PWM, 3991
Pulse interface
Principle, 6146
Pulse stretching, 2005
Pulse width modulation
ET 200MP, 2000
PUT, 4393
PUT / GET, 1012
PWM
ET 200MP, 2000

Q

Q address, 1045
QB, 4841
QD, 4841
QoS, 1391
QoS Trust, 1245, 1539
QRY_CINT, 3807
QRY_DINT, 3833
QRY_TINT, 3820
QW, 4841
QX, 4841

R

R, 598, 2579, 2904
R_TRIG, 2594, 2919, 3219
Rack error OB, 1035
Rack or station failure, 1035
Racks, 561
Inserting a module, 564
RADIUS, 1501, 1706
RALRM, 3636
Range
Value outside range, 2681, 3006
Value within range, 2679, 3004
Range (... - ...)
Graphics list, 5342
RCV_CFG, 4589
RCV_PTP, 4599
RCV_RST, 4601
RCVREC, 3704
RD_ADDR, 4083
RD_DPAR, 3783
RD_DPARA, 3789
RD_DPARM, 3791
RD_LGADR, 4089
RD_LOC_T, 3471
RD_REC, 3694
RD_SINFO, 3930
RD_SYS_T, 3469
RDREC, 3578
RE_TRIGR, 2835, 3159, 3405
Read continuously
Tag, 5447
Read field, 2790, 3115
Read memory address, 3332
Read memory bit, 3334
Read PLC tags
String or character tags in expressions, 1063
Tags of the type String and Character, 1063
Read system time with TIME_TCK, 3486
READ_BIG, 3345
READ_DBL, 4050
READ_ERR, 3848
READ_LITTLE, 3341
Readback data record, 2013, (See feedback data record)
Readback data record 197, 1934, 1991, 2013
ReadFromArrayDB, 2772, 3097, 3323
ReadFromArrayDBL, 2777, 3102, 3327
Reading
A data record with RD_REC, 3694
Asynchronously- data record of a module with RD_DPARA, 3789
Data record of a module with RD_DPAR, 3783
from a data block in load memory with READ_DBL, 4050
Reading data
From DP standard slaves/PROFINET IO devices with DPRD_DAT, 3700
From remote CPU with GET, 4388
Reading out
LED status with LED, 3945
Local time with RD_LOC_T, 3471

Reading out a diagnostics buffer, 2046
Reading tags, 1062
Read-only, 1007
REAL, 2257, 2355, 2426, 2478, 2515, 2539, 2559, 5005
REAL_TO_, 2426, 2515, 2559
Receiving data
on I-device with RCVREC, 3704
Recipe, 1055, 5565, 5566, 5567, 5824
Application example: Machine parameter assignment, 5566
Application example:Batch production, 5566
Basics, 5565, 5566
Creating new, 5585
Data record, 5567
Entry, 5567
Export format, 5725
Exporting, 5582, 5820
Exporting to CSV file, 3996
Importing, 5582, 5822
Importing into data block, 3999
Memory requirements, 5961
Synchronizing recipe tag, 5607
Use, 5566
Use of text lists, 5570
Recipe data
Migration, 197
Recipe data record
Changing, 5606
Copy, 5606
Creating new, 5585
Creating on the HMI device, 5599, 5600, 5605
Deleting, 5606
Export format, 5725
Exporting, 5582
Importing, 5582
load, 5606
Synchronizing, 5607
Use of text lists, 5570
Recipe data record name
Writing to a tag, 5575, 5580
Recipe data record number
Writing to a tag, 5575, 5580
Recipe element
Creating new, 5585
Recipe list, 5574
Recipe name
Writing to a tag, 5575, 5580
Recipe number
Writing to a tag, 5575, 5580
Recipe tag
Synchronizing, 5607
Recipe view, 5573, 5602
Advanced, 5578
Application, 5602
Behavior with screen change, 5581, 5605
Column headers, 5413
Configurable events, 5576, 5581
Configuring, 5591, 5593
Display number, 5412
Displaying one recipe only, 5579
Displaying values only, 5575
Dynamic properties, 5577
Label, 5412
Layout, 5410, 5411
Operating element, 5603
Operation, 5602
operation using function keys, 5599, 5604
Operation using the function key, 5581
Operation with the keyboard, 5604
Operation with the mouse, 5604
Operator controls, 5412
Recipe data record, 5568, 5582
Simple, 5574
Toolbar, 5410
Update, 5578
Updating, 5574
Use, 5410
Using as a drop-down list, 5580
RecipeExport, 3996
RecipeImport, 3999
RecipeViewBack, 5738
RecipeViewClearDataRecord, 5734
RecipeViewGetDataRecordFromPLC, 5734
RecipeViewMenu, 5735
RecipeViewNewDataRecord, 5733
RecipeViewOpen, 5735
RecipeViewRenameDataRecord, 5738
RecipeViewSaveAsDataRecord, 5737
RecipeViewSaveDataRecord, 5734
RecipeViewSetDataRecordToPLC, 5736
RecipeViewShowOperatorNotes, 5738
RecipeViewSynchronizeDataRecordWithTags, 5737
ReconfigIOSystem, 3677
Reconfiguration via user program, 1085, 1934, 1939, 1981, 1991, 2013
Reconfiguring IO systems, 3677
Reconnecting
Tag, 5442
Reconnecting a PLC tag, 5442
Record Protocol, 919
Recording a log file for the modem, 6627
Rectangle
Configuring, 5327

Inserting, 5326
Inserting and configuring, 5326
Radius corners X, 5410
Radius corners Y, 5410
Redoing actions
Basics of redoing actions, 316
Redoing actions, 319
Reducing
Project size, 5932
Redundancy methods
HRP, 1252
Reference channel, 1954
Reference channel error, 1996
Reference channel of the module, 1998
Reference junction, 1749, 1915, 1930, 1954, 1957, 1997, 1998
Reference language, 407, 5851
Selecting, 5855
Reference object
Defining, 5292
Reference project
Basics, 344
Closing, 344
Comparing, 346
Opening, 344
Reference temperature, 1954, 1957, 1997, 1998
References, 2211
Regions in SCL, 5026, 5029, 5030, 5032, 5033
Reinitialization of technology objects, 6284
release, 5775
Release key, 5777
ReleaseButton, 5677
Remainder of division, 2703, 3028
Remote access user, 736
Remote connection
as CPU-controlled remote connection, 6619
as dial-up connection, 6583
As VPN connection, 6606
Remote connection cannot be established, 6606
Remote connection from the TS Adapter is not established, 6630
Remote connection to the TS Adapter is not established, 6628
Remote system access to a programming device/ personal computer, 6620
Remove
Formatting in the alarm text, 5502
Object from the group, 5320
Removing
Folder link, 5314
Rename, 573
Screen, 5260
Tag, 5436
Template, 5266
Renaming recipe data records, 5606
Renewing the CA group certificate, 817
REPEAT, 3392
Replace,
Cross-references, 5845
Replace delimiter, 5441
Replace modules during operation, 2003
Replacing
Module, 572
Replacing devices, 572
Limitations on connections, 5876
Principle, 5876
Requirements for establishing a remote connection, 6605
Requirements for sending an SMS, 6623
Requirements for sending emails, 6624
Reset, 1333, 1608
Bit field, 2582, 2907
IEC timer, 2619, 3237
Operand, 2579, 2585, 2904, 2910
to factory settings, 5954
Reset timer, 2944
RESET_BF, 2582, 2907
ResetBit, 5701
ResetBitInTag, 5703
Resetting
to factory settings, 5954
Resetting the connection, 4533
Resetting the MPI/PROFIBUS settings, 6510
Resetting the TCP/IP settings, 6502
Resetting the user interface layout, 269
Resetting to factory settings, 2055
Resistance thermometers, (see Thermal resistor)
Resources, 5141
Code work memory, 5141
Data work memory, 5141
Introduction, 5141
Load memory, 5141
Retentive memory, 5141
Retentive memory data, 5141
Work memory, 5141
Resources tab
Structure, 5143
Restart, 1333, 1608
Restart of technology objects, 6284
Restore
Data of the HMI device, 5952
Restore process
License key, 142
Restore window layout, 267

Restoring
Data of the HMI device, 5951
Restoring a backup, 6490, 6494
Restoring a device, 6490, 6494
Restoring data
HMI device, 5951, 5952
Restrictions due to user rights, 280, 281
Recognizing, 281
RET, 2827, 3152
Retain, 2104
Retentive behavior
Bit memory, timers, counters, 4848
Retentive bit memories
Enabling the display, 5129
Retentive memory, 5141
Retentivity, 968, 2102
Bit memory, timers, counters, 4848
Block interface, 4769
Data block, 4809, 4810
PLC tag, 4848
Retentivity of IP address parameters, 1779
Retrieving projects, 342
Return, 92
RFC 5280, 916
RIGHT, 3536
Ring buffer, 2320
Ring redundancy,
HRP, 1388
MRP, 1387
RIP
RIPv2 statistics, 1313
RIPng
Configuration, 1494
Interfaces, 1494
RIPng Statistics, 1321
RIPv2
Configuration, 1469
Interfaces, 1469
RLO
Invert, 2577, 2901
RMON
History, 1441
Statistics, 1440
ROL, 2869, 3194, 3430
Role name, 737
Roles, 736, 1496, 1701
System-defined, 736
User-defined, 736
Root bridge, 1556
Root certificate, 923
Root certification authorities, 731
Root, square, 3280
ROR, 2867, 3192, 3428
Rotate
Left, 2869, 3194, 3430
Object, 5286
Right, 2867, 3192, 3428
Round, 2805, 2806, 2807, 2809, 3130, 3131, 3133, 3134, 3365, 3366, 3368
Routed connection
Configuring, 6671
Router, 601
Router IP address, 870
Routing, 1261
IPv6 routing table, 1314, 1572
Routing table, 1306
Static routes, 1261
VRRP, 1261
Routing connection
Configuring, 6667
Routing mode, 784
RS, 2585, 2910
RS-232/PIP multi-master cable, 1002
RT, 2619, 2944, 3237
RT class, 1763
RT_INFO, 3940
RTM, 2328, 3487
Rules for configuring MPI networks
Rules for assigning the MPI address, 594
Rules for loop monitoring with S7-1200/1500, 5182
Rules for loop monitoring with S7-300/400, 5182
Rules for monitoring structures (S7-1200/1500), 5184
Rules for network configuration for PROFIBUS networks
Assigning device addresses, 594, 595
Rules for working in local sessions, 6789
RUN, 963
Rung
Deleting, 4923
Inserting, 4923
Use, 4922
RUNTIME, 2846, 3170, 3413
Execution of the function list,
Measure program runtime, 2846, 3170, 3413
Measuring with RT_INFO,
Simulating,
Start screen,
starting on a panel,
Runtime data
Migration, 197
Runtime language, 5851, 5865
Font, 5869
Log, 5473, 5521, 5871

Order for language switching, 5868
Selecting, 5867
Runtime languages, 5908
Runtime settings
User administration, 5262, 5617, 5634
Runtime start
Basic Panel, 5925
Runtime Stop, 5777
Runtime version, 5915

S

S, 2580, 2905
S_AVERZ, 2633
S_CD, 2655, 2980
S_COMP, 3492
S_CONV, 91, 3494
S_CU, 2653, 2979, 3264
S_CUD, 2982
S_EVERZ, 2628
S_MOVE, 3490
S_ODT, 2628, 2953, 3245
S_ODTS, 2630, 2956, 3248
S_OFFDT, 2633, 2958, 3250
S_PEXT, 2625, 2950, 3243
S_PULSE, 2622, 2947, 3240
S_SEVERZ, 2630
S5TIME, 2261, 2358, 2432, 2541, 2561
S5TIME_TO_, 2432, 2561
S7 communication, 690
Parameter, 4385
Receive data from a remote partner instruction with BRCV, 4410
Receive data from a remote partner instruction with URCV, 4401
Send and receive parameters, 4386
Send data to a remote partner instruction with BSEND, 4407
Send data to a remote partner instruction with USEND, 4399
User data size, 4387
S7 connection, 653
TSAP, 664
S7 CPU
Load memory, 966
Operand area, 967
Work memory, 967
S7 routing, 78
Configuring, 6671
Connection resources, 690
Loading, 5917
via IE/PB Link, 78
S7-1200, 76
S7-1200 modules, 76
S7-1200 V2
Data types, 5429
S7-CP, 720
S7-modular Embedded Controller, 1143
Configuring, 1144
S7-PCT, 1814
SA lifetime, 810
Safe AS-i output, 1151
SafelyRemoveHardware, 5675
Safety guidelines, 301
Safety instruction
Changing the recipe data record in the background, 5574
Recipe data record in background, 5578
Safety precautions when forcing tags, 5225, 5243
Sampling rate (high-speed analog modules), 1964
Save
Data of the HMI device, 5952
Global library, 5804
Save window layout, 266
SaveDataRecord, 5742
Saving a force table, 5230
Saving a watch table, 5202
Saving phone books, 6578
Saving the user interface
Saving the window layout, 266
SC, 2659, 2984
Scalable measuring range, 1968
SCALANCE S, 720
Scale, 2810, 2815, 3135, 3140, 3371, 3381
SCALE_X, 2810, 3135, 3371
Scaling
Applying linear scaling to a tag, 5447, 5449
Scan operand for negative signal edge, 2588, 2913
Scan operand for positive signal edge, 2586, 2911
Scan RLO for negative signal edge, 2593, 2918
Scan RLO for positive signal edge, 2592, 2917
Scheduler, 5780
Deactivated task, 5783
Event trigger, 5783, 5784, 5787
Function list, 5783
Triggers, 5783
Work area, 5781
SCL, 4991, 5053, 5055
Copying and inserting regions, 5030
Delete region, 5033
Detailed comparison, 5102
Insert comment, 3397
Insert pragmas, 5049
Inserting regions, 5029

Navigating in regions, 5032
Programming window, 5015
Using regions, 5026
SCL instruction
Changing the data type, 5035
Data type basics, 5034
Inserting, 5024, 5025
Rules, 5023
Selecting, 5051
SCL program
Inserting a block call, 5039, 5040, 5042, 5043, 5044, 5045
Inserting an instruction, 5024, 5025
Inserting comments, 5050
Program status display, 5191
Selecting an instruction, 5051
Showing and hiding the parameter list, 5048
Screen
Availability for specific devices, 5255
Copy, 5260
Copying, 5886
Creating, 5259
Creating a type, 5817
Deleting, 5260
Font, 5256
Inserting, 5260
Library, 5817
Move, 5260
Rename, 5260
Use template, 5267
Working steps in creating, 5259
Zooming, 5258
Screen change, 5772
Screen keyboard
Basic Panels, 5936
Basic Panels 2nd Generation, 5936
Screen object
Dynamic movement, 5352
Screen resolution, 5907
Screen template, 5361
ScreenObjectCursorDown, 5745
ScreenObjectCursorLeft, 5745
ScreenObjectCursorRight, 5746
ScreenObjectCursorUp, 5744
ScreenObjectPageDown, 5747
ScreenObjectPageUp, 5746
Script
Creating a type, 5816
Library, 5816
Updating tag value, 5425
Scrolling back key, 5940
Scrolling up key, 5940
SD, 2637, 2639, 2964
SD card, 68, (See Memory card)
SE, 2637, 2963
Search
Additional options for searching in the editor, 313
Basics of searching, 303
Basics of searching in the project, 304
Examples for searching in the project, 310
Find and replace in the editor, 315
Find and replace palette, 314
Hardware catalog, 547
Replacing search terms in the editor, 315
Search editor, 305
Search in entire project, 307
Secure communication, 916
Secure Socket Layer, 919
Security, 81
Security Configuration Tool
Configuration views, 721
Security event, 1012, 1013
Security events, 833, 835
Security module, 720
Security online diagnostics of S7 CPs, 79
Security program, 134, 138
Security settings, 1352, 1628
Roles and rights, 937
SEG, 2891, 3216, 3451
Segment diagnostics
Graphic display, 2076
Icons, 2076
Text display, 2076
Segmented circular log, 5516
SEL, 2855, 3180, 3417
Select, 2855, 3180, 3417
Multiple objects, 5291
Select alarm classes
Alarm view, 5403
Selecting
Multiple objects, 5286
Selection changed, 5770
Selection list
Open, 5940
Self-signed certificates, 921
Send buffer, 1186
Send clock cycle, 1789
SEND_CFG, 4587
SEND_PTP, 4597
Sequencer, 2871, 3196, 3432
Serial number, 1280, 1559
Serialize, 2739, 3064, 3293
Service data
Saving, 2050

Service groups, 766, 769
Servo motor, 6141
Servo OB, 1036, 1037, 6160, 6162
Set
  Bit field, 2581, 2906
  Counter start value, 2659, 2984
  Operand, 2580, 2584, 2905, 2908
Set limit value, 2712, 3037, 3277
Set operand on negative signal edge, 2591, 2916
Set operand on positive signal edge, 2589, 2914
SET_BF, 2581, 2906
SET_CINT, 3805
SET_TIMEZONE, 3476
SET_TINT, 3813
SET_TINTL, 3816
SetBacklightColor, 5752
SetBit, 5706
SetBitInTag, 5708
SetBitWhileKeyPressed, 5710
SetBrightness, 5751
SetConnectionMode, 5715
SetDataRecordTagsToPLC, 5740
SetDataRecordToPLC, 5739
SetDeviceMode, 5712
SETIO, 3616
SETIO_PART, 3626
SetLanguage, 5714
SetPLCMode, 5713
SetRecipeTags, 5741
SetTag, 5695
Setting
  Deadband, 5500
  Delay time, 5499
  Language, 5946
  Languages in the operating system, 5852
  Style, 5310
  View options for the assignment list, 5125
  View options for the dependency structure, 5139
Setting AS-i network configuration parameters, 1147
Setting ENO automatically, 5013
Setting language, 5946
Setting parameters for the Ethernet interface, 6501
  Adding a dynamic IP address, 6501
  Connecting to a subnet, 6500
  Deleting dynamic IP addresses, 6502
  Modifiable parameters, 6499
  Options in the parameter settings, 6499
Setting the communication load, 601
Setting the font, size and color, 5016
Setting the mnemonics, 4695
Setting the tab spacing, 5016
Settings
  Changing, 236, 4711, 4881, 4940, 5014
  FBD, 4881, 4939
  General, 230, 232, 233, 234, 235, 4708
  LAD, 4881, 4939
  Online and diagnostic functions, 235
  SCL, 5013
Settings for tags, 5440
Settings for the PG/PC interface, 6497
  Automatic bus parameter detection, 6504
  Setting parameters for the MPI interface, 6505
  Setting parameters for the PROFIBUS interface, 6507
Settings for user help, 430
Settings in the cross-reference list, 5149
Seven-segment display, 2891, 3216, 3451
Severity, 783
SF, 2643, 2968
SFP diagnostics, 1384
SGN_GET, 4602
SGN_SET, 4603
SHA algorithm, 1352, 1628
SHA1, 802
Shared device, 1763
  Associated IO controllers, 2037
Shift
  Key,
  Left, 2865, 3190, 3426
  Right, 2863, 3188, 3425
SHL, 2865, 3190, 3426
Short-circuit to ground, 1952
Short-circuit to L+, 1953
Show line numbers, 5016
Show/hide interfaces, 6497
ShowAlarmWindow, 5723
Showing and hiding absolute operands, 5016
Showing and hiding columns, 369, 4790, 4819, 4861, 4877
ShowLogonDialog, 5691
ShowOperatorNotes, 5681
SHR, 2863, 3188, 3425
Shut down (PC system), 2834, 3158, 3404
Shut down target system, 2834, 3158, 3404
SHUT_DOWN, 2834, 3158, 3404
SiClock, 769
Sign, 2707, 3033
Signal board, 1049
  Inserting, 1050
Signal recorder, 1674
Signature, 922
signature check
  Switching off, 5949

SIMATIC ET 200AL, 1947, 1974, 1978
SIMATIC Manager, 6674
SIMATIC memory card
Formatting, 2056
SIMATIC NET version
Onboard interface, 1142
PC Station, 1142
SIMATIC S7 Embedded Controller, 77
SIMATIC Logon, 5908
SIMATIC S7 200
Migrating data types, 208
SIMATIC S7 300/400
Migrating data types, 208
SIMATIC-ACC, 664
Simple alarm view
Application, 5527
Control element, 5528
Operation, 5528
simple alarm window
Application, 5527
Simple recipe view, 5574
Behavior, 5596
Layout, 5595
Menu command, 5596
Operation, 5596
Restrictions, 5577
Simple user view
Application, 5385
Display in Runtime, 5386
Layout, 5386
Number of lines, 5386
Simulate hardware, 6634
Simulate modules, 6634
Simulate software, 6634
SimulateSystemKey, 5680
SimulateTag, 5678
Simulating
Project with tag simulator, 5928
Simulating devices, 6634
Simulation
Runtime, 5926
SIN, 2719, 3044, 3283
Sine, 2719, 3044, 3283
Single instance
Correcting the call type, 4902, 4959
Definition, 2111
Example, 2111
SINT, 2248, 2341, 2398, 2472, 2497
SINT_TO_, 2398, 2497
Size
Editing multiple objects, 5312
Slice access, 2154
Slicing, 2212
Slot, 4073
Determining logical address with LGC_GADR, 4092
Determining logical address with LOG_GEO, 4088
Racks, 562
Select, 563
Slot rules, 563
DP slave, 1743
ET 200S, 1748
SMC, 2886, 3211, 3447
Smoothing, 1929, 1996
SMS text (TCON_Phone), 633
SMTP Client, 1344, 1620
SNC_RTCB, 3485
SNMP, 739, 1247, 1328, 1352, 1553, 1600, 1628
Groups, 1352, 1628
SNMP trap, 1352, 1627
SNMPv1, 1247, 1553
SNMPv2c, 1247, 1553
SNMPv3, 1247, 1553
Users, 1353, 1629
SNMPv1, 802
SNMPv3, 802
Softkey, 5359, 5363
global, 5361
SOFTNET Security Client
Configuring in the project, 837
Creating a configuration file, 837
Database, 837
Using, 836
SOFTNET Security Client, 720
Software
HMI device, 5949
Software controller
Configuring, 5995
Software requirements, 131
Software requirements for AS-AS remote link
Communication instruction "AS_DIAL", 6623
Software requirements on the programming device/ PC side
Communication instruction "PG_DIAL", 6621
Software requirements on the system side
Communication instruction "AS_MAIL", 6625
Communication instruction "PG_DIAL", 6621
Communication instruction "SMS_SEND", 6624
Software version, 1280, 1559
Software-controlled data flow control, 1000
Sorting
Alarm view, 5404
SP, 2635, 2961

Spanning Tree
Information, 1285, 1564
Rapid Spanning Tree, 1251, 1556
Special characters
in tags for Web server, 1061
Special features of CHM files, 435
Special tag (web server), 1065
Spectrum analyzer, 1678
Speedy Splitter, 528, 530, 537, 553
SPLIT, 3518
SQR, 2714, 3039, 3279
SQRT, 2716, 3041, 3280
Square, 2714, 3039, 3279
Square root, 2716, 3041, 3280
SR, 2584, 2908
SRT_DINT, 3830
SS, 2641, 2966
SSH server, 1326
SSL, 919
Stamping, 5310
Standard application, 1099
Standard device, 1806
Standard router, 870
Standard slave, 1761
Standard user, 736
Standby module
Substitute value, 1754
Standby redundancy, 1253
Standstill signal, 6220
Start address, 971, 1045
Start information, 2088
Start off-delay timer, 2938
Start on-delay timer, 2936
Start pulse timer, 2933
Start screen, 5261, 5907
Start the online and diagnostics view, 2031
Start the simulation, 6634
Start value
Tag, 5445, 5446
Starting
Runtime on a panel, 5926
Starting removal, 153
Starting the migration tool, 177
Starting the topology view, 704
StartLogging, 5686
STARTUP,
Function, 960
Organization blocks,
STARTUP activities, 961
Warm restart, 961
Startup language
determine:Startup language, 5522
Startup OB
Description, 1021
Start-up parameters, 962
Startup routine, 1021
Startup type, 982
Stateful packet inspection, 747
Static routes
IPv6 routes, 1477
Station
Copying, 570
Deleting, 575
Moving, 572
Multiple selection, 567
Read information with GetStationInfo, 3957
Rename, 573
Renaming, 716
Station name, (See device name)
Status, 1026
Module defective, 2044
Status interrupt, 1026
Status interrupt OB, 1026
Status of the online connection, 2072
Stepper motor, 6141
STL
Detailed comparison, 5099
STL program
Program status display, 5189
STL source
Generating blocks, 4744
STOP, 963
Stop forcing, 5249, 5250
StopLogging, 5686
StopRuntime, 5679
Storage location
File - TXT (Unicode), 5564
Storing
External graphic, 5315
Storing data
Storage location for retentive data, 1136
STP, 2836, 3160, 3405
Strg_TO_Chars, 3506
STRG_VAL, 3497
STRING, 94, 2168, 2270, 2379, 2460, 2485, 2486, 2528, 2530, 2544, 2566
Addressing, 2276
Comparing tags with S_COMP,
Declaration in the block interface, 4778
STRING_TO_, 2460, 2528, 2530, 2566
STRUCT
Addressing, 2285
Declaration in global data blocks, 4804
Declaration in the block interface, 4777

Declaring in a PLC data type, 4872
Structure, 2282
Structure, 1932, 1975, 1990
Call structure, 5132
of the cross-reference list, 5147
Of the dependency structure, 5137
Resources tab, 5143
STRUCT, 2282
Structured programming, 2086
Style
Changing, 5273
Creating a new type, 5818
Define, 5270
Deleting, 5272
Duplicate, 5272
Edit, 5272
HMI device, 5907
Setting, 5310
Style sheet
Copying, 5281
Creating, 5278
Creating a new type, 5818
Deleting, 5280
Editing, 5280
Use, 5282
SUB, 2699, 3024
Sub-cycle, 1963
Subnet, 603
Subnet mask, 601, 1238, 1535, 1769
Subnets
Configuration (IPv6), 1475
Connected Subnets (IPv6), 1475
Substitute value
Standby module, 1754
Subtract, 2699, 3024
Subtracting
Subtracting timers with T_SUB, 3461
Supply voltage, 1954
Supported file formats, 433
Supported objects for Multiuser Engineering
Multiuser Engineering, (Supported objects)
SWAP, 2770, 3096, 3322
Swap bytes, 2770
SWITCH, 2824, 3149
Type,
Switch OFF, 5778
Switch ON, 5778
Switching between basic mode and expanded mode in the force table, 5227
Switching between basic mode and expanded mode in the watch table, 5198
Switching operating mode, 2051
Switchover time, 1003
Switchport, 1797
Symbol
Alarm classes, 5509
For value comparison, 6002
Symbolic addressing, 2213
of a tag, 5426
Symbolic connection
Tag, 6681
Symbolic I/O field, 5416
Mode, 5416
Text list, 5336, 5417
Symbolic programming
Displaying absolute addresses, 4696
Symbolism, 2209
Symbols
In the assignment list, 5123
In the call structure, 5131
In the dependency structure, 5137
Symmetric encryption, 918
Sync domain, 1798, 2038
Sync domain properties, 2037
SYNC_PI, 3568
SYNC_PO, 3569
Synchronization
From DP slaves with DP_SYC_FR, 3709
Slave clocks with SNC_RTCB, 3485
Synchronization (user-defined web pages), 1057
Synchronization of variables, 5908
Synchronize, 5424, 5440
Synchronizing
Recipe data record, 5607
Synchronizing a PLC tag, 5442
Synchronizing user pages, 4575
Synchronizing WWW user pages, 4575
Synchronous errors
Masking with MSK_FLT, 3846
Querying the error register with READ_ERR, 3848
Unmasking with DMSK_FLT, 3847
Syntax errors
Basics, 4711
Finding errors, 4712
Syntax for AWP commands, 1061
Syslog Client, 1365, 1640
Syslog server, 825
System
Configuration, 1599
General information, 1329, 1601
System configuration, 1326
System alarm, 5485
Meaning, 5534, 5535, 5541, 5547

System block
System blocks folder, 4714
System constant, 4853
System diagnostic view
Detail view, 5419
Device view, 5418
Diagnostic buffer view, 5418
System diagnostics, 5788
Detail view, 5788
Device view, 5788
Diagnostic buffer view, 5789
System diagnostics view, 5788
System diagnostics view
Configuring, 5791
System event, 5485
Editing, 5502
Meaning, 5533, 5534, 5535, 5538, 5541, 5542, 5544, 5545, 5546, 5547, 5548, 5549, 5550, 5552, 5555, 5556
Parameters, 5533
System events, 832, 834, 5908
Configuration, 1617
Configuring, 5493
System function
Application, 5650
Function list, 5652
HTMLBrowserScrollLeft, 5755
In the function list, 5650
SafelyRemoveHardware, 5675
Usage, 5649
Use, 5650
System functions, 5649
AcknowledgeAlarm, 5722
ActivateCleanScreen, 5743
ActivatePreviousScreen, 5673
ActivateScreen, 5671
ActivateScreenByNumber, 5672
AdjustContrast, 5749
AlarmViewAcknowledgeAlarm, 5721
AlarmViewEditAlarm, 5720
AlarmViewShowOperatorNotes, 5721
CalibrateTouchScreen, 5750
ChangeConnection, 5716
ClearAlarmBuffer, 5718
ClearAlarmBufferProTool, 5719
ClearDataRecord, 5731
ClearDataRecordMemory, 5732
ClearLog, 5683
CloseAllLogs, 5685
DecreaseFocusedValue, 5680
DecreaseTag, 5697
EditAlarm, 5718
ExportDataRecords, 5723
GetDataRecordFromPLC, 5728
GetDataRecordTagsFromPLC, 5730
GetGroupNumber, 5689
GetPassword, 5690
GetUserName, 5689
GoToEnd, 5749
GoToHome, 5748
HTMLBrowerZoomIn, 5754
HTMLBrowserBack, 5758
HTMLBrowserForward, 5757
HTMLBrowserHome, 5757
HTMLBrowserPageDown, 5756
HTMLBrowserPageUp, 5757
HTMLBrowserRefresh, 5754
HTMLBrowserScrollDown, 5753
HTMLBrowserScrollRight, 5756
HTMLBrowserScrollUp, 5754
HTMLBrowserStop, 5753
HTMLBrowserZoomOut, 5755
ImportDataRecords, 5726
IncreaseFocusedValue, 5674
IncreaseTag, 5692
InvertBit, 5698
InvertBitInTag, 5700
InvertLinearScaling, 5693
LinearScaling, 5694
LoadDataRecord, 5728
Logoff, 5688
Logon, 5688
LogTag, 5682
OpenAllLogs, 5684
OpenControlPanelDialog, 5675
PageDown, 5748
PageUp, 5747
PressButton, 5676
RecipeViewBack, 5738
RecipeViewClearDataRecord, 5734
RecipeViewGetDataRecordFromPLC, 5734
RecipeViewMenu, 5735
RecipeViewNewDataRecord, 5733
RecipeViewOpen, 5735
RecipeViewRenameDataRecord, 5738
RecipeViewSaveAsDataRecord, 5737
RecipeViewSaveDataRecord, 5734
RecipeViewSetDataRecordToPLC, 5736
RecipeViewShowOperatorNotes, 5738
RecipeViewSynchronizeDataRecordWithTags, 5737
ReleaseButton, 5677
ResetBit, 5701
ResetBitInTag, 5703

SaveDataRecord, 5742
ScreenObjectCursorDown, 5745
ScreenObjectCursorLeft, 5745
ScreenObjectCursorRight, 5746
ScreenObjectCursorUp, 5744
ScreenObjectPageDown, 5747
ScreenObjectPageUp, 5746
SetBit, 5706
SetBitInTag, 5708
SetBitWhileKeyPressed, 5710
SetBrightness, 5751
SetConnectionMode, 5715
SetDataRecordTagsToPLC, 5740
SetDataRecordToPLC, 5739
SetDeviceMode, 5712
SetLanguage, 5714
SetRecipeTags, 5741
SetTag, 5695
ShowAlarmWindow, 5723
ShowLogonDialog, 5691
ShowOperatorNotes, 5681
SimulateSystemKey, 5680
SimulateTag, 5678
StartLogging, 5686
StopLogging, 5686
StopRuntime, 5679
TraceUserChange, 5691
TrendViewBackToBeginning, 5763
TrendViewCompress, 5760
TrendViewExtend, 5759
TrendViewRulerLeft, 5761
TrendViewRulerRight, 5760
TrendViewScrollBack, 5759
TrendViewScrollForward, 5758
TrendViewSetRulerMode, 5762
TrendViewStartStop, 5762
UpdateTag, 5674
System libraries, 440
System limits
HMI device, 5962, 5966
System log, 832, 834
System memory, 982, 1006
Diagnostics buffer, 971, 2070
Operand areas, 967
process image input/output, 969
System power supply, 1990
System requirement for WinCC Basic, 135
System requirements STEP 7 Basic, 131
System texts, 407
System Time
NTP Client, 1360, 1636
PTP Client, 1363
SNTP Client, 1358, 1634
System-defined role
Administrator, 736
diagnostics, 736
Remote access, 736
standard, 736
System-defined text lists
Editing, 419
Modifying texts, 420
System-relevant information
Module information, 2033
Vendor information, 2033

T

T_ADD, 3458
T_COMBINE, 3465
T_COMP, 3455
T_CONFIG, 4547
T_CONV, 3457
T_DIAG, 4538
T_DIFF, 3463
T_RESET, 4533
T_SUB, 3461
TAB key, 5939
Table:Designing a border, 5306
TADDR_Param, 4523
TADDR_RCV_IP, 4525
TADDR_SEND_QDN, 4525
Tag
absolute addressing, 5425
Acquisition cycle, 5446, 5453, 5468
Acquisition mode, 5446
Addressing, 5430
Changing the PLC, 5450
Comment, 5433
Configuration, 5437
Configuring several tags, 5438
Connection to PLC, 5430
Copy, 5436
Creating external tags, 5430
Creating internal tags, 5433
Data log, 5470, 5475
Data type, 5433
Deleting, 5436
Displaying or hiding tag information, 4915, 4972, 5036
Displaying values, 5480
Event, 5452
Exporting, 5831
External tag, 5423

Importing, 5833
in Runtime, 5446
Index tag, 5450, 5451
Indirect addressing, 5450, 5451
Insert, 6682
Internal tag, 5428
Length, 5433
Limit range, 5443
Limit value, 5444
Linear scaling, 5447, 5449
Logging, 5470, 5475
Multiplexing, 5450
Name, 5433
Negative, 2595, 2920
Object list, 5432
Output in alarm, 5500
Overlaying, 2155
Overview, 2140
PLC tag, 4837
PLC tags and DB tags, 2140
Positive, 2594, 2919
Read continuously, 5447
Reconnecting, 5442
Rename, 5436
Start value, 5445, 5446
symbolic addressing, 5426
Symbolic connection, 6681
Tolerance band, 5470, 5475
Update, 5447, 5453
Tag array, 5464
Tag counting with arrays, 5465
Tag data
Structure for the import, 5834, 5839
Tag declaration
Automatically filling in cells, 4790, 4819, 4861, 4877
Based on a PLC data type, 4806
Block interface, 4771
Declare STRING, 4778
Declaring ARRAY, 4775
Declaring PLC data type, 4778
Declaring STRUCT, 4777
Declaring tags, 4772, 4773, 4779
Deleting a tag, 4789, 4818, 4860, 4876
Importing and exporting tags, 4791, 4820
Inserting a table row, 4788, 4817, 4859, 4875
Inserting table rows at the end, 4789, 4817, 4876
Multi-instance, 4780
Overlaying tags, 4779
Purpose of variable declaration, 4764
Reserved key words, 2135
Retentivity, 4769
Showing and hiding columns, 4790, 4819, 4861, 4877
Sorting rows, 4860
Tag properties, 4784, 4786, 4811, 4815, 4816
Updating the block interface, 4791
Valid data types, 4767, 4769
Tag import
Tag data structure, 5834, 5839
Tag list
Indirect addressing, 5450, 5451
Tag name
Web server (PLC), 1061
Tag simulator, 5930
Tag table
Default, 5422
for HMI devices, 5422
user-defined, 5422
Tag value
Output, 5480
Tags
Basics, 5421
Editing, 5437
Migrating, 190
Migration, 204
Monitor all, 5214, 5240
Monitor now, 5240
Monitor once and now, 5215
TAN, 2722, 3047
Tangent, 2722, 3047, 3285
Task, 5781
Deactivate, 5783
Task card, 5256
Changing the pane mode, 259
Find instructions, 4697
Function, 257
Hardware catalog, 545
Online Tools,
Reducing automatically, 249
Tools,
Task switching, 5907
TCI (Tool Calling Interface), 1813
TCON, 4475, 4478
TCON_FDL, 633
TCON_IP_RFC, 631
TCON_IP_v4, 625
TCON_IP_V4_SEC, 627
TCON_Param, 623
TCON_PHONE for telecontrol connections, 632
TCON_QDN, 629
TCON_QDN_SEC, 630

TCP, 764
Characteristics, 621
Port numbers, 637
TDISCON, 4484
Team Engineering
Creating a master project, 6636
Example of the program structure in the master project, 6643
Integrating project copies into the master project once again, 6637
Introduction to shared commissioning, 6635
Notes on compatibility, 6637
Procedure for creating the master project, 6639
Procedure for integrating project copies into the master project, 6641
Procedure for manual synchronization of competing changes, 6641
Procedure for modifying central objects within the master project, 6642
Procedure for shared commissioning, 6640
Requirements for shared, parallel working on the project, 6639
Rules for downloading to the CPU, 6645
Rules for editing shared central objects, 6645
Rules for online functions, 6646
Rules for shared working on one CPU, 6644
Rules for the master project, 6642
Rules for working with organization blocks, 6645
Software and hardware requirements, 6638
Working with project copies, 6636
Technology object command table: Shortcut menu commands, 6248
Technology objects
PID_3Step, 6060
PID_Compact, 6023
PID_Temp, 6096
Telegram S7-1200 Motion Control, 6151
TeleService
Access to phone books, 6575
AS_DIAL, 6621
Callback variants, 6588
Communications instruction:"AS_DIAL", 6574
Communications instruction:"AS_MAIL", 6574
Communications instruction:"PG_DIAL", 6574
Communications instruction:"SMS_SEND", 6574
Communications instruction:"TM_MAIL", 6574
Communications instruction:"TMAIL_C", 6574
Connection establishment options:S7-1200 CPUs, 6619
Connection establishment options:S7-1500 CPUs, 6619
Connection establishment options:S7-300/400 CPUs, 6619
Defining dialing rules, 6581
Establish connection to AS, 6621
Export phone book, 6580
Functionality, 6574
Gateways, 6585
Import phone book, 6578
Inserting rows in the phone book, 6578
Meaning of the icons, 6577
Modem support, 6584
Modem types / media, 6585
Open phone book, 6577
Performance in telephone networks, 6585
Phone book, 6574
Phone book properties, 6575
Printing the phone book, 6581
Save phone book, 6578
Showing or hiding columns in the phone book, 6578
Structure of the phone book, 6576
Transferring e-mail with TM_MAIL, 4676
Working with the phone book, 6575
TeleService callback options, 6590
TeleService phone book, 6574
TeleService via WAN, 1218
Temperature coefficient, 1928, 1996
Temperature compensation, 1915, 1957
Template, 5359
Copy, 5266
Creating, 5265
Deleting, 5266
Global screen, 5265
Inserting, 5266
Move, 5266
Rename, 5266
Template,
Use in screen, 5267
Temporary connection, 1214
Tens complement, 2893, 3217, 3453
Terminal modules and electronic modules, 1912
Terminating the remote connection, 6606
Test options in the force table, 5224
Text
Acknowledgment group, 5908
Button, 5415
Text field, 5420
Text field, 5419
Size, 5420
Text field length
Asian languages, 5872

Text list
Application, 5329
Bit (0, 1), 5334
Bit number (0 - 31),
Creating, 5330
Exporting, 5837
Importing, 5839
Symbolic I/O field, 5336, 5417
Value/Range, 5333
Text lists
Bit selection, 5907
Introduction, 417
System-defined, 418
Text lists editor, 417
Use in recipe data records, 5570
User-defined, 418
Texts
Exporting all texts, 414
Exporting individual texts, 413
Exporting texts of a device, 413
Importing, 415
Migrating, 194
Textual cross-references
Display, 5848
TFTP
Load/save, 1337, 1614
THEN, 3385
Thermal resistor, 1957, 1968
Thermocouple, 1957, 1968
THIS, 2150
TIA Portal
Exiting, 229
Starting, 229
TiaDefaultProjectPath, 70
TiaUserSettingsPath, 70
TIME, 2262, 2360, 2434, 2480, 2521, 2540, 2560
Time accumulator, 2605, 2930, 2941, 3234
Time delay, 2845, 3169, 3413
Time delay interrupt, 1028
Time duration, 2620, 2945, 3238
Time error interrupt, 1032
Time error OB, 1032
Time expired, 5779
Time of day
Precision Time Protocol, 1363
Setting the time of day in the Online and Diagnostics view, 2052
SNTP (Simple Network Time Protocol), 1358, 1634
System Time, 1355, 1630
Time-of-day synchronization, 1358, 1634
Time stamp, 5492
Time stamping, 1916
Time synchronization, 796, 982
TIME_TCK, 3486
TIME_TO_, 2434, 2521, 2560
Time-delay interrupt
Canceling with CAN_DINT, 3832
Instructions, 3829
Querying with QRY_DINT, 3833
Starting with SRT_DINT, 3830
Time-delay interrupt OB, 1028
Time-of-day, 2264, 2265, 2266, 2267
Calculating local time with SET_TIMEZONE, 3476
Reading the system time of the CPU with TIME_TCK, 3486
Reading time-of-day and date of CPU with RD_SYS_T, 3469
Setting for CPU with WR_SYS_T, 3467
Time-of-day function
Basics, 990
Clock parameters, 991
Reading the time-of-day, 991
Setting the time-of-day, 991
Time-of-day format, 990
Time-of-day interrupt
Activating with ACT_TINT, 3819
Cancel, 1024
Canceling with CAN_TINT, 3818
Function, 1024
Querying with QRY_TINT, 3820
Rules, 1024
Set and activate, 1024
Setting with SET_TINT, 3813, 3816
Status query, 1024
Time-of-day interrupt OB
Parameter assignment, 1039
Timer, 2262, 2622, 2947
Recording, 2616
Timers
Adding with the instruction T_ADD, 3458
Combine date and time with T_COMBINE, 3465
Comparing time tags with T_COMP, 3455
Converting with T_CONV, 3457
Determining difference with T_DIFF, 3463
Subtracting with T_SUB, 3461
TLS, 919
TM_MAIL, 4676
TMAIL_C, 4453
TMAIL_C V4.0, 4457
TO, 3388
TO_PositioningAxis, 6182
TOD, 2264, 2372, 2451, 2482, 2524, 2543, 2563

TOD_TO_, 2451, 2524, 2563
TOF, 2602, 2614, 2927, 2938, 3231
Toggle
  Between Runtime languages, 5866
  Key, 5941
Tolerance band
  Tags, 5470, 5475
TON, 2599, 2611, 2924, 2936, 3228
TONR, 2605, 2616, 2930, 2941, 3234
Tool Calling Interface (TCI), 1813
Toolbar
  Order, 5286, 5295
Tools, 5283
Tooltip
  Show limits, 5907
Topology discovery, 1798
Topology editor, 603
Topology overview, 537
Topology view
  Adding device, 557
  Adopt devices identified online, 719
  Adopt port interconnections identified online, 718
  Advanced offline/online comparison, 711
  Compare offline/online, 709
  Configured topology, 705, 706
  Diagnostics status in the graphic view, 707
  Diagnostics status in the table view, 708
  Differences compared with the network view, 703
  Functions, 702
  Hardware and network editor, 537
  Interconnecting ports, 712
TP, 2597, 2608, 2922, 2933, 3225
Trace, 6513, (Trigger time)
  Align overlay measurements, 6556
  Analyze an ongoing recording, 6546
  Apply overlay measurement, 6543
  Bit track, 6525
  Configure project navigator into groups, 6544
  Copy measurement, 78
  CPU load, 6559
  Creating a trace configuration, 6542
  Curve diagram, 6525, 6550
  Data storage, 6516
  Deleting measurements, 78
  Deleting measurements in the device, 78
  Deleting measurements on the memory card, 78
  Displaying a diagram, 6543
  Displaying a trace configuration, 6543
  Installed measurements (memory card), 6516, 6522
  Installed trace, 6516
  Lifetime of the values, 6558
  Measurement, 6516, 6547, 6548
  Measurement cursor, 6550
  Measurement cursor pane, 6534
  Memory usage, 78
  Mouse wheel, 6526
  Overlay measurement, 6516, 6523, 6531, 6554
  Pre-trigger, 6564
  Printing, 6556
  Project navigator, 6519
  Quantity structure, 6559
  Quick start, 6537
  Recordable tags, 6557
  Recording, 6515, 6545, 6546
  Recording conditions, 6561, 6566
  Recording cycle, 6539, 6570
  Recording duration, 6570
  Recording levels, 6558
  Reduction, 6562
  Sampling, 6539
  Save installed measurements, 6565, 6572
  Saving the trace configuration, 6550
  Signal group, 6553
  Signal table, 6529, 6552
  Signals, 6560, 6570
  Snapshots pane, 6535
  Status, 6520
  Supported devices, 6513
  Trace configuration, 6515, 6516, 6545, 6548, 6549, 6569
  Trigger, 6571
  Trigger mode, 6563
  Trigger tag, 6563, 6566
  User interface, 6517, 6536, 6559
Trace function, 6513
Trace S7-1200/1500, 6557
TraceUserChange, 5691
Transfer card, 964
Transfer card; see Transfer card, 964
Translate
  Editor, 5856
Translating texts, 407
Transmission medium/duplex, 1793
Transmission mode, 1196
Transparency
  In graphic, 5313
transparent color, 5393
  Display on panels, 5393
Transport Layer Security, 919
TRCV, 4493, 4498
TRCV_C, 4432, 4438

Trend view, 5397
Button, 5398
Configuring for values from the PLC, 5480
TrendViewBackToBeginning, 5763
TrendViewCompress, 5760
TrendViewExtend, 5759
TrendViewRulerLeft, 5761
TrendViewRulerRight, 5760
TrendViewScrollBack, 5759
TrendViewScrollForward, 5758
TrendViewSetRulerMode, 5762
TrendViewStartStop, 5762
Trigger tag, 5493
Trigger tag - resetting, 1195
Triggers, 5783
Event trigger, 5783, 5784, 5787
TRUE, 2594, 2595, 2919, 2920, 3219, 3220
TRUNC, 2809, 3134, 3369
Trust Mode, 1391
TS Adapter IE Advanced
Brief description, 6615
Connection types, 6615
Parameter assignment, 6617
Parameter assignment options, 6616
TS Adapter MPI
Brief description, 6592
Default parameter assignment, 6593
Direct connection, 6593
Establishing a modem connection, 6594
Establishing the direct connection, 6593
Exporting adapter parameters, 6597
Importing adapter parameters, 6598
Modem connection, 6594
Parameter assignment, 6595, 6596
Parameter assignment options, 6595
Principle of operation, 6592
Restoring the default parameter assignments, 6597
Setting up access protection, 6589
TS Adapter IE
Brief description, 6599
Default parameter assignment, 6600
Parameter assignment, 6603, 6617
Principle of operation, 6600
TS Adapter IE Basic
Connection to the GSM network, 6602
Connection to the telephone network, 6601
Connection to the telephone network through an external modem, 6602
Connection types, 6600
TSAP
ASCII code table, 641
Structure, 639, 664
TSAP assignment
Examples, 641
TSEND, 4486, 4490
TSEND_C, 4419, 4423
Tuning, 6264
TURCV, 4520
TUSEND, 4516
Twos complement, 2704, 3029
Type
Creating an instance, 498
Displaying the properties of a type, 480
Displaying the properties of a version, 480
Library, 5797
Type of CPU startup, 1132
typed, 5010
TypeOf, 3269
TypeOfElements, 3270
Types
Copying instances, 486
Creating an instance, 486
Pasting instances, 486
Using types, 486
Types of DP slaves, 1743

U

UART data transmission, 997
UBLKMOV, 2797, 3122, 3358
UDINT, 2253, 2349, 2416, 2477, 2512
UDINT_TO_, 2416, 2512
UDP, 764, 771, 796
Characteristics, 622
Port numbers, 637
UDT, 2211, 2229
UFILL_BLK, 2767, 3092, 3319
UINT, 2251, 2345, 2409, 2475, 2506
UINT_TO_, 2409, 2506
ULINT, 2255, 2353, 2423
ULINT_TO_, 2423
UMOVE_BLK, 2761, 3086, 3313
Underflow, 1930, 1954
Undoing actions
Basics of undoing, 316
Undoing last action, 318
Undoing multiple actions, 318
Uninstalling license keys, 130
Uniqueness
of object names, 182

Universal Serial Interface Protocol, (See USS protocol)
Universal symbolism, 2209
Unlock
  User, 5634
Unplugged module, 535
UNSCALE, 2817, 3142, 3383
Unsolicited, 1196
Unspecified CPU, 558
UNTIL, 3392
UPDAT_PI, 3562
UPDAT_PO, 3565
Update, 1027
  Tag, 5447, 5453
Update cycle, 5468
Update interrupt, 1027
Update interrupt OB, 1027
Update time, 1789
UpdateTag, 5674
Updating
  Device proxy, 6660, 6665
  IPE data, 6660, 6665
  Operating system, 5955
Updating a library, 505, 5807
Updating the device version, 5953
Updating the firmware, 5953
Updating the operating system, 5953
Updating types to the latest version, 505, 5807
Upgrade
  Global library, 5895
  Project, 5894
  Project version, 5894
Upgrading a global library, 461
Upload, 1816
Uploading
  from a device, 354
URCV, 4401
USB card readers, 68
Use, 5566
  Of recipes, 5566
  Recipe view, 5410
  Screen navigation function keys, 5372
  Style sheet, 5282
  System function, 5650
  User view, 5387
Use global assignment
  Function key, 5264
Useful information on configuring the TS Adapter MPI, 6595
USEND, 4399
User
  Assigning roles, 737
  Changing, 5633
  Creating roles, 736
  Deleting in runtime, 5633
  Logon, 5634
  Setting up, 734
  Unlock, 5634
User administration, 5612, 5908
  Migration, 197
  Object with access protection, 5637
  Runtime settings, 5262, 5617, 5634
  Setting up, 5640
User change, 5771, 5786
User data
  Area, 971
  Backup, 5614
  Restoring, 5614
User group
  Administer authorizations, 5625
  Assigning, 5646
  Assigning users, 5624
  Change displayed name, 5626
  Changing in runtime, 5633
  Changing the name, 5626
  Creating, 5644
  Deleting, 5626
  Managing, 5625
  Unauthorized, 5636
User Groups, 1498, 1703
User interface
  "Reference projects" palette, 259
  Details view, 260
  Inspector window, 255
  Maximizing the work area, 249
  Minimizing the work area, 249
  Overview window, 262
  Portal view, 239
  Project tree, 243
  Project view, 240
  Task card, 257
  Views, 238
  Work area, 247
User interface language, 5851
  Selecting, 5853
User libraries, 440
User name, 735
User pages, (See user-defined web pages)
User program
  Finding errors, 4712
  Function, 2085
  Testing, 5175
User rights
  Hierarchy levels, 5908

User texts, 407
User view, 5627
Column width, 5388
Complex user view, 5629
Configuring, 5630
Layout, 5387
Moving columns, 5388
Number of lines, 5387
Simple user view, 5628
Use, 5387
User-defined documentation, 426
Enabling the call log, 429, 430
Specifying settings, 429, 430
Specifying the central file directory, 429, 430
User-defined functions
In function lists, 5652
User-defined roles, 736
User-defined text lists
Creating, 418
Editing, 419
Editing value ranges and texts, 419
User-defined web pages, 1056, 1059, 1069, 1070, 1071, 1072, 1073
Users
Assigning a user group, 5624
Changing the name, 5624
Create, 5622
Creating, 5646
Creating in runtime, 5630
Deleting, 5625
Logging logons, 5639
Managing, 5624, 5632
Updating after change of user, 5786
Using a TS Adapter for TeleService, 6583, 6606
Using frames and cover pages from the library, 387
Using tags, 5389
Using the keyboard
Editing texts, 4688
FBD, 4688
GRAPH, 4688
LAD, 4688
Program editor, 4688
SCL, 4688
STL, 4688
Using the on-screen keyboard, 280
Using UDTs, 2229
USINT, 2249, 2342, 2402, 2473, 2500
USINT_TO_, 2402, 2500
USS communication
Controlling a transmission to a drive with USS_PORT, 4609
Data exchange with drives via USS_DRIVE, 4610
Modifying parameters in the drive with USS_WPM, 4614
Reading parameters from the drive with USS_RPM, 4613
USS protocol
Instructions, 4605
USS status codes, 4616
USS_DRIVE, 4610
USS_PORT, 4609
USS_RPM, 4613
USS_WPM, 4614

V

VAL_STRG, 3501
Value, 2707, 3033
Value assignment, 5002
Value change, 5779
Value/Range
Text list, 5333
Values
Comparing, 6002
Variable index, 2215
VARIANT, 2308, 2311, 2312, 2316, 2320, 2684, 2686, 2688, 2690, 2783, 2784, 2786, 3009, 3011, 3013, 3015, 3108, 3109, 3111, 3269, 3270, 3349, 3350, 3352, 3376, 3378
VARIANT_TO_DB_ANY, 3376
VariantGet, 2783, 3108, 3349
VariantPut, 2784, 3109, 3350
Vendor ID, 1280, 1559
Version overview, 1043
Versioning of types
Released version, 478
Version in test, 478
Versions in WinCC, 5891
Vertical movement
Animation, 5353
Vertical radius, 5392
View options
For the assignment list, 5125
Setting the call structure, 5133
Setting the dependency structure, 5139
Virtual environment, 70
Virtual Machine (VM)
Supported virtualization platforms, 133, 137
Virus scanners, 138
Visualization, 1144
VLAN, 1243, 1540
Day, 1399, 1683
Port VID, 1399, 1684
Priority, 1399, 1683

VLAN ID, 1246, 1540
VLAN tag, 1244, 1539
VLAN operation, 807
VLAN tagging, 807
VOID, 2325
VPN connection
Deleting CA certificates, 6612
Establishing, 6613
Installing CA certificate, 6609
Terminating, 6614
VPN connection is not established
Check and remedy, 6633
VPN group, 803
VRRP
Address overview, 1455
Backup router, 1262
Configuration, 1453
Interface Tracking, 1456, 1461, 1487
Master router, 1262
Router, 1452
Virtual router, 1262
VRRP addresses overview (IPv4), 1460
VRRP router, 1262
VRRP routers (IPv4), 1457, 1483
VRRP Statistics, 1287
VRRPv3 address configuration (IPv6), 1487
VRRPv3 Addresses Configuration (IPv4), 1460
VRRPv3 addresses overview (IPv6), 1486
VRRPv3 Configuration (IPv4), 1458
VRRPv3 Configuration (IPv6), 1485
VRRPv3
VRRPv3 Statistics, 1289

W

WAIT, 2845, 3169, 3413
Wake on LAN, 3741
WAN IP address
Specifying, 822
Warm restart, 961
Watch table
Basic mode, 5197
Changed display formats, 85
Copying, 5201
Creating, 5200
Example of filling out a watch table, 5202
Expanded mode, 5197
Layout, 5197
Loading data blocks during an active control job, 84
Meaning of the columns, 5197
Meaning of the icons, 5199
Monitoring and modifying modes, 5212
Multiple access to the same CPU, 84
Opening, 5201
Overview of the display formats, 5206
Overview of the test options, 5196
Permitted operands, 5203
Permitted operands for modify values, 5204
Possible applications, 5196
Saving, 5202
Switching between basic mode and expanded mode, 5198
Syntax check, 5202
Testing wiring, 5196
Watchdog time, 1791
WChar, 5463
WCHAR_TO_, 2458
WDS, 1665
Web application, 1056
Web Control DB, 1074
Web data blocks, 1071, 1072
Web pages in browser, 1079
Web server, 1050, 1052, 1056, 1059, 1071, 1072, (See Web server)
Enable, 1053
Enumerations, 1069
Fragment, 1070
HTTPS, 1054
Rules PLC tag names, 1061
User-defined web pages, 1056
When dialog is closed, 5771
When dialog is opened, 5771
WHILE, 3391
WinCC flexible project
Migrate, 184, 202
WinCC RT), 1098
WinCC version
Compatibility, 5892
Window layouts
Changing order, 268
Deleting window layouts, 268
Windows, 131
Wire break, 1929, 1995
Wireless LAN/PB link, 1764
Wiring tests, 5196
Wizard
Device wizard, 5900
WLAN
WLAN authentication log, 1562
WLAN statistics
Fault, 1591
Received data frames, 1595
Sent data frames, 1595

WORD, 2245, 2336, 2387, 2470, 2490, 2536, 2548
WORD_TO_, 2387, 2490, 2548
WORD_TO_BLOCK_DB, 2567
Work area
  Embedding floating elements, 251
  Floating elements, 251
  Function, 247
  Maximizing, 249
  Maximizing elements, 254
  Minimizing, 249
  Minimizing elements, 254
  Saving a layout of editors and tables, 268
  Splitting, 250
  Switching between elements, 254
  Using grouped elements, 252
Work memory, 967, 5141
Working step
  to create screens, 5259
Working with local sessions, 6789
Working with Multiuser Engineering, 6687, 6689
WR_DPARM, 3793
WR_LOC_T, 3473
WR_REC, 3699
WR_SYS_T, 3467
WRIT_DBL, 4054
Write field, 2792, 3117
Write memory address, 3335
Write memory area, 3339
Write memory bit, 3337
Write protection, 4756
WRITE_BIG, 3347
WRITE_LITTLE, 3343
WriteToArrayDB, 2774, 3100, 3325
WriteToArrayDBL, 2780, 3105, 3329
Writing
  A data record with WR_REC, 3699
Writing data
  in remote CPU with PUT, 4393
  To DP standard slaves/PROFINET IO devices with DPWR_DAT, 3702
Writing tags, 1064
WRREC, 3596
WSTRING, 2168, 2273, 2380, 2463
  Addressing, 2276
WSTRING_TO_, 2463
WWW (instruction), 1073

X

X.509, 916
XDB configuration, 1111, 1112
xlsx file, 5825, 5833, 5837, 5838
  Importing tags, 5832
  Preparing for the import, 5832
XOR, 2850, 2898, 2899, 3175, 5000, 5001

Z

Zoom
  Adjusting the zoom setting, 528, 531, 537
  Keyboard operation, 554
Zoom in/out, (See Zoom)
Zooming
  Screen, 5258